TPC Benchmark™ H Full Disclosure Report

for

IBM @server xSeries 250

using

Microsoft SQL Server 2000 Enterprise Edition

Submitted for Review

March 23, 2001

First Edition - March 2001

THE INFORMATION CONTAINED IN THIS DOCUMENT IS DISTRIBUTED ON AN AS IS BASIS WITHOUT ANY WARRANTY EITHER EXPRESSED OR IMPLIED. The use of this information or the implementation of any of these techniques is the customer's responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item has been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used; any functionally equivalent program may be used.

This publication was produced in the United States. IBM may not offer the products, services, or features discussed in this document in other countries, and the information is subject to change without notice. Consult your local IBM representative for information on products and services available in your area.

Trademarks

IBM, the IBM e-business logo, xSeries and Netfinity are trademarks or registered trademarks of International Business Machines Corporation.

The following terms used in this publication are trademarks of other companies as follows: TPC Benchmark, TPC-H, QppH QthH and QphH are trademarks of Transaction Processing Performance Council; Intel, Pentium and Xeon are registered trademarks of Intel Corporation; Microsoft and Windows are trademarks or registered trademarks of Microsoft Corporation. Other company, product, or service names, which may be denoted by two asterisks (**), may be trademarks or service marks of others.

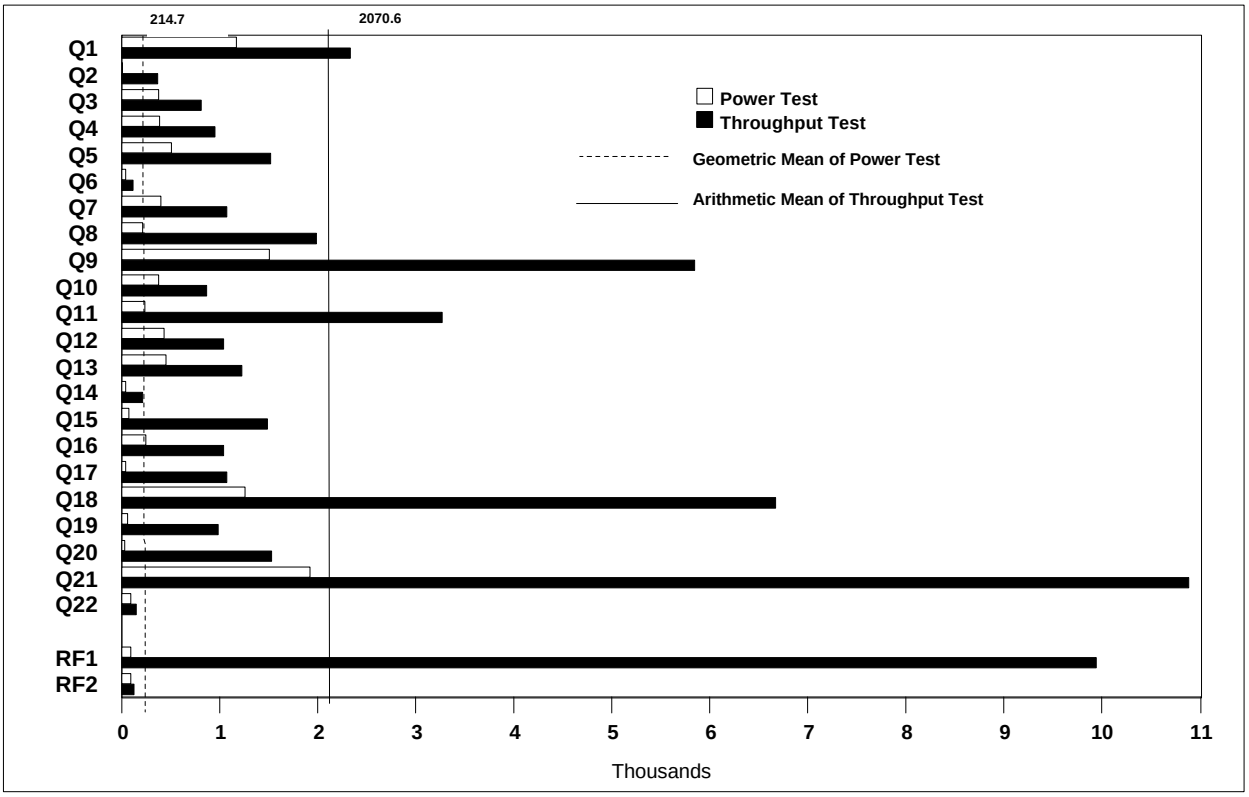
Notes

[1] MHz only measures microprocessor internal clock speed, not application performance. Many factors affect application performance.
[2] When referring to hard disk capacity, one GB equals one billion bytes. Total user-accessible capacity may be less.

IBM	IBM @server xSeries 250 with Microsoft SQL Server 2000	TPC-H Rev 1.3.0 Report Date: Mar. 23, 2001

Total System Cost	Composite Query-per-Hour Metric	Price/Performance
$159,482	**1147.9 QphH @ 100GB**	**$139 per QphH@100GB**

Database Size	Database Manager	Operating System	Other	Availability Date
100GB	**Microsoft SQL Server 2000 Enterprise Edition**	**Microsoft Windows 2000 Advanced Server**	**Visual Studio Professional 6.0**	**April 13, 2001**



Database Load Time: 07:29:59	**Load Included Backup: Y**	**Total Data Storage / Database Size: 10.25**
RAID (Base Tables Only): N	**RAID (Base Tables and Auxiliary Data Structures): N**	**RAID (All): N**

Configuration		
Processors	**4**	**Intel Pentium III Xeon 900MHz / 2MB L2 Cache**
Memory	**4**	**1GB SDRAM ECC RDIMM II**
Disk Controllers	**5**	**Mylex eXtremeRAID 2000 Adapter**
Disk Drives	**121**	**9.1GB 15K Ultra160 SCSI Drive**
Total Disk Storage		**1024.87GB**

IBM @server xSeries 250 Microsoft SQL Server 2000

TPC-H Revision 1.3.0

Report Date: March 23, 2001

Description	Part Number	Source	Unit Price	Qty	Ext. Price	5-Yr. Maint.
Server Hardware						
xSeries 250 / 900MHz/2MB L2 Cache Pentium III Xeon	8665-8RY	1	9,879	1	9,879	10,295
900MHz / 2MB L2 Cache Processor	19K4635	1	6,189	3	18,567	0
1GB SDRAM ECC RDIMM II	33L3119	1	1,689	4	6,756	0
Mylex eXtremeRAID 2000 Adapter*	E2000-4-32NB	3	1,872	7	13,104	0
Netfinity 4.2M Ultra2 SCSI Cable	03K9311	1	105	18	1,890	0
250W Hot-Swap Redundant Power Supply	37L3760	1	224	1	224	0
E54 15” (13.8” Viewable) Color Monitor	6331N2N	1	159	1	159	270
Netfinity Rack	9306900	1	1,725	1	1,725	1,308
Side Panel Kit	94G6669	1	195	1	195	0
				Subtotal	$52,499	$11,873
Server Storage						
Netfinity EXP300 Rack Storage Expansion Enclosure	35311RU	1	3,179	9	28,611	7,074
Netfinity 9.1GB 15K Ultra160 SCSI Drive	19K0655	1	405	121	49,005	0
				Subtotal	$77,616	$7,074
Server Software						
Microsoft SQL Server 2000 Enterprise Edition	810-00652	2	5,549	1	5,549	10,475
Microsoft SQL Server 2000 Client Licenses	359-00532	2	7,300	1	7,300	InclAbove
Microsoft Windows 2000 Advanced Server	C10-00475	2	2,399	1	2,399	InclAbove
Visual Studio Professional 6.0 Win32	659-00390	2	1,079	1	1,079	InclAbove
				Subtotal	$16,327	$10,475
				Total	$146,442	$29,422
Large volume discount of 14% on IBM hardware; prices vary if purchased separately.					($16,382)	

Notes: Standard 3-Year and extended warranties upgraded to 7x24, 4-hour response time coverage. *10% or minimum 2 spares are added in place of on-site service; product has a 5-year return-to-vendor warranty.
Pricing: 1 - IBM List Price; 2 - Microsoft Corp.; 3 - Mylex
Audited by Brad Askins of InfoSizing, Inc.

5-Year Cost of Ownership:	**$159,482**
QphH @ 100GB:	**1,147.9**
$ / QphH @ 100GB:	**$139**

Prices used in TPC benchmarks reflect the actual prices a customer would pay for a one-time purchase of the stated components. Individually negotiated discounts are not permitted. Special prices based on assumptions about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed components. For complete details, see the pricing sections of the TPC benchmark specification. If you find that stated prices are not available according to these terms, please inform the TPC at pricing@tpc.org. Thank you.

Measurement Results:

Database Scale Factor	100
Total Data Storage/Database Size	10.25
Start of Database Load	08:13:11
End of Database Load	15:43:10
Database Load Time	07:29:59
Query Streams for Throughput Test	5
TPC-H Power	1676.5
TPC-H Throughput	785.9
TPC-H Composite Query-per-Hour (QphH@100GB)	1,147.9
Total System Price over 5 Years	$159,482
TPC-H Price/Performance Metric ($/QphH@100GB)	$139

Measurement Interval:

Measurement Interval in Throughput Test (Ts) = 50,385 seconds

Duration of Stream Execution:

	Seed	Query Start Date/Time Query End Date/Time	RF1 Start Date/Time RF1 End Date/Time	RF2 Start Date/Time RF2 End Date/Time	Duration
Stream 00	311154310	03/12/01 11:16:45	03/12/01 11:15:00	03/12/01 14:04:32	02:51:16
		03/12/01 14:04:31	03/12/01 11:16:40	03/12/01 14:06:16	
Stream 01	311154311	03/12/01 14:06:18	03/12/01 14:06:18	03/13/01 03:46:05	13:37:54
		03/13/01 03:44:12	03/13/01 03:46:01	03/13/01 03:48:18	
Stream 02	311154312	03/12/01 14:06:18	03/13/01 03:48:19	03/13/01 03:50:36	12:35:31
		03/13/01 02:41:49	03/13/01 03:50:34	03/13/01 03:52:52	
Stream 03	311154313	03/12/01 14:06:18	03/13/01 03:52:54	03/13/01 03:55:11	13:00:38
		03/13/01 03:06:56	03/13/01 03:55:10	03/13/01 03:57:25	
Stream 04	311154314	03/12/01 14:06:18	03/13/01 03:57:27	03/13/01 03:59:46	12:38:48
		03/13/01 02:45:07	03/13/01 03:59:45	03/13/01 04:01:52	
Stream 05	311154315	03/12/01 14:06:19	03/13/01 04:01:54	03/13/01 04:04:00	11:23:19
		03/13/01 01:29:38	03/13/01 04:03:59	03/13/01 04:06:03	

TPC-H Timing Intervals (in seconds):

Query	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
Stream 00	1,182.2	21.2	382.5	395.0	519.8	50.3	404.9	221.8	1,518.9	382.7	248.0	434.9
Stream 01	1,165.3	300.4	932.7	399.7	1,423.5	153.8	1,211.3	5,226.5	4,720.7	815.2	1,804.1	936.6
Stream 02	2,167.7	373.6	1,021.4	942.7	1,593.1	170.1	1,258.9	557.0	6,980.2	915.7	3,178.1	1,163.3
Stream 03	3,014.4	295.9	502.2	1,043.2	1,519.8	97.9	1,025.2	3,183.0	6,863.8	910.9	4,650.8	551.9
Stream 04	2,775.6	455.3	739.5	1,437.5	1,698.8	105.2	960.1	579.3	7,062.4	943.3	4,134.6	992.8
Stream 05	2,557.5	459.1	880.6	968.5	1,392.1	105.0	928.6	399.9	3,610.1	789.3	2,582.5	1,579.8
Minimum	1,165.3	295.9	502.2	399.7	1,392.1	97.9	960.1	399.9	3,610.1	789.3	1,804.1	551.9
Average	2,336.1	376.9	815.3	958.3	1,525.5	126.4	1,076.8	1,989.1	5,847.4	874.9	3,270.0	1,044.9
Maximum	3,014.4	459.1	1,021.4	1,437.5	1,698.8	170.1	1,258.9	5,226.5	7,062.4	943.3	4,650.8	1,579.8

Stream ID	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	RF1	RF2
Stream 00	460.7	51.6	81.6	260.5	45.7	1,266.2	74.1	37.1	1,923.1	103.1	100.1	103.8
Stream 01	2,090.5	195.8	1,548.7	792.2	829.6	7,405.0	514.0	6,070.6	10,434.8	103.4	49,183.2	132.7
Stream 02	1,074.4	254.2	1,518.6	2,001.5	1,046.1	5,875.8	2,185.6	253.9	10,240.9	198.2	135.4	136.0
Stream 03	744.7	253.2	1,579.8	322.3	1,378.1	7,383.9	398.4	354.7	10,594.4	169.5	136.0	134.5
Stream 04	1,306.2	323.7	1,529.0	610.1	294.0	5,970.2	934.4	908.0	11,564.4	203.6	137.9	126.0
Stream 05	922.0	109.9	1,263.4	1,500.6	1,504.0	6,729.9	915.0	76.7	11,587.9	136.9	124.6	123.0
Minimum	744.7	109.9	1,263.4	322.3	294.0	5,875.8	398.4	76.7	10,240.9	103.4	124.6	123.0
Average	1,227.6	227.4	1,487.9	1,045.3	1,082.4	6,673.0	989.5	1,532.8	10,884.5	162.3	9,943.4	130.4
Maximum	2,090.5	323.7	1,579.8	2,001.5	1,504.0	7,405.0	2,185.6	6,070.6	11,587.9	203.6	49,183.2	136.0

Table of Contents

Preface

The TPC Benchmark H is a decision support benchmark. It consists of a suite of business-oriented ad hoc queries and concurrent data modifications. The queries and the data populating the database have been chosen to have broad industrywide relevance while maintaining a sufficient degree of ease of implementation. This benchmark illustrates decision support systems that:

- Examine large volumes of data;
- Execute queries with a high degree of complexity;
- Give answers to critical business questions.

TPC-H evaluates the performance of various decision support systems by the execution of set of queries against a standard database under controlled conditions. The TPC-H queries:

- Give answers to real-world business questions;
- Simulate generated ad-hoc queries (e.g., via a point-and-click GUI interface);
- Are far more complex than most OLTP transactions;
- Include a rich breadth of operators and selectivity constraints;
- Generate intensive activity on the part of the database server component of the system under test;
- Are executed against a database complying with specific population and scaling requirements;
- Are implemented with constraints derived from staying closely synchronized with an on-line production database.

The TPC-H operations are modeled as follows:

- The database is continuously available 24 hours a day, 7 days a week, for ad-hoc queries from multiple end users and data modifications against all tables, exeat possibly during infrequent (e.g., once a month) maintenance sessions.
- The TPC-H database tracks, possibly with some delay, the state of the OLTP database through ongoing refresh functions, which batch together a number of modifications impacting some part of the decision support database.
- Due to the worldwide nature of the business data stored in the TPC-H database, the queries and the refresh functions may be executed against the database at any time, especially in relation to each other. In addition, this mix of queries and refresh functions is subject to specific ACIDity requirements, since queries and refresh functions may execute concurrently.
- To achieve the optimal compromise between performance and operational requirements, the database administrator can set, once and for all, the locking levels and the concurrent scheduling rules for queries and refresh functions.

The minimum database required to run the benchmark holds business data from 10,000 suppliers. It contains almost 10 million rows representing a raw storage capacity of about 1 gigabyte. Compliant benchmark implementations may also use one of the larger permissible database populations (e.g., 100 gigabytes), as defined in Clause 4.1.3).

The performance metrics reported by TPC-H is called the TPC-H Composite Query-per-Hour Performance Metric (QphH@Size), and reflects multiple aspects of the capability of the system to process queries. These aspects include the selected database size against which the queries are executed, the query processing power when queries are submitted by a single stream , and the query throughput when queries are submitted by multiple concurrent users. The TPC-H Price/Performance metric is expressed as $/QphH@Size. To be compliant with the TPC-H standard, all references to TPC-H results for a given configuration must include all required reporting components (see Clause 5.4.6). The TPC believes that comparisons of TPC-H results measured against different database sizes are misleading and discourages such comparisons.

The TPC-H database must be implemented using a commercially available database management system (DBMS), and the queries executed via an interface using dynamic SQL. The specification provides for variants of SQL, as implementers are not required to have implemented a specific SQL standard in full.

Benchmarks results are highly dependent upon workload, specific application requirements, and systems design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, TPC-H

should not be used as a substitute for specific customer application benchmarking when critical capacity planning and/or product evaluation decisions are contemplated.

1 General Items

1.1 Benchmark Sponsor

A statement identifying the benchmark sponsor(s) and other participating companies must be provided.

IBM Corporation sponsored this TPC-H benchmark.

1.2 Parameter Settings

Settings must be provided for all customer-tunable parameters and options that have been changed from the defaults found in actual products, including but not limited to:

- *Database tuning options*
- *Optimizer/Query execution options*
- *Query Processing tool/language configuration parameters*
- *Recovery/commit options*
- *Consistency/locking options*
- *Operating system and configuration parameters*
- *Configuration parameters and options for any other software component incorporated into the pricing structure*
- *Compiler optimization options.*

Appendix A contains the database and operating system parameters.

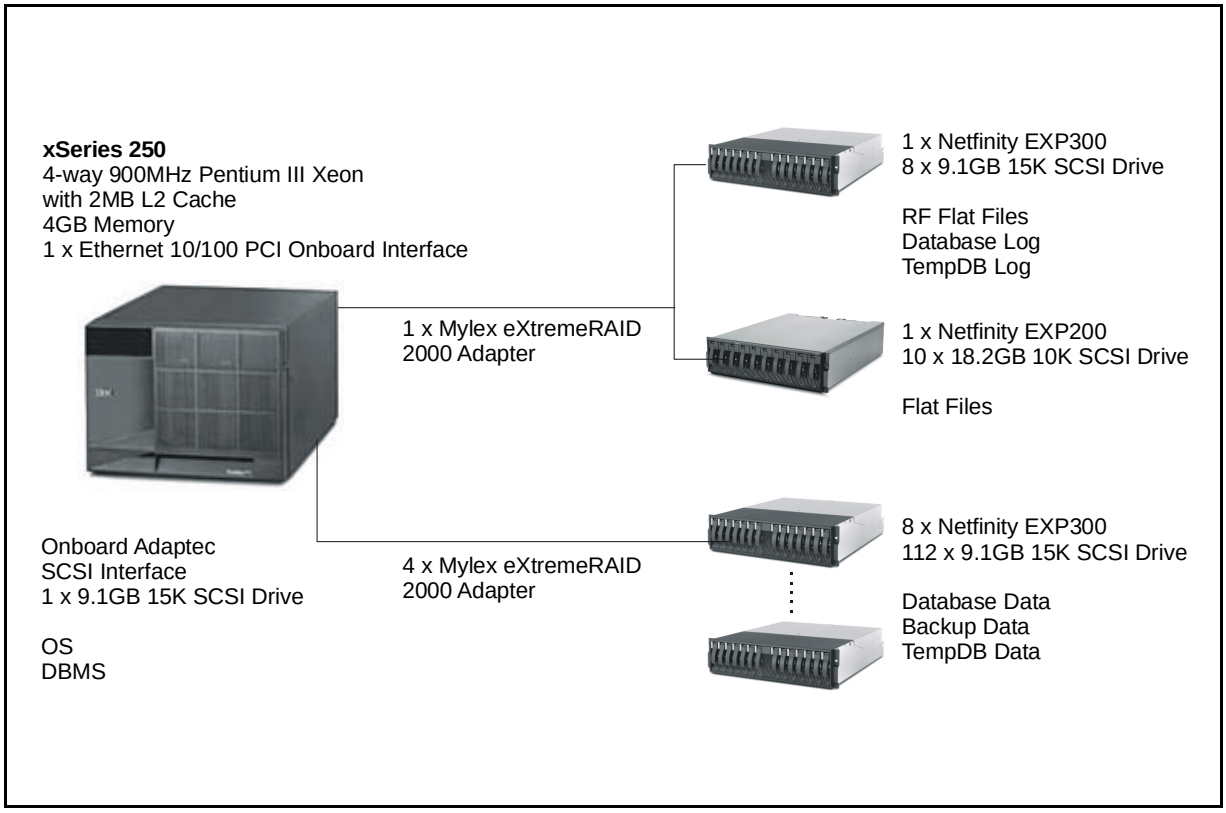
1.3 Configuration Diagrams

Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences. This includes, but is not limited to:

- *Number and type of processors*
- *Size of allocated memory and any specific mapping/partitioning of memory unique to the test and type of disk units (and controllers, if applicable)*
- *Number and type of disk units (and controllers, if applicable)*
- *Number of channels or bus connections to disk units, including their protocol type*
- *Number of LAN (e.g., Ethernet) connections, including routers, workstations, terminals, etc., that were physically used in the test or are incorporated into the pricing structure*
- *Type and run-time execution location of software components (e.g., DBMS, query processing tools/languages, middleware components, software drivers, etc.).*

The configuration diagram for the tested and priced system is provided on the following page.

1.3.1 Measured Configuration



The measured configuration for the xSeries 250 contained:

- Four Intel Pentium III Xeon 900MHz processors, each with 2MB of L2 cache
- 4GB of memory
- One 10/100 Ethernet PCI onboard interface
- One onboard Adaptec SCSI interface
- Five Mylex eXtremeRAID 2000 Adapters
- One hundred twenty-one (121) 9.1GB 15K Ultra160 SCSI Disk Drives
- Nine (9) EXP300 Storage Expansion Enclosures
- One (1) EXP200 Storage Expansion Enclosure (held flat files and was not priced)
- Ten (10) 18.2GB 10K Wide Ultra SCSI Disk Drives (held flat files and were not priced)

1.3.2 Priced Configuration

The priced and measured configurations were identical with one exception:

- Ten (10) 18.2GB Wide Ultra SCSI Disk Drives and one (1) EXP200 Storage Expansion Enclosure, which held flat files, were not priced.

2 Clause 1: Logical Database Design Related Items

2.1 Database Table Definitions

Listings must be provided for all table definition statements and all other statements used to set up the test and qualification databases. (8.1.2.1)

Appendix B contains the scripts that were used to set up the TPC-H test and qualification databases .

2.2 Database Organization

The physical organization of tables and indexes within the test and qualification databases must be disclosed. If the column ordering of any table is different from that specified in Clause 1.4, it must be noted.

Appendix B contains the scripts that were used to create the indexes on the test and qualification databases.

2.3 Horizontal Partitioning

Horizontal partitioning of tables and rows in the test and qualification databases must be disclosed (see Clause 1.5.4).

Horizontal partitioning was not used.

2.4 Replication

Any replication of physical objects must be disclosed and must conform to the requirements of Clause 1.5.6).

No replication was used.

3 Clause 2: Queries and Update Functions Related Items

3.1 Query Language

The query language used to implement the queries must be identified.

SQL was the query language used.

3.2 Random Number Generation

The method of verification for the random number generation must be described unless the supplied DBGEN and QGEN were used.

The TPC-supplied DBGEN version 1.3.0 and QGEN version 1.3.0 were used to generate all database populations.

3.3 Substitution Parameters Generation

The method used to generate values for substitution parameters must be disclosed. If QGEN is not used for this purpose, then the source code of any non-commercial tool used must be disclosed. If QGEN is used, the version number, release number, modification number and patch level of QGEN must be disclosed.

The supplied QGEN version 1.3.0 was used to generate the substitution parameters.

3.4 Query Text and Output Data from Database

The executable query text used for query validation must be disclosed along with the corresponding output data generated during the execution of the query text against the qualification database. If minor modifications (see Clause 2.2.3) have been applied to any functional query definitions or approved variants in order to obtain executable query text, these modifications must be disclosed and justified. The justification for a particular minor query modification can apply collectively to all queries for which it has been used. The output data for the power and throughput tests must be made available electronically upon request.

Appendix C contains the query text and query output. The following modifications were used:

- In Q1, Q4, Q5, Q6, Q10, Q12, Q14, Q15 and Q20, the "dateadd" function is used to perform date arithmetic.
- In Q7, Q8 and Q9, the "datepart" function is used to extract part of a date (e.g., "YY").
- In Q2, Q3, Q10, Q18 and Q21, the "top" function is used to restrict the number of output rows.

3.5 Query Substitution Parameters and Seeds Used

All query substitution parameters used for all performance tests must be disclosed in tabular format, along with the seeds used to generate these parameters.

Appendix D contains the seed and query substitution parameters used.

3.6 Query Isolation Level

The isolation level used to run the queries must be disclosed. If the isolation level does not map closely to one of the isolation levels defined in Clause 3.4, additional descriptive detail must be provided.

The queries and transactions were run with isolation level 1.

3.7 Refresh Function Implementation

The details of how the refresh functions were implemented must be disclosed (including source code of any non-commercial program used).

Appendix E contains the source code for the refresh functions.

4 Clause 3: Database System Properties Related Items

4.1 ACID Properties

The ACID (Atomicity, Consistency, Isolation and Durability) properties of transaction processing systems must be supported by the system under test during the timed portion of this benchmark. Since the TPC-H is not a transaction processing benchmark, the ACID properties must be evaluated outside the timed portion of the test.

4.2 Atomicity

The system under test must guarantee that transactions are atomic; the system will either perform all individual operations on the data, or will assure that no partially completed operations leave any effects on the data.

4.2.1 Completed Transactions

Perform the ACID transactions for a randomly selected set of input data and verify that the appropriate rows have been changed in the ORDER, LINEITEM and HISTORY tables.

The following steps were performed to verify the Atomicity of completed transactions.

1. The total price from the ORDER table and the extended price from the LINEITEM table were retrieved for a randomly selected order key.
2. The ACID Transaction was performed using the order key from step 1.
3. The ACID Transaction committed.
4. The total price from the ORDER table and the extended price from the LINEITEM table were retrieved for the same order key. It was verified that the appropriate rows had been changed.

4.2.2 Aborted Transactions

Perform the ACID transactions for a randomly selected set of input data, submitting a ROLLBACK of the transaction for the COMMIT of the transaction. Verify that the appropriate rows have not been changed in the ORDER, LINEITEM and HISTORY tables.

The following steps were performed to verify the Atomicity of the aborted ACID transaction:

1. The total price from the ORDER table and the extended price from the LINEITEM table were retrieved for a randomly selected order key.
2. The ACID Transaction was performed using the doer key from step 1. The transaction was stopped prior to the commit.
3. The ACID Transaction was ROLLED BACK. .
4. The total price from the ORDER table and the extended price from the LINEITEM table were retrieved for the same order key used in steps 1 and 2. It was verified that the appropriate rows had not been changed.

4.3 Consistency

Consistency is the property of the application that requires any execution of transactions to take the database from one consistent state to another.

4.3.1 Consistency Test

Verify that ORDER and LINEITEM tables are initially consistent, submit the prescribed number of ACID Transactions with randomly selected input parameters, and reverify the consistency of the ORDER and LINEITEM.

The following steps were performed to verify consistency:

1. The consistency of the ORDER and LINEITEM tables was verified based on a sample of O_ORDERKEYs.
2. One hundred ACID Transactions were submitted from each of two execution streams.

3. The consistency of the ORDER and LINEITEM tables was reverified.

4.4 Isolation

Operations of concurrent transactions must yield results, which are indistinguishable from the results that would be obtained by forcing each transaction to be serially executed to completion in some order.

4.4.1 Read-Write Conflict with Commit

Demonstrate isolation for the read-write conflict of a read-write transaction and a read-only transaction when the read-write transaction is committed.

The following steps were performed to satisfy the test of isolation for a read-only and a read-write committed transaction:

1. An ACID Transaction was started for a randomly selected O_KEY, L_KEY and DELTA. The ACID Transaction was suspended prior to Commit.
2. An ACID query was started for the same O_KEY used in step 1. The ACID query blocked and did not see any uncommitted changes made by the ACID Transaction.
3. The ACID Transaction was resumed and committed.
4. The ACID query completed. It returned the data as committed by the ACID Transaction.

4.4.2 Read-Write Conflict with Rollback

Demonstrate isolation for the read-write conflict of a read-write transaction and read-only transaction when the read-write transaction is rolled back.

The following steps were performed to satisfy the test of isolation for read-only and a rolled back read-write transaction:

1. An ACID transaction was started for a randomly selected O_KEY, L_KEY and DELTA. The ACID Transaction was suspended prior to Rollback.
2. An ACID query was started for the same O_KEY used in step 1. The ACID query did not see any uncommitted changes made by the ACID Transaction.
3. The ACID Transaction was ROLLED BACK.
4. The ACID query completed.

4.4.3 Write-Write Conflict with Commit

Demonstrate isolation for the write-write conflict of two update transactions when the first transaction is committed.

The following steps were performed to verify isolation of two update transactions:

1. An ACID Transaction T1 was started for a randomly selected O_KEY, L_KEY and DELTA. The ACID transaction T1 was suspended prior to Commit.
2. Another ACID Transaction T2 was started using the same O_KEY and L_KEY and a randomly selected DELTA.
3. T2 waited.
4. The ACID transaction T1 was allowed to Commit and T2 completed.
5. It was verified that:
   T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE +
   (DELTA1*(T1.L_EXTENDEDPRICE/T1.L_QUANTITY))

4.4.4 Write-Write Conflict with Rollback

Demonstrates isolation for the write-write conflicts of two update transactions when the first transaction is rolled back.

The following steps were performed to verify the isolation of two update transactions after the first one is rolled back:

1. An ACID Transaction T1 was started for a randomly selected O_KEY, L_KEY and DELTA. The ACID Transaction T1 was suspended prior to Rollback.
2. Another ACID Transaction T2 was started using the same O_KEY and L_KEY used in step 1 and a randomly selected DELTA.
3. T2 waited.
4. T1 was allowed to ROLLBACK and T2 completed.
5. It was verified that T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE.

4.4.5 Concurrent Progress of Read and Write on Different Tables

Demonstrate the ability of read and write transactions affecting different database tables to make progress concurrently.

The following steps were performed:

1. An ACID Transaction T1 for a randomly selected O_KEY, L_KEY and DELTA. The ACID Transaction T1 was suspended prior to Commit.
2. Another ACID Transaction T2 was started using random values for PS_PARTKEY and PS_SUPPKEY.
3. T2 completed.
4. T1 completed and the appropriate rows in the ORDER, LINEITEM and HISTORY tables were changed.

4.4.6 Updates Not Indefinitely Delayed by Reads on Same Table

Demonstrate that the continuous submission of arbitrary (read-only) queries against one or more tables of the database does not indefinitely delay update transactions affecting those tables from making progress.

The following steps were performed:

1. An ACID Transaction T1 was started, executing Q1 against the qualification database. The substitution parameter was chosen from the interval [0..2159] so that the query ran for a sufficient amount of time.
2. Before T1 completed, an ACID Transaction T2 was started using randomly selected values of O_KEY, L_KEY and DELTA.
3. T2 completed before T1 completed.
4. It was verified that the appropriate rows in the ORDER, LINEITEM and HISTORY tables were changed.

4.5 Durability

The tested system must guarantee durability: the ability to preserve the effects of committed transactions and ensure database consistency after recovery from any one of the failures listed in Clause 3.5.2.

4.5.1 Failure of a Durable Medium

Guarantee the database and committed updates are preserved across a permanent irrecoverable failure of any single durable medium containing TPC-H database tables or recovery log tables.

The database log was stored on a RAID-1 protected array of six physical drives. The tables for the database were stored on 14 RAID-0 arrays each containing 14 physical drives. Two backups of the database were taken. Each backup was spread across a distinct set of seven RAID-0 arrays.

The tests were conducted on the qualification database. The steps performed are shown below.

1. The complete database was backed up twice to different sets of disk drives.

2. Six streams of ACID transactions were started. Each stream executed a minimum of 100 transactions.
3. While the test was running, one of the disks from the database RAID-1 log was removed.
4. After it was determined that the test would still run with the loss of a log disk, one physical drive of a RAID-0 data volume was removed.
5. A checkpoint was issued to force a failure.
6. The six streams of ACID transactions failed and recorded their number of committed transaction in success files.
7. The database log was dumped to disk.
8. The data/backup file disk was swapped with another disk in the same array. The database data partitions containing the failed disk were deleted and recreated to ensure that the original database data on the drive was lost.
9. The failed physical drive affected only one of the two backup copies. A database restore was done using the unaffected copy.
10. A command was issued causing the database to run through its roll-forward recovery.
11. The counts in the success files and the HISTORY table count were compared and were found to match.
12. The database log disk was replaced and a rebuild function was initiated to restore the RAID-1 log array to its protected status. The rebuild completed successfully.

4.5.2 System Crash

Guarantee the database and committed updates are preserved across an instantaneous interruption (system crash/system hang) in processing which requires the system to reboot to recover.

1. Six streams of ACID transactions were started. Each stream executed a minimum of 100 transactions.
2. While the streams of ACID transactions were running, the system was powered off.
3. When power was restored, the system rebooted and the database was restarted.
4. The database went through a recovery period.
5. The success file and the HISTORY table counts were compared and were found to match.

4.5.3 Memory Failure

Guarantee the database and committed updates are preserved across failure of all or part of memory (loss of contents).

See the previous section.

5 Clause 4: Scaling and Database Population Related Items

5.1 Cardinality of Tables

The cardinality (e.g., the number of rows) of each table of the test database, as it existed at the completion of the database load (see Clause 4.2.5), must be disclosed.

Table Name	Rows
Order	150,000,000
Lineitem	600,037,902
Customer	15,000,000
Part	20,000,000
Supplier	1,000,000
Partsupp	80,000,000
Nation	25
Region	5

5.2 Distribution of Tables and Logs

The distribution of tables and logs across all media must be explicitly described.

The following series of tables show the distribution of tables and logs across all media.

Controller	Drives	Partition*	Size	Use
Adaptec	1 - 9.1GB	C: D:	2.93GB (NTFS) 5.54GB (NTFS)	OS, DBMS Misc. (TPC-H Kit, etc.)
Mylex 0	6 - 9.1GB 2 - 9.1GB 10 - 18.2GB	G: K: L: U:	25.41GB 14.65GB 2.29GB (NTFS) 169.49GB (NTFS)	LogFile (RAID-1) TempLog (RAID-0) RF Flat Files (RAID-0) Flat Files (RAID-0)
Mylex 1	14 - 9.1GB 14 - 9.1GB	D:\devjp\lineitem_tables_file_1 D:\devjp\general_tables_file_1 D:\devjp\tempdev1 D:\devjp\backupdev1 D:\devjp\backup1gdev1 D:\devjp\lineitem_tables_file_2 D:\devjp\general_tables_file_2 D:\devjp\tempdev2 D:\devjp\backupdev2 D:\devjp\backup1gdev2	28000MB 11200MB 22400MB 47000MB (NTFS) 2000MB 10.32GB 28000MB 11200MB 22400MB 47000MB (NTFS) 2000MB 10.32GB	Lineitem File Group General File Group TempDB Backup (Qual. & Test DB) Unused Unused Lineitem File Group General File Group TempDB Backup (Qual. &Test DB) Unused Unused

Controller	Drives	Partition*	Size	Use
Mylex 2	14 - 9.1GB	D:\devjp\lineitem_tables_file_3 D:\devjp\general_tables_file_3 D:\devjp\tempdev3 D:\devjp\backupdev3 D:\devjp\backup1gdev3	28000MB 11200MB 22400MB 47000MB (NTFS) 2000MB 10.32GB	Lineitem File Group General File Group TempDB Backup (Qual. &Test DB) Unused Unused
	14 - 9.1GB	D:\devjp\lineitem_tables_file_4 D:\devjp\general_tables_file_4 D:\devjp\tempdev4 D:\devjp\backupdev4 D:\devjp\backup1gdev4	28000MB 11200MB 22400MB 47000MB (NTFS) 2000MB 10.32GB	Lineitem File Group General File Group TempDB Backup (Qual.&Test DB) Unused Unused
Mylex 3	14 - 9.1GB	D:\devjp\lineitem_tables_file_5 D:\devjp\general_tables_file_5 D:\devjp\tempdev5 D:\devjp\backupdev5 D:\devjp\backup1gdev5	28000MB 11200MB 22400MB 47000MB (NTFS) 2000MB 10.32GB	Lineitem File Group General File Group TempDB Backup (Qual. &Test DB) Unused Unused
	14 - 9.1GB	D:\devjp\lineitem_tables_file_6 D:\devjp\general_tables_file_6 D:\devjp\tempdev6 D:\devjp\backupdev6 D:\devjp\backup1gdev6	28000MB 11200MB 22400MB 47000MB (NTFS) 2000MB 10.32GB	Lineitem File Group General File Group TempDB Backup (Qual. &Test DB) Unused Unused
Mylex 4	14 - 9.1GB	D:\devjp\lineitem_tables_file_7 D:\devjp\general_tables_file_7 D:\devjp\tempdev7 D:\devjp\backupdev7	28000MB 11200MB 22400MB 58.17GB (NTFS)	Lineitem File Group General File Group TempDB Backup (Test DB)
	14 - 9.1GB	D:\devjp\lineitem_tables_file_8 D:\devjp\general_tables_file_8 D:\devjp\tempdev8 D:\devjp\backupdev8, X:	28000MB 11200MB 22400MB 58.17GB (NTFS)	Lineitem File Group General File Group TempDB Backup (Test DB), Page File

* The physical drives that do not have drive letters are assigned Windows 2000 junction points.

The priced configuration used 121 disks. An additional 10 disks were used to store the 100GB database files and were not priced.

5.3 Database Partition / Replication Mapping

The mapping of database partitions/replications must be explicitly described.

Database partitioning/replication was not used.

5.4 RAID Implementation

Implementations may use some form of RAID to ensure high availab ility. If used for data, auxiliary storage (e.g., indexes) or temporary space, the level of RAID must be disclosed for each device.

RAID-1 was used for log disks. RAID-0 was used for all other database disks and for the TempLog.

5.5 DBGEN Modifications

Any modifications to the DBGEN (see Clause 4.2.1) source code must be disclosed. In the event that a program other than DBGEN was used to populate the database, it must be disclosed in its entirety.

The standard distribution DBGEN version 1.3.0 was used for database population. No modifications were made.

5.6 Database Load Time

The database load time for the test database (see Clause 4.3) must be disclosed.

See the Executive Summary at the beginning of this report.

5.7 Data Storage Ratio

The data storage ratio must be disclosed. It is computed as the ratio between the total amount of priced disk space and the chosen test database size as defined in Clause 4.1.3.

The calculation of the data storage ratio is shown in the following table.

Disk Type	Number of Disks	Formatted Space per Disk	Total Disk Space	Scale Factor	Storage Ratio
9.1GB 15K Ultra160 SCSI Drive	121	8.47GB	1,024.87	100	10.25

The data storage ratio is 10.25, derived by dividing 1024.87GB by the database size of 100GB.

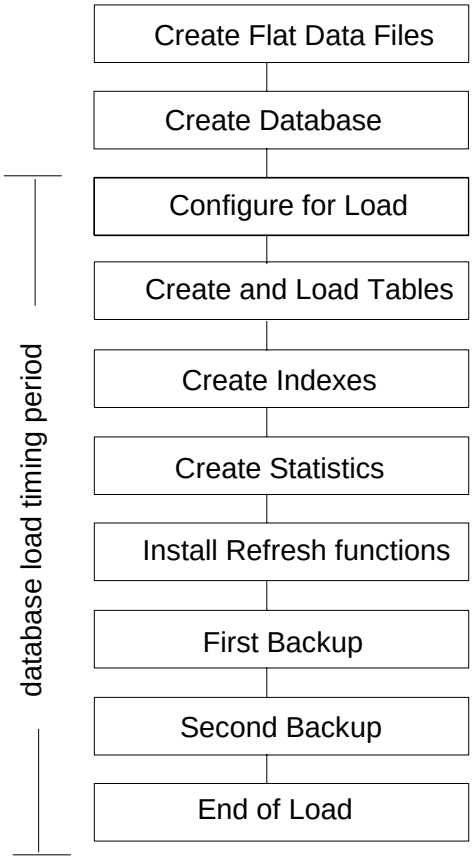
5.8 Database Load Mechanism Details and Illustration

The details of the database load must be disclosed, including a block diagram illustrating the overall process. Disclosure of the load procedure includes all steps. scripts, input and configuration files required to completely reproduce the test and qualification databases.

Flat files for each of the tables were created using DBGEN.

The tables were loaded as depicted in Figure 4-1.

Figure 4-1. Database Load Procedure



5.9 Qualification Database Configuration

Any differences between the configuration of the qualification database and the test database must be disclosed.

The qualification database used identical scripts and disk structure to create and load the data with adjustments for size difference.

6 Clause 5: Performance Metrics and Execution Rules Related Items

6.1 System Activity between Load and Performance Tests

Any system activity on the SUT that takes place between the conclusion of the load test and the beginning of the performance test must be fully disclosed.

The database system was restarted.

6.2 Steps in the Power Test

The details of the steps followed to implement the power test (e.g., system reboot, database restart) must be disclosed.

The following steps were used to implement the power test:

1. RF1 Refresh Transaction
2. Stream 00 Execution
3. RF2 Refresh Transaction

6.3 Timing Intervals for Each Query and Refresh Function

The timing intervals for each query of the measured set and for both update functions must be reported for the power test.

See the Numerical Quantities Summary in the Executive Summary at the beginning of this report.

6.4 Number of Streams for the Throughput Test

The number of execution streams used for the throughput test must be disclosed.

Five streams were used for the throughput test.

6.5 Start and End Date/Times for Each Query Stream

The start time and finish time for each query execution stream must be reported for the throughput test.

See the Numerical Quantities Summary in the Executive Summary at the beginning of this report.

6.6 Total Elapsed Time for the Measurement Interval

The total elapsed time for the measurement interval must be reported for the throughput test.

See the Numerical Quantities Summary in the Executive Summary at the beginning of this report..

6.7 Refresh Function Start Date/Time and Finish Date/Time

The start time and finish time for each update function in the update stream must be reported for the throughput test.

See the Numerical Quantities Summary in the Executive Summary at the beginning of this report.

6.8 Timing Intervals for Each Query and Each Refresh Function for Each Stream

The timing intervals for each query of each stream and for each update function must be reported for the throughput test.

See the Numerical Quantities Summary in the Executive Summary at the beginning of this report.

6.9 Performance Metrics

The computed performance metrics, related numerical quantities, and the price/performance metric must be reported.

See the Numerical Quantities Summary in the Executive Summary at the beginning of this report.

6.10 Performance Metric and Numerical Quantities from Both Runs

The performance metric and numerical quantities from both runs must be disclosed.

Two consecutive runs of the TPC-H benchmark were performed. The following table contains the results for both runs.

	QppH @ 100GB	QthH @ 100GB	QphH @ 100GB
Run1	1695.6	794.6	1160.7
Run2	1676.5	785.9	1147.9
% Difference	1.12%	1.09%	1.12%

6.11 System Activity between Tests

Any activity on the SUT that takes place between the conclusion of Run1 and the beginning of Run2 must be disclosed.

A database checkpoint was issued. The database server was restarted between runs.

7 Clause 6: SUT and Driver Implementation Related Items

7.1 Driver

A detailed textual description of how the driver performs its functions, how its various components interact and any product functionality or environmental setting on which it relies must be provided. All related source code, scripts and configurations must be disclosed. The information provided should be sufficient for an independent reconstruction of the driver.

Two scripts were used from the Microsoft tool named StepMaster. The first one is Data Load, which was used to create and load the database as well as the backup. The second one is Power/Throughput Run, which was used to run the Power and Throughput tests. The source of this program is contained in Appendix F.

The power test is invoked by calling tpcdbatch with the stream number 0 specified, an indication that the update functions must be run, and the SQL file that contains the power stream queries.

7.2 Implementation-Specific Layer

If an implementation-specific layer is used, then a detailed description of how it performs its functions must be supplied, including any related source code or scripts. This description should allow an independent reconstruction of the implementation-specific layer.

The StepMaster program is used to control and track the execution of queries. The scripts are listed in Appendix F. The following steps are performed to accomplish the Power and Throughput runs.

7.2.1 Power Run

1. Execute 16 concurrent RF1 processes, each of which will apply a segment of an update set generated by DBGEN. Each process submits multiple transactions, where a transaction spans a set of orders and their associated line items.
2. Execute the Stream 0 queries in the prescribed order.
3. Execute 16 concurrent RF2 processes, each of which will apply a segment of an update set generated by DBGEN. Each process submits multiple transactions, where a transaction spans a set of orders and their associated line items

7.2.2 Throughput Run

1. Execute five concurrent query streams. Each stream executes queries in the prescribed order for the appropriate Stream Id (1-5). Upon completion of each stream, a semaphore is set to indicate completion.
2. Execute five consecutive RF1/RF2 transactions against ascending Update sets produced by DBGEN. The first RF1 waits on a semaphore prior to beginning its insert operations.

Each step is timed by the script. The timing information is stored in an output file in StepMaster for later analysis.

8 Clause 7: Pricing Related Items

8.1 Hardware and Software Components

A detailed list of the hardware and software used in the priced system must be reported. Each item must have a vendor part number, description and release/revision level, and either general availability status or committed delivery date. If package-pricing is used, contents of the package must be disclosed. Pricing source(s) and effective date(s) must also be reported.

A detailed list of all hardware and software, including the 5-year price, is provided in the Executive Summary at the front of this report. The price quotations are included in Appendix H.

8.2 Five-Year Cost of System Configuration

The total 5-year price of the entire configuration must be reported, including hardware, software and maintenance charges. Separate component pricing is recommended. The basis of all discounts must be disclosed.

A detailed list of all hardware and software, including the 5-year price, is provided in the Executive Summary at the front of this report. The price quotations are included in Appendix H.

8.3 Availability Dates

The committed delivery date for general availability (availability date) of products used in the price calculations must be reported. When the priced system includes products with different availability dates, availability date reported on the Executive Summary must be the date by which all components are committed to being available. The Full Disclosure Report must report availability dates individually for at least each of the categories for which a pricing subtotal must be provided (see Clause 7.3.1.3).

The operating system and database software used in this benchmark are generally available. The 900MHz model of the xSeries 250 server and the processor option will be generally available April 13, 2001. The total solution availability date is April 13, 2001.

8.4 Country-Specific Pricing

Additional Clause 7 related items may be included in the Full Disclosure Report for each country-specific priced configuration. Country-specific pricing is subject to Clause 7.1.7.

The configuration is priced for the United States of America.

Clause 9: Audit Related Items

9.1 Auditor's Report

The auditor's agency name, address, phone number, and Attestation letter with a brief audit summary report indicating compliance must be included in the Full Disclosure Report. A statement should be included specifying who to contact in order to obtain further information regarding the audit process.

This implementation of the TPC Benchmark H was audited by Brad Askins of InfoSizing, Inc. Further information may be obtained from:

InfoSizing, Inc.
1373 North Franklin Street
Colorado Springs, CO 80903
Phone: 719-473-7555
Fax: 719-473-7554

INFOSIZING

TPC TRANSACTION PROCESSING PERFORMANCE COUNCIL CERTIFIED AUDITOR

Benchmark Sponsors: Richard Laviano
Mgr. Server Systems Performance
IBM Personal Systems Group
3039 Cornwallis Road
Research Triangle Park, NC
27709

September 25, 2001

I verified the TPC Benchmark™ H performance of the following configuration:

Platform:	**IBM @server xSeries 250**
Database Manager:	**Microsoft SQL Server 2000 Enterprise Edition**
Operating System:	**Microsoft Windows 2000 Advanced Server**

The results were:

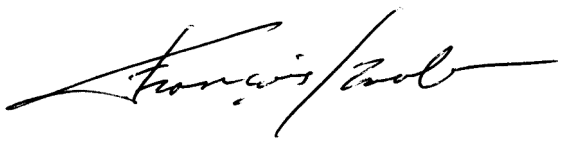
CPU (Speed)	Memory	Disks	**QphH@100GB**
IBM @server xSeries 250			
8 x Intel Pentium III Xeon (900 MHz)	2MB Cache/cpu 4 GB Main	121 x 9.1 GB	**1,147.9**

In my opinion, this performance result was produced in compliance with the TPC's requirements for the benchmark. The following verification items were given special attention:

- The database records were defined with the proper layout and size
- The database population was generated using DBGEN
- The database was properly scaled to 100GB and populated accordingly
- The compliance of the database auxiliary data structures was verified
- The database load time was correctly measured and reported

- The required ACID properties were verified and met
- The query input variables were generated by QGEN
- The query text was produced using minor modifications
- The execution of the queries against the SF1 database produced compliant answers
- A compliant implementation specific layer was used to drive the tests
- The throughput tests involved 5 query streams
- The ratio between the longest and the shortest query was such that no query timing was adjusted
- The execution times for queries and refresh functions were correctly measured and reported
- The repeatability of the measured results was verified
- The required amount of database log was configured
- The system pricing was verified for major components and maintenance
- The major pages from the FDR were verified for accuracy

Respectfully Yours,



François Raab, President

Bradley J. Askins, Auditor

1373 North Franklin Street • Colorado Springs, CO 80903-2527 • Office: 719/473-7555 • Fax: 719/473-7554

Appendix A: Tunable Parameters and System Configuration

Microsoft SQL Server 2000 Version

The following text was output was generated by executing the select @@version command:

Microsoft SQL Server 2000 - 8.00.194 (Intel X86)
Aug 6 2000 00:57:48
Copyright (c) 1988-2000 Microsoft Corporation
Enterprise Edition on Windows NT 5.0 (Build 2195:)

Microsoft SQL Server 2000 Installation

Microsoft SQL Server 2000 was installed on the SUT. All default options were selected during the install except:

- The SQL Server 2000 prerequisites were not installed.
- Database Server Enterprise Edition was installed.
- "Custom installation" was selected. The SQL Server Development tools were not installed.
- Binary Sort Order was used. (Collation Settings>Collation Designator>Binary)
- Services Accounts > Customized > SQL Server > Use Local System Account Authentication Mode > Mixed > Blank Password

Microsoft SQL Server 2000 Startup Parameters

SQLSERVR -x -c -g100 -t3502

Where :

- -x Disable the Keeping of CPU time and cache-hit ratio statistics.
- -c Start SQL Server independently of Windows NT Service Control Manager
- -g Reserve 100MB for non-buffer pool allocations.
- -t3502 Display Checkpoints in the database Log

Microsoft SQL Server 2000 Configuration Parameters

```
1> 2> sp_configure
 name
                minimum     maximum     config_value run_value
 ------------------------------------------------------------------------
                ----------- ----------- ------------ -----------
 affinity mask
                      0  2147483647         255          0
 allow updates
                      0           1           1          1
 awe enabled
                      0           1           0          0
 c2 audit mode
                      0           1           0          0
 cost threshold for parallelism
                      0       32767           0          0
 cursor threshold
                     -1  2147483647          -1         -1
 default full-text language
                      0  2147483647        1033       1033
 default language
                      0        9999           0          0
 fill factor (%)
                      0         100           0          0
 index create memory (KB)
                    704  2147483647           0          0
 lightweight pooling
                      0           1           1          1
 locks
                   5000  2147483647           0          0
 max degree of parallelism
                      0          32           0          0
 max server memory (MB)
                      4  2147483647  2147483647  2147483647
 max text repl size (B)
                      0  2147483647       65536      65536
 max worker threads
                     32       32767         512        512
 media retention
                      0         365           0          0
 min memory per query (KB)
                    512  2147483647         512        512
 min server memory (MB)
                      0  2147483647           0          0
 nested triggers
                      0           1           1          1
 network packet size (B)
                    512       65536       32767      32767
 open objects
                      0  2147483647           0          0
 priority boost
                      0           1           0          0
 query governor cost limit
                      0  2147483647           0          0
 query wait (s)
                     -1  2147483647  2147483647  2147483647
 recovery interval (min)
                      0       32767       32767      32767
 remote access
                      0           1           1          1
 remote login timeout (s)
                      0  2147483647          20         20
 remote proc trans
                      0           1           0          0
 remote query timeout (s)
                      0  2147483647         600        600
 scan for startup procs
                      0           1           0          0
 set working set size
                      0           1           0          0
 show advanced options
                      0           1           1          1
 two digit year cutoff
                   1753        9999        2049       2049
 user connections
                      0       32767           0          0
 user options
                      0       32767           0          0

1>
```

Microsoft Windows 2000 Advanced Server Installation

The default installation of Windows 2000 Advanced Server Build 2195 was used. All default options were selected during the install except:

The indexing service, IIS, Networking Services and Script Debugger were not installed.

A TCP/IP address was configured on the system.

Terminal Services was installed. The auditor used Terminal Services to complete data verification after the runs completed.

Microsoft Windows 2000 Advanced Server Configuration Parameters

Updated installation to optimize performance for applications. (System Properties > Advanced > Performance Options > Select Applications.)

Microsoft Windows 2000 Advanced Server Boot.ini Parameters

4GB of memory was enabled for Windows 2000 Advanced Server by adding the "/3GB" switch in the boot.ini file and selecting it when the system was rebooted.

Microsoft Windows 2000 Advanced Server Registry Parameters

Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management]
"ClearPageFileAtShutdown"=dword:00000000
"DisablePagingExecutive"=dword:00000000
"IoPageLockLimit"=dword:00000000
"LargeSystemCache"=dword:00000000
"NonPagedPoolQuota"=dword:00000000
"NonPagedPoolSize"=dword:00000000
"PagedPoolQuota"=dword:00000000
"PagedPoolSize"=dword:00000000
"PagingFiles"=hex(7):43,00,3a,00,5c,00,70,00,61,00,67,00,65,00,66,00,69,00,6c,\

00,65,00,2e,00,73,00,79,00,73,00,20,00,34,00,30,00,39,00,32,00,20,00,34,00,\
30,00,39,00,32,00,00,00,00,00
"SecondLevelDataCache"=dword:00000000
"SystemPages"=dword:00100000
"PhysicalAddressExtension"=dword:00000000
"DontVerifyRandomDrivers"=dword:00000001

System Hardware Information

System Information report written at: 03/08/2001 03:26:58 PM
[System Information]

[Following are sub-categories of this main category]

[System Summary]

Item	Value
OS Name	Microsoft Windows 2000 Advanced Server
Version	5.0.2195 Build 2195
OS Manufacturer	Microsoft Corporation
System Name	TPCHM
System Manufacturer	IBM
System Model	SERMOHAK
System Type	X86-based PC
Processor	x86 Family 6 Model 10 Stepping 4 GenuineIntel ~900 Mhz
Processor	x86 Family 6 Model 10 Stepping 4 GenuineIntel ~900 Mhz
Processor	x86 Family 6 Model 10 Stepping 4 GenuineIntel ~900 Mhz
Processor	x86 Family 6 Model 10 Stepping 4 GenuineIntel ~900 Mhz
BIOS Version	Ver 5.0
Windows Directory	C:\WINNT
System Directory	C:\WINNT\System32
Boot Device	\Device\Harddisk0\Partition1
Locale	United States
User Name	TPCHM\Administrator
Time Zone	Eastern Standard Time
Total Physical Memory	3,735,044 KB
Available Physical Memory	3,372,984 KB
Total Virtual Memory	11,512,876 KB
Available Virtual Memory	10,876,288 KB
Page File Space	7,777,832 KB
Page File	X:\pagefile.sys

[Hardware Resources]

[Following are sub-categories of this main category]

[Conflicts/Sharing]

Resource	Device
IRQ 17	Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller
IRQ 17	Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller

[DMA]

Channel	Device	Status
2	Standard floppy disk controller	OK
4	Direct memory access controller	OK

[Forced Hardware]

Device	PNP Device ID
No Forced Hardware	

[I/O]

Address Range	Device	Status
0x0000-0x03AF	PCI bus	OK
0x0000-0x03AF	Direct memory access controller	OK
0x03B0-0x03BB	PCI bus	OK
0x03B0-0x03BB	S3 Inc. Trio3D	OK
0x03BC-0x03BF	PCI bus	OK
0x03C0-0x03DF	PCI bus	OK
0x03C0-0x03DF	S3 Inc. Trio3D	OK
0x03E0-0x221F	PCI bus	OK
0x2000-0x20FF	Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller	OK
0x2100-0x21FF	Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller	OK
0x2200-0x221F	IBM Netfinity Fault Tolerance PCI Adapter	OK
0x0A79-0x0A79	ISAPNP Read Data Port	OK

0x0279-0x0279 ISAPNP Read Data Port OK
0x02F4-0x02F7 ISAPNP Read Data Port OK
0x002E-0x002F Motherboard resources OK
0x0438-0x0439 Motherboard resources OK
0x0430-0x0437 Motherboard resources OK
0x0060-0x0060 Standard 101/102-Key or Microsoft Natural PS/2 Keyboard OK
0x0064-0x0064 Standard 101/102-Key or Microsoft Natural PS/2 Keyboard OK
0x03F0-0x03F5 Standard floppy disk controller OK
0x03F7-0x03F7 Standard floppy disk controller OK
0x0378-0x037F Printer Port (LPT1) OK
0x03F8-0x03FF Communications Port (COM1) OK
0x02F8-0x02FF Communications Port (COM2) OK
0x00E8-0x00E9 Not Available OK
0x0020-0x0021 Advanced programmable interrupt controller OK
0x00A0-0x00A1 Advanced programmable interrupt controller OK
0x0080-0x008F Direct memory access controller OK
0x00C0-0x00DF Direct memory access controller OK
0x0040-0x0043 System timer OK
0x0070-0x0073 System CMOS/real time clock OK
0x0061-0x0061 System speaker OK
0x00F0-0x00FF Numeric data processor OK
0x0700-0x0700 Motherboard resources OK
0x0374-0x0375 Motherboard resources OK
0x0377-0x0377 Motherboard resources OK
0x0F50-0x0F58 Motherboard resources OK
0x0900-0x090F Not Available OK
0x0840-0x084F Standard Dual Channel PCI IDE Controller OK
0x01F0-0x01F7 Primary IDE Channel OK
0x03F6-0x03F6 Primary IDE Channel OK
0x0170-0x0177 Secondary IDE Channel OK
0x0376-0x0376 Secondary IDE Channel OK
0x2220-0x4FFF PCI bus OK
0x4000-0x4FFF DEC 21154 PCI to PCI bridge OK
0x4000-0x4FFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0x3000-0x3FFF DEC 21154 PCI to PCI bridge OK
0x3000-0x3FFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0x5000-0xFFFF PCI bus OK
0x5000-0xFFFF DEC 21154 PCI to PCI bridge OK
0x5000-0xFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0x7000-0x7FFF DEC 21154 PCI to PCI bridge OK
0x7000-0x7FFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0x6000-0x6FFF DEC 21154 PCI to PCI bridge OK
0x6000-0x6FFF Mylex eXtremeRAID 2000 Disk Array Controller OK

[IRQs]

IRQ Number Device
9 Microsoft ACPI-Compliant System
17 Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller
17 Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller
16 IBM Netfinity Fault Tolerance PCI Adapter
1 Standard 101/102-Key or Microsoft Natural PS/2 Keyboard
12 PS/2 Compatible Mouse
6 Standard floppy disk controller
4 Communications Port (COM1)
3 Communications Port (COM2)
5 Not Available
8 System CMOS/real time clock
13 Numeric data processor
14 Primary IDE Channel
26 Standard OpenHCD USB Host Controller
20 Mylex eXtremeRAID 2000 Disk Array Controller
21 Mylex eXtremeRAID 2000 Disk Array Controller
22 Mylex eXtremeRAID 2000 Disk Array Controller
23 Mylex eXtremeRAID 2000 Disk Array Controller
24 Mylex eXtremeRAID 2000 Disk Array Controller

[Memory]

Range Device Status
0xA0000-0xBFFFF PCI bus OK
0xA0000-0xBFFFF S3 Inc. Trio3D OK
0xF8000000-0xFFFFFFFF PCI bus OK
0xF8000000-0xFFFFFFFF S3 Inc. Trio3D OK
0xFEBFF000-0xFEBFFFFF Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller OK
0xFEBFE000-0xFEBFEFFF Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller OK
0xFEBFDC00-0xFEBFDC1F IBM Netfinity Fault Tolerance PCI Adapter OK
0xFEBFC000-0xFEBFCFFF Standard OpenHCD USB Host Controller OK
0xEB000000-0xEBFFFFFF PCI bus OK
0xEB000000-0xEBFFFFFF DEC 21154 PCI to PCI bridge OK
0xEB000000-0xEBFFFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0xF3000000-0xF7FFFFFF PCI bus OK
0xF4000000-0xF57FFFFF DEC 21154 PCI to PCI bridge OK
0xF5000000-0xF57FFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0xF6000000-0xF77FFFFF DEC 21154 PCI to PCI bridge OK
0xEB800000-0xEBFFFFFF DEC 21154 PCI to PCI bridge OK
0xEB800000-0xEBFFFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0xF7000000-0xF77FFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0xE4000000-0xE6BFFFFF PCI bus OK
0xE6C00000-0xEAFFFFFF PCI bus OK
0xEC000000-0xF2FFFFFF PCI bus OK
0xED000000-0xEE7FFFFF DEC 21154 PCI to PCI bridge OK
0xE9800000-0xE9FFFFFF DEC 21154 PCI to PCI bridge OK
0xE9800000-0xE9FFFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0xEE000000-0xEE7FFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0xEF000000-0xF07FFFFF DEC 21154 PCI to PCI bridge OK
0xEA000000-0xEA7FFFFF DEC 21154 PCI to PCI bridge OK
0xEA000000-0xEA7FFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0xF0000000-0xF07FFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0xF1000000-0xF27FFFFF DEC 21154 PCI to PCI bridge OK
0xEA800000-0xEAFFFFFF DEC 21154 PCI to PCI bridge OK
0xEA800000-0xEAFFFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK
0xF2000000-0xF27FFFFF Mylex eXtremeRAID 2000 Disk Array Controller OK

[Components]

[Following are sub-categories of this main category]

[Multimedia]

[Following are sub-categories of this main category]

[Audio Codecs]

Codec Manufacturer Description Status
File Version Size Creation Date
c:\winnt\system32\msg723.acm Microsoft Corporation
OK C:\WINNT\System32\MSG723.ACM 4.4.3385
106.77 KB (109,328 bytes) 2/23/2001 1:14:25 PM
c:\winnt\system32\lhacm.acm Microsoft Corporation
OK C:\WINNT\System32\LHACM.ACM 4.4.3385
33.27 KB (34,064 bytes) 2/23/2001 1:14:26 PM
c:\winnt\system32\iac25_32.ax Intel Corporation Indeo® audio
software OK C:\WINNT\System32\IAC25_32.AX
2.05.53 195.00 KB (199,680 bytes) 12/7/1999 6:00:00 AM
c:\winnt\system32\tssoft32.acm DSP GROUP, INC. OK
C:\WINNT\System32\TSSOFT32.ACM 1.01 9.27 KB
(9,488 bytes) 12/7/1999 6:00:00 AM
c:\winnt\system32\msgsm32.acm Microsoft Corporation
OK C:\WINNT\System32\MSGSM32.ACM 5.00.2134.1
22.27 KB (22,800 bytes) 12/7/1999 6:00:00 AM
c:\winnt\system32\msg711.acm Microsoft Corporation
OK C:\WINNT\System32\MSG711.ACM 5.00.2134.1
10.27 KB (10,512 bytes) 12/7/1999 6:00:00 AM
c:\winnt\system32\msadp32.acm Microsoft Corporation
OK C:\WINNT\System32\MSADP32.ACM 5.00.2134.1
14.77 KB (15,120 bytes) 12/7/1999 6:00:00 AM
c:\winnt\system32\imaadp32.acm Microsoft Corporation
OK C:\WINNT\System32\IMAADP32.ACM
5.00.2134.1 16.27 KB (16,656 bytes) 12/7/1999
6:00:00 AM

[Video Codecs]

Codec Manufacturer Description Status
File Version Size Creation Date
c:\winnt\system32\ir50_32.dll Intel Corporation Indeo® video
5.10 OK C:\WINNT\System32\IR50_32.DLL
R.5.10.15.2.55 737.50 KB (755,200 bytes) 12/7/1999
6:00:00 AM
c:\winnt\system32\msh261.drv Microsoft Corporation
OK C:\WINNT\System32\MSH261.DRV 4.4.3385
163.77 KB (167,696 bytes) 2/23/2001 1:14:25 PM
c:\winnt\system32\msh263.drv Microsoft Corporation
OK C:\WINNT\System32\MSH263.DRV 4.4.3385
252.27 KB (258,320 bytes) 2/23/2001 1:13:53 PM
c:\winnt\system32\msvidc32.dll Microsoft Corporation
OK C:\WINNT\System32\MSVIDC32.DLL 5.00.2134.1
27.27 KB (27,920 bytes) 12/7/1999 6:00:00 AM
c:\winnt\system32\msrle32.dll Microsoft Corporation
OK C:\WINNT\System32\MSRLE32.DLL 5.00.2134.1
10.77 KB (11,024 bytes) 12/7/1999 6:00:00 AM
c:\winnt\system32\ir32_32.dll Intel(R) Corporation OK
C:\WINNT\System32\IR32_32.DLL Not Available
194.50 KB (199,168 bytes) 12/7/1999 6:00:00 AM
c:\winnt\system32\iccvid.dll Radius Inc. OK
C:\WINNT\System32\ICCVID.DLL 1.10.0.6 108.00 KB
(110,592 bytes) 12/7/1999 6:00:00 AM

[CD-ROM]

Item Value
Drive E:
Description CD-ROM Drive
Media Loaded True
Media Type CD-ROM
Name LITEON CD-ROM LTN403
Manufacturer (Standard CD-ROM drives)
Status OK
Transfer Rate 1043.48 kbytes/sec
SCSI Target ID 0
PNP Device ID
IDE\CDROMLITEON_CD-ROM_LTN403_____
DU26____\5&326853DD&0&0.0.0

[Sound Device]

Item Value
No sound devices

[Display]

Item Value
Name S3 Inc. Trio3D
PNP Device ID
PCI\VEN_5333&DEV_8904&SUBSYS_00DB1014&REV_01\3&26
7A616A&0&30
Adapter Type S3 Trio3D, S3 compatible
Adapter Description S3 Inc. Trio3D
Adapter RAM 4.00 MB (4,194,304 bytes)
Installed Drivers s3mt3d.sys
Driver Version 5.01.526.0007
INF File s3trio3d.inf (S3Inc section)
Color Planes 1
Color Table Entries 65536
Resolution 1024 x 768 x 60 hertz
Bits/Pixel 16

[Infrared]

Item Value
No infrared devices

[Input]

[Following are sub-categories of this main category]

[Keyboard]

Item Value
Description Standard 101/102-Key or Microsoft Natural
PS/2 Keyboard
Name Enhanced (101- or 102-key)
Layout 00000409
PNP Device ID ACPI\PNP0303\4&23FD4C84&0
NumberOfFunctionKeys 12

[Pointing Device]

Item Value
Hardware Type PS/2 Compatible Mouse
Number of Buttons 3
Status OK
PNP Device ID ACPI\PNP0F13\4&23FD4C84&0
Power Management Supported False
Double Click Threshold 4
Handedness Right Handed Operation

[Modem]

Item Value
No modems

[Network]

[Following are sub-categories of this main category]

[Adapter]

Item Value
Name [00000000] IBM Netfinity Fault Tolerance PCI Adapter
Adapter Type Ethernet 802.3
Product Name IBM Netfinity Fault Tolerance PCI Adapter
Installed True
PNP Device ID PCI\VEN_1022&DEV_2000&SUBSYS_20001014&REV_44\3&267A616A&0&28
Last Reset 3/8/2001 8:17:07 AM
Index 0
Service Name PCNet5
IP Address 192.168.200.9
IP Subnet 255.255.255.0
Default IP Gateway Not Available
DHCP Enabled False
DHCP Server Not Available
DHCP Lease Expires Not Available
DHCP Lease Obtained Not Available
MAC Address 00:06:29:55:67:FA
Service Name PCnet
IRQ Number 16
I/O Port 0x2200-0x221F
Driver c:\winnt\system32\drivers\pcntn5m.sys (29968, 4.09.00)

Name [00000001] RAS Async Adapter
Adapter Type Not Available
Product Name RAS Async Adapter
Installed True
PNP Device ID Not Available
Last Reset 3/8/2001 8:17:07 AM
Index 1
Service Name AsyncMac
IP Address Not Available
IP Subnet Not Available
Default IP Gateway Not Available
DHCP Enabled False
DHCP Server Not Available
DHCP Lease Expires Not Available
DHCP Lease Obtained Not Available
MAC Address Not Available
Service Name Not Available

Name [00000002] WAN Miniport (L2TP)
Adapter Type Not Available
Product Name WAN Miniport (L2TP)
Installed True
PNP Device ID ROOT\MS_L2TPMINIPORT\0000
Last Reset 3/8/2001 8:17:07 AM
Index 2
Service Name Rasl2tp
IP Address Not Available
IP Subnet Not Available
Default IP Gateway Not Available
DHCP Enabled False
DHCP Server Not Available
DHCP Lease Expires Not Available
DHCP Lease Obtained Not Available
MAC Address Not Available
Service Name Rasl2tp
Driver c:\winnt\system32\drivers\rasl2tp.sys (50800, 5.00.2179.1)

Name [00000003] WAN Miniport (PPTP)
Adapter Type Wide Area Network (WAN)
Product Name WAN Miniport (PPTP)
Installed True
PNP Device ID ROOT\MS_PPTPMINIPORT\0000
Last Reset 3/8/2001 8:17:07 AM
Index 3
Service Name PptpMiniport
IP Address Not Available
IP Subnet Not Available
Default IP Gateway Not Available
DHCP Enabled False
DHCP Server Not Available
DHCP Lease Expires Not Available
DHCP Lease Obtained Not Available
MAC Address 50:50:54:50:30:30
Service Name PptpMiniport
Driver c:\winnt\system32\drivers\raspptp.sys (47856, 5.00.2160.1)

Name [00000004] Direct Parallel
Adapter Type Not Available
Product Name Direct Parallel
Installed True
PNP Device ID ROOT\MS_PTIMINIPORT\0000
Last Reset 3/8/2001 8:17:07 AM
Index 4
Service Name Raspti
IP Address Not Available
IP Subnet Not Available
Default IP Gateway Not Available
DHCP Enabled False
DHCP Server Not Available
DHCP Lease Expires Not Available
DHCP Lease Obtained Not Available
MAC Address Not Available
Service Name Raspti
Driver c:\winnt\system32\drivers\raspti.sys (16880, 5.00.2146.1)

Name [00000005] WAN Miniport (IP)
Adapter Type Not Available
Product Name WAN Miniport (IP)
Installed True
PNP Device ID ROOT\MS_NDISWANIP\0000
Last Reset 3/8/2001 8:17:07 AM
Index 5
Service Name NdisWan
IP Address Not Available
IP Subnet Not Available
Default IP Gateway Not Available
DHCP Enabled False
DHCP Server Not Available
DHCP Lease Expires Not Available
DHCP Lease Obtained Not Available
MAC Address Not Available
Service Name NdisWan
Driver c:\winnt\system32\drivers\ndiswan.sys (90768, 5.00.2184.1)

[Protocol]

Item Value
Name MSAFD Tcpip [TCP/IP]
ConnectionlessService False
GuaranteesDelivery True
GuaranteesSequencing True
MaximumAddressSize 16 bytes
MaximumMessageSize 0 bytes
MessageOriented False
MinimumAddressSize 16 bytes

PseudoStreamOriented False
SupportsBroadcasting False
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption False
SupportsExpeditedData True
SupportsGracefulClosing True
SupportsGuaranteedBandwidth False
SupportsMulticasting False

Name MSAFD Tcpip [UDP/IP]
ConnectionlessService True
GuaranteesDelivery False
GuaranteesSequencing False
MaximumAddressSize 16 bytes
MaximumMessageSize 65467 bytes
MessageOriented True
MinimumAddressSize 16 bytes
PseudoStreamOriented False
SupportsBroadcasting True
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption False
SupportsExpeditedData False
SupportsGracefulClosing False
SupportsGuaranteedBandwidth False
SupportsMulticasting True

Name RSVP UDP Service Provider
ConnectionlessService True
GuaranteesDelivery False
GuaranteesSequencing False
MaximumAddressSize 16 bytes
MaximumMessageSize 65467 bytes
MessageOriented True
MinimumAddressSize 16 bytes
PseudoStreamOriented False
SupportsBroadcasting True
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption True
SupportsExpeditedData False
SupportsGracefulClosing False
SupportsGuaranteedBandwidth False
SupportsMulticasting True

Name RSVP TCP Service Provider
ConnectionlessService False
GuaranteesDelivery True
GuaranteesSequencing True
MaximumAddressSize 16 bytes
MaximumMessageSize 0 bytes
MessageOriented False
MinimumAddressSize 16 bytes
PseudoStreamOriented False
SupportsBroadcasting False
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption True
SupportsExpeditedData True
SupportsGracefulClosing True
SupportsGuaranteedBandwidth False
SupportsMulticasting False

Name MSAFD NetBIOS

[\Device\NetBT_Tcpip_{4A43F74B-342E-4147-8160-9002708F74B2}] SEQPACKET 0
ConnectionlessService False
GuaranteesDelivery True
GuaranteesSequencing True
MaximumAddressSize 20 bytes
MaximumMessageSize 64000 bytes
MessageOriented True
MinimumAddressSize 20 bytes
PseudoStreamOriented False
SupportsBroadcasting False
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption False
SupportsExpeditedData False
SupportsGracefulClosing False
SupportsGuaranteedBandwidth False
SupportsMulticasting False

Name MSAFD NetBIOS

[\Device\NetBT_Tcpip_{4A43F74B-342E-4147-8160-9002708F74B2}] DATAGRAM 0
ConnectionlessService True
GuaranteesDelivery False
GuaranteesSequencing False
MaximumAddressSize 20 bytes
MaximumMessageSize 64000 bytes
MessageOriented True
MinimumAddressSize 20 bytes
PseudoStreamOriented False
SupportsBroadcasting True
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption False
SupportsExpeditedData False
SupportsGracefulClosing False
SupportsGuaranteedBandwidth False
SupportsMulticasting False

Name MSAFD NetBIOS

[\Device\NetBT_Tcpip_{3DB92953-B85C-4FBA-95D0-50356598A164}] SEQPACKET 1
ConnectionlessService False
GuaranteesDelivery True
GuaranteesSequencing True
MaximumAddressSize 20 bytes
MaximumMessageSize 64000 bytes
MessageOriented True
MinimumAddressSize 20 bytes
PseudoStreamOriented False
SupportsBroadcasting False
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption False
SupportsExpeditedData False
SupportsGracefulClosing False
SupportsGuaranteedBandwidth False
SupportsMulticasting False

Name MSAFD NetBIOS

[\Device\NetBT_Tcpip_{3DB92953-B85C-4FBA-95D0-50356598A164}] DATAGRAM 1
ConnectionlessService True
GuaranteesDelivery False
GuaranteesSequencing False
MaximumAddressSize 20 bytes
MaximumMessageSize 64000 bytes

MessageOriented True
MinimumAddressSize 20 bytes
PseudoStreamOriented False
SupportsBroadcasting True
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption False
SupportsExpeditedData False
SupportsGracefulClosing False
SupportsGuaranteedBandwidth False
SupportsMulticasting False

Name MSAFD NetBIOS

[\Device\NetBT_Tcpip_{47E041D0-A902-4BA9-B7D2-510F15E327B0}] SEQPACKET 2
ConnectionlessService False
GuaranteesDelivery True
GuaranteesSequencing True
MaximumAddressSize 20 bytes
MaximumMessageSize 64000 bytes
MessageOriented True
MinimumAddressSize 20 bytes
PseudoStreamOriented False
SupportsBroadcasting False
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption False
SupportsExpeditedData False
SupportsGracefulClosing False
SupportsGuaranteedBandwidth False
SupportsMulticasting False

Name MSAFD NetBIOS

[\Device\NetBT_Tcpip_{47E041D0-A902-4BA9-B7D2-510F15E327B0}] DATAGRAM 2
ConnectionlessService True
GuaranteesDelivery False
GuaranteesSequencing False
MaximumAddressSize 20 bytes
MaximumMessageSize 64000 bytes
MessageOriented True
MinimumAddressSize 20 bytes
PseudoStreamOriented False
SupportsBroadcasting True
SupportsConnectData False
SupportsDisconnectData False
SupportsEncryption False
SupportsExpeditedData False
SupportsGracefulClosing False
SupportsGuaranteedBandwidth False
SupportsMulticasting False

[WinSock]

Item Value
File c:\winnt\system32\winsock.dll
Version 3.10
Size 2.80 KB (2,864 bytes)

File c:\winnt\system32\wsock32.dll
Version 5.00.2152.1
Size 21.27 KB (21,776 bytes)

[Ports]

[Following are sub-categories of this main category]

[Serial]

Item Value
Name COM1
Status OK
PNP Device ID ACPI\PNP0501\1
Maximum Input Buffer Size 0
Maximum Output Buffer Size False
Settable Baud Rate True
Settable Data Bits True
Settable Flow Control True
Settable Parity True
Settable Parity Check True
Settable Stop Bits True
Settable RLSD True
Supports RLSD True
Supports 16 Bit Mode False
Supports Special Characters False
Baud Rate 9600
Bits/Byte 8
Stop Bits 1
Parity None
Busy 0
Abort Read/Write on Error 0
Binary Mode Enabled -1
Continue XMit on XOff 0
CTS Outflow Control 0
Discard NULL Bytes 0
DSR Outflow Control 0
DSR Sensitivity 0
DTR Flow Control Type Enable
EOF Character 0
Error Replace Character 0
Error Replacement Enabled 0
Event Character 0
Parity Check Enabled 0
RTS Flow Control Type Enable
XOff Character 19
XOffXMit Threshold 512
XOn Character 17
XOnXMit Threshold 2048
XOnXOff InFlow Control 0
XOnXOff OutFlow Control 0
IRQ Number 4
I/O Port 0x03F8-0x03FF
Driver c:\winnt\system32\drivers\serial.sys (62448, 5.00.2134.1)

Name COM2
Status OK
PNP Device ID ACPI\PNP0501\2
Maximum Input Buffer Size 0
Maximum Output Buffer Size False
Settable Baud Rate True
Settable Data Bits True
Settable Flow Control True
Settable Parity True
Settable Parity Check True
Settable Stop Bits True
Settable RLSD True
Supports RLSD True
Supports 16 Bit Mode False
Supports Special Characters False
Baud Rate 9600
Bits/Byte 8
Stop Bits 1

Parity None
Busy 0
Abort Read/Write on Error 0
Binary Mode Enabled -1
Continue XMit on XOff 0
CTS Outflow Control 0
Discard NULL Bytes 0
DSR Outflow Control 0
DSR Sensitivity 0
DTR Flow Control Type Enable
EOF Character 0
Error Replace Character 0
Error Replacement Enabled 0
Event Character 0
Parity Check Enabled 0
RTS Flow Control Type Enable
XOff Character 19
XOffXMit Threshold 512
XOn Character 17
XOnXMit Threshold 2048
XOnXOff InFlow Control 0
XOnXOff OutFlow Control 0
IRQ Number 3
I/O Port 0x02F8-0x02FF
Driver c:\winnt\system32\drivers\serial.sys (62448, 5.00.2134.1)

[Parallel]

Item Value
Name LPT1
PNP Device ID ACPI\PNP0400\1

[Storage]

[Following are sub-categories of this main category]

[Drives]

Item Value
Drive A:
Description 3 1/2 Inch Floppy Drive

Drive C:
Description Local Fixed Disk
Compressed False
File System NTFS
Size 2.93 GB (3,142,021,120 bytes)
Free Space 1.99 GB (2,139,570,176 bytes)
Volume Name
Volume Serial Number FC017BDF
Partition Disk #0, Partition #0
Partition Size 2.93 GB (3,142,024,704 bytes)
Starting Offset 32256 bytes
Drive Description Disk drive
Drive Manufacturer (Standard disk drives)
Drive Model IBM-PSG ST39251LC !# SCSI Disk Device
Drive BytesPerSector 512
Drive MediaLoaded True
Drive MediaType Fixed hard disk media
Drive Partitions 2
Drive SCSIBus 0
Drive SCSILogicalUnit 0
Drive SCSIPort 3
Drive SCSITargetId 0
Drive SectorsPerTrack 63
Drive Size 9097159680 bytes
Drive TotalCylinders 1106
Drive TotalSectors 17767890
Drive TotalTracks 282030
Drive TracksPerCylinder 255

Drive D:
Description Local Fixed Disk
Compressed False
File System NTFS
Size 5.54 GB (5,946,843,136 bytes)
Free Space 4.08 GB (4,376,539,136 bytes)
Volume Name New Volume
Volume Serial Number 28AB7E97
Partition Disk #0, Partition #1
Partition Size 5.54 GB (5,946,877,440 bytes)
Starting Offset Not Available
Drive Description Disk drive
Drive Manufacturer (Standard disk drives)
Drive Model IBM-PSG ST39251LC !# SCSI Disk Device
Drive BytesPerSector 512
Drive MediaLoaded True
Drive MediaType Fixed hard disk media
Drive Partitions 2
Drive SCSIBus 0
Drive SCSILogicalUnit 0
Drive SCSIPort 3
Drive SCSITargetId 0
Drive SectorsPerTrack 63
Drive Size 9097159680 bytes
Drive TotalCylinders 1106
Drive TotalSectors 17767890
Drive TotalTracks 282030
Drive TracksPerCylinder 255

Drive G:
Description Local Fixed Disk
Compressed Not Available
File System Not Available
Size Not Available
Free Space Not Available
Volume Name Not Available
Volume Serial Number Not Available

Drive K:
Description Local Fixed Disk
Compressed Not Available
File System Not Available
Size Not Available
Free Space Not Available
Volume Name Not Available
Volume Serial Number Not Available

Drive L:
Description Local Fixed Disk
Compressed False
File System NTFS
Size 2.24 GB (2,409,971,712 bytes)
Free Space 5.42 MB (5,685,248 bytes)
Volume Name RF Flatfiles
Volume Serial Number CCF93EB6
Partition Disk #6, Partition #0
Partition Size 16.89 GB (18,136,742,400 bytes)
Starting Offset 8225280 bytes
Drive Description \\.\PHYSICALDRIVE6
Drive Manufacturer Not Available
Drive Model Not Available
Drive BytesPerSector 512
Drive MediaLoaded True

Drive MediaType Fixed hard disk media
Drive Partitions 2
Drive SCSIBus 4
Drive SCSILogicalUnit 0
Drive SCSIPort 6
Drive SCSITargetId 1
Drive SectorsPerTrack 63
Drive Size 18144967680 bytes
Drive TotalCylinders 2206
Drive TotalSectors 35439390
Drive TotalTracks 562530
Drive TracksPerCylinder 255

Drive M:
Description Local Fixed Disk
Compressed False
File System NTFS
Size 45.90 GB (49,285,844,992 bytes)
Free Space 2.61 GB (2,800,754,688 bytes)
Volume Name back4
Volume Serial Number 7C8C3C9E
Partition Disk #4, Partition #0
Partition Size 118.33 GB (127,055,900,160 bytes)
Starting Offset 8225280 bytes
Drive Description \\.\PHYSICALDRIVE4
Drive Manufacturer Not Available
Drive Model Not Available
Drive BytesPerSector 512
Drive MediaLoaded True
Drive MediaType Fixed hard disk media
Drive Partitions 5
Drive SCSIBus 4
Drive SCSILogicalUnit 0
Drive SCSIPort 5
Drive SCSITargetId 1
Drive SectorsPerTrack 63
Drive Size 127064125440 bytes
Drive TotalCylinders 15448
Drive TotalSectors 248172120
Drive TotalTracks 3939240
Drive TracksPerCylinder 255

Drive U:
Description Local Fixed Disk
Compressed False
File System NTFS
Size 169.29 GB (181,770,428,416 bytes)
Free Space 60.60 GB (65,068,666,880 bytes)
Volume Name Flat Files
Volume Serial Number E0F236B3
Partition Disk #7, Partition #0
Partition Size 169.29 GB (181,770,462,720 bytes)
Starting Offset 8225280 bytes
Drive Description \\.\PHYSICALDRIVE7
Drive Manufacturer Not Available
Drive Model Not Available
Drive BytesPerSector 512
Drive MediaLoaded True
Drive MediaType Fixed hard disk media
Drive Partitions 1
Drive SCSIBus 4
Drive SCSILogicalUnit 0
Drive SCSIPort 6
Drive SCSITargetId 2
Drive SectorsPerTrack 63
Drive Size 181778688000 bytes
Drive TotalCylinders 22100
Drive TotalSectors 355036500
Drive TotalTracks 5635500
Drive TracksPerCylinder 255

Drive X:
Description Local Fixed Disk
Compressed False
File System NTFS
Size 58.17 GB (62,462,742,528 bytes)
Free Space 11.43 GB (12,273,233,920 bytes)
Volume Name back8
Volume Serial Number 50127009
Partition Disk #11, Partition #0
Partition Size 118.33 GB (127,055,900,160 bytes)
Starting Offset 8225280 bytes
Drive Description \\.\PHYSICALDRIVE11
Drive Manufacturer Not Available
Drive Model Not Available
Drive BytesPerSector 512
Drive MediaLoaded True
Drive MediaType Fixed hard disk media
Drive Partitions 4
Drive SCSIBus 4
Drive SCSILogicalUnit 0
Drive SCSIPort 8
Drive SCSITargetId 1
Drive SectorsPerTrack 63
Drive Size 127064125440 bytes
Drive TotalCylinders 15448
Drive TotalSectors 248172120
Drive TotalTracks 3939240
Drive TracksPerCylinder 255

Drive F:
Description Network Connection
Provider Name \\tpchws3\tpchdata

[SCSI]

Item Value
Name Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller
Caption Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller
Driver aic78u2
Status OK
PNP Device ID
PCI\VEN_9005&DEV_005F&SUBSYS_080F9005&REV_00\3&267A616A&0&08
Device ID
PCI\VEN_9005&DEV_005F&SUBSYS_080F9005&REV_00\3&267A616A&0&08
Device Map Not Available
Index Not Available
Max Number Controlled Not Available
IRQ Number 17
I/O Port 0x2000-0x20FF
Driver c:\winnt\system32\drivers\aic78u2.sys (65168, v3.00a)

Name Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller
Caption Adaptec AIC-7896/AIC-7897 PCI Ultra2 SCSI Controller
Driver aic78u2
Status OK
PNP Device ID
PCI\VEN_9005&DEV_005F&SUBSYS_080F9005&REV_00\3&267A616A&0&09
Device ID
PCI\VEN_9005&DEV_005F&SUBSYS_080F9005&REV_00\3&267A616A&0&09
Device Map Not Available

Index Not Available
Max Number Controlled Not Available
IRQ Number 17
I/O Port 0x2100-0x21FF
Driver c:\winnt\system32\drivers\aic78u2.sys (65168, v3.00a)

Name Mylex eXtremeRAID 2000 Disk Array Controller
Caption Mylex eXtremeRAID 2000 Disk Array Controller
Driver dac2w2k
Status OK
PNP Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&1138865F&0&4008
Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&1138865F&0&4008
Device Map Not Available
Index Not Available
Max Number Controlled Not Available
IRQ Number 20
I/O Port 0x4000-0x4FFF
Driver c:\winnt\system32\drivers\dac2w2k.sys (185584, 9.00-04)

Name Mylex eXtremeRAID 2000 Disk Array Controller
Caption Mylex eXtremeRAID 2000 Disk Array Controller
Driver dac2w2k
Status OK
PNP Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&30418910&0&4010
Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&30418910&0&4010
Device Map Not Available
Index Not Available
Max Number Controlled Not Available
IRQ Number 21
I/O Port 0x3000-0x3FFF
Driver c:\winnt\system32\drivers\dac2w2k.sys (185584, 9.00-04)

Name Mylex eXtremeRAID 2000 Disk Array Controller
Caption Mylex eXtremeRAID 2000 Disk Array Controller
Driver dac2w2k
Status OK
PNP Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&27CD9B08&0&4018
Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&27CD9B08&0&4018
Device Map Not Available
Index Not Available
Max Number Controlled Not Available
IRQ Number 22
I/O Port 0x7000-0x7FFF
Driver c:\winnt\system32\drivers\dac2w2k.sys (185584, 9.00-04)

Name Mylex eXtremeRAID 2000 Disk Array Controller
Caption Mylex eXtremeRAID 2000 Disk Array Controller
Driver dac2w2k
Status OK
PNP Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&8C49857&0&4020
Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&8C49857&0&4020
Device Map Not Available
Index Not Available
Max Number Controlled Not Available
IRQ Number 23
I/O Port 0x6000-0x6FFF
Driver c:\winnt\system32\drivers\dac2w2k.sys (185584, 9.00-04)

Name Mylex eXtremeRAID 2000 Disk Array Controller
Caption Mylex eXtremeRAID 2000 Disk Array Controller
Driver dac2w2k
Status OK
PNP Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&375C4928&0&4028
Device ID
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&375C4928&0&4028
Device Map Not Available
Index Not Available
Max Number Controlled Not Available
IRQ Number 24
I/O Port 0x5000-0xFFFF
Driver c:\winnt\system32\drivers\dac2w2k.sys (185584, 9.00-04)

[Printing]

Name Port Name Server Name
No printing information

[Problem Devices]

Device PNP Device ID Error Code
Not Available ACPI\IBM37C0\4&23FD4C84&0 28
Not Available ACPI\IBM37D0\4&23FD4C84&0 28

[USB]

Device PNP Device ID
Standard OpenHCD USB Host Controller
PCI\VEN_1166&DEV_0220&SUBSYS_02201166&REV_04\3&267A616A&0&7A
USB Root Hub USB\ROOT_HUB\4&372644EA&0

[Software Environment]

[Following are sub-categories of this main category]

[Drivers]

Name Description File Type Started
Start Mode State Status Error Control
Accept Pause Accept Stop
abiosdsk Abiosdsk Not Available Kernel Driver
False Disabled Stopped OK Ignore False
False
abp480n5 abp480n5 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
acpi Microsoft ACPI Driver
c:\winnt\system32\drivers\acpi.sys Kernel Driver
True Boot Running OK Normal False
True
acpiec ACPIEC c:\winnt\system32\drivers\acpiec.sys
Kernel Driver False Disabled Stopped OK
Normal False False
adpu160m adpu160m Not Available Kernel Driver
False Disabled Stopped OK Normal False
False

afd AFD Networking Support Environment
c:\winnt\system32\drivers\afd.sysKernel Driver True
Auto Running OK Normal False True
aha154x Aha154x Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
aic116x aic116x Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
aic78u2 aic78u2 c:\winnt\system32\drivers\aic78u2.sys
Kernel Driver True Boot Running OK
Normal False True
aic78xx aic78xx Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
ami0nt ami0nt Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
amsint amsint Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
asc asc Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
asc3350p asc3350p Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
asc3550 asc3550 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
asyncmac RAS Asynchronous Media Driver
c:\winnt\system32\drivers\asyncmac.sys Kernel Driver
False Manual Stopped OK Normal False
False
atapi Standard IDE/ESDI Hard Disk Controller
c:\winnt\system32\drivers\atapi.sys Kernel Driver
True Boot Running OK Normal False
True
atdisk Atdisk Not Available Kernel Driver
False Disabled Stopped OK Ignore False
False
atmarpc ATM ARP Client Protocol
c:\winnt\system32\drivers\atmarpc.sys Kernel Driver
False Manual Stopped OK Normal False
False
audstub Audio Stub Driver
c:\winnt\system32\drivers\audstub.sys Kernel Driver
True Manual Running OK Normal False
True
beep Beep c:\winnt\system32\drivers\beep.sys
Kernel Driver True System Running OK
Normal False True
buslogic BusLogic Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
cd20xrnt cd20xrnt Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
cdaudio Cdaudio c:\winnt\system32\drivers\cdaudio.sys
Kernel Driver False System Stopped OK
Ignore False False
cdfs Cdfs c:\winnt\system32\drivers\cdfs.sys
File System Driver True Disabled Running OK
Normal False True
cdrom CD-ROM Driver c:\winnt\system32\drivers\cdrom.sys
Kernel Driver True System Running OK
Normal False True
changer Changer Not Available Kernel Driver
False System Stopped OK Ignore False
False
cpqarray Cpqarray Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
cpqarry2 cpqarry2 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
cpqfcalm cpqfcalm Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
cpqfws2e cpqfws2e Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
dac2w2k dac2w2k c:\winnt\system32\drivers\dac2w2k.sys
Kernel Driver True Boot Running OK
Normal False True
dac960nt dac960nt Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
deckzpsx deckzpsx Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
dfsdriver DfsDriver c:\winnt\system32\drivers\dfs.sysFile System
Driver True Boot Running OK Normal
False True
disk Disk Driver c:\winnt\system32\drivers\disk.sys
Kernel Driver True Boot Running OK
Normal False True
diskperf Diskperf c:\winnt\system32\drivers\diskperf.sys
Kernel Driver False Disabled Stopped OK
Normal False False
dmboot dmboot c:\winnt\system32\drivers\dmboot.sys
Kernel Driver False Disabled Stopped OK
Normal False False
dmio Logical Disk Manager Driver
c:\winnt\system32\drivers\dmio.sys Kernel Driver
True Boot Running OK Normal False
True
dmload dmload c:\winnt\system32\drivers\dmload.sys
Kernel Driver True Boot Running OK
Normal False True
efs EFS c:\winnt\system32\drivers\efs.sysFile System
Driver True Disabled Running OK Normal
False True
fastfat Fastfat c:\winnt\system32\drivers\fastfat.sys
File System Driver True Disabled Running OK
Normal False True
fd16_700 Fd16_700 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
fdc Floppy Disk Controller Driver
c:\winnt\system32\drivers\fdc.sysKernel Driver True
Manual Running OK Normal False True
fireport fireport Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
flashpnt flashpnt Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
flpydisk Floppy Disk Driver
c:\winnt\system32\drivers\flpydisk.sys Kernel Driver
True Manual Running OK Normal False
True
ftdisk Volume Manager Driver
c:\winnt\system32\drivers\ftdisk.sys Kernel Driver

```
True Boot Running OK Normal False
True
gamdrv gamdrv c:\winnt\system32\drivers\gamdrv.sys
Kernel Driver True Boot Running OK
Normal False True
gpc Generic Packet Classifier
c:\winnt\system32\drivers\msgpc.sys Kernel Driver
True Manual Running OK Normal False
True
i8042prt i8042 Keyboard and PS/2 Mouse Port Driver
c:\winnt\system32\drivers\i8042prt.sys Kernel Driver
True System Running OK Normal False
True
ini910u ini910u Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
intelide IntelIde Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
ipfilterdriver IP Traffic Filter Driver
c:\winnt\system32\drivers\ipfltdrv.sys Kernel Driver
False Manual Stopped OK Normal False
False
ipinip IP in IP Tunnel Driverc:\winnt\system32\drivers\ipinip.sys
Kernel Driver False Manual Stopped OK
Normal False False
ipnat IP Network Address Translator
c:\winnt\system32\drivers\ipnat.sys Kernel Driver
False Manual Stopped OK Normal False
False
ipsec IPSEC driver c:\winnt\system32\drivers\ipsec.sys
Kernel Driver True Manual Running OK
Normal False True
ipsraidn ipsraidn Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
isapnp PnP ISA/EISA Bus Driver
c:\winnt\system32\drivers\isapnp.sys Kernel Driver
True Boot Running OK Critical False
True
kbdclass Keyboard Class Driver
c:\winnt\system32\drivers\kbdclass.sys Kernel Driver
True System Running OK Normal False
True
ksecdd KSecDD c:\winnt\system32\drivers\ksecdd.sys
Kernel Driver True Boot Running OK
Normal False True
lbrtfdc lbrtfdc Not Available Kernel Driver
False System Stopped OK Ignore False
False
lp6nds35 lp6nds35 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
macdisk macdisk c:\winnt\system32\drivers\mac2w2k.sys
Kernel Driver True Boot Running OK
Normal False True
mnmdd mnmdd c:\winnt\system32\drivers\mnmdd.sys
Kernel Driver True System Running OK
Ignore False True
modem Modem c:\winnt\system32\drivers\modem.sys
Kernel Driver False Manual Stopped OK
Ignore False False
mouclass Mouse Class Driver
c:\winnt\system32\drivers\mouclass.sys Kernel Driver
True System Running OK Normal False
True
mountmgr MountMgr c:\winnt\system32\drivers\mountmgr.sys
Kernel Driver True Boot Running OK
Normal False True
mraid35x mraid35x Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
mrxsmb MRXSMB c:\winnt\system32\drivers\mrxsmb.sys
File System Driver True System Running OK
Normal False True
msfs Msfs c:\winnt\system32\drivers\msfs.sys
File System Driver True System Running OK
Normal False True
msksrv Microsoft Streaming Service Proxy
c:\winnt\system32\drivers\msksrv.sys Kernel Driver
False Manual Stopped OK Normal False
False
mspclock Microsoft Streaming Clock Proxy
c:\winnt\system32\drivers\mspclock.sys Kernel Driver
False Manual Stopped OK Normal False
False
mspqm Microsoft Streaming Quality Manager Proxy
c:\winnt\system32\drivers\mspqm.sys Kernel Driver
False Manual Stopped OK Normal False
False
mup Mup c:\winnt\system32\drivers\mup.sys
File System Driver True Boot Running OK
Normal False True
ncrc710 Ncrc710 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
ndis NDIS System Driver c:\winnt\system32\drivers\ndis.sys
Kernel Driver True Boot Running OK
Normal False True
ndistapi Remote Access NDIS TAPI Driver
c:\winnt\system32\drivers\ndistapi.sys Kernel Driver
True Manual Running OK Normal False
True
ndiswan Remote Access NDIS WAN Driver
c:\winnt\system32\drivers\ndiswan.sys Kernel Driver
True Manual Running OK Normal False
True
ndproxy NDIS Proxy
c:\winnt\system32\drivers\ndproxy.sys Kernel Driver
True Manual Running OK Normal False
True
netbios NetBIOS Interface
c:\winnt\system32\drivers\netbios.sys File System Driver
True System Running OK Normal False
True
netbt NetBios over Tcpip c:\winnt\system32\drivers\netbt.sys
Kernel Driver True System Running OK
Normal False True
netdetect NetDetect c:\winnt\system32\drivers\netdtect.sys
Kernel Driver False Manual Stopped OK
Normal False False
npfs Npfs c:\winnt\system32\drivers\npfs.sys
File System Driver True System Running OK
Normal False True
ntfs Ntfs c:\winnt\system32\drivers\ntfs.sys
File System Driver True Disabled Running OK
Normal False True
null Null c:\winnt\system32\drivers\null.sys
Kernel Driver True System Running OK
Normal False True
nwlnkflt IPX Traffic Filter Driver
c:\winnt\system32\drivers\nwlnkflt.sys Kernel Driver
False Manual Stopped OK Normal False
False
```

nwlnkfwd IPX Traffic Forwarder Driver
c:\winnt\system32\drivers\nwlnkfwd.sys Kernel Driver
False Manual Stopped OK Normal False
False
openhci Microsoft USB Open Host Controller Driver
c:\winnt\system32\drivers\openhci.sys Kernel Driver
True Manual Running OK Normal False
True
parallel Parallel class driver
c:\winnt\system32\drivers\parallel.sys Kernel Driver
True Manual Running OK Normal False
True
parport Parallel port driver
c:\winnt\system32\drivers\parport.sys Kernel Driver
True System Running OK Ignore False
True
partmgr PartMgr c:\winnt\system32\drivers\partmgr.sys
Kernel Driver True Boot Running OK
Normal False True
parvdm ParVdm c:\winnt\system32\drivers\parvdm.sys
Kernel Driver True Auto Running OK
Ignore False True
pci PCI Bus Driver c:\winnt\system32\drivers\pci.sys
Kernel Driver True Boot Running OK
Critical False True
pcidump PCIDump Not Available Kernel Driver
False System Stopped OK Ignore False
False
pciide PCIIde c:\winnt\system32\drivers\pciide.sys
Kernel Driver True Boot Running OK
Normal False True
pcmcia Pcmcia c:\winnt\system32\drivers\pcmcia.sys
Kernel Driver False Disabled Stopped OK
Normal False False
pcnet AMD PCNET Compatable Adapter Driver
c:\winnt\system32\drivers\pcntn5m.sys Kernel Driver
True Manual Running OK Normal False
True
pdcomp PDCOMP Not Available Kernel Driver
False Manual Stopped OK Ignore False
False
pdframe PDFRAME Not Available Kernel Driver
False Manual Stopped OK Ignore False
False
pdreli PDRELI Not Available Kernel Driver
False Manual Stopped OK Ignore False
False
pdrframe PDRFRAME Not Available Kernel Driver
False Manual Stopped OK Ignore False
False
pptpminiport WAN Miniport (PPTP)
c:\winnt\system32\drivers\raspptp.sys Kernel Driver
True Manual Running OK Normal False
True
ptilink Direct Parallel Link Driver
c:\winnt\system32\drivers\ptilink.sys Kernel Driver
True Manual Running OK Normal False
True
ql1080 ql1080 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
ql10wnt Ql10wnt Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
ql1240 ql1240 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
ql2100 ql2100 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
rasacd Remote Access Auto Connection Driver
c:\winnt\system32\drivers\rasacd.sys Kernel Driver
True System Running OK Normal False
True
rasl2tp WAN Miniport (L2TP)
c:\winnt\system32\drivers\rasl2tp.sys Kernel Driver
True Manual Running OK Normal False
True
raspti Direct Parallel c:\winnt\system32\drivers\raspti.sys
Kernel Driver True Manual Running OK
Normal False True
rca Microsoft Streaming Network Raw Channel Access
c:\winnt\system32\drivers\rca.sysKernel Driver False
Manual Stopped OK Normal False False
rdbss Rdbss c:\winnt\system32\drivers\rdbss.sys
File System Driver True System Running OK
Normal False True
rdpdr Terminal Server Device Redirector Driver
c:\winnt\system32\drivers\rdpdr.sys Kernel Driver
True Manual Running OK Normal False
True
rdpwd RDPWD c:\winnt\system32\drivers\rdpwd.sys
Kernel Driver True Manual Running OK
Ignore False True
redbook Digital CD Audio Playback Filter Driver
c:\winnt\system32\drivers\redbook.sys Kernel Driver
False System Stopped OK Normal False
False
s3inc S3Inc c:\winnt\system32\drivers\s3mt3d.sys
Kernel Driver True Manual Running OK
Ignore False True
serenum Serenum Filter Driver
c:\winnt\system32\drivers\serenum.sys Kernel Driver
True Manual Running OK Normal False
True
serial Serial port driver c:\winnt\system32\drivers\serial.sys
Kernel Driver True System Running OK
Ignore False True
sfloppy Sfloppy c:\winnt\system32\drivers\sfloppy.sys
Kernel Driver False System Stopped OK
Ignore False False
sglfb sglfb Not Available Kernel Driver
False System Stopped OK Normal False
False
simbad Simbad Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
sparrow Sparrow Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
srv Srv c:\winnt\system32\drivers\srv.sysFile System
Driver True Manual Running OK Normal
False True
swenum Software Bus Driver
c:\winnt\system32\drivers\swenum.sys Kernel Driver
True Manual Running OK Normal False
True
symc810 symc810 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
symc8xx symc8xx Not Available Kernel Driver
False Disabled Stopped OK Normal False
False

sym_hi sym_hi Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
tcpip TCP/IP Protocol Driver
c:\winnt\system32\drivers\tcpip.sys Kernel Driver
True System Running OK Normal False
True
tdasync TDASYNC
c:\winnt\system32\drivers\tdasync.sys Kernel Driver
False Manual Stopped OK Ignore False
False
tdipx TDIPX c:\winnt\system32\drivers\tdipx.sys
Kernel Driver False Manual Stopped OK
Ignore False False
tdnetb TDNETB c:\winnt\system32\drivers\tdnetb.sys
Kernel Driver False Manual Stopped OK
Ignore False False
tdpipe TDPIPE c:\winnt\system32\drivers\tdpipe.sys
Kernel Driver False Manual Stopped OK
Ignore False False
tdspx TDSPX c:\winnt\system32\drivers\tdspx.sys
Kernel Driver False Manual Stopped OK
Ignore False False
tdtcp TDTCP c:\winnt\system32\drivers\tdtcp.sys
Kernel Driver True Manual Running OK
Ignore False True
termdd Terminal Device Driver
c:\winnt\system32\drivers\termdd.sys Kernel Driver
True Auto Running OK Normal False
True
tga tga Not Available Kernel Driver
False System Stopped OK Ignore False
False
udfs Udfs c:\winnt\system32\drivers\udfs.sys
File System Driver False Disabled Stopped OK
Normal False False
ultra66 ultra66 Not Available Kernel Driver
False Disabled Stopped OK Normal False
False
update Microcode Update Driver
c:\winnt\system32\drivers\update.sys Kernel Driver
True Manual Running OK Normal False
True
usbhub Microsoft USB Standard Hub Driver
c:\winnt\system32\drivers\usbhub.sys Kernel Driver
True Manual Running OK Normal False
True
vgasave VgaSave c:\winnt\system32\drivers\vga.sys
Kernel Driver True System Running OK
Ignore False True
wanarp Remote Access IP ARP Driver
c:\winnt\system32\drivers\wanarp.sys Kernel Driver
True Manual Running OK Normal False
True
wdica WDICA Not Available Kernel Driver
False Manual Stopped OK Ignore False
False

[Environment Variables]

Variable Value User Name
ComSpec %SystemRoot%\system32\cmd.exe <SYSTEM>
Os2LibPath %SystemRoot%\system32\os2\dll;
<SYSTEM>
Path
%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System3
2\Wbem;C:\Program Files\Microsoft SQL Server\80\Tools\BINN
<SYSTEM>
windir %SystemRoot% <SYSTEM>
OS Windows_NT <SYSTEM>
PROCESSOR_ARCHITECTURE x86 <SYSTEM>
PROCESSOR_LEVEL 6 <SYSTEM>
PROCESSOR_IDENTIFIER x86 Family 6 Model 10 Stepping 4,
GenuineIntel <SYSTEM>
PROCESSOR_REVISION 0a04 <SYSTEM>
NUMBER_OF_PROCESSORS 4 <SYSTEM>
PATHEXT
.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
<SYSTEM>
TEMP %SystemRoot%\TEMP <SYSTEM>
TMP %SystemRoot%\TEMP <SYSTEM>
TEMP %USERPROFILE%\Local Settings\Temp
TPCHM\Administrator
TMP %USERPROFILE%\Local Settings\Temp
TPCHM\Administrator

[Jobs]

[Following are sub-categories of this main category]

[Print]

Document Size Owner Notify Status Time
Submitted Start Time Until Time Elapsed Time Pages Printed
Job ID Priority Parameters Driver Name Print
Processor Host Print Queue Data Type Name
No print jobs

[Network Connections]

Local Name Remote Name Type Status
User Name
F: \\tpchws3\tpchdata Disk OK
TPCHM\Administrator
H: \\tpchws3\fdrive Disk Error

[Running Tasks]

Name Path Process ID Priority Min Working Set
Max Working Set Start Time Version Size File Date
system idle process Not Available 0 0
Not Available Not Available Not Available
Unknown Unknown Unknown
system Not Available 8 8 0
1413120 Not Available Unknown Unknown Unknown
smss.exe c:\winnt\system32\smss.exe 184 11
204800 1413120 3/8/2001 1:19:10 PM 5.00.2170.1
44.27 KB (45,328 bytes) 12/7/1999 6:00:00 AM
csrss.exe Not Available 212 13 Not Available
Not Available 3/8/2001 1:19:16 PM Unknown Unknown
Unknown
winlogon.exe c:\winnt\system32\winlogon.exe 236 13
204800 1413120 3/8/2001 1:19:18 PM 5.00.2182.1
173.27 KB (177,424 bytes) 12/7/1999 6:00:00 AM
services.exe c:\winnt\system32\services.exe 268 9
204800 1413120 3/8/2001 1:19:19 PM 5.00.2134.1
86.77 KB (88,848 bytes) 12/7/1999 6:00:00 AM
lsass.exe c:\winnt\system32\lsass.exe 280 13
204800 1413120 3/8/2001 1:19:19 PM 5.00.2184.1
32.77 KB (33,552 bytes) 12/7/1999 6:00:00 AM
svchost.exe c:\winnt\system32\svchost.exe 432 8
204800 1413120 3/8/2001 1:19:21 PM 5.00.2134.1
7.77 KB (7,952 bytes) 12/7/1999 6:00:00 AM

spoolsv.exe c:\winnt\system32\spoolsv.exe 456 8 204800 1413120 3/8/2001 1:19:21 PM 5.00.2161.1 43.77 KB (44,816 bytes) 2/23/2001 7:56:07 AM
msdtc.exe c:\winnt\system32\msdtc.exe 484 8 204800 1413120 3/8/2001 1:19:21 PM 1999.9.3421.3 6.77 KB (6,928 bytes) 2/23/2001 8:12:07 AM
svchost.exe c:\winnt\system32\svchost.exe 648 8 204800 1413120 3/8/2001 1:19:23 PM 5.00.2134.1 7.77 KB (7,952 bytes) 12/7/1999 6:00:00 AM
gamscm.exe c:\winnt\system32\gamserv\gamscm.exe 660 8 204800 1413120 3/8/2001 1:19:23 PM Not Available 119.28 KB (122,144 bytes) 2/23/2001 3:12:48 PM
llssrv.exe c:\winnt\system32\llssrv.exe 684 9 204800 1413120 3/8/2001 1:19:23 PM 5.00.2167.1 114.27 KB (117,008 bytes) 12/7/1999 6:00:00 AM
gamserv.exe c:\winnt\system32\gamserv\gamserv.exe 692 13 204800 1413120 3/8/2001 1:19:23 PM Not Available 128.20 KB (131,281 bytes) 2/23/2001 3:12:48 PM
gamevent.exe c:\winnt\system32\gamserv\gamevent.exe 700 13 204800 1413120 3/8/2001 1:19:23 PM Not Available 88.71 KB (90,834 bytes) 2/23/2001 3:12:48 PM
gamevlog.exe c:\winnt\system32\gamserv\gamevlog.exe 708 13 204800 1413120 3/8/2001 1:19:23 PM Not Available 187.35 KB (191,842 bytes) 2/23/2001 3:12:48 PM
regsvc.exe c:\winnt\system32\regsvc.exe 752 8 204800 1413120 3/8/2001 1:19:23 PM 5.00.2155.1 65.27 KB (66,832 bytes) 12/7/1999 6:00:00 AM
mstask.exe c:\winnt\system32\mstask.exe 768 8 204800 1413120 3/8/2001 1:19:23 PM 4.71.2137.1 115.27 KB (118,032 bytes) 2/23/2001 1:14:12 PM
termsrv.exe c:\winnt\system32\termsrv.exe 864 10 204800 1413120 3/8/2001 1:19:24 PM 5.00.2182.1 136.77 KB (140,048 bytes) 2/23/2001 8:12:11 AM
winmgmt.exe c:\winnt\system32\wbem\winmgmt.exe 904 8 204800 1413120 3/8/2001 1:19:24 PM 1.50.1085.0001 188.05 KB (192,567 bytes) 12/7/1999 6:00:00 AM
mssearch.exe c:\program files\common files\system\mssearch\bin\mssearch.exe 920 8 204800 1413120 3/8/2001 1:19:24 PM 9.107.5512.0 72.00 KB (73,728 bytes) 7/12/2000 7:44:20 PM
dfssvc.exe c:\winnt\system32\dfssvc.exe 984 8 204800 1413120 3/8/2001 1:19:31 PM 5.00.2191.1 85.27 KB (87,312 bytes) 12/7/1999 6:00:00 AM
svchost.exe c:\winnt\system32\svchost.exe 1164 8 204800 1413120 3/8/2001 1:19:37 PM 5.00.2134.1 7.77 KB (7,952 bytes) 12/7/1999 6:00:00 AM
wordpad.exe c:\program files\windows nt\accessories\wordpad.exe 1208 8 204800 1413120 3/8/2001 1:22:56 PM 5.00.2170.1 180.77 KB (185,104 bytes) 2/23/2001 8:13:02 AM
explorer.exe c:\winnt\explorer.exe 1300 8 204800 1413120 3/8/2001 1:28:21 PM 5.00.2920.0000 232.77 KB (238,352 bytes) 12/7/1999 6:00:00 AM
cmd.exe c:\winnt\system32\cmd.exe 1304 8 204800 1413120 3/8/2001 1:28:25 PM 5.00.2144.1 230.77 KB (236,304 bytes) 12/7/1999 6:00:00 AM
sqlservr.exe c:\program files\microsoft sql server\mssql\binn\sqlservr.exe 1120 8 204800 1413120 3/8/2001 1:41:18 PM 2000.080.0194.00 7.10 MB (7,442,493 bytes) 2/23/2001 4:06:52 PM
isqlw.exe c:\program files\microsoft sql server\80\tools\binn\isqlw.exe 1340 8 204800 1413120 3/8/2001 1:41:40 PM 2000.080.0194.00 344.06 KB (352,319 bytes) 2/23/2001 4:07:08 PM
stepmaster.exe c:\program files\stepmaster\stepmaster.exe 1412 8 204800 1413120 3/8/2001 1:53:44 PM 2.03 1.77 MB (1,861,120 bytes) 2/23/2001 3:33:39 PM
wordpad.exe c:\program files\windows nt\accessories\wordpad.exe 12264 8 204800 1413120 3/8/2001 2:14:21 PM 5.00.2170.1 180.77 KB (185,104 bytes) 2/23/2001 8:13:02 AM
cmd.exe c:\winnt\system32\cmd.exe 12288 8 204800 1413120 3/8/2001 2:21:14 PM 5.00.2144.1 230.77 KB (236,304 bytes) 12/7/1999 6:00:00 AM
mmc.exe c:\winnt\system32\mmc.exe 324 8 204800 1413120 3/8/2001 3:25:37 PM 5.00.2153.1 589.27 KB (603,408 bytes) 12/7/1999 6:00:00 AM
rsvp.exe c:\winnt\system32\rsvp.exe 12456 8 204800 1413120 3/8/2001 3:26:29 PM 5.00.2167.1 172.77 KB (176,912 bytes) 12/7/1999 6:00:00 AM

[Loaded Modules]

Name Version Size File Date Manufacturer Path
traffic.dll 5.00.2139.1 30.77 KB (31,504 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\traffic.dll
rsvp.exe 5.00.2167.1 172.77 KB (176,912 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rsvp.exe
wbemprox.dll 1.50.1085.0001 40.05 KB (41,016 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\wbemprox.dll
mlang.dll 5.00.2920.0000 510.77 KB (523,024 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mlang.dll
rassapi.dll 5.00.2188.1 14.27 KB (14,608 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rassapi.dll
adsnt.dll 5.00.2191.1 194.27 KB (198,928 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\adsnt.dll
dbghelp.dll 5.00.2195.1 159.27 KB (163,088 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\dbghelp.dll
localsec.dll 5.00.2134.1 227.27 KB (232,720 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\localsec.dll
devmgr.dll 5.00.2166.1 215.77 KB (220,944 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\devmgr.dll
filemgmt.dll 5.00.2134.1 287.27 KB (294,160 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\filemgmt.dll
pdh.dll 5.00.2174.1 143.27 KB (146,704 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\pdh.dll
smlogcfg.dll 5.00.2163.1 273.27 KB (279,824 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\smlogcfg.dll
cabinet.dll 5.00.2147.1 54.77 KB (56,080 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\cabinet.dll
msinfo32.dll 5.00.2177.1 312.27 KB (319,760 bytes) 2/23/2001 1:14:21 PM Microsoft Corporation c:\program files\common files\microsoft shared\msinfo\msinfo32.dll
riched32.dll 5.00.2134.1 3.77 KB (3,856 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\riched32.dll

els.dll 5.00.2175.1 151.27 KB (154,896 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\els.dll
ntmsmgr.dll 1,0,0,1 427.77 KB (438,032 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation and HighGround Systems, Inc. c:\winnt\system32\ntmsmgr.dll
mmfutil.dll 1.50.1085.0000 32.06 KB (32,829 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mmfutil.dll
logdrive.dll 1.50.1085.0000 200.06 KB (204,863 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\logdrive.dll
dfrgres.dll 5.00.2150.1 27.50 KB (28,160 bytes) 12/7/1999 6:00:00 AM Executive Software International, Inc. c:\winnt\system32\dfrgres.dll
dfrgsnap.dll 5.00.2150.1 41.77 KB (42,768 bytes) 12/7/1999 6:00:00 AM Executive Software International, Inc. c:\winnt\system32\dfrgsnap.dll
dmdskres.dll 2191.1.296.2 119.00 KB (121,856 bytes) 12/7/1999 6:00:00 AM Microsoft Corp., VERITAS Software c:\winnt\system32\dmdskres.dll
dmutil.dll 2191.1.296.2 41.77 KB (42,768 bytes) 12/7/1999 6:00:00 AM VERITAS Software Corp. c:\winnt\system32\dmutil.dll
ntmsapi.dll 5.00.1948.1 50.27 KB (51,472 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntmsapi.dll
dmdskmgr.dll 2191.1.296.2 158.77 KB (162,576 bytes) 12/7/1999 6:00:00 AM Microsoft Corp., VERITAS Software c:\winnt\system32\dmdskmgr.dll
mycomput.dll 5.00.2134.1 107.77 KB (110,352 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mycomput.dll
mmcndmgr.dll 5.00.2178.1 815.27 KB (834,832 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mmcndmgr.dll
mmc.exe 5.00.2153.1 589.27 KB (603,408 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mmc.exe
comsvcs.dll 1999.9.3422.14 1.16 MB (1,219,856 bytes) 2/23/2001 8:12:01 AM Microsoft Corporation c:\winnt\system32\comsvcs.dll
mtxdm.dll 1999.9.3421.3 4.77 KB (4,880 bytes)2/23/2001 8:12:03 AM Microsoft Corporation c:\winnt\system32\mtxdm.dll
msrdo20.dll 6.00.8450 388.00 KB (397,312 bytes) 2/23/2001 3:33:38 PM Microsoft Corporation c:\winnt\system32\msrdo20.dll
comct232.ocx 6.00.8022 160.30 KB (164,144 bytes) 2/23/2001 3:33:38 PM Microsoft Corporation c:\winnt\system32\comct232.ocx
msflxgrd.ocx 6.00.8418 238.51 KB (244,232 bytes) 2/23/2001 3:33:38 PM Microsoft Corporation c:\winnt\system32\msflxgrd.ocx
tabctl32.ocx 6.00.8418 204.50 KB (209,408 bytes) 2/23/2001 3:33:38 PM Microsoft Corporation c:\winnt\system32\tabctl32.ocx
comctl32.ocx 6.00.8022 595.30 KB (609,584 bytes) 2/23/2001 3:33:38 PM Microsoft Corporation c:\winnt\system32\comctl32.ocx
vbajet32.dll 6.1.8268 30.27 KB (30,992 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\vbajet32.dll
msjtes40.dll 4.00.2927.8 232.27 KB (237,840 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msjtes40.dll
expsrv.dll 6.0.8540 370.27 KB (379,152 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\expsrv.dll
mswstr10.dll 4.00.2927.10 600.27 KB (614,672 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mswstr10.dll
msjet40.dll 4.00.2927.4 1.43 MB (1,495,312 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msjet40.dll
dao360.dll 03.60.2927.12 544.27 KB (557,328 bytes) 2/23/2001 1:14:03 PM Microsoft Corporation c:\program files\common files\microsoft shared\dao\dao360.dll
msvbvm50.dll 05.02.8244 (SP2) 1.29 MB (1,355,776 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msvbvm50.dll
smtime.dll 1, 0, 0, 1 40.00 KB (40,960 bytes) 2/23/2001 3:33:38 PM c:\winnt\system32\smtime.dll
stepmaster.exe 2.03 1.77 MB (1,861,120 bytes) 2/23/2001 3:33:39 PM Microsoft Corporation c:\program files\stepmaster\stepmaster.exe
dbmslpcn.dll 2000.080.0194.00 28.06 KB (28,734 bytes) 2/23/2001 4:06:54 PM Microsoft Corporation c:\winnt\system32\dbmslpcn.dll
dbnetlib.dll 2000.080.0194.00 84.06 KB (86,082 bytes) 2/23/2001 4:06:18 PM Microsoft Corporation c:\winnt\system32\dbnetlib.dll
sqllex.dll 2000.080.0194.00 148.06 KB (151,616 bytes) 2/23/2001 4:07:18 PM Microsoft Corporation c:\program files\microsoft sql server\80\tools\binn\sqllex.dll
objmgr.rll 2000.080.0194.00 56.00 KB (57,344 bytes) 2/23/2001 4:07:08 PM Microsoft Corporation c:\program files\microsoft sql server\80\tools\binn\resources\1033\objmgr.rll
objmgr.dll 2000.080.0194.00 308.06 KB (315,456 bytes) 2/23/2001 4:07:08 PM Microsoft Corporation c:\program files\microsoft sql server\80\tools\binn\objmgr.dll
odbccp32.dll 3.520.6526.0 100.27 KB (102,672 bytes) 2/23/2001 4:06:14 PM Microsoft Corporation c:\winnt\system32\odbccp32.dll
sqlsrv32.rll 2000.080.0194.00 88.00 KB (90,112 bytes) 2/23/2001 4:06:19 PM Microsoft Corporation c:\winnt\system32\sqlsrv32.rll
sqlsrv32.dll 2000.080.0194.00 460.08 KB (471,119 bytes) 2/23/2001 4:06:19 PM Microsoft Corporation c:\winnt\system32\sqlsrv32.dll
isqlw.rll 2000.080.0194.00 240.00 KB (245,760 bytes) 2/23/2001 4:07:08 PM Microsoft Corporation c:\program files\microsoft sql server\80\tools\binn\resources\1033\isqlw.rll
sqlqry.rll 2000.080.0194.00 180.00 KB (184,320 bytes) 2/23/2001 4:07:08 PM Microsoft Corporation c:\program files\microsoft sql server\80\tools\binn\resources\1033\sqlqry.rll
pfutil80.rll 2000.080.0194.00 144.00 KB (147,456 bytes) 2/23/2001 4:07:18 PM Microsoft Corporation c:\program files\microsoft sql server\80\tools\binn\resources\1033\pfutil80.rll
pfclnt80.rll 2000.080.0194.00 28.00 KB (28,672 bytes) 2/23/2001 4:07:17 PM Microsoft Corporation c:\program files\microsoft sql server\80\tools\binn\resources\1033\pfclnt80.rll
semsfc.rll 2000.080.0194.00 24.00 KB (24,576 bytes) 2/23/2001 4:07:18 PM Microsoft Corporation c:\program files\microsoft sql server\80\tools\binn\resources\1033\semsfc.rll
sqlgui.rll 2000.080.0194.00 56.00 KB (57,344 bytes) 2/23/2001 4:07:18 PM Microsoft Corporation

c:\program files\microsoft sql server\80\tools\binn\resources\1033\sqlgui.rll
sqlsvc.rll 2000.080.0194.00 24.00 KB (24,576 bytes) 2/23/2001 4:07:17 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\resources\1033\sqlsvc.rll
pfclnt80.dll 2000.080.0194.00 404.06 KB (413,762 bytes) 2/23/2001 4:07:17 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\pfclnt80.dll
semsfc.dll 2000.080.0194.00 224.06 KB (229,440 bytes) 2/23/2001 4:07:17 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\semsfc.dll
pfutil80.dll 2000.080.0194.00 268.06 KB (274,498 bytes) 2/23/2001 4:07:18 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\pfutil80.dll
sqlqry.dll 2000.080.0194.00 392.06 KB (401,472 bytes) 2/23/2001 4:07:08 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\sqlqry.dll
imm32.dll 5.00.2180.1 93.77 KB (96,016 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\imm32.dll
sqlsvc.dll 2000.080.0194.00 92.06 KB (94,272 bytes) 2/23/2001 4:07:17 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\sqlsvc.dll
w95scm.dll 2000.080.0194.00 48.06 KB (49,216 bytes) 2/23/2001 4:07:17 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\w95scm.dll
sqlgui.dll 2000.080.0194.00 444.06 KB (454,720 bytes) 2/23/2001 4:07:17 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\sqlgui.dll
sqlresld.dll 2000.080.0194.00 28.06 KB (28,738 bytes) 2/23/2001 4:07:17 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\sqlresld.dll
isqlw.exe 2000.080.0194.00 344.06 KB (352,319 bytes) 2/23/2001 4:07:08 PM Microsoft Corporation
c:\program files\microsoft sql server\80\tools\binn\isqlw.exe
xpstar.rll 2000.080.0194.00 48.00 KB (49,152 bytes) 2/23/2001 4:06:53 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\resources\1033\xpstar.rll
sqlsvc.rll 2000.080.0194.00 24.00 KB (24,576 bytes) 2/23/2001 4:07:14 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\resources\1033\sqlsvc.rll
odbcint.dll 3.520.6526.0 88.00 KB (90,112 bytes) 2/23/2001 4:06:14 PM Microsoft Corporation
c:\winnt\system32\odbcint.dll
w95scm.dll 2000.080.0194.00 48.06 KB (49,216 bytes) 2/23/2001 4:07:14 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\w95scm.dll
odbcbcp.dll 2000.080.0194.00 28.07 KB (28,742 bytes) 2/23/2001 4:06:19 PM Microsoft Corporation
c:\winnt\system32\odbcbcp.dll
odbc32.dll 3.520.6526.0 216.27 KB (221,456 bytes) 2/23/2001 4:06:14 PM Microsoft Corporation
c:\winnt\system32\odbc32.dll
sqlsvc.dll 2000.080.0194.00 92.06 KB (94,272 bytes) 2/23/2001 4:07:14 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\sqlsvc.dll
sqlresld.dll 2000.080.0194.00 28.06 KB (28,738 bytes) 2/23/2001 4:07:14 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\sqlresld.dll
sqlunirl.dll 2000.080.0194.00 176.06 KB (180,290 bytes) 8/6/2000 2:51:56 AM Microsoft Corporation
c:\winnt\system32\sqlunirl.dll
xpstar.dll 2000.080.0194.00 264.06 KB (270,400 bytes) 2/23/2001 4:06:53 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\xpstar.dll
mssws.dll 9.107.5512.0 18.94 KB (19,392 bytes) 7/12/2000 7:44:16 PM Microsoft Corporation
c:\progra~1\common~1\system\mssearch\bin\mssws.dll
srchadm.dll 9.107.5512.0 278.50 KB (285,184 bytes) 7/12/2000 7:48:50 PM Microsoft Corporation
c:\progra~1\common~1\system\mssearch\bin\srchadm.dll
oledb32r.dll 2.60.6526.0 68.27 KB (69,904 bytes) 2/23/2001 4:06:14 PM Microsoft Corporation
c:\program files\common files\system\ole db\oledb32r.dll
oledb32.dll 2.60.6526.0 448.27 KB (459,024 bytes) 2/23/2001 4:06:14 PM Microsoft Corporation
c:\program files\common files\system\ole db\oledb32.dll
msdatl3.dll 2.60.6526.0 92.27 KB (94,480 bytes) 2/23/2001 4:06:13 PM Microsoft Corporation
c:\program files\common files\system\ole db\msdatl3.dll
msdart.dll 2.60.6526.0 144.27 KB (147,728 bytes) 2/23/2001 4:06:13 PM Microsoft Corporation
c:\winnt\system32\msdart.dll
sqloledb.dll 2000.080.0194 480.06 KB (491,584 bytes) 2/23/2001 4:06:20 PM Microsoft Corporation
c:\program files\common files\system\ole db\sqloledb.dll
sqlftqry.dll 2000.080.0194.00 108.07 KB (110,668 bytes) 2/23/2001 4:06:58 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\sqlftqry.dll
ssmslpcn.dll 2000.080.0194.00 28.06 KB (28,734 bytes) 2/23/2001 4:06:53 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\ssmslpcn.dll
ssnetlib.dll 2000.080.0194.00 84.06 KB (86,078 bytes) 2/23/2001 4:06:53 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\ssnetlib.dll
ssnmpn70.dll 2000.080.0194.00 24.06 KB (24,638 bytes) 2/23/2001 4:06:53 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\ssnmpn70.dll
sqlevn70.rll 2000.080.0194.00 28.00 KB (28,672 bytes) 2/23/2001 4:06:54 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\resources\1033\sqlevn70.rll
msvcirt.dll 6.10.8637.0 76.05 KB (77,878 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\msvcirt.dll
sqlsort.dll 2000.080.0194.00 576.06 KB (589,885 bytes) 2/23/2001 4:06:53 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\sqlsort.dll
ums.dll 2000.080.0194.00 48.06 KB (49,210 bytes) 2/23/2001 4:06:53 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\ums.dll
opends60.dll 2000.080.0194.00 24.06 KB (24,639 bytes) 2/23/2001 4:06:53 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\opends60.dll
sqlservr.exe 2000.080.0194.00 7.10 MB (7,442,493 bytes) 2/23/2001 4:06:52 PM Microsoft Corporation
c:\program files\microsoft sql server\mssql\binn\sqlservr.exe
cmd.exe 5.00.2144.1 230.77 KB (236,304 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\cmd.exe
shdoclc.dll 5.00.2920.0000 324.50 KB (332,288 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\shdoclc.dll
wininet.dll 5.00.2920.0000 456.77 KB (467,728 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\wininet.dll
faxshell.dll 5.00.2134.1 8.27 KB (8,464 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\faxshell.dll
msacm32.dll 5.00.2134.1 65.27 KB (66,832 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\msacm32.dll

avifil32.dll 5.00.2134.1 76.27 KB (78,096 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\avifil32.dll
msvfw32.dll 5.00.2134.1 113.77 KB (116,496 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msvfw32.dll
docprop2.dll 5.00.2178.1 297.77 KB (304,912 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\docprop2.dll
urlmon.dll 5.00.2920.0000 426.77 KB (437,008 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\urlmon.dll
browselc.dll 5.00.2920.0000 34.50 KB (35,328 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\browselc.dll
msi.dll 1.10.1029.0 1.71 MB (1,794,320 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msi.dll
linkinfo.dll 5.00.2134.1 15.77 KB (16,144 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\linkinfo.dll
powrprof.dll 5.00.2920.0000 13.27 KB (13,584 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\powrprof.dll
batmeter.dll 5.00.2920.0000 20.27 KB (20,752 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\batmeter.dll
stobject.dll 5.00.2144.1 81.77 KB (83,728 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\stobject.dll
webcheck.dll 5.00.2920.0000 251.77 KB (257,808 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\webcheck.dll
ntshrui.dll 5.00.2134.1 46.77 KB (47,888 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntshrui.dll
mydocs.dll 5.00.2920.0000 55.77 KB (57,104 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mydocs.dll
browseui.dll 5.00.2920.0000 793.27 KB (812,304 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\browseui.dll
shdocvw.dll 5.00.2920.0000 1.05 MB (1,104,144 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\shdocvw.dll
explorer.exe 5.00.2920.0000 232.77 KB (238,352 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\explorer.exe
oledlg.dll 1.0 115.27 KB (118,032 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\oledlg.dll
riched20.dll 5.30.23.1200 421.27 KB (431,376 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\riched20.dll
wordpad.exe 5.00.2170.1 180.77 KB (185,104 bytes) 2/23/2001 8:13:02 AM Microsoft Corporation c:\program files\windows nt\accessories\wordpad.exe
tapisrv.dll 5.00.2186.1 168.77 KB (172,816 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\tapisrv.dll
dfssvc.exe 5.00.2191.1 85.27 KB (87,312 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\dfssvc.exe
athprxy.dll 9.107.5512.0 32.00 KB (32,768 bytes) 7/12/2000 7:44:14 PM Microsoft Corporation c:\winnt\system32\athprxy.dll
iprop.dll 5.00.2181.1 4.27 KB (4,368 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\iprop.dll
srchidx.dll 9.107.5512.0 433.50 KB (443,904 bytes) 7/12/2000 7:44:18 PM Microsoft Corporation c:\progra~1\common~1\system\mssearch\bin\srchidx.dll
propdefs.dll 9.107.5512.0 164.00 KB (167,936 bytes) 7/12/2000 7:44:16 PM Microsoft Corporation c:\progra~1\common~1\system\mssearch\bin\propdefs.dll
lcdetect.dll 9.107.5512.0 31.00 KB (31,744 bytes) 7/12/2000 7:44:16 PM Microsoft Corporation c:\program files\common files\system\mssearch\bin\lcdetect.dll
tquery.dll 9.107.5512.0 1.61 MB (1,690,112 bytes) 7/12/2000 7:48:52 PM Microsoft Corporation c:\program files\common files\system\mssearch\bin\tquery.dll
security.dll 5.00.2154.1 5.77 KB (5,904 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\security.dll
mssrch.dll 9.107.5512.0 1.49 MB (1,566,976 bytes) 7/12/2000 7:44:16 PM Microsoft Corporation c:\progra~1\common~1\system\mssearch\bin\mssrch.dll
mssws.dll 9.107.5512.0 18.94 KB (19,392 bytes) 7/12/2000 7:44:16 PM Microsoft Corporation c:\program files\common files\system\mssearch\bin\mssws.dll
mssearch.exe 9.107.5512.0 72.00 KB (73,728 bytes) 7/12/2000 7:44:20 PM Microsoft Corporation c:\program files\common files\system\mssearch\bin\mssearch.exe
wshnetbs.dll 5.00.2134.1 7.77 KB (7,952 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wshnetbs.dll
rapilib.dll 5.00.2167.1 25.27 KB (25,872 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rapilib.dll
rsvpsp.dll 5.00.2167.1 74.77 KB (76,560 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rsvpsp.dll
ntmarta.dll 5.00.2158.1 98.77 KB (101,136 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntmarta.dll
provthrd.dll 1.50.1085.0000 68.07 KB (69,708 bytes) 2/23/2001 1:14:11 PM Microsoft Corporation c:\winnt\system32\wbem\provthrd.dll
ntevt.dll 1.50.1085.0000 192.06 KB (196,669 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\ntevt.dll
perfos.dll 5.00.2155.1 21.27 KB (21,776 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\perfos.dll
psapi.dll 5.00.2134.1 28.27 KB (28,944 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\psapi.dll
framedyn.dll 1.50.1085.0000 164.05 KB (167,992 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\framedyn.dll
cimwin32.dll 1.50.1085.0000 1.03 MB (1,077,306 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\cimwin32.dll
wbemsvc.dll 1.50.1085.0000 140.07 KB (143,430 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\wbemsvc.dll
wbemess.dll 1.50.1085.0001 352.05 KB (360,503 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\wbemess.dll
fastprox.dll 1.50.1085.0001 144.08 KB (147,534 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\fastprox.dll

wbemcore.dll 1.50.1085.0001 632.05 KB (647,224 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\wbemcore.dll
wbemcomn.dll 1.50.1085.0001 684.05 KB (700,472 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\wbemcomn.dll
winmgmt.exe 1.50.1085.0001 188.05 KB (192,567 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wbem\winmgmt.exe
rdpwsx.dll 5.00.2180.1 94.40 KB (96,664 bytes) 2/23/2001 8:12:10 AM Microsoft Corporation c:\winnt\system32\rdpwsx.dll
ntlsapi.dll 5.00.2134.1 6.77 KB (6,928 bytes)12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntlsapi.dll
mstlsapi.dll 5.00.2181.1 24.77 KB (25,360 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mstlsapi.dll
icaapi.dll 5.00.2134.1 118.77 KB (121,616 bytes) 2/23/2001 8:12:09 AM Microsoft Corporation c:\winnt\system32\icaapi.dll
regapi.dll 5.00.2155.1 35.27 KB (36,112 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\regapi.dll
termsrv.exe 5.00.2182.1 136.77 KB (140,048 bytes) 2/23/2001 8:12:11 AM Microsoft Corporation c:\winnt\system32\termsrv.exe
msidle.dll 5.00.2920.0000 6.27 KB (6,416 bytes)12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msidle.dll
mstask.exe4.71.2137.1 115.27 KB (118,032 bytes) 2/23/2001 1:14:12 PM Microsoft Corporation c:\winnt\system32\mstask.exe
regsvc.exe 5.00.2155.1 65.27 KB (66,832 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\regsvc.exe
gamevlog.exe Not Available 187.35 KB (191,842 bytes) 2/23/2001 3:12:48 PM Not Available c:\winnt\system32\gamserv\gamevlog.exe
gamevent.exe Not Available 88.71 KB (90,834 bytes) 2/23/2001 3:12:48 PM Not Available c:\winnt\system32\gamserv\gamevent.exe
gamserv.exe Not Available 128.20 KB (131,281 bytes) 2/23/2001 3:12:48 PM Not Available c:\winnt\system32\gamserv\gamserv.exe
llsrpc.dll 5.00.2149.1 45.77 KB (46,864 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\llsrpc.dll
llssrv.exe 5.00.2167.1 114.27 KB (117,008 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\llssrv.exe
gamscm.exe Not Available 119.28 KB (122,144 bytes) 2/23/2001 3:12:48 PM Not Available c:\winnt\system32\gamserv\gamscm.exe
wmi.dll 5.00.2191.1 6.27 KB (6,416 bytes)12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wmi.dll
netshell.dll5.00.2176.1 456.77 KB (467,728 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\netshell.dll
netman.dll 5.00.2175.1 88.77 KB (90,896 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\netman.dll
ntmsdba.dll 5.00.2187.1 167.77 KB (171,792 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntmsdba.dll
rasdlg.dll 5.00.2194.1 514.27 KB (526,608 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rasdlg.dll
netcfgx.dll 5.00.2175.1 533.77 KB (546,576 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\netcfgx.dll
rasmans.dll 5.00.2188.1 146.77 KB (150,288 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rasmans.dll
sens.dll 5.00.2163.1 36.77 KB (37,648 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\sens.dll
ntmssvc.dll 5.00.2187.1 390.77 KB (400,144 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntmssvc.dll
es.dll 1999.9.3422.21 231.77 KB (237,328 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\es.dll
mtxoci.dll 1999.9.3421.3 109.27 KB (111,888 bytes) 2/23/2001 8:12:08 AM Microsoft Corporation c:\winnt\system32\mtxoci.dll
resutils.dll 5.00.2191.1 39.77 KB (40,720 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\resutils.dll
clusapi.dll 5.00.2179.1 50.27 KB (51,472 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\clusapi.dll
msvcp50.dll 5.00.7051 552.50 KB (565,760 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msvcp50.dll
xolehlp.dll 1999.9.3421.3 17.27 KB (17,680 bytes) 2/23/2001 8:12:07 AM Microsoft Corporation c:\winnt\system32\xolehlp.dll
msdtclog.dll 1999.9.3421.3 89.77 KB (91,920 bytes) 2/23/2001 8:12:07 AM Microsoft Corporation c:\winnt\system32\msdtclog.dll
mtxclu.dll 1999.9.3421.3 50.27 KB (51,472 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mtxclu.dll
msdtcprx.dll 1999.9.3422.10 619.27 KB (634,128 bytes) 2/23/2001 8:12:08 AM Microsoft Corporation c:\winnt\system32\msdtcprx.dll
txfaux.dll 1999.9.3422.24 341.27 KB (349,456 bytes) 2/23/2001 8:12:07 AM Microsoft Corporation c:\winnt\system32\txfaux.dll
msdtctm.dll 1999.9.3422.12 1.02 MB (1,070,864 bytes) 2/23/2001 8:12:08 AM Microsoft Corporation c:\winnt\system32\msdtctm.dll
msdtc.exe 1999.9.3421.3 6.77 KB (6,928 bytes)2/23/2001 8:12:07 AM Microsoft Corporation c:\winnt\system32\msdtc.exe
inetpp.dll 5.00.2161.1 63.27 KB (64,784 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\inetpp.dll
win32spl.dll 5.00.2162.1 92.27 KB (94,480 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\win32spl.dll
usbmon.dll 5.00.2165.1 11.27 KB (11,536 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\usbmon.dll
tcpmon.dll 5.00.2165.1 40.77 KB (41,744 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\tcpmon.dll
pjlmon.dll 5.00.2165.1 12.77 KB (13,072 bytes) 2/26/2001 10:36:22 AM Microsoft Corporation c:\winnt\system32\pjlmon.dll

cnbjmon.dll 5.00.2134.1 43.77 KB (44,816 bytes) 11/30/1999 6:38:48 PM Microsoft Corporation c:\winnt\system32\cnbjmon.dll
localspl.dll 5.00.2191.1 244.77 KB (250,640 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\localspl.dll
spoolss.dll 5.00.2161.1 61.77 KB (63,248 bytes) 2/23/2001 7:56:07 AM Microsoft Corporation c:\winnt\system32\spoolss.dll
spoolsv.exe 5.00.2161.1 43.77 KB (44,816 bytes) 2/23/2001 7:56:07 AM Microsoft Corporation c:\winnt\system32\spoolsv.exe
rpcss.dll 5.00.2181.1 229.27 KB (234,768 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rpcss.dll
svchost.exe 5.00.2134.1 7.77 KB (7,952 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\svchost.exe
dssbase.dll 5.00.2150.1 140.77 KB (144,144 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\dssbase.dll
oakley.dll 5.00.2174.1 420.27 KB (430,352 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\oakley.dll
mfc42u.dll 6.00.8665.0 972.05 KB (995,384 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mfc42u.dll
polagent.dll 5.00.2183.1 108.27 KB (110,864 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\polagent.dll
scecli.dll 5.00.2191.1 105.27 KB (107,792 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\scecli.dll
atl.dll 3.00.8449 57.56 KB (58,938 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\atl.dll
certcli.dll 5.00.2175.1 132.27 KB (135,440 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\certcli.dll
esent.dll 6.0.3939.6 1.07 MB (1,120,016 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\esent.dll
ntdsatq.dll 5.00.2181.1 31.27 KB (32,016 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntdsatq.dll
ntdsa.dll 5.00.2195.1 993.27 KB (1,017,104 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntdsa.dll
kdcsvc.dll 5.00.2181.1 133.77 KB (136,976 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\kdcsvc.dll
sfmapi.dll 5.00.2134.1 38.77 KB (39,696 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\sfmapi.dll
rassfm.dll 5.00.2168.1 21.27 KB (21,776 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rassfm.dll
schannel.dll 5.00.2170.1 139.77 KB (143,120 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\schannel.dll
netlogon.dll 5.00.2182.1 347.77 KB (356,112 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\netlogon.dll
msv1_0.dll 5.00.2164.1 94.77 KB (97,040 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msv1_0.dll
kerberos.dll 5.00.2181.1 196.77 KB (201,488 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\kerberos.dll
msprivs.dll 5.00.2154.1 41.50 KB (42,496 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msprivs.dll
samsrv.dll 5.00.2192.1 357.77 KB (366,352 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\samsrv.dll
lsasrv.dll 5.00.2184.1 487.77 KB (499,472 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\lsasrv.dll
lsass.exe 5.00.2184.1 32.77 KB (33,552 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\lsass.exe
wmicore.dll 5.00.2178.1 70.77 KB (72,464 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wmicore.dll
rasadhlp.dll 5.00.2168.1 7.27 KB (7,440 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rasadhlp.dll
winrnr.dll 5.00.2160.1 18.77 KB (19,216 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\winrnr.dll
rnr20.dll 5.00.2152.1 35.77 KB (36,624 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rnr20.dll
mswsock.dll 5.00.2152.1 62.27 KB (63,760 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mswsock.dll
msgsvc.dll 5.00.2181.1 33.77 KB (34,576 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msgsvc.dll
browser.dll 5.00.2142.1 48.27 KB (49,424 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\browser.dll
alrsvc.dll 5.00.2134.1 17.77 KB (18,192 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\alrsvc.dll
trkwks.dll 5.00.2166.1 88.77 KB (90,896 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\trkwks.dll
seclogon.dll 5.00.2135.1 15.77 KB (16,144 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\seclogon.dll
psbase.dll 5.00.2146.1 111.77 KB (114,448 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\psbase.dll
wshtcpip.dll 5.00.2134.1 17.27 KB (17,680 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wshtcpip.dll
msafd.dll 5.00.2153.1 54.27 KB (55,568 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msafd.dll
cryptsvc.dll 5.00.2181.1 61.77 KB (63,248 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\cryptsvc.dll
cryptdll.dll 5.00.2135.1 41.27 KB (42,256 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\cryptdll.dll
wkssvc.dll 5.00.2181.1 95.27 KB (97,552 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wkssvc.dll
srvsvc.dll 5.00.2178.1 79.27 KB (81,168 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\srvsvc.dll

cfgmgr32.dll 5.00.2134.1 16.77 KB (17,168 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\cfgmgr32.dll
dmserver.dll 2191.1.296.2 11.77 KB (12,048 bytes) 12/7/1999 6:00:00 AM VERITAS Software Corp. c:\winnt\system32\dmserver.dll
lmhsvc.dll 5.00.2134.1 9.27 KB (9,488 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\lmhsvc.dll
dnsrslvr.dll 5.00.2181.1 88.27 KB (90,384 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\dnsrslvr.dll
tapi32.dll 5.00.2182.1 123.27 KB (126,224 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\tapi32.dll
rasman.dll 5.00.2188.1 54.77 KB (56,080 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rasman.dll
rasapi32.dll 5.00.2188.1 189.77 KB (194,320 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rasapi32.dll
rtutils.dll 5.00.2168.1 43.77 KB (44,816 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rtutils.dll
adsldpc.dll 5.00.2172.1 127.77 KB (130,832 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\adsldpc.dll
activeds.dll 5.00.2172.1 172.77 KB (176,912 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\activeds.dll
mprapi.dll 5.00.2181.1 79.27 KB (81,168 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mprapi.dll
iphlpapi.dll 5.00.2173.2 67.77 KB (69,392 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\iphlpapi.dll
icmp.dll 5.00.2134.1 7.27 KB (7,440 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\icmp.dll
dhcpcsvc.dll 5.00.2153.1 88.77 KB (90,896 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\dhcpcsvc.dll
eventlog.dll 5.00.2178.1 43.77 KB (44,816 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\eventlog.dll
ntdsapi.dll 5.00.2160.1 56.27 KB (57,616 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntdsapi.dll
scesrv.dll 5.00.2188.1 225.77 KB (231,184 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\scesrv.dll
umpnpmgr.dll 5.00.2182.1 86.27 KB (88,336 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\umpnpmgr.dll
services.exe 5.00.2134.1 86.77 KB (88,848 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\services.exe
clbcatq.dll 1999.9.3422.14 479.27 KB (490,768 bytes) 2/23/2001 8:12:01 AM Microsoft Corporation c:\winnt\system32\clbcatq.dll
oleaut32.dll 2.40.4512 600.27 KB (614,672 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\oleaut32.dll
netmsg.dll 5.00.2137.1 152.50 KB (156,160 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\netmsg.dll
comdlg32.dll 5.00.2920.0000 236.77 KB (242,448 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\comdlg32.dll
netui2.dll 5.00.2134.1 280.27 KB (286,992 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\netui2.dll
mprui.dll 5.00.2134.1 54.77 KB (56,080 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mprui.dll
netui1.dll 5.00.2134.1 210.27 KB (215,312 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\netui1.dll
netui0.dll 5.00.2134.1 70.27 KB (71,952 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\netui0.dll
ntlanman.dll 5.00.2157.1 35.27 KB (36,112 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ntlanman.dll
mpr.dll 5.00.2146.1 53.27 KB (54,544 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mpr.dll
cscui.dll 5.00.2172.1 227.27 KB (232,720 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\cscui.dll
winspool.drv 5.00.2167.1 109.77 KB (112,400 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\winspool.drv
winscard.dll 5.00.2134.1 77.27 KB (79,120 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\winscard.dll
wlnotify.dll 5.00.2164.1 53.27 KB (54,544 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wlnotify.dll
cscdll.dll 5.00.2189.1 98.27 KB (100,624 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\cscdll.dll
lz32.dll 5.00.2134.1 9.77 KB (10,000 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\lz32.dll
version.dll 5.00.2134.1 15.77 KB (16,144 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\version.dll
rsabase.dll 5.00.2150.1 128.77 KB (131,856 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\rsabase.dll
mscat32.dll 5.131.2134.1 7.77 KB (7,952 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\mscat32.dll
ole32.dll 5.00.2181.1 966.27 KB (989,456 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\ole32.dll
imagehlp.dll 5.00.2195.1 125.27 KB (128,272 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\imagehlp.dll
msasn1.dll 5.00.2134.1 51.27 KB (52,496 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\msasn1.dll
crypt32.dll 5.131.2173.1 465.77 KB (476,944 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\crypt32.dll
wintrust.dll 5.131.2143.1 162.27 KB (166,160 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\wintrust.dll
setupapi.dll 5.00.2183.1 554.27 KB (567,568 bytes) 12/7/1999 6:00:00 AM Microsoft Corporation c:\winnt\system32\setupapi.dll

winmm.dll 5.00.2161.1 184.77 KB (189,200 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\winmm.dll
comctl32.dll 5.81 540.27 KB (553,232 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\comctl32.dll
shlwapi.dll 5.00.2920.0000 282.77 KB (289,552 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\shlwapi.dll
shell32.dll 5.00.2920.0000 2.24 MB (2,352,400 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\shell32.dll
msgina.dll 5.00.2191.1 309.77 KB (317,200 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\msgina.dll
winsta.dll 5.00.2134.1 36.27 KB (37,136 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\winsta.dll
wsock32.dll 5.00.2152.1 21.27 KB (21,776 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\wsock32.dll
dnsapi.dll 5.00.2181.1 129.77 KB (132,880 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\dnsapi.dll
wldap32.dll 5.00.2168.1 155.77 KB (159,504
bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\wldap32.dll
ws2help.dll 5.00.2134.1 17.77 KB (18,192 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\ws2help.dll
ws2_32.dll 5.00.2134.1 69.77 KB (71,440 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\ws2_32.dll
samlib.dll 5.00.2160.1 46.27 KB (47,376 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\samlib.dll
netrap.dll 5.00.2134.1 11.27 KB (11,536 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\netrap.dll
netapi32.dll 5.00.2194.1 302.77 KB (310,032
bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\netapi32.dll
profmap.dll 5.00.2181.1 29.27 KB (29,968 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\profmap.dll
secur32.dll 5.00.2154.1 46.77 KB (47,888 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\secur32.dll
sfc.dll 5.00.2164.1 84.27 KB (86,288 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\sfc.dll
nddeapi.dll 5.00.2137.1 15.27 KB (15,632 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\nddeapi.dll
userenv.dll 5.00.2185.1 361.27 KB (369,936 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\userenv.dll
user32.dll 5.00.2180.1 393.27 KB (402,704 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\user32.dll
gdi32.dll 5.00.2180.1 228.77 KB (234,256 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\gdi32.dll
rpcrt4.dll 5.00.2193.1 434.27 KB (444,688 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\rpcrt4.dll
advapi32.dll 5.00.2191.1 349.27 KB (357,648
bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\advapi32.dll
kernel32.dll 5.00.2191.1 715.27 KB (732,432
bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\kernel32.dll
msvcrt.dll 6.10.8637.0 288.09 KB (295,000 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\msvcrt.dll
winlogon.exe 5.00.2182.1 173.27 KB (177,424
bytes) 12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\winlogon.exe
sfcfiles.dll 5.00.2195.1 973.27 KB (996,624 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\sfcfiles.dll
ntdll.dll 5.00.2163.1 469.77 KB (481,040 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\ntdll.dll
smss.exe 5.00.2170.1 44.27 KB (45,328 bytes)
12/7/1999 6:00:00 AM Microsoft Corporation
c:\winnt\system32\smss.exe

[Services]

Display Name Name State Start Mode
Service Type Path Error Control Start Name
Tag ID
Alerter Alerter Running Auto Share Process
c:\winnt\system32\services.exe Normal LocalSystem 0
Application Management AppMgmt Stopped Manual
Share Process c:\winnt\system32\services.exe Normal
LocalSystem 0
Computer Browser Browser Running Auto Share Process
c:\winnt\system32\services.exe Normal LocalSystem 0
Indexing Service cisvc Stopped Manual Share Process
c:\winnt\system32\cisvc.exe Normal LocalSystem 0
ClipBook ClipSrv Stopped Manual Own Process
c:\winnt\system32\clipsrv.exe Normal LocalSystem 0
Distributed File System Dfs Running Auto
Own Process c:\winnt\system32\dfssvc.exe Normal
LocalSystem 0
DHCP Client Dhcp Running Auto Share Process
c:\winnt\system32\services.exe Normal LocalSystem 0
Logical Disk Manager Administrative Service dmadmin
Stopped Manual Share Process
c:\winnt\system32\dmadmin.exe /com Normal LocalSystem
0
Logical Disk Manager dmserver Running Auto
Share Process c:\winnt\system32\services.exe Normal
LocalSystem 0
DNS Client Dnscache Running Auto Share Process
c:\winnt\system32\services.exe Normal LocalSystem 0
Event Log Eventlog Running Auto Share Process
c:\winnt\system32\services.exe Normal LocalSystem 0
COM+ Event System EventSystem Running Manual
Share Process c:\winnt\system32\svchost.exe -k netsvcs
Normal LocalSystem 0
Fax Service Fax Stopped Manual Own Process
c:\winnt\system32\faxsvc.exe Normal LocalSystem 0
Mylex Global Array Manager Server gamscm Running
Auto Own Process
c:\winnt\system32\gamserv\gamscm.exe Normal LocalSystem
0
Intersite Messaging IsmServ Stopped Disabled Own Process
c:\winnt\system32\ismserv.exe Normal LocalSystem 0
Kerberos Key Distribution Center kdc Stopped
Disabled Share Process c:\winnt\system32\lsass.exe
Normal LocalSystem 0

Server lanmanserver Running Auto Share Process c:\winnt\system32\services.exe Normal LocalSystem 0
Workstation lanmanworkstation Running Auto Share Process c:\winnt\system32\services.exe Normal LocalSystem 0
License Logging Service LicenseService Running Auto Own Process c:\winnt\system32\llssrv.exe Normal LocalSystem 0
TCP/IP NetBIOS Helper Service LmHosts Running Auto Share Process c:\winnt\system32\services.exe Normal LocalSystem 0
Messenger Messenger Running Auto Share Process c:\winnt\system32\services.exe Normal LocalSystem 0
NetMeeting Remote Desktop Sharing mnmsrvc Stopped Manual Own Process c:\winnt\system32\mnmsrvc.exe Normal LocalSystem 0
Distributed Transaction Coordinator MSDTC Running Auto Own Process c:\winnt\system32\msdtc.exe Normal LocalSystem 0
Windows Installer MSIServer Stopped Manual Share Process c:\winnt\system32\msiexec.exe /v Normal LocalSystem 0
Microsoft Search MSSEARCH Running Auto Share Process "c:\program files\common files\system\mssearch\bin\mssearch.exe" Normal LocalSystem 0
MSSQLSERVER MSSQLSERVER Stopped Manual Own Process c:\progra~1\micros~2\mssql\binn\sqlservr.exe Normal LocalSystem 0
MSSQLServerADHelper MSSQLServerADHelper Stopped Manual Own Process c:\program files\microsoft sql server\80\tools\binn\sqladhlp.exe Normal LocalSystem 0
Network DDE NetDDE Stopped Manual Share Process c:\winnt\system32\netdde.exe Normal LocalSystem 0
Network DDE DSDM NetDDEdsdm Stopped Manual Share Process c:\winnt\system32\netdde.exe Normal LocalSystem 0
Net Logon Netlogon Stopped Manual Share Process c:\winnt\system32\lsass.exe Normal LocalSystem 0
Network Connections Netman Running Manual Share Process c:\winnt\system32\svchost.exe -k netsvcs Normal LocalSystem 0
File Replication NtFrs Stopped Manual Own Process c:\winnt\system32\ntfrs.exe Ignore LocalSystem 0
NT LM Security Support Provider NtLmSsp Running Manual Share Process c:\winnt\system32\lsass.exe Normal LocalSystem 0
Removable Storage NtmsSvc Running Auto Share Process c:\winnt\system32\svchost.exe -k netsvcs Normal LocalSystem 0
Plug and Play PlugPlay Running Auto Share Process c:\winnt\system32\services.exe Normal LocalSystem 0
IPSEC Policy Agent PolicyAgent Running Auto Share Process c:\winnt\system32\lsass.exe Normal LocalSystem 0
Protected Storage ProtectedStorage Running Auto Share Process c:\winnt\system32\services.exe Normal LocalSystem 0
Remote Access Auto Connection Manager RasAuto Stopped Manual Share Process c:\winnt\system32\svchost.exe -k netsvcs Normal LocalSystem 0
Remote Access Connection Manager RasMan Stopped Manual Share Process c:\winnt\system32\svchost.exe -k netsvcs Normal LocalSystem 0
Routing and Remote Access RemoteAccess Stopped Disabled Share Process c:\winnt\system32\svchost.exe -k netsvcs Normal LocalSystem 0
Remote Registry Service RemoteRegistry Running Auto Own Process c:\winnt\system32\regsvc.exe Normal LocalSystem 0
Remote Procedure Call (RPC) Locator RpcLocator Stopped Manual Own Process c:\winnt\system32\locator.exe Normal LocalSystem 0
Remote Procedure Call (RPC) RpcSs Running Auto Share Process c:\winnt\system32\svchost -k rpcss Normal LocalSystem 0
QoS RSVP RSVP Running Manual Own Process c:\winnt\system32\rsvp.exe -s Normal LocalSystem 0
Security Accounts Manager SamSs Running Auto Share Process c:\winnt\system32\lsass.exe Normal LocalSystem 0
Smart Card Helper SCardDrv Stopped Manual Share Process c:\winnt\system32\scardsvr.exe Ignore LocalSystem 0
Smart Card SCardSvr Stopped Manual Share Process c:\winnt\system32\scardsvr.exe Ignore LocalSystem 0
Task Scheduler Schedule Running Auto Share Process c:\winnt\system32\mstask.exe Normal LocalSystem 0
RunAs Service seclogon Running Auto Share Process c:\winnt\system32\services.exe Ignore LocalSystem 0
System Event Notification SENS Running Auto Share Process c:\winnt\system32\svchost.exe -k netsvcs Normal LocalSystem 0
Internet Connection Sharing SharedAccess Stopped Manual Share Process c:\winnt\system32\svchost.exe -k netsvcs Normal LocalSystem 0
Print Spooler Spooler Running Auto Own Process c:\winnt\system32\spoolsv.exe Normal LocalSystem 0
SQLSERVERAGENT SQLSERVERAGENT Stopped Manual Own Process c:\progra~1\micros~2\mssql\binn\sqlagent.exe Normal LocalSystem 0
Performance Logs and Alerts SysmonLog Stopped Manual Own Process c:\winnt\system32\smlogsvc.exe Normal LocalSystem 0
Telephony TapiSrv Running Manual Share Process c:\winnt\system32\svchost.exe -k tapisrv Normal LocalSystem 0
Terminal Services TermService Running Auto Own Process c:\winnt\system32\termsrv.exe Normal LocalSystem 0
Telnet TlntSvr Stopped Manual Own Process c:\winnt\system32\tlntsvr.exe Normal LocalSystem 0
Distributed Link Tracking Server TrkSvr Stopped Manual Share Process c:\winnt\system32\services.exe Normal LocalSystem 0
Distributed Link Tracking Client TrkWks Running Auto Share Process c:\winnt\system32\services.exe Normal LocalSystem 0
Uninterruptible Power Supply UPS Stopped Manual Own Process c:\winnt\system32\ups.exe Normal LocalSystem 0
Utility Manager UtilMan Stopped Manual Own Process c:\winnt\system32\utilman.exe Normal LocalSystem 0
Windows Time W32Time Stopped Manual Share Process c:\winnt\system32\services.exe Normal LocalSystem 0
Windows Management Instrumentation WinMgmt Running Auto Own Process c:\winnt\system32\wbem\winmgmt.exe Ignore LocalSystem 0
Windows Management Instrumentation Driver Extensions Wmi Running Manual Share Process c:\winnt\system32\services.exe Normal LocalSystem 0

[Program Groups]

Group Name Name User Name
Accessories Default User:Accessories Default User
Accessories\Accessibility Default User:Accessories\Accessibility Default User
Accessories\Entertainment Default User:Accessories\Entertainment Default User
Accessories\System Tools Default User:Accessories\System Tools Default User
Startup Default User:Startup Default User
Accessories All Users:Accessories All Users
Accessories\Accessibility All Users:Accessories\Accessibility All Users
Accessories\Communications All Users:Accessories\Communications All Users
Accessories\Entertainment All Users:Accessories\Entertainment All Users
Accessories\Games All Users:Accessories\Games All Users
Accessories\System Tools All Users:Accessories\System Tools All Users
Administrative Tools All Users:Administrative Tools All Users
Microsoft SQL Server All Users:Microsoft SQL Server All Users
Microsoft SQL Server - Switch All Users:Microsoft SQL Server - Switch All Users
Startup All Users:Startup All Users
Accessories TPCHM\Administrator:Accessories TPCHM\Administrator
Accessories\Accessibility TPCHM\Administrator:Accessories\Accessibility TPCHM\Administrator
Accessories\Entertainment TPCHM\Administrator:Accessories\Entertainment TPCHM\Administrator
Accessories\System Tools TPCHM\Administrator:Accessories\System Tools TPCHM\Administrator
Administrative Tools TPCHM\Administrator:Administrative Tools TPCHM\Administrator
Startup TPCHM\Administrator:Startup TPCHM\Administrator

[Startup Programs]

Program Command User Name Location
No startup program information

[OLE Registration]

Object Local Server
Sound (OLE2) sndrec32.exe
Media Clip mplay32.exe
Video Clip mplay32.exe /avi
MIDI Sequence mplay32.exe /mid
Sound Not Available
Media Clip Not Available
Image Document "C:\Program Files\Windows NT\Accessories\ImageVue\KodakImg.exe"
WordPad Document "%ProgramFiles%\Windows NT\Accessories\WORDPAD.EXE"
Windows Media Services DRM Storage object Not Available
Bitmap Image mspaint.exe

[Internet Explorer 5]

[Following are sub-categories of this main category]

[Summary]

Item Value
Version 5.00.2920.0000
Build 52920
Product ID 51879-270-9329847-05790
Application Path C:\Program Files\Internet Explorer
Language English (United States)
Active Printer Not Available

Cipher Strength 56-bit
Content Advisor Disabled
IEAK Install No

[File Versions]

File Version Size Date Path Company
advapi32.dll 5.0.2191.1 349 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
advpack.dll 5.0.2920.0 87 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
browselc.dll 5.0.2920.0 35 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
browseui.dll 5.0.2920.0 793 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
ckcnv.exe 5.0.2189.1 9 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
comctl32.dll 5.81.2920.0 540 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
crypt32.dll 5.131.2173.1 466 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
enhsig.dll <File Missing> Not Available Not Available Not Available Not Available
iemigrat.dll <File Missing> Not Available Not Available Not Available Not Available
iesetup.dll 5.0.2920.0 57 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
iexplore.exe 5.0.2920.0 59 KB 12/7/1999 6:00:00 AM C:\Program Files\Internet Explorer Microsoft Corporation
imagehlp.dll 5.0.2195.1 125 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
imghelp.dll <File Missing> Not Available Not Available Not Available Not Available
inseng.dll 5.0.2920.0 72 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
jobexec.dll 5.0.0.1 47 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
jscript.dll 5.1.0.4615 476 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
jsproxy.dll 5.0.2920.0 13 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
msaahtml.dll <File Missing> Not Available Not Available Not Available Not Available
mshtml.dll 5.0.2920.0 2302 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
msjava.dll 5.0.3234.0 918 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
msoss.dll <File Missing> Not Available Not Available Not Available Not Available
msxml.dll 5.0.2920.0 521 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
occache.dll 5.0.2920.0 86 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
ole32.dll 5.0.2181.1 966 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
oleaut32.dll 2.40.4512.1 600 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
olepro32.dll 5.0.4512.1 160 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
rsabase.dll 5.0.2150.1 129 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation

rsaenh.dll <File Missing> Not Available Not Available Not Available Not Available
rsapi32.dll <File Missing> Not Available Not Available Not Available Not Available
rsasig.dll <File Missing> Not Available Not Available Not Available Not Available
schannel.dll 5.0.2170.0 140 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
shdoc401.dll <File Missing> Not Available Not Available Not Available Not Available
shdocvw.dll 5.0.2920.0 1078 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
shell32.dll 5.0.2920.0 2297 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
shlwapi.dll 5.0.2920.0 283 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
url.dll 5.0.2920.0 82 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
urlmon.dll 5.0.2920.0 427 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
vbscript.dll 5.1.0.4615 428 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
webcheck.dll 5.0.2920.0 252 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
win.com 5.0.2134.1 24 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
wininet.dll 5.0.2920.0 457 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
winsock.dll 3.10.0.103 3 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
wintrust.dll 5.131.2143.1 162 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
wsock.vxd <File Missing> Not Available Not Available Not Available Not Available
wsock32.dll 5.0.2152.1 21 KB 12/7/1999 6:00:00 AM C:\WINNT\system32 Microsoft Corporation
wsock32n.dll <File Missing> Not Available Not Available Not Available Not Available

[Connectivity]

Item Value
Connection Preference Never dial
EnableHttp1.1 1
ProxyHttp1.1 0

LAN Settings

AutoConfigProxy wininet.dll
AutoProxyDetectMode Enabled
AutoConfigURL
Proxy Disabled
ProxyServer
ProxyOverride

[Cache]

[Following are sub-categories of this main category]

[Summary]

Item Value
Page Refresh Type Automatic
Temporary Internet Files Folder C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files
Total Disk Space 2996 MB
Available Disk Space 2040 MB
Maximum Cache Size 93 MB
Available Cache Size 94 MB

[List of Objects]

Program File Status CodeBase
No cached object information available

[Content]

[Following are sub-categories of this main category]

[Summary]

Item Value
Content Advisor Disabled

[Personal Certificates]

Issued To Issued By Validity Signature Algorithm
Administrator Administrator 2/23/2001 to 1/30/2101 sha1RSA

[Other People Certificates]

Issued To Issued By Validity Signature Algorithm
No other people certificate information available

[Publishers]

Name
No publisher information available

[Security]

Zone Security Level
Local intranet Medium-low
Trusted sites Low
Internet Medium
Restricted sites High

Disk Controller Configuration Parameters

Mylex eXtremeRAID 2000 Adapter 1

Begin

BeginGroup

PhysicalDevice0 = Channel=0, Target=0, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice1 = Channel=1, Target=9, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice2 = Channel=2, Target=0, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice3 = Channel=3, Target=8, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice4 = Channel=0, Target=1, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice5 = Channel=1, Target=8, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice6 = Channel=2, Target=1, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice7 = Channel=3, Target=9, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice8 = Channel=0, Target=2, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice9 = Channel=1, Target=10, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice10 = Channel=2, Target=2, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice11 = Channel=3, Target=10, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice12 = Channel=0, Target=3, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice13 = Channel=1, Target=11, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

LogicalDevice0 = StripeSize=64kb, Raid=0, WriteThrough=1, Size=121184mb, BIOSGeometry=8GB,

(PhysicalDevice0, StartAddress=0mb, Size=8656mb),

(PhysicalDevice1, StartAddress=0mb, Size=8656mb),

(PhysicalDevice2, StartAddress=0mb, Size=8656mb),

(PhysicalDevice3, StartAddress=0mb, Size=8656mb),

(PhysicalDevice4, StartAddress=0mb, Size=8656mb),

(PhysicalDevice5, StartAddress=0mb, Size=8656mb),

(PhysicalDevice6, StartAddress=0mb, Size=8656mb),

(PhysicalDevice7, StartAddress=0mb, Size=8656mb),

(PhysicalDevice8, StartAddress=0mb, Size=8656mb),

(PhysicalDevice9, StartAddress=0mb, Size=8656mb),

(PhysicalDevice10, StartAddress=0mb, Size=8656mb),

(PhysicalDevice11, StartAddress=0mb, Size=8656mb),

(PhysicalDevice12, StartAddress=0mb, Size=8656mb),

(PhysicalDevice13, StartAddress=0mb, Size=8656mb);

PhysicalDevice14 = Channel=2, Target=3, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice15 = Channel=3, Target=11, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice16 = Channel=0, Target=4, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice17 = Channel=1, Target=12, Size=8656mb, State=Online,

```
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice18 = Channel=2, Target=4, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice19 = Channel=3, Target=12, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice20 = Channel=0, Target=5, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice21 = Channel=1, Target=13, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice22 = Channel=2, Target=5, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice23 = Channel=3, Target=13, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice24 = Channel=0, Target=6, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice25 = Channel=1, Target=14, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice26 = Channel=2, Target=6, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice27 = Channel=3, Target=14, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        LogicalDevice1 = StripeSize=64kb, Raid=0,
WriteThrough=1, Size=121184mb, BIOSGeometry=8GB,

                (PhysicalDevice14, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice15, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice16, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice17, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice18, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice19, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice20, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice21, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice22, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice23, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice24, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice25, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice26, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice27, StartAddress=0mb,
Size=8656mb);
 EndGroup

 BeginControllerParameter

        ControllerName = eXtremeRAID 2000;

        ControllerType = 28;

        FirmwareVersion = 5.60;

        CacheLineSize = 8KB;

        BackgroundTaskRate = 50;

        InitiatorID = 7;

        DiskStartupMode = AutoSpin;

        DevicesPerSpin = 2;

        InitialDelay = 6S;

        SequentialDelay = 0S;

        EnableDriveSizing = 0;

        EnableClustering = 0;
```

EnableBGInit = 0;

EnableReadAhead = 0;

EnableBiosLoadDelay = 0;

EnableForcedUnitAccess = 1;

DisableBios = 0;

EnableCDROMBoot = 0;

EnableStorageWorks = 0;

EnableSAFTE = 0;

EnableSES = 0;

EnableARM = 1;

EnableOFM = 1;

OEMCode = 0;

StartupOption = 0;

EndControllerParameter

End

Mylex eXtremeRAID 2000 Adapter 2

Begin

BeginGroup

PhysicalDevice0 = Channel=0, Target=1, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice1 = Channel=1, Target=8, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice2 = Channel=2, Target=0, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice3 = Channel=3, Target=8, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice4 = Channel=0, Target=0, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice5 = Channel=1, Target=9, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice6 = Channel=2, Target=1, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice7 = Channel=3, Target=9, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice8 = Channel=0, Target=2, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice9 = Channel=1, Target=10, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice10 = Channel=2, Target=2, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice11 = Channel=3, Target=10, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice12 = Channel=0, Target=3, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice13 = Channel=1, Target=11, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

LogicalDevice0 = StripeSize=64kb, Raid=0, WriteThrough=1, Size=121184mb, BIOSGeometry=8GB,

(PhysicalDevice0, StartAddress=0mb, Size=8656mb),

(PhysicalDevice1, StartAddress=0mb, Size=8656mb),

(PhysicalDevice2, StartAddress=0mb, Size=8656mb),

(PhysicalDevice3, StartAddress=0mb, Size=8656mb),

(PhysicalDevice4, StartAddress=0mb, Size=8656mb),

(PhysicalDevice5, StartAddress=0mb, Size=8656mb),

(PhysicalDevice6, StartAddress=0mb, Size=8656mb),

(PhysicalDevice7, StartAddress=0mb, Size=8656mb),

(PhysicalDevice8, StartAddress=0mb, Size=8656mb),

(PhysicalDevice9, StartAddress=0mb, Size=8656mb),

(PhysicalDevice10, StartAddress=0mb, Size=8656mb),

(PhysicalDevice11, StartAddress=0mb, Size=8656mb),

(PhysicalDevice12, StartAddress=0mb, Size=8656mb),

(PhysicalDevice13, StartAddress=0mb, Size=8656mb);

PhysicalDevice14 = Channel=2, Target=3, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice15 = Channel=3, Target=11, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice16 = Channel=0, Target=4, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice17 = Channel=1, Target=12, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice18 = Channel=2, Target=4, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice19 = Channel=3, Target=12, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice20 = Channel=0, Target=5, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice21 = Channel=1, Target=13, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice22 = Channel=2, Target=5, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice23 = Channel=3, Target=13, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice24 = Channel=0, Target=6, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice25 = Channel=1, Target=14, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice26 = Channel=2, Target=6, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice27 = Channel=3, Target=14, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

LogicalDevice1 = StripeSize=64kb, Raid=0, WriteThrough=1, Size=121184mb, BIOSGeometry=8GB,

(PhysicalDevice14, StartAddress=0mb, Size=8656mb),

(PhysicalDevice15, StartAddress=0mb, Size=8656mb),

(PhysicalDevice16, StartAddress=0mb, Size=8656mb),

(PhysicalDevice17, StartAddress=0mb, Size=8656mb),

(PhysicalDevice18, StartAddress=0mb, Size=8656mb),

(PhysicalDevice19, StartAddress=0mb, Size=8656mb),

(PhysicalDevice20, StartAddress=0mb, Size=8656mb),

(PhysicalDevice21, StartAddress=0mb, Size=8656mb),

(PhysicalDevice22, StartAddress=0mb, Size=8656mb),

(PhysicalDevice23, StartAddress=0mb, Size=8656mb),

(PhysicalDevice24, StartAddress=0mb, Size=8656mb),

(PhysicalDevice25, StartAddress=0mb, Size=8656mb),

(PhysicalDevice26, StartAddress=0mb, Size=8656mb),

(PhysicalDevice27, StartAddress=0mb, Size=8656mb);

EndGroup

BeginControllerParameter

ControllerName = eXtremeRAID 2000;

ControllerType = 28;

FirmwareVersion = 5.60;

CacheLineSize = 8KB;

BackgroundTaskRate = 50;

InitiatorID = 7;

DiskStartupMode = AutoSpin;

DevicesPerSpin = 2;

InitialDelay = 6S;

SequentialDelay = 0S;

EnableDriveSizing = 0;

EnableClustering = 0;

EnableBGInit = 0;

EnableReadAhead = 0;

EnableBiosLoadDelay = 0;

EnableForcedUnitAccess = 1;

DisableBios = 0;

EnableCDROMBoot = 0;

EnableStorageWorks = 0;

EnableSAFTE = 0;

EnableSES = 0;

EnableARM = 1;

EnableOFM = 1;

OEMCode = 0;

StartupOption = 0;

EndControllerParameter

End

Mylex eXtremeRAID 2000 Adapter 3

Begin

BeginGroup

PhysicalDevice0 = Channel=0, Target=0, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice1 = Channel=1, Target=8, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice2 = Channel=0, Target=1, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice3 = Channel=1, Target=9, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice4 = Channel=0, Target=2, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice5 = Channel=1, Target=10, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

IntermediateDevice0 = StripeSize=64kb, Raid=1, WriteThrough=1, Size=8656mb,

(PhysicalDevice0, StartAddress=0mb, Size=8656mb),

(PhysicalDevice1, StartAddress=0mb, Size=8656mb);

```
        IntermediateDevice1 = StripeSize=64kb, Raid=1,
WriteThrough=1, Size=8656mb,

                (PhysicalDevice2, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice3, StartAddress=0mb,
Size=8656mb);

        IntermediateDevice2 = StripeSize=64kb, Raid=1,
WriteThrough=1, Size=8656mb,

                (PhysicalDevice4, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice5, StartAddress=0mb,
Size=8656mb);

        LogicalDevice0 = StripeSize=64kb, Raid=12,
WriteThrough=1, Size=25968mb, BIOSGeometry=8GB,

                (IntermediateDevice0, StartAddress=0mb,
Size=8656mb),

                (IntermediateDevice1, StartAddress=0mb,
Size=8656mb),

                (IntermediateDevice2, StartAddress=0mb,
Size=8656mb);

        PhysicalDevice6 = Channel=0, Target=3, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice7 = Channel=1, Target=11, Size=8656mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        LogicalDevice1 = StripeSize=64kb, Raid=0,
WriteThrough=1, Size=17312mb, BIOSGeometry=8GB,

                (PhysicalDevice6, StartAddress=0mb,
Size=8656mb),

                (PhysicalDevice7, StartAddress=0mb,
Size=8656mb);

        PhysicalDevice8 = Channel=2, Target=0, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice9 = Channel=3, Target=0, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice10 = Channel=2, Target=1, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice11 = Channel=3, Target=1, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice12 = Channel=2, Target=2, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice13 = Channel=3, Target=2, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice14 = Channel=2, Target=3, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice15 = Channel=3, Target=3, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice16 = Channel=2, Target=4, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        PhysicalDevice17 = Channel=3, Target=4, Size=17336mb,
State=Online,

                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;

        LogicalDevice2 = StripeSize=64kb, Raid=0,
WriteThrough=1, Size=173360mb, BIOSGeometry=8GB,

                (PhysicalDevice8, StartAddress=0mb,
Size=17336mb),

                (PhysicalDevice9, StartAddress=0mb,
Size=17336mb),

                (PhysicalDevice10, StartAddress=0mb,
Size=17336mb),

                (PhysicalDevice11, StartAddress=0mb,
Size=17336mb),

                (PhysicalDevice12, StartAddress=0mb,
Size=17336mb),

                (PhysicalDevice13, StartAddress=0mb,
Size=17336mb),

                (PhysicalDevice14, StartAddress=0mb,
Size=17336mb),
```

(PhysicalDevice15, StartAddress=0mb, Size=17336mb),

(PhysicalDevice16, StartAddress=0mb, Size=17336mb),

(PhysicalDevice17, StartAddress=0mb, Size=17336mb);

EndGroup

BeginControllerParameter

ControllerName = eXtremeRAID 2000;

ControllerType = 28;

FirmwareVersion = 5.60;

CacheLineSize = 8KB;

BackgroundTaskRate = 50;

InitiatorID = 7;

DiskStartupMode = AutoSpin;

DevicesPerSpin = 2;

InitialDelay = 6S;

SequentialDelay = 0S;

EnableDriveSizing = 0;

EnableClustering = 0;

EnableBGInit = 1;

EnableReadAhead = 0;

EnableBiosLoadDelay = 0;

EnableForcedUnitAccess = 1;

DisableBios = 0;

EnableCDROMBoot = 0;

EnableStorageWorks = 0;

EnableSAFTE = 0;

EnableSES = 0;

EnableARM = 0;

EnableOFM = 1;

OEMCode = 0;

StartupOption = 0;

EndControllerParameter

End

Mylex eXtremeRAID 2000 Adapter 4

Begin

BeginGroup

PhysicalDevice0 = Channel=0, Target=0, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice1 = Channel=1, Target=8, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice2 = Channel=2, Target=0, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice3 = Channel=3, Target=8, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice4 = Channel=0, Target=1, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice5 = Channel=1, Target=9, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice6 = Channel=2, Target=1, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice7 = Channel=3, Target=9, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice8 = Channel=0, Target=2, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice9 = Channel=1, Target=10, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice10 = Channel=2, Target=2, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice11 = Channel=3, Target=10, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice12 = Channel=0, Target=3, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice13 = Channel=1, Target=11, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

LogicalDevice0 = StripeSize=64kb, Raid=0, WriteThrough=1, Size=121184mb, BIOSGeometry=8GB,

(PhysicalDevice0, StartAddress=0mb, Size=8656mb),

(PhysicalDevice1, StartAddress=0mb, Size=8656mb),

(PhysicalDevice2, StartAddress=0mb, Size=8656mb),

(PhysicalDevice3, StartAddress=0mb, Size=8656mb),

(PhysicalDevice4, StartAddress=0mb, Size=8656mb),

(PhysicalDevice5, StartAddress=0mb, Size=8656mb),

(PhysicalDevice6, StartAddress=0mb, Size=8656mb),

(PhysicalDevice7, StartAddress=0mb, Size=8656mb),

(PhysicalDevice8, StartAddress=0mb, Size=8656mb),

(PhysicalDevice9, StartAddress=0mb, Size=8656mb),

(PhysicalDevice10, StartAddress=0mb, Size=8656mb),

(PhysicalDevice11, StartAddress=0mb, Size=8656mb),

(PhysicalDevice12, StartAddress=0mb, Size=8656mb),

(PhysicalDevice13, StartAddress=0mb, Size=8656mb);

PhysicalDevice14 = Channel=2, Target=3, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice15 = Channel=3, Target=11, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice16 = Channel=0, Target=4, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice17 = Channel=1, Target=12, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice18 = Channel=2, Target=4, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice19 = Channel=3, Target=12, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice20 = Channel=0, Target=5, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice21 = Channel=1, Target=13, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice22 = Channel=2, Target=5, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice23 = Channel=3, Target=13, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice24 = Channel=0, Target=6, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice25 = Channel=1, Target=14, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice26 = Channel=2, Target=6, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice27 = Channel=3, Target=14, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

LogicalDevice1 = StripeSize=64kb, Raid=0, WriteThrough=1, Size=121184mb, BIOSGeometry=8GB,

(PhysicalDevice14, StartAddress=0mb, Size=8656mb),

(PhysicalDevice15, StartAddress=0mb, Size=8656mb),

(PhysicalDevice16, StartAddress=0mb, Size=8656mb),

(PhysicalDevice17, StartAddress=0mb, Size=8656mb),

(PhysicalDevice18, StartAddress=0mb, Size=8656mb),

(PhysicalDevice19, StartAddress=0mb, Size=8656mb),

(PhysicalDevice20, StartAddress=0mb, Size=8656mb),

(PhysicalDevice21, StartAddress=0mb, Size=8656mb),

(PhysicalDevice22, StartAddress=0mb, Size=8656mb),

(PhysicalDevice23, StartAddress=0mb, Size=8656mb),

(PhysicalDevice24, StartAddress=0mb, Size=8656mb),

(PhysicalDevice25, StartAddress=0mb, Size=8656mb),

(PhysicalDevice26, StartAddress=0mb, Size=8656mb),

(PhysicalDevice27, StartAddress=0mb, Size=8656mb);

EndGroup

BeginControllerParameter

ControllerName = eXtremeRAID 2000;

ControllerType = 28;

FirmwareVersion = 5.60;

CacheLineSize = 8KB;

BackgroundTaskRate = 50;

InitiatorID = 7;

DiskStartupMode = AutoSpin;

DevicesPerSpin = 2;

InitialDelay = 6S;

SequentialDelay = 0S;

EnableDriveSizing = 0;

EnableClustering = 0;

EnableBGInit = 0;

EnableReadAhead = 0;

EnableBiosLoadDelay = 0;

EnableForcedUnitAccess = 1;

DisableBios = 0;

EnableCDROMBoot = 0;

EnableStorageWorks = 0;

EnableSAFTE = 0;

EnableSES = 0;

EnableARM = 1;

EnableOFM = 1;

OEMCode = 0;

StartupOption = 0;

EndControllerParameter

End

Mylex eXtremeRAID 2000 Adapter 5

Begin

BeginGroup

PhysicalDevice0 = Channel=0, Target=0, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice1 = Channel=1, Target=8, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice2 = Channel=2, Target=0, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice3 = Channel=3, Target=8, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice4 = Channel=0, Target=1, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice5 = Channel=1, Target=9, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice6 = Channel=2, Target=1, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice7 = Channel=3, Target=9, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice8 = Channel=0, Target=2, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice9 = Channel=1, Target=10, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice10 = Channel=2, Target=2, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice11 = Channel=3, Target=10, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice12 = Channel=0, Target=3, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice13 = Channel=1, Target=11, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

LogicalDevice0 = StripeSize=64kb, Raid=0, WriteThrough=1, Size=121184mb, BIOSGeometry=8GB,

(PhysicalDevice0, StartAddress=0mb, Size=8656mb),

(PhysicalDevice1, StartAddress=0mb, Size=8656mb),

(PhysicalDevice2, StartAddress=0mb, Size=8656mb),

(PhysicalDevice3, StartAddress=0mb, Size=8656mb),

(PhysicalDevice4, StartAddress=0mb, Size=8656mb),

(PhysicalDevice5, StartAddress=0mb, Size=8656mb),

(PhysicalDevice6, StartAddress=0mb, Size=8656mb),

(PhysicalDevice7, StartAddress=0mb, Size=8656mb),

(PhysicalDevice8, StartAddress=0mb, Size=8656mb),

(PhysicalDevice9, StartAddress=0mb, Size=8656mb),

(PhysicalDevice10, StartAddress=0mb, Size=8656mb),

(PhysicalDevice11, StartAddress=0mb, Size=8656mb),

(PhysicalDevice12, StartAddress=0mb, Size=8656mb),

(PhysicalDevice13, StartAddress=0mb, Size=8656mb);

PhysicalDevice14 = Channel=2, Target=3, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

PhysicalDevice15 = Channel=3, Target=11, Size=8656mb, State=Online,

TransferSpeed=80MHz, TransferWidth=16Bit, MaxTag=16;

```
        PhysicalDevice16 = Channel=0, Target=4, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice17 = Channel=1, Target=12, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice18 = Channel=2, Target=4, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice19 = Channel=3, Target=12, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice20 = Channel=0, Target=5, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice21 = Channel=1, Target=13, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice22 = Channel=2, Target=5, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice23 = Channel=3, Target=13, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice24 = Channel=0, Target=6, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice25 = Channel=1, Target=14, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice26 = Channel=2, Target=6, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        PhysicalDevice27 = Channel=3, Target=14, Size=8656mb,
State=Online,
                TransferSpeed=80MHz, TransferWidth=16Bit,
MaxTag=16;
        LogicalDevice1 = StripeSize=64kb, Raid=0,
WriteThrough=1, Size=121184mb, BIOSGeometry=8GB,
                (PhysicalDevice14, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice15, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice16, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice17, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice18, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice19, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice20, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice21, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice22, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice23, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice24, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice25, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice26, StartAddress=0mb,
Size=8656mb),
                (PhysicalDevice27, StartAddress=0mb,
Size=8656mb);
 EndGroup
 BeginControllerParameter
        ControllerName = eXtremeRAID 2000;
        ControllerType = 28;
        FirmwareVersion = 5.60;
        CacheLineSize = 8KB;
        BackgroundTaskRate = 50;
        InitiatorID = 7;
        DiskStartupMode = AutoSpin;
        DevicesPerSpin = 2;
```

```
        InitialDelay = 6S;
        SequentialDelay = 0S;
        EnableDriveSizing = 0;
        EnableClustering = 0;
        EnableBGInit = 0;
        EnableReadAhead = 0;
        EnableBiosLoadDelay = 0;
        EnableForcedUnitAccess = 1;
        DisableBios = 0;
        EnableCDROMBoot = 0;
        EnableStorageWorks = 0;
        EnableSAFTE = 0;
        EnableSES = 0;
        EnableARM = 1;
        EnableOFM = 1;
        OEMCode = 0;
        StartupOption = 0;
 EndControllerParameter
End
```

Appendix B: Database, Table, Index Creation and Backup, Restore, TempDB and TempLog Scripts

CreateDatabase.sql (Test Database)

```
-- CreateDatabase
-- for use with StepMaster
-- Uses FileGroups

--
--          Drop the existing database
--

if exists (select name from sysdatabases where name = 'tpch100g')
        drop database tpch100g

CREATE DATABASE tpch100g
ON PRIMARY
(       NAME                    = tpch100g_root,
        FILENAME = "d:\tpch100g_root.mdf",
        SIZE                    = 15MB,
        FILEGROWTH              = 0),
FILEGROUP       LINEITEM_FG
(       NAME                    = tpch100g_li_1,
        FILENAME = "d:\devjp\lineitem_table_files_1\",
        SIZE                    =  27520MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_li_2,
        FILENAME = "d:\devjp\lineitem_table_files_2\",
        SIZE                    =  27520MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_li_3,
        FILENAME = "d:\devjp\lineitem_table_files_3\",
        SIZE                    =  27520MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_li_4,
        FILENAME = "d:\devjp\lineitem_table_files_4\",
        SIZE                    =  27520MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_li_5,
        FILENAME = "d:\devjp\lineitem_table_files_5\",
        SIZE                    =  27520MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_li_6,
        FILENAME = "d:\devjp\lineitem_table_files_6\",
        SIZE                    =  27520MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_li_7,
        FILENAME = "d:\devjp\lineitem_table_files_7\",
        SIZE                    =  27520MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_li_8,
        FILENAME = "d:\devjp\lineitem_table_files_8\",
        SIZE                    =  27520MB,
        FILEGROWTH              = 0),

FILEGROUP       GENERAL_FG
(       NAME                    = tpch100g_gen_1,
        FILENAME = "d:\devjp\general_table_files_1\",
        SIZE                    = 10752MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_gen_2,
        FILENAME = "d:\devjp\general_table_files_2\",
        SIZE                    = 10752MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_gen_3,
        FILENAME = "d:\devjp\general_table_files_3\",
        SIZE                    = 10752MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_gen_4,
        FILENAME = "d:\devjp\general_table_files_4\",
        SIZE                    = 10752MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_gen_5,
        FILENAME = "d:\devjp\general_table_files_5\",
        SIZE                    = 10752MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_gen_6,
        FILENAME = "d:\devjp\general_table_files_6\",
        SIZE                    = 10752MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_gen_7,
        FILENAME = "d:\devjp\general_table_files_7\",
        SIZE                    = 10752MB,
        FILEGROWTH              = 0),
(       NAME                    = tpch100g_gen_8,
        FILENAME = "d:\devjp\general_table_files_8\",
        SIZE                    = 10752MB,
        FILEGROWTH              = 0)

LOG ON
(       NAME                    = tpch100g_log,
        FILENAME = "G:",
        SIZE                    = 20000MB,
        FILEGROWTH              = 0)
```

CreateTables.sql

```
-- File:    CREATETABLES.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

create table PART
        (P_PARTKEY          int                     not null,
         P_NAME             varchar(55)  not null,
         P_MFGR             char(25)     not null,
         P_BRAND char(10)   not null,
         P_TYPE             varchar(25)  not null,
         P_SIZE             int                     not null,
         P_CONTAINER        char(10)     not null,
         P_RETAILPRICE      money                   not null,
         P_COMMENT          varchar(23)  not null)
on GENERAL_FG

create table SUPPLIER
        (S_SUPPKEY          int                     not null,
         S_NAME             char(25)     not null,
         S_ADDRESS          varchar(40)  not null,
         S_NATIONKEY        int                     not null,
         S_PHONE  char(15)  not null,
         S_ACCTBAL          money                   not null,
         S_COMMENT          varchar(101)not null)
on GENERAL_FG

create table PARTSUPP
        (PS_PARTKEY         int                     not null,
         PS_SUPPKEY         int                     not null,
         PS_AVAILQTY        int                     not null,
         PS_SUPPLYCOST      money                   not null,
         PS_COMMENT         varchar(199)not null)
on GENERAL_FG

create table CUSTOMER
        (C_CUSTKEY          int                     not null,
         C_NAME             varchar(25)  not null,
         C_ADDRESS          varchar(40)  not null,
         C_NATIONKEY        int                     not null,
         C_PHONE  char(15)  not null,
         C_ACCTBAL          money                   not null,
         C_MKTSEGMENT       char(10)     not null,
         C_COMMENT          varchar(117)not null)
on GENERAL_FG

create table ORDERS
```

```
            (O_ORDERKEY         int                 not null,
             O_CUSTKEY          int                 not null,
             O_ORDERSTATUS      char(1)             not null,
             O_TOTALPRICE       money               not null,
             O_ORDERDATE        datetime    not null,
             O_ORDERPRIORITY char(15)       not null,
             O_CLERK  char(15)  not null,
             O_SHIPPRIORITY     int                 not null,
             O_COMMENT          varchar(79) not null)
on GENERAL_FG

create table LINEITEM
            (L_ORDERKEY         int                 not null,
             L_PARTKEY          int                 not null,
             L_SUPPKEY          int                 not null,
             L_LINENUMBER       int                 not null,
             L_QUANTITY         money               not null,
             L_EXTENDEDPRICE money                  not null,
             L_DISCOUNT         money               not null,
             L_TAX              money               not null,
             L_RETURNFLAG       char(1)             not null,
             L_LINESTATUS       char(1)             not null,
             L_SHIPDATE         datetime    not null,
             L_COMMITDATE       datetime    not null,
             L_RECEIPTDATE      datetime    not null,
             L_SHIPINSTRUCT     char(25)    not null,
             L_SHIPMODE         char(10)    not null,
             L_COMMENT          varchar(44) not null)
on LINEITEM_FG

create table NATION
            (N_NATIONKEY        int                 not null,
             N_NAME             char(25)    not null,
             N_REGIONKEY        int                 not null,
             N_COMMENT          varchar(152)not null)
on GENERAL_FG

create table REGION
            (R_REGIONKEY        int                 not null,
             R_NAME             char(25)    not null,
             R_COMMENT          varchar(152)not null)
on GENERAL_FG
```

PinBaseTables.sql

```
exec sp_tableoption 'NATION','pintable',1
exec sp_tableoption 'REGION','pintable',1
```

CreateIndexesStream1.sql

```
-- File:    CREATEINDEXESSTREAM1.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

 create unique index O_OKEY_IDX
            on ORDERS(O_ORDERKEY)
            with fillfactor=95, SORT_IN_TEMPDB
            on GENERAL_FG
```

CreateIndexesStream2.sql

```
-- File:    CREATEINDEXESSTREAM2.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

 create index O_CUSTKEY_IDX
            on ORDERS(O_CUSTKEY)
            with fillfactor=95, SORT_IN_TEMPDB
            on GENERAL_FG
```

CreateIndexesStream3.sql

```
-- File:    CREATEINDEXESSTREAM3.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

 create index L_ORDERKEY_IDX
      on LINEITEM(L_ORDERKEY)
      with fillfactor=95, SORT_IN_TEMPDB
      on LINEITEM_FG

create index PS_SUPPKEY_IDX
      on PARTSUPP(PS_SUPPKEY)
      with fillfactor=100, SORT_IN_TEMPDB
      on GENERAL_FG

create index N_REGIONKEY_IDX
      on NATION(N_REGIONKEY)
      with fillfactor=100, SORT_IN_TEMPDB
      on GENERAL_FG

create index S_NATIONKEY_IDX
            on SUPPLIER(S_NATIONKEY)
            with fillfactor=100, SORT_IN_TEMPDB
            on GENERAL_FG

create index C_NATIONKEY_IDX
            on CUSTOMER(C_NATIONKEY)
            WITH FILLFACTOR=100, SORT_IN_TEMPDB
            on GENERAL_FG
```

CreateIndexesStream4.sql

```
-- File:    CREATEINDEXESSTREAM4.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

 create index L_PARTKEY_SUPPKEY_IDX
            on LINEITEM(L_PARTKEY,L_SUPPKEY)
            with FILLFACTOR=95, SORT_IN_TEMPDB
            on LINEITEM_FG
```

CreateClusteredIndexes.sql

```
-- File:    CREATECLUSTEREDINDEXES.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

 create clustered index L_SHIPDATE_CLUIDX
            on LINEITEM(L_SHIPDATE)
            with FILLFACTOR=95, SORT_IN_TEMPDB
            on LINEITEM_FG

create unique clustered index N_KEY_CLUIDX
            on NATION(N_NATIONKEY)
            with SORT_IN_TEMPDB
            on GENERAL_FG

create unique clustered index R_KEY_CLUIDX
            on REGION(R_REGIONKEY)
            with SORT_IN_TEMPDB
            on GENERAL_FG

create unique clustered index P_KEY_CLUIDX
            on PART(P_PARTKEY)
            with SORT_IN_TEMPDB
            on GENERAL_FG

create unique clustered index S_KEY_CLUIDX
            on SUPPLIER(S_SUPPKEY)
```

```
        with SORT_IN_TEMPDB
        on GENERAL_FG

create unique clustered index C_KEY_CLUIDX
        on CUSTOMER(C_CUSTKEY)
        with SORT_IN_TEMPDB
        on GENERAL_FG

create clustered index O_ORDERDATE_CLUIDX
        on ORDERS(O_ORDERDATE)
        with FILLFACTOR=95, SORT_IN_TEMPDB
        on GENERAL_FG

create unique clustered index PS_KEY_CLUIDX
        on PARTSUPP(PS_PARTKEY,PS_SUPPKEY)
        with SORT_IN_TEMPDB
        on GENERAL_FG
```

BackupDatabase1.sql (Test Database)

```
--          File:       BACKUPDATABASE.SQL
--                      Microsoft TPC-H Benchmark Kit Version 1.00
--                      Copyright Microsoft, 1999
--

backup database tpch100g to backupdev1, backupdev2, backupdev3, backupdev4
                                with init, stats=10
```

BackupDatabase2.sql (Test Database)

```
--          File:       BACKUPDATABASE.SQL
--                      Microsoft TPC-H Benchmark Kit Version 1.00
--                      Copyright Microsoft, 1999
--

backup database tpch100g to  backupdev5, backupdev6, backupdev7,
backupdev8 with init, stats=10
```

RestoreDatabase1.sql (Test Database)

```
--          File:       RESTOREDATABASE.SQL
--                      BACKUP STORE AT  M:\  N:\ AND O:\
--                      DRIVES      03/08/2000
--

restore database tpch100g from backupdev1, backupdev2, backupdev3,
                        backupdev4 with stats=10
```

RestoreDatabase2.sql (Test Database)

```
--          File:       RESTOREDATABASE.SQL
--                      BACKUP STORE AT  M:\  N:\ AND O:\
--                      DRIVES      03/08/2000
--

restore database tpch100g from backupdev5, backupdev6, backupdev7,
backupdev8 with stats=10
```

ResizeTempDB.sql

```
-- File:    MOVETEMPDB.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999

--  NEW MODIFY CODE BELOW   2/26/2000


alter database tempdb modify file
      (name = 'tempdev', size = 21952MB)
```

ResizeTempDB2.sql

```
-- File:    MOVETEMPDB.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999

-- alter database tempdb modify file
--   (name='tempdev',filename='X:\TempDB.mdf')

--  NEW MODIFY CODE BELOW   2/26/20002

alter database tempdb add file
            (name = 'tempdev2', filename = 'd:\devjp\tempdev2\',size =
21952MB),
            (name = 'tempdev3', filename = 'd:\devjp\tempdev3\',size =
21952MB),
            (name = 'tempdev4', filename = 'd:\devjp\tempdev4\',size =
21952MB),
            (name = 'tempdev5', filename = 'd:\devjp\tempdev5\',size =
21952MB),
            (name = 'tempdev6', filename = 'd:\devjp\tempdev6\',size =
21952MB),
      (name = 'tempdev7', filename = 'd:\devjp\tempdev7\',size =  21952MB),
            (name = 'tempdev8', filename = 'd:\devjp\tempdev8\',size =
21952MB)
```

ResizeTempLog.sql

```
-- Need to restart the SQL Server to setup the size  - - -

--  Note:  need to increase the size of templog to 8500MB)


alter database tempdb modify file
      (name = 'templog', size = 8500MB)
```

CreateBackupDevices1.sql (Test Database)

```
-- File:    CREATEBACKUPDEVICES.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

if exists (select name from sysdevices where name = 'backupdev1')
            exec sp_dropdevice backupdev1
GO

if exists (select name from sysdevices where name = 'backupdev2')
            exec sp_dropdevice backupdev2
GO

if exists (select name from sysdevices where name = 'backupdev3')
            exec sp_dropdevice backupdev3
GO
if exists (select name from sysdevices where name = 'backupdev4')
            exec sp_dropdevice backupdev4
GO


sp_addumpdevice 'disk', 'backupdev1', 'D:\devjp\backupdev1\tpchbackupdev1'
GO
sp_addumpdevice 'disk', 'backupdev2', 'D:\devjp\backupdev2\tpchbackupdev2'
GO
sp_addumpdevice 'disk', 'backupdev3', 'D:\devjp\backupdev3\tpchbackupdev3'
GO
sp_addumpdevice 'disk', 'backupdev4', 'D:\devjp\backupdev4\tpchbackupdev4'
GO
```

CreateBackupDevices2.sql (Test Database)

```
-- File:    CREATEBACKUPDEVICES.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--


if exists (select name from sysdevices where name = 'backupdev6')
                exec sp_dropdevice backupdev6
GO

if exists (select name from sysdevices where name = 'backupdev7')
                exec sp_dropdevice backupdev7
GO

if exists (select name from sysdevices where name = 'backupdev8')
                exec sp_dropdevice backupdev8
GO

if exists (select name from sysdevices where name = 'backupdev5')
                exec sp_dropdevice backupdev5
GO

sp_addumpdevice 'disk', 'backupdev5', 'D:\devjp\backupdev5\tpchbackupdev5'
GO
sp_addumpdevice 'disk', 'backupdev6', 'D:\devjp\backupdev6\tpchbackupdev6'
GO
sp_addumpdevice 'disk', 'backupdev7', 'D:\devjp\backupdev7\tpchbackupdev7'
GO
sp_addumpdevice 'disk', 'backupdev8', 'D:\devjp\backupdev8\tpchbackupdev8'
GO
```

CreateDeleteTables.sql

```
-- File:    CREATEDELETETABLES.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

if exists (select name from sysobjects where
name='OLDORDERS_%DELETE_SEGMENT%')
  drop table OLDORDERS_%DELETE_SEGMENT%
GO

create table OLDORDERS_%DELETE_SEGMENT% (O_ORDERKEY int) on
GENERAL_FG
GO
```

CreateInsertTables.sql

```
-- File:    CREATEINSERTTABLES.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

if exists (select name from sysobjects where
name='NEWORDERS_%INSERT_SEGMENT%')
  drop table NEWORDERS_%INSERT_SEGMENT%
GO

create table NEWORDERS_%INSERT_SEGMENT% (
      O_ORDERKEY int not null,
      O_CUSTKEY int not null,
      O_ORDERSTATUS char(1)  not null,
      O_TOTALPRICE  money  not null,
      O_ORDERDATE datetime not null,
      O_ORDERPRIORITY char(15)  not null,
      O_CLERK  char(15) not null,
      O_SHIPPRIORITY  int not null,
      O_COMMENT varchar(79) not null
) on GENERAL_FG
GO

if exists (select name from sysobjects where
name='NEWLINEITEM_%INSERT_SEGMENT%')
  drop table NEWLINEITEM_%INSERT_SEGMENT%
GO

create table NEWLINEITEM_%INSERT_SEGMENT%(
      L_ORDERKEY  int  not null,
      L_PARTKEY int not null,
      L_SUPPKEY int not null,
      L_LINENUMBER int not null,
      L_QUANTITY money not null,
      L_EXTENDEDPRICE money not null,
      L_DISCOUNT money not null,
      L_TAX money not null,
      L_RETURNFLAG char(1) not null,
      L_LINESTATUS char(1) not null,
      L_SHIPDATE datetime not null,
      L_COMMITDATE datetime not null,
      L_RECEIPTDATE datetime not null,
      L_SHIPINSTRUCT char(25)  not null,
      L_SHIPMODE char(10 ) not null,
      L_COMMENT varchar(44) not null
) on GENERAL_FG
GO
```

MoveTempDB.sql

```
-- File:    MOVETEMPDB.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999


alter database tempdb modify file
      (name = 'tempdev', filename = 'd:\devjp\tempdev1\')
```

MoveTempLog.sql

```
-- File:    MOVETEMPDB.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999

--  NEW MODIFY CODE BELOW   2/26/2000

-- need to move the templog also

alter database tempdb modify file
      (name = 'templog', filename = 'K:')
```

CreateDatabase.sql (Qualification Database)

```
-- CreateDatabase
-- for use with StepMaster
-- Uses FileGroups

--
--          Drop the existing database
--

if exists (select name from sysdatabases where name = 'tpch1g')
                drop database tpch1g

CREATE DATABASE tpch1g
ON PRIMARY
(               NAME                = tpch1g_root,
                FILENAME            = "d:\tpch1g_root.mdf",
```

```
        SIZE            = 15MB,
        FILEGROWTH      = 0),
FILEGROUP       LINEITEM_FG
(       NAME            = tpch1g_li_1,
        FILENAME        = "d:\devjp\lineitem_table_files_1\",
        SIZE            =  230MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_li_2,
        FILENAME        = "d:\devjp\lineitem_table_files_2\",
        SIZE            =  230MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_li_3,
        FILENAME        = "d:\devjp\lineitem_table_files_3\",
        SIZE            =  230MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_li_4,
        FILENAME        = "d:\devjp\lineitem_table_files_4\",
        SIZE            =  230MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_li_5,
        FILENAME        = "d:\devjp\lineitem_table_files_5\",
        SIZE            =  230MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_li_6,
        FILENAME        = "d:\devjp\lineitem_table_files_6\",
        SIZE            =  230MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_li_7,
        FILENAME        = "d:\devjp\lineitem_table_files_7\",
        SIZE            =  230MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_li_8,
        FILENAME        = "d:\devjp\lineitem_table_files_8\",
        SIZE            =  230MB,
        FILEGROWTH      = 0),


FILEGROUP       GENERAL_FG
(       NAME            = tpch1g_gen_1,
        FILENAME        = "d:\devjp\general_table_files_1\",
        SIZE            = 100MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_gen_2,
        FILENAME        = "d:\devjp\general_table_files_2\",
        SIZE            = 100MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_gen_3,
        FILENAME        = "d:\devjp\general_table_files_3\",
        SIZE            = 100MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_gen_4,
        FILENAME        = "d:\devjp\general_table_files_4\",
        SIZE            = 100MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_gen_5,
        FILENAME        = "d:\devjp\general_table_files_5\",
        SIZE            = 100MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_gen_6,
        FILENAME        = "d:\devjp\general_table_files_6\",
        SIZE            = 100MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_gen_7,
        FILENAME        = "d:\devjp\general_table_files_7\",
        SIZE            = 100MB,
        FILEGROWTH      = 0),
(       NAME            = tpch1g_gen_8,
        FILENAME        = "d:\devjp\general_table_files_8\",
        SIZE            = 100MB,
        FILEGROWTH      = 0)


LOG ON
(       NAME            = tpch1g_log,
        FILENAME        = "G:",
        SIZE            = 5000MB,
        FILEGROWTH      = 0)
```

CreateBackupDevices1.sql (Qualification Database)

```
-- File:    CREATEBACKUPDEVICES.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

if exists (select name from sysdevices where name = 'backup1gdev1')
            exec sp_dropdevice backup1gdev1
GO

if exists (select name from sysdevices where name = 'backup1gdev2')
            exec sp_dropdevice backup1gdev2
GO

if exists (select name from sysdevices where name = 'backup1gdev3')
            exec sp_dropdevice backup1gdev3
GO

sp_addumpdevice 'disk', 'backup1gdev1',
'D:\devjp\backupdev1\tpchbackup1gdev1'
GO
sp_addumpdevice 'disk', 'backup1gdev2',
'D:\devjp\backupdev2\tpchbackup1gdev2'
GO
sp_addumpdevice 'disk', 'backup1gdev3',
'D:\devjp\backupdev3\tpchbackup1gdev3'
GO
```

CreateBackupDevices2.sql (Qualification Database)

```
-- File:    CREATEBACKUPDEVICES.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

if exists (select name from sysdevices where name = 'backup1gdev4')
            exec sp_dropdevice backup1gdev4
GO

if exists (select name from sysdevices where name = 'backup1gdev5')
            exec sp_dropdevice backup1gdev5
GO

if exists (select name from sysdevices where name = 'backup1gdev6')
            exec sp_dropdevice backup1gdev6
GO


sp_addumpdevice 'disk', 'backup1gdev4',
'D:\devjp\backupdev4\tpchbackup1gdev4'
GO
```

```
sp_addumpdevice 'disk', 'backup1gdev5',
'D:\devjp\backupdev5\tpchbackup1gdev5'
GO
sp_addumpdevice 'disk', 'backup1gdev6',
'D:\devjp\backupdev6\tpchbackup1gdev6'
```

RestoreDatabase1.sql (Qualification Database)

```
--          File:       BACKUPDATABASE.SQL
--                      Microsoft TPC-H Benchmark Kit Version 1.00
--                      Copyright Microsoft, 1999
--

restore database tpch1g from backup1gdev1, backup1gdev2,
backup1gdev3
```

RestoreDatabase2.sql (Qualification Database)

```
--          File:       BACKUPDATABASE.SQL
--                      Microsoft TPC-H Benchmark Kit Version 1.00
--                      Copyright Microsoft, 1999
--

restore database tpch1g from backup1gdev4, backup1gdev5,
backup1gdev6
```

BackupDatabase1.sql (Qualification Database)

```
--          File:       BACKUPDATABASE.SQL
--                      Microsoft TPC-H Benchmark Kit Version 1.00
--                      Copyright Microsoft, 1999
--

backup database tpch1g to backup1gdev1, backup1gdev2,
backup1gdev3  with init, stats=10
```

BackupDatabase2.sql (Qualification Database)

```
--          File:       BACKUPDATABASE.SQL
--                      Microsoft TPC-H Benchmark Kit Version 1.00
--                      Copyright Microsoft, 1999
--

backup database tpch1g to backup1gdev4, backup1gdev5,
backup1gdev6  with init, stats=10
```

Appendix C: Query Validation Scripts and Output

Validation_Query_1.sql

```
-- using default substitutions


/* TPC_H  Query 1 - Pricing Summary Report */

SELECT   L_RETURNFLAG,
         L_LINESTATUS,
         SUM(L_QUANTITY)
         AS SUM_QTY,
         SUM(L_EXTENDEDPRICE)
AS SUM_BASE_PRICE,
         SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT))
AS SUM_DISC_PRICE,

SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT)*(1+L_TAX))    AS
SUM_CHARGE,
         AVG(L_QUANTITY)
AS AVG_QTY,
         AVG(L_EXTENDEDPRICE)
AS AVG_PRICE,
         AVG(L_DISCOUNT)
AS AVG_DISC,
         COUNT(*)
AS COUNT_ORDER
FROM     LINEITEM
WHERE    L_SHIPDATE          <= dateadd(dd, -90, '1998-12-01')
GROUP    BY          L_RETURNFLAG,
                     L_LINESTATUS
ORDER    BY          L_RETURNFLAG,
                     L_LINESTATUS
```

Validation_Query_1.out

```
L_RETURNFLAG L_LINESTATUS SUM_QTY
SUM_BASE_PRICE       SUM_DISC_PRICE       SUM_CHARGE
      AVG_QTY             AVG_PRICE           AVG_DISC
COUNT_ORDER
------------ ------------ --------------------- ----------------------
---------------------- ---------------------- ---------------------- ----------------------
---------------------- -----------
A            F            37734107.0000        56586554400.7300
53758257134.8700      55909065224.7041      25.5220
38273.1297           .0499                1478493
N            F            991417.0000          1487504710.3800
1413082168.0541       1469649223.2395       25.5164
38284.4677           .0500                38854
N            O            74476040.0000        111701729697.7400
106118230307.6056     110367043876.2372     25.5022
38249.1179           .0499                2920374
R            F            37719753.0000        56568041380.9000
53741292684.6040      55889619121.7027      25.5057
38250.8546           .0500                1478870

(4 row(s) affected)
```

Validation_Query_2.sql

```
-- using default substitutions

/* TPC_H  Query 2 - Minimum Cost Supplier */

SELECT   TOP 100
         S_ACCTBAL,
         S_NAME,
         N_NAME,
         P_PARTKEY,
         P_MFGR,
         S_ADDRESS,
         S_PHONE,
         S_COMMENT
FROM     PART,
         SUPPLIER,
         PARTSUPP,
         NATION,
         REGION
WHERE    P_PARTKEY           = PS_PARTKEY AND
         S_SUPPKEY           = PS_SUPPKEY AND
         P_SIZE              = 15 AND
         P_TYPE              LIKE '%%BRASS' AND
         S_NATIONKEY         = N_NATIONKEY AND
         N_REGIONKEY         = R_REGIONKEY AND
         R_NAME              = 'EUROPE' AND
         PS_SUPPLYCOST       = (          SELECT
MIN(PS_SUPPLYCOST)
                                          FROM     PARTSUPP,
                                                   SUPPLIER,
                                                   NATION,
                                                   REGION
                                          WHERE    P_PARTKEY
= PS_PARTKEY AND
                                                   S_SUPPKEY
= PS_SUPPKEY AND

S_NATIONKEY       = N_NATIONKEY AND

N_REGIONKEY       = R_REGIONKEY AND
                                                   R_NAME
= 'EUROPE'
                                 )
ORDER    BY          S_ACCTBAL DESC,
                     N_NAME,
                     S_NAME,
                     P_PARTKEY
```

Validation_Query_2.out

```
S_ACCTBAL          S_NAME               N_NAME
P_PARTKEY   P_MFGR               S_ADDRESS
 S_PHONE       S_COMMENT

---------------------- ------------------------- ------------------------- ----------
------------------------ ----------------------------------------- ---------------
-------------------------------------------------------------------------------------------
---------------
9938.5300          Supplier#000005359       UNITED KINGDOM
   185358      Manufacturer#4
QKuHYh,vZGiwu2FWEJoLDx04             33-429-790-6131 blithely
silent pinto beans are furiously. slyly final deposits acros
```

9937.8400 Supplier#000005969 ROMANIA
108438 Manufacturer#1
ANDENSOSmk,miq23Xfb5RWt6dvUcvt6Qa 29-520-692-3537
carefully slow deposits use furiously. slyly ironic platelets above the
ironic
9936.2200 Supplier#000005250 UNITED KINGDOM
249 Manufacturer#4 B3rqp0xbSEim4Mpy2RH J
33-320-228-2957 blithely special packages are. stealthily
express deposits across the closely final instructi
9923.7700 Supplier#000002324 GERMANY
29821 Manufacturer#4 y3OD9UywSTOk
17-779-299-1839 quickly express packages breach quiet pinto beans.
requ
9871.2200 Supplier#000006373 GERMANY
43868 Manufacturer#5 J8fcXWsTqM
17-813-485-8637 never silent deposits integrate furiously blit

9870.7800 Supplier#000001286 GERMANY
81285 Manufacturer#2
YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH 17-516-924-4574
final theodolites cajole slyly special,

9870.7800 Supplier#000001286 GERMANY
181285 Manufacturer#4
YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH 17-516-924-4574
final theodolites cajole slyly special,

9852.5200 Supplier#000008973 RUSSIA
18972 Manufacturer#2 t5L67YdBYYH6o,Vz24jpDyQ9
32-188-594-7038 quickly regular instructions wake-- carefully
unusual braids into the expres
9847.8300 Supplier#000008097 RUSSIA
130557 Manufacturer#2 xMe97bpE69NzdwLoX
32-375-640-3593 slyly regular dependencies sleep slyly furiously
express dep
9847.5700 Supplier#000006345 FRANCE
86344 Manufacturer#1
VSt3rzk3qG698u6ld8HhOByvrTcSTSvQlDQDag 16-886-766-7945
silent pinto beans should have to snooze carefully along the final
reques
9847.5700 Supplier#000006345 FRANCE
173827 Manufacturer#2
VSt3rzk3qG698u6ld8HhOByvrTcSTSvQlDQDag 16-886-766-7945
silent pinto beans should have to snooze carefully along the final
reques
9836.9300 Supplier#000007342 RUSSIA
4841 Manufacturer#4 JOlK7C1,7xrEZSSOw
32-399-414-5385 final accounts haggle. bold accounts are furiously
dugouts. furiously silent asymptotes are slyly
9817.1000 Supplier#000002352 RUSSIA
124815 Manufacturer#2 4LfoHUZjgjEbAKw
TgdKcgOc4D4uCYw 32-551-831-1437 blithely pending
packages across the ironic accounts grow slyly after the furiously

9817.1000 Supplier#000002352 RUSSIA
152351 Manufacturer#3 4LfoHUZjgjEbAKw
TgdKcgOc4D4uCYw 32-551-831-1437 blithely pending
packages across the ironic accounts grow slyly after the furiously

9739.8600 Supplier#000003384 FRANCE
138357 Manufacturer#2 o,Z3v4POifevE k9U1b 6J1ucX,I
16-494-913-5925 slyly ironic theodolites hag

9721.9500 Supplier#000008757 UNITED KINGDOM
156241 Manufacturer#3 Atg6GnM4dT2
33-821-407-2995 ironic, even dolphins above the furiously ironic
foxes sleep slyly around the caref

9681.3300 Supplier#000008406 RUSSIA
78405 Manufacturer#1 ,qUuXcftUl
32-139-873-8571 furiously even deposits affix thinly special
theodolites. furiou
9643.5500 Supplier#000005148 ROMANIA
107617 Manufacturer#1 kT4ciVFslx9z4s79p Js825
29-252-617-4850 doggedly even ideas boost furiously against the
furiously express
9624.8200 Supplier#000001816 FRANCE
34306 Manufacturer#3
e7vab91vLJPWxxZnewmnDBpDmxYHrb 16-392-237-6726
blithely regular accounts cajole furiously. regular

9624.7800 Supplier#000009658 ROMANIA
189657 Manufacturer#1 oE9uBgEfSS4opIcepXyAYM,x
29-748-876-2014 regular deposits haggle. furiously express
asympto
9612.9400 Supplier#000003228 ROMANIA
120715 Manufacturer#2
KDdpNKN3cWu7ZSrbdqp7AfSLxx,qWB 29-325-784-8187
carefully pending accounts serve. furiously close deposits boost slyly.
q
9612.9400 Supplier#000003228 ROMANIA
198189 Manufacturer#4
KDdpNKN3cWu7ZSrbdqp7AfSLxx,qWB 29-325-784-8187
carefully pending accounts serve. furiously close deposits boost slyly.
q
9571.8300 Supplier#000004305 ROMANIA
179270 Manufacturer#2 qNHZ7WmCzygwMPRDO9Ps
29-973-481-1831 furiously final deposits

9558.1000 Supplier#000003532 UNITED KINGDOM
88515 Manufacturer#4
EOeuiiOn21OVpTlGguufFDFsbN1p0lhpxHp 33-152-301-2164
daring, sly accounts breach about th

9492.7900 Supplier#000005975 GERMANY
25974 Manufacturer#5 S6mIiCTx82z7lV
17-992-579-4839 always pending packages boost slyly.

9461.0500 Supplier#000002536 UNITED KINGDOM
20033 Manufacturer#1 8mmGbyzaU
7ZS2wJumTibypncu9pNkDc4FYA 33-556-973-5522 even foxes
are quickly furiously express requests. packages

9453.0100 Supplier#000000802 ROMANIA
175767 Manufacturer#1 ,6HYXb4uaHITmtMBj4Ak57Pd
29-342-882-6463 final, regular packages across the slowly
regular packag
9408.6500 Supplier#000007772 UNITED KINGDOM
117771 Manufacturer#4 AiC5YAH,gdu0i7
33-152-491-1126 blithely final ideas sleep carefully. requests are

9359.6100 Supplier#000004856 ROMANIA
62349 Manufacturer#5 HYogcF3Jb yh1
29-334-870-9731 carefully unusual packages sleep carefully even
ideas. dogged accoun
9357.4500 Supplier#000006188 UNITED KINGDOM
138648 Manufacturer#1 g801,ssP8wpTk4Hm
33-583-607-1633 carefully regular deposits wake carefully
furiously even i
9352.0400 Supplier#000003439 GERMANY
170921 Manufacturer#4 qYPDgoiBGhCYxjgC
17-128-996-4650 fluffily regular pinto beans wake. unusual, final
ideas c
9312.9700 Supplier#000007807 RUSSIA
90279 Manufacturer#5 oGYMPCk9XHGB2PBfKRnHA

32-673-872-5854 unusual asymptotes above the

9312.9700 Supplier#000007807 RUSSIA
100276 Manufacturer#5 oGYMPCk9XHGB2PBfKRnHA
32-673-872-5854 unusual asymptotes above the

9280.2700 Supplier#000007194 ROMANIA
47193 Manufacturer#3 zhRUQkBSrFYxIAXTfInj
vyGRQjeK 29-318-454-2133 slyly ironic requests despite the
unusual ins
9274.8000 Supplier#000008854 RUSSIA
76346 Manufacturer#3 1xhLoOUM7I3mZ1mKnerw
OSqdbb4QbGa 32-524-148-5221 ruthlessly ironic instructions
along the regular, furious requests integrate car
9249.3500 Supplier#000003973 FRANCE
26466 Manufacturer#1
d18GiDsL6Wm2IsGXM,RZf1jCsgZAOjNYVThTRP4
16-722-866-1658 quickly ironic sauternes use b

9249.3500 Supplier#000003973 FRANCE
33972 Manufacturer#1
d18GiDsL6Wm2IsGXM,RZf1jCsgZAOjNYVThTRP4
16-722-866-1658 quickly ironic sauternes use b

9208.7000 Supplier#000007769 ROMANIA
40256 Manufacturer#5 rsimdze 5o9P Ht7xS
29-964-424-9649 furiously ruthless epitaphs among the furiously
regular accounts use slowly fluffily ev
9201.4700 Supplier#000009690 UNITED KINGDOM
67183 Manufacturer#5 CB BnUTlmi5zdeEl7R7
33-121-267-9529 blithely unusual accounts integrate slyly.
platelets
9192.1000 Supplier#000000115 UNITED KINGDOM
85098 Manufacturer#3 nJ
2t0f7Ve,wL1,6WzGBJLNBUCKlsV 33-597-248-1220 slyly
bold pinto beans boost across the furiously regular packages. carefully
regu
9189.9800 Supplier#000001226 GERMANY
21225 Manufacturer#4 qsLCqSvLyZfuXIpjz
17-725-903-1381 final, express instruction

9128.9700 Supplier#000004311 RUSSIA
146768 Manufacturer#5 I8IjnXd7NSJRs594RxsRR0
32-155-440-7120 regular pinto beans sleep ca

9104.8300 Supplier#000008520 GERMANY
150974 Manufacturer#4 RqRVDgD0ER J9 b41vR2,3
17-728-804-1793 deposits sleep carefully e

9101.0000 Supplier#000005791 ROMANIA
128254 Manufacturer#5 zub2zCV,jhHPPQqi,P2INAjE1zI
n66cOEoXFG 29-549-251-5384 carefully ironic packages after the

9094.5700 Supplier#000004582 RUSSIA
39575 Manufacturer#1 WB0XkCSG3r,mnQ
n,h9VIxjjr9ARHFvKgMDf 32-587-577-1351 asymptotes above the
slyly even requests haggle furiously about the regular accounts

8996.8700 Supplier#000004702 FRANCE
102191 Manufacturer#5 8XVcQK23akp
16-811-269-8946 stealthy requests haggle c

8996.1400 Supplier#000009814 ROMANIA
139813 Manufacturer#2
af0O5pg83lPU4IDVmEylXZVqYZQzSDlYLAmR
29-995-571-8781 ironic theodolites are evenly unusual requests--
pending pinto beans across the in

8968.4200 Supplier#000010000 ROMANIA
119999 Manufacturer#5 aTGLEusCiL4F
PDBdv665XBJhPyCOB0i 29-578-432-2146 furiously final ideas
believe furiously. furiously final ideas
8936.8200 Supplier#000007043 UNITED KINGDOM
109512 Manufacturer#1
FVajceZInZdbJE6Z9XsRUxrUEpiwHDrOXi,1Rz 33-784-177-8208
furiously regular excuses wake after the blithely special pinto beans?
even instructions sl
8929.4200 Supplier#000008770 FRANCE
173735 Manufacturer#4 R7cG26TtXrHAP9 HckhfRi
16-242-746-9248 final accounts sleep furiously. blithely ironic
foxes wake boldly across the furiously s
8920.5900 Supplier#000003967 ROMANIA
26460 Manufacturer#1 eHoAXe62SY9
29-194-731-3944 quickly even requests should have to affix blithely--
fur
8920.5900 Supplier#000003967 ROMANIA
173966 Manufacturer#2 eHoAXe62SY9
29-194-731-3944 quickly even requests should have to affix blithely--
fur
8913.9600 Supplier#000004603 UNITED KINGDOM
137063 Manufacturer#2
OUzlvMUr7n,utLxmPNeYKSf3T24OXskxB5 33-789-255-7342
slyly ironic packages detect furious accounts. ironic de

8877.8200 Supplier#000007967 FRANCE
167966 Manufacturer#5 A3pi1BARM4nx6R,qrwFoRPU
16-442-147-9345 final deposits after the silent deposits ha

8862.2400 Supplier#000003323 ROMANIA
73322 Manufacturer#3 W9 lYcsC9FwBqk3ItL
29-736-951-3710 unusual, pending theodolites integrate furiously
slyly even pinto beans. unusual sheaves sleep befor
8841.5900 Supplier#000005750 ROMANIA
100729 Manufacturer#5
Erx3lAgu0g62iaHF9x50uMH4EgeN9hEG 29-344-502-5481
excuses after the blithely regular packages mold carefully deposits.
regular a
8781.7100 Supplier#000003121 ROMANIA
13120 Manufacturer#5 wNqTogx238ZYCamFb,50v,bj
4IbNFW9Bvw1xP 29-707-291-5144 packages are quickly after the
final, even packages. furiously regular
8754.2400 Supplier#000009407 UNITED KINGDOM
179406 Manufacturer#4 CHRCbkaWcf5B
33-903-970-9604 regular dependencies haggle across the carefully
bold
8691.0600 Supplier#000004429 UNITED KINGDOM
126892 Manufacturer#2 k,BQms5UhoAF1B2Asi,fLib
33-964-337-5038 quickly special foxes against the furiously
silent platelets wake quickly after t
8655.9900 Supplier#000006330 RUSSIA
193810 Manufacturer#2 UozlaENr0ytKe2w6CeIEWFWn
iO3S8Rae7Ou 32-561-198-3705 blithely even packages alongside

8638.3600 Supplier#000002920 RUSSIA
75398 Manufacturer#1 Je2a8bszf3L
32-122-621-7549 express deposits wake. furiously silent requests
wake carefully silent instru
8638.3600 Supplier#000002920 RUSSIA
170402 Manufacturer#3 Je2a8bszf3L
32-122-621-7549 express deposits wake. furiously silent requests
wake carefully silent instru
8607.6900 Supplier#000006003 UNITED KINGDOM
76002 Manufacturer#2
EH9wADcEiuenM0NR08zDwMidw,52Y2RyILEiA
33-416-807-5206 always special foxes wake slyly bold, ironic
accounts. ironic instructions affix carefull

8569.5200 Supplier#000005936 RUSSIA 5935 Manufacturer#5 jXaNZ6vwnEWJ2ksLZJpjtgt0bY2a3AU 32-644-251-7916 packages sleep furiously. special requests about the fluffily even accounts detect

8564.1200 Supplier#000000033 GERMANY 110032 Manufacturer#1 gfeKpYw3400L0SDywXA6Ya1Qmq1w6YB9f3R 17-138-897-9374 ironic instructions are. special pearls above

8553.8200 Supplier#000003979 ROMANIA 143978 Manufacturer#4 BfmVhCAnCMY3jzpjUMy4CNWs9 HzpdQR7INJU 29-124-646-4897 express, ironic pinto beans cajole around the express, even packages. qu

8517.2300 Supplier#000009529 RUSSIA 37025 Manufacturer#5 e44R8o7JAIS9iMcr 32-565-297-8775 furiously silent requests cajole furiously furiously ironic foxes. slyly express p

8517.2300 Supplier#000009529 RUSSIA 59528 Manufacturer#2 e44R8o7JAIS9iMcr 32-565-297-8775 furiously silent requests cajole furiously furiously ironic foxes. slyly express p

8503.7000 Supplier#000006830 RUSSIA 44325 Manufacturer#4 BC4WFCYRUZyaIgchU 4S 32-147-878-5069 quickly regular excuses detect evenly around

8457.0900 Supplier#000009456 UNITED KINGDOM 19455 Manufacturer#1 7SBhZs8gP1cJjT0Qf433YBk 33-858-440-4349 carefully final accounts sleep blithely special foxes. slyly regular pinto beans alon

8441.4000 Supplier#000003817 FRANCE 141302 Manufacturer#2 hU3fz3xL78 16-339-356-5115 blithely blithe ideas are

8432.8900 Supplier#000003990 RUSSIA 191470 Manufacturer#1 wehBBp1RQbfxAYDASS75MsywmsKHRVdkrvNe6m 32-839-509-9301 final requests along the blithely ironic packages kindle against the carefully fina

8431.4000 Supplier#000002675 ROMANIA 5174 Manufacturer#1 HJFStOu9R5NGPOegKhgbzBdyvrG2yh8w 29-474-643-1443 express, final deposits cajole carefully. stealthily unusual requests

8407.0400 Supplier#000005406 RUSSIA 162889 Manufacturer#4 j7 gYF5RW8DC5UrjKC 32-626-152-4621 quickly final sheaves boost. car

8386.0800 Supplier#000008518 FRANCE 36014 Manufacturer#3 2jqzqqAVe9crMVGP,n9nTsQXulNLTUYoJjEDcqWV 16-618-780-7481 slyly ironic theodolites are slyly. dogged, pendin

8376.5200 Supplier#000005306 UNITED KINGDOM 190267 Manufacturer#5 9t8Y8 QqSIsoADPt6NLdk,TP5zyRx41oBUlgoGc9 33-632-514-7931 furiously even instructions integrate during the furiously regular re

8348.7400 Supplier#000008851 FRANCE 66344 Manufacturer#4 nWxi7GwEbjhw1 16-796-240-2472 ironic instructions nag slyly against the slyly even theodolites. requests alongside of

8338.5800 Supplier#000007269 FRANCE 17268 Manufacturer#4 ZwhJSwABUoiB04,3 16-267-277-4365 ruthlessly regular asymptotes a

8328.4600 Supplier#000001744 ROMANIA 69237 Manufacturer#5 oLo3fV64q2,FKHa3p,qHnS7Yzv,ps8 29-330-728-5873 blithely silent excuses are slyly above the furiously even courts

8307.9300 Supplier#000003142 GERMANY 18139 Manufacturer#1 dqblvV8dCNAorGlJ 17-595-447-6026 theodolites sleep blithely carefully regular warhorses. slyly regular ins

8231.6100 Supplier#000009558 RUSSIA 192000 Manufacturer#2 mcdgen,yT1iJDHDS5fV 32-762-137-5858 slyly regular theodolites sleep fluffily express depos

8152.6100 Supplier#000002731 ROMANIA 15227 Manufacturer#4 nluXJCuY1tu 29-805-463-2030 gifts use. slyly silent ideas are carefully beneath the silent instructions. slyly sil

8109.0900 Supplier#000009186 FRANCE 99185 Manufacturer#1 wgfosrVPexl9pEXWywaqlBMDYYf 16-668-570-1402 quickly pending requests are blithely along the ironic, final requests; instr

8102.6200 Supplier#000003347 UNITED KINGDOM 18344 Manufacturer#5 m CtXS2S16i 33-454-274-8532 packages grow special orbits. regular theodolites about the carefully pe

8046.0700 Supplier#000008780 FRANCE 191222 Manufacturer#3 AczzuE0UK9osj ,Lx0Jmh 16-473-215-6395 regular epitaphs integrate slyly.

8042.0900 Supplier#000003245 RUSSIA 135705 Manufacturer#4 Dh8Ikg39onrbOL4DyTfGw8a9oKUX3d9Y 32-836-132-8872 carefully regular instructions integrate blithely silent foxes. furiously express instructions haggl

8042.0900 Supplier#000003245 RUSSIA 150729 Manufacturer#1 Dh8Ikg39onrbOL4DyTfGw8a9oKUX3d9Y 32-836-132-8872 carefully regular instructions integrate blithely silent foxes. furiously express instructions haggl

7992.4000 Supplier#000006108 FRANCE 118574 Manufacturer#1 8tBydnTDwUqfBfFV4l3 16-974-998-8937 regular pinto beans are after

7980.6500 Supplier#000001288 FRANCE 13784 Manufacturer#4 zE,7HgVPrCn 16-646-464-8247 unusual pinto beans cajole furiously according t

7950.3700 Supplier#000008101 GERMANY 33094 Manufacturer#5 kkYvL6IuvojJgTNG IKkaXQDYgx8ILohj 17-627-663-8014 quickly regular requests are furiously. pending deposits wake

7937.9300 Supplier#000009012 ROMANIA 83995 Manufacturer#2 iUiTziH,Ek3i4lwSgunXMgrcTzwdb 29-250-925-9690 blithely bold ideas haggle quickly final, regular request

7914.4500 Supplier#000001013 RUSSIA 125988 Manufacturer#2 riRcntps4KEDtYScjpMIWeYF6mNnR 32-194-698-3365 final, ironic theodolites alongside of the ironic

7912.9100 Supplier#000004211 GERMANY 159180 Manufacturer#5 2wQRVovHrm3,v03IKzfTd,1PYsFXQFFOG 17-266-947-7315 final requests integrate slyly above the silent, even

```
7912.9100       Supplier#000004211      GERMANY
184210  Manufacturer#4
2wQRVovHrm3,v03IKzfTd,1PYsFXQFFOG       17-266-947-7315
final requests integrate slyly above the silent, even

7894.5600       Supplier#000007981      GERMANY
85472   Manufacturer#4          NSJ96vMROAbeXP
 17-963-404-3760 regular, even theodolites integrate carefully. bold,
special theodolites are slyly fluffily iron
7887.0800       Supplier#000009792      GERMANY
164759  Manufacturer#3          Y28ITVeYriT3kIGdV2K8fSZ
V2UqT5H1Otz     17-988-938-4296 pending, ironic packages sleep
among the carefully ironic accounts. quickly final accounts
7871.5000       Supplier#000007206      RUSSIA
104695  Manufacturer#1          3w fNCnrVmvJjE95sgWZzvW
        32-432-452-7731 furiously dogged pinto beans cajole. bold,
express notornis until the slyly pending
7852.4500       Supplier#000005864      RUSSIA
8363    Manufacturer#4          WCNfBPZeSXh3h,c
32-454-883-3821 blithely regular deposits

7850.6600       Supplier#000001518      UNITED KINGDOM
   86501    Manufacturer#1          ONda3YJiHKJOC
   33-730-383-3892 furiously final accounts wake carefully idle
requests. even dolphins wake acc
7843.5200       Supplier#000006683      FRANCE
11680   Manufacturer#4
2Z0JGkiv01Y00oCFwUGfviIbhzCdy           16-464-517-8943
carefully bold accounts doub


(100 row(s) affected)
```

Validation_Query_3.sql

```
 -- using default substitutions

/* TPC_H  Query 3 - Shipping Priority */

SELECT  TOP 10
        L_ORDERKEY,
        SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT))     AS
REVENUE,
        O_ORDERDATE,
        O_SHIPPRIORITY
FROM    CUSTOMER,
        ORDERS,
        LINEITEM
WHERE   C_MKTSEGMENT    = 'BUILDING' AND
        C_CUSTKEY       = O_CUSTKEY AND
        L_ORDERKEY      = O_ORDERKEY AND
        O_ORDERDATE     < '1995-03-15' AND
        L_SHIPDATE      > '1995-03-15'
GROUP   BY      L_ORDERKEY,
                O_ORDERDATE,
                O_SHIPPRIORITY
ORDER   BY      REVENUE DESC,
                O_ORDERDATE
```

Validation_Query_3.out

```
L_ORDERKEY  REVENUE             O_ORDERDATE
O_SHIPPRIORITY
----------- --------------------- ----------------------- --------------
2456423     406181.0111         1995-03-05 00:00:00.000 0
3459808     405838.6989         1995-03-04 00:00:00.000 0
492164      390324.0610         1995-02-19 00:00:00.000 0
1188320     384537.9359         1995-03-09 00:00:00.000 0
2435712     378673.0558         1995-02-26 00:00:00.000 0
4878020     378376.7952         1995-03-12 00:00:00.000 0
5521732     375153.9215         1995-03-13 00:00:00.000 0
2628192     373133.3094         1995-02-22 00:00:00.000 0
993600      371407.4595         1995-03-05 00:00:00.000 0
2300070     367371.1452         1995-03-13 00:00:00.000 0

(10 row(s) affected)
```

Validation_Query_4.sql

```
-- using default substitutions

/* TPC_H  Query 4 - Order Priority Checking */

SELECT  O_ORDERPRIORITY,
        COUNT(*)                        AS ORDER_COUNT
FROM    ORDERS
WHERE   O_ORDERDATE     >= '1993-07-01' AND
        O_ORDERDATE     < dateadd (mm, 3, '1993-07-01')
AND
        EXISTS          (       SELECT  *
                                FROM    LINEITEM
                                WHERE
L_ORDERKEY      = O_ORDERKEY AND

L_COMMITDATE    < L_RECEIPTDATE
                        )
GROUP   BY      O_ORDERPRIORITY
ORDER   BY      O_ORDERPRIORITY
```

Validation_Query_4.out

```
O_ORDERPRIORITY ORDER_COUNT
--------------- -----------
1-URGENT        10594
2-HIGH          10476
3-MEDIUM        10410
4-NOT SPECIFIED 10556
5-LOW           10487

(5 row(s) affected)
```

Validation_Query_5.sql

```
-- using default substitutions

/* TPC_H  Query 5 - Local Supplier Volume */

SELECT  N_NAME,
        SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT))     AS
REVENUE
FROM    CUSTOMER,
        ORDERS,
        LINEITEM,
        SUPPLIER,
        NATION,
        REGION
WHERE   C_CUSTKEY       = O_CUSTKEY AND
        L_ORDERKEY      = O_ORDERKEY AND
        L_SUPPKEY       = S_SUPPKEY AND
```

```
            C_NATIONKEY     = S_NATIONKEY AND
            S_NATIONKEY     = N_NATIONKEY AND
            N_REGIONKEY     = R_REGIONKEY AND
            R_NAME          = 'ASIA' AND
            O_ORDERDATE     >= '1994-01-01' AND
            O_ORDERDATE     < DATEADD(YY, 1, '1994-01-01')
GROUP   BY      N_NAME
ORDER   BY      REVENUE DESC
```

Validation_Query_5.out

```
N_NAME                    REVENUE
------------------------- --------------------
INDONESIA                  55502041.1697
VIETNAM                    55295086.9967
CHINA                     53724494.2566
INDIA                     52035512.0002
JAPAN                      45410175.6954

(5 row(s) affected)
```

Validation_Query_6.sql

```
-- using default substitutions

/* TPC_H  Query 6 - Forecasting Revenue Change */

SELECT  SUM(L_EXTENDEDPRICE*L_DISCOUNT)     AS
REVENUE
FROM    LINEITEM
WHERE   L_SHIPDATE      >= '1994-01-01' AND
        L_SHIPDATE      < dateadd (yy, 1, '1994-01-01') AND
        L_DISCOUNT      BETWEEN .06 - 0.01 AND .06 +
0.01 AND
        L_QUANTITY      < 24
```

Validation_Query_6.out

```
REVENUE
---------------------
123141078.2283

(1 row(s) affected)
```

Validation_Query_7.sql

```
-- using default substitutions

/* TPC_H  Query 7 - Volume Shipping */

SELECT  SUPP_NATION,
        CUST_NATION,
        L_YEAR,
        SUM(VOLUME)     AS REVENUE
FROM    (       SELECT  N1.N_NAME
AS SUPP_NATION,
                        N2.N_NAME
AS CUST_NATION,
                        datepart(yy,L_SHIPDATE)
AS L_YEAR,

L_EXTENDEDPRICE*(1-L_DISCOUNT)  AS VOLUME
                FROM    SUPPLIER,
                        LINEITEM,
                        ORDERS,
                        CUSTOMER,
                        NATION N1,
                        NATION N2
                WHERE   S_SUPPKEY       =
L_SUPPKEY AND
                        O_ORDERKEY      =
L_ORDERKEY AND
                        C_CUSTKEY       =
O_CUSTKEY AND
                        S_NATIONKEY     =
N1.N_NATIONKEY AND
                        C_NATIONKEY     =
N2.N_NATIONKEY AND
                        (       (N1.N_NAME      =
'FRANCE' AND N2.N_NAME  = 'GERMANY')
                                OR
                                (N1.N_NAME      =
'GERMANY'       AND N2.N_NAME   = 'FRANCE')
                        ) AND
                        L_SHIPDATE      BETWEEN
'1995-01-01' AND '1996-12-31'
        )       AS SHIPPING
GROUP   BY      SUPP_NATION,
                CUST_NATION,
                L_YEAR
ORDER   BY      SUPP_NATION,
                CUST_NATION,
                L_YEAR
```

Validation_Query_7.out

```
SUPP_NATION               CUST_NATION               L_YEAR
REVENUE
------------------------- ------------------------- ----------- ---------------------
FRANCE                    GERMANY                   1995
54639732.7336
FRANCE                    GERMANY                   1996
54633083.3076
GERMANY                   FRANCE                    1995
52531746.6697
GERMANY                   FRANCE                    1996
52520549.0224

(4 row(s) affected)
```

Validation_Query_8.sql

```
-- using default substitutions

/* TPC_H  Query 8 - National Market Share */

SELECT  O_YEAR,
        SUM(CASE        WHEN    NATION  = 'BRAZIL'
                        THEN    VOLUME
                        ELSE    0
                        END) / SUM(VOLUME)      AS
MKT_SHARE
FROM    (       SELECT  datepart(yy,O_ORDERDATE)
AS O_YEAR,
                        L_EXTENDEDPRICE *
(1-L_DISCOUNT)  AS VOLUME,
                        N2.N_NAME
        AS NATION
```

```
                FROM    PART,
                        SUPPLIER,
                        LINEITEM,
                        ORDERS,
                        CUSTOMER,
                        NATION N1,
                        NATION N2,
                        REGION
                WHERE   P_PARTKEY       =
L_PARTKEY AND
                        S_SUPPKEY       =
L_SUPPKEY AND
                        L_ORDERKEY      =
O_ORDERKEY AND
                        O_CUSTKEY       =
C_CUSTKEY AND
                        C_NATIONKEY     =
N1.N_NATIONKEY AND
                        N1.N_REGIONKEY  =
R_REGIONKEY AND
                        R_NAME          = 'AMERICA'
AND
                        S_NATIONKEY     =
N2.N_NATIONKEY AND
                        O_ORDERDATE     BETWEEN
'1995-01-01' AND '1996-12-31' AND
                        P_TYPE          =
'ECONOMY ANODIZED STEEL'
        )       AS      ALL_NATIONS
GROUP   BY      O_YEAR
ORDER   BY      O_YEAR
```

Validation_Query_8.out

```
O_YEAR      MKT_SHARE
----------- ----------------------
1995        .0344
1996        .0414

(2 row(s) affected)
```

Validation_Query_9.sql

```
-- using default substitutions

/* TPC_H  Query 9 - Product Type Profit Measure */

SELECT  NATION,
        O_YEAR,
        SUM(AMOUNT)     AS SUM_PROFIT
FROM    (       SELECT  N_NAME
                        AS NATION,
                        datepart(yy, O_ORDERDATE)
                AS O_YEAR,

L_EXTENDEDPRICE*(1-L_DISCOUNT)-PS_SUPPLYCOST*L_Q
UANTITYAS AMOUNT
                FROM    PART,
                        SUPPLIER,
                        LINEITEM,
                        PARTSUPP,
                        ORDERS,
                        NATION
                WHERE   S_SUPPKEY       =
L_SUPPKEY AND
                        PS_SUPPKEY      =
L_SUPPKEY AND
                        PS_PARTKEY      =
L_PARTKEY AND
                        P_PARTKEY       =
L_PARTKEY AND
                        O_ORDERKEY      =
L_ORDERKEY AND
                        S_NATIONKEY     =
N_NATIONKEY AND
                        P_NAME          LIKE
'%%green%%'
        )       AS PROFIT
GROUP   BY      NATION,
                O_YEAR
ORDER   BY      NATION,
                O_YEAR  DESC
```

Validation_Query_9.out

```
NATION                    O_YEAR      SUM_PROFIT
------------------------- ----------- ---------------------
ALGERIA                    1998       31342867.2345
ALGERIA                    1997       57138193.0233
ALGERIA                    1996       56140140.1330
ALGERIA                    1995       53051469.6534
ALGERIA                    1994       53867582.1286
ALGERIA                    1993       54942718.1324
ALGERIA                    1992       54628034.7127
ARGENTINA                  1998       30211185.7081
ARGENTINA                  1997       50805741.7523
ARGENTINA                  1996       51923746.5755
ARGENTINA                  1995       49298625.7666
ARGENTINA                  1994       50835610.1095
ARGENTINA                  1993       51646079.1775
ARGENTINA                  1992       50410314.9948
BRAZIL                     1998       27217924.3832
BRAZIL                     1997       48378669.1989
BRAZIL                     1996       50482870.3572
BRAZIL                     1995       47623383.6349
BRAZIL                     1994       47840165.7256
BRAZIL                     1993       49054694.0351
BRAZIL                     1992       48667639.0842
CANADA                     1998       30379833.7685
CANADA                     1997       50465052.3114
CANADA                     1996       52560501.3904
CANADA                     1995       52375332.8092
CANADA                     1994       52600364.6587
CANADA                     1993       52644504.0735
CANADA                     1992       53932871.6970
CHINA                      1998       31075466.1649
CHINA                      1997       50551874.4499
CHINA                      1996       51039293.8754
CHINA                      1995       49287534.6169
CHINA                      1994       50851090.0674
CHINA                      1993       54229629.8330
CHINA                      1992       52400529.3720
EGYPT                      1998       29054433.3856
EGYPT                      1997       50627611.4524
EGYPT                      1996       49542212.8446
EGYPT                      1995       48311550.3207
EGYPT                      1994       49790644.7360
EGYPT                      1993       48904292.9692
EGYPT                      1992       49434932.6192
ETHIOPIA                   1998       28040717.2674
ETHIOPIA                   1997       47455009.8664
ETHIOPIA                   1996       46491097.5729
```

```
ETHIOPIA        1995    46804449.3009
ETHIOPIA        1994    48516143.9170
ETHIOPIA        1993    46551891.5629
ETHIOPIA        1992    44934648.6428
FRANCE          1998    32226407.8392
FRANCE          1997    47121485.8602
FRANCE          1996    47263135.4960
FRANCE          1995    47275997.5707
FRANCE          1994    47067209.3315
FRANCE          1993    51163370.1062
FRANCE          1992    47846235.3313
GERMANY         1998    28624942.6598
GERMANY         1997    49309074.8773
GERMANY         1996    49918683.1684
GERMANY         1995    52650718.7242
GERMANY         1994    50346900.4226
GERMANY         1993    50991895.8059
GERMANY         1992    48274126.0991
INDIA           1998    29943144.3544
INDIA           1997    50665453.2312
INDIA           1996    50283092.2912
INDIA           1995    50006774.6446
INDIA           1994    48995190.7556
INDIA           1993    50286902.8525
INDIA           1992    50850329.4023
INDONESIA       1998    27672339.9965
INDONESIA       1997    50512145.7256
INDONESIA       1996    51653060.1166
INDONESIA       1995    51508779.5940
INDONESIA       1994    52817950.3218
INDONESIA       1993    47959994.9551
INDONESIA       1992    51776605.0323
IRAN            1998    29065736.2381
IRAN            1997    50042063.0545
IRAN            1996    50926653.1881
IRAN            1995    51249667.6485
IRAN            1994    50337085.8650
IRAN            1993    51730763.4902
IRAN            1992    49955856.5634
IRAQ            1998    31624551.0017
IRAQ            1997    55121749.0195
IRAQ            1996    55897663.7937
IRAQ            1995    54815472.5170
IRAQ            1994    54408516.1266
IRAQ            1993    53633167.9770
IRAQ            1992    55891939.3396
JAPAN           1998    27934179.6695
JAPAN           1997    44517162.5463
JAPAN           1996    42545606.1204
JAPAN           1995    43749356.3996
JAPAN           1994    44840243.0700
JAPAN           1993    44660015.5328
JAPAN           1992    45410249.1216
JORDAN          1998    26901488.5781
JORDAN          1997    45471878.4099
JORDAN          1996    46794325.7918
JORDAN          1995    45178828.5760
JORDAN          1994    45333636.5076
JORDAN          1993    47971496.0979
JORDAN          1992    44717239.1774
KENYA           1998    28597614.3370
KENYA           1997    47949733.7274
KENYA           1996    46886924.6229
KENYA           1995    46072338.7552
KENYA           1994    45772061.1711
KENYA           1993    46308728.2345
KENYA           1992    47257780.8406
MOROCCO         1998    26732115.5800
MOROCCO         1997    45637304.2499
MOROCCO         1996    45558221.7451
MOROCCO         1995    47851318.8874
MOROCCO         1994    46272172.9445
MOROCCO         1993    46764326.1818
MOROCCO         1992    48122783.5831
MOZAMBIQUE      1998    30712392.0112
MOZAMBIQUE      1997    50316528.7625
MOZAMBIQUE      1996    51640320.2509
MOZAMBIQUE      1995    50693774.5060
MOZAMBIQUE      1994    49253277.6260
MOZAMBIQUE      1993    49153016.5367
MOZAMBIQUE      1992    48247551.8503
PERU            1998    29326102.3196
PERU            1997    49753780.3951
PERU            1996    50935170.2927
PERU            1995    53309883.4072
PERU            1994    50643531.7968
PERU            1993    51584622.0020
PERU            1992    47523899.0547
ROMANIA         1998    30368667.3999
ROMANIA         1997    50365683.8526
ROMANIA         1996    49598999.0148
ROMANIA         1995    47537642.8697
ROMANIA         1994    51455283.0095
ROMANIA         1993    50407136.8922
ROMANIA         1992    48185385.1286
RUSSIA          1998    28322384.0266
RUSSIA          1997    50106685.1822
RUSSIA          1996    51753342.4302
RUSSIA          1995    49215820.3647
RUSSIA          1994    52205666.4407
RUSSIA          1993    51860230.0340
RUSSIA          1992    53251677.1530
SAUDI ARABIA    1998    31541259.8100
SAUDI ARABIA    1997    52438750.8076
SAUDI ARABIA    1996    52543737.8197
SAUDI ARABIA    1995    52938696.5331
SAUDI ARABIA    1994    51389601.9668
SAUDI ARABIA    1993    52937508.8818
SAUDI ARABIA    1992    54843459.6414
UNITED KINGDOM  1998    28494874.0040
UNITED KINGDOM  1997    49381810.8986
UNITED KINGDOM  1996    51386853.9604
UNITED KINGDOM  1995    51509586.7885
UNITED KINGDOM  1994    48086499.7115
UNITED KINGDOM  1993    49166827.2235
UNITED KINGDOM  1992    49349122.0825
UNITED STATES   1998    25126238.9461
UNITED STATES   1997    50077306.4186
UNITED STATES   1996    48048649.4703
UNITED STATES   1995    48809032.4226
UNITED STATES   1994    49296747.1827
UNITED STATES   1993    48029946.8014
UNITED STATES   1992    48671944.4983
VIETNAM         1998    30442736.0594
VIETNAM         1997    50309179.7942
VIETNAM         1996    50488161.4100
VIETNAM         1995    49658284.6125
VIETNAM         1994    50596057.2607
VIETNAM         1993    50953919.1519
VIETNAM         1992    49613838.3151

(175 row(s) affected)
```

Validation_Query_10.sql

```
-- using default substitutions

/* TPC_H  Query 10 - Returned Item Reporting */

SELECT   TOP 20
         C_CUSTKEY,
         C_NAME,
         SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT))      AS
REVENUE,
         C_ACCTBAL,
         N_NAME,
         C_ADDRESS,
         C_PHONE,
         C_COMMENT
FROM     CUSTOMER,
         ORDERS,
         LINEITEM,
         NATION
WHERE    C_CUSTKEY          = O_CUSTKEY
AND
         L_ORDERKEY         = O_ORDERKEY
AND
         O_ORDERDATE        >= '1993-10-01'
AND
         O_ORDERDATE        < dateadd(mm, 3, '1993-10-01')
AND
         L_RETURNFLAG       = 'R'
AND
         C_NATIONKEY        = N_NATIONKEY
GROUP    BY          C_CUSTKEY,
                     C_NAME,
                     C_ACCTBAL,
                     C_PHONE,
                     N_NAME,
                     C_ADDRESS,
                     C_COMMENT
ORDER    BY          REVENUE              DESC
```

Validation_Query_10.out

```
 C_CUSTKEY  C_NAME                 REVENUE
C_ACCTBAL          N_NAME                C_ADDRESS
          C_PHONE        C_COMMENT

---------- ------------------------- ---------------------- ---------------------
----------------------- ---------------------------------------- ---------------
--------------------------------------------------------------------------------
------------------------------
57040      Customer#000057040      734235.2455          632.8700
       JAPAN                Eioyzjf4pp
22-895-641-3466 requests sleep blithely about the furiously i

143347     Customer#000143347      721002.6948          2557.4700
       EGYPT                1aReFYv,Kw4
14-742-935-3718 fluffily bold excuses haggle finally after the u

60838      Customer#000060838      679127.3077          2454.7700
       BRAZIL               64EaJ5vMAHWJlBOxJklpNc2RJiWE
    12-913-494-9813 furiously even pinto beans integrate under the
ruthless foxes; ironic, even dolphins across the slyl
101998     Customer#000101998      637029.5667          3790.8900
       UNITED KINGDOM       01c9CILnNtfOQYmZj
    33-593-865-6378 accounts doze blithely! enticing, final deposits
sleep blithely special accounts. slyly express accounts pla
125341     Customer#000125341      633508.0860          4983.5100
       GERMANY              S29ODD6bceU8QSuuEJznkNaK
    17-582-695-5962 quickly express requests wake quickly blithely

25501      Customer#000025501      620269.7849          7725.0400
       ETHIOPIA
W556MXuoiaYCCZamJI,Rn0B4ACUGdkQ8DZ    15-874-808-6793
quickly special requests sleep evenly among the special deposits.
special deposi
115831     Customer#000115831      596423.8672          5098.1000
       FRANCE               rFeBbEEyk dl
ne7zV5fDrmiq1oK09wV7pxqCgIc 16-715-386-3788 carefully bold
excuses sleep alongside of the thinly idle

84223      Customer#000084223      594998.0239          528.6500
       UNITED KINGDOM       nAVZCs6BaWap rrM27N
2qBnzc5WBauxbA      33-442-824-8191 pending, final ideas haggle
final requests. unusual, regular asymptotes affix according to the even
foxes.
54289      Customer#000054289      585603.3918          5583.0200
       IRAN                 vXCxoCsU0Bad5JQI ,oobkZ
20-834-292-4707 express requests sublate blithely regular requests.
regular, even ideas solve.
39922      Customer#000039922      584878.1134          7321.1100
       GERMANY
Zgy4s50l2GKN4pLDPBU8m342gIw6R          17-147-757-8036 even
pinto beans haggle. slyly bold accounts inte

6226       Customer#000006226      576783.7606          2230.0900
       UNITED KINGDOM
8gPu8,NPGkfyQQ0hcIYUGPIBWc,ybP5g,      33-657-701-3391
quickly final requests against the regular instructions wake blithely
final instructions. pa
922        Customer#000000922      576767.5333          3869.2500
       GERMANY
Az9RFaut7NkPnc5zSD2PwHgVwr4jRzq        17-945-916-9648
boldly final requests cajole blith

147946     Customer#000147946      576455.1320          2030.1300
       ALGERIA              iANyZHjqhyy7Ajah0pTrYyhJ
10-886-956-3143 furiously even accounts are blithely above the
furiousl
115640     Customer#000115640      569341.1933          6436.1000
       ARGENTINA            Vtgfia9qI 7EpHgecU1X
11-411-543-4901 final instructions are slyly according to the

73606      Customer#000073606      568656.8578          1785.6700
       JAPAN                xuR0Tro5yChDfOCrjkd2ol
22-437-653-6966 furiously bold orbits about the furiously busy
requests wake across the furiously quiet theodolites. d
110246     Customer#000110246      566842.9815          7763.3500
       VIETNAM              7KzflgX MDOq7sOkI
31-943-426-9837 dolphins sleep blithely among the slyly final

142549     Customer#000142549      563537.2368          5085.9900
       INDONESIA            ChqEoK43OysjdHbtKCp6dKqjNyvvi9
       19-955-562-2398 regular, unusual dependencies boost slyly;
ironic attainments nag fluffily into the unusual packages?
146149     Customer#000146149      557254.9865          1791.5500
       ROMANIA              s87fvzFQpU
29-744-164-6487 silent, unusual requests detect quickly slyly regul

52528      Customer#000052528      556397.3509          551.7900
       ARGENTINA            NFztyTOR10UOJ
11-208-192-3205 unusual requests detect. slyly dogged theodolites
use slyly. deposit
23431      Customer#000023431      554269.5360          3381.8600
       ROMANIA              HgiV0phqhaIa9aydNoIlb
29-915-458-2654 instructions nag quickly. furiously bold accounts
cajol
```

(20 row(s) affected)

Validation_Query_11.sql

```
 -- using default substitutions

/* TPC_H  Query 11 - Important Stock Indentification */

SELECT   PS_PARTKEY,
         SUM(PS_SUPPLYCOST*PS_AVAILQTY)AS VALUE
FROM     PARTSUPP,
         SUPPLIER,
         NATION
WHERE    PS_SUPPKEY      = S_SUPPKEY      AND
         S_NATIONKEY     = N_NATIONKEY    AND
         N_NAME          = 'GERMANY'
GROUP    BY       PS_PARTKEY
HAVING   SUM(PS_SUPPLYCOST*PS_AVAILQTY)>
                  (        SELECT
SUM(PS_SUPPLYCOST*PS_AVAILQTY)* 0.0001000000
                           FROM     PARTSUPP,
                                    SUPPLIER,
                                    NATION
                           WHERE    PS_SUPPKEY      =
S_SUPPKEY        AND
                                    S_NATIONKEY     =
N_NATIONKEY      AND
                                    N_NAME          =
'GERMANY'
                  )
ORDER    BY       VALUE    DESC
```

Validation_Query_11.out

```
PS_PARTKEY  VALUE
----------- ---------------------
129760      17538456.8600
166726      16503353.9200
191287      16474801.9700
161758      16101755.5400
34452       15983844.7200
139035      15907078.3400
9403        15451755.6200
154358      15212937.8800
38823       15064802.8600
85606       15053957.1500
33354       14408297.4000
154747      14407580.6800
82865       14235489.7800
76094       14094247.0400
222         13937777.7400
121271      13908336.0000
55221       13716120.4700
22819       13666434.2800
76281       13646853.6800
85298       13581154.9300
85158       13554904.0000
139684      13535538.7200
31034       13498025.2500
87305       13482847.0400
10181       13445148.7500
62323       13411824.3000
26489       13377256.3800
96493       13339057.8300
56548       13329014.9700
55576       13306843.3500
159751      13306614.4800
92406       13287414.5000
182636      13223726.7400
199969      13135288.2100
62865       13001926.9400
7284        12945298.1900
197867      12944510.5200
11562       12931575.5100
75165       12916918.1200
97175       12911283.5000
140840      12896562.2300
65241       12890600.4600
166120      12876927.2200
9035        12863828.7000
144616      12853549.3000
176723      12832309.7400
170884      12792136.5800
29790       12723300.3300
95213       12555483.7300
183873      12550533.0500
171235      12476538.3000
21533       12437821.3200
17290       12432159.5000
156397      12260623.5000
122611      12222812.9800
139155      12220319.2500
146316      12215800.6100
171381      12199734.5200
198633      12078226.9500
167417      12046637.6200
59512       12043468.7600
31688       12034893.6400
159586      12001505.8400
8993        11963814.3000
120302      11857707.5500
43536       11779340.5200
9552        11776909.1600
86223       11772205.0800
53776       11758669.6500
131285      11616953.7400
91628       11611114.8300
169644      11567959.7200
182299      11567462.0500
33107       11453818.7600
104184      11436657.4400
67027       11419127.1400
176869      11371451.7100
30885       11369674.7900
54420       11345076.8800
72240       11313951.0500
178708      11294635.1700
81298       11273686.1300
158324      11243442.7200
117095      11242535.2400
176793      11237733.3800
86091       11177793.7900
116033      11145434.3600
129058      11119112.2000
193714      11104706.3900
117195      11077217.9600
49851       11043701.7800
19791       11030662.6200
75800       11012401.6200
161562      10996371.6900
10119       10980015.7500
39185       10970042.5600
47223       10950022.1300
```

175594	10942923.0500
111295	10893675.6100
155446	10852764.5700
156391	10839810.3800
40884	10837234.1900
141288	10837130.2100
152388	10830977.8200
33449	10830858.7200
149035	10826130.0200
162620	10814275.6800
118324	10791788.1000
38932	10777541.7500
121294	10764225.2200
48721	10762582.4900
63342	10740132.6000
5614	10724668.8000
62266	10711143.1000
100202	10696675.5500
197741	10688560.7200
169178	10648522.8000
5271	10639392.6500
34499	10584177.1000
71108	10569117.5600
137132	10539880.4700
78451	10524873.2400
150827	10503810.4800
107237	10488030.8400
101727	10473558.1000
58708	10466280.4400
89768	10465477.2200
146493	10444291.5800
55424	10444006.4800
16560	10425574.7400
133114	10415097.9000
195810	10413625.2000
76673	10391977.1800
97305	10390890.5700
134210	10387210.0200
188536	10386529.9200
122255	10335760.3200
2682	10312966.1000
43814	10303086.6100
34767	10290405.1800
165584	10273705.8900
2231	10270415.5500
111259	10263256.5600
195578	10239795.8200
21093	10217531.3000
29856	10216932.5400
133686	10213345.7600
87745	10185509.4000
135153	10179379.7000
11773	10167410.8400
76316	10165151.7000
123076	10161225.7800
91894	10130462.1900
39741	10128387.5200
111753	10119780.9800
142729	10104748.8900
116775	10097750.4200
102589	10034784.3600
186268	10012181.5700
44545	10000286.4800
23307	9966577.5000
124281	9930018.9000
69604	9925730.6400
21971	9908982.0300
58148	9895894.4000
16532	9886529.9000
159180	9883744.4300
74733	9877582.8800
35173	9858275.9200
7116	9856881.0200
124620	9838589.1400
122108	9829949.3500
67200	9828690.6900
164775	9821424.4400
9039	9816447.7200
14912	9803102.2000
190906	9791315.7000
130398	9781674.2700
119310	9776927.2100
10132	9770930.7800
107211	9757586.2500
113958	9757065.5000
37009	9748362.6900
66746	9743528.7600
134486	9731922.0000
15945	9731096.4500
55307	9717745.8000
56362	9714922.8300
57726	9711792.1000
57256	9708621.0000
112292	9701653.0800
87514	9699492.5300
174206	9680562.0200
72865	9679043.3400
114357	9671017.4400
112807	9665019.2100
115203	9661018.7300
177454	9658906.3500
161275	9634313.7100
61893	9617095.4400
122219	9604888.2000
183427	9601362.5800
59158	9599705.9600
61931	9584918.9800
5532	9579964.1400
20158	9576714.3800
167199	9557413.0800
38869	9550279.5300
86949	9541943.7000
198544	9538613.9200
193762	9538238.9400
108807	9536247.1600
168324	9535647.9900
115588	9532195.0400
141372	9529702.1400
175120	9526068.6600
163851	9522808.8300
160954	9520359.4500
117757	9517882.8000
52594	9508325.7600
60960	9498843.0600
70272	9495775.6200
44050	9495515.3600
152213	9494756.9600
121203	9492601.3000
70114	9491012.3000
167588	9484741.1100
136455	9476241.7800
4357	9464355.6400
6786	9463632.5700
61345	9455336.7000
160826	9446754.8400
71275	9440138.4000

77746 9439118.3500
91289 9437472.0000
56723 9435102.1600
86647 9434604.1800
131234 9432120.0000
198129 9427651.3600
165530 9426193.6800
69233 9425053.9200
6243 9423304.6600
90110 9420422.7000
191980 9419368.3600
38461 9419316.0700
167873 9419024.4900
159373 9416950.1500
128707 9413428.5000
45267 9410863.7800
48460 9409793.9300
197672 9406887.6800
60884 9403442.4000
15209 9403245.3100
138049 9401262.1000
199286 9391770.7000
19629 9391236.4000
134019 9390615.1500
169475 9387639.5800
165918 9379510.4400
135602 9374251.5400
162323 9367566.5100
96277 9360850.6800
98336 9359671.2900
119781 9356395.7300
34440 9355365.0000
57362 9355180.1000
167236 9352973.8400
38463 9347530.9400
86749 9346826.4400
170007 9345699.9000
193087 9343744.0000
150383 9332576.7500
60932 9329582.0200
128420 9328206.3500
162145 9327722.8800
55686 9320304.4000
163080 9304916.9600
160583 9303515.9200
118153 9298606.5600
152634 9282184.5700
84731 9276586.9200
119989 9273814.2000
114584 9269698.6500
131817 9268570.0800
29068 9256583.8800
44116 9255922.0000
115818 9253311.9100
103388 9239218.0800
186118 9236209.1200
155809 9235410.8400
147003 9234847.9900
27769 9232511.6400
112779 9231927.3600
124851 9228982.6800
158488 9227216.4000
83328 9224792.2000
136797 9222927.0900
141730 9216370.6800
87304 9215695.5000
156004 9215557.9000
140740 9215329.2000
100648 9212185.0800
174774 9211718.0000
37644 9211578.6000
48807 9209496.2400
95940 9207948.4000
141586 9206699.2200
147248 9205654.9500
61372 9205228.7600
52970 9204415.9500
26430 9203710.5100
28504 9201669.2000
25810 9198878.5000
125329 9198688.5000
167867 9194022.7200
134767 9191444.7200
127745 9191271.5600
69208 9187110.0000
155222 9186469.1600
196916 9182995.8200
195590 9176353.1200
169155 9175176.0900
81558 9171946.5000
185136 9171293.0400
114790 9168509.1000
194142 9165836.6100
167639 9161165.0000
11241 9160789.4600
82628 9160155.5400
41399 9148338.0000
30755 9146196.8400
6944 9143574.5800
6326 9138803.1600
101296 9135657.6200
181479 9121093.3000
76898 9120983.1000
64274 9118745.2500
175826 9117387.9900
142215 9116876.8800
103415 9113128.6200
119765 9110768.7900
107624 9108837.4500
84215 9105257.3600
73774 9102651.9200
173972 9102069.0000
69817 9095513.8800
86943 9092253.0000
138859 9087719.3000
162273 9085296.4800
175945 9080401.2100
16836 9075715.4400
70224 9075265.9500
139765 9074755.8900
30319 9073233.1000
3851 9072657.2400
181271 9070631.5200
162184 9068835.7800
81683 9067258.4700
153028 9067010.5100
123324 9061870.9500
186481 9058608.3000
167680 9052908.7600
165293 9050545.7000
122148 9046298.1700
138604 9045840.8000
78851 9044822.6000
137280 9042355.3400
8823 9040855.1000
163900 9040848.4800

75600	9035392.4500
81676	9031999.4000
46033	9031460.5800
194917	9028500.0000
133936	9026949.0200
33182	9024971.1000
34220	9021485.3900
20118	9019942.6000
178258	9019881.6600
15560	9017687.2800
111425	9016198.5600
95942	9015585.1200
132709	9015240.1500
39731	9014746.9500
154307	9012571.2000
23769	9008157.6000
93328	9007211.2000
142826	8998297.4400
188792	8996014.0000
68703	8994982.2200
145280	8990941.0500
150725	8985686.1600
172046	8982469.5200
70476	8967629.5000
124988	8966805.2200
17937	8963319.7600
177372	8954873.6400
137994	8950916.7900
84019	8950039.9800
40389	8946158.2000
69187	8941054.1400
4863	8939044.9200
50465	8930503.1400
43686	8915543.8400
131352	8909053.5900
198916	8906940.0300
135932	8905282.9500
104673	8903682.0000
152308	8903244.0800
135298	8900323.2000
156873	8899429.1000
157454	8897339.2000
75415	8897068.0900
46325	8895569.0900
1966	8895117.0600
24576	8895034.7500
19425	8890156.6000
169735	8890085.5600
32225	8889829.2800
124537	8889770.7100
146327	8887836.2300
121562	8887740.4000
44731	8882444.9500
93141	8881850.8800
187871	8873506.1800
71709	8873057.2800
151913	8869321.1700
33786	8868955.3900
35902	8868126.0600
23588	8867769.9000
24508	8867616.0000
161282	8866661.4300
188061	8862304.0000
132847	8862082.0000
166843	8861200.8000
30609	8860214.7300
56191	8856546.9600
160740	8852685.4300
71229	8846106.9900
91208	8845541.2800
10995	8845306.5600
78094	8839938.2900
36489	8838538.1000
198437	8836494.8400
151693	8833807.6400
185367	8829791.3700
65682	8820622.8900
65421	8819329.2400
122225	8816821.8600
85330	8811013.1600
64555	8810643.1200
104188	8808211.0200
54411	8805703.4000
39438	8805282.5600
70795	8800060.9200
20383	8799073.2800
21952	8798624.1900
63584	8796590.0000
158768	8796422.9500
166588	8796214.3800
120600	8793558.0600
157202	8788287.8800
55358	8786820.7500
168322	8786670.7300
25143	8786324.8000
5368	8786274.1400
114025	8786201.1200
97744	8785315.9400
164327	8784503.8600
76542	8782613.2800
4731	8772846.7000
157590	8772006.4500
154276	8771733.9100
28705	8771576.6400
100226	8769455.0000
179195	8769185.1600
184355	8768118.0500
120408	8768011.1200
63145	8761991.9600
53135	8753491.8000
173071	8750508.8000
41087	8749436.7900
194830	8747438.4000
43496	8743359.3000
30235	8741611.0000
26391	8741399.6400
191816	8740258.7200
47616	8737229.6800
152101	8734432.7600
163784	8730514.3400
5134	8728424.6400
155241	8725429.8600
188814	8724182.4000
140782	8720378.7500
153141	8719407.5100
169373	8718609.0600
41335	8714773.8000
197450	8714617.3200
87004	8714017.7900
181804	8712257.7600
122814	8711119.1400
109939	8709193.1600
98094	8708780.0400
74630	8708040.7500
197291	8706519.0900
184173	8705467.4500

192175 8705411.1200
19471 8702536.1200
18052 8702155.7000
135560 8698137.7200
152791 8697325.8000
170953 8696909.1900
116137 8696687.1700
7722 8696589.4000
49788 8694846.7100
13252 8694822.4200
12633 8694559.3600
193438 8690426.7200
17326 8689329.1600
96124 8679794.5800
143802 8676626.4800
30389 8675826.6000
75250 8675257.1400
72613 8673524.9400
123520 8672456.2500
325 8667741.2800
167291 8667556.1800
150119 8663403.5400
88420 8663355.4000
179784 8653021.3400
130884 8651970.0000
172611 8648217.0000
85373 8647796.2200
122717 8646758.5400
113431 8646348.3400
66015 8643349.4000
33141 8643243.1800
69786 8637396.9200
181857 8637393.2800
122939 8636378.0000
196223 8635391.0200
50532 8632648.2400
58102 8632614.5400
93581 8632372.3600
52804 8632109.2500
755 8627091.6800
16597 8623357.0500
119041 8622397.0000
89050 8621185.9800
98696 8620784.8200
94399 8620524.0000
151295 8616671.0200
56417 8613450.3500
121322 8612948.2300
126883 8611373.4200
29155 8610163.6400
114530 8608471.7400
131007 8607394.8200
128715 8606833.6200
72522 8601479.9800
144061 8595718.7400
83503 8595034.2000
112199 8590717.4400
9227 8587350.4200
116318 8585910.6600
41248 8585559.6400
159398 8584821.0000
105966 8582308.7900
137876 8580641.3000
122272 8580400.7700
195717 8577278.1000
165295 8571121.9200
5840 8570728.7400
120860 8570610.4400
66692 8567540.5200
135596 8563276.3100
150576 8562794.1000
7500 8562393.8400
107716 8561541.5600
100611 8559995.8500
171192 8557390.0800
107660 8556696.6000
13461 8556545.1200
90310 8555131.5100
141493 8553782.9300
71286 8552682.0000
136423 8551300.7600
54241 8550785.2500
120325 8549976.6000
424 8547527.1000
196543 8545907.0900
13042 8542717.1800
58332 8536074.6900
9191 8535663.9200
134357 8535429.9000
96207 8534900.6000
92292 8530618.7800
181093 8528303.5200
105064 8527491.6000
59635 8526854.0800
136974 8524351.5600
126694 8522783.3700
6247 8522606.9000
139447 8522521.9200
96313 8520949.9200
108454 8520916.2500
181254 8519496.1000
71117 8519223.0000
131703 8517215.2800
59312 8510568.3600
2903 8509960.3500
102838 8509527.6900
162806 8508906.0500
41527 8508222.3600
118416 8505858.3600
180203 8505024.1600
14773 8500598.2800
140446 8499514.2400
199641 8497362.5900
109240 8494617.1200
150268 8494188.3800
45310 8492380.6500
36552 8490733.6000
199690 8490145.8000
185353 8488726.6800
163615 8484985.0100
196520 8483545.0400
133438 8483482.3500
77285 8481442.3200
55824 8476893.9000
76753 8475522.1200
46129 8472717.9600
28358 8472515.5000
9317 8472145.3200
33823 8469721.4400
39055 8469145.0700
91471 8468874.5600
142299 8466039.5500
97672 8464119.8000
134712 8461781.7900
157988 8460123.2000
102284 8458652.4400

73533	8458453.3200
90599	8457874.8600
112160	8457863.3600
124792	8457633.7000
66097	8457573.1500
165271	8456969.0100
146925	8454887.9100
164277	8454838.5000
131290	8454811.2000
179386	8450909.9000
90486	8447873.8600
175924	8444421.6600
185922	8442394.8800
38492	8436438.3200
172511	8436287.3400
139539	8434180.2900
11926	8433199.5200
55889	8431449.8800
163068	8431116.4000
138772	8428406.3600
126821	8425180.6800
22091	8420687.8800
55981	8419434.3800
100960	8419403.4600
172568	8417955.2100
63135	8415945.5300
137651	8413170.3500
191353	8413039.8400
62988	8411571.4800
103417	8411541.1200
12052	8411519.2800
104260	8408516.5500
157129	8405730.0800
77254	8405537.2200
112966	8403512.8900
168114	8402764.5600
49940	8402328.2000
52017	8398753.6000
176179	8398087.0000
100215	8395906.6100
61256	8392811.2000
15366	8388907.8000
109479	8388027.2000
66202	8386522.8300
81707	8385761.1900
51727	8385426.4000
9980	8382754.6200
174403	8378575.7300
54558	8378041.9200
3141	8377378.2200
134829	8377105.5200
145056	8376920.7600
194020	8375157.6400
7117	8373982.2700
120146	8373796.2000
126843	8370761.2800
62117	8369493.4400
111221	8367525.8100
159337	8366092.2600
173903	8365428.4800
136438	8364065.4500
56684	8363198.0000
137597	8363185.9400
20039	8361138.2400
121326	8359635.5200
48435	8352863.1000
1712	8349107.0000
167190	8347238.7000
32113	8346452.0400
40580	8342983.3200
74785	8342519.1300
14799	8342236.7500
177291	8341736.8300
198956	8340370.6500
69179	8338465.9900
118764	8337616.5600
128814	8336435.5600
82729	8331766.8800
152048	8330638.9900
171085	8326259.5000
126730	8325974.4000
77525	8323282.5000
170653	8322840.5000
5257	8320350.7800
67350	8318987.5600
109008	8317836.5400
199043	8316603.5400
139969	8316551.5400
22634	8316531.2400
173309	8315750.2500
10887	8315019.3600
42392	8312895.9600
126040	8312623.2000
101590	8304555.4200
46891	8302192.1200
138721	8301745.6200
113715	8301533.2000
78778	8299685.6400
142908	8299447.7700
64419	8297631.8000
21396	8296272.2700
4180	8295646.9200
63534	8295383.6700
135957	8294389.8600
30126	8291920.3200
158427	8288938.0000
14545	8288395.9200
75548	8288287.2000
64473	8286137.4400
149553	8285714.8800
151284	8283526.6500
171091	8282934.3600
194256	8278985.3400
952	8276136.0000
121541	8275390.2600
177664	8275315.2000
51117	8274504.3000
66770	8273407.8000
37238	8272728.0600
46679	8270486.5500
165852	8268312.6000
99458	8266564.4700
114519	8265493.5400
7231	8264881.5000
19033	8264826.5600
125123	8262732.6500
18642	8261578.9900
50386	8261380.0500
193770	8259578.8200
7276	8258101.6000
178045	8253904.1500
49033	8253696.2300
187195	8251334.5800
10590	8249227.4000
143779	8247057.7000
35205	8245675.1700

19729	8245081.6000
144946	8240479.8000
123786	8239581.2400
70843	8237973.2000
112437	8236907.5200
5436	8236039.5700
163754	8235471.1600
115945	8234811.3600
27918	8233957.8800
105712	8233571.8600
41007	8229431.7900
40476	8226640.4100
145620	8221371.6000
7771	8220413.3300
86424	8215572.6100
129137	8215478.4000
76020	8210495.3600
140213	8209831.8000
32379	8208338.8800
130616	8207715.7500
195469	8206609.8000
191805	8205147.7500
90906	8200951.2000
170910	8195558.0100
105399	8193122.6300
123798	8192385.9700
90218	8191689.1600
114766	8189339.5400
11289	8187354.7200
178308	8185750.5000
71271	8185519.2400
1115	8184903.3800
152636	8184530.7200
151619	8182909.0500
116943	8181072.6900
28891	8181051.5400
47049	8180955.0000
158827	8180470.9000
92620	8179671.5500
20814	8176953.5400
179323	8176795.5500
193453	8174343.9400
56888	8173342.0000
28087	8169876.3000
164254	8169632.3500
57661	8168848.1600
7363	8167538.0500
164499	8167512.0800
197557	8165940.4500
5495	8164805.2200
966	8163824.7900
98435	8161771.4500
127227	8161344.9200
194100	8160978.7800
40134	8160358.0800
107341	8159952.0500
6790	8158792.6600
43851	8157101.4000
51295	8156419.2000
69512	8151537.0000
164274	8149869.9300
130854	8145338.8500
186865	8143586.8200
176629	8141411.2000
193739	8141377.7700
6810	8139822.6000
27732	8136724.9600
50616	8134089.8200
123908	8128920.5400
140994	8128470.8200
99039	8128290.7800
62735	8124940.5000
47829	8122796.5000
192635	8122687.5700
192429	8119268.0000
145812	8119165.6300
42896	8118529.8000
146877	8118266.1600
60882	8116095.0400
18254	8114783.0400
165464	8114571.8000
57936	8111927.2500
52226	8110723.3200
128571	8106788.8000
100308	8105837.0400
8872	8102395.6200
58867	8102033.1900
145153	8100222.8400
172088	8098138.2000
59398	8095845.4500
89395	8093576.1000
171961	8093538.0000
88736	8090762.1600
174053	8090350.1100
102237	8089103.2200
43041	8086537.9000
110219	8085296.9000
126738	8084199.2000
44787	8083628.4000
31277	8083580.7600
93595	8082188.8000
189040	8080257.2100
59851	8079024.2400
175100	8077904.0100
43429	8076729.9600
154199	8074940.7600
60963	8073894.4000
8768	8072760.9600
66095	8071421.7000
111552	8068184.4800
24563	8067500.4000
16167	8067495.2400
12662	8067248.8500
94540	8063727.1600
23308	8063463.1800
27390	8062823.2500
130660	8062787.4800
8608	8062411.1600
181552	8062008.3000
199319	8060248.5600
55475	8058850.9200
142711	8057926.5800
103499	8056978.0000
105943	8056698.7500
8432	8053052.1600
149392	8049675.6900
101248	8048855.4900
140962	8047260.7000
87101	8046651.8300
133107	8046476.7300
45126	8045924.4000
87508	8042966.3900
124711	8042722.7200
173169	8042224.4100
175161	8041331.9800
167787	8040075.7800

3242	8038855.5300
114789	8038628.3500
43833	8038545.8300
141198	8035110.7200
137248	8034109.3500
96673	8033491.2000
32180	8032380.7200
166493	8031902.4000
66959	8031839.4000
85628	8029693.4400
110971	8029469.7000
130395	8027463.9200
7757	8026840.3700
178446	8025379.0900
41295	8024785.5300
100956	8024179.3000
131917	8021604.7800
24224	8020463.5200
2073	8020009.6400
121622	8018462.1700
14357	8016906.3000
135601	8016209.4400
58458	8016192.5200
73036	8015799.0000
184722	8015680.3100
151664	8014821.9600
195090	8012680.2000
162609	8011241.0000
83532	8009753.8500
50166	8007137.8900
181562	8006805.9600
175165	8005319.7600
62500	8005316.2800
36342	8004333.4000
128435	8004242.8800
92516	8003836.8000
30802	8003710.8800
107418	8000430.3000
46620	7999778.3500
191803	7994734.1500
106343	7993087.7600
59362	7990397.4600
8329	7990052.9000
75133	7988244.0000
179023	7986829.6200
135899	7985726.6400
5824	7985340.0200
148579	7984889.5600
95888	7984735.7200
9791	7982699.7900
170437	7982370.7200
39782	7977858.2400
20605	7977556.0000
28682	7976960.0000
42172	7973399.0000
56137	7971405.4000
64729	7970769.7200
98643	7968603.7300
153787	7967535.5800
8932	7967222.1900
20134	7965713.2800
197635	7963507.5800
80408	7963312.1700
37728	7961875.6800
26624	7961772.3100
44736	7961144.1000
29763	7960605.0300
36147	7959463.6800
146040	7957587.6600
115469	7957485.1400
142276	7956790.6300
181280	7954037.3500
115096	7953047.5500
109650	7952258.7300
93862	7951992.2400
158325	7950728.3000
55952	7950387.0600
122397	7947106.2700
28114	7946945.7200
11966	7945197.4800
47814	7944083.0000
85096	7943691.0600
51657	7943593.7700
196680	7943578.8900
13141	7942730.3400
193327	7941036.2500
152612	7940663.7100
139680	7939242.3600
31134	7938318.3000
45636	7937240.8500
56694	7936015.9500
8114	7933921.8800
71518	7932261.6900
72922	7930400.6400
146699	7929167.4000
92387	7928972.6700
186289	7928786.1900
95952	7927972.7800
196514	7927180.7000
4403	7925729.0400
2267	7925649.3700
45924	7925047.6800
11493	7916722.2300
104478	7916253.6000
166794	7913842.0000
161995	7910874.2700
23538	7909752.0600
41093	7909579.9200
112073	7908617.5700
92814	7908262.5000
88919	7907992.5000
79753	7907933.8800
108765	7905338.9800
146530	7905336.6000
71475	7903367.5800
36289	7901946.5000
61739	7900794.0000
52338	7898638.0800
194299	7898421.2400
105235	7897829.9400
77207	7897752.7200
96712	7897575.2700
10157	7897046.2500
171154	7896814.5000
79373	7896186.0000
113808	7893353.8800
27901	7892952.0000
128820	7892882.7200
25891	7890511.2000
122819	7888881.0200
154731	7888301.3300
101674	7879324.6000
51968	7879102.2100
72073	7877736.1100
5182	7874521.7300

(1048 row(s) affected)

Validation_Query_12.sql

```
-- using default substitutions

/* TPC_H  Query 12 - Shipping Modes and Order Priority */

SELECT   L_SHIPMODE,
         SUM(      CASE      WHEN O_ORDERPRIORITY  =
'1-URGENT'         OR
                                 O_ORDERPRIORITY  =
'2-HIGH'
                                 THEN 1
                                 ELSE 0
                   END)      AS HIGH_LINE_COUNT,
         SUM(      CASE      WHEN O_ORDERPRIORITY <>
'1-URGENT'         AND
                                 O_ORDERPRIORITY <>
'2-HIGH'
                                 THEN 1
                                 ELSE 0
                   END)      AS LOW_LINE_COUNT
FROM     ORDERS,
         LINEITEM
WHERE    O_ORDERKEY        = L_ORDERKEY
AND
         L_SHIPMODE        IN ('MAIL','SHIP')
AND
         L_COMMITDATE      < L_RECEIPTDATE
AND
         L_SHIPDATE        < L_COMMITDATE
AND
         L_RECEIPTDATE     >= '1994-01-01'
AND
         L_RECEIPTDATE     < dateadd(yy, 1, '1994-01-01')
GROUP    BY        L_SHIPMODE
ORDER    BY        L_SHIPMODE
```

Validation_Query_12.out

```
L_SHIPMODE HIGH_LINE_COUNT LOW_LINE_COUNT
---------- --------------- --------------
MAIL      6202          9324
SHIP      6200          9262

(2 row(s) affected)
```

Validation_Query_13.sql

```
-- using default substitutions

/* TPC_H  Query 13 - Customer Distribution */

SELECT   C_COUNT,
         COUNT(*)          AS CUSTDIST
FROM     (         SELECT  C_CUSTKEY,
                           COUNT(O_ORDERKEY)
                   FROM    CUSTOMER left outer join
ORDERS on
                           C_CUSTKEY         =
O_CUSTKEY                  AND
                           O_COMMENT         not like
'%%special%%requests%%'
                   GROUP   BY        C_CUSTKEY
         )         AS C_ORDERS (C_CUSTKEY, C_COUNT)
GROUP    BY        C_COUNT
ORDER    BY        CUSTDIST          DESC,
                   C_COUNT                     DESC
```

Validation_Query_13.out

```
 C_COUNT     CUSTDIST
----------- -----------
0          50004
9          6641
10          6566
11          6058
8          5949
12          5553
13          4989
19          4748
7          4707
18          4625
15          4552
17          4530
14          4484
20          4461
16          4323
21          4217
22          3730
6          3334
23          3129
24          2622
25          2079
5          1972
26          1593
27          1185
4          1033
28          869
29          559
3          398
30          373
31          235
2          144
32          128
33          71
34          48
35          33
1          23
36          17
37          7
40          4
38          4
39          2
41          1

(42 row(s) affected)
```

Validation_Query_14.sql

```
-- using default substitutions

/* TPC_H  Query 14 - Promotion Effect */

SELECT   100.00 * SUM      (         CASE      WHEN
P_TYPE LIKE 'PROMO%%'
                                               THEN
L_EXTENDEDPRICE*(1-L_DISCOUNT)
```

```
                                        ELSE 0
                                    END) /
SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT))     AS
PROMO_REVENUE
FROM    LINEITEM,
        PART
WHERE   L_PARTKEY       = P_PARTKEY     AND
        L_SHIPDATE      >= '1995-09-01'
AND
        L_SHIPDATE      < dateadd(mm, 1, '1995-09-01')
```

Validation_Query_14.out

```
PROMO_REVENUE
---------------------------------------
16.380779266357422

(1 row(s) affected)
```

Validation_Query_15.sql

```
-- using default substitutions

/* TPC_H  Query 15 - Create View for Top Supplier Query */

CREATE  VIEW REVENUE0 (SUPPLIER_NO,
TOTAL_REVENUE)
AS
SELECT  L_SUPPKEY,
        SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT))
FROM    LINEITEM
WHERE   L_SHIPDATE      >= '1996-01-01' AND
        L_SHIPDATE      < dateadd(mm, 3, '1996-01-01')
GROUP   BY      L_SUPPKEY
GO


/* TPC_H  Query 15 - Top Supplier */

SELECT  S_SUPPKEY,
        S_NAME,
        S_ADDRESS,
        S_PHONE,
        TOTAL_REVENUE
FROM    SUPPLIER,
        REVENUE0
WHERE   S_SUPPKEY       = SUPPLIER_NO AND
        TOTAL_REVENUE  = (      SELECT
MAX(TOTAL_REVENUE)
                                FROM    REVENUE0
                            )
ORDER   BY      S_SUPPKEY

DROP VIEW REVENUE0
```

Validation_Query_15.out

```
S_SUPPKEY   S_NAME              S_ADDRESS
  S_PHONE       TOTAL_REVENUE
----------- ------------------------ ---------------------------------------
--------------- ---------------------
8449        Supplier#000008449      Wp34zim9qYFbVctdW
     20-469-856-8873 1772627.2087

(1 row(s) affected)
```

Validation_Query_16.sql

```
-- using default substitutions

/* TPC_H  Query 16 - Parts/Supplier Relationship */

SELECT  P_BRAND,
        P_TYPE,
        P_SIZE,
        COUNT(DISTINCT PS_SUPPKEY)      AS
SUPPLIER_CNT
FROM    PARTSUPP,
        PART
WHERE   P_PARTKEY       = PS_PARTKEY
        AND
        P_BRAND                 <> 'Brand#45'
                        AND
        P_TYPE          NOT LIKE 'MEDIUM
POLISHED%%'                             AND
        P_SIZE          IN (49, 14, 23, 45, 19, 3, 36, 9)
AND
        PS_SUPPKEY      NOT IN  (       SELECT
S_SUPPKEY
                                        FROM
SUPPLIER
                                        WHERE
S_COMMENT       LIKE '%%Customer%%Complaints%%'
                                        )
GROUP   BY      P_BRAND,
                P_TYPE,
                P_SIZE
ORDER   BY      SUPPLIER_CNT    DESC,
                P_BRAND,
                P_TYPE,
                P_SIZE
```

Validation_Query_16.out

```
P_BRAND    P_TYPE                 P_SIZE      SUPPLIER_CNT
---------- ---------------------- ----------- ------------
Brand#41   MEDIUM BRUSHED TIN       3          28
Brand#54   STANDARD BRUSHED COPPER  14          27
Brand#11   STANDARD BRUSHED TIN     23          24
Brand#11   STANDARD BURNISHED BRASS 36          24
Brand#15   MEDIUM ANODIZED NICKEL   3           24

(18304 row(s) omitted)

Brand#52   MEDIUM BRUSHED BRASS     45          3
Brand#53   MEDIUM BRUSHED TIN       45          3
Brand#54   ECONOMY POLISHED BRASS   9           3
Brand#55   PROMO PLATED BRASS       19          3
Brand#55   STANDARD PLATED TIN      49          3

(18314 row(s) affected)
```

Validation_Query_17.sql

```
-- using default substitutions

/* TPC_H  Query 17 - Small-Quantity-Order Revenue */
```

```
SELECT   SUM(L_EXTENDEDPRICE)/7.0              AS
AVG_YEARLY
FROM     LINEITEM,
         PART
WHERE    P_PARTKEY          = L_PARTKEY        AND
         P_BRAND                    = 'Brand#23'
AND
         P_CONTAINER        = 'MED BOX'
AND
         L_QUANTITY         <          (       SELECT   0.2
* AVG(L_QUANTITY)
                                               FROM
LINEITEM
                                               WHERE
L_PARTKEY          = P_PARTKEY
                                    )
```

Validation_Query_17.out

```
AVG_YEARLY
--------------------------
348406.0625000

(1 row(s) affected)
```

Validation_Query_18.sql

```
-- using default substitutions

/* TPC_H  Query 18 - Large Volume Customer */

SELECT   TOP 100
         C_NAME,
         C_CUSTKEY,
         O_ORDERKEY,
         O_ORDERDATE,
         O_TOTALPRICE,
         SUM(L_QUANTITY)
FROM     CUSTOMER,
         ORDERS,
         LINEITEM
WHERE    O_ORDERKEY         IN         (       SELECT
L_ORDERKEY
                                               FROM
LINEITEM
                                               GROUP    BY
L_ORDERKEY HAVING SUM(L_QUANTITY) > 300
                                    )          AND
         C_CUSTKEY          = O_CUSTKEY        AND
         O_ORDERKEY         = L_ORDERKEY
GROUP    BY         C_NAME,
                    C_CUSTKEY,
                    O_ORDERKEY,
                    O_ORDERDATE,
                    O_TOTALPRICE
ORDER    BY         O_TOTALPRICE       DESC,
                    O_ORDERDATE
```

Validation_Query_18.out

```
C_NAME                C_CUSTKEY   O_ORDERKEY
O_ORDERDATE           O_TOTALPRICE
------------------------- ----------- ----------- -----------------------
---------------------- ----------------------
Customer#000128120        128120      4722021     1994-04-07
00:00:00.000 544089.0900          323.0000
Customer#000144617        144617      3043270     1997-02-12
00:00:00.000 530604.4400          317.0000
Customer#000013940        13940       2232932     1997-04-13
00:00:00.000 522720.6100          304.0000
Customer#000066790        66790       2199712     1996-09-30
00:00:00.000 515531.8200          327.0000
Customer#000046435        46435       4745607     1997-07-03
00:00:00.000 508047.9900          309.0000
Customer#000015272        15272       3883783     1993-07-28
00:00:00.000 500241.3300          302.0000
Customer#000146608        146608      3342468     1994-06-12
00:00:00.000 499794.5800          303.0000
Customer#000096103        96103       5984582     1992-03-16
00:00:00.000 494398.7900          312.0000
Customer#000024341        24341       1474818     1992-11-15
00:00:00.000 491348.2600          302.0000
Customer#000137446        137446      5489475     1997-05-23
00:00:00.000 487763.2500          311.0000
Customer#000107590        107590      4267751     1994-11-04
00:00:00.000 485141.3800          301.0000
Customer#000050008        50008       2366755     1996-12-09
00:00:00.000 483891.2600          302.0000
Customer#000015619        15619       3767271     1996-08-07
00:00:00.000 480083.9600          318.0000
Customer#000077260        77260       1436544     1992-09-12
00:00:00.000 479499.4300          307.0000
Customer#000109379        109379      5746311     1996-10-10
00:00:00.000 478064.1100          302.0000
Customer#000054602        54602       5832321     1997-02-09
00:00:00.000 471220.0800          307.0000
Customer#000105995        105995      2096705     1994-07-03
00:00:00.000 469692.5800          307.0000
Customer#000148885        148885      2942469     1992-05-31
00:00:00.000 469630.4400          313.0000
Customer#000114586        114586      551136      1993-05-19
00:00:00.000 469605.5900          308.0000
Customer#000105260        105260      5296167     1996-09-06
00:00:00.000 469360.5700          303.0000
Customer#000147197        147197      1263015     1997-02-02
00:00:00.000 467149.6700          320.0000
Customer#000064483        64483       2745894     1996-07-04
00:00:00.000 466991.3500          304.0000
Customer#000136573        136573      2761378     1996-05-31
00:00:00.000 461282.7300          301.0000
Customer#000016384        16384       502886      1994-04-12
00:00:00.000 458378.9200          312.0000
Customer#000117919        117919      2869152     1996-06-20
00:00:00.000 456815.9200          317.0000
Customer#000012251        12251       735366      1993-11-24
00:00:00.000 455107.2600          309.0000
Customer#000120098        120098      1971680     1995-06-14
00:00:00.000 453451.2300          308.0000
Customer#000066098        66098       5007490     1992-08-07
00:00:00.000 453436.1600          304.0000
Customer#000117076        117076      4290656     1997-02-05
00:00:00.000 449545.8500          301.0000
Customer#000129379        129379      4720454     1997-06-07
00:00:00.000 448665.7900          303.0000
Customer#000126865        126865      4702759     1994-11-07
00:00:00.000 447606.6500          320.0000
Customer#000088876        88876       983201      1993-12-30
00:00:00.000 446717.4600          304.0000
Customer#000036619        36619       4806726     1995-01-17
00:00:00.000 446704.0900          328.0000
Customer#000141823        141823      2806245     1996-12-29
00:00:00.000 446269.1200          310.0000
```

```
Customer#000053029      53029      2662214     1993-08-13
00:00:00.000 446144.4900          302.0000
Customer#000018188      18188      3037414     1995-01-25
00:00:00.000 443807.2200          308.0000
Customer#000066533      66533      29158       1995-10-21
00:00:00.000 443576.5000          305.0000
Customer#000037729      37729      4134341     1995-06-29
00:00:00.000 441082.9700          309.0000
Customer#000003566      3566       2329187     1998-01-04
00:00:00.000 439803.3600          304.0000
Customer#000045538      45538      4527553     1994-05-22
00:00:00.000 436275.3100          305.0000
Customer#000081581      81581      4739650     1995-11-04
00:00:00.000 435405.9000          305.0000
Customer#000119989      119989     1544643     1997-09-20
00:00:00.000 434568.2500          320.0000
Customer#000003680      3680       3861123     1998-07-03
00:00:00.000 433525.9700          301.0000
Customer#000113131      113131     967334      1995-12-15
00:00:00.000 432957.7500          301.0000
Customer#000141098      141098     565574      1995-09-24
00:00:00.000 430986.6900          301.0000
Customer#000093392      93392      5200102     1997-01-22
00:00:00.000 425487.5100          304.0000
Customer#000015631      15631      1845057     1994-05-12
00:00:00.000 419879.5900          302.0000
Customer#000112987      112987     4439686     1996-09-17
00:00:00.000 418161.4900          305.0000
Customer#000012599      12599      4259524     1998-02-12
00:00:00.000 415200.6100          304.0000
Customer#000105410      105410     4478371     1996-03-05
00:00:00.000 412754.5100          302.0000
Customer#000149842      149842     5156581     1994-05-30
00:00:00.000 411329.3500          302.0000
Customer#000010129      10129      5849444     1994-03-21
00:00:00.000 409129.8500          309.0000
Customer#000069904      69904      1742403     1996-10-19
00:00:00.000 408513.0000          305.0000
Customer#000017746      17746      6882        1997-04-09
00:00:00.000 408446.9300          303.0000
Customer#000013072      13072      1481925     1998-03-15
00:00:00.000 399195.4700          301.0000
Customer#000082441      82441      857959      1994-02-07
00:00:00.000 382579.7400          305.0000
Customer#000088703      88703      2995076     1994-01-30
00:00:00.000 363812.1200          302.0000

(57 row(s) affected)
```

Validation_Query_19.sql

```
-- using default substitutions

/* TPC_H  Query 19 - Discounted Revenue */

SELECT   SUM(L_EXTENDEDPRICE* (1 - L_DISCOUNT))     AS
REVENUE
FROM     LINEITEM,
         PART
WHERE    (          P_PARTKEY          = L_PARTKEY
                                       AND
                    P_BRAND                      = 'Brand#12'

AND
                    P_CONTAINER        IN ( 'SM CASE', 'SM
BOX', 'SM PACK', 'SM PKG')             AND
                    L_QUANTITY         >= 1
                                       AND
                    L_QUANTITY         <= 1 + 10
                              AND
                    P_SIZE             BETWEEN 1 AND 5
                                       AND
                    L_SHIPMODE         IN ('AIR', 'AIR REG')
                              AND
                    L_SHIPINSTRUCT     = 'DELIVER IN
PERSON'
         )
         OR
         (          P_PARTKEY          = L_PARTKEY
                                       AND
                    P_BRAND                      = 'Brand#23'

AND
                    P_CONTAINER        IN ( 'MED BAG', 'MED
BOX', 'MED PKG', 'MED PACK')           AND
                    L_QUANTITY         >= 10
                                       AND
                    L_QUANTITY         <= 10 + 10
                                       AND
                    P_SIZE             BETWEEN 1 AND 10
                                       AND
                    L_SHIPMODE         IN ('AIR', 'AIR REG')
                              AND
                    L_SHIPINSTRUCT     = 'DELIVER IN
PERSON'
         )
         OR
         (          P_PARTKEY          = L_PARTKEY
                                       AND
                    P_BRAND                      = 'Brand#34'

AND
                    P_CONTAINER        IN ( 'LG CASE', 'LG
BOX', 'LG PACK', 'LG PKG')             AND
                    L_QUANTITY         >= 20
                                       AND
                    L_QUANTITY         <= 20 + 10
                                       AND
                    P_SIZE             BETWEEN 1 AND 15
                                       AND
                    L_SHIPMODE         IN ('AIR', 'AIR REG')
                              AND
                    L_SHIPINSTRUCT     = 'DELIVER IN
PERSON'
         )
```

Validation_Query_19.out

```
REVENUE
---------------------
3083843.0578

(1 row(s) affected)
```

Validation_Query_20.sql

```
-- using default substitutions

/* TPC_H  Query 20 - Potential Part Promotion */

SELECT   S_NAME,
         S_ADDRESS
```

```
FROM    SUPPLIER,
        NATION
WHERE   S_SUPPKEY       IN      (       SELECT
PS_SUPPKEY
                                        FROM
PARTSUPP
                                        WHERE
PS_PARTKEY in   (       SELECT  P_PARTKEY

                FROM    PART

                WHERE   P_NAME like 'forest%%'

        )       AND

PS_AVAILQTY     >       (       SELECT  0.5 *
sum(L_QUANTITY)

                FROM    LINEITEM

                WHERE   L_PARTKEY       =
PS_PARTKEY      AND

                        L_SUPPKEY       =
PS_SUPPKEY      AND

                        L_SHIPDATE      >=
'1994-01-01'            AND

                        L_SHIPDATE      <
dateadd(yy,1,'1994-01-01')

        )
                                )       AND
        S_NATIONKEY     = N_NATIONKEY   AND
        N_NAME          = 'CANADA'
ORDER   BY      S_NAME
```

Validation_Query_20.out

```
S_NAME                  S_ADDRESS
------------------------- ----------------------------------------
Supplier#000000020      iybAE,RmTymrZVYaFZva2SH,j
Supplier#000000091
YV45D7TkfdQanOOZ7q9QxkyGUapU1oOWU6q3
Supplier#000000197      YC2Acon6kjY3zj3Fbxs2k4Vdf7X0cd2F

Supplier#000000226      83qOdU2EYRdPQAQhEtn GRZEd

Supplier#000000285      Br7e1nnt1yxrw6ImgpJ7YdhFDjuBf
Supplier#000000378      FfbhyCxWvcPrO8ltp9
Supplier#000000402      i9Sw4DoyMhzhKXCH9By,AYSgmD

Supplier#000000530      0qwCMwobKY OcmLyfRXlagA8ukENJv,

Supplier#000000688      D
fw5ocppmZpYBBIPI718hCihLDZ5KhKX
Supplier#000000710      f19YPvOyb
QoYwjKC,oPycpGfieBAcwKJo
Supplier#000000736
l6i2nMwVuovfKnuVgaSGK2rDy65DlAFLegiL7
Supplier#000000761
zlSLelQUj2XrvTTFnv7WAcYZGvvMTx882d4
Supplier#000000884      bmhEShejaS
Supplier#000000887      urEaTejH5POADP2ARrf
Supplier#000000935      ij98czM 2KzWe7dDTOxB8sq0UfCdvrX

Supplier#000000975      ,AC e,tBpNwKb5xMUzeohxlRn,
hdZJo73gFQF8y
Supplier#000001263      rQWr6nf8ZhB2TAiIDIvo5Io
Supplier#000001399      LmrocnIMSyYOWuANx7
Supplier#000001446      lch9HMNU1R7a0LIybsUodVknk6

Supplier#000001454      TOpimgu2TVXIjhiL93h,
Supplier#000001500      wDmF5xLxtQch9ctVu,
Supplier#000001602      uKNWIeafaM644
Supplier#000001626      UhxNRzUu1dtFmp0
Supplier#000001682      pXTkGxrTQVyH1Rr
Supplier#000001699      Q9C4rfJ26oijVPqqcqVXeRI
Supplier#000001700      7hMlCof1Y5zLFg
Supplier#000001726      TeRY7TtTH24sEword7yAaSkjx8
Supplier#000001730      Rc8e,1Pybn r6zo0VJIEiD0UD vhk
Supplier#000001746
qWsendlOekQG1aW4uq06uQaCm51se8lirv7 hBRd
Supplier#000001752      Fra7outx41THYJaRThdOGiBk
Supplier#000001856
jXcRgzYF0ah05iR8p6w5SbJJLcUGyYiURPvFwUWM
Supplier#000001931      FpJbMU2h6ZR2eBv8I9NIxF
Supplier#000001939      Nrk,JA4bfReUs
Supplier#000001990
DSDJkCgBJzuPg1yuM,CUdLnsRliOxkkHezTCA
Supplier#000002020      jB6r1d7MxP6co
Supplier#000002022       dwebGX7Id2pc25YvY33
Supplier#000002036      20ytTtVObjKUUI2WCB0A
Supplier#000002204      uYmlr46C06udCqanj0KiRsoTQakZsEyssL

Supplier#000002243      nSOEV3JeOU79
Supplier#000002245      hz2qWXWVjOyKhqPYMoEwz6zFkrTaDM

Supplier#000002282      ES21K9dxoW1I1TzWCj7ekdlNwSWnv1Z
6mQ,BKn
Supplier#000002303      nCoWfpB6YOymbgOht7ltfklpkHl
Supplier#000002373      RzHSxOTQmElCjxIBiVA52Z
JB58rJhPRylR
Supplier#000002419      qydBQd14I5l5mVXa4fYY
Supplier#000002481      nLKHUOn2Ml9TOA06Znq9GEMcIlMO2

Supplier#000002571      JZUugz04c iJFLrlGsz9O N,W 1rVHNIReyq

Supplier#000002585      CsPoKpw2QuTY4AV1NkWuttneIa4SN

Supplier#000002630      ZIQAvjNUY9KH5ive zm7k VlPiDl7CCo21

Supplier#000002719      4nnzQI2CbqREQUuIsXTBVUkaP4mNS3

Supplier#000002721      HVdFAN2JHMQSpKm
Supplier#000002730      lIFxR4fzm31C6,muzJwl84z
Supplier#000002775      yDclaDaBD4ihH
Supplier#000002853      rTNAOItXka
Supplier#000002875      6JgMi
9Qt6VmwL3Ltt1SRlKww0keLQ,RAZa
Supplier#000002934      m,trBENywSArwg3DhB
Supplier#000002941      Naddba 8YTEKekZyP0
Supplier#000002960
KCPCEsRGGo6vx8TygHh60nAYf9rStQT2T
Supplier#000002980
B9k9yVsyaXvWktOSHezqHiAEp9id0SKzkw
Supplier#000003062      LSQNgqY1xnOzz9zBCapy7HwOZQ

Supplier#000003087      ANwe8QsZ4rgj1HSqVz991eWQ
Supplier#000003089      s5b VCIZqMSZVa r g7LTdcg29GbTE7rI1x

Supplier#000003095      HxON3jJhUi3zjt,r mTD
```

Supplier#000003201 E87yws6I,t0qNs4QW7UzExKiJnJDZWue

Supplier#000003213 pxrRP4irQ1VoyfQ,dTf3
Supplier#000003241 j06SU,LS9O3mwjAMOViANeIhb

Supplier#000003275 9xO4nyJ2QJcX6vGf
Supplier#000003288 EDdfNt7E5Uc,xLTupoIgYL4yY7ujh,

Supplier#000003313
El2I7we,049SPrvomUm4hZwJoOhZkvLxLJXgVH
Supplier#000003314
jnisU8MzqO4iUB3zsPcrysMw3DDUojS4q7LD
Supplier#000003380
jPv0V,pszouuFT3YsAqlP,kxT3u,gTFiEbRt,x
Supplier#000003403 e3X2o ,KCG9tsHji8A XXCxiF2hZWBw

Supplier#000003421 Sh3dt9W5oeofFWovnFhrg,
Supplier#000003441 zvFJIzS,oUuShHjpcX
Supplier#000003590 sy79CMLxqb,Cbo
Supplier#000003607 lNqFHQYjwSAkf
Supplier#000003625
qY588W0Yk5iaUy1RXTgNrEKrMAjBYHcKs
Supplier#000003656 eEYmmO2gmD JdfG32XtDgJV,db56

Supplier#000003782 iVsPZg7bk06TqNMwi0LKbLUrC1zmrg

Supplier#000003918 meRvRCsJoAbfqd0Re4
Supplier#000003941 Pmb05mQfBMS618O7WKqZJ 9vyv

Supplier#000003994 W00LZp3NjK0
Supplier#000004005 V723F1wCy2eA4OgIu8TjBtOVUHp

Supplier#000004033 ncsAhv9Je,kFXTNjfb2
Supplier#000004140 0hL7DJyYjcHL
Supplier#000004165 wTJ2dZNQA8P2oi99N6DT47ndHy,XKD2

Supplier#000004207 tF64pwiOM4IkWjN3mS,e06WuAjLx

Supplier#000004236
dl,HPtJmGipxYsSqn9wmqkuWjst,mCeJ8O6T
Supplier#000004246 Xha aXQF7u4qU3LsHD
Supplier#000004278 bBddbpBxIVp Di9
Supplier#000004343 GK3sbopqrQEkWLMvVBFCG
Supplier#000004346 S3076LEOwo
Supplier#000004388 VfZ l1J,mwp4aS
Supplier#000004406
Ah0ZaLu6VwufPWUz,7kbXgYZhauEaHqGIg
Supplier#000004430 yvSsKNSTL5HLXBET4luOsPNLxKzAMk

Supplier#000004522 xXtCKwsZDArxIBGDfzX2PgobGZsBg

Supplier#000004527 p pVXCnxgcklWF6A1o3OHY3qW6

Supplier#000004542 NJSbLJDroYG2y1r3rDiKg
Supplier#000004574 1HvGwnVueZ5CIndc
Supplier#000004655 67NqBc4 t3PG3F8aO IsqWNq4kGaPowYL

Supplier#000004701 6jX4u47URzIMHf
Supplier#000004711 bEzjp1QdQu ls2ERMxv0km vn6bu2zXlL1

Supplier#000004987 UFx1upJ8MvOvgFjA8
Supplier#000005000 DeX804 w0H8FrCUvahgy ilbuzBX3NK

Supplier#000005100 OfvYPs3Io,wEvvLHNaLuCX
Supplier#000005192 JDp4rhXiDw0kf6RH
Supplier#000005195 Woi3b2ZaicPh ZSfu1EfXhE
Supplier#000005283 5fxYXxwXy,TQX,MqDC2hxzyQ

Supplier#000005300 gXG28YqpxU
Supplier#000005386 Ub6AAfHpWLWP
Supplier#000005426 9Dz2OVT1q sb4BK71ljQ1XjPBYRPvO

Supplier#000005484 saFdOR
qW7AFY,3asPqiiAa11Mo22pCoN0BtPrKo
Supplier#000005505 d2sbjG43KwMPX
Supplier#000005506 On f5ypzoWgB
Supplier#000005516
XsN99Ks9wEvcohU6jRD2MeebQFf76mD8vovuY
Supplier#000005536 Nzo9tGkpgbHT,EZ4D,77MYKl4ah1C

Supplier#000005605 7Vj6Eil0mThqkM
Supplier#000005631 14TVrjlzo2SJEBYCDgpMwTlvwSqC

Supplier#000005730 5rkb0PSews HvxkL8JaD41UpnSF2cg8H1

Supplier#000005736 2dq XTYhtYWSfp
Supplier#000005737 dmEWcS32C3kx,d,B95 OmYn48
Supplier#000005797 ,o,OebwRbSDmVl9gN9fpWPCiqB
UogvlSR
Supplier#000005836 tx3SjPD2ZuWGFBRH,
Supplier#000005875
lK,sYiGzB94hSyHy9xvSZFbVQNCZe2LXZuGbS
Supplier#000005974 REhR5jE,lLusQXvf54SwYySgsSSVFhu

Supplier#000005989 rjFY,5kgLpBu7c
Supplier#000006059 4m0cv8MwJ9yX2vlwI Z
Supplier#000006065 UiI2Cy3W4Tu5sLk LuvXLRy6KihlGv

Supplier#000006070 TalC5m0pDrO6DZbngfmGmqe
Supplier#000006109 rY5gbfh3dKHnylcQUTPGCwnbe
Supplier#000006121 S92ycWwEzYYw4GspCBJN1WMuHhoZ

Supplier#000006215 j2iEbTsl,5PWdqWZ7k1yiISb7qtiiZljDIPEo

Supplier#000006217 RVN23SYT9jenUeaWGXUd
Supplier#000006274 S3yTZWqxTKUq g QQgcW9
AqhCkNZsW51hHuwU
Supplier#000006435 xIgE69XszYbnO4Eon7cHHO8y
Supplier#000006463 7 wkdj2EO49iotley2kmIM
ADpLSszGV3RNWj
Supplier#000006493 ojV f,sNaB6Hm7r,fknDVTL63raJgAjZK

Supplier#000006521 b9 2zjHzxR
Supplier#000006607 3F 2e2gqD5u5B
Supplier#000006706 Ak4ga,ePu1QZ6C3qkrqjosaX0gxvqS9vkbe

Supplier#000006761 n4jhxGMqB5prD1HhpLvwrWStOLlla

Supplier#000006808 HGd2Xo 9nEcHJhZvXjXxWKIpApT

Supplier#000006858 fnlINT885vBBhsWwTGiZ0o22thwGY16h
GHJj21
Supplier#000006872 XIDPiA7PLXCWK6SeEcld
Supplier#000006949 mLxYUJhsGcLtKe ,GFirNu183AvT

Supplier#000006985 PrUUiboQpy,OtgJ01Z4BxJQUyrw9c3I

Supplier#000007072 2tRyX9M1a 4Rcm57s779F1ANG9jlpK

Supplier#000007098
G3j8g0KC4OcbAu2OVoPHrXQWMCUdjq8wgCHOExu
Supplier#000007135 ls DoKV7V5ulfQy9V
Supplier#000007160
TqDGBULB3cTqIT6FKDvm9BS4e4v,zwYiQPb
Supplier#000007169 tEc95D2moN9S84nd55O,dlnW

```
Supplier#000007322      wr7dgte5q MAjiY0uwmi3MyDkSMX1

Supplier#000007365      51xhROLvQMJ05DndtZWt
Supplier#000007398      V8eE6oZ00OFNU,
Supplier#000007402      4UVv58ery1rjmqSR5
Supplier#000007448      yhhpWiJi7EJ6Q5VCaQ
Supplier#000007477      9m9j0wfhWzCvVHxkU,PpAxwSH0h

Supplier#000007509      q8,V6LJRoHJjHcOuSG7aLTMg
Supplier#000007561      rMcFg2530VC
Supplier#000007789      rQ7cUcPrtudOyO3svNSkimqH6qrfWT2Sz

Supplier#000007801      69fi,U1r6enUb
Supplier#000007818      yhhc2CQec Jrvc8zqBi83
Supplier#000007885      u3sicchh5ZpyTUpN1cJKNcAoabIWgY

Supplier#000007918      r,v9mBQ6LoEYyj1
Supplier#000007926      ErzCF80K9Uy
Supplier#000007957      ELwnio14ssoU1 dRyZIL OK3Vtzb
Supplier#000007965      F7Un5lJ7p5hhj
Supplier#000007968
DsF9UlZ2Fo6HXN9aErvyg1ikHoD582HSGZpP
Supplier#000007998      LnASFBfYRFOo9d6d,asBvVq9Lo2P

Supplier#000008168      aOa82a8ZbKCnfDLX
Supplier#000008231      IK7eGw Yj90sTdpsP,vcqWxLB
Supplier#000008243      2AyePMkDqmzVzjGTizXthFLo8h
EiudCMxOmIIG
Supplier#000008275      BlbNDfWg,gpXKQlLN
Supplier#000008323      75I18sZmASwm
POeheRMdj9tmpyeQ,BfCXN5BIAb
Supplier#000008366
h778cEj14BuW9OEKlvPTWq4iwASR6EBBXN7zeS8
Supplier#000008423      RQhKnkAhR0DAr3Ix4Q1weMMn00hNe
Kq
Supplier#000008480      4sSDA4ACReklNjEm5T6b
Supplier#000008532
Uc29q4,5xVdDOF87UZrxhr4xWS0ihEUXuh
Supplier#000008595      MH0iB73GQ3z UW3O DbCbqmc

Supplier#000008610
SgVgP90vP452sUNTgzL9zKwXHXAzV6tV
Supplier#000008705      aE,trRNdPx,4yinTD9O3DebDIp
Supplier#000008742      HmPlQEzKCPEcTUL14,kKq
Supplier#000008841      I 85Lu1sekbg2xrSIzm0
Supplier#000008895      2cH4okfaLSZTTg8sKRbbJQxkmeFu2Esj

Supplier#000008967      2kwEHyMG
7FwozNImAUE6mH0hYtqYculJM
Supplier#000008972      w2vF6 D5YZO3visPXsqVfLADTK

Supplier#000009032      qK,trB6Sdy4Dz1BRUFNy
Supplier#000009147      rOAuryHxpZ9eOvx
Supplier#000009252      F7cZaPUHwh1 ZKyj3xmAVWC1XdP
ue1p5m,i
Supplier#000009278      RqYTzgxj93CLX 0mcYfCENOefD

Supplier#000009327      uoqMdf7e7Gj9dbQ53
Supplier#000009430      igRqmneFt
Supplier#000009567      r4Wfx4c3xsEAjcGj71HHZByornl
D9vrztXlv4
Supplier#000009601      51m637bO,Rw5DnHWFUvLacRx9

Supplier#000009709      rRnCbHYgDgl9PZYnyWKVYSUW0vKg

Supplier#000009753      wLhVEcRmd7PkJF4FBnGK7Z
Supplier#000009796      z,y4Idmr15DOvPUqYG
Supplier#000009799      4wNjXGa4OKWl
Supplier#000009811      E3iuyq7UnZxU7oPZIe2Gu6
Supplier#000009812
APFRMy3lCbgFga53n5t9DxzFPQPgnjrGt32
Supplier#000009862      rJzweWeN58
Supplier#000009868      ROjGgx5gvtkmnUUoeyy7v
Supplier#000009869
ucLqxzrpBTRMewGSM29t0rNTM30g1Tu3Xgg3mKag
Supplier#000009899      7XdpAHrzr1t,UQFZE
Supplier#000009974      7wJ,J5DKcxSU4Kp1cQLpbcAvB5AsvKT


(204 row(s) affected)
```

Validation_Query_21.sql

```
-- using default substitutions

/* TPC_H  Query 21 - Suppliers Who Kept Orders Waiting */

SELECT  TOP 100
        S_NAME,
        COUNT(*)            AS NUMWAIT
FROM    SUPPLIER,
        LINEITEM L1,
        ORDERS,
        NATION
WHERE   S_SUPPKEY                   = L1.L_SUPPKEY
AND
        O_ORDERKEY                  = L1.L_ORDERKEY
AND
        O_ORDERSTATUS               = 'F'
AND
        L1.L_RECEIPTDATE            > L1.L_COMMITDATE
AND
        EXISTS  (           SELECT  *
                            FROM    LINEITEM L2
                            WHERE   L2.L_ORDERKEY   =
L1.L_ORDERKEY   AND
                                    L2.L_SUPPKEY    <>
L1.L_SUPPKEY
                    )       AND
        NOT EXISTS          (       SELECT  *
                                    FROM    LINEITEM L3
                                    WHERE
L3.L_ORDERKEY               = L1.L_ORDERKEY
AND

L3.L_SUPPKEY                <> L1.L_SUPPKEY
AND

L3.L_RECEIPTDATE            > L3.L_COMMITDATE
                            )       AND
        S_NATIONKEY         = N_NATIONKEY   AND
        N_NAME              = 'SAUDI ARABIA'
GROUP   BY      S_NAME
ORDER   BY      NUMWAIT             DESC,
                S_NAME
```

Validation_Query_21.out

```
S_NAME                    NUMWAIT
------------------------- -----------
Supplier#000002829        20
Supplier#000005808        18
Supplier#000000262        17
```

```
Supplier#000000496      17
Supplier#000002160      17
Supplier#000002301      17
Supplier#000002540      17
Supplier#000003063      17
Supplier#000005178      17
Supplier#000008331      17
Supplier#000002005      16
Supplier#000002095      16
Supplier#000005799      16
Supplier#000005842      16
Supplier#000006450      16
Supplier#000006939      16
Supplier#000009200      16
Supplier#000009727      16
Supplier#000000486      15
Supplier#000000565      15
Supplier#000001046      15
Supplier#000001047      15
Supplier#000001161      15
Supplier#000001336      15
Supplier#000001435      15
Supplier#000003075      15
Supplier#000003335      15
Supplier#000005649      15
Supplier#000006027      15
Supplier#000006795      15
Supplier#000006800      15
Supplier#000006824      15
Supplier#000007131      15
Supplier#000007382      15
Supplier#000008913      15
Supplier#000009787      15
Supplier#000000633      14
Supplier#000001960      14
Supplier#000002323      14
Supplier#000002490      14
Supplier#000002993      14
Supplier#000003101      14
Supplier#000004489      14
Supplier#000005435      14
Supplier#000005583      14
Supplier#000005774      14
Supplier#000007579      14
Supplier#000008180      14
Supplier#000008695      14
Supplier#000009224      14
Supplier#000000357      13
Supplier#000000436      13
Supplier#000000610      13
Supplier#000000788      13
Supplier#000000889      13
Supplier#000001062      13
Supplier#000001498      13
Supplier#000002056      13
Supplier#000002312      13
Supplier#000002344      13
Supplier#000002596      13
Supplier#000002615      13
Supplier#000002978      13
Supplier#000003048      13
Supplier#000003234      13
Supplier#000003727      13
Supplier#000003806      13
Supplier#000004472      13
Supplier#000005236      13
Supplier#000005906      13
Supplier#000006241      13
Supplier#000006326      13
Supplier#000006384      13
Supplier#000006394      13
Supplier#000006624      13
Supplier#000006629      13
Supplier#000006682      13
Supplier#000006737      13
Supplier#000006825      13
Supplier#000007021      13
Supplier#000007417      13
Supplier#000007497      13
Supplier#000007602      13
Supplier#000008134      13
Supplier#000008234      13
Supplier#000009435      13
Supplier#000009436      13
Supplier#000009564      13
Supplier#000009896      13
Supplier#000000379      12
Supplier#000000673      12
Supplier#000000762      12
Supplier#000000811      12
Supplier#000000821      12
Supplier#000001337      12
Supplier#000001916      12
Supplier#000001925      12
Supplier#000002039      12
Supplier#000002357      12
Supplier#000002483      12

(100 row(s) affected)
```

Validation_Query_22.sql

```
-- using default substitutions

/* TPC_H  Query 22 - Global Sales Opportunity */

SELECT   CNTRYCODE,
         COUNT(*)            AS NUMCUST,
         SUM(C_ACCTBAL) AS TOTACCTBAL
FROM     (          SELECT   SUBSTRING(C_PHONE,1,2)    AS
CNTRYCODE,
                             C_ACCTBAL
                   FROM     CUSTOMER
                   WHERE    SUBSTRING(C_PHONE,1,2)    IN
('13', '31', '23', '29', '30', '18', '17') AND
                             C_ACCTBAL                  >
(          SELECT   AVG(C_ACCTBAL)

           FROM     CUSTOMER

           WHERE    C_ACCTBAL          > 0.00      AND

                    SUBSTRING(C_PHONE,1,2)    IN
('13', '31', '23', '29', '30', '18', '17')

)          AND
                             NOT EXISTS         (
SELECT   *

FROM     ORDERS

WHERE    O_CUSTKEY          = C_CUSTKEY
                                                    )
         )          AS CUSTSALE
GROUP    BY         CNTRYCODE
```

```
ORDER    BY          CNTRYCODE
```

Validation_Query_22.out

```
CNTRYCODE NUMCUST     TOTACCTBAL
--------- ----------- ----------------------
13        888         6737713.9900
17        861         6460573.7200
18        964         7236687.4000
23        892         6701457.9500
29        948         7158866.6300
30        909         6808436.1300
31        922         6806670.1800

(7 row(s) affected)
```

Appendix D: Seed and Query Substitution Parameters

Substitution Parameters 00
Seed value: 311154310

1 103
2 33 COPPER AFRICA
3 BUILDING 1995-03-05 1995-03-05
4 1993-10-01 1993-10-01
5 ASIA 1993-01-01 1993-01-01
6 1993-01-01 1993-01-01 .06 24
7 KENYA IRAQ IRAQ KENYA
8 IRAQ MIDDLE EAST STANDARD PLATED COPPER
9 blush
10 1993-09-01 1993-09-01
11 VIETNAM 0.0000010000 VIETNAM
12 SHIP AIR 1993-01-01
13 special packages
14 1993-11-01 1993-11-01
15 1993-11-01 1993-11-01
16 Brand#35 SMALL PLATED 25, 16, 32, 18, 26, 49, 13, 9
17 Brand#52 LG JAR
18 313
19 Brand#22 3 Brand#35 11 Brand#35 25
20 frosted 1995-01-01 RUSSIA
21 IRAQ
22 '16', '17', '34', '23', '15', '11', '31', '16', '17', '34', '23', '15', '11', '31'

Substitution Parameters 01
Seed value: 311154311

1 111
2 21 STEEL EUROPE
3 HOUSEHOLD 1995-03-22 1995-03-22
4 1996-05-01 1996-05-01
5 EUROPE 1993-01-01 1993-01-01
6 1993-01-01 1993-01-01 0.03 24
7 ETHIOPIA CANADA CANADA ETHIOPIA
8 CANADA AMERICA STANDARD ANODIZED COPPER
9 azure
10 1994-06-01 1994-06-01
11 INDONESIA 0.0000010000 INDONESIA
12 FOB AIR 1994-01-01
13 special packages
14 1994-03-01 1994-03-01
15 1996-06-01 1996-06-01
16 Brand#15 LARGE BURNISHED 19, 29, 8, 47, 34, 38, 2, 11
17 Brand#54 LARGE CAN
18 314
19 Brand#24 8 Brand#13 12 Brand#35 21
20 puff 1993-01-01 1993-01-01 JAPAN
21 CANADA
22 '22', '11', '12', '28', '14', '19', '16', '22', '11', '12', '28', '14', '19', '16'

Substitution Parameters 02
Seed value: 311154312

1 119
2 9 BRASS AMERICA
3 'BUILDING' '1995-03-07' '1995-03-07'
4 1994-02-01 1994-02-01
5 MIDDLE EAST 1993-01-01 1993-01-01
6 1993-01-01 1993-01-01 .09 25
7 RUSSIA SAUDI ARABIA SAUDI ARABIA RUSSIA
8 SAUDI ARABIA MIDDLE EASTPROMO BRUSHED TIN
9 wheat
10 1993-03-01 1993-03-01
11 'RUSSIA' 0.0000010000 'RUSSIA'
12 TRUCK AIR 1994-01-01
13 pending requests
14 1994-06-01 1994-06-01
15 1994-02-01 1994-02-01
16 Brand#55 STANDARD ANODIZED 22, 48, 41, 5, 30, 15, 11, 28
17 Brand#51 MED BOX
18 312
19 Brand#31 3 Brand#51 13 Brand#24 28
20 chartreuse 1997-01-01 BRAZIL
21 SAUDI ARABIA
22 '12', '19', '21', '14', '17', '30', '10', '12', '19', '21', '14', '17', '30', '10'

Substitution Parameters 03
Seed value: 311154313

1 66
2 47 TIN EUROPE
3 HOUSEHOLD 1995-03-24 1995-03-24
4 1996-09-01 1996-09-01
5 AFRICA 1994-01-01 1994-01-01
6 1994-01-01 1994-01-01 .06 24
7 KENYA JAPAN JAPAN KENYA
8 JAPAN ASIA PROMO PLATED TIN
9 steel
10 1993-12-01 1993-12-01
11 IRAN 0.0000010000 IRAN
12 'RAIL','FOB' 1995-01-01
13 pending requests
14 1994-09-01 1994-09-01
15 1996-09-01 1996-09-01
16 Brand#35 MEDIUM PLATED, 25, 11, 14, 38, 50, 31, 33, 28
17 Brand#53 MED JAR
18 313
19 Brand#33 8 Brand#34 14 Brand#23 25
20 mint 1995-01-01 PERU
21 JAPAN
22 '27', '17', '11', '23', '20', '21', '26' '27', '17', '11', '23', '20', '21', '26'

Substitution Parameters 04
Seed value: 311154314

1 74
2 35 COPPER AMERICA
3 AUTOMOBILE 1995-03-9 1995-03-9
4 1994-06-01 1994-06-01
5 ASIA 1994-01-01 1994-01-01
6 1994-01-01 1994-01-01 .04 24
7 FRANCE EGYPT EGYPT FRANCE
8 EGYPT MIDDLE EAST PROMO ANODIZED TIN
9 sienna
10 1994-10-01 1994-10-01
11 UNITED STATES 0.0000010000 UNITED STATES

12 'AIR','RAIL' 1994-01-01
13 pending requests
14 1994-12-01 1994-12-01
15 1994-06-01 1994-06-01
16 Brand#15 ECONOMY POLISHED 29, 22, 39, 14, 12, 18, 9, 34
17 Brand#54 MED CAN
18 315
19 Brand#35 4 Brand#12 15 Brand#22 21
20 white 1993-01-01 FRANCE
21 EGYPT
22 '26', '33', '18', '11', '32', '21', '30', '26', '33', '18', '11', '32', '21', '30'

Substitution Parameters 05
Seed value: 311154315

1 82
2 22 STEEL MIDDLE EAST
3 HOUSEHOLD 1995-03-26 1995-03-26
4 1997-01-01 1997-01-01
5 EUROPE 1994-01-01 1994-01-01
6 1994-01-01 1994-01-01 0.09 25
7 UNITED KINGDOM VIETNAMVIETNAM UNITED KINGDOM
8 VIETNAM ASIA ECONOMY POLISHED TIN
9 rosy
10 1993-07-01 1993-07-01
11 IRAQ 0.0000010000 IRAQ
12 'REG AIR', 'RAIL' 1995-01-01
13 pending requests
14 1995-03-01 1995-03-01
15 1997-01-01 1997-01-01
16 Brand#55 SMALL ANODIZED 32, 34, 13, 43, 33, 27, 4, 30
17 Brand#51 JUMBO BOX
18 312
19 Brand#43 9 Brand#55 16 Brand#11 28
20 indian 1997-01-01 VIETNAM
21 'VIETNAM'
22 '31', '27', '19', '10', '24', '28', '23', '31', '27', '19', '10', '24', '28', '23'

Appendix E: Refresh Function Source Code

```
-- File:    CREATERF1PROC.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

if exists (select name from sysobjects where name =
'RF1_%INSERT_SEGMENT%')
              drop procedure RF1_%INSERT_SEGMENT%
GO

--
-- Create a stored RefreshInsert procedure which will catch the deadlock
-- victim abort and restart the insert transaction.
--
CREATE PROCEDURE RF1_%INSERT_SEGMENT%
      @insert_sets INTEGER
AS
BEGIN

DECLARE @min_orderkey INTEGER
DECLARE @max_orderkey INTEGER
DECLARE @range INTEGER
DECLARE @max_set INTEGER
DECLARE @SQLstring NVARCHAR(255)
DECLARE @updateset INTEGER

--
-- Get the current update set value
--
SELECT @updateset=updateset from TPCH_AUX_TABLE

--
-- Delete any previous columns from the insert table
--
DELETE FROM NEWORDERS_%INSERT_SEGMENT%
DELETE FROM NEWLINEITEM_%INSERT_SEGMENT%

--
-- Generate an SQL statement inserting the current updateset value into
--  the command.  Next execute the statement to bulk load the new lineitem
-- insert values.
--
SET @SQLstring='bulk insert
%DBNAME%..NEWLINEITEM_%INSERT_SEGMENT% from
"%RF_FLATFILE_DIR%\Lineitem.tbl.u'+ RTRIM(Convert(char,@updateset)) +
 '.%INSERT_SEGMENT%" with (FieldTerminator = "|", RowTerminator
="|\n",tablock)'
EXEC sp_executesql @SQLstring

--
-- Generate an SQL statement inserting the current updateset value into
-- the command.  Next execute the statement to bulk load the new order
-- insert values.
--
SET @SQLstring='bulk insert
%DBNAME%..NEWORDERS_%INSERT_SEGMENT% from
"%RF_FLATFILE_DIR%\Orders.tbl.u'+ RTRIM(Convert(char,@updateset)) +
'.%INSERT_SEGMENT%" with (FieldTerminator = "|", RowTerminator
="|\n",tablock)'
EXEC sp_executesql @SQLstring

--
-- Obtain minimum and maximum order key and compute the range of each
-- set to be inserted into the ORDERS and LINEITEM tables.
--
SELECT @min_orderkey=MIN(O_ORDERKEY) FROM
NEWORDERS_%INSERT_SEGMENT%
SELECT @max_orderkey=MAX(O_ORDERKEY) FROM
NEWORDERS_%INSERT_SEGMENT%
SET @range = (@max_orderkey - @min_orderkey) / @insert_sets

--
-- This handles the case when the max-min/insert_sets is less that 1
--
IF @range = 0
-- BEGIN
              SET @range = (@max_orderkey - @min_orderkey) / 1
-- END

--
-- Loop through the order keys only inserting a sets into the
-- ORDERS and LINTEITEM tables
--
SET @max_set = @min_orderkey - 1
WHILE @max_set < @max_orderkey
BEGIN
 --
 -- Set the range from min_orderkey to max_set
 --
 SET @max_set = @min_orderkey + @range
 if @max_set > @max_orderkey
   SET @max_set = @max_orderkey + 1

 --
 -- Insert into ORDERS and LINEITEM tables
 --
 INSERT_TRANS:
 begin transaction
    insert into ORDERS SELECT * FROM
NEWORDERS_%INSERT_SEGMENT%
         WHERE O_ORDERKEY >= @min_orderkey AND O_ORDERKEY <
@max_set
    insert into LINEITEM SELECT * FROM
NEWLINEITEM_%INSERT_SEGMENT%
         WHERE L_ORDERKEY >= @min_orderkey AND L_ORDERKEY <
@max_set
 commit transaction

 --
 -- If deadlock victim abort then restart the transaction
 --
 if (@@error = 1205)
  BEGIN
   print 'Insert deadlock - restarting RF1_%INSERT_SEGMENT% transaction'
   rollback transaction
   GOTO INSERT_TRANS
  END

 --
 -- Move min_orderkey to start of next insert set
 --
 SET @min_orderkey = @max_set
END
END

GO
-- File:    CREATERF2PROC.SQL
--          Microsoft TPC-H Benchmark Kit Ver. 1.00
--          Copyright Microsoft, 1999
--

if exists (select name from sysobjects where name =
'RF2_%DELETE_SEGMENT%')
              drop procedure RF2_%DELETE_SEGMENT%
GO

--
-- Create a stored Refresh Delete procedure which will catch the deadlock
-- victim abort and restart the delete transaction.
--
CREATE PROCEDURE RF2_%DELETE_SEGMENT%
      @delete_sets INTEGER
AS
BEGIN

DECLARE @min_orderkey INTEGER
DECLARE @max_orderkey INTEGER
DECLARE @range INTEGER
DECLARE @max_set INTEGER
DECLARE @SQLstring NVARCHAR(255)
```

```
DECLARE @updateset INTEGER
--
-- Get the current update set value
--
SELECT @updateset=updateset from TPCH_AUX_TABLE

--
--  Delete any existing index(s) on the temporary table(s)
--
if exists (select name from sysindexes where name =
'OLDORDERS_%DELETE_SEGMENT%_idx')
        drop index

OLDORDERS_%DELETE_SEGMENT%.OLDORDERS_%DELETE_SEGME
NT%_idx

--
--  Delete any previous columns from the delete table
--
DELETE FROM OLDORDERS_%DELETE_SEGMENT%

--
--  Generate an SQL statement inserting the current updateset value into
--  the command.  Next execute the statement to bulk load the old order
--  delete values
--
SET @SQLstring='bulk insert
%DBNAME%..OLDORDERS_%DELETE_SEGMENT% from
"%RF_FLATFILE_DIR%\Delete.u' + RTRIM(Convert(char,@updateset)) +
'.%DELETE_SEGMENT%" with (FieldTerminator = "|", RowTerminator
="\n",tablock)'
EXEC sp_executesql @SQLstring
--
--  Create index on OLDORDERS
--
SET @SQLstring='create unique index
OLDORDERS_%DELETE_SEGMENT%_idx on
OLDORDERS_%DELETE_SEGMENT% (O_ORDERKEY)'
EXEC sp_executesql @SQLstring
--
--  Obtain minimum and maximum order key and compute the
--  range of each delete set
--
SELECT @min_orderkey=MIN(O_ORDERKEY) FROM
OLDORDERS_%DELETE_SEGMENT%
SELECT @max_orderkey=MAX(O_ORDERKEY) FROM
OLDORDERS_%DELETE_SEGMENT%
SET @range = (@max_orderkey - @min_orderkey) / @delete_sets

--
--  This handles the case when the max-min/delete_sets is less that 1
--
IF @range = 0
-- BEGIN
                SET @range = (@max_orderkey - @min_orderkey) / 1
-- END

--
--  Loop through the order keys only deleting sets from orders
--  and lineitem tables
--
SET @max_set = @min_orderkey - 1
WHILE @max_set < @max_orderkey
BEGIN
  --
  -- Set the range from min_orderkey to max_set
  --
  SET @max_set = @min_orderkey + @range
  if @max_set > @max_orderkey
     SET @max_set = @max_orderkey + 1

  --
  --  Delete from ORDERS and LINEITEM table
  --
  DELETE_TRANS:
  begin transaction
    delete from ORDERS where O_ORDERKEY in
        (select * from OLDORDERS_%DELETE_SEGMENT% WHERE
O_ORDERKEY >= @min_orderkey AND O_ORDERKEY < @max_set)
    delete from LINEITEM where L_ORDERKEY in
        (select * from OLDORDERS_%DELETE_SEGMENT% WHERE
O_ORDERKEY >= @min_orderkey AND O_ORDERKEY < @max_set)

  commit transaction
  --
  --  If deadlock victim abort then restart the transaction
  --
  if (@@error = 1205)
    BEGIN
      print 'Delete deadlock - restarting RF2_%DELETE_SEGMENT% transaction'
      rollback transaction
      GOTO DELETE_TRANS
    END
  --
  --  Move min_orderkey to start of next delete set
  --
  SET @min_orderkey = @max_set

  END
END

GO
```

Appendix F: StepMaster Source Code

Att_workspaces

workspace_id workspace_name
archived_flag

7 ACID - TPC-H Kit V1.01d (Spec Revision 1.2.0)
0

Workspace_parameters

workspace_id = 7 parameter_id 118 parameter_type
0

parameter_name DUR_RUN_ID

parameter_valu 464

workspace_id = 7 parameter_id 117 parameter_type
3
parameter_name OUTPUT_DIR

parameter_valu D:\MSTPCH~1.101\OUTPUT\535

workspace_id = 7 parameter_id 114 parameter_type
3

parameter_name RUN_ID
parameter_valu 535

workspace_id = 7 parameter_id 77 parameter_type
0

parameter_name KIT_DIR

parameter_valu D:\MSTPCH.101D

workspace_id = 7 parameter_id 76 parameter_type
0
parameter_name TOOLS_DIR

parameter_valu %KIT_DIR%\tools

workspace_id = 7 parameter_id 75 parameter_type
0

parameter_name SERVER_NAME
parameter_valu tpcha

workspace_id = 7 parameter_id 74 parameter_type
0

parameter_name NUM_STREAMS

parameter_valu 6

workspace_id = 7 parameter_id 73 parameter_type
0
parameter_name DEFAULT_DIR

parameter_valu %KIT_DIR%\OUTPUT

Friday, March 09, 2001
Page 1 of 130

workspace_id = 7 parameter_id 72 parameter_type
0

parameter_name DBNAME
parameter_valu tpch1g

workspace_id = 7 parameter_id 71 parameter_type
0

parameter_name ACID_DIR

parameter_valu %KIT_DIR%\ACID

Connection_dtls

workspac connection_na connection_name
connection_string_name connection
e_id me_id
_type

7 49 Static_Connection_to_DB_10
DBCONNECTION 1

7 48 Static_Connection_to_DB_9
DBCONNECTION 1

7 47 Static_Connection_to_DB_8
DBCONNECTION 1

7 46 Static_Connection_to_DB_7
DBCONNECTION 1

7 45 Static_Connection_to_DB_6
DBCONNECTION 1

7 44 Static_Connection_to_DB_5
DBCONNECTION 1

7 43 Static_Connection_to_DB_4
DBCONNECTION 1

7 42 Static_Connection_to_DB_3
DBCONNECTION 1

7 41 Static_Connection_to_DB_2
DBCONNECTION 1

7 40 Static_Connection_to_DB_1
DBCONNECTION 1

7 24 Dynamic_Connection_to_DB
DBCONNECTION 2

Friday, March 09, 2001
Page 2 of 130

Workspace_connections

workspace_i 7

connection_i 8
connection_name DBCONNECTION

connection_valu DRIVER=SQL

Server;SERVER=%SERVER_NAME%;UID=sa;PWD=;DATABASE
=%DBNAME%;

descripti
no_count_displa 0

no_execut 0

parse_query_onl 0
ANSI_quoted_identifie 0

ANSI_nulls -1

show_query_plan 0
show_stats_tim 0

show_stats_i 0

parse_odbc_msg_prefix -1
row_count 0

tsql_batch_separa GO

query_time_ou 0
server_languag (Default)

character_translat -1

regional_setti 0

Att_steps

workspace_id = 7

step_label = Setup ACID Tables and Values step_id =
507 global_flag = 0

sequence_no = 1 step_level = 0 parent_step_id =
0 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 14.0

start_directory
parent_version_no = 0.0
=
step_text =

Friday, March 09, 2001
Page 3 of 130

step_label = TPC-H Atomicity step_id =
508 global_flag = 0

sequence_no = 2 step_level = 0 parent_step_id =
0 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 8.0

start_directory
parent_version_no = 0.0
=
step_text =

step_label = TPC-H Isolation step_id = 509
global_flag = 0

sequence_no = 3 step_level = 0 parent_step_id =
0 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 7.0

start_directory
parent_version_no = 0.0
=
step_text =

step_label = TPC-H Consistency step_id =
510 global_flag = 0

sequence_no = 4 step_level = 0 parent_step_id =
0 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

```
 step_file_name =
version_no =  4.0

 start_directory
parent_version_no =     0.0
 =
 step_text =


 step_label = TPC-H Durability (Pre-Test Components)
step_id =     511      global_flag =    0

 sequence_no =        5  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  6.0

 start_directory
parent_version_no =     0.0
 =
 step_text =


 Friday, March 09, 2001
     Page 4 of 130


 step_label = TPC-H Durability (Post-Test Components)
step_id =     512      global_flag =    0

 sequence_no =        6  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  4.0

 start_directory
parent_version_no =     0.0
 =
 step_text =


 step_label = Cleanup ACID Tables and Stored Procs
step_id =     513      global_flag =    0

 sequence_no =        7  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory
parent_version_no =     0.0
 =
 step_text =


 step_label = Create ACID History Tables                    step_id =
514      global_flag =    0

 sequence_no =        1  step_level =       1       parent_step_id =
507 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =    %ACID_DIR%\acidproc\acid_hist.sql
                version_no =  1.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     14.0
 =
 step_text =


 step_label = Create ACID Values Table                      step_id =
515      global_flag =    0

 sequence_no =        2  step_level =       1       parent_step_id =
507 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =    %ACID_DIR%\acidproc\acid_table.sql
                version_no =  1.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     14.0
 =
 step_text =


 Friday, March 09, 2001
     Page 5 of 130


 step_label = Create ACID Stored Procedures                 step_id =
  516      global_flag =    0

 sequence_no =        3  step_level =       1       parent_step_id =
507 enabled_flag =      -1

 iterator_name =     PROC_NAME                  degree_parallelism
1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  3.0

 start_directory     %ACID_DIR%
parent_version_no =     14.0
 =
```

step_text = isql -Usa -P -S%SERVER_NAME% -d%DBNAME% -iacidproc\%PROC_NAME%

step_label = Populate ACID Values Table step_id = 517 global_flag = 0

sequence_no = 4 step_level = 1 parent_step_id = 507 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2 failure_details =

step_file_name = version_no = 1.0

start_directory %ACID_DIR% parent_version_no = 14.0 =
step_text = isql -Usa -P -S%SERVER_NAME% -d%DBNAME% -w 128 -Q"tpch_acid_values %NUM_STREAMS%"

step_label = Execute Atomicity Rollback Test step_id = 519 global_flag = 0

sequence_no = 2 step_level = 1 parent_step_id = 508 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = %ACID_DIR%\atom\scripts\atom_r.sql version_no = 4.0

start_directory Static_Connection_to_DB_1 parent_version_no = 8.0 =
step_text =

step_label = Clear any Existing Semaphores step_id = 520 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id = 509 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2 failure_details =

step_file_name = version_no = 0.0

start_directory %TOOLS_DIR% parent_version_no = 7.0 =
step_text = %TOOLS_DIR%\Utility\killsemaphore.exe

step_label = Isolation 1 step_id = 521 global_flag = 0

sequence_no = 2 step_level = 1 parent_step_id = 509 enabled_flag = -1

iterator_name = degree_parallelism 3

execution_mechanism = 0 continuation_criteria = 0 failure_details =

step_file_name = version_no = 1.0

start_directory parent_version_no = 7.0 =
step_text =

step_label = Isolation 2 step_id = 522 global_flag = 0

sequence_no = 3 step_level = 1 parent_step_id = 509 enabled_flag = -1

iterator_name = degree_parallelism 3

execution_mechanism = 0 continuation_criteria = 0 failure_details =

step_file_name = version_no = 0.1

start_directory parent_version_no = 7.0 =
step_text =

step_label = Isolation 3 step_id = 523 global_flag = 0

sequence_no = 4 step_level = 1 parent_step_id = 509 enabled_flag = -1

iterator_name = degree_parallelism 3

execution_mechanism = 0 continuation_criteria = 0 failure_details =

step_file_name = version_no = 1.1

start_directory parent_version_no = 7.0 =
step_text =

step_label = Isolation 4 step_id = 524 global_flag = 0

sequence_no = 5 step_level = 1 parent_step_id = 509 enabled_flag = -1

iterator_name = degree_parallelism 3

```
 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  1.1

 start_directory
parent_version_no =     7.0
 =
 step_text =


 Friday, March 09, 2001
     Page 7 of 130


 step_label = Isolation 5                                step_id =     525
global_flag =    0

 sequence_no =        6  step_level =       1        parent_step_id =
509 enabled_flag =      -1

 iterator_name =                              degree_parallelism  3

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  1.1

 start_directory
parent_version_no =     7.0
 =
 step_text =


 step_label = Isolation 6                                step_id =     526
global_flag =    0

 sequence_no =        7  step_level =       1        parent_step_id =
509 enabled_flag =      -1

 iterator_name =                              degree_parallelism  4

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  1.1

 start_directory
parent_version_no =     7.0
 =
 step_text =



 step_label = Setup Consistency Directory                step_id =
527      global_flag =    0

 sequence_no =        2  step_level =       1        parent_step_id =
510 enabled_flag =      -1

 iterator_name =     STREAM                   degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =
```

```
 step_file_name =
version_no =  7.0

 start_directory     %ACID_DIR%
parent_version_no =     4.0
 =
 step_text =  ::
           :: Build header for each consistency stream output file
           ::
           ECHO CONSISTENCY STREAM %STREAM% OUTPUT
> CONS\Stream%STREAM%.out
           ECHO ---------------------------- >>
CONS\Stream%STREAM%.out




 Friday, March 09, 2001
     Page 8 of 130


 step_label = Truncate HISTORY_ACI Table                 step_id
=     528      global_flag =    0

 sequence_no =        3  step_level =       1        parent_step_id =
510 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  2.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     4.0
 =
 step_text =  --
           -- This worker will clear the HISTORY_ACI table used
during the durability process
           --
           TRUNCATE TABLE HISTORY_ACI


 step_label = Execute Consistency Test 1                 step_id =
529      global_flag =    0

 sequence_no =        4  step_level =       1        parent_step_id =
510 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =    %ACID_DIR%\cons\scripts\con1.sql
               version_no =  3.0

 start_directory     Static_Connection_to_DB_1
  parent_version_no =     4.0
 =
 step_text =


 step_label = ACID Transaction Streams                   step_id =
530      global_flag =    0
```

```
 sequence_no =        5  step_level =       1        parent_step_id =
510 enabled_flag =      -1

 iterator_name =                                degree_parallelism
%NUM_STREAMS%

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  1.8

 start_directory
parent_version_no =     4.0
 =
 step_text =


 step_label = Execute Consistency Test 2                        step_id =
531      global_flag =    0

 sequence_no =        6  step_level =       1        parent_step_id =
510 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =    %ACID_DIR%\cons\scripts\con2.sql
               version_no =  3.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     4.0
 =
 step_text =


 Friday, March 09, 2001
     Page 9 of 130


 step_label = Setup Durability Directory                        step_id =
532      global_flag =    0

 sequence_no =        2  step_level =       1        parent_step_id =
511 enabled_flag =      -1

 iterator_name =     STREAM                     degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %ACID_DIR%
parent_version_no =     6.0
 =
 step_text =  ::
           :: Build header for each durability stream output file
           ::
           ECHO DURABILITY STREAM %STREAM% OUTPUT >
DUR\Stream%STREAM%.out
           ECHO ---------------------------- >>
DUR\Stream%STREAM%.out

 step_label = Truncate HISTORY_DUR Table                        step_id
=     533      global_flag =    0

 sequence_no =        3  step_level =       1        parent_step_id =
511 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     6.0
 =
 step_text =  --
           -- This worker will clear the HISTORY_DUR table used
during the durability process
           --
           TRUNCATE TABLE HISTORY_DUR


 step_label = Execute Durability Verification Test 1            step_id =
   534      global_flag =    0

 sequence_no =        4  step_level =       1        parent_step_id =
511 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =    D:\mstpch.101D\acid\dur\scripts\dur_verify1.sql
               version_no =  1.0

 start_directory     Static_Connection_to_DB_1
  parent_version_no =     6.0
 =
 step_text =


 Friday, March 09, 2001
    Page 10 of 130


 step_label = Durability Streams                                step_id =     535
    global_flag =    0

 sequence_no =        5  step_level =       1        parent_step_id =
511 enabled_flag =      -1

 iterator_name =                                degree_parallelism
%NUM_STREAMS%

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.2

 start_directory
parent_version_no =     6.0
 =
 step_text =
```

```
step_label = Execute Durability Verification Test 2              step_id =
  536     global_flag =    0

sequence_no =        1  step_level =      1       parent_step_id =
512 enabled_flag =     -1

iterator_name =                                  degree_parallelism  1

execution_mechanism =     1                      continuation_criteria = 2
      failure_details =

step_file_name =    %ACID_DIR%\dur\scripts\dur_verify2.sql
                version_no =  4.0

start_directory     Dynamic_Connection_to_DB
  parent_version_no =     4.0
=
step_text =


step_label = Drop ACID History Table                             step_id =
537     global_flag =    0

sequence_no =        1  step_level =      1       parent_step_id =
513 enabled_flag =     -1

iterator_name =                                  degree_parallelism  1

execution_mechanism =     1                      continuation_criteria = 2
      failure_details =

step_file_name =
version_no =  0.0

start_directory     Dynamic_Connection_to_DB
  parent_version_no =     0.0
=
step_text =  if exists ( select name from sysobjects where name =
'HISTORY_ACI' )
           drop table HISTORY_ACI
           GO

           if exists ( select name from sysobjects where name =
'HISTORY_DUR' )
           drop table HISTORY_DUR
           GO


Friday, March 09, 2001
   Page 11 of 130


step_label = Drop ACID Values Table                              step_id =
538     global_flag =    0

sequence_no =        2  step_level =      1       parent_step_id =
513 enabled_flag =     -1

iterator_name =                                  degree_parallelism  1

execution_mechanism =     1                      continuation_criteria = 2
      failure_details =

step_file_name =
version_no =  0.0

start_directory     Dynamic_Connection_to_DB
  parent_version_no =     0.0
=
step_text =  if exists ( select name from sysobjects where name =
'acid_values' )
           drop table acid_values


step_label = Drop ACID Stored Procedures                         step_id =
  539     global_flag =    0

sequence_no =        3  step_level =      1       parent_step_id =
513 enabled_flag =     -1

iterator_name =                                  degree_parallelism  1

execution_mechanism =     1                      continuation_criteria = 2
      failure_details =

step_file_name =
version_no =  0.0

start_directory     Dynamic_Connection_to_DB
  parent_version_no =     0.0
=
step_text =  if exists ( select name from sysobjects where name =
'acid_tran_com' )
                drop procedure acid_tran_com
           GO
           if exists ( select name from sysobjects where name =
'acid_dur_tran_com' )
                drop procedure acid_dur_tran_com
           GO
           if exists ( select name from sysobjects where name =
'tran_w_com' )
                drop procedure tran_w_com
           GO
           if exists ( select name from sysobjects where name =
'tran_w_rb' )
                drop procedure tran_w_rb
           GO
           if exists ( select name from sysobjects where name =
'iso3_ver' )
                drop procedure iso3_ver
           GO
           if exists ( select name from sysobjects where name =
'iso5_parts' )
                drop procedure iso5_parts
           GO
           if exists ( select name from sysobjects where name =
'iso_query' )
                drop procedure iso_query
           GO
           if exists ( select name from sysobjects where name =
'iso6_tran_w_com' )
                drop procedure iso6_tran_w_com
           GO


Friday, March 09, 2001
   Page 12 of 130


step_label = Prepare Isolation 1 Output                          step_id =
540     global_flag =    0
```

```
 sequence_no =        1  step_level =       2        parent_step_id =
521 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =    %ACID_DIR%\iso\scripts\iso1_out.cmd
                  version_no =  3.0

 start_directory     %ACID_DIR%
parent_version_no =     1.0
 =
 step_text =


 step_label = Isolation 1_1                                 step_id =     541
 global_flag =    0

 sequence_no =        2  step_level =       2        parent_step_id =
521 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.9

 start_directory
parent_version_no =     1.0
 =
 step_text =


 step_label = Isolation 1_2                                 step_id =     542
 global_flag =    0

 sequence_no =        3  step_level =       2        parent_step_id =
521 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.5

 start_directory
parent_version_no =     1.0
 =
 step_text =



 step_label = Prepare Isolation 2 Output                    step_id =
543      global_flag =    0

 sequence_no =        1  step_level =       2        parent_step_id =
522 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =    %ACID_DIR%\iso\scripts\iso2_out.cmd
                  version_no =  1.0

 start_directory     %ACID_DIR%
parent_version_no =     0.1
 =
 step_text =


 Friday, March 09, 2001
    Page 13 of 130


 step_label = Isolation 2_1                                 step_id =     544
 global_flag =    0

 sequence_no =        2  step_level =       2        parent_step_id =
522 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.2

 start_directory
parent_version_no =     0.1
 =
 step_text =


 step_label = Isolation 2_2                                 step_id =     545
 global_flag =    0

 sequence_no =        3  step_level =       2        parent_step_id =
522 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.1

 start_directory
parent_version_no =     0.1
 =
 step_text =


 step_label = Prepare Isolation 3 Output                    step_id =
546      global_flag =    0

 sequence_no =        1  step_level =       2        parent_step_id =
523 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =    %ACID_DIR%\iso\scripts\iso3_out.cmd
                  version_no =  1.0
```

start_directory %ACID_DIR%
parent_version_no = 1.1
=
step_text =

step_label = Isolation 3_1 step_id = 547
global_flag = 0

sequence_no = 2 step_level = 2 parent_step_id =
523 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 1.1
=
step_text =

step_label = Isolation 3_2 step_id = 548
global_flag = 0

sequence_no = 3 step_level = 2 parent_step_id =
523 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 1.1
=
step_text =

step_label = Prepare Isolation 4 Output step_id =
549 global_flag = 0

sequence_no = 1 step_level = 2 parent_step_id =
524 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\iso4_out.cmd
version_no = 1.0

start_directory %ACID_DIR%
parent_version_no = 1.1
=
step_text =

step_label = Isolation 4_1 step_id = 550
global_flag = 0

sequence_no = 2 step_level = 2 parent_step_id =
524 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 1.1
=
step_text =

step_label = Isolation 4_2 step_id = 551
global_flag = 0

sequence_no = 3 step_level = 2 parent_step_id =
524 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 1.1
=
step_text =

step_label = Prepare Isolation 5 Output step_id =
552 global_flag = 0

sequence_no = 1 step_level = 2 parent_step_id =
525 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\iso5_out.cmd
version_no = 1.0

start_directory %ACID_DIR%
parent_version_no = 1.1
=
step_text =

step_label = Isolation 5_1 step_id = 553
global_flag = 0

sequence_no = 2 step_level = 2 parent_step_id = 525 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 1.1
=
step_text =

step_label = Isolation 5_2 step_id = 554
global_flag = 0

sequence_no = 3 step_level = 2 parent_step_id = 525 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 1.1
=
step_text =

step_label = Prepare Isolation 6 Output step_id = 555 global_flag = 0

sequence_no = 1 step_level = 2 parent_step_id = 526 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\iso6_out.cmd
version_no = 1.0

start_directory %ACID_DIR%
parent_version_no = 1.1
=
step_text =

step_label = Isolation 6_1 step_id = 556
global_flag = 0

sequence_no = 2 step_level = 2 parent_step_id = 526 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 1.1
=
step_text =

step_label = Isolation 6_2 step_id = 557
global_flag = 0

sequence_no = 3 step_level = 2 parent_step_id = 526 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 1.1
=
step_text =

step_label = Isolation 6_3 step_id = 558
global_flag = 0

sequence_no = 4 step_level = 2 parent_step_id = 526 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 1.1
=
step_text =

step_label = Consistency Stream 1 step_id = 559 global_flag = 0

sequence_no = 1 step_level = 2 parent_step_id = 530 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.8

start_directory
parent_version_no = 1.8
=
step_text =

Friday, March 09, 2001
Page 17 of 130

step_label = Consistency Stream 2 step_id =
560 global_flag = 0

sequence_no = 2 step_level = 2 parent_step_id =
530 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.1

start_directory
parent_version_no = 1.8
=
step_text =

step_label = Consistency Stream 3 step_id =
561 global_flag = 0

sequence_no = 3 step_level = 2 parent_step_id =
530 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 2.0

start_directory
parent_version_no = 1.8
=
step_text =

step_label = Consistency Stream 4 step_id =
562 global_flag = 0

sequence_no = 4 step_level = 2 parent_step_id =
530 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 2.0

start_directory
parent_version_no = 1.8
=
step_text =

Friday, March 09, 2001
Page 18 of 130

step_label = Consistency Stream 5 step_id =
563 global_flag = 0

sequence_no = 5 step_level = 2 parent_step_id =
530 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory
parent_version_no = 1.8
=
step_text =

step_label = Consistency Stream 6 step_id =
564 global_flag = 0

sequence_no = 6 step_level = 2 parent_step_id =
530 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory
parent_version_no = 1.8
=
step_text =

step_label = Consistency Stream 7 step_id =
565 global_flag = 0

sequence_no = 7 step_level = 2 parent_step_id =
530 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 1.8
=
step_text =

step_label = Consistency Stream 8 step_id =
566 global_flag = 0

sequence_no = 8 step_level = 2 parent_step_id =
530 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 1.8
=
step_text =

step_label = Consistency Stream 9 step_id =
567 global_flag = 0

sequence_no = 9 step_level = 2 parent_step_id =
530 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 1.8
=
step_text =

step_label = Consistency Stream 10 step_id =
568 global_flag = 0

sequence_no = 10 step_level = 2 parent_step_id =
530 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 1.8
=
step_text =

step_label = Durability Stream 1 step_id =
569 global_flag = 0

sequence_no = 1 step_level = 2 parent_step_id =
535 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.2
=
step_text =

step_label = Durability Stream 2 step_id =
570 global_flag = 0

sequence_no = 2 step_level = 2 parent_step_id =
535 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 0.2
=
step_text =

step_label = Durability Stream 3 step_id =
571 global_flag = 0

sequence_no = 3 step_level = 2 parent_step_id =
535 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.2
=
step_text =

Friday, March 09, 2001
Page 20 of 130

step_label = Durability Stream 4 step_id =
572 global_flag = 0

sequence_no = 4 step_level = 2 parent_step_id =
535 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.2
=
step_text =

step_label = Durability Stream 5 step_id =
573 global_flag = 0

sequence_no = 5 step_level = 2 parent_step_id =
535 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.2
=
step_text =

step_label = Durability Stream 6 step_id =
574 global_flag = 0

sequence_no = 6 step_level = 2 parent_step_id =
535 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.2
=
step_text =

Friday, March 09, 2001
Page 21 of 130

step_label = Durability Stream 7 step_id =
575 global_flag = 0

sequence_no = 7 step_level = 2 parent_step_id =
535 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.2
=
step_text =

step_label = Durability Stream 8 step_id =
576 global_flag = 0

sequence_no = 8 step_level = 2 parent_step_id =
535 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.2
=
step_text =

step_label = Durability Stream 9 step_id =
577 global_flag = 0

sequence_no = 9 step_level = 2 parent_step_id =
535 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.2
=
step_text =

step_label = Durability Stream 10 step_id =
578 global_flag = 0

sequence_no = 10 step_level = 2 parent_step_id =
535 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.2
=
step_text =

step_label = Execute Isolation 1_1 step_id =
579 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
541 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol1_1.sql
version_no = 9.0

start_directory Static_Connection_to_DB_1
parent_version_no = 0.9
=
step_text =

step_label = Signal End of ISOL1_1 step_id =
580 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
541 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.9
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO1_1

step_label = Execute Isolation 1_2 step_id =
581 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
542 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol1_2.sql
version_no = 5.0

start_directory Static_Connection_to_DB_2
parent_version_no = 0.5
=
step_text =

step_label = Signal End of ISOL1_2 step_id =
582 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
542 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.5
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO1_2

step_label = Execute Isolation 2_1 step_id =
583 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
544 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol2_1.sql
version_no = 2.0

start_directory Static_Connection_to_DB_1
parent_version_no = 0.2
=
step_text =

Friday, March 09, 2001
Page 23 of 130

step_label = Signal End of ISOL2_1 step_id =
584 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
544 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.2
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO2_1

step_label = Execute Isolation 2_2 step_id =
585 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
545 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol2_2.sql
version_no = 1.0

start_directory Static_Connection_to_DB_2
parent_version_no = 0.1
=
step_text =

step_label = Signal End of ISOL2_2 step_id =
586 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
545 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.1
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO2_2

step_label = Execute Isolation 3_1 step_id =
587 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
547 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol3_1.sql
version_no = 1.0

start_directory Static_Connection_to_DB_1
parent_version_no = 0.1
=
step_text =

Friday, March 09, 2001
Page 24 of 130

step_label = Signal End of ISOL3_1 step_id =
588 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
547 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.1
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO3_1

step_label = Execute Isolation 3_2 step_id =
589 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
548 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol3_2.sql
version_no = 1.0

start_directory Static_Connection_to_DB_2
parent_version_no = 0.1
=
step_text =

step_label = Signal End of ISOL3_2 step_id =
590 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
548 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.1
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO3_2

step_label = Execute Isolation 4_1 step_id =
591 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
550 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol4_1.sql
version_no = 1.0

start_directory Static_Connection_to_DB_1
parent_version_no = 0.1
=
step_text =

step_label = Signal End of ISOL4_1 step_id =
592 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
550 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.1
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO4_1

step_label = Execute Isolation 4_2 step_id =
593 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
551 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol4_2.sql
version_no = 1.0

start_directory Static_Connection_to_DB_2
parent_version_no = 0.1
=
step_text =

step_label = Signal End of ISOL4_2 step_id =
594 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
551 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.1
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO4_2

step_label = Execute Isolation 5_1 step_id =
595 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
553 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol5_1.sql
version_no = 1.0

start_directory Static_Connection_to_DB_1
parent_version_no = 0.1
=
step_text =

Friday, March 09, 2001
Page 26 of 130

step_label = Signal End of ISOL5_1 step_id =
596 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
553 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.1
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO5_1

step_label = Execute Isolation 5_2 step_id =
597 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
554 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol5_2.sql
version_no = 1.0

start_directory Static_Connection_to_DB_2
parent_version_no = 0.1
=
step_text =

step_label = Signal End of ISOL5_2 step_id =
598 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
554 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.1
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO5_2

step_label = Execute Isolation 6_1 step_id =
599 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
556 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol6_1.sql
version_no = 1.0

start_directory Static_Connection_to_DB_1
parent_version_no = 0.1
=
step_text =

Friday, March 09, 2001
Page 27 of 130

step_label = Signal End of ISOL6_1 step_id =
600 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
556 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.1
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO6_1

step_label = Execute Isolation 6_2 step_id = 601 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 557 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol6_2.sql version_no = 1.0

start_directory = Static_Connection_to_DB_2 parent_version_no = 0.1 =

step_text =

step_label = Signal End of ISOL6_2 step_id = 602 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id = 557 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2 failure_details =

step_file_name = version_no = 0.0

start_directory = %TOOLS_DIR% parent_version_no = 0.1 =

step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO6_2

step_label = Execute Isolation 6_3 step_id = 603 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 558 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = %ACID_DIR%\iso\scripts\isol6_3.sql version_no = 1.0

start_directory = Static_Connection_to_DB_3 parent_version_no = 0.1 =

step_text =

step_label = Signal End of ISOL6_3 step_id = 604 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id = 558 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2 failure_details =

step_file_name = version_no = 0.0

start_directory = %TOOLS_DIR% parent_version_no = 0.1 =

step_text = %TOOLS_DIR%\Utility\semaphore -signal ISO6_3

step_label = Cons - ACID Transactions Stream 1 step_id = 605 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 559 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = version_no = 3.0

start_directory = Static_Connection_to_DB_1 parent_version_no = 0.8 =

step_text = EXEC cons_tran %COUNTER%

step_label = Cons - ACID Transactions Stream 2 step_id = 606 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 560 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = version_no = 1.0

start_directory Static_Connection_to_DB_2
parent_version_no = 1.1
=
step_text = EXEC cons_tran %COUNTER%

Friday, March 09, 2001
Page 29 of 130

step_label = Cons - ACID Transactions Stream 3 step_id
= 607 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
561 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 1.0

start_directory Static_Connection_to_DB_3
parent_version_no = 2.0
=
step_text = EXEC cons_tran %COUNTER%

step_label = Cons - ACID Transactions Stream 4 step_id
= 608 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
562 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 1.0

start_directory Static_Connection_to_DB_4
parent_version_no = 2.0
=
step_text = EXEC cons_tran %COUNTER%

step_label = Cons - ACID Transactions Stream 5 step_id
= 609 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
563 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 1.0

start_directory Static_Connection_to_DB_5
parent_version_no = 1.0
=
step_text = EXEC cons_tran %COUNTER%

Friday, March 09, 2001
Page 30 of 130

step_label = Cons - ACID Transactions Stream 6 step_id
= 610 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
564 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 1.0

start_directory Static_Connection_to_DB_6
parent_version_no = 1.0
=
step_text = EXEC cons_tran %COUNTER%

step_label = Cons - ACID Transactions Stream 7 step_id
= 611 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
565 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_7
parent_version_no = 0.0
=
step_text = EXEC cons_tran %COUNTER%

step_label = Cons - ACID Transactions Stream 8 step_id
= 612 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
566 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_8
parent_version_no = 0.0
=
step_text = EXEC cons_tran %COUNTER%

Friday, March 09, 2001
Page 31 of 130

step_label = Cons - ACID Transactions Stream 9 step_id
= 613 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
567 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_9
parent_version_no = 0.0
=
step_text = EXEC cons_tran %COUNTER%

step_label = Cons - ACID Transactions Stream 10 step_id
= 614 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
568 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_10
parent_version_no = 0.0
=
step_text = EXEC cons_tran %COUNTER%

step_label = Dur - ACID Transactions Stream 1 step_id
= 615 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
569 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_1
parent_version_no = 0.0
=
step_text = EXEC dur_tran %COUNTER%

Friday, March 09, 2001
Page 32 of 130

step_label = Dur - ACID Transactions Stream 2 step_id
= 616 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
570 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 1.0

start_directory Static_Connection_to_DB_2
parent_version_no = 0.1
=
step_text = EXEC dur_tran %COUNTER%

step_label = Dur - ACID Transactions Stream 3 step_id
= 617 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
571 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_3
parent_version_no = 0.0
=
step_text = EXEC dur_tran %COUNTER%

step_label = Dur - ACID Transactions Stream 4 step_id = 618 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 572 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_4
parent_version_no = 0.0
=
step_text = EXEC dur_tran %COUNTER%

step_label = Dur - ACID Transactions Stream 5 step_id = 619 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 573 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_5
parent_version_no = 0.0
=
step_text = EXEC dur_tran %COUNTER%

step_label = Dur - ACID Transactions Stream 6 step_id = 620 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 574 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_6
parent_version_no = 0.0
=
step_text = EXEC dur_tran %COUNTER%

step_label = Dur - ACID Transactions Stream 7 step_id = 621 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 575 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_7
parent_version_no = 0.0
=
step_text = EXEC dur_tran %COUNTER%

step_label = Dur - ACID Transactions Stream 8 step_id = 622 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 576 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Static_Connection_to_DB_8
parent_version_no = 0.0
=
step_text = EXEC dur_tran %COUNTER%

step_label = Dur - ACID Transactions Stream 9 step_id = 623 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 577 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

```
 step_file_name =
version_no =  0.0

 start_directory     Static_Connection_to_DB_9
  parent_version_no =     0.0
 =
 step_text =  EXEC dur_tran %COUNTER%




 step_label = Dur - ACID Transactions Stream 10                  step_id
=    624      global_flag =    0

 sequence_no =        1  step_level =       3        parent_step_id =
578 enabled_flag =      -1

 iterator_name =     COUNTER                        degree_parallelism  1

 execution_mechanism =     1                     continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     Static_Connection_to_DB_10
   parent_version_no =     0.0
 =
 step_text =  EXEC dur_tran %COUNTER%




 step_label = Execute Atomicity Commit Test                      step_id =
   814      global_flag =    0

 sequence_no =        1  step_level =       1        parent_step_id =
508 enabled_flag =      -1

 iterator_name =                                  degree_parallelism  1

 execution_mechanism =     1                     continuation_criteria = 2
       failure_details =

 step_file_name =    %ACID_DIR%\atom\scripts\atom_c.sql
                version_no =  4.0

 start_directory     Static_Connection_to_DB_1
  parent_version_no =     8.0
 =
 step_text =



 Friday, March 09, 2001
    Page 35 of 130




 step_label = Copy Output Files to Atomicity Directory           step_id
=    815      global_flag =    0

 sequence_no =        3  step_level =       1        parent_step_id =
508 enabled_flag =      -1

 iterator_name =                                  degree_parallelism  1

 execution_mechanism =     2                     continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  6.0

 start_directory     %ACID_DIR%
parent_version_no =     8.0
 =
 step_text =  ::
              :: Delete any old files first
              ::
              IF EXIST %ACID_DIR%\atom\atomc.ver DEL
%ACID_DIR%\atom\atomc.ver > nul
              IF EXIST %ACID_DIR%\atom\atomrb.ver DEL
%ACID_DIR%\atom\atomrb.ver > nul

              ::
              :: Now copy the output from the RUN_ID directory
              ::
              copy
%OUTPUT_DIR%\Execute_Atomicity_Commit_Test.out
%ACID_DIR%\atom\atomc.ver
              copy
%OUTPUT_DIR%\Execute_Atomicity_Rollback_Test.out
%ACID_DIR%\atom\atomrb.ver




 step_label = Cons - Concatenate Stream 1 Output Files
step_id =     818      global_flag =    0

 sequence_no =        1  step_level =       4        parent_step_id =
847 enabled_flag =      -1

 iterator_name =     COUNTER                        degree_parallelism  1

 execution_mechanism =     2                     continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  8.0

 start_directory     %ACID_DIR%
parent_version_no =     0.1
 =
 step_text =  IF EXIST CONS\Stream1.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_1.%COUNT
ER%.out >>
              CONS\Stream1.out



 step_label = Cons - Concatenate Stream 2 Output Files
step_id =     819      global_flag =    0

 sequence_no =        1  step_level =       4        parent_step_id =
848 enabled_flag =      -1

 iterator_name =     COUNTER                        degree_parallelism  1

 execution_mechanism =     2                     continuation_criteria = 2
       failure_details =
```

step_file_name =
version_no = 3.0

start_directory %ACID_DIR%
parent_version_no = 0.1
=
step_text = IF EXIST CONS\Stream2.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_2.%COUNT
ER%.out >>
CONS\Stream2.out

step_label = Cons - Concatenate Stream 3 Output Files
step_id = 820 global_flag = 0

sequence_no = 1 step_level = 4 parent_step_id =
849 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 3.0

start_directory %ACID_DIR%
parent_version_no = 0.1
=
step_text = IF EXIST CONS\Stream3.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_3.%COUNT
ER%.out >>
CONS\Stream3.out

step_label = Cons - Concatenate Stream 4 Output Files
step_id = 821 global_flag = 0

sequence_no = 1 step_level = 4 parent_step_id =
850 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 2.0

start_directory %ACID_DIR%
parent_version_no = 0.0
=
step_text = IF EXIST CONS\Stream4.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_4.%COUNT
ER%.out >>
CONS\Stream4.out

step_label = Cons - Concatenate Stream 5 Output Files
step_id = 822 global_flag = 0

sequence_no = 1 step_level = 4 parent_step_id =
851 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 1.0

start_directory %ACID_DIR%
parent_version_no = 1.0
=
step_text = IF EXIST CONS\Stream5.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_5.%COUNT
ER%.out >>
CONS\Stream5.out

step_label = Cons - Concatenate Stream 6 Output Files
step_id = 823 global_flag = 0

sequence_no = 1 step_level = 4 parent_step_id =
854 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 1.0

start_directory %ACID_DIR%
parent_version_no = 0.0
=
step_text = IF EXIST CONS\Stream6.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_6.%COUNT
ER%.out >>
CONS\Stream6.out

step_label = Cons - Concatenate Stream 7 Output Files
step_id = 824 global_flag = 0

```
 sequence_no =        1  step_level =       4        parent_step_id =
855 enabled_flag =       0

 iterator_name =     COUNTER                       degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %ACID_DIR%
parent_version_no =     0.0
 =
 step_text =  IF EXIST CONS\Stream7.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_7.%COUNT
ER%.out >>
           CONS\Stream7.out




 step_label = Cons - Concatenate Stream 8 Output Files
step_id =     825      global_flag =    0

 sequence_no =        1  step_level =       4        parent_step_id =
856 enabled_flag =       0

 iterator_name =     COUNTER                       degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %ACID_DIR%
parent_version_no =     0.0
 =
 step_text =  IF EXIST CONS\Stream8.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_8.%COUNT
ER%.out >>
           CONS\Stream8.out



 Friday, March 09, 2001
    Page 38 of 130



 step_label = Cons - Concatenate Stream 9 Output Files
step_id =     826      global_flag =    0

 sequence_no =        1  step_level =       4        parent_step_id =
857 enabled_flag =       0

 iterator_name =     COUNTER                       degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %ACID_DIR%
parent_version_no =     0.0
 =
 step_text =  IF EXIST CONS\Stream9.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_9.%COUNT
ER%.out >>
           CONS\Stream9.out




 step_label = Cons - Concatenate Stream 10 Output Files
step_id =     827      global_flag =    0

 sequence_no =        1  step_level =       4        parent_step_id =
858 enabled_flag =       0

 iterator_name =     COUNTER                       degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %ACID_DIR%
parent_version_no =     0.0
 =
 step_text =  IF EXIST CONS\Stream10.out TYPE

%OUTPUT_DIR%\Cons_-_ACID_Transactions_Stream_10.%COUN
TER%.out >>
           CONS\Stream10.out




 step_label = Prepare .VER Consistency Output Files                step_id
=     828      global_flag =    0

 sequence_no =        1  step_level =       2        parent_step_id =
834 enabled_flag =      -1

 iterator_name =                                  degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  1.0

 start_directory     %ACID_DIR%
parent_version_no =     3.0
 =
 step_text =  ::
           :: Copy and rename the output files from the 2 Consistency
tests
           ::;

           COPY %OUTPUT_DIR%\Execute_Consistency_Test_1.out
CONS\CONS1.Ver
           COPY %OUTPUT_DIR%\Execute_Consistency_Test_2.out
CONS\CONS2.Ver
```

step_label = Post Process Durability Output step_id = 832 global_flag = 0

sequence_no = 2 step_level = 1 parent_step_id = 512 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.3

start_directory
parent_version_no = 4.0
=
step_text =

step_label = Dur - Concatenate Stream 1 Output Files step_id = 833 global_flag = 0

sequence_no = 1 step_level = 4 parent_step_id = 859 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 7.0

start_directory %ACID_DIR%
parent_version_no = 1.0
=
step_text = IF EXIST DUR\Stream1.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_Stream_1.%COUNTER%.out >> DUR\Stream1.out

step_label = Post-Process Consistency Output step_id = 834 global_flag = 0

sequence_no = 7 step_level = 1 parent_step_id = 510 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 3.0

start_directory
parent_version_no = 4.0
=
step_text =

step_label = Dur - Concatenate Stream 2 Output Files step_id = 835 global_flag = 0

sequence_no = 1 step_level = 4 parent_step_id = 860 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 7.0

start_directory %ACID_DIR%
parent_version_no = 1.0
=
step_text = IF EXIST DUR\Stream2.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_Stream_2.%COUNTER%.out >> DUR\Stream2.out

step_label = Dur - Concatenate Stream 3 Output Files step_id = 836 global_flag = 0

sequence_no = 1 step_level = 4 parent_step_id = 861 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 10.0

start_directory %ACID_DIR%
parent_version_no = 0.4
=
step_text = IF EXIST DUR\Stream3.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_Stream_3.%COUNTER%.out >> DUR\Stream3.out

step_label = Dur - Concatenate Stream 4 Output Files step_id = 837 global_flag = 0

```
 sequence_no =        1  step_level =      4       parent_step_id =
862 enabled_flag =     -1

 iterator_name =     COUNTER                    degree_parallelism  1

 execution_mechanism =     2                   continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory     %ACID_DIR%
parent_version_no =     1.0
 =
 step_text =  IF EXIST DUR\Stream4.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_
Stream_4.%COUNTER%.out >> DUR\Stream4.out


 Friday, March 09, 2001
    Page 41 of 130




 step_label = Dur - Concatenate Stream 5 Output Files             step_id
=     838      global_flag =    0

 sequence_no =        1  step_level =      4       parent_step_id =
863 enabled_flag =     -1

 iterator_name =     COUNTER                    degree_parallelism  1

 execution_mechanism =     2                   continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory     %ACID_DIR%
parent_version_no =     1.0
 =
 step_text =  IF EXIST DUR\Stream5.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_
Stream_5.%COUNTER%.out >> DUR\Stream5.out




 step_label = Dur - Concatenate Stream 6 Output Files             step_id
=     839      global_flag =    0

 sequence_no =        1  step_level =      4       parent_step_id =
864 enabled_flag =     -1

 iterator_name =     COUNTER                    degree_parallelism  1

 execution_mechanism =     2                   continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory     %ACID_DIR%
parent_version_no =     1.0
 =
 step_text =  IF EXIST DUR\Stream6.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_
Stream_6.%COUNTER%.out >> DUR\Stream6.out




 step_label = Dur - Concatenate Stream 7 Output Files             step_id
=     840      global_flag =    0

 sequence_no =        1  step_level =      4       parent_step_id =
865 enabled_flag =     -1

 iterator_name =     COUNTER                    degree_parallelism  1

 execution_mechanism =     2                   continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory     %ACID_DIR%
parent_version_no =     0.1
 =
 step_text =  IF EXIST DUR\Stream7.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_
Stream_7.%COUNTER%.out >> DUR\Stream7.out


 Friday, March 09, 2001
    Page 42 of 130




 step_label = Dur - Concatenate Stream 8 Output Files             step_id
=     841      global_flag =    0

 sequence_no =        1  step_level =      4       parent_step_id =
866 enabled_flag =     -1

 iterator_name =     COUNTER                    degree_parallelism  1

 execution_mechanism =     2                   continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory     %ACID_DIR%
parent_version_no =     0.1
 =
 step_text =  IF EXIST DUR\Stream8.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_
Stream_8.%COUNTER%.out >> DUR\Stream8.out




 step_label = Dur - Concatenate Stream 9 Output Files             step_id
=     842      global_flag =    0
```

```
 sequence_no =        1  step_level =       4        parent_step_id =
867 enabled_flag =      -1

 iterator_name =     COUNTER                      degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory     %ACID_DIR%
parent_version_no =     0.1
 =
 step_text =  IF EXIST DUR\Stream9.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_
Stream_9.%COUNTER%.out >> DUR\Stream9.out




 step_label = Dur - Concatenate Stream 10 Output Files
step_id =     843      global_flag =    0

 sequence_no =        1  step_level =       4        parent_step_id =
868 enabled_flag =      -1

 iterator_name =     COUNTER                      degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory     %ACID_DIR%
parent_version_no =     0.1
 =
 step_text =  IF EXIST DUR\Stream10.out TYPE

%DEFAULT_DIR%\%DUR_RUN_ID%\Dur_-_ACID_Transactions_
Stream_10.%COUNTER%.out >> DUR\Stream10.out



 Friday, March 09, 2001
    Page 43 of 130




 step_label = Prepare .VER Durability Output Files                step_id
=     844      global_flag =    0

 sequence_no =        1  step_level =       2        parent_step_id =
832 enabled_flag =      -1

 iterator_name =                                  degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  3.0
```

```
 start_directory     %ACID_DIR%
parent_version_no =     0.3
 =
 step_text =  ::
            :: Copy and rename the output files from the 2 Consistency
tests
            ::;

            COPY
%DEFAULT_DIR%\%DUR_RUN_ID%\Execute_Durability_Verific
ation_Test_1.out DUR\DUR1.Ver
            COPY
%OUTPUT_DIR%\Execute_Durability_Verification_Test_2.out
DUR\DUR2.Ver




 step_label = Dur - Concatenate Stream Output Files               step_id
=     845      global_flag =    0

 sequence_no =        2  step_level =       2        parent_step_id =
832 enabled_flag =      -1

 iterator_name =                                  degree_parallelism
%NUM_STREAMS%

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.3

 start_directory
parent_version_no =     0.3
 =
 step_text =



 step_label = Cons - Concatenate Stream Output Files              step_id
=     846      global_flag =    0

 sequence_no =        2  step_level =       2        parent_step_id =
834 enabled_flag =      -1

 iterator_name =                                  degree_parallelism
%NUM_STREAMS%

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.3

 start_directory
parent_version_no =     3.0
 =
 step_text =



 Friday, March 09, 2001
    Page 44 of 130
```

step_label = Cons - Stream 1 Output step_id = 847 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 846 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Cons - Stream 2 Output step_id = 848 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id = 846 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Cons - Stream 3 Output step_id = 849 global_flag = 0

sequence_no = 3 step_level = 3 parent_step_id = 846 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Cons - Stream 4 Output step_id = 850 global_flag = 0

sequence_no = 4 step_level = 3 parent_step_id = 846 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.3
=
step_text =

Friday, March 09, 2001
Page 45 of 130

step_label = Cons - Stream 5 Output step_id = 851 global_flag = 0

sequence_no = 5 step_level = 3 parent_step_id = 846 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Clean-Up Consistency Directory step_id = 852 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id = 510 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 1.0

start_directory %ACID_DIR%
parent_version_no = 4.0
=
step_text = ::
:: Delete any existing output files for the consistency streams
::
IF EXIST CONS\Stream%COUNTER%.OUT DEL CONS\Stream%COUNTER%.out > nul

step_label = Clean-Up Durability Directory step_id = 853 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id = 511 enabled_flag = -1

iterator_name = COUNTER degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %ACID_DIR%
parent_version_no = 6.0
=
step_text = ::
:: Delete any existing output files for the consistency streams
::
IF EXIST DUR\Stream%COUNTER%.OUT DEL DURS\Stream%COUNTER%.out > nul

step_label = Cons - Stream 6 Output step_id = 854 global_flag = 0

sequence_no = 6 step_level = 3 parent_step_id = 846 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Cons - Stream 7 Output step_id = 855 global_flag = 0

sequence_no = 7 step_level = 3 parent_step_id = 846 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Cons - Stream 8 Output step_id = 856 global_flag = 0

sequence_no = 8 step_level = 3 parent_step_id = 846 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Cons - Stream 9 Output step_id = 857 global_flag = 0

sequence_no = 9 step_level = 3 parent_step_id = 846 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Cons - Stream 10 Output step_id = 858 global_flag = 0

sequence_no = 10 step_level = 3 parent_step_id = 846 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Dur - Stream 1 Output step_id =
859 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
845 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Dur - Stream 2 Output step_id =
860 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
845 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Dur - Stream 3 Output step_id =
861 global_flag = 0

sequence_no = 3 step_level = 3 parent_step_id =
845 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.4

start_directory
parent_version_no = 0.3
=
step_text =

Friday, March 09, 2001
Page 48 of 130

step_label = Dur - Stream 4 Output step_id =
862 global_flag = 0

sequence_no = 4 step_level = 3 parent_step_id =
845 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Dur - Stream 5 Output step_id =
863 global_flag = 0

sequence_no = 5 step_level = 3 parent_step_id =
845 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Dur - Stream 6 Output step_id =
864 global_flag = 0

sequence_no = 6 step_level = 3 parent_step_id =
845 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Dur - Stream 7 Output step_id =
865 global_flag = 0

sequence_no = 7 step_level = 3 parent_step_id =
845 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Dur - Stream 8 Output step_id =
866 global_flag = 0

sequence_no = 8 step_level = 3 parent_step_id =
845 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Dur - Stream 9 Output step_id =
867 global_flag = 0

sequence_no = 9 step_level = 3 parent_step_id =
845 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 0.3
=
step_text =

step_label = Dur - Stream 10 Output step_id =
868 global_flag = 0

sequence_no = 10 step_level = 3 parent_step_id =
845 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 0.3
=
step_text =

Iterator_values

workspace_id	step_id	version_no	type	iterator_value	sequence_no
7	613	0.0	3	1	0

7	616	1.0	2	410	0
7	616	1.0	1	211	0
7	615	0.0	2	210	0
7	615	0.0	1	11	0
7	615	0.0	3	1	0
7	614	0.0	2	550	0
7	516	3.0	4	cons_tran.sql	7

7 614 0.0 1 451
0

7 617 0.0 3 1
0

7 613 0.0 2 500
0

7 613 0.0 1 401
0

7 612 0.0 1 351
0

7 612 0.0 3 1
0

7 612 0.0 2 450
0

7 611 0.0 1 301
0

7 614 0.0 3 1
0

7 619 0.0 3 1
0

7 622 0.0 2 1610
0

7 622 0.0 1 1411
0

7 621 0.0 1 1211
0

7 621 0.0 2 1410
0

7 621 0.0 3 1
0

7 620 0.0 2 1210
0

7 616 1.0 3 1
0

7 620 0.0 3 1
0

7 617 0.0 1 411
0

7 619 0.0 2 1010
0

7 619 0.0 1 811
0

7 618 0.0 2 810
0

7 618 0.0 3 1
0

7 618 0.0 1 611
0

7 617 0.0 2 610
0

7 610 1.0 1 251
0

7 620 0.0 1 1011
0

7 516 3.0 4 acid_tran.sql
1

7 611 0.0 2 400
0

7 532 0.0 2 %NUM_STREAMS%
0

7 532 0.0 1 1
0

7 527 7.0 3 1
0

7 527 7.0 1 1
0

7 527 7.0 2 %NUM_STREAMS%
0

7 605 3.0 1 11
0

7 516 3.0 4 acid_values.sql
0

7 605 3.0 2 110
0

7 516 3.0 4 iso34_tran_w_com.sql
10

7 516 3.0 4 iso6_tran_w_com.sql
9

7 516 3.0 4 dur_tran.sql
8

7 516 3.0 4 acid_dur_tran.sql
6

7 516 3.0 4 iso_query.sql
4

7 516 3.0 4 tran_w_rb.sql
3

7 516 3.0 4 tran_w_com.sql
2

7 516 3.0 4 iso5_parts.sql
5

7 608 1.0 3 1
0

7 623 0.0 3 1
0

7 610 1.0 2 350
0

7 610 1.0 3 1
0

7 609 1.0 1 201
0

7 609 1.0 3 1
0

7 609 1.0 2 300
0

7 532 0.0 3 1
0

7 608 1.0 2 250
0

7 611 0.0 3 1
0

7 607 1.0 2 200
0

7 607 1.0 3 1
0

7 607 1.0 1 101
0

7 606 1.0 3 1
0

7 606 1.0 1 51
0

7 606 1.0 2 150
0

7 605 3.0 3 1
0

7 608 1.0 1 151
0

7 836 10.0 3 1
0

7 833 7.0 1 11
0

7 839 7.0 1 1011
0

7 838 7.0 3 1
0

7 838 7.0 2 1010
0

7 838 7.0 1 811
0

7 837 7.0 3 1
0

7 839 7.0 3 1
0

7 837 7.0 1 611
0

7 840 7.0 2 1410
0

7 836 10.0 2 610
0

7 836 10.0 1 411
0

7 835 7.0 1 211
0

7 835 7.0 3 1
0

7 835 7.0 2 410
0

7 833 7.0 2 210
0

7 622 0.0 3 1
0

7 837 7.0 2 810
0

7 842 7.0 3 1
0

7 853 0.0 1 1
0

7 853 0.0 2 10
0

7 852 1.0 1 1
0

7 852 1.0 2 10
0

7 852 1.0 3 1
0

7 843 7.0 3 1
0

7 839 7.0 2 1210
0

7 843 7.0 1 1811
0

7 827 0.0 3 1
0

7 842 7.0 1 1611
0

7 842 7.0 2 1810
0

7 841 7.0 1 1411
0

7 841 7.0 2 1610
0

7 841 7.0 3 1
0

7 840 7.0 1 1211
0

7 840 7.0 3 1
0

7 843 7.0 2 2010
0

7 818 8.0 1 11
0

7 833 7.0 3 1
0

7 821 2.0 3 1
0

7 820 3.0 1 101
0

7 820 3.0 3 1
0

7 820 3.0 2 200
0

7 819 3.0 1 51
0

7 821 2.0 2 250
0

7 819 3.0 2 150
0

7 822 1.0 1 201
0

7 818 8.0 2 110
0

7 818 8.0 3 1
0

7 624 0.0 2 2010
0

7 624 0.0 3 1
0

7 624 0.0 1 1811
0

7 623 0.0 2 1810
0

7 853 0.0 3 1
0

7 819 3.0 3 1
0

7 824 0.0 3 1
0

7 827 0.0 2 550
0

```
      7     827   0.0            1 451
 0

      7     826   0.0            1 401
 0

      7     826   0.0            2 500
 0

      7     826   0.0            3 1
 0

      7     825   0.0            3 1
 0

      7     821   2.0            1 151
 0

      7     825   0.0            1 351
 0

      7     623   0.0            1 1611
 0

      7     824   0.0            1 301
 0

      7     824   0.0            2 400
 0

      7     823   1.0            1 251
 0

      7     823   1.0            3 1
 0

      7     823   1.0            2 350
 0

      7     822   1.0            3 1
 0

      7     822   1.0            2 300
 0

      7     825   0.0            2 450
 0
```


Workspace_parameters

```
workspace_id =    8   parameter_id          92  parameter_type
  3
parameter_name        DEFAULT_DIR
parameter_valu        %KIT_DIR%\OUTPUT

workspace_id =    8   parameter_id          79  parameter_type
  0
parameter_name        DBNAME
parameter_valu        tpch100g

workspace_id =    8   parameter_id          81  parameter_type
  0
parameter_name        SETUP_DIR
parameter_valu        %KIT_DIR%\Setup

workspace_id =    8   parameter_id          82  parameter_type
  3
parameter_name        OUTPUT_DIR
parameter_valu        D:\MSTPCH~1.101\OUTPUT\556

workspace_id =    8   parameter_id          83  parameter_type
  0
parameter_name        SCALEFACTOR
parameter_valu        100

workspace_id =    8   parameter_id          84  parameter_type
  0
parameter_name        UPDATE_SETS
parameter_valu        24

workspace_id =    8   parameter_id          85  parameter_type
  0
parameter_name        DELETE_SEGMENTS_PER_UPDATE_SET
parameter_valu        16

workspace_id =    8   parameter_id          86  parameter_type
  0
parameter_name        INSERT_SEGMENTS_PER_UPDATE_SET
parameter_valu        16

workspace_id =    8   parameter_id          87  parameter_type
  0
parameter_name        VALIDATION_DIR
parameter_valu        %SETUP_DIR%\Validation
```

workspace_id = 8 parameter_id 88 parameter_type 0

parameter_name INDEX_CREATE_PARALLELISM
parameter_valu 16

workspace_id = 8 parameter_id 78 parameter_type 0

parameter_name TRACEFLAGS

parameter_valu -c -x -t3502 -g100

workspace_id = 8 parameter_id 91 parameter_type 0
parameter_name RF_FLATFILE_DIR

parameter_valu L:\RF_Flat_Files

workspace_id = 8 parameter_id 135 parameter_type 0

parameter_name FLATFILE_DIR_1
parameter_valu U:\FLAT_FILES_1

workspace_id = 8 parameter_id 93 parameter_type 0

parameter_name TOOLS_DIR

parameter_valu %KIT_DIR%\tools

workspace_id = 8 parameter_id 94 parameter_type 0
parameter_name MAX_STREAMS

parameter_valu 5

workspace_id = 8 parameter_id 95 parameter_type 0

parameter_name TEMPLATE_DIR
parameter_valu %RUN_DIR%\templates

workspace_id = 8 parameter_id 96 parameter_type 0

parameter_name QUERY_DIR

parameter_valu %RUN_DIR%\Queries

workspace_id = 8 parameter_id 97 parameter_type 0
parameter_name RUN_DIR

parameter_valu %KIT_DIR%\run

workspace_id = 8 parameter_id 98 parameter_type 0

parameter_name KIT_DIR
parameter_valu D:\mstpch.101d

workspace_id = 8 parameter_id 115 parameter_type 3

parameter_name RUN_ID
parameter_valu 556

workspace_id = 8 parameter_id 138 parameter_type 0

parameter_name FLATFILE_DIR_4

parameter_valu U:\FLAT_FILES_4

workspace_id = 8 parameter_id 137 parameter_type 0
parameter_name FLATFILE_DIR_3

parameter_valu U:\FLAT_FILES_3

workspace_id = 8 parameter_id 136 parameter_type 0

parameter_name FLATFILE_DIR_2
parameter_valu U:\FLAT_FILES_2

workspace_id = 8 parameter_id 89 parameter_type 0

parameter_name TABLE_LOAD_PARALLELISM

parameter_valu 16

Connection_dtls

workspac connection_na connection_name
connection_string_name connection
e_id me_id
_type

8 39 Static_Connection_to_Master
MASTERDBCONNECTION 1

8 23 Static_Connection_to_DB
DBCONNECTION 1

8 22 Dynamic_Connection_to_Master
MASTERDBCONNECTION 2

8 21 Dynamic_Connection_to_DB
DBCONNECTION 2

Friday, March 09, 2001
Page 58 of 130

Workspace_connections

workspace_i 8

connection_i 10
connection_name MASTERDBCONNECTION

connection_valu DRIVER=SQL
Server;SERVER=;UID=sa;PWD=;

descripti
no_count_displa 0

no_execut 0

parse_query_onl 0
ANSI_quoted_identifie 0

ANSI_nulls -1

show_query_plan 0
show_stats_tim 0

show_stats_i 0

parse_odbc_msg_prefix -1
row_count 0

tsql_batch_separa GO

query_time_ou 0
server_languag (Default)

character_translat -1

regional_setti 0

workspace_i 8

connection_i 9
connection_name DBCONNECTION

connection_valu DRIVER=SQL
Server;SERVER=;UID=sa;PWD=;DATABASE=%DBNAME%;

descripti
no_count_displa 0

no_execut 0

parse_query_onl 0
ANSI_quoted_identifie 0

ANSI_nulls -1

show_query_plan 0
show_stats_tim 0

show_stats_i 0

parse_odbc_msg_prefix -1
row_count 0

tsql_batch_separa GO

query_time_ou 0
server_languag (Default)

character_translat -1
Friday, March 09, 2001
Page 59 of 130

ion
d
ngs

e
on

y 8

e
y
step_label = SqlServer Startup step_id = 625
global_flag = -1
rs
sequence_no = 1 step_level = 0 parent_step_id =
0 enabled_flag = 0

degree_parallelism 0

e nism = continuation_criteria = 0
failure_details =

```
 o          e =
version_no =  5.0
 es
 start_directory
parent_version_no =     0.0

 step_text =  start sqlservr  %TRACEFLAGS%
 tor              S_DIR%
 t

 e

 step_label = SqlServer Shutdown                          step_id =
626      global_flag =   -1

 sequence_no =        2  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                              degree_parallelism  0

 execution_mechanism =     2                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  3.0

 start_directory
parent_version_no =     0.0
 =
 step_text =  isql -Usa -P -t10 -Q"shutdown"
           %TOOLS_DIR%\Utility\sleep.exe30




 step_label = Wait For SQL Server                         step_id =
627      global_flag =   -1

 sequence_no =        3  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                              degree_parallelism  0

 execution_mechanism =     2                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory     %TOOLS_DIR%
parent_version_no =     0.0
 =
 step_text =  %TOOLS_DIR%\Utility\sleep.exe30



 Friday, March 09, 2001
    Page 60 of 130




 step_label = Generate FlatFiles                          step_id =     629
   global_flag =    0

 sequence_no =        1  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  23.0

 start_directory
parent_version_no =     0.0
 =
 step_text =



 step_label = Start SqlServer                             step_id =     630
  global_flag =    0

 sequence_no =        2  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  41.0

 start_directory
parent_version_no =     0.0
 =
 step_text =



 step_label = Create database                             step_id =     631
   global_flag =    0

 sequence_no =        3  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  58.0

 start_directory
parent_version_no =     0.0
 =
 step_text =



 step_label = Configure SqlServer                         step_id =
632      global_flag =    0

 sequence_no =        4  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =
```

step_file_name =
version_no = 55.0

start_directory
parent_version_no = 0.0
=
step_text =

Friday, March 09, 2001
Page 61 of 130

step_label = Create and Load Tables step_id =
633 global_flag = 0

sequence_no = 5 step_level = 0 parent_step_id =
0 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 55.0

start_directory
parent_version_no = 0.0
=
step_text =

step_label = Create Indexes step_id = 634
global_flag = 0

sequence_no = 6 step_level = 0 parent_step_id =
0 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 45.0

start_directory
parent_version_no = 0.0
=
step_text =

step_label = Set Stats, Prep Database, End of Load step_id
= 635 global_flag = 0

sequence_no = 7 step_level = 0 parent_step_id =
0 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 30.23

start_directory
parent_version_no = 0.0
=
step_text =

step_label = Backup TPC-H Database (If not used for ACID)
step_id = 636 global_flag = 0

sequence_no = 8 step_level = 0 parent_step_id =
0 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 10.0

start_directory
parent_version_no = 0.0
=
step_text =

Friday, March 09, 2001
Page 62 of 130

step_label = Capture Database and Table Space Used (Do Not
step_id = 637 global_flag = 0
Use on Performance Load)
sequence_no = 9 step_level = 0 parent_step_id =
0 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 11.0

start_directory
parent_version_no = 0.0
=
step_text =

step_label = Generate Queries via QGen step_id =
638 global_flag = 0

sequence_no = 10 step_level = 0 parent_step_id =
0 enabled_flag = -1

iterator_name = degree_parallelism 1

```
 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  18.0

 start_directory
parent_version_no =     0.0
 =
 step_text =



 step_label = Validation (for SCALEFACTOR=1 Only)
step_id =     639      global_flag =    0

 sequence_no =       11  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  12.0

 start_directory
parent_version_no =     0.0
 =
 step_text =



 step_label = Generate FlatFiles in Parallel                    step_id =
642      global_flag =    0

 sequence_no =        1  step_level =       1       parent_step_id =
629 enabled_flag =      -1

 iterator_name =                              degree_parallelism  16

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  10.0

 start_directory
parent_version_no =     23.0
 =
 step_text =



 Friday, March 09, 2001
    Page 63 of 130


step_label = Generate Update Files                             step_id =
643      global_flag =    0

 sequence_no =        2  step_level =       1       parent_step_id =
629 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1
```

```
 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  4.0

 start_directory     %RF_FLATFILE_DIR%
   parent_version_no =     23.0
 =
 step_text =  %TOOLS_DIR%\DBGEN\dbgen -q -b
%TOOLS_DIR%\dists.dss -U %UPDATE_SETS% -s
%SCALEFACTOR% -f -C
            %UPDATE_SETS% -i
%INSERT_SEGMENTS_PER_UPDATE_SET% -d
%DELETE_SEGMENTS_PER_UPDATE_SET%




 step_label = Start SqlServer                                   step_id =     644
  global_flag =    0

 sequence_no =        1  step_level =       1       parent_step_id =
630 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  3.0

 start_directory
parent_version_no =     41.0
 =
 step_text =  start sqlservr %TRACEFLAGS%




 step_label = Create TPC-H Database                             step_id =
645      global_flag =    0

 sequence_no =        1  step_level =       1       parent_step_id =
631 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
D:\MSTPCH.101D\setup\tpch100g\CreateDatabase.sql
version_no =  15.0

 start_directory     Static_Connection_to_Master
 parent_version_no =     58.0
 =
 step_text =



 Friday, March 09, 2001
    Page 64 of 130
```

```
 step_label = Set Database Options                          step_id =
646      global_flag =    0

 sequence_no =        2  step_level =       1        parent_step_id =
631 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  2.0

 start_directory     Static_Connection_to_Master
 parent_version_no =     58.0
 =
 step_text =  sp_dboption %DBNAME%, 'select into/bulkcopy',true
              GO
              sp_dboption %DBNAME%, 'torn page detection',false
              GO
              sp_dboption %DBNAME%, 'trunc. log on chkpt.',true
              GO




 step_label = Load Start Timestamp                          step_id =
648      global_flag =    0

 sequence_no =        1  step_level =       1        parent_step_id =
632 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory     Dynamic_Connection_to_Master
    parent_version_no =     55.0
 =
 step_text =  -- Create temporary table for timing in the Master
Database
              --
              --            This is just temporary until we build the
TPCH_AUX_TABLE later
              --

              --
              --  Delete any existing tpch_temp_timer table
              --
              if exists ( select name from sysobjects where name =
'tpch_temp_timer' )
               drop table tpch_temp_timer

              --
              --  Create the temporary table
              --
              create table tpch_temp_timer
              (
               load_start_time    datetime
              )

              --
              --  Store the starting time in the temporary table
              --
              insert into tpch_temp_timer values (getdate())
```



```
 step_label = Set sp_configure Options                      step_id =
651      global_flag =    0

 sequence_no =       11  step_level =       1        parent_step_id =
632 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =    %SETUP_DIR%\Utility\conf_tpch.sql
                version_no =  22.0

 start_directory     Dynamic_Connection_to_Master
    parent_version_no =     55.0
 =
 step_text =



 step_label = (Drop/)Create/Pin Tables                      step_id =
652      global_flag =    0

 sequence_no =        1  step_level =       1        parent_step_id =
633 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  1.11

 start_directory
parent_version_no =     55.0
 =
 step_text =



 step_label = Parallel Partitioned Table Load               step_id =
653      global_flag =    0

 sequence_no =        2  step_level =       1        parent_step_id =
633 enabled_flag =      -1

 iterator_name =     TABLE                      degree_parallelism
%TABLE_LOAD_PARALLELISM%
```

```
 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  7.0

 start_directory
parent_version_no =     55.0
 =
 step_text =



 step_label = Parallel Load Simple Tables                    step_id =
654      global_flag =    0

 sequence_no =        3  step_level =       1        parent_step_id =
633 enabled_flag =      -1

 iterator_name =                              degree_parallelism
%TABLE_LOAD_PARALLELISM%

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  5.0

 start_directory
parent_version_no =     55.0
 =
 step_text =


 Friday, March 09, 2001
    Page 66 of 130



 step_label = Drop Existing Indexes                          step_id =
655      global_flag =    0

 sequence_no =        1  step_level =       1        parent_step_id =
634 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  1.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     45.0
 =
 step_text =  declare @index      sysname
           declare @table      sysname

           -- Drop the non-clustered indexes first

           declare nc_index cursor for
           select sysindexes.name,sysobjects.name
             from sysindexes,sysobjects
             where
               sysobjects.id=sysindexes.id and
               sysobjects.type='U'     and
               sysindexes.indid>1      and
               sysindexes.status<>96
             order by sysindexes.name

           open nc_index
           fetch nc_index into @index,@table
           while @@fetch_status = 0
             begin
               print 'dropping NC index ' +  @table + '.' + @index
               exec('drop index '+@table+'.'+@index)
               fetch nc_index into @index,@table
             end

           -- Drop the Clustered Indexes last

           declare cl_index cursor for
           select sysindexes.name,sysobjects.name
             from sysindexes,sysobjects
             where
               sysobjects.id=sysindexes.id and
               sysobjects.type='U'     and
               sysindexes.indid=1      and
               sysindexes.status<>96
             order by sysindexes.name

           open cl_index
           fetch cl_index into @index,@table
           while @@fetch_status = 0
             begin
               print 'dropping Clustered index ' +  @table + '.' + @index
               exec('drop index '+@table+'.'+@index)
               fetch cl_index into @index,@table
             end


 Friday, March 09, 2001
    Page 67 of 130



 step_label = Create Clustered Indexes                       step_id =
656      global_flag =    0

 sequence_no =        2  step_level =       1        parent_step_id =
634 enabled_flag =      -1

 iterator_name =                              degree_parallelism  4

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
%SETUP_DIR%\%DBNAME%\CreateClusteredIndexes.sql
       version_no =  1.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     45.0
 =
 step_text =



 step_label = Create NC Indexes in Parallel                  step_id =
657      global_flag =    0
```

```
 sequence_no =        3  step_level =       1        parent_step_id =
634 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  1.0

 start_directory
parent_version_no =     45.0
 =
 step_text =



 step_label = Create Statistics and Disable AutoUpdate              step_id
=     658      global_flag =    0

 sequence_no =        1  step_level =       1        parent_step_id =
635 enabled_flag =       0

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  32.0

 start_directory
parent_version_no =     30.23
 =
 step_text =



 step_label = Reset DB_Option "Trunc"                              step_id =
660      global_flag =    0

 sequence_no =        3  step_level =       1        parent_step_id =
635 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  14.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     30.23
 =
 step_text =  sp_dboption %DBNAME%, 'trunc. ',false



 Friday, March 09, 2001
    Page 68 of 130



 step_label = Backup TPC-H Database (Only if using backup to
step_id =     661      global_flag =    0
           satisfy ACID)
 sequence_no =        5  step_level =       1        parent_step_id =
635 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  6.5

 start_directory
parent_version_no =     30.23
 =
 step_text =



 step_label = Install Refresh Function Stored Procedures           step_id
=     662      global_flag =    0

 sequence_no =        4  step_level =       1        parent_step_id =
635 enabled_flag =      -1

 iterator_name =                                degree_parallelism
%MAX_STREAMS%

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  9.0

 start_directory
parent_version_no =     30.23
 =
 step_text =



 step_label = End of Load                                  step_id =     663
  global_flag =    0

 sequence_no =        6  step_level =       1        parent_step_id =
635 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  10.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     30.23
 =
 step_text =  UPDATE TPCH_AUX_TABLE SET LoadFinish =
getdate()
           GO

           SELECT LoadStart    AS 'Load Start TimeStamp',
            LoadFinish  AS 'Load Finish TimeStamp',
            QgenSeed  AS 'Generated QGen Seed'
           FROM    TPCH_AUX_TABLE
           GO
```

step_label = Verify TPC-H Load step_id = 664 global_flag = 0

sequence_no = 7 step_level = 1 parent_step_id = 635 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = %SETUP_DIR%\Utility\VerifyTpchLoad.sql version_no = 7.0

start_directory Dynamic_Connection_to_DB parent_version_no = 30.23 =

step_text =

step_label = Generate QGEN Seed step_id = 665 global_flag = 0

sequence_no = 8 step_level = 1 parent_step_id = 635 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = version_no = 8.0

start_directory Dynamic_Connection_to_DB parent_version_no = 30.23 =

```
step_text =  DECLARE @Finish datetime
        DECLARE @seed0 integer

        --
        -- Get ending time of database load
        --
        SELECT @Finish=LoadFinish FROM TPCH_AUX_TABLE

        --
        -- Calculate seed per clause 2.1.3.3
        --
        SET @seed0 =

CONVERT(integer,100000000*DATEPART(MM,@Finish)+100000
0*DATEPART(DD,@Finish)+10000*DATEPART(HH,@F

inish)+100*DATEPART(MI,@Finish)+DATEPART(SS,@Finish))

        --
        -- Update the benchmark auxillary table
        --
        UPDATE TPCH_AUX_TABLE SET QgenSeed=@seed0

        SELECT * from TPCH_AUX_TABLE
```


step_label = Create Backup Device(s) step_id = 666 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id = 636 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = %SETUP_DIR%\%DBNAME%\CreateBackupDevices.sql version_no = 2.0

start_directory Dynamic_Connection_to_Master parent_version_no = 10.0 =

step_text =

step_label = Execute the backup step_id = 667 global_flag = 0

sequence_no = 2 step_level = 1 parent_step_id = 636 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = %SETUP_DIR%\%DBNAME%\BackupDatabase.sql version_no = 2.0

start_directory Dynamic_Connection_to_Master parent_version_no = 10.0 =

step_text =

step_label = Execute DBCC UPDATEUSAGE step_id = 668 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id = 637 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

```
 step_file_name =
version_no =  1.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     11.0
 =
 step_text =  --
            -- Correct any potential inaccuracies in the system tables
before running sp_spaceused
            --

            dbcc updateusage (%DBNAME%)
            GO

            dbcc updateusage (tempdb)
            GO
```



```
 step_label = Execute SpaceUsed Procedure                    step_id =
  669      global_flag =    0

 sequence_no =         2  step_level =       1        parent_step_id =
637 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  2.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     11.0
 =
 step_text =  sp_spaceused
            GO
            sp_spaceused 'REGION'
            GO
            sp_spaceused 'NATION'
            GO
            sp_spaceused 'PART'
            GO
            sp_spaceused 'SUPPLIER'
            GO
            sp_spaceused 'PARTSUPP'
            GO
            sp_spaceused 'CUSTOMER'
            GO
            sp_spaceused 'ORDERS'
            GO
            sp_spaceused 'LINEITEM'
            GO
```



```
 step_label = Execute Table Row Counts                       step_id =
 670      global_flag =    0

 sequence_no =         3  step_level =       1        parent_step_id =
637 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  1.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     11.0
 =
 step_text =  print 'Count of REGION Table'
            GO
            select count(*) from REGION
            GO
            print 'Count of NATION Table'
            GO
            select count(*) from NATION
            GO
            print 'Count of PART Table'
            GO
            select count(*) from PART
            GO
            print 'Count of SUPPLIER Table'
            GO
            select count(*) from SUPPLIER
            GO
            print 'Count of  PARTSUPP Table'
            GO
            select count(*) from PARTSUPP
            GO
            print 'Count of CUSTOMER Table'
            GO
            select count(*) from CUSTOMER
            GO
            print 'Count of ORDERS Table'
            GO
            select count(*) from ORDERS
            GO
            print 'Count of  LINEITEM Table'
            GO
            select count(*) from LINEITEM
            GO


 step_label = Execute Log File Spaceused                     step_id =
671      global_flag =    0

 sequence_no =         4  step_level =       1        parent_step_id =
637 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  1.0
```

start_directory Dynamic_Connection_to_DB
parent_version_no = 11.0
=
step_text = dbcc sqlperf(logspace)

Friday, March 09, 2001
Page 73 of 130

step_label = Execute TempDB spaceused step_id = 672 global_flag = 0

sequence_no = 5 step_level = 1 parent_step_id = 637 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 1.0

start_directory Dynamic_Connection_to_Master
parent_version_no = 11.0
=
step_text = use tempdb
GO

sp_spaceused
GO

use %DBNAME%
GO

step_label = Generate Power Queries step_id = 673 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id = 638 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 18.0
=
step_text =

step_label = Generate Stream Queries step_id = 674 global_flag = 0

sequence_no = 2 step_level = 1 parent_step_id = 638 enabled_flag = -1

iterator_name = STREAM_NUM
degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 18.0
=
step_text =

Friday, March 09, 2001
Page 74 of 130

step_label = Generate Validation Queries via QGEN
step_id = 675 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id = 639 enabled_flag = -1

iterator_name = degree_parallelism
%MAX_STREAMS%

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory
parent_version_no = 12.0
=
step_text =

step_label = Execute Validation Queries step_id = 676 global_flag = 0

sequence_no = 2 step_level = 1 parent_step_id = 639 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.1

start_directory
parent_version_no = 12.0
=
step_text =

```
step_label = Generate FlatFile                    step_id =     677
  global_flag =    0

sequence_no =        1  step_level =      2       parent_step_id =
642 enabled_flag =      0

iterator_name =     PARALLEL_PROCESS
degree_parallelism  1

execution_mechanism =     2                  continuation_criteria = 1
      failure_details =

step_file_name =
version_no =  4.0

start_directory     %FLATFILE_DIR%
parent_version_no =     10.0
=
step_text =  %TOOLS_DIR%\DBGen\dbgen -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
          -S%PARALLEL_PROCESS%
          if %PARALLEL_PROCESS% NEQ 1 goto :EOF
          if 16 NEQ 1 goto :EOF
          if exist lineitem.tbl.1 del lineitem.tbl.1
          rename lineitem.tbl lineitem.tbl.1
          if exist orders.tbl.1 del orders.tbl.1
          rename orders.tbl orders.tbl.1
          if exist customer.tbl.1 del customer.tbl.1
          rename customer.tbl customer.tbl.1
          if exist part.tbl.1 del part.tbl.1
          rename part.tbl part.tbl.1
          if exist supplier.tbl.1 del supplier.tbl.1
          rename supplier.tbl supplier.tbl.1
          if exist partsupp.tbl.1 del partsupp.tbl.1
          rename partsupp.tbl partsupp.tbl.1
```



```
step_label = Drop Tables                          step_id =     678
 global_flag =    0

sequence_no =        1  step_level =      2       parent_step_id =
652 enabled_flag =     -1

iterator_name =                          degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =
version_no =  13.0

start_directory     Dynamic_Connection_to_DB
  parent_version_no =     1.11
=
step_text =  declare @table     sysname

          declare tables cursor for
          select name from sysobjects
            where sysobjects.type='U'
          open tables
          fetch tables into @table
          while @@fetch_status = 0
           begin
             exec('drop table '+@table)
             fetch tables into @table
           end

          -- clean up after ourselves
          close tables
          deallocate tables
```

```
step_label = Create Base Tables                   step_id =
680     global_flag =    0

sequence_no =        3  step_level =      2       parent_step_id =
652 enabled_flag =     -1

iterator_name =                          degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 1
      failure_details =

step_file_name =    %SETUP_DIR%\%DBNAME%\CreateTables.sql
                     version_no =  7.0

start_directory     Dynamic_Connection_to_DB
  parent_version_no =     1.11
=
step_text =
```

```
step_label = Pin Base Tables                      step_id =     681
  global_flag =    0

sequence_no =        4  step_level =      2       parent_step_id =
652 enabled_flag =     -1

iterator_name =                          degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =
version_no =  6.0

start_directory     Dynamic_Connection_to_DB
  parent_version_no =     1.11
=
step_text =  exec sp_tableoption 'NATION','pintable',1
          exec sp_tableoption 'REGION','pintable',1
```



```
step_label = Create Insert Tables                 step_id =     682
  global_flag =    0
```

```
 sequence_no =        5  step_level =      2       parent_step_id =
652 enabled_flag =     -1

 iterator_name =     INSERT_SEGMENT
degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
%SETUP_DIR%\%DBNAME%\RefreshTables\CreateInsertTables.sql
           version_no =  7.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     1.11
=
 step_text =



 step_label = Create Delete Tables                        step_id =
683      global_flag =    0

 sequence_no =        6  step_level =      2       parent_step_id =
652 enabled_flag =     -1

 iterator_name =     DELETE_SEGMENT
degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =

%SETUP_DIR%\%DBNAME%\RefreshTables\CreateDeleteTables.s
ql              version_no =  7.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     1.11
=
 step_text =



 step_label = Load Partitioned Tables                     step_id =
684      global_flag =    0

 sequence_no =        1  step_level =      2       parent_step_id =
653 enabled_flag =      0

 iterator_name =     PARALLEL_PROCESS
degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  4.0

 start_directory     Dynamic_Connection_to_Master
    parent_version_no =     7.0
=
 step_text =  bulk insert %DBNAME%..%TABLE%
          from
'%FLATFILE_DIR%\%TABLE%.tbl.%PARALLEL_PROCESS%'
          with (FieldTerminator = '|', RowTerminator ='|\n',tablock)
```



```
 step_label = Load Nation Table                           step_id =
685      global_flag =    0

 sequence_no =        1  step_level =      2       parent_step_id =
654 enabled_flag =     -1

 iterator_name =                          degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     5.0
=
 step_text =  bulk insert %DBNAME%..NATION
          from '%FLATFILE_DIR_1%\nation.tbl'
          with (FieldTerminator = '|', RowTerminator ='|\n',tablock)



 step_label = Load Region Table                           step_id =
686      global_flag =    0

 sequence_no =        2  step_level =      2       parent_step_id =
654 enabled_flag =     -1

 iterator_name =                          degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     5.0
=
 step_text =  bulk insert %DBNAME%..REGION
          from '%FLATFILE_DIR_1%\region.tbl'
          with (FieldTerminator = '|', RowTerminator ='|\n',tablock)



 step_label = CreateIndexStream1                          step_id =
687      global_flag =    0

 sequence_no =        1  step_level =      2       parent_step_id =
657 enabled_flag =     -1

 iterator_name =                          degree_parallelism  4

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =
```

```
 step_file_name =
%SETUP_DIR%\%DBNAME%\CreateIndexesStream1.sql
       version_no =  1.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     1.0
 =
 step_text =




 Friday, March 09, 2001
    Page 78 of 130



 step_label = CreateIndexStream2                          step_id =
688      global_flag =    0

 sequence_no =        2  step_level =       2        parent_step_id =
657 enabled_flag =      -1

 iterator_name =                                degree_parallelism  4

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
%SETUP_DIR%\%DBNAME%\CreateIndexesStream2.sql
       version_no =  0.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     1.0
 =
 step_text =



 step_label = CreateIndexStream3                          step_id =
689      global_flag =    0

 sequence_no =        3  step_level =       2        parent_step_id =
657 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
%SETUP_DIR%\%DBNAME%\CreateIndexesStream3.sql
       version_no =  0.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     1.0
 =
 step_text =



 step_label = CreateIndexStream4                          step_id =
690      global_flag =    0

 sequence_no =        4  step_level =       2        parent_step_id =
657 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
%SETUP_DIR%\%DBNAME%\CreateIndexesStream4.sql
       version_no =  0.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     1.0
 =
 step_text =



 step_label = Create Statistics                           step_id =     691
  global_flag =    0

 sequence_no =        1  step_level =       2        parent_step_id =
658 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  2.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     32.0
 =
 step_text =  sp_createstats



 Friday, March 09, 2001
    Page 79 of 130



 step_label = Turn Off Auto Create Statistics                  step_id =
692      global_flag =    0

 sequence_no =        2  step_level =       2        parent_step_id =
658 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  1.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     32.0
 =
 step_text =  sp_dboption '%DBNAME%','auto create statistics','OFF'



 step_label = Turn On Update Statistics                   step_id =
693      global_flag =    0
```

```
 sequence_no =        3  step_level =      2       parent_step_id =
658 enabled_flag =     -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  2.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     32.0
 =
 step_text =  sp_dboption '%DBNAME%','auto update statistics','ON'




 step_label = Create 1st Backup Device(s) (For ACID)
step_id =     694      global_flag =    0

 sequence_no =        1  step_level =      2       parent_step_id =
661 enabled_flag =     -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
%SETUP_DIR%\%DBNAME%\CreateBackupDevices1.sql
        version_no =  3.0

 start_directory     Dynamic_Connection_to_Master
   parent_version_no =     6.5
 =
 step_text =



 step_label = Execute the 1st backup (For ACID)                 step_id
=     695      global_flag =    0

 sequence_no =        2  step_level =      2       parent_step_id =
661 enabled_flag =     -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
%SETUP_DIR%\%DBNAME%\BackupDatabase1.sql
      version_no =  3.0

 start_directory     Dynamic_Connection_to_Master
   parent_version_no =     6.5
 =
 step_text =



 Friday, March 09, 2001
   Page 80 of 130
```

```
 step_label = Install RF1 Stored Procedure(s)                   step_id =
696      global_flag =    0

 sequence_no =        1  step_level =      2       parent_step_id =
662 enabled_flag =     -1

 iterator_name =     INSERT_SEGMENT
degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =    %SETUP_DIR%\StoredProcs\CreateRF1Proc.sql
                     version_no =  6.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     9.0
 =
 step_text =



 step_label = Install RF2 Stored Procedure(s)                   step_id =
697      global_flag =    0

 sequence_no =        2  step_level =      2       parent_step_id =
662 enabled_flag =     -1

 iterator_name =     DELETE_SEGMENT
degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =    %SETUP_DIR%\StoredProcs\CreateRF2Proc.sql
                     version_no =  6.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     9.0
 =
 step_text =



 step_label = Generate Validation Queries                       step_id =
698      global_flag =    0

 sequence_no =        1  step_level =      2       parent_step_id =
675 enabled_flag =     -1

 iterator_name =     QUERY                          degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %TEMPLATE_DIR%
  parent_version_no =     1.0
 =
 step_text =  %TOOLS_DIR%\QGen\qgen -d -b
%TOOLS_DIR%\dists.dss  %QUERY% >
           %VALIDATION_DIR%\Queries\v%QUERY%.sql
```

step_label = Validation Query 1 step_id = 699 global_flag = 0

sequence_no = 1 step_level = 2 parent_step_id = 676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v1.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 2 step_id = 700 global_flag = 0

sequence_no = 2 step_level = 2 parent_step_id = 676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v2.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 3 step_id = 701 global_flag = 0

sequence_no = 3 step_level = 2 parent_step_id = 676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v3.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 4 step_id = 702 global_flag = 0

sequence_no = 4 step_level = 2 parent_step_id = 676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v4.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 5 step_id = 703 global_flag = 0

sequence_no = 5 step_level = 2 parent_step_id = 676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v5.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 6 step_id = 704 global_flag = 0

sequence_no = 6 step_level = 2 parent_step_id = 676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

```
 step_file_name =    %VALIDATION_DIR%\Queries\v6.sql
                  version_no =  1.0

 start_directory     Static_Connection_to_DB
parent_version_no =     0.1
 =
 step_text =



 step_label = Validation Query 7                          step_id =
705      global_flag =    0

 sequence_no =        7  step_level =      2       parent_step_id =
676 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =    %VALIDATION_DIR%\Queries\v7.sql
                  version_no =  1.0

 start_directory     Static_Connection_to_DB
parent_version_no =     0.1
 =
 step_text =



 step_label = Validation Query 8                          step_id =
706      global_flag =    0

 sequence_no =        8  step_level =      2       parent_step_id =
676 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =    %VALIDATION_DIR%\Queries\v8.sql
                  version_no =  1.0

 start_directory     Static_Connection_to_DB
parent_version_no =     0.1
 =
 step_text =



 step_label = Validation Query 9                          step_id =
707      global_flag =    0

 sequence_no =        9  step_level =      2       parent_step_id =
676 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =    %VALIDATION_DIR%\Queries\v9.sql
                  version_no =  1.0

 start_directory     Static_Connection_to_DB
parent_version_no =     0.1
 =
 step_text =



Friday, March 09, 2001
   Page 83 of 130



 step_label = Validation Query 10                         step_id =
708      global_flag =    0

 sequence_no =       10  step_level =      2       parent_step_id =
676 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =    %VALIDATION_DIR%\Queries\v10.sql
                  version_no =  1.0

 start_directory     Static_Connection_to_DB
parent_version_no =     0.1
 =
 step_text =



 step_label = Validation Query 11                         step_id =
709      global_flag =    0

 sequence_no =       11  step_level =      2       parent_step_id =
676 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =    %VALIDATION_DIR%\Queries\v11.sql
                  version_no =  1.0

 start_directory     Static_Connection_to_DB
parent_version_no =     0.1
 =
 step_text =



 step_label = Validation Query 12                         step_id =
710      global_flag =    0

 sequence_no =       12  step_level =      2       parent_step_id =
676 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =    %VALIDATION_DIR%\Queries\v12.sql
                  version_no =  1.0
```

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 13 step_id =
711 global_flag = 0

sequence_no = 13 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v13.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 14 step_id =
712 global_flag = 0

sequence_no = 14 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v14.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 15 step_id =
713 global_flag = 0

sequence_no = 15 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v15.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 16 step_id =
714 global_flag = 0

sequence_no = 16 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v16.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 17 step_id =
715 global_flag = 0

sequence_no = 17 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v17.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 18 step_id =
716 global_flag = 0

sequence_no = 18 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v18.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 19 step_id =
717 global_flag = 0

sequence_no = 19 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v19.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 20 step_id =
718 global_flag = 0

sequence_no = 20 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v20.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Validation Query 21 step_id =
719 global_flag = 0

sequence_no = 21 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v21.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

Friday, March 09, 2001
Page 86 of 130

step_label = Validation Query 22 step_id =
720 global_flag = 0

sequence_no = 22 step_level = 2 parent_step_id =
676 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %VALIDATION_DIR%\Queries\v22.sql
version_no = 1.0

start_directory Static_Connection_to_DB
parent_version_no = 0.1
=
step_text =

step_label = Generate Power Query Directory step_id =
721 global_flag = 0

sequence_no = 1 step_level = 2 parent_step_id =
673 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 1
failure_details =

step_file_name =
version_no = 0.0

start_directory %RUN_DIR%
parent_version_no = 0.0
=
step_text = if not exist %QUERY_DIR%\Power mkdir
%QUERY_DIR%\Power

step_label = Generate QGen Command File step_id =
722 global_flag = 0

sequence_no = 2 step_level = 2 parent_step_id =
673 enabled_flag = -1

iterator_name = degree_parallelism 1

```
 execution_mechanism =     2                    continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %TEMPLATE_DIR%
  parent_version_no =     0.0
 =
 step_text =  ::
            :: First use OSQL to grab the QgenSeed value from
TPCH_AUX_TABLE
            ::
            osql -Usa -P -n -d%DBNAME% -w 255
-i%SETUP_DIR%\Generate\GenQGENcmd.sql>
            %SETUP_DIR%\Generate\QgenCmd.cmd
```



```
 step_label = Parallel Power Query Generation                step_id =
  723      global_flag =    0

 sequence_no =         3  step_level =       2        parent_step_id =
673 enabled_flag =      -1

 iterator_name =                              degree_parallelism
%MAX_STREAMS%

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory
parent_version_no =     0.0
 =
 step_text =


 step_label = Set Seed Value for Stream                      step_id =
724      global_flag =    0

 sequence_no =         1  step_level =       2        parent_step_id =
674 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     0.0
 =
 step_text =  --
            --   Increment QGen Seed in TPCH_AUX_TABLE
            --
            UPDATE   TPCH_AUX_TABLE
            SET          QgenSeed = QgenSeed + %STREAM_NUM%


 step_label = Create Query Stream Directories                step_id =
 725      global_flag =    0

 sequence_no =         2  step_level =       2        parent_step_id =
674 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %QUERY_DIR%
parent_version_no =     0.0
 =
 step_text =  if not exist
%QUERY_DIR%\STREAM%STREAM_NUM% mkdir
%QUERY_DIR%\STREAM%STREAM_NUM%
```



```
 step_label = Generate QGen Stream Command File
step_id =     726      global_flag =    0

 sequence_no =         3  step_level =       2        parent_step_id =
674 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %TEMPLATE_DIR%
  parent_version_no =     0.0
 =
 step_text =  ::
            :: First use OSQL to grab the QgenSeed value from
TPCH_AUX_TABLE
            ::
            osql -Usa -P -n -d%DBNAME% -w 255
-i%SETUP_DIR%\Generate\GenQgenStreamCmd.sql>
            %SETUP_DIR%\Generate\QgenStreamCmd.cmd


 step_label = Parallel Stream Query Generation               step_id =
  727      global_flag =    0

 sequence_no =         4  step_level =       2        parent_step_id =
674 enabled_flag =      -1
```

```
 iterator_name =                          degree_parallelism
%MAX_STREAMS%

 execution_mechanism =     0              continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory
parent_version_no =     0.0
 =
 step_text =



 step_label = Re-set QGen Seed Value                    step_id =
728      global_flag =    0

 sequence_no =        5  step_level =       2        parent_step_id =
674 enabled_flag =      -1

 iterator_name =                          degree_parallelism  1

 execution_mechanism =     1              continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     0.0
 =
 step_text =  --
          --   Resets the QGen seed kept in the temporary table
          --
          UPDATE    TPCH_AUX_TABLE
          SET          QgenSeed  =  QgenSeed - %STREAM_NUM%



 Friday, March 09, 2001
    Page 89 of 130



 step_label = Power - Execute Generated QGen Command File
step_id =     729      global_flag =    0

 sequence_no =        1  step_level =       3        parent_step_id =
723 enabled_flag =      -1

 iterator_name =     QUERY                    degree_parallelism  1

 execution_mechanism =     2              continuation_criteria = 1
       failure_details =

 step_file_name =    %SETUP_DIR%\Generate\QgenCmd.cmd
                  version_no =  0.0

 start_directory     %TEMPLATE_DIR%
  parent_version_no =     0.0
 =
 step_text =


 step_label = Throughput - Execute Generated QGen Command File
step_id =     730      global_flag =    0

 sequence_no =        1  step_level =       3        parent_step_id =
727 enabled_flag =      -1

 iterator_name =     QUERY                    degree_parallelism  1

 execution_mechanism =     2              continuation_criteria = 2
       failure_details =

 step_file_name =    %SETUP_DIR%\Generate\QgenStreamCmd.cmd
                  version_no =  0.0

 start_directory     %TEMPLATE_DIR%
  parent_version_no =     0.0
 =
 step_text =



 step_label = Create Auxilliary Table                   step_id =
812      global_flag =    0

 sequence_no =        2  step_level =       2        parent_step_id =
652 enabled_flag =      -1

 iterator_name =                          degree_parallelism  1

 execution_mechanism =     1              continuation_criteria = 2
       failure_details =

 step_file_name =    %SETUP_DIR%\Utility\CreateBuildTimer.sql
                  version_no =  5.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     1.11
 =
 step_text =



 step_label = Ensure DB_Option "Select Into" is True          step_id
=     813      global_flag =    0

 sequence_no =        2  step_level =       1        parent_step_id =
635 enabled_flag =      -1

 iterator_name =                          degree_parallelism  1

 execution_mechanism =     1              continuation_criteria = 1
       failure_details =

 step_file_name =
version_no =  14.0

 start_directory     Static_Connection_to_Master
 parent_version_no =     30.23
 =
 step_text =  sp_dboption %DBNAME%, 'select ',true



 Friday, March 09, 2001
    Page 90 of 130
```

```
 step_label = Relocate Validation Query Output Files              step_id
=    869     global_flag =    0

 sequence_no =        3  step_level =      1       parent_step_id =
639 enabled_flag =     -1

 iterator_name =    QUERY                        degree_parallelism  1

 execution_mechanism =    2                    continuation_criteria = 2
      failure_details =

 step_file_name =
version_no =  1.0

 start_directory    %VALIDATION_DIR%
   parent_version_no =    12.0
 =
 step_text =  ::
          :: Copy the StepMaster Query output file to the
Validation\Run_Output directory
          ::
          copy %OUTPUT_DIR%\Validation_Query_%QUERY%.out


%VALIDATION_DIR%\RUN_OUTPUT\Validation_Query_%QUER
Y%.out




 step_label = Generate Query Plans                          step_id =
917     global_flag =    0

 sequence_no =       12  step_level =      0       parent_step_id =
0 enabled_flag =      0

 iterator_name =                              degree_parallelism  1

 execution_mechanism =    0                    continuation_criteria = 0
      failure_details =

 step_file_name =
version_no =  10.0

 start_directory
parent_version_no =    0.0
 =
 step_text =




 Friday, March 09, 2001
    Page 91 of 130




 step_label = Execute SHOWPLAN ALL for Each Query
step_id =    918     global_flag =    0

 sequence_no =        1  step_level =      1       parent_step_id =
917 enabled_flag =     -1

 iterator_name =    QUERY                        degree_parallelism  1

 execution_mechanism =    2                    continuation_criteria = 2
      failure_details =

 step_file_name =
version_no =  9.0

 start_directory    %SETUP_DIR%
parent_version_no =    10.0
 =
 step_text =  ::
          :: First Build the query with the appropriate SQL ShowPlan
commands
          ::
          copy
%SETUP_DIR%\showplan_sql\set_showplan_all_on.sql

%SETUP_DIR%\showplan_queries\SP_ALL_%QUERY%.SQL
          type  %QUERY_DIR%\Power\%QUERY%.SQL
        >>

%SETUP_DIR%\showplan_queries\SP_ALL_%QUERY%.SQL
          type
%SETUP_DIR%\showplan_sql\set_showplan_all_off.sql      >>

%SETUP_DIR%\showplan_queries\SP_ALL_%QUERY%.SQL

          ::
          ::  Now execute the query, placing the output in
%SETUP_DIR%\Showplan_Out
          ::
          osql -l -t600 -Usa -P -d %DBNAME% -e -i
%SETUP_DIR%\showplan_queries\SP_ALL_%QUERY%.sql -w
4096
          -o
%SETUP_DIR%\showplan_Out\SP_ALL_%QUERY%.out




 step_label = Execute SHOWPLAN TEXT for Each Query
step_id =    919     global_flag =    0

 sequence_no =        2  step_level =      1       parent_step_id =
917 enabled_flag =     -1

 iterator_name =    QUERY                        degree_parallelism  1

 execution_mechanism =    2                    continuation_criteria = 2
      failure_details =

 step_file_name =
version_no =  9.0

 start_directory    %SETUP_DIR%
parent_version_no =    10.0
 =
 step_text =  ::
          ::  First Build the query with the appropriate SQL ShowPlan
commands
          ::
          copy
%SETUP_DIR%\showplan_sql\set_showplan_text_on.sql

%SETUP_DIR%\showplan_queries\SP_TEXT_%QUERY%.SQL
          type  %QUERY_DIR%\Power\%QUERY%.SQL
        >>
```

```
%SETUP_DIR%\showplan_queries\SP_TEXT_%QUERY%.SQL
        type
%SETUP_DIR%\showplan_sql\set_showplan_text_off.sql      >>

%SETUP_DIR%\showplan_queries\SP_TEXT_%QUERY%.SQL

        ::
        ::  Now execute the query, placing the output in
%SETUP_DIR%\Showplan_Out
        ::
        osql -l -t600 -Usa -P -d %DBNAME% -e -i
%SETUP_DIR%\showplan_queries\SP_TEXT_%QUERY%.sql -w
4096
         -o
%SETUP_DIR%\showplan_Out\SP_TEXT_%QUERY%.out
```



```
step_label = Generate Flat Files to FlatFile_Dir_1              step_id =
  920      global_flag =    0

sequence_no =        2  step_level =       2        parent_step_id =
642 enabled_flag =      -1

iterator_name =     PARALLEL_PROCESS
degree_parallelism  3

execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

step_file_name =
version_no =  6.0

start_directory     %FLATFILE_DIR_1%
parent_version_no =     10.0
=
step_text =  :: Generate Data for REGION and NATION Tables
        %TOOLS_DIR%\DBGen\dbgen -T l -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for CUSTOMER Table
        %TOOLS_DIR%\DBGen\dbgen -T c -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for ORDERS and LINEITEM Tables
        %TOOLS_DIR%\DBGen\dbgen -T o -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for PARTS/PARTSUPP Table
        %TOOLS_DIR%\DBGen\dbgen -T p -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for SUPPLIER Table
        %TOOLS_DIR%\DBGen\dbgen -T s -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%
```



```
step_label = Generate Flat Files to FlatFile_Dir_2              step_id =
  921      global_flag =    0

sequence_no =        3  step_level =       2        parent_step_id =
642 enabled_flag =      -1

iterator_name =     PARALLEL_PROCESS
degree_parallelism  1

execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

step_file_name =
version_no =  4.0

start_directory     %FLATFILE_DIR_2%
parent_version_no =     10.0
=
step_text =  :: Generate Data for CUSTOMER Table
        %TOOLS_DIR%\DBGen\dbgen -T c -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for ORDERS and LINEITEM Tables
        %TOOLS_DIR%\DBGen\dbgen -T o -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for PARTS/PARTSUPP Table
        %TOOLS_DIR%\DBGen\dbgen -T p -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for SUPPLIER Table
        %TOOLS_DIR%\DBGen\dbgen -T s -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%



step_label = Generate Flat Files to FlatFile_Dir_3              step_id =
  922      global_flag =    0

sequence_no =        4  step_level =       2        parent_step_id =
642 enabled_flag =      -1

iterator_name =     PARALLEL_PROCESS
degree_parallelism  1

execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

step_file_name =
version_no =  3.0

start_directory     %FLATFILE_DIR_3%
parent_version_no =     10.0
=
step_text =  :: Generate Data for CUSTOMER Table
```

```
        %TOOLS_DIR%\DBGen\dbgen -T c -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for ORDERS and LINEITEM Tables
        %TOOLS_DIR%\DBGen\dbgen -T o -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for PARTS/PARTSUPP Table
        %TOOLS_DIR%\DBGen\dbgen -T p -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for SUPPLIER Table
        %TOOLS_DIR%\DBGen\dbgen -T s -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%
```



```
step_label = Generate Flat Files to FlatFile_Dir_4          step_id =
 923    global_flag =   0

sequence_no =       5  step_level =     2      parent_step_id =
642 enabled_flag =    -1

iterator_name =    PARALLEL_PROCESS
degree_parallelism  1

execution_mechanism =    2                  continuation_criteria = 2
      failure_details =

step_file_name =
version_no =  4.0

start_directory    %FLATFILE_DIR_4%
parent_version_no =    10.0
=
step_text =  :: Generate Data for CUSTOMER Table
        %TOOLS_DIR%\DBGen\dbgen -T c -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for ORDERS and LINEITEM Tables
        %TOOLS_DIR%\DBGen\dbgen -T o -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for PARTS/PARTSUPP Table
        %TOOLS_DIR%\DBGen\dbgen -T p -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%

        :: Generate Data for SUPPLIER Table
        %TOOLS_DIR%\DBGen\dbgen -T s -fF -q -b
%TOOLS_DIR%\dists.dss -s%SCALEFACTOR% -C16
        -S%PARALLEL_PROCESS%
```

```
step_label = Load Partitioned Tables from FlatFile_Dir_1
step_id =    924     global_flag =   0

sequence_no =       2  step_level =     2      parent_step_id =
653 enabled_flag =    -1

iterator_name =    PARALLEL_PROCESS
degree_parallelism  1

execution_mechanism =    1                  continuation_criteria = 2
      failure_details =

step_file_name =
version_no =  6.0

start_directory    Dynamic_Connection_to_Master
   parent_version_no =    7.0
=
step_text =  bulk insert %DBNAME%..%TABLE%
           from
'%FLATFILE_DIR_1%\%TABLE%.tbl.%PARALLEL_PROCESS%'
           with (FieldTerminator = '|', RowTerminator ='|\n',tablock)
```



```
step_label = Load Partitioned Tables from FlatFile_Dir_2
step_id =    925     global_flag =   0

sequence_no =       3  step_level =     2      parent_step_id =
653 enabled_flag =    -1

iterator_name =    PARALLEL_PROCESS
degree_parallelism  1

execution_mechanism =    1                  continuation_criteria = 2
      failure_details =

step_file_name =
version_no =  4.0

start_directory    Dynamic_Connection_to_Master
   parent_version_no =    7.0
=
step_text =  bulk insert %DBNAME%..%TABLE%
           from
'%FLATFILE_DIR_2%\%TABLE%.tbl.%PARALLEL_PROCESS%'
           with (FieldTerminator = '|', RowTerminator ='|\n',tablock)


step_label = Load Partitioned Tables from FlatFile_Dir_3
step_id =    926     global_flag =   0

sequence_no =       4  step_level =     2      parent_step_id =
653 enabled_flag =    -1

iterator_name =    PARALLEL_PROCESS
degree_parallelism  1
```

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 4.0

start_directory Dynamic_Connection_to_Master
parent_version_no = 7.0
=
step_text = bulk insert %DBNAME%..%TABLE%
from
'%FLATFILE_DIR_3%\%TABLE%.tbl.%PARALLEL_PROCESS%'
with (FieldTerminator = '|', RowTerminator ='|\n',tablock)

step_label = Load Partitioned Tables from FlatFile_Dir_4
step_id = 927 global_flag = 0

sequence_no = 5 step_level = 2 parent_step_id =
653 enabled_flag = -1

iterator_name = PARALLEL_PROCESS
degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 4.0

start_directory Dynamic_Connection_to_Master
parent_version_no = 7.0
=
step_text = bulk insert %DBNAME%..%TABLE%
from
'%FLATFILE_DIR_4%\%TABLE%.tbl.%PARALLEL_PROCESS%'
with (FieldTerminator = '|', RowTerminator ='|\n',tablock)

step_label = Create 2nd Backup Device(s) (For ACID)
step_id = 933 global_flag = 0

sequence_no = 3 step_level = 2 parent_step_id =
661 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
%SETUP_DIR%\%DBNAME%\CreateBackupDevices2.sql
version_no = 2.0

start_directory Dynamic_Connection_to_Master
parent_version_no = 6.5
=
step_text =

step_label = Execute the 2nd backup (For ACID) step_id
= 934 global_flag = 0

sequence_no = 4 step_level = 2 parent_step_id =
661 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
%SETUP_DIR%\%DBNAME%\BackupDatabase2.sql
version_no = 2.0

start_directory Dynamic_Connection_to_Master
parent_version_no = 6.5
=
step_text =

step_label = Shutdown SQL Server step_id =
939 global_flag = 0

sequence_no = 13 step_level = 0 parent_step_id =
0 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 2.0

start_directory
parent_version_no = 0.0
=
step_text =

step_label = Shutdown SQL Server1 step_id =
941 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id =
939 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 1
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 2.0
=
step_text = isql -Usa -P -t10 -Q"shutdown"

step_label = Wait 5 minutes step_id = 942
global_flag = -1

sequence_no = 4 step_level = 0 parent_step_id = 0 enabled_flag = 0

iterator_name = degree_parallelism 0

execution_mechanism = 2 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 1.0

start_directory %TOOLS_DIR%
parent_version_no = 0.0
=
step_text = %TOOLS_DIR%\Utility\sleep.exe 300

step_label = Move TempDB step_id = 943 global_flag = 0

sequence_no = 2 step_level = 1 parent_step_id = 632 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
D:\MSTPCH.101D\setup\tpch100g\tempDB\MoveTempDB.sql
version_no = 9.0

start_directory Dynamic_Connection_to_DB
parent_version_no = 55.0
=
step_text =

step_label = Resize TempDB1 step_id = 944 global_flag = 0

sequence_no = 8 step_level = 1 parent_step_id = 632 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
D:\MSTPCH.101D\setup\tpch100g\tempDB\ResizeTempDB.sql
version_no = 7.0

start_directory Dynamic_Connection_to_DB
parent_version_no = 55.0
=
step_text =

step_label = Resize TempDB2 step_id = 945 global_flag = 0

sequence_no = 9 step_level = 1 parent_step_id = 632 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
D:\MSTPCH.101D\setup\tpch100g\tempDB\ResizeTempDB2.sql
version_no = 7.0

start_directory Dynamic_Connection_to_DB
parent_version_no = 55.0
=
step_text =

step_label = Move TempLog step_id = 946 global_flag = 0

sequence_no = 3 step_level = 1 parent_step_id = 632 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
D:\MSTPCH.101D\setup\tpch100g\tempDB\MoveTempLog.sql
version_no = 10.0

start_directory Dynamic_Connection_to_DB
parent_version_no = 55.0
=
step_text =

step_label = Resize TempLog step_id = 947 global_flag = 0

sequence_no = 10 step_level = 1 parent_step_id = 632 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

```
 step_file_name =
D:\MSTPCH.101D\setup\tpch100g\tempDB\ReSizeTemplog.sql
      version_no =  6.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     55.0
 =
 step_text =



 step_label = Wait for 3 minutes                          step_id =     948
    global_flag =   -1

 sequence_no =        5  step_level =       0       parent_step_id =
0 enabled_flag =       0

 iterator_name =                              degree_parallelism  0

 execution_mechanism =     2                  continuation_criteria = 0
      failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %TOOLS_DIR%
parent_version_no =     0.0
 =
 step_text =  %TOOLS_DIR%\Utility\sleep.exe 180



 step_label = Start SqlServer 1                           step_id =     949
   global_flag =    0

 sequence_no =        5  step_level =       1       parent_step_id =
632 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     2                  continuation_criteria = 1
      failure_details =

 step_file_name =
version_no =  8.0

 start_directory     %TOOLS_DIR%
parent_version_no =     55.0
 =
 step_text =  start sqlservr %TRACEFLAGS%




 Friday, March 09, 2001
    Page 99 of 130



 step_label = Shutdown SQL Server11                       step_id =
950      global_flag =    0

 sequence_no =        4  step_level =       1       parent_step_id =
632 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     2                  continuation_criteria = 1
      failure_details =

 step_file_name =
version_no =  8.0

 start_directory     %TOOLS_DIR%
parent_version_no =     55.0
 =
 step_text =  isql -Usa -P -t10 -Q"shutdown"



 step_label = Sleep for 3 minutes                         step_id =     951
    global_flag =    0

 sequence_no =        6  step_level =       1       parent_step_id =
632 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     2                  continuation_criteria = 2
      failure_details =

 step_file_name =
version_no =  6.0

 start_directory     %TOOLS_DIR%
parent_version_no =     55.0
 =
 step_text =  %TOOLS_DIR%\Utility\sleep.exe 180



 step_label = dummy step                                  step_id =     952
   global_flag =    0

 sequence_no =        7  step_level =       1       parent_step_id =
632 enabled_flag =      -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

 step_file_name =
version_no =  5.0

 start_directory     Dynamic_Connection_to_DB
   parent_version_no =     55.0
 =
 step_text =  sp_who



 Friday, March 09, 2001
   Page 100 of 130
```

step_label = dummy step1 step_id = 953
global_flag = 0

sequence_no = 12 step_level = 1 parent_step_id = 632 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 4.0

start_directory Dynamic_Connection_to_DB
parent_version_no = 55.0
=
step_text = sp_who

step_label = Set Recovery Model Bulk_Logged step_id = 956 global_flag = 0

sequence_no = 4 step_level = 2 parent_step_id = 658 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory Dynamic_Connection_to_DB
parent_version_no = 32.0
=
step_text = alter database %DBNAME% set recovery bulk_logged

Step_constraints

workspace_id	constraint_id	step_id	version_no	constraint_type	global_state	global_version_no	sequence_no
8	86	950	8.0	2	627	7.0	0
8	87	952	5.0	2	627	7.0	0
8	16	651	22.0	2	625	5.0	1
8	83	651	22.0	2	942	1.0	3
8	18	651	22.0	2	626	3.0	0
8	21	651	22.0	2	627	7.0	2
8	77	678	13.0	1	627	7.0	0
8	19	644	3.0	2	627	7.0	0

Iterator_values

workspace_id	step_id	version_no	type	iterator_value	sequence_no
8	919	9.0	1	1	0
8	921	4.0	3	1	0
8	869	1.0	3	1	0
8	869	1.0	2	22	0
8	918	9.0	2	22	0
8	918	9.0	3	1	0
8	918	9.0	1	1	0
8	730	0.0	1	1	0
8	919	9.0	2	22	0
8	730	0.0	3	1	0
8	677	4.0	1	1	0
8	677	4.0	3	1	0
8	677	4.0	2	16	0
8	927	4.0	2	16	0

8 927 4.0 3 1
0

8 927 4.0 1 13
0

8 919 9.0 3 1
0

8 698 0.0 2 22
0

8 921 4.0 2 8
0

8 921 4.0 1 5
0

8 674 0.0 3 1
0

8 674 0.0 2 %MAX_STREAMS%
0

8 674 0.0 1 1
0

8 684 4.0 1 1
0

8 869 1.0 1 1
0

8 684 4.0 2 %DBGEN_SETS%
0

8 926 4.0 1 9
0

8 698 0.0 3 1
0

8 698 0.0 1 1
0

8 729 0.0 2 22
0

8 729 0.0 3 1
0

8 729 0.0 1 1
0

8 730 0.0 2 22
0

8 684 4.0 3 1
0

8 682 7.0 2
%INSERT_SEGMENTS_PER_UPDATE_SET% 0

8 922 3.0 3 1
0

8 920 6.0 3 1
0

8 920 6.0 2 4
0

8 920 6.0 1 1
0

8 683 7.0 2
%DELETE_SEGMENTS_PER_UPDATE_SET% 0

8 683 7.0 1 1
0

8 926 4.0 3 1
0

8 682 7.0 3 1
0

8 923 4.0 3 1
0

8 682 7.0 1 1
0

8 653 7.0 4 PARTSUPP
4

8 653 7.0 4 CUSTOMER
5

8 653 7.0 4 SUPPLIER
3

8 653 7.0 4 PART
2

8 653 7.0 4 LINEITEM
1

8 683 7.0 3 1
0

8 697 6.0 2
%DELETE_SEGMENTS_PER_UPDATE_SET% 0

8 653 7.0 4 ORDERS
0

8 925 4.0 3 1
0

8 925 4.0 2 8
0

8 925 4.0 1 5
0

8 924 6.0 1 1
0

8 924 6.0 2 4
0

8 922 3.0 2 12
0

8 697 6.0 1 1
0

8 922 3.0 1 9
0

8 697 6.0 3 1
0

8 696 6.0 3 1
0

8 696 6.0 1 1
0

8 696 6.0 2
%INSERT_SEGMENTS_PER_UPDATE_SET% 0

8 923 4.0 1 13
0

8 923 4.0 2 16
0

8 926 4.0 2 12
0

8 924 6.0 3 1
0

9 Power/Throughput Runs - TPC-H Kit V1.01d (Spec Revision 0
1.2.0)

Workspace_parameters

workspace_id = 9 parameter_id 116 parameter_type 3
parameter_name RUN_ID
parameter_valu 558

workspace_id = 9 parameter_id 113 parameter_type 0
parameter_name KIT_DIR
parameter_valu d:\mstpch.101d

workspace_id = 9 parameter_id 112 parameter_type 0
parameter_name BATCH_SIZE
parameter_valu 100

workspace_id = 9 parameter_id 111 parameter_type 0
parameter_name TOOLS_DIR
parameter_valu %KIT_DIR%\Tools

workspace_id = 9 parameter_id 110 parameter_type 0
parameter_name RF_FLATFILE_DIR
parameter_valu L:\RF_Flat_Files

workspace_id = 9 parameter_id 109 parameter_type 3
parameter_name DEFAULT_DIR
parameter_valu %KIT_DIR%\output

workspace_id = 9 parameter_id 107 parameter_type 0
parameter_name MAX_STREAMS
parameter_valu 5

workspace_id = 9 parameter_id 106 parameter_type 0
parameter_name INSERT_SEGMENTS_PER_UPDATE_SET
parameter_valu 16

workspace_id = 9 parameter_id 105 parameter_type 0
parameter_name DELETE_SEGMENTS_PER_UPDATE_SET
parameter_valu 16

workspace_id = 9 parameter_id 104 parameter_type 0

parameter_name INSERT_PARALLELISM
parameter_valu 16

workspace_id = 9 parameter_id 103 parameter_type 0

parameter_name DELETE_PARALLELISM

parameter_valu 16

workspace_id = 9 parameter_id 102 parameter_type 0
parameter_name DBNAME

parameter_valu tpch100g

workspace_id = 9 parameter_id 101 parameter_type 0

parameter_name QUERY_DIR
parameter_valu %RUN_DIR%\Queries

workspace_id = 9 parameter_id 100 parameter_type 3

parameter_name OUTPUT_DIR

parameter_valu d:\MSTPCH~1.101\output\558

workspace_id = 9 parameter_id 99 parameter_type 0
parameter_name RUN_DIR

parameter_valu %KIT_DIR%\Run

Connection_dtls

workspac connection_na connection_name connection_string_name connection
e_id me_id
_type

9 38 Power_Dynamic_DB_Connection DBCONNECTION 2

9 37 Throughput_Static_Stream_10_Connection DBCONNECTION 1

9 36 Throughput_Static_Stream_9_Connection DBCONNECTION 1

9 35 Throughput_Static_Stream_8_Connection DBCONNECTION 1

9 34 Throughput_Static_Stream_7_Connection DBCONNECTION 1

9 33 Throughput_Static_Stream_6_Connection DBCONNECTION 1

9 32 Throughput_Static_Stream_5_Connection DBCONNECTION 1

9 31 Throughput_Static_Stream_4_Connection DBCONNECTION 1

9 30 Throughput_Static_Stream_3_Connection DBCONNECTION 1

9 29 Throughput_Static_Stream_2_Connection DBCONNECTION 1

9 28 Throughput_Static_Stream_1_Connection DBCONNECTION 1

9 27 Throughput_RF_Connection DBCONNECTION 2

9 26 Dynamic_Connection_to_DB DBCONNECTION 2

9 25 Power_Static_DB_Connection DBCONNECTION 1

Workspace_connections

workspace_i 9

connection_i 12
connection_name MASTERCONNECTION

connection_valu DRIVER=SQL Server;SERVER=;UID=sa;PWD=;

descripti
no_count_displa 0

no_execut 0

parse_query_onl 0
ANSI_quoted_identifie 0

```
ANSI_nulls                -1

show_query_plan            0
show_stats_tim             0

show_stats_i               0

parse_odbc_msg_prefix      -1
row_count                  0

tsql_batch_separa      GO

query_time_ou              0
server_languag         (Default)

character_translat         -1

regional_setti             0

workspace_i                9

connection_i               11
connection_name        DBCONNECTION

connection_valu        DRIVER=SQL
Server;SERVER=;UID=sa;PWD=;DATABASE=%DBNAME%;

descripti
no_count_displa            0

no_execut                  0

parse_query_onl            0
ANSI_quoted_identifie      0

ANSI_nulls                -1

show_query_plan            0
show_stats_tim             0

show_stats_i               0

Friday, March 09, 2001
  Page 107 of 130

tsql_batch_separa      GO
query_time_ou              0
server_languag         (Default)
character_translat         -1

o
d
es


tor
e
t
e
y
ion
e
regional_setti             0

rs
 Att_steps

 workspace_id =     9
e


 step_label = Clear any Outstanding Semaphores                step_id
=    731     global_flag =    0

 sequence_no =       1  step_level =      0       parent_step_id =
0 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     2                  continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  6.0

 start_directory     %TOOLS_DIR%
parent_version_no =     0.0
 =
 step_text =  ::
           ::  This step must always be run to insure that a
semaphore.exe was not left open by a
           ::  previous run
           ::
           ::  If there are no open semaphore.exe's then the 'KILL' will
do nothing.
           ::
           %TOOLS_DIR%\Utility\KILL.EXESEMAPHORE.EXE




 step_label = Execute Power Run                          step_id =
733     global_flag =    0

 sequence_no =       2  step_level =      0       parent_step_id =
0 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     0                  continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  9.3

 start_directory
parent_version_no =     0.0
 =
 step_text =



 Friday, March 09, 2001
   Page 108 of 130




 step_label = Execute Throughput Run                          step_id =
734     global_flag =    0

 sequence_no =       3  step_level =      0       parent_step_id =
0 enabled_flag =     -1
```

iterator_name = degree_parallelism 2

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 41.0

start_directory
parent_version_no = 0.0
=
step_text =

step_label = Power - Parallel RF1 Execution step_id =
736 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id =
733 enabled_flag = -1

iterator_name = degree_parallelism
%INSERT_PARALLELISM%

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 10.0

start_directory
parent_version_no = 9.3
=
step_text =

step_label = Power - Sequential Query Execution step_id
= 737 global_flag = 0

sequence_no = 2 step_level = 1 parent_step_id =
733 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 19.0

start_directory
parent_version_no = 9.3
=
step_text =

step_label = Power - Parallel RF2 Execution step_id =
738 global_flag = 0

sequence_no = 3 step_level = 1 parent_step_id =
733 enabled_flag = -1

iterator_name = degree_parallelism
%DELETE_PARALLELISM%

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 12.0

start_directory
parent_version_no = 9.3
=
step_text =

Friday, March 09, 2001
Page 109 of 130

step_label = Power - Increment Update Set step_id =
739 global_flag = 0

sequence_no = 4 step_level = 1 parent_step_id =
733 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 10.0

start_directory Dynamic_Connection_to_DB
parent_version_no = 9.3
=
step_text = UPDATE TPCH_AUX_TABLE SET
updateset=updateset+1

step_label = Sequential Refresh Stream Execution step_id
= 740 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id =
734 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.5

start_directory
parent_version_no = 41.0
=
step_text =

```
 step_label = Parallel Stream Execution                    step_id =
741     global_flag =    0

 sequence_no =        2  step_level =      1       parent_step_id =
734 enabled_flag =     -1

 iterator_name =                              degree_parallelism
%MAX_STREAMS%

 execution_mechanism =     0                  continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  3.0

 start_directory
parent_version_no =     41.0
 =
 step_text =
```



```
 step_label = Power - Execute RF1                          step_id =
742     global_flag =    0

 sequence_no =        1  step_level =      2       parent_step_id =
736 enabled_flag =     -1

 iterator_name =     INSERT_SEGMENT
degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  8.0

 start_directory     Power_Dynamic_DB_Connection
       parent_version_no =     10.0
 =
 step_text =  --
           --  Execute the Refresh RF1 Stored Procedure
           --

           EXEC  RF1_%INSERT_SEGMENT% %BATCH_SIZE%
```

```
 step_label = Power - Execute Query 14                     step_id =
743     global_flag =    0

 sequence_no =        1  step_level =      2       parent_step_id =
737 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =    %QUERY_DIR%\Power\14.sql
              version_no =  6.0

 start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
 =
 step_text =
```

```
 step_label = Power - Execute Query 02                     step_id =
744     global_flag =    0

 sequence_no =        2  step_level =      2       parent_step_id =
737 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =    %QUERY_DIR%\Power\2.sql
              version_no =  7.0

 start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
 =
 step_text =
```



```
 step_label = Power - Execute Query 09                     step_id =
745     global_flag =    0

 sequence_no =        3  step_level =      2       parent_step_id =
737 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =

 step_file_name =    %QUERY_DIR%\Power\9.sql
              version_no =  6.0

 start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
 =
 step_text =
```

```
 step_label = Power - Execute Query 20                     step_id =
746     global_flag =    0

 sequence_no =        4  step_level =      2       parent_step_id =
737 enabled_flag =     -1

 iterator_name =                              degree_parallelism  1

 execution_mechanism =     1                  continuation_criteria = 2
       failure_details =
```

```
step_file_name =    %QUERY_DIR%\Power\20.sql
             version_no =  6.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =



step_label = Power - Execute Query 06                    step_id =
747     global_flag =    0

sequence_no =        5  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\6.sql
             version_no =  8.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =



step_label = Power - Execute Query 17                    step_id =
748     global_flag =    0

sequence_no =        6  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\17.sql
             version_no =  8.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =
```



```
step_label = Power - Execute Query 18                    step_id =
749     global_flag =    0

sequence_no =        7  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\18.sql
             version_no =  7.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =



step_label = Power - Execute Query 08                    step_id =
750     global_flag =    0

sequence_no =        8  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\8.sql
             version_no =  7.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =



step_label = Power - Execute Query 21                    step_id =
751     global_flag =    0

sequence_no =        9  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\21.sql
             version_no =  7.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =



step_label = Power - Execute Query 13                    step_id =
752     global_flag =    0

sequence_no =       10  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =
```

step_file_name = %QUERY_DIR%\Power\13.sql
version_no = 8.0

start_directory Power_Static_DB_Connection
parent_version_no = 19.0
=
step_text =

step_label = Power - Execute Query 03 step_id =
753 global_flag = 0

sequence_no = 11 step_level = 2 parent_step_id =
737 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %QUERY_DIR%\Power\3.sql
version_no = 8.0

start_directory Power_Static_DB_Connection
parent_version_no = 19.0
=
step_text =

step_label = Power - Execute Query 22 step_id =
754 global_flag = 0

sequence_no = 12 step_level = 2 parent_step_id =
737 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %QUERY_DIR%\Power\22.sql
version_no = 8.0

start_directory Power_Static_DB_Connection
parent_version_no = 19.0
=
step_text =

step_label = Power - Execute Query 16 step_id =
755 global_flag = 0

sequence_no = 13 step_level = 2 parent_step_id =
737 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %QUERY_DIR%\Power\16.sql
version_no = 7.0

start_directory Power_Static_DB_Connection
parent_version_no = 19.0
=
step_text =

step_label = Power - Execute Query 04 step_id =
756 global_flag = 0

sequence_no = 14 step_level = 2 parent_step_id =
737 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %QUERY_DIR%\Power\4.sql
version_no = 8.0

start_directory Power_Static_DB_Connection
parent_version_no = 19.0
=
step_text =

step_label = Power - Execute Query 11 step_id =
757 global_flag = 0

sequence_no = 15 step_level = 2 parent_step_id =
737 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name = %QUERY_DIR%\Power\11.sql
version_no = 8.0

start_directory Power_Static_DB_Connection
parent_version_no = 19.0
=
step_text =

```
step_label = Power - Execute Query 15                    step_id =
758     global_flag =    0

sequence_no =       16  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\15.sql
             version_no =  8.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =



step_label = Power - Execute Query 01                    step_id =
759     global_flag =    0

sequence_no =       17  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\1.sql
             version_no =  7.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =



step_label = Power - Execute Query 10                    step_id =
760     global_flag =    0

sequence_no =       18  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\10.sql
             version_no =  8.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =
```

```
step_label = Power - Execute Query 19                    step_id =
761     global_flag =    0

sequence_no =       19  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\19.sql
             version_no =  8.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =



step_label = Power - Execute Query 05                    step_id =
762     global_flag =    0

sequence_no =       20  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\5.sql
             version_no =  8.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =



step_label = Power - Execute Query 07                    step_id =
763     global_flag =    0

sequence_no =       21  step_level =      2       parent_step_id =
737 enabled_flag =     -1

iterator_name =                              degree_parallelism  1

execution_mechanism =     1                  continuation_criteria = 2
      failure_details =

step_file_name =    %QUERY_DIR%\Power\7.sql
             version_no =  8.0

start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
step_text =
```

```
 step_label = Power - Execute Query 12                    step_id =
764      global_flag =    0

 sequence_no =       22  step_level =      2       parent_step_id =
737 enabled_flag =     -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =    %QUERY_DIR%\Power\12.sql
              version_no =  7.0

 start_directory     Power_Static_DB_Connection
  parent_version_no =     19.0
=
 step_text =
```



```
 step_label = Power - Execute RF2                         step_id =
765      global_flag =    0

 sequence_no =       1  step_level =      2       parent_step_id =
738 enabled_flag =     -1

 iterator_name =     DELETE_SEGMENT
degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  3.0

 start_directory     Power_Dynamic_DB_Connection
       parent_version_no =     12.0
=
 step_text =  --
           -- Execute the Refresh RF2 Stored Procedure
           --

           EXEC RF2_%DELETE_SEGMENT% %BATCH_SIZE%




 step_label = Throughput - Semaphore Loop for RF Delay
step_id =     766      global_flag =    0

 sequence_no =       1  step_level =      2       parent_step_id =
740 enabled_flag =     -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     2                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  3.0

 start_directory     %TOOLS_DIR%
parent_version_no =     0.5
 =
 step_text =  %TOOLS_DIR%\Utility\semaphore -waitgroup S -count
%MAX_STREAMS%




 step_label = Throughput - Refresh Streams                step_id =
 767      global_flag =    0

 sequence_no =       2  step_level =      2       parent_step_id =
740 enabled_flag =     -1

 iterator_name =     STREAM_NUM
degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.2

 start_directory
parent_version_no =     0.5
 =
 step_text =
```



```
 step_label = Stream 1 Manager                            step_id =
769      global_flag =    0

 sequence_no =       1  step_level =      2       parent_step_id =
741 enabled_flag =     -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory
parent_version_no =     3.0
 =
 step_text =




 step_label = Stream 2 Manager                            step_id =
770      global_flag =    0

 sequence_no =       2  step_level =      2       parent_step_id =
741 enabled_flag =     -1
```

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 3.0
=
step_text =

step_label = Stream 3 Manager step_id =
771 global_flag = 0

sequence_no = 3 step_level = 2 parent_step_id =
741 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 3.0
=
step_text =

step_label = Stream 4 Manager step_id =
772 global_flag = 0

sequence_no = 4 step_level = 2 parent_step_id =
741 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 3.0
=
step_text =

step_label = Stream 5 Manager step_id =
773 global_flag = 0

sequence_no = 5 step_level = 2 parent_step_id =
741 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 3.0
=
step_text =

step_label = Stream 6 Manager step_id =
774 global_flag = 0

sequence_no = 6 step_level = 2 parent_step_id =
741 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 3.0
=
step_text =

step_label = Stream 7 Manager step_id =
775 global_flag = 0

sequence_no = 7 step_level = 2 parent_step_id =
741 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 3.0
=
step_text =

step_label = Stream 8 Manager step_id =
776 global_flag = 0

sequence_no = 8 step_level = 2 parent_step_id =
741 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 3.0
=
step_text =

Friday, March 09, 2001
Page 119 of 130

step_label = Stream 9 Manager step_id =
777 global_flag = 0

sequence_no = 9 step_level = 2 parent_step_id =
741 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 3.0
=
step_text =

step_label = Stream 10 Manager step_id =
778 global_flag = 0

sequence_no = 10 step_level = 2 parent_step_id =
741 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 3.0
=
step_text =

step_label = Throughput - Query Stream 1 step_id =
779 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
769 enabled_flag = -1

iterator_name = QUERY degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
%QUERY_DIR%\STREAM1\Stream1Q%QUERY%.sql
version_no = 0.0

start_directory Throughput_Static_Stream_1_Connection
parent_version_no = 0.0
=
step_text =

step_label = Throughput - Post to Semaphore (S1) step_id
= 780 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
769 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.0
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal S.1

Friday, March 09, 2001
Page 120 of 130

step_label = Throughput - Query Stream 2 step_id =
781 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
770 enabled_flag = -1

iterator_name = QUERY degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
%QUERY_DIR%\STREAM2\Stream2Q%QUERY%.sql
version_no = 0.0

start_directory Throughput_Static_Stream_2_Connection
parent_version_no = 0.0
=
step_text =

step_label = Throughput - Post to Semaphore (S2) step_id
= 782 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
770 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.0
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal S.2

step_label = Throughput - Query Stream 3 step_id =
783 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
771 enabled_flag = -1

iterator_name = QUERY degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
%QUERY_DIR%\STREAM3\Stream3Q%QUERY%.sql
version_no = 0.0

start_directory Throughput_Static_Stream_3_Connection
parent_version_no = 0.0
=
step_text =

step_label = Throughput - Post to Semaphore (S3) step_id
= 784 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
771 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.0
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal S.3

step_label = Throughput - Query Stream 4 step_id =
785 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id =
772 enabled_flag = -1

iterator_name = QUERY degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2
failure_details =

step_file_name =
%QUERY_DIR%\STREAM4\Stream4Q%QUERY%.sql
version_no = 0.0

start_directory Throughput_Static_Stream_4_Connection
parent_version_no = 0.0
=
step_text =

step_label = Throughput - Post to Semaphore (S4) step_id
= 786 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id =
772 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory %TOOLS_DIR%
parent_version_no = 0.0
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal S.4

step_label = Throughput - Query Stream 5 step_id =
787 global_flag = 0

```
 sequence_no =        1  step_level =       3        parent_step_id =
773 enabled_flag =      -1

 iterator_name =     QUERY                          degree_parallelism  1

 execution_mechanism =     1                     continuation_criteria = 2
       failure_details =

 step_file_name =
%QUERY_DIR%\STREAM5\Stream5Q%QUERY%.sql
         version_no =  0.0

 start_directory     Throughput_Static_Stream_5_Connection
         parent_version_no =     0.0
 =
 step_text =



 step_label = Throughput - Post to Semaphore (S5)                step_id
=     788      global_flag =    0

 sequence_no =        2  step_level =       3        parent_step_id =
773 enabled_flag =      -1

 iterator_name =                                  degree_parallelism  1

 execution_mechanism =     2                     continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %TOOLS_DIR%
parent_version_no =     0.0
 =
 step_text =  %TOOLS_DIR%\Utility\semaphore -signal S.5




 step_label = Throughput - Query Stream 6                        step_id =
 789      global_flag =    0

 sequence_no =        1  step_level =       3        parent_step_id =
774 enabled_flag =      -1

 iterator_name =     QUERY                          degree_parallelism  1

 execution_mechanism =     1                     continuation_criteria = 2
       failure_details =

 step_file_name =
%QUERY_DIR%\STREAM6\Stream6Q%QUERY%.sql
         version_no =  0.0

 start_directory     Throughput_Static_Stream_6_Connection
         parent_version_no =     0.0
 =
 step_text =



Friday, March 09, 2001
   Page 123 of 130
```

```
 step_label = Throughput - Post to Semaphore (S6)                step_id
=     790      global_flag =    0

 sequence_no =        2  step_level =       3        parent_step_id =
774 enabled_flag =      -1

 iterator_name =                                  degree_parallelism  1

 execution_mechanism =     2                     continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %TOOLS_DIR%
parent_version_no =     0.0
 =
 step_text =  %TOOLS_DIR%\Utility\semaphore -signal S.6




 step_label = Throughput - Query Stream 7                        step_id =
 791      global_flag =    0

 sequence_no =        1  step_level =       3        parent_step_id =
775 enabled_flag =      -1

 iterator_name =     QUERY                          degree_parallelism  1

 execution_mechanism =     1                     continuation_criteria = 2
       failure_details =

 step_file_name =
%QUERY_DIR%\STREAM7\Stream7Q%QUERY%.sql
         version_no =  0.0

 start_directory     Throughput_Static_Stream_7_Connection
         parent_version_no =     0.0
 =
 step_text =



 step_label = Throughput - Post to Semaphore (S7)                step_id
=     792      global_flag =    0

 sequence_no =        2  step_level =       3        parent_step_id =
775 enabled_flag =      -1

 iterator_name =                                  degree_parallelism  1

 execution_mechanism =     2                     continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     %TOOLS_DIR%
parent_version_no =     0.0
 =
 step_text =  %TOOLS_DIR%\Utility\semaphore -signal S.7
```

step_label = Throughput - Query Stream 8 step_id = 793 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 776 enabled_flag = -1

iterator_name = QUERY degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = %QUERY_DIR%\STREAM8\Stream8Q%QUERY%.sql version_no = 0.0

start_directory Throughput_Static_Stream_8_Connection parent_version_no = 0.0
=
step_text =

step_label = Throughput - Post to Semaphore (S8) step_id = 794 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id = 776 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2 failure_details =

step_file_name = version_no = 0.0

start_directory %TOOLS_DIR% parent_version_no = 0.0
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal S.8

step_label = Throughput - Query Stream 9 step_id = 795 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 777 enabled_flag = -1

iterator_name = QUERY degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = %QUERY_DIR%\STREAM9\Stream9Q%QUERY%.sql version_no = 0.0

start_directory Throughput_Static_Stream_9_Connection parent_version_no = 0.0
=
step_text =

step_label = Throughput - Post to Semaphore (S9) step_id = 796 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id = 777 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2 failure_details =

step_file_name = version_no = 0.0

start_directory %TOOLS_DIR% parent_version_no = 0.0
=
step_text = %TOOLS_DIR%\Utility\semaphore -signal S.9

step_label = Throughput - Query Stream 10 step_id = 797 global_flag = 0

sequence_no = 1 step_level = 3 parent_step_id = 778 enabled_flag = -1

iterator_name = QUERY degree_parallelism 1

execution_mechanism = 1 continuation_criteria = 2 failure_details =

step_file_name = %QUERY_DIR%\STREAM10\Stream10Q%QUERY%.sql version_no = 0.0

start_directory Throughput_Static_Stream_10_Connection parent_version_no = 0.0
=
step_text =

step_label = Throughput - Post to Semaphore (S10) step_id = 798 global_flag = 0

sequence_no = 2 step_level = 3 parent_step_id = 778 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2 failure_details =

```
 step_file_name =
version_no =  0.0

 start_directory     %TOOLS_DIR%
parent_version_no =     0.0
 =
 step_text =  %TOOLS_DIR%\Utility\semaphore -signal S.10
```



```
 step_label = Throughput - RF1 - Stream%STREAM_NUM%
  step_id =     799      global_flag =    0

 sequence_no =        1  step_level =       3        parent_step_id =
767 enabled_flag =      -1

 iterator_name =                                degree_parallelism
%INSERT_PARALLELISM%

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory
parent_version_no =     0.2
 =
 step_text =



 step_label = Throughput - RF2 - Stream%STREAM_NUM%
  step_id =     800      global_flag =    0

 sequence_no =        2  step_level =       3        parent_step_id =
767 enabled_flag =      -1

 iterator_name =                                degree_parallelism
%DELETE_PARALLELISM%

 execution_mechanism =     0                    continuation_criteria = 0
       failure_details =

 step_file_name =
version_no =  1.0

 start_directory
parent_version_no =     0.2
 =
 step_text =



 step_label = Throughput - Stream - Increment Update Set
step_id =     801      global_flag =    0

 sequence_no =        3  step_level =       3        parent_step_id =
767 enabled_flag =      -1

 iterator_name =                                degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  1.0

 start_directory     Throughput_RF_Connection
  parent_version_no =     0.2
 =
 step_text =  UPDATE TPCH_AUX_TABLE SET
updateset=updateset+1
```



```
 step_label = Throughput - Execute Stream%STREAM_NUM% RF1
      step_id =     805      global_flag =    0

 sequence_no =        1  step_level =       4        parent_step_id =
799 enabled_flag =      -1

 iterator_name =     INSERT_SEGMENT
degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     Throughput_RF_Connection
  parent_version_no =     0.0
 =
 step_text =  --
              --  Execute the Refresh RF1 Stored Procedure
              --
              EXEC  RF1_%INSERT_SEGMENT% %BATCH_SIZE%



 step_label = Throughput - Execute Stream%STREAM_NUM% RF2
      step_id =     806      global_flag =    0

 sequence_no =        1  step_level =       4        parent_step_id =
800 enabled_flag =      -1

 iterator_name =     DELETE_SEGMENT
degree_parallelism  1

 execution_mechanism =     1                    continuation_criteria = 2
       failure_details =

 step_file_name =
version_no =  0.0

 start_directory     Throughput_RF_Connection
  parent_version_no =     1.0
 =
 step_text =  --
              --  Execute the Refresh RF2 Stored Procedure
              --
              EXEC RF2_%DELETE_SEGMENT% %BATCH_SIZE%
```

step_label = Shutdown SQL Server step_id = 954 global_flag = 0

sequence_no = 4 step_level = 0 parent_step_id = 0 enabled_flag = 0

iterator_name = degree_parallelism 1

execution_mechanism = 0 continuation_criteria = 0
failure_details =

step_file_name =
version_no = 7.0

start_directory
parent_version_no = 0.0
=
step_text =

step_label = Shutdown SQL Server at end of run step_id = 955 global_flag = 0

sequence_no = 1 step_level = 1 parent_step_id = 954 enabled_flag = -1

iterator_name = degree_parallelism 1

execution_mechanism = 2 continuation_criteria = 2
failure_details =

step_file_name =
version_no = 0.0

start_directory
parent_version_no = 7.0
=
step_text = isql -Usa -P -t10 -Q"shutdown"

Iterator_values

workspace_id	step_id	version_n	type	iterator_value	sequence_no	o
9	783	0.0	1	1	0	
9	789	0.0	1	1	0	
9	789	0.0	3	1	0	
9	789	0.0	2	22	0	
9	787	0.0	2	22	0	
9	787	0.0	3	1	0	
9	787	0.0	1	1	0	
9	785	0.0	3	1	0	
9	785	0.0	2	22	0	
9	742	8.0	2	%INSERT_SEGMENTS_PER_UPDATE_SET%		0
9	783	0.0	3	1	0	
9	791	0.0	1	1	0	
9	783	0.0	2	22	0	
9	781	0.0	3	1	0	
9	781	0.0	2	22	0	

9	781	0.0	1	1	0	
9	779	0.0	1	1	0	
9	779	0.0	2	22	0	
9	779	0.0	3	1	0	
9	742	8.0	1	1	0	
9	742	8.0	3	1	0	
9	785	0.0	1	1	0	
9	797	0.0	1	1	0	

```
        9     765   3.0           2
%DELETE_SEGMENTS_PER_UPDATE_SET%                0

        9     765   3.0           3 1
    0

        9     767   0.2           2 %MAX_STREAMS%
              0

        9     767   0.2           3 1
    0

        9     767   0.2           1 1
    0

        9     806   0.0           3 1
    0


        9     806   0.0           1 1
    0

        9     806   0.0           2
%DELETE_SEGMENTS_PER_UPDATE_SET%                0

        9     805   0.0           3 1
    0

        9     791   0.0           2 22
     0

        9     805   0.0           2
%INSERT_SEGMENTS_PER_UPDATE_SET%                0


        9     791   0.0           3 1
    0

        9     797   0.0           3 1
    0

        9     797   0.0           2 22
     0

        9     795   0.0           1 1
    0

        9     795   0.0           2 22
     0

        9     795   0.0           3 1
    0

        9     793   0.0           1 1
    0


        9     793   0.0           2 22
     0

        9     793   0.0           3 1
    0

        9     765   3.0           1 1
    0

        9     805   0.0           1 1
    0
```

Appendix G: Implementation Specific Layer and Source Code

```
/*********************************************************
  DllData file -- generated by MIDL compiler

      DO NOT ALTER THIS FILE

  This file is regenerated by MIDL on every IDL file compile.

  To completely reconstruct this file, delete it and rerun MIDL
  on all the IDL files in this DLL, specifying this file for the
  /dlldata command line option

*********************************************************/

#define PROXY_DELEGATION

#include <rpcproxy.h>

#ifdef __cplusplus
extern "C"   {
#endif

EXTERN_PROXY_FILE( ExecuteDll )


PROXYFILE_LIST_START
/* Start of list */
  REFERENCE_PROXY_FILE( ExecuteDll ),
/* End of list */
PROXYFILE_LIST_END


DLLDATA_ROUTINES( aProxyFileList, GET_DLL_CLSID )

#ifdef __cplusplus
}  /*extern "C" */
#endif

/* end of generated dlldata file */


// Execute.cpp : Implementation of CExecute
#include "stdafx.h"

#include "ExecuteDll.h"
#include "SMExecute.h"
#include "Execute.h"

extern SQLHENV henv;

extern SM_Connection_Info              *p_Connections;
                // Pointer to open connections
extern int
iConnectionCount;                                       // Number of open
connections
extern CRITICAL_SECTION                         hConnections;
                        // Critical section to serialize access to available
connections

#ifdef _TPCH_AUDIT
                extern FILE *pfLogFile;
                                                        // Log file containing
timestamps
                extern CRITICAL_SECTION hLogFileWrite;
                        // Handle to critical section
#endif

////////////////////////////////////////////////////////////////////////////
// CExecute

char * CExecute::m_szOdbcOps[] = {
                                "SQLAllocHandle",
                                "SQLDriverConnect",
                                "SQLExecDirect",
                                "SQLSetStmtAttr",
                                "SQLCancel",
                                "SQLNumResultCols",
                                "SQLDescribeCol",
                                "SQLColAttribute",
                                "SQLFetch",
                                "SQLGetData",
                                "SQLRowCount",
                                "SQLMoreResults"
                };

STDMETHODIMP CExecute::InterfaceSupportsErrorInfo(REFIIDriid)
{
                static const IID* arr[] =
                {
                                &IID_IExecute
                };
                for (int i=0; i < sizeof(arr) / sizeof(arr[0]); i++)
                {
                                if (InlineIsEqualGUID(*arr[i],riid))
                                                return S_OK;
                }
                return S_FALSE;
}

STDMETHODIMP CExecute::put_OutputFile(BSTR newVal)
{
                assert(m_pOutputFile);
                m_OutputFile = newVal;

                HRESULT   hr = m_pOutputFile->put_FileName(newVal);
                if FAILED(hr)
                {
                                m_pOutputFile->Release();
                                m_pOutputFile = NULL;
                }
                return hr;
}

//DEL STDMETHODIMP CExecute::put_LogFile(BSTR newVal)
//DEL {
//DEL           assert(m_pLogFile);
//DEL
//DEL           m_pLogFile->put_FileName(newVal);
//DEL           return S_OK;
//DEL }

STDMETHODIMP CExecute::put_ErrorFile(BSTR newVal)
{
                assert(m_pErrorFile);
                m_ErrorFile = newVal;

                HRESULT   hr = m_pErrorFile->put_FileName(newVal);
                if FAILED(hr)
                {
                                m_pErrorFile->Release();
                                m_pErrorFile = NULL;
                }
                return hr;
}

STDMETHODIMP CExecute::DoExecute(BSTR szCommand, BSTR
szExecutionDtls, ExecutionType ExecMethod, \

                BOOL bNoCount, BOOL bNoExecute, BOOL bParseOnly, BOOL
bQuotedIds, \

                BOOL bAnsiNulls, BOOL bShowQP, BOOL bStatsTime, BOOL
bStatsIO, \

                long lRowCount, long lQueryTmout, BSTR szConnection)
{
  HANDLE                hThrd;
  DWORD                 tid;

  _CrtSetReportFile(_CRT_WARN, _CRTDBG_FILE_STDOUT);
```

```
		m_szCommand = szCommand;
		m_szExecDtls = szExecutionDtls;

		m_ExecMthd = ExecMethod;
		if (m_ExecMthd == execODBC)
	{
	    m_bNoCount = bNoCount;
			m_bNoExecute = bNoExecute;
			m_bParseOnly = bParseOnly;
			m_bQuotedIds = bQuotedIds;
			m_bAnsiNulls = bAnsiNulls;
			m_bShowQP = bShowQP;
			m_bStatsTime = bStatsTime;
			m_bStatsIO = bStatsIO;
			m_lRowCount = lRowCount;
			m_lQueryTmout = lQueryTmout;
			m_szConnection = szConnection;
	}

		if((hThrd = CreateThread( 0, 0,
(LPTHREAD_START_ROUTINE)ExecutionThread,
				this, 0, &tid)) == NULL)
			return(RaiseSystemError());

		CloseHandle(hThrd);

		return S_OK;
}

STDMETHODIMP CExecute::Abort()
{
		if (m_ExecMthd == execShell)
			return(AbortShell());
		else
			return(AbortODBC());
}

void ExecutionThread(LPVOID lpParameter)
{

		CExecute	*MyExecute = (CExecute*)lpParameter;

		MyExecute->m_tElapsedTime = 0;

		GetLocalTime(&MyExecute->m_tStartTime);
		MyExecute->PostMessage(WM_TASK_START, 0, 0);

#ifdef _TPCH_AUDIT
		char			szBuffer[MAXLOGCMDBUF];
		char			szFmt[MAXBUFLEN];

		sprintf(szFmt, "Start Step: '%%.%ds' at '%d/%d/%d
%d:%d:%d:%d'\n",
			MAXLOGCMDLEN,
			MyExecute->m_tStartTime.wMonth,
MyExecute->m_tStartTime.wDay,
			MyExecute->m_tStartTime.wYear,
MyExecute->m_tStartTime.wHour,
			MyExecute->m_tStartTime.wMinute,
MyExecute->m_tStartTime.wSecond,
			MyExecute->m_tStartTime.wMilliseconds);
		if (MyExecute->m_ExecMthd == execShell)

WriteFileToTpchLog((LPSTR)MyExecute->m_szCommand,szFmt);
		else
		{
			sprintf(szBuffer, szFmt,
(LPSTR)MyExecute->m_szCommand);
			WriteToTpchLog(szBuffer);
		}
#endif

		// Initialize the run status for the step to running. The completion
status for
		// the step will be initialized by the Shell and ODBC execution
functions.
	MyExecute->m_StepStatus = gintRunning;
		if (MyExecute->m_ExecMthd == execShell)
			MyExecute->m_tElapsedTime =
MyExecute->ExecuteShell(MyExecute);
		else
			MyExecute->m_tElapsedTime =
MyExecute->ExecuteODBC(MyExecute);

		// Close the output, log and error files
		if (MyExecute->m_pOutputFile)
			MyExecute->m_pOutputFile->Release();
		MyExecute->m_pOutputFile = NULL;

		MyExecute->m_ExecTime = NULL;

		GetLocalTime(&MyExecute->m_tEndTime);

#ifdef _TPCH_AUDIT
		sprintf(szFmt, "Complete Step: '%%.%ds' at '%d/%d/%d
%d:%d:%d:%d'\n",
			MAXLOGCMDLEN,
			MyExecute->m_tEndTime.wMonth,
MyExecute->m_tEndTime.wDay,
			MyExecute->m_tEndTime.wYear,
MyExecute->m_tEndTime.wHour,
			MyExecute->m_tEndTime.wMinute,
MyExecute->m_tEndTime.wSecond,
			MyExecute->m_tEndTime.wMilliseconds);
		if (MyExecute->m_ExecMthd == execShell)

WriteFileToTpchLog((LPSTR)MyExecute->m_szCommand,szFmt);
		else
		{
			sprintf(szBuffer, szFmt,
(LPSTR)MyExecute->m_szCommand);
			WriteToTpchLog(szBuffer);
		}
#endif

		MyExecute->PostMessage(WM_TASK_FINISH, 0, 0);

		return;
}

#ifdef _TPCH_AUDIT

void WriteFileToTpchLog(LPSTR szFile, LPSTR szFmt)
{
		// Reads a maximum of MAXLOGCMDBUF characters from the
command file and writes it to the log
		FILE	*fpCmd;
		int			iRead;
		char	szBuf[MAXLOGCMDBUF];
		char	szCmd[MAXLOGCMDLEN];

		if ( pfLogFile != NULL )
		{
			if ( (fpCmd = fopen(szFile, FILE_ACCESS_READ)) !=
NULL)
			{
				iRead = fread(szCmd, sizeof(char),
sizeof(szCmd) / sizeof(char), fpCmd);
				if (iRead < MAXLOGCMDLEN)
					szCmd[iRead] = '\0';
				else
					szCmd[MAXLOGCMDLEN
- 1] = '\0';
				sprintf(szBuf, szFmt, szCmd);
				WriteToTpchLog(szBuf);
				fclose(fpCmd);
			}
		}
}

void WriteToTpchLog(char *szMsg)
{
		if (pfLogFile != NULL)
		{
			EnterCriticalSection(&hLogFileWrite);
```

```
                        fprintf(pfLogFile,szMsg);
                        LeaveCriticalSection(&hLogFileWrite);
                }

                return;
}
#endif

TC_TIME CExecute::ExecuteShell(CExecute*p)
{
                STARTUPINFOA                                    Start;
                PROCESS_INFORMATION             proc;
                DWORD
exitCode;
                TC_TIME
tElapsed = 0;
                _bstr_t
szCommand("cmd/c ");
                LPSTR
szStartDir;
                CURRENCY
Elapsed;

                szCommand += p->m_szCommand;

                // Redirect output and error information
                szCommand += " > " + m_OutputFile + " 2> " + m_ErrorFile;

                // Initialize the STARTUPINFO structure:
                memset(&Start, 0, sizeof(STARTUPINFOA));
                Start.cb                = sizeof(Start);
                Start.dwFlags           = STARTF_USESHOWWINDOW;
                Start.wShowWindow       = SW_SHOWMINNOACTIVE;

                memset(&proc, 0, sizeof(PROCESS_INFORMATION));

                szStartDir = strcmp((LPCTSTR)m_szExecDtls, "") == 0 ? NULL :
(LPSTR)m_szExecDtls;

                p->m_ExecTime->Start();

                // Start the shelled application:
                if (!CreateProcessA( NULL, (LPSTR)szCommand, NULL, NULL,
FALSE,
                        NORMAL_PRIORITY_CLASS, NULL, szStartDir,
&Start, &proc ))
                {
                        m_StepStatus = gintFailed;
                        LogSystemError(p->m_pErrorFile);

                        p->m_ExecTime->Stop(&Elapsed);
                        return((TC_TIME)Elapsed.int64);
                }

                m_hHandle = proc.hProcess;
                // Give the process time to execute and finish
                WaitForSingleObject(m_hHandle,INFINITE);
                p->m_ExecTime->Stop(&Elapsed);

                if (!GetExitCodeProcess(m_hHandle,&exitCode))
                {
                        m_StepStatus = gintFailed;
                        LogSystemError(p->m_pErrorFile);
                }
                else
                        m_StepStatus = gintComplete;

                // Close all open handles to the shelled process
                CloseHandle(m_hHandle);

                return((TC_TIME)Elapsed.int64);
}

STDMETHODIMP CExecute::AbortShell()
{
                if (m_hHandle != SQL_NULL_HSTMT)
                        if (!TerminateProcess(m_hHandle,0))
                                return(RaiseSystemError());
                return(S_OK);
}

TC_TIME CExecute::ExecuteODBC(CExecute*p)
{
                TC_TIME                                         tElapsed = 0;
                HDBC                                            m_hdbc;
                SQLRETURN                                       rc;
                LPSTR                                           szCmd;
                CURRENCY                                        Elapsed;
                BOOL                                            bDoConnect =
FALSE;

                // ODBC specific initialization
                m_hdbc = SQL_NULL_HDBC;

                // Allocate a new connection if we are creating a dynamic connection
or if
                // the named connection doesn't exist
                if (!InitializeConnection(&m_hdbc,&bDoConnect))
                        return(tElapsed);

                if (bDoConnect)
                {
                        // Allocate connection handle, open a connection and
set connection attributes.
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLDriverConnect.\n");
#endif

                        if (m_bAbort)
                                return(tElapsed);

                        // Connect to the server using the passed in connection
string
                        rc = SQLDriverConnect(m_hdbc, NULL,
                                (unsigned char
*)(LPSTR)p->m_szExecDtls, SQL_NTS,
                                NULL, 0, NULL,
SQL_DRIVER_NOPROMPT);
                        if (rc != SQL_SUCCESS)
                        {
                                if (!HandleODBCError(rc,
SQL_HANDLE_DBC, m_hdbc, &m_hdbc, SMSQLDriverConnect))
                                        return(tElapsed);
                        }
                }

#ifdef _DEBUG
                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLAllocHandle for hdbc.\n");
#endif

                if (!m_bAbort && (rc = SQLAllocHandle(SQL_HANDLE_STMT,
m_hdbc, &m_hHandle)) != SQL_SUCCESS)
                {
                        if (!HandleODBCError(rc, SQL_HANDLE_DBC,
m_hdbc, &m_hdbc, SMSQLAllocHandle))
                                return(tElapsed);
                }

                // Set connection attributes if any have been modified from the
default values
                if (m_lRowCount > 0)
                {
                        char
szConnOptions[512];

                        sprintf(szConnOptions, "SET ROWCOUNT %d ",
m_lRowCount);
                        if (!SetConnectionOption(szConnOptions,&m_hdbc))
                                return(tElapsed);
                }

                if (m_bQuotedIds)
                {
```

```
                        if (!SetConnectionOption("SET
QUOTED_IDENTIFIER ON ", &m_hdbc))
                                return(tElapsed);
        }

        if (!m_bAnsiNulls)
        {
                        if (!SetConnectionOption("SET
ANSI_NULL_DFLT_OFF ON ", &m_hdbc))
                                return(tElapsed);
        }
        if (!m_bAbort && m_lQueryTmout > 0)
        {
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLSetStmtAttr.\n");
#endif

                        // Set the query timeout on the statement handle
                        rc = SQLSetStmtAttr(m_hHandle,
SQL_ATTR_QUERY_TIMEOUT, &m_lQueryTmout,
                                SQL_IS_UINTEGER);

                        if (!HandleODBCError(rc, SQL_HANDLE_STMT,
m_hHandle, &m_hdbc, SMSQLSetStmtAttr))
                                return(tElapsed);
        }

        if (m_bNoExecute)
        {
                        if (!SetConnectionOption("SETNOEXEC ON ",
&m_hdbc))
                                return(tElapsed);
        }
        else if (m_bParseOnly)
        {
                        if (!SetConnectionOption("SETPARSEONLY ON ",
&m_hdbc))
                                return(tElapsed);
        }
        else if (m_bShowQP)
        {
                        // Important to ensure that this is the last connection
attributes being set -
                        // otherwise showplans are generated for all remaining
SET statements
                        if (!SetConnectionOption("SETSHOWPLAN_TEXT
ON ", &m_hdbc))
                                return(tElapsed);
        }
        else
        {
                        if (m_bNoCount)
                        {
                                if (!SetConnectionOption("SET
NOCOUNT ON ", &m_hdbc))
                                        return(tElapsed);
                        }

                        if (m_bStatsIO)
                        {
                                if (!SetConnectionOption("SET
STATISTICS IO ON ", &m_hdbc))
                                        return(tElapsed);
                        }

                        // Important to ensure that this is the last connection
attributes being set -
                        // otherwise timing statistics are generated for all
remaining SET statements
                        if (m_bStatsTime)
                        {
                                if (!SetConnectionOption("SET
STATISTICS TIME ON ", &m_hdbc))
                                        return(tElapsed);
                        }
        }

        m_szCmd = (LPSTR)p->m_szCommand;
        p->m_ExecTime->Start();

        while ((szCmd = NextCmdInBatch((LPSTR)p->m_szCommand))!=
NULL && !m_bAbort)
        {
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLExecDirect.\n");
#endif

                        // Execute the ODBC command
                        rc = SQLExecDirect(m_hHandle,(unsigned char
*)szCmd, SQL_NTS);
                        if (!HandleODBCError(rc, SQL_HANDLE_STMT,
m_hHandle, &m_hdbc, SMSQLExecDirect))
                                return(tElapsed);

                        free(szCmd);

                        // Call a procedure to log the results to the output file
                        ProcessResultsets();
        }
        p->m_ExecTime->Stop(&Elapsed);

        ResetConnectionProperties(&m_hdbc);

        ODBCCleanup(&m_hdbc, &m_hHandle);

        if (m_StepStatus != gintFailed)
                m_StepStatus = gintComplete;

        return((DWORD)Elapsed.int64);
}

BOOL CExecute::InitializeConnection(HDBC*phdbc, BOOL *pbDoConnect)
{
        SQLRETURN                                       rc;

        *pbDoConnect = TRUE;

        if (IsDynamicConnection())
        {
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLAllocHandle for m_hdbc.\n");
#endif

                        if (!m_bAbort && (rc =
SQLAllocHandle(SQL_HANDLE_DBC,henv, phdbc)) != SQL_SUCCESS)
                        {
                                if (!HandleODBCError(rc,
SQL_HANDLE_ENV, henv, phdbc, SMSQLAllocHandle))
                                        return FALSE;
                        }
                        return TRUE;
        }

        EnterCriticalSection(&hConnections);
        // Returns the connection handle if the connection,m_szConnection,
exists
        for (m_iConnectionIndex = iConnectionCount - 1;
m_iConnectionIndex >= 0; m_iConnectionIndex--)
        {
                        if (!strcmp( (p_Connections +
m_iConnectionIndex)->szConnectionName,(LPSTR)m_szConnection))
                        {
                                if (!(p_Connections +
m_iConnectionIndex)->bInUse)
                                {
                                        *phdbc = (p_Connections +
m_iConnectionIndex)->hdbc;
                                        (p_Connections +
m_iConnectionIndex)->bInUse= TRUE;

                                        *pbDoConnect = FALSE;
                                        break;
                                }
```

```
                                        else
                                        {

LeaveCriticalSection(&hConnections);

                                                m_StepStatus = gintFailed;
                                                _bstr_t
temp(SM_ERR_CONN_IN_USE);
                                                if (m_pErrorFile)

m_pErrorFile->WriteLine((BSTR)temp);

                                                return FALSE;
                                        }
                                }
                        }

                        if (m_iConnectionIndex < 0)
                        {
                                // Connection was not found. Allocate connection
handle and add it to list of
                                // available connections.
#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLAllocHandle for m_hdbc.\n");
#endif

                                if (!m_bAbort && (rc =
SQLAllocHandle(SQL_HANDLE_DBC, henv, phdbc)) != SQL_SUCCESS)
                                {
                                        if (!HandleODBCError(rc,
SQL_HANDLE_ENV, henv, phdbc, SMSQLAllocHandle))
                                                return FALSE;
                                }

                                m_iConnectionIndex = iConnectionCount++;

                                p_Connections = (SM_Connection_Info
*)realloc(p_Connections, iConnectionCount * sizeof(SM_Connection_Info));

                                strcpy((p_Connections +
m_iConnectionIndex)->szConnectionName,(LPSTR)m_szConnection);
                                (p_Connections + m_iConnectionIndex)->hdbc =
*phdbc;
                                (p_Connections + m_iConnectionIndex)->bInUse =
TRUE;
                        }

                        LeaveCriticalSection(&hConnections);

                        return TRUE;
}

void CExecute::ResetConnectionUsage()
{
                        if(m_iConnectionIndex >= 0 && m_iConnectionIndex <
iConnectionCount)
                        {
                                EnterCriticalSection(&hConnections);
                                (p_Connections + m_iConnectionIndex)->bInUse =
FALSE;
                                LeaveCriticalSection(&hConnections);
                        }

                        return;
}

BOOL CExecute::ResetConnectionProperties(HDBC *p_hdbc)
{
                        SQLRETURN                               rc;

                        // Reset connection attributes if any have been modified from the
default values

                        if (m_bNoExecute)
                        {
                                if (!SetConnectionOption("SET NOEXEC OFF ",
p_hdbc))
                                        return FALSE;
                        }
                        else if (m_bParseOnly)
                        {
                                if (!SetConnectionOption("SET PARSEONLY OFF ",
p_hdbc))
                                        return FALSE;
                        }
                        else if (m_bShowQP)
                        {
                                // Reset connection attributes in reverse order
                                if (!SetConnectionOption("SET SHOWPLAN_TEXT
OFF ", p_hdbc))
                                        return FALSE;
                        }
                        else
                        {
                                // Reset connection attributes in reverse order
                                if (m_bStatsTime)
                                {
                                        if (!SetConnectionOption("SET
STATISTICS TIME OFF ", p_hdbc))
                                                return FALSE;
                                }

                                if (m_bNoCount)
                                {
                                        if (!SetConnectionOption("SET
NOCOUNT OFF ", p_hdbc))
                                                return FALSE;
                                }

                                if (m_bStatsIO)
                                {
                                        if (!SetConnectionOption("SET
STATISTICS IO OFF ", p_hdbc))
                                                return FALSE;
                                }
                        }

                        if (m_lRowCount > 0)
                        {
                                char
szConnOptions[512];

                                sprintf(szConnOptions, "SET ROWCOUNT 0 ");
                                if (!SetConnectionOption(szConnOptions, p_hdbc))
                                        return FALSE;
                        }

                        if (m_bQuotedIds)
                        {
                                if (!SetConnectionOption("SET
QUOTED_IDENTIFIER OFF ", p_hdbc))
                                        return FALSE;
                        }

                        if (!m_bAnsiNulls)
                        {
                                if (!SetConnectionOption("SET
ANSI_NULL_DFLT_OFF OFF ", p_hdbc))
                                        return FALSE;
                        }

                        if (m_lQueryTmout > 0)
                        {
                                SQLUINTEGER                     lQueryTmout =
0;

#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLSetStmtAttr.\n");
#endif

                                // Set the query timeout on the statement handle
                                rc = SQLSetStmtAttr(m_hHandle,
SQL_ATTR_QUERY_TIMEOUT, &lQueryTmout,
                                        SQL_IS_UINTEGER);
```

```
                        if (!HandleODBCError(rc, SQL_HANDLE_STMT,
m_hHandle, p_hdbc, SMSQLSetStmtAttr))
                                return FALSE;
        }

        return TRUE;
}

LPSTR CExecute::NextCmdInBatch(LPSTR szBatch)
{
        LPSTR   szCmd, szSeparator, szStart;
        char    szNext;

        szStart = m_szCmd;

        while ( (szSeparator = strstr(szStart, CMD_SEPARATOR)) !=
NULL)
        {
                szNext = *(szSeparator +
strlen(CMD_SEPARATOR));
                if ( szNext == '\n' || szNext == '\r' || szNext == '\0')
                        break;
                else
                        szStart = szSeparator +
strlen(CMD_SEPARATOR);
        }

        if (!szSeparator)
        {
                // No more GO's
                if (strlen(m_szCmd) > 0)
                {
                        szCmd =
(LPSTR)malloc(strlen(m_szCmd)+ 1);
                        strcpy(szCmd, m_szCmd);
                        m_szCmd += strlen(m_szCmd);
                }
                else
                        szCmd = NULL;
        }
        else if (szSeparator - m_szCmd > 0)
        {
                // Strip the succeeding newline
                szCmd = (LPSTR)malloc(szSeparator - m_szCmd);
                strncpy(szCmd, m_szCmd, szSeparator - m_szCmd -
1);
                *(szCmd + (szSeparator - m_szCmd - 1)) = '\0';
                m_szCmd += szSeparator - m_szCmd +
strlen(CMD_SEPARATOR);
                if ( szNext == '\n' || szNext == '\r')
                        m_szCmd += 1;
        }
        else
                szCmd = NULL;

        return(szCmd);
}

BOOL CExecute::SetConnectionOption(LPSTR szConn, HDBC *pHdbc)
{
        // Executes the passed in connection options 'set' statement. Returns
True if it succeeded
        char                    szConnOptions[512];
        SQLRETURN               rc;

        sprintf(szConnOptions, szConn);

#ifdef _DEBUG
        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLExecDirect for connection option.\n");
#endif

        if (m_bAbort)
                return FALSE;

        rc = SQLExecDirect(m_hHandle, (unsigned char *)szConnOptions,
SQL_NTS);
        if (rc != SQL_SUCCESS)
                LogODBCErrors(rc, SQL_HANDLE_STMT,
m_hHandle, SMSQLExecDirect);

        if (!SQL_SUCCEEDED(rc))
        {
                ODBCCleanup(pHdbc, &m_hHandle);
                return FALSE;
        }

        return TRUE;
}

BOOL CExecute::HandleODBCError(SQLRETURN rc, SWORD fHandleType,
SQLHANDLE handle, HDBC *pHdbc, OdbcOperations OdbcOp)
{
        if (rc != SQL_SUCCESS)
  {
                LogODBCErrors(rc, fHandleType, handle, OdbcOp);
                if (!SQL_SUCCEEDED(rc))
                {
                        ODBCCleanup(pHdbc, &m_hHandle);
                        return FALSE;
                }
  }
        return TRUE;
}

STDMETHODIMP CExecute::AbortODBC()
{
        m_bAbort = TRUE;

        if (m_hHandle != SQL_NULL_HSTMT)
        {
#ifdef _DEBUG
                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLCancel.\n");
#endif

                SQLRETURN rc = SQLCancel(m_hHandle);
                if (rc != SQL_SUCCESS)
                        LogODBCErrors(rc,
SQL_HANDLE_STMT, m_hHandle, SMSQLCancel);
        }

        return(S_OK);
}

void CExecute::ProcessResultsets()
{
  SQLSMALLINT *CTypeArray, *CScaleArray;
  SQLINTEGER  *ColLenArray, *DispLenArray;
        SQLSMALLINT iColNameLen, SQLType, iColNull, i, NumCols =
0;
        SQLINTEGER      iDispLen, iRowCount, LenOrInd;
  SQLRETURN     rc;
        char            szColName[MAX_DATA_LEN + 1];
  void          *DataPtr;


        if (!m_pOutputFile|| m_bAbort)
                return;

        do
        {
#ifdef _DEBUG
                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLNumResultCols.\n");
#endif

                // Determine the number of result set columns.
                rc = SQLNumResultCols(m_hHandle,&NumCols);
                if (rc != SQL_SUCCESS)
                {
                        LogODBCErrors(rc,
SQL_HANDLE_STMT, m_hHandle, SMSQLNumResultCols);
                        if (!SQL_SUCCEEDED(rc))
                                break;
```

```
                }

                if (NumCols > 0)
                {

                        // Allocate arrays to hold the C type, scale,
column and display length of the data
                        CTypeArray = (SQLSMALLINT *)
malloc(NumCols * sizeof(SQLSMALLINT));
                        CScaleArray = (SQLSMALLINT *)
malloc(NumCols * sizeof(SQLSMALLINT));
                        ColLenArray = (SQLINTEGER *)
malloc(NumCols * sizeof(SQLINTEGER));
                        DispLenArray = (SQLINTEGER *)
malloc(NumCols * sizeof(SQLINTEGER));

                        for (i = 0; i < NumCols && !m_bAbort;
i++)
                        {
#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLDescribeCol.\n");
#endif

                                // Get the column description,
include the SQL type
                                rc =
SQLDescribeCol(m_hHandle, ((SQLUSMALLINT) i)+1,
                                        (unsigned char
*)szColName, sizeof(szColName), &iColNameLen,
                                        &SQLType,
(unsigned long *)&ColLenArray[i], &CScaleArray[i], &iColNull);
                                if (rc != SQL_SUCCESS)
                                {

LogODBCErrors(rc, SQL_HANDLE_STMT, m_hHandle, SMSQLDescribeCol);
                                        if
(!SQL_SUCCEEDED(rc))

return;
                                }

#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLColAttribute.\n");
#endif

                                if (m_bAbort)
                                        return;

                                rc =
SQLColAttribute(m_hHandle, ((SQLUSMALLINT) i)+1,
SQL_DESC_DISPLAY_SIZE, NULL, 0, NULL, &iDispLen);
                                if (rc != SQL_SUCCESS)
                                {

LogODBCErrors(rc, SQL_HANDLE_STMT, m_hHandle, SMSQLColAttribute);
                                        if
(!SQL_SUCCEEDED(rc))

return;
                                }

                                // GetDefaultCType contains
a switch statement that returns the default C type
                                // for each SQL type.
                                CTypeArray[i] =
GetDefaultCType(SQLType);
                                if ( (CTypeArray[i] ==
SQL_C_CHAR || CTypeArray[i] == SQL_C_BINARY) && ColLenArray[i] >
MAX_DATA_LEN)
                                {
                                        ColLenArray[i]
= MAX_DATA_LEN;
                                        iDispLen =
MAX_DATA_LEN;
                                }

                                DispLenArray[i] =
max(iColNameLen, iDispLen);
                                DispLenArray[i] =
max(DispLenArray[i], sizeof(S_NULL));

                                // Print the column names in
the header
                                PrintData(szColName,
SQL_C_CHAR, DispLenArray[i], 0, m_pOutputFile);

                                // Add a byte for the
null-termination character
                                ColLenArray[i] += 1;
                                ColLenArray[i] =
ALIGNBUF(ColLenArray[i]);
                        }
                        m_pOutputFile->WriteLine(NULL);

                        // Underline each column name
                        for (i = 0; i < NumCols; i++)
                        {
                                memset(szColName, '-',
DispLenArray[i]);
                                *(szColName +
DispLenArray[i]) = '\0';
                                PrintData(szColName,
SQL_C_CHAR, DispLenArray[i], 0, m_pOutputFile);
                        }
                        m_pOutputFile->WriteLine(NULL);

                        // Retrieve and print each row. PrintData
accepts a pointer to the data, its C type,
                        // and its byte length/indicator.
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0,
NULL, "Executing SQLFetch.\n");
#endif

                        while (!m_bAbort && (rc =
SQLFetch(m_hHandle)) != SQL_NO_DATA)
                        {
                                if (!SQL_SUCCEEDED(rc))
                                {

LogODBCErrors(rc, SQL_HANDLE_STMT, m_hHandle, SMSQLFetch);
                                        break;
                                }

                                for (i = 0; i < NumCols; i++)
                                {
                                        // Allocate the
data buffer.
                                        DataPtr =
malloc(ColLenArray[i]);

#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing SQLGetData.\n");
#endif
                                        while
(!m_bAbort && (rc=SQLGetData(m_hHandle, i + 1, CTypeArray[i],

DataPtr, ColLenArray[i], &LenOrInd)) != SQL_NO_DATA)
                                        {
                                                if
(!SQL_SUCCEEDED(rc))
                                                {

LogODBCErrors(rc, SQL_HANDLE_STMT, m_hHandle, SMSQLGetData);

if (!SQL_SUCCEEDED(rc))

        return;
                                                }
```

```
                                                                        if
(LenOrInd == SQL_NULL_DATA)

PrintData(S_NULL, SQL_C_CHAR, DispLenArray[i], 0, m_pOutputFile);
                                                                        else
                                                                        {

PrintData((SQLCHAR *)DataPtr, CTypeArray[i], DispLenArray[i],

                CScaleArray[i], m_pOutputFile);

// Currently  printing a maximum ofMAX_DATA_LEN chars.

break;
                                                                        }
                                                                }

                                                                free(DataPtr);
                                                        }

m_pOutputFile->WriteLine(NULL);
                                        }
                                        m_pOutputFile->WriteLine(NULL);

                                        free(CTypeArray);
                                        free(CScaleArray);
                                        free(ColLenArray);
                                        free(DispLenArray);
                                }

                                // Write io statistics, if applicable
                                LogODBCErrors(rc, SQL_HANDLE_STMT,
m_hHandle, SMSQLFetch);

#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLRowCount.\n");
#endif

                                if (m_bAbort)
                                        break;

                                // action (insert, update, delete) query
                                rc = SQLRowCount(m_hHandle, &iRowCount);
                                if (rc != SQL_SUCCESS)
                                {
                                        LogODBCErrors(rc,
SQL_HANDLE_STMT, m_hHandle, SMSQLRowCount);
                                        if (!SQL_SUCCEEDED(rc))
                                                break;
                                }

                                if (!m_bNoCount && iRowCount != -1)
                                {
                                        sprintf(szColName, "(%d row(s)
affected)", iRowCount);
                                        _bstr_t temp(szColName);
                                        m_pOutputFile->WriteLine((BSTR)temp);
                                        m_pOutputFile->WriteLine(NULL);
                                }

#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLFreeStmt.\n");
#endif

                                if (m_bAbort)
                                        break;

                                SQLFreeStmt(m_hHandle,SQL_UNBIND);

#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLMoreResults.\n");
#endif

                                if (m_bAbort)
                                        break;

                                // Process the next resultset. This function returns
'success with info' even
                                // if there is no other resultset and there are statistics
messages to be printed.
                                // Hence the check for -1 rows before printing.
                                rc=SQLMoreResults(m_hHandle);
                                if (rc != SQL_SUCCESS)
                                {
                                        LogODBCErrors(rc,
SQL_HANDLE_STMT, m_hHandle, SMSQLMoreResults);

                                        if (!SQL_SUCCEEDED(rc))
                                                break;
                                }

        } while (rc != SQL_NO_DATA);

                return;
}

void CExecute::PrintData(void *vData, SQLSMALLINT CType, SQLINTEGER
IndPtr, SQLSMALLINT iScale, ISMLog *pOutput)
{
                // PrintData accepts a pointer to the data, its C type,
                // and its byte length/indicator. It contains a switch statement that
casts and prints
                // the data according to its type.

                char                    *s;
                char                    fmt[MAXBUFLEN];
                int                             j = 0;
                SQLINTEGER              iColLen = IndPtr + 1;

                assert(iColLen);
                s = (LPSTR)malloc(iColLen+ 1);

                if (s)
                {
                        if (vData)
                        {
                                switch(CType)
                                {
                                case SQL_C_CHAR:
                                case SQL_C_WCHAR:
                                case SQL_C_TYPE_DATE:
                                case SQL_C_TYPE_TIME:
                                case SQL_C_TYPE_TIMESTAMP:
                                case SQL_C_INTERVAL_YEAR:
                                case SQL_C_INTERVAL_MONTH:
                                case
SQL_C_INTERVAL_YEAR_TO_MONTH:
                                case SQL_C_INTERVAL_DAY:
                                case SQL_C_INTERVAL_HOUR:
                                case SQL_C_INTERVAL_MINUTE:
                                case SQL_C_INTERVAL_SECOND:
                                case
SQL_C_INTERVAL_DAY_TO_HOUR:
                                case
SQL_C_INTERVAL_DAY_TO_MINUTE:
                                case
SQL_C_INTERVAL_DAY_TO_SECOND:
                                case
SQL_C_INTERVAL_HOUR_TO_MINUTE:
                                case
SQL_C_INTERVAL_HOUR_TO_SECOND:
                                case
SQL_C_INTERVAL_MINUTE_TO_SECOND:
                                case SQL_C_BINARY:
                                        sprintf(fmt, "%%.%ds",
iColLen);
                                        j = sprintf(s, fmt, (char
*)vData);
                                        break;

                                case SQL_C_SHORT:
                                        j = sprintf(s, "%d", *(short
*)vData);
```

```
                                        break;

                                case SQL_C_LONG:
                                        j = sprintf(s, "%ld", *(long
*)vData);
                                        break;

                                case SQL_C_UBIGINT:
                                        j = sprintf(s, "%I64d",
*(__int64 *)vData);
                                        break;

                                case SQL_C_FLOAT:
                                        sprintf(fmt, "%%.0%df",
iScale);
                                        j = sprintf(s, fmt, *(float
*)vData);
                                        break;

                                case SQL_C_DOUBLE:
                                case SQL_C_NUMERIC:
                                        sprintf(fmt, "%%.0%df",
iScale);
                                        j = sprintf(s, fmt, *(double
*)vData);
                                        break;

                                default:
                                        j = sprintf(s, "%s", vData);
                                        break;
                                }
                        }

                        // Strip off terminating null character and pad the string
with blanks
                        if (iColLen - j > 0)
                                memset(s + j, ' ', iColLen - j);

                        *(s + iColLen) = '\0';

                        // Write the field to the output file
                        _bstr_t temp(s);
                        pOutput->WriteField((BSTR)temp);
                        free(s);
                }

                return;
}

SQLSMALLINT CExecute::GetDefaultCType(SQLINTEGER SQLType)
{
                // GetDefaultCType returns the C type for the passed in SQL
datatype.

   switch(SQLType)
   {
   case SQL_CHAR:
                case SQL_VARCHAR:
                case SQL_LONGVARCHAR:
                case SQL_WCHAR:
                case SQL_WVARCHAR:
                case SQL_WLONGVARCHAR:
        return(SQL_C_CHAR);

   case SQL_TINYINT:
                                        return(SQL_C_CHAR);

   case SQL_SMALLINT:
                                        return(SQL_C_SHORT);

   case SQL_INTEGER:
                                        return(SQL_C_LONG);

   case SQL_BIGINT:
                                        return(SQL_C_UBIGINT);

   case SQL_REAL:
                                        return(SQL_C_FLOAT);

   case SQL_FLOAT:
   case SQL_DOUBLE:
//                case SQL_DECIMAL:
                                        return(SQL_C_DOUBLE);

                case SQL_DECIMAL:
                                        return(SQL_C_CHAR);

                case SQL_BIT:
                                        return(SQL_C_CHAR);

   case SQL_BINARY:
                case SQL_VARBINARY:
                case SQL_LONGVARBINARY:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_BINARY);

   case SQL_TYPE_DATE:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_TYPE_DATE);

   case SQL_TYPE_TIME:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_TYPE_TIME);

   case SQL_TYPE_TIMESTAMP:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_TYPE_TIMESTAMP);

                case SQL_NUMERIC:
        return(SQL_C_FLOAT);

   case SQL_INTERVAL_YEAR:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_YEAR);

   case SQL_INTERVAL_MONTH:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_MONTH);

   case SQL_INTERVAL_YEAR_TO_MONTH:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_YEAR_TO_MONTH);

   case SQL_INTERVAL_DAY:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_DAY);

   case SQL_INTERVAL_HOUR:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_HOUR);

   case SQL_INTERVAL_MINUTE:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_MINUTE);

   case SQL_INTERVAL_SECOND:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_SECOND);

   case SQL_INTERVAL_DAY_TO_HOUR:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_DAY_TO_HOUR);

   case SQL_INTERVAL_DAY_TO_MINUTE:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_DAY_TO_MINUTE);

   case SQL_INTERVAL_DAY_TO_SECOND:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_DAY_TO_SECOND);

   case SQL_INTERVAL_HOUR_TO_MINUTE:
```

```
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_HOUR_TO_MINUTE);

   case SQL_INTERVAL_HOUR_TO_SECOND:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_HOUR_TO_SECOND);

   case SQL_INTERVAL_MINUTE_TO_SECOND:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_MINUTE_TO_SECOND);

                default:
                        assert(TRUE);
                        return(SQL_C_CHAR);
                        break;
   }
}

/*
   FUNCTION: LogODBCErrors(SQLRETURN rc, SWORD fHandleType,
SQLHANDLE handle)
   COMMENTS: Formats ODBC errors or warnings and logs them. Also
initializes the
                                        completion status for the step to failure, if
an ODBC error has occurred.
*/

void CExecute::LogODBCErrors(SQLRETURN nResult, SWORD fHandleType,
SQLHANDLE handle, OdbcOperations FailedOp)
{
                // Messages returned by the server (e.g. Print statements) will be
logged to the output file
                // ODBC warnings will be logged to the log file
                // All other ODBC errors will be logged to the error file.

                UCHAR                   szErrState[SQL_SQLSTATE_SIZE+1];
                // SQL Error State string
                UCHAR
szErrText[SQL_MAX_MESSAGE_LENGTH+1];    // SQL Error Text string
                char
szBuffer[SQL_SQLSTATE_SIZE+SQL_MAX_MESSAGE_LENGTH+MAXBU
FLEN+1] = "";


// formatted Error text Buffer
                SWORD                   wErrMsgLen;
                                                        // Error message
length
                SQLINTEGER              dwErrCode;
                                        // Native Error code
                SQLRETURN               nErrResult;
                                        // Return Code from
SQLGetDiagRec
                SWORD                   sMsgNum = 1;
                                        // Error sequence number
                _bstr_t                 temp;

                if (IsErrorReturn(nResult))
                {
                        sprintf(szBuffer, "ODBC Operation: '%s' returned error
code: %d",
                                        m_szOdbcOps[FailedOp], nResult);
                        temp = szBuffer;
                        m_pErrorFile->WriteLine((BSTR)temp);
                        m_StepStatus = gintFailed;
                }

                if (handle == SQL_NULL_HSTMT)
                        return;

#ifdef _DEBUG
                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLGetDiagRec.\n");
#endif
                // call SQLGetDiagRec function with proper ODBC handles,
repeatedly until
                // function returns SQL_NO_DATA.
                while (!m_bAbort && (nErrResult = SQLGetDiagRec(fHandleType,
handle, sMsgNum++,
                        szErrState, &dwErrCode, szErrText,
SQL_MAX_MESSAGE_LENGTH-1, &wErrMsgLen))
                        != SQL_NO_DATA)
                {
                        if (!SQL_SUCCEEDED(nErrResult))
                                break;

                        if (m_pOutputFile && IsServerMessage(dwErrCode,
szErrText))
                        {
                                wsprintf(szBuffer,
SM_SQLMSG_FORMAT, (LPSTR)szErrText);
                                temp = szBuffer;
                                m_pOutputFile->WriteLine((BSTR)temp);
                        }
                        else if (IsODBCWarning(szErrState) && dwErrCode
!= 5701 && dwErrCode != 5703)
                        {
                                // Suppress warnings - 'Changed database
context to...' and 'Changed language setting to...'
                                wsprintf(szBuffer,
SM_SQLMSG_FORMAT, ParseOdbcMsgPrefixes((LPCSTR)szErrText));
                                temp = szBuffer;
                                m_pOutputFile->WriteLine((BSTR)temp);
                        }
                        else if (m_pErrorFile &&
!IsODBCWarning(szErrState))
                        {
                                wsprintf(szBuffer,
SM_SQLERR_FORMAT, (LPSTR)szErrState, dwErrCode, (LPSTR)szErrText);
                                temp = szBuffer;
                                m_pErrorFile->WriteLine((BSTR)temp);
                        }
                }
}

/*
   FUNCTION: ODBCCleanup(HDBC *hdbc, HSTMT *hstmt)
   COMMENTS: Cleanup of all ODBC structures
*/

void CExecute::ODBCCleanup(HDBC *hdbc, HSTMT *hstmt)
{
                SQLRETURN               lReturn;

#ifdef _DEBUG
                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
ODBCCleanup.\n");
#endif

                if (*hstmt != SQL_NULL_HSTMT)
                {
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLCloseCursor.\n");
#endif
                        SQLCloseCursor(hstmt);
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLFreeHandle for hstmt.\n");
#endif
                        SQLFreeHandle(SQL_HANDLE_STMT, hstmt);
                        *hstmt = SQL_NULL_HSTMT;
                }

                // Cleanup connection if it is a dynamic connection
                if (IsDynamicConnection())
                {
                        if (*hdbc != SQL_NULL_HDBC)
                        {
#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0,
NULL, "Executing SQLDisconnect.\n");
```

```
#endif
                                        lReturn = SQLDisconnect(*hdbc);
#ifdef _DEBUG
                                        _CrtDbgReport(_CRT_WARN, NULL, 0,
NULL, "Executing SQLFreeHandle for hdbc.\n");
#endif
                                        SQLFreeHandle(SQL_HANDLE_DBC,
hdbc);
                                        *hdbc = SQL_NULL_HDBC;
                                }
                        }
                        else
                                ResetConnectionUsage();

                        return;
}

// Wrapper function that raises an error if a Windows Api fails
STDMETHODIMP CExecute::RaiseSystemError(void)
{
                        char s[MAXBUFLEN];

                        GetSystemError(s);
                        return Error(s, 0, NULL, GUID_NULL);
}

// Wrapper function that logs the error raised by an Api function to the passed in
file
void CExecute::LogSystemError(ISMLog*pFile)
{
                        if (pFile)
                        {
                                char s[MAXBUFLEN];
                                GetSystemError(s);

                                _bstr_t temp(s);
                                pFile->WriteLine((BSTR)temp);
                        }
}

// Populates the passed in string with the last Windows Api error that occurred
void CExecute::GetSystemError(LPSTR s)
{
                        long c;
                        DWORD e;

                        e = GetLastError();

                        c = sprintf(s, "Error code: %ld. ", e);
                        c = FormatMessage(FORMAT_MESSAGE_FROM_SYSTEM |
FORMAT_MESSAGE_IGNORE_INSERTS,
                                NULL, e, 0, s + c, MAXBUFLEN - c, NULL);

                        return;
}

STDMETHODIMP CExecute::get_StepStatus(InstanceStatus *pVal)
{
                        *pVal = m_StepStatus;
                        return S_OK;
}

STDMETHODIMP CExecute::WriteError(BSTR szMsg)
{
                        if (m_pErrorFile)
                                return(m_pErrorFile->WriteLine(szMsg));

                        return S_OK;
}


// Execute.h : Declaration of the CExecute

#ifndef __EXECUTE_H_
#define __EXECUTE_H_

#include <atlwin.h>
#include <comdef.h>
#include <stdio.h>
#include "resource.h"       // main symbols
#include "ExecuteDllCP.h"
#include "..\LogWriter\LogWriter.h"
#include "..\LogWriter\SMLog.h"
#include "..\common\SMTime\SMTime.h"
#include "..\common\SMTime\SMTimer.h"

// ODBC-specific includes
#define DBNTWIN32
#include <sqltypes.h>
#include <sql.h>
#include <sqlext.h>

/////////////////////////////////////////////////////////////////////////////
// CExecute

#define WM_TASK_START  (WM_USER + 101)
#define WM_TASK_FINISH  (WM_USER + 102)

#define SM_SQLERR_FORMAT                "SQL Error State:%s, Native
Error Code: %ld\r\nODBC Error: %s"

                                                                        //
format for ODBC error messages
#define SM_SQLWARN_FORMAT               SM_SQLERR_FORMAT
// format for ODBC warnings
#define SM_SQLMSG_FORMAT                "%s"
        // format for messages from the server

#define SM_SQL_STATE_WARNING            "01000"
#define SM_MSG_SERVER
"[Microsoft][ODBC SQL Server Driver][SQL Server]"

#define SM_ERR_CONN_IN_USE              "StepMaster Error:
Connection is already in use."

#define CMD_SEPARATOR                                   "\nGO"

#define INV_ARRAY_INDEX     -1                          // invalid index
into an array

#define MAXBUFLEN           256     // display buffer size
#define MAXLOGCMDLEN        256     // maximum characters in command
that will be

                                    // printed to log
#define MAXLOGCMDBUF        512     // maximum characters in command
that will be

                                    // printed to log
#define MAX_DATA_LEN        4000    // maximum buffer size for
variable-length data types

                                    // viz. character and binary fields
#define FILE_ACCESS_READ    "r"                         // Open file for
read access

#define ALIGNSIZE 4
#define S_NULL              "NULL"
#define ALIGNBUF(Length) Length % ALIGNSIZE ? \
   Length + ALIGNSIZE - (Length % ALIGNSIZE) : Length

class ATL_NO_VTABLE CExecute :
        public CWindowImpl<CExecute>,
        public CComObjectRootEx<CComSingleThreadModel>,
        public CComCoClass<CExecute, &CLSID_Execute>,
        public IConnectionPointContainerImpl<CExecute>,
        public ISupportErrorInfo,
        public IDispatchImpl<IExecute, &IID_IExecute,
&LIBID_EXECUTEDLLLib>,
        public CProxy_IExecuteEvents< CExecute >
{
public:
        CExecute()
        {
                m_pErrorFile = NULL;
                //m_pLogFile = NULL;
```

```
                m_pOutputFile = NULL;

                // Initialize the elapsed time for the step
                m_tElapsedTime = 0;

                // Initialize the run status for the step
                m_StepStatus = gintPending;

                m_hHandle = SQL_NULL_HSTMT;
                m_bAbort = FALSE;

                m_iConnectionIndex = INV_ARRAY_INDEX;
        }

        ~CExecute()
        {
        }

public:
        DECLARE_WND_CLASS("Execute")

                BEGIN_MSG_MAP(CExecute)
                MESSAGE_HANDLER(WM_TASK_FINISH,
OnTaskFinished)
                MESSAGE_HANDLER(WM_TASK_START,
OnTaskStarted)
                END_MSG_MAP()
public:

        LRESULT OnTaskStarted(UINT uMsg, WPARAM wParam,
                LPARAM lParam, BOOL& bHandled)
        {
                CURRENCY        CStartTime =
Get64BitTime(&m_tStartTime);

                Fire_Start(CStartTime);
                return 0;
        }

        LRESULT OnTaskFinished(UINT uMsg, WPARAM wParam,
                LPARAM lParam, BOOL& bHandled)
        {
                CURRENCY        CEndTime =
Get64BitTime(&m_tEndTime);

                Fire_Complete(CEndTime, (long)m_tElapsedTime);
                return 0;
        }

        HRESULT FinalConstruct()
        {
                HRESULT         hr;
                RECT            rect;

                rect.left=0;
                rect.right=100;
                rect.top=0;
                rect.bottom=100;

                HWND hwnd = Create( NULL, rect,
"ExecuteWindow", WS_POPUP);

                if (!hwnd)
                        return
HRESULT_FROM_WIN32(GetLastError());

                hr = CoCreateInstance(CLSID_SMLog, NULL,
CLSCTX_INPROC,
                        IID_ISMLog, (void **)&m_pErrorFile);
                if FAILED(hr)
                        return(hr);
                m_pErrorFile->put_Append(TRUE);

                //hr = CoCreateInstance(CLSID_SMLog, NULL,
CLSCTX_INPROC,
                //      IID_ISMLog, (void **)&m_pLogFile);
                //if FAILED(hr)
                //      return(hr);

                hr = CoCreateInstance(CLSID_SMLog, NULL,
CLSCTX_INPROC,
                        IID_ISMLog, (void **)&m_pOutputFile);
                if FAILED(hr)
                        return(hr);
                m_pOutputFile->put_Append(TRUE);

                hr = CoCreateInstance(CLSID_SMTimer, NULL,
CLSCTX_INPROC,
                        IID_ISMTimer, (void **)&m_ExecTime);
                if FAILED(hr)
                        return(hr);

                return S_OK;
        }

        void FinalRelease()
        {
                if (m_hWnd != NULL)
                        DestroyWindow();

                // Close the log and error files
                if (m_pErrorFile)
                        m_pErrorFile->Release();
                m_pErrorFile = NULL;

                //if (m_pLogFile)
                //      m_pLogFile->Release();
                //m_pLogFile = NULL;

                if (m_ExecTime)
                        m_ExecTime->Release();
                m_ExecTime = NULL;
        }

DECLARE_REGISTRY_RESOURCEID(IDR_EXECUTE)

DECLARE_PROTECT_FINAL_CONSTRUCT()

BEGIN_COM_MAP(CExecute)
        COM_INTERFACE_ENTRY(IExecute)
        COM_INTERFACE_ENTRY(ISupportErrorInfo)
        COM_INTERFACE_ENTRY(IDispatch)
        COM_INTERFACE_ENTRY(IConnectionPointContainer)
        COM_INTERFACE_ENTRY_IMPL(IConnectionPointContainer)
END_COM_MAP()
BEGIN_CONNECTION_POINT_MAP(CExecute)
CONNECTION_POINT_ENTRY(DIID__IExecuteEvents)
END_CONNECTION_POINT_MAP()

// ISupportsErrorInfo
STDMETHOD(InterfaceSupportsErrorInfo)(REFIIDriid);

// IExecute
public:
        STDMETHOD(put_ErrorFile)(/*[in]*/BSTR newVal);
        STDMETHOD(put_OutputFile)(/*[in]*/BSTR newVal);
        STDMETHOD(WriteError)(BSTR szMsg);
        STDMETHOD(Abort)();
        STDMETHOD(get_StepStatus)(/*[out, retval]*/ InstanceStatus
*pVal);
        STDMETHOD(DoExecute)(/*[in]*/BSTR szCommand, /*[in]*/
BSTR szExecutionDtls, /*[in]*/ ExecutionType ExecMethod,
                /*[in]*/ BOOL bNoCount, /*[in]*/ BOOL bNoExecute,
/*[in]*/ BOOL bParseOnly,
                /*[in]*/ BOOL bQuotedIds, /*[in]*/ BOOL bAnsiNulls,
/*[in]*/ BOOL bShowQP,
                /*[in]*/ BOOL bStatsTime, /*[in]*/ BOOL bStatsIO,
/*[in]*/ long lRowCount,
                /*[in]*/ long lQueryTmout, /*[in]*/ BSTR
szConnection);

        TC_TIME                 ExecuteShell(CExecute *p);
        TC_TIME                 ExecuteODBC(CExecute *p);
        STDMETHODIMP    AbortShell();
        STDMETHODIMP    AbortODBC();
```

```
        _bstr_t                 m_szCommand;
        _bstr_t                 m_szExecDtls;
        _bstr_t                 m_szConnection;
        DWORD                   m_lMode;
        //DATE                  m_CurTime;
        SYSTEMTIME              m_tStartTime;
        SYSTEMTIME              m_tEndTime;
        TC_TIME                 m_tElapsedTime;
        ISMLog                  *m_pErrorFile;
        //ILog                  *m_pLogFile;
        ISMLog                  *m_pOutputFile;
        ISMTimer        *m_ExecTime;
        ExecutionType   m_ExecMthd;
        InstanceStatus  m_StepStatus;
        HANDLE                  m_hHandle;      //
Process handle for shell commands and

                // Statement handle for ODBC commands
        LPSTR                   m_szCmd;

private:
        typedef enum OdbcOperations
        {
                SMSQLAllocHandle,
                SMSQLDriverConnect,
                SMSQLExecDirect,
                SMSQLSetStmtAttr,
                SMSQLCancel,
                SMSQLNumResultCols,
                SMSQLDescribeCol,
                SMSQLColAttribute,
                SMSQLFetch,
                SMSQLGetData,
                SMSQLRowCount,
                SMSQLMoreResults
        };

        LPSTR                   NextCmdInBatch(LPSTR
szBatch);
        void                    ProcessResultsets();
        SQLSMALLINT
GetDefaultCType(SQLINTEGER SQLType);
        void                    PrintData(void *vData,
SQLSMALLINT CType, SQLINTEGER IndPtr, SQLSMALLINT iScale,
ISMLog *pOutput);
        void
LogODBCErrors(SQLRETURN nResult, SWORD fHandleType, SQLHANDLE
handle, OdbcOperations FailedOp);
        void                    ODBCCleanup(HDBC
*hdbc, HSTMT *hstmt);
        STDMETHODIMP    RaiseSystemError(void);
        void                    LogSystemError(ISMLog
*pFile);
        void                    GetSystemError(LPSTR s);
        BOOL                    SetConnectionOption(LPSTR
szConn, HDBC *pHdbc);
        BOOL
ResetConnectionProperties(HDBC *p_hdbc);
        BOOL
HandleODBCError(SQLRETURN rc, SWORD fHandleType, SQLHANDLE
handle, HDBC *pHdbc, OdbcOperations OdbcOp);

        BOOL                    InitializeConnection(HDBC
*phdbc, BOOL *pbDoConnect);
        void                    ResetConnectionUsage();

        int
m_iConnectionIndex;

        static char     *m_szOdbcOps[];

        BOOL                    m_bNoCount,
m_bNoExecute, m_bParseOnly, m_bQuotedIds, m_bAnsiNulls, \
                                                m_bShowQP,
m_bStatsTime, m_bStatsIO;
        long                    m_lRowCount;
        SQLUINTEGER             m_lQueryTmout;
        _bstr_t                 m_ErrorFile, m_OutputFile;
        BOOL                    m_bAbort;

private:
inline BOOL IsServerMessage(SQLINTEGER lNativeError, UCHAR *szErr){
        return( (strstr((LPCTSTR)szErr, SM_MSG_SERVER) != NULL) ?
(lNativeError == 0) : FALSE); }
inline BOOL IsODBCWarning(UCHAR *szSqlState){
return(strcmp((LPCSTR)szSqlState, SM_SQL_STATE_WARNING) == 0);}
inline BOOL IsErrorReturn(SQLRETURN iRetCode){
        return( (iRetCode != SQL_SUCCESS) && (iRetCode !=
SQL_SUCCESS_WITH_INFO) && (iRetCode != SQL_NO_DATA) );}
inline LPCSTR ParseOdbcMsgPrefixes(LPCSTR szMsg){ char *pDest;
        return( (pDest = strstr(szMsg, SM_MSG_SERVER)) == NULL ?
szMsg : pDest + strlen(SM_MSG_SERVER));}
inline BOOL IsDynamicConnection(){ return(!strcmp((LPSTR)m_szConnection,
""));}
};

void                            ExecutionThread(LPVOID lpParameter);

#ifdef _TPCH_AUDIT
        void                    WriteFileToTpchLog(LPSTR
szFile, LPSTR szFmt);
        void                    WriteToTpchLog(char
*szMsg);
#endif

#endif //__EXECUTE_H_

HKCR
{
        ExecuteDll.Execute.1 = s 'Execute Class'
        {
                CLSID = s
'{2EFC198E-AA8D-11D2-BC0C-00A0C90D2CA5}'
        }
        ExecuteDll.Execute = s 'Execute Class'
        {
                CLSID = s
'{2EFC198E-AA8D-11D2-BC0C-00A0C90D2CA5}'
                CurVer = s 'ExecuteDll.Execute.1'
        }
        NoRemove CLSID
        {
                ForceRemove
{2EFC198E-AA8D-11D2-BC0C-00A0C90D2CA5} = s 'Execute Class'
                {
                        ProgID = s 'ExecuteDll.Execute.1'
                        VersionIndependentProgID = s
'ExecuteDll.Execute'
                        ForceRemove 'Programmable'
                        InprocServer32 = s '%MODULE%'
                        {
                                val ThreadingModel = s
'Apartment'
                        }
                        'TypeLib' = s
'{551AC525-AB1C-11D2-BC0C-00A0C90D2CA5}'
                }
        }
}


// ExecuteDll.cpp : Implementation of DLL Exports.


// Note: Proxy/Stub Information
//      To build a separate proxy/stub DLL,
//      run nmake -f ExecuteDllps.mk in the project directory.

#include "stdafx.h"
#include "resource.h"
#include <initguid.h>

#include "..\LogWriter\LogWriter.h"
#include "..\LogWriter\LogWriter_i.c"

#include "..\common\SMTime\SMTime.h"
```

```
#include "..\common\SMTime\SMTime_i.c"

#include "ExecuteDll.h"
#include "SMExecute.h"

#include "ExecuteDll_i.c"
#include "Execute.h"

CComModule _Module;

BEGIN_OBJECT_MAP(ObjectMap)
OBJECT_ENTRY(CLSID_Execute, CExecute)
END_OBJECT_MAP()

SQLHENV henv = NULL;
                                        // ODBC environment handle

static char szCaption[] = "StepMaster";
        // Message box caption

CRITICAL_SECTION hConnections;
                // Critical section to serialize access to available
connections
SM_Connection_Info      *p_Connections = NULL;
        // Pointer to open connections
int
iConnectionCount = 0;                                   // Number of
open connections

#ifdef _TPCH_AUDIT
        FILE *pfLogFile = NULL;
                                        // Log file containing
timestamps
        CRITICAL_SECTION hLogFileWrite;
                        // Critical section to serialize writes to log

        static char szFileOpenModeAppend[] = "a+";
// Log file open mode

        static char szEnvVarLogFile[] = "TPCH_LOG_FILE";        //
Environment variable - initialized to


// log file name if timing information


// is to be logged
#endif

void ShowODBCErrors(SWORD fHandleType, SQLHANDLE handle);
void CloseOpenConnections();

////////////////////////////////////////////////////////////////////////
// DLL Entry Point

extern "C"
BOOL WINAPI DllMain(HINSTANCE hInstance, DWORD dwReason,
LPVOID /*lpReserved*/)
{

   if (dwReason == DLL_PROCESS_ATTACH)
   {
      _Module.Init(ObjectMap, hInstance, &LIBID_EXECUTEDLLLib);
      DisableThreadLibraryCalls(hInstance);

#ifdef _TPCH_AUDIT
                char szMsg[MAXBUFLEN];
                LPSTR szLogFileName = getenv(szEnvVarLogFile);

                if (szLogFileName == NULL)
                {
                        sprintf(szMsg, "The environment variable
'%s' does not exist. "
                                "Step timing information will
not be written to a log.", szEnvVarLogFile);
                        MessageBox(NULL, szMsg, szCaption,
MB_OK);
                }
                else
                {
                        if ( (pfLogFile = fopen(szLogFileName,
szFileOpenModeAppend)) == NULL )
                        {
                                sprintf(szMsg, "The file '%s'
does not exist. "
                                        "Step timing
information will not be written to log.",szLogFileName);
                                MessageBox(NULL, szMsg,
szCaption, MB_OK);
                        }
                        else

InitializeCriticalSection(&hLogFileWrite);
                }
#endif

                InitializeCriticalSection(&hConnections);

                p_Connections = NULL;
                iConnectionCount = 0;

                if (!SQL_SUCCEEDED(SQLSetEnvAttr(NULL,
SQL_ATTR_CONNECTION_POOLING,
(SQLPOINTER)SQL_CP_ONE_PER_HENV , 0)))
                        ShowODBCErrors(SQL_HANDLE_ENV,
henv);

                if
(!SQL_SUCCEEDED(SQLAllocHandle(SQL_HANDLE_ENV,
SQL_NULL_HANDLE, &henv)))
                        return FALSE;
                /*
                SQLUINTEGER CpMatch;
                if (!SQL_SUCCEEDED(SQLGetEnvAttr(henv,
SQL_ATTR_CP_MATCH, &CpMatch, 0, NULL)))
                        ShowODBCErrors(SQL_HANDLE_ENV,
henv);

                if (!SQL_SUCCEEDED(SQLSetEnvAttr(henv,
SQL_ATTR_CP_MATCH, (SQLPOINTER)SQL_CP_STRICT_MATCH,
SQL_IS_INTEGER)))
                        ShowODBCErrors(SQL_HANDLE_ENV,
henv);
                        */

                if (!SQL_SUCCEEDED(SQLSetEnvAttr(henv,
SQL_ATTR_ODBC_VERSION, (LPVOID)SQL_OV_ODBC3, 0)))
                        ShowODBCErrors(SQL_HANDLE_ENV,
henv);

   }
   else if (dwReason == DLL_PROCESS_DETACH)
        {

#ifdef _TPCH_AUDIT
                if (pfLogFile != NULL)
                {
                        fclose(pfLogFile);

                        DeleteCriticalSection(&hLogFileWrite);
                }
#endif

                CloseOpenConnections();

                if (henv != NULL)
                        SQLFreeEnv(henv);

                DeleteCriticalSection(&hConnections);

      _Module.Term();
        }
   return TRUE;    // ok
}

void ShowODBCErrors(SWORD fHandleType, SQLHANDLE handle)
```

```
{
	UCHAR		szErrState[SQL_SQLSTATE_SIZE+1];
	// SQL Error State string
	UCHAR
szErrText[SQL_MAX_MESSAGE_LENGTH+1];	// SQL Error Text string
	char
szBuffer[SQL_SQLSTATE_SIZE+SQL_MAX_MESSAGE_LENGTH+MAXBU
FLEN+1] = "";


// formatted Error text Buffer
	SWORD		wErrMsgLen;
							// Error message
length
	SQLINTEGER	dwErrCode;
							// Native Error code
	SQLRETURN	nErrResult;
							// Return Code from
SQLGetDiagRec
	SWORD		sMsgNum = 1;
							// Error sequence number

	// call SQLGetDiagRec function with proper ODBC handles,
repeatedly until
	// function returns SQL_NO_DATA.
	while ((nErrResult = SQLGetDiagRec(fHandleType, handle,
sMsgNum++,
			szErrState, &dwErrCode, szErrText,
SQL_MAX_MESSAGE_LENGTH-1, &wErrMsgLen))
			!= SQL_NO_DATA)
	{
			if (!SQL_SUCCEEDED(nErrResult))
				break;

			wsprintf(szBuffer, SM_SQLERR_FORMAT,
(LPSTR)szErrState, dwErrCode, (LPSTR)szErrText);

			MessageBox(NULL, szBuffer, szCaption, MB_OK);
	}
}

void CloseOpenConnections()
{
	// Closes all open connections

	if (p_Connections)
	{
			for (int iConnIndex = iConnectionCount - 1;
iConnIndex >= 0; iConnIndex--)
			{
				if ((p_Connections + iConnIndex)->hdbc
!= SQL_NULL_HDBC)
				{
#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLDisconnect.\n");
#endif

SQLDisconnect((p_Connections + iConnIndex)->hdbc);
#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing SQLFreeHandle
for hdbc.\n");
#endif

SQLFreeHandle(SQL_HANDLE_DBC, (p_Connections + iConnIndex)->hdbc );
							(p_Connections +
iConnIndex)->hdbc = SQL_NULL_HDBC;
				}

			}

			free(p_Connections);
	}
	p_Connections = NULL;

	return;
}

////////////////////////////////////////////////////////////////////////
// Used to determine whether the DLL can be unloaded by OLE

STDAPI DllCanUnloadNow(void)
{
  return (_Module.GetLockCount()==0) ? S_OK : S_FALSE;
}

////////////////////////////////////////////////////////////////////////
// Returns a class factory to create an object of the requested type

STDAPI DllGetClassObject(REFCLSID rclsid, REFIID riid, LPVOID* ppv)
{
  return _Module.GetClassObject(rclsid, riid, ppv);
}

////////////////////////////////////////////////////////////////////////
// DllRegisterServer - Adds entries to the system registry

STDAPI DllRegisterServer(void)
{
  // registers object, typelib and all interfaces in typelib
  return _Module.RegisterServer(TRUE);
}

////////////////////////////////////////////////////////////////////////
// DllUnregisterServer - Removes entries from the system registry

STDAPI DllUnregisterServer(void)
{
  return _Module.UnregisterServer(TRUE);
}
HKCR
{
	ExecuteDll.Execute.1 = s 'Execute Class'
	{
		CLSID = s
'{2EFC198E-AA8D-11D2-BC0C-00A0C90D2CA5}'
	}
	ExecuteDll.Execute = s 'Execute Class'
	{
		CLSID = s
'{2EFC198E-AA8D-11D2-BC0C-00A0C90D2CA5}'
		CurVer = s 'ExecuteDll.Execute.1'
	}
	NoRemove CLSID
	{
		ForceRemove
{2EFC198E-AA8D-11D2-BC0C-00A0C90D2CA5} = s 'Execute Class'
		{
			ProgID = s 'ExecuteDll.Execute.1'
			VersionIndependentProgID = s
'ExecuteDll.Execute'
			ForceRemove 'Programmable'
			InprocServer32 = s '%MODULE%'
			{
				val ThreadingModel = s
'Apartment'
			}
			'TypeLib' = s
'{551AC525-AB1C-11D2-BC0C-00A0C90D2CA5}'
		}
	}
}
// ExecuteDll.cpp : Implementation of DLL Exports.


// Note: Proxy/Stub Information
//      To build a separate proxy/stub DLL,
//      run nmake -f ExecuteDllps.mk in the project directory.

#include "stdafx.h"
#include "resource.h"
#include <initguid.h>

#include "..\LogWriter\LogWriter.h"
```

```
#include "..\LogWriter\LogWriter_i.c"

#include "..\common\SMTime\SMTime.h"
#include "..\common\SMTime\SMTime_i.c"

#include "ExecuteDll.h"
#include "SMExecute.h"

#include "ExecuteDll_i.c"
#include "Execute.h"

CComModule _Module;

BEGIN_OBJECT_MAP(ObjectMap)
OBJECT_ENTRY(CLSID_Execute, CExecute)
END_OBJECT_MAP()

SQLHENV henv = NULL;
                                                // ODBC environment handle

static char szCaption[] = "StepMaster";
        // Message box caption

CRITICAL_SECTION hConnections;
                        // Critical section to serialize access to available
connections
SM_Connection_Info      *p_Connections = NULL;
        // Pointer to open connections
int
iConnectionCount = 0;                                   // Number of
open connections

#ifdef _TPCH_AUDIT
        FILE *pfLogFile = NULL;
                                                // Log file containing
timestamps
        CRITICAL_SECTION hLogFileWrite;
                                // Critical section to serialize writes to log

        static char szFileOpenModeAppend[] = "a+";
// Log file open mode

        static char szEnvVarLogFile[] = "TPCH_LOG_FILE";        //
Environment variable - initialized to


// log file name if timing information


// is to be logged
#endif

void ShowODBCErrors(SWORD fHandleType, SQLHANDLE handle);
void CloseOpenConnections();

////////////////////////////////////////////////////////////////////////
// DLL Entry Point

extern "C"
BOOL WINAPI DllMain(HINSTANCE hInstance, DWORD dwReason,
LPVOID /*lpReserved*/)
{

   if (dwReason == DLL_PROCESS_ATTACH)
   {
      _Module.Init(ObjectMap, hInstance, &LIBID_EXECUTEDLLLib);
      DisableThreadLibraryCalls(hInstance);

#ifdef _TPCH_AUDIT
                        char szMsg[MAXBUFLEN];
                        LPSTR szLogFileName = getenv(szEnvVarLogFile);

                        if (szLogFileName == NULL)
                        {
                                sprintf(szMsg, "The environment variable
'%s' does not exist. "
                                                "Step timing information will
not be written to a log.", szEnvVarLogFile);
                                MessageBox(NULL, szMsg, szCaption,
MB_OK);
                        }
                        else
                        {
                                if ( (pfLogFile = fopen(szLogFileName,
szFileOpenModeAppend)) == NULL )
                                {
                                        sprintf(szMsg, "The file '%s'
does not exist. "
                                                        "Step timing
information will not be written to log.",szLogFileName);
                                        MessageBox(NULL, szMsg,
szCaption, MB_OK);
                                }
                                else

InitializeCriticalSection(&hLogFileWrite);
                        }
#endif

                        InitializeCriticalSection(&hConnections);

                        p_Connections = NULL;
                        iConnectionCount = 0;

                        if (!SQL_SUCCEEDED(SQLSetEnvAttr(NULL,
SQL_ATTR_CONNECTION_POOLING,
(SQLPOINTER)SQL_CP_ONE_PER_HENV , 0)))
                                ShowODBCErrors(SQL_HANDLE_ENV,
henv);

                        if
(!SQL_SUCCEEDED(SQLAllocHandle(SQL_HANDLE_ENV,
SQL_NULL_HANDLE, &henv)))
                                return FALSE;
                        /*
                        SQLUINTEGER CpMatch;
                        if (!SQL_SUCCEEDED(SQLGetEnvAttr(henv,
SQL_ATTR_CP_MATCH, &CpMatch, 0, NULL)))
                                ShowODBCErrors(SQL_HANDLE_ENV,
henv);

                        if (!SQL_SUCCEEDED(SQLSetEnvAttr(henv,
SQL_ATTR_CP_MATCH, (SQLPOINTER)SQL_CP_STRICT_MATCH,
SQL_IS_INTEGER)))
                                ShowODBCErrors(SQL_HANDLE_ENV,
henv);
                                */

                        if (!SQL_SUCCEEDED(SQLSetEnvAttr(henv,
SQL_ATTR_ODBC_VERSION, (LPVOID)SQL_OV_ODBC3, 0)))
                                ShowODBCErrors(SQL_HANDLE_ENV,
henv);

   }
   else if (dwReason == DLL_PROCESS_DETACH)
        {

#ifdef _TPCH_AUDIT
                        if (pfLogFile != NULL)
                        {
                                fclose(pfLogFile);

                                DeleteCriticalSection(&hLogFileWrite);
                        }
#endif

                        CloseOpenConnections();

                        if (henv != NULL)
                                SQLFreeEnv(henv);

                        DeleteCriticalSection(&hConnections);

      _Module.Term();
        }
   return TRUE;    // ok
```

```
}

void ShowODBCErrors(SWORD fHandleType, SQLHANDLE handle)
{
	UCHAR		szErrState[SQL_SQLSTATE_SIZE+1];
	// SQL Error State string
	UCHAR
szErrText[SQL_MAX_MESSAGE_LENGTH+1];	// SQL Error Text string
	char
szBuffer[SQL_SQLSTATE_SIZE+SQL_MAX_MESSAGE_LENGTH+MAXBU
FLEN+1] = "";


// formatted Error text Buffer
	SWORD		wErrMsgLen;
						// Error message
length
	SQLINTEGER	dwErrCode;
						// Native Error code
	SQLRETURN	nErrResult;
						// Return Code from
SQLGetDiagRec
	SWORD		sMsgNum = 1;
						// Error sequence number

	// call SQLGetDiagRec function with proper ODBC handles,
repeatedly until
	// function returns SQL_NO_DATA.
	while ((nErrResult = SQLGetDiagRec(fHandleType, handle,
sMsgNum++,
			szErrState, &dwErrCode, szErrText,
SQL_MAX_MESSAGE_LENGTH-1, &wErrMsgLen))
			!= SQL_NO_DATA)
	{
			if (!SQL_SUCCEEDED(nErrResult))
				break;

			wsprintf(szBuffer, SM_SQLERR_FORMAT,
(LPSTR)szErrState, dwErrCode, (LPSTR)szErrText);

			MessageBox(NULL, szBuffer, szCaption, MB_OK);
	}
}

void CloseOpenConnections()
{
	// Closes all open connections

	if (p_Connections)
	{
			for (int iConnIndex = iConnectionCount - 1;
iConnIndex >= 0; iConnIndex--)
			{
				if ((p_Connections + iConnIndex)->hdbc
!= SQL_NULL_HDBC)
				{
#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLDisconnect.\n");
#endif

SQLDisconnect((p_Connections + iConnIndex)->hdbc);
#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing SQLFreeHandle
for hdbc.\n");
#endif

SQLFreeHandle(SQL_HANDLE_DBC,(p_Connections + iConnIndex)->hdbc );
						(p_Connections +
iConnIndex)->hdbc = SQL_NULL_HDBC;
				}

			}

			free(p_Connections);
	}
		p_Connections = NULL;

		return;
}

//////////////////////////////////////////////////////////////////////
// Used to determine whether the DLL can be unloaded by OLE

STDAPI DllCanUnloadNow(void)
{
  return (_Module.GetLockCount()==0) ? S_OK : S_FALSE;
}

//////////////////////////////////////////////////////////////////////
// Returns a class factory to create an object of the requested type

STDAPI DllGetClassObject(REFCLSID rclsid, REFIID riid, LPVOID* ppv)
{
  return _Module.GetClassObject(rclsid, riid, ppv);
}

//////////////////////////////////////////////////////////////////////
// DllRegisterServer - Adds entries to the system registry

STDAPI DllRegisterServer(void)
{
  // registers object, typelib and all interfaces in typelib
  return _Module.RegisterServer(TRUE);
}

//////////////////////////////////////////////////////////////////////
// DllUnregisterServer - Removes entries from the system registry

STDAPI DllUnregisterServer(void)
{
  return _Module.UnregisterServer(TRUE);
}
; ExecuteDll.def : Declares the module parameters.

LIBRARY      "ExecuteDll.DLL"

EXPORTS
		DllCanUnloadNow     @1 PRIVATE
		DllGetClassObject   @2 PRIVATE
		DllRegisterServer   @3 PRIVATE
		DllUnregisterServer         @4 PRIVATE


# Microsoft Developer Studio Project File - Name="ExecuteDll" - Package
Owner=<4>
# Microsoft Developer Studio Generated Build File, Format Version 6.00
# ** DO NOT EDIT **

# TARGTYPE "Win32 (x86) Dynamic-Link Library" 0x0102

CFG=ExecuteDll - Win32 Debug
!MESSAGE This is not a valid makefile. To build this project using NMAKE,
!MESSAGE use the Export Makefile command and run
!MESSAGE
!MESSAGE NMAKE /f "ExecuteDll.mak".
!MESSAGE
!MESSAGE You can specify a configuration when running NMAKE
!MESSAGE by defining the macro CFG on the command line. For example:
!MESSAGE
!MESSAGE NMAKE /f "ExecuteDll.mak" CFG="ExecuteDll - Win32 Debug"
!MESSAGE
!MESSAGE Possible choices for configuration are:
!MESSAGE
!MESSAGE "ExecuteDll - Win32 Debug" (based on "Win32 (x86) Dynamic-Link
Library")
!MESSAGE "ExecuteDll - Win32 Unicode Debug" (based on "Win32 (x86)
Dynamic-Link Library")
!MESSAGE "ExecuteDll - Win32 Release MinSize" (based on "Win32 (x86)
Dynamic-Link Library")
!MESSAGE "ExecuteDll - Win32 Release MinDependency" (based on "Win32
(x86) Dynamic-Link Library")
!MESSAGE "ExecuteDll - Win32 Unicode Release MinSize" (based on "Win32
(x86) Dynamic-Link Library")
```

```
!MESSAGE "ExecuteDll - Win32 Unicode Release MinDependency" (based on
"Win32 (x86) Dynamic-Link Library")
!MESSAGE

# Begin Project
# PROP AllowPerConfigDependencies 0
# PROP Scc_ProjName ""$/StepMaster", KVMAAAAA"
# PROP Scc_LocalPath ".."
CPP=cl.exe
MTL=midl.exe
RSC=rc.exe

!IF  "$(CFG)" == "ExecuteDll - Win32 Debug"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 1
# PROP BASE Output_Dir "Debug"
# PROP BASE Intermediate_Dir "Debug"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 1
# PROP Output_Dir "lib\x86"
# PROP Intermediate_Dir "objects\x86"
# PROP Ignore_Export_Lib 0
# PROP Target_Dir ""
# ADD BASE CPP /nologo /MTd /W3 /Gm /ZI /Od /D "WIN32" /D "_DEBUG"
/D "_WINDOWS" /D "_MBCS" /D "_USRDLL" /Yu"stdafx.h" /FD /GZ /c
# ADD CPP /nologo /MTd /W3 /Gm /ZI /Od /D "_DEBUG" /D "_MBCS" /D
"WIN32" /D "_WINDOWS" /D "_USRDLL" /D "_TPCH_AUDIT" /FR
/YX"stdafx.h" /FD /GZ /c
# ADD BASE RSC /l 0x409 /d "_DEBUG"
# ADD RSC /l 0x409 /d "_DEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib /nologo /subsystem:windows /dll /debug /machine:I386
/pdbtype:sept
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib odbccp32.lib
..\common\lib\x86\SMTime.lib..\LogWriter\lib\x86\LogWriter.lib/nologo
/subsystem:windows /dll /map /debug /machine:I386 /nodefaultlib:"libcd.lib"
/pdbtype:sept
# Begin Custom Build - Performing registration
OutDir=.\lib\x86
TargetPath=.\lib\x86\ExecuteDll.dll
InputPath=.\lib\x86\ExecuteDll.dll
SOURCE="$(InputPath)"

"$(OutDir)\regsvr32.trg": $(SOURCE) "$(INTDIR)" "$(OUTDIR)"
                regsvr32 /s /c "$(TargetPath)"
                echo regsvr32 exec. time > "$(OutDir)\regsvr32.trg"

# End Custom Build

!ELSEIF  "$(CFG)" == "ExecuteDll - Win32 Unicode Debug"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 1
# PROP BASE Output_Dir "DebugU"
# PROP BASE Intermediate_Dir "DebugU"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 1
# PROP Output_Dir "lib\x86"
# PROP Intermediate_Dir "objects\x86"
# PROP Ignore_Export_Lib 0
# PROP Target_Dir ""
# ADD BASE CPP /nologo /MTd /W3 /Gm /ZI /Od /D "WIN32" /D "_DEBUG"
/D "_WINDOWS" /D "_USRDLL" /D "_UNICODE" /Yu"stdafx.h" /FD /GZ /c
# ADD CPP /nologo /MDd /W3 /Gm /ZI /Od /D "_DEBUG" /D "_UNICODE"
/D "WIN32" /D "_WINDOWS" /D "_USRDLL" /D "_TPCH_AUDIT"
/Yu"stdafx.h" /FD /GZ /c
# ADD BASE RSC /l 0x409 /d "_DEBUG"
# ADD RSC /l 0x409 /d "_DEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib /nologo /subsystem:windows /dll /debug /machine:I386
/pdbtype:sept
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib odbccp32.lib
..\LogWriter\lib\x86\LogWriter.lib/nologo /subsystem:windows /dll /debug
/machine:I386 /pdbtype:sept
# Begin Custom Build - Performing registration
OutDir=.\lib\x86
TargetPath=.\lib\x86\ExecuteDll.dll
InputPath=.\lib\x86\ExecuteDll.dll
SOURCE="$(InputPath)"

"$(OutDir)\regsvr32.trg": $(SOURCE) "$(INTDIR)" "$(OUTDIR)"
                if "%OS%"=="" goto NOTNT
                if not "%OS%"=="Windows_NT" goto NOTNT
                regsvr32 /s /c "$(TargetPath)"
                echo regsvr32 exec. time > "$(OutDir)\regsvr32.trg"
                goto end
                :NOTNT
                echo Warning : Cannot register Unicode DLL on Windows 95
                :end

# End Custom Build

!ELSEIF  "$(CFG)" == "ExecuteDll - Win32 Release MinSize"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 0
# PROP BASE Output_Dir "ReleaseMinSize"
# PROP BASE Intermediate_Dir "ReleaseMinSize"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 0
# PROP Output_Dir "lib\x86"
# PROP Intermediate_Dir "objects\x86"
# PROP Ignore_Export_Lib 0
# PROP Target_Dir ""
# ADD BASE CPP /nologo /MT /W3 /O1 /D "WIN32" /D "NDEBUG" /D
"_WINDOWS" /D "_MBCS" /D "_USRDLL" /D "_ATL_DLL" /D
"_ATL_MIN_CRT" /Yu"stdafx.h" /FD /c
# ADD CPP /nologo /MD /W3 /O1 /D "_MBCS" /D "_ATL_DLL" /D
"NDEBUG" /D "WIN32" /D "_WINDOWS" /D "_USRDLL" /D
"_TPCH_AUDIT" /Yu"stdafx.h" /FD /c
# ADD BASE RSC /l 0x409 /d "NDEBUG"
# ADD RSC /l 0x409 /d "NDEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib /nologo /subsystem:windows /dll /machine:I386
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib odbccp32.lib
..\LogWriter\lib\x86\LogWriter.lib/nologo /subsystem:windows /dll
/machine:I386
# Begin Custom Build - Performing registration
OutDir=.\lib\x86
TargetPath=.\lib\x86\ExecuteDll.dll
InputPath=.\lib\x86\ExecuteDll.dll
SOURCE="$(InputPath)"

"$(OutDir)\regsvr32.trg": $(SOURCE) "$(INTDIR)" "$(OUTDIR)"
                regsvr32 /s /c "$(TargetPath)"
                echo regsvr32 exec. time > "$(OutDir)\regsvr32.trg"

# End Custom Build

!ELSEIF  "$(CFG)" == "ExecuteDll - Win32 Release MinDependency"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 0
# PROP BASE Output_Dir "ReleaseMinDependency"
```

```
# PROP BASE Intermediate_Dir "ReleaseMinDependency"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 0
# PROP Output_Dir "lib\x86"
# PROP Intermediate_Dir "objects\x86"
# PROP Ignore_Export_Lib 0
# PROP Target_Dir ""
# ADD BASE CPP /nologo /MT /W3 /O1 /D "WIN32" /D "NDEBUG" /D
"_WINDOWS" /D "_MBCS" /D "_USRDLL" /D "_ATL_STATIC_REGISTRY"
/D "_ATL_MIN_CRT" /Yu"stdafx.h" /FD /c
# ADD CPP /nologo /MD /W3 /O1 /D "_MBCS" /D
"_ATL_STATIC_REGISTRY" /D "NDEBUG" /D "WIN32" /D "_WINDOWS"
/D "_USRDLL" /D "_TPCH_AUDIT" /Yu"stdafx.h" /FD /c
# ADD BASE RSC /l 0x409 /d "NDEBUG"
# ADD RSC /l 0x409 /d "NDEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib /nologo /subsystem:windows /dll /machine:I386
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib odbccp32.lib
..\LogWriter\lib\x86\LogWriter.lib/nologo /subsystem:windows /dll
/machine:I386
# Begin Custom Build - Performing registration
OutDir=.\lib\x86
TargetPath=.\lib\x86\ExecuteDll.dll
InputPath=.\lib\x86\ExecuteDll.dll
SOURCE="$(InputPath)"

"$(OutDir)\regsvr32.trg": $(SOURCE) "$(INTDIR)" "$(OUTDIR)"
        regsvr32 /s /c "$(TargetPath)"
        echo regsvr32 exec. time > "$(OutDir)\regsvr32.trg"

# End Custom Build

!ELSEIF  "$(CFG)" == "ExecuteDll - Win32 Unicode Release MinSize"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 0
# PROP BASE Output_Dir "ReleaseUMinSize"
# PROP BASE Intermediate_Dir "ReleaseUMinSize"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 0
# PROP Output_Dir "lib\x86"
# PROP Intermediate_Dir "objects\x86"
# PROP Ignore_Export_Lib 0
# PROP Target_Dir ""
# ADD BASE CPP /nologo /MT /W3 /O1 /D "WIN32" /D "NDEBUG" /D
"_WINDOWS" /D "_USRDLL" /D "_UNICODE" /D "_ATL_DLL" /D
"_ATL_MIN_CRT" /Yu"stdafx.h" /FD /c
# ADD CPP /nologo /MD /W3 /O1 /D "_UNICODE" /D "_ATL_DLL" /D
"NDEBUG" /D "WIN32" /D "_WINDOWS" /D "_USRDLL" /D
"_TPCH_AUDIT" /Yu"stdafx.h" /FD /c
# ADD BASE RSC /l 0x409 /d "NDEBUG"
# ADD RSC /l 0x409 /d "NDEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib /nologo /subsystem:windows /dll /machine:I386
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib odbccp32.lib
..\LogWriter\lib\x86\LogWriter.lib/nologo /subsystem:windows /dll
/machine:I386
# Begin Custom Build - Performing registration
OutDir=.\lib\x86
TargetPath=.\lib\x86\ExecuteDll.dll
InputPath=.\lib\x86\ExecuteDll.dll
SOURCE="$(InputPath)"

"$(OutDir)\regsvr32.trg": $(SOURCE) "$(INTDIR)" "$(OUTDIR)"
        if "%OS%"=="" goto NOTNT
        if not "%OS%"=="Windows_NT" goto NOTNT
        regsvr32 /s /c "$(TargetPath)"
        echo regsvr32 exec. time > "$(OutDir)\regsvr32.trg"
        goto end
        :NOTNT
        echo Warning : Cannot register Unicode DLL on Windows 95
        :end

# End Custom Build

!ELSEIF  "$(CFG)" == "ExecuteDll - Win32 Unicode Release MinDependency"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 0
# PROP BASE Output_Dir "ReleaseUMinDependency"
# PROP BASE Intermediate_Dir "ReleaseUMinDependency"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 0
# PROP Output_Dir "lib\x86"
# PROP Intermediate_Dir "objects\x86"
# PROP Ignore_Export_Lib 0
# PROP Target_Dir ""
# ADD BASE CPP /nologo /MT /W3 /O1 /D "WIN32" /D "NDEBUG" /D
"_WINDOWS" /D "_USRDLL" /D "_UNICODE" /D
"_ATL_STATIC_REGISTRY" /D "_ATL_MIN_CRT" /Yu"stdafx.h" /FD /c
# ADD CPP /nologo /MD /W3 /O1 /D "_UNICODE" /D
"_ATL_STATIC_REGISTRY" /D "NDEBUG" /D "WIN32" /D "_WINDOWS"
/D "_USRDLL" /D "_TPCH_AUDIT" /FR /Yu"stdafx.h" /FD /c
# ADD BASE RSC /l 0x409 /d "NDEBUG"
# ADD RSC /l 0x409 /d "NDEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib /nologo /subsystem:windows /dll /machine:I386
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib odbccp32.lib
..\LogWriter\lib\x86\LogWriter.lib/nologo /subsystem:windows /dll
/machine:I386
# Begin Custom Build - Performing registration
OutDir=.\lib\x86
TargetPath=.\lib\x86\ExecuteDll.dll
InputPath=.\lib\x86\ExecuteDll.dll
SOURCE="$(InputPath)"

"$(OutDir)\regsvr32.trg": $(SOURCE) "$(INTDIR)" "$(OUTDIR)"
        if "%OS%"=="" goto NOTNT
        if not "%OS%"=="Windows_NT" goto NOTNT
        regsvr32 /s /c "$(TargetPath)"
        echo regsvr32 exec. time > "$(OutDir)\regsvr32.trg"
        goto end
        :NOTNT
        echo Warning : Cannot register Unicode DLL on Windows 95
        :end

# End Custom Build

!ENDIF

# Begin Target

# Name "ExecuteDll - Win32 Debug"
# Name "ExecuteDll - Win32 Unicode Debug"
# Name "ExecuteDll - Win32 Release MinSize"
# Name "ExecuteDll - Win32 Release MinDependency"
# Name "ExecuteDll - Win32 Unicode Release MinSize"
# Name "ExecuteDll - Win32 Unicode Release MinDependency"
# Begin Group "Source Files"

# PROP Default_Filter "cpp;c;cxx;rc;def;r;odl;idl;hpj;bat"
# Begin Source File

SOURCE=.\Execute.cpp
# End Source File
```

```
# Begin Source File

SOURCE=.\ExecuteDll.cpp
# End Source File
# Begin Source File

SOURCE=.\ExecuteDll.def
# End Source File
# Begin Source File

SOURCE=.\ExecuteDll.idl
# ADD MTL /tlb ".\ExecuteDll.tlb" /h "ExecuteDll.h" /iid "ExecuteDll_i.c" /Oicf
# End Source File
# Begin Source File

SOURCE=.\ExecuteDll.rc
# End Source File
# Begin Source File

SOURCE=.\StdAfx.cpp
# ADD CPP /Yc"stdafx.h"
# End Source File
# End Group
# Begin Group "Header Files"

# PROP Default_Filter "h;hpp;hxx;hm;inl"
# Begin Source File

SOURCE=.\Execute.h
# End Source File
# Begin Source File

SOURCE=.\ExecuteDllCP.h
# End Source File
# Begin Source File

SOURCE=.\Resource.h
# End Source File
# Begin Source File

SOURCE=.\SMExecute.h
# End Source File
# Begin Source File

SOURCE=.\StdAfx.h
# End Source File
# End Group
# Begin Group "Resource Files"

# PROP Default_Filter "ico;cur;bmp;dlg;rc2;rct;bin;rgs;gif;jpg;jpeg;jpe"
# Begin Source File

SOURCE=.\Execute.rgs
# End Source File
# End Group
# End Target
# End Project
Microsoft Developer Studio Workspace File, Format Version 6.00
# WARNING: DO NOT EDIT OR DELETE THIS WORKSPACE FILE!

###############################################################################
#############

Project: "ExecuteDll"=".\ExecuteDll.dsp" - Package Owner=<4>

Package=<5>
{{{
}}}

Package=<4>
{{{
}}}

###############################################################################
#############

Global:

Package=<5>
{{{
}}}

Package=<3>
{{{
}}}

###############################################################################
#############
/* this ALWAYS GENERATED file contains the definitions for the interfaces */


/* File created by MIDL compiler version 5.01.0164 */
/* at Fri Sep 24 12:00:06 1999
 */
/* Compiler settings for D:\Rst\StepMaster-VB5.0\ExecuteDll\ExecuteDll.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32, ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
*/
//@@MIDL_FILE_HEADING( )


/* verify that the <rpcndr.h> version is high enough to compile thisfile*/
#ifndef __REQUIRED_RPCNDR_H_VERSION__
#define __REQUIRED_RPCNDR_H_VERSION__ 440
#endif

#include "rpc.h"
#include "rpcndr.h"

#ifndef __ExecuteDll_h__
#define __ExecuteDll_h__

#ifdef __cplusplus
extern "C"{
#endif

/* Forward Declarations */

#ifndef ___IExecuteEvents_FWD_DEFINED__
#define ___IExecuteEvents_FWD_DEFINED__
typedef interface _IExecuteEvents _IExecuteEvents;
#endif          /* ___IExecuteEvents_FWD_DEFINED__ */


#ifndef __IExecute_FWD_DEFINED__
#define __IExecute_FWD_DEFINED__
typedef interface IExecute IExecute;
#endif          /* __IExecute_FWD_DEFINED__ */


#ifndef __Execute_FWD_DEFINED__
#define __Execute_FWD_DEFINED__

#ifdef __cplusplus
typedef class Execute Execute;
#else
typedef struct Execute Execute;
#endif /* __cplusplus */

#endif          /* __Execute_FWD_DEFINED__ */


/* header files for imported files */
#include "oaidl.h"
#include "ocidl.h"

void __RPC_FAR * __RPC_USER MIDL_user_allocate(size_t);
void __RPC_USER MIDL_user_free( void __RPC_FAR * );

/* interface __MIDL_itf_ExecuteDll_0000 */
/* [local] */

typedef /* [helpstring][uuid] */
enum ExecutionType
    {           execODBC  = 0x1,
                execShell     = 0x2
```

```
    }       ExecutionType;

typedef /* [helpstring][uuid] */
enum InstanceStatus
    {       gintDisabled= 0x1,
            gintPending = 0x2,
            gintRunning = 0x3,
            gintComplete        = 0x4,
            gintFailed    = 0x5,
            gintAborted = 0x6
    }       InstanceStatus;



extern RPC_IF_HANDLE __MIDL_itf_ExecuteDll_0000_v0_0_c_ifspec;
extern RPC_IF_HANDLE __MIDL_itf_ExecuteDll_0000_v0_0_s_ifspec;


#ifndef __EXECUTEDLLLib_LIBRARY_DEFINED__
#define __EXECUTEDLLLib_LIBRARY_DEFINED__

/* library EXECUTEDLLLib */
/* [helpstring][version][uuid] */


EXTERN_C const IID LIBID_EXECUTEDLLLib;

#ifndef ___IExecuteEvents_DISPINTERFACE_DEFINED__
#define ___IExecuteEvents_DISPINTERFACE_DEFINED__

/* dispinterface _IExecuteEvents */
/* [helpstring][uuid] */


EXTERN_C const IID DIID__IExecuteEvents;

#if defined(__cplusplus) && !defined(CINTERFACE)

    MIDL_INTERFACE("551AC532-AB1C-11D2-BC0C-00A0C90D2CA5")
    _IExecuteEvents : public IDispatch
    {
    };

#else       /* C style interface */

    typedef struct _IExecuteEventsVtbl
    {
        BEGIN_INTERFACE

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *QueryInterface )(
            _IExecuteEvents __RPC_FAR * This,
            /* [in] */ REFIID riid,
            /* [iid_is][out] */ void __RPC_FAR *__RPC_FAR *ppvObject);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *AddRef )(
            _IExecuteEvents __RPC_FAR * This);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *Release )(
            _IExecuteEvents __RPC_FAR * This);

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *GetTypeInfoCount
)(
            _IExecuteEvents __RPC_FAR * This,
            /* [out] */ UINT __RPC_FAR *pctinfo);

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *GetTypeInfo )(
            _IExecuteEvents __RPC_FAR * This,
            /* [in] */ UINT iTInfo,
            /* [in] */ LCID lcid,
            /* [out] */ ITypeInfo __RPC_FAR *__RPC_FAR *ppTInfo);

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *GetIDsOfNames )(
            _IExecuteEvents __RPC_FAR * This,
            /* [in] */ REFIID riid,
            /* [size_is][in] */ LPOLESTR __RPC_FAR *rgszNames,
            /* [in] */ UINT cNames,
            /* [in] */ LCID lcid,
            /* [size_is][out] */ DISPID __RPC_FAR *rgDispId);

        /* [local] */ HRESULT ( STDMETHODCALLTYPE __RPC_FAR *Invoke
)(
            _IExecuteEvents __RPC_FAR * This,
            /* [in] */ DISPID dispIdMember,
            /* [in] */ REFIID riid,
            /* [in] */ LCID lcid,
            /* [in] */ WORD wFlags,
            /* [out][in] */ DISPPARAMS __RPC_FAR *pDispParams,
            /* [out] */ VARIANT __RPC_FAR *pVarResult,
            /* [out] */ EXCEPINFO __RPC_FAR *pExcepInfo,
            /* [out] */ UINT __RPC_FAR *puArgErr);

        END_INTERFACE
    } _IExecuteEventsVtbl;

    interface _IExecuteEvents
    {
        CONST_VTBL struct _IExecuteEventsVtbl __RPC_FAR *lpVtbl;
    };



#ifdef COBJMACROS


#define _IExecuteEvents_QueryInterface(This,riid,ppvObject)   \
    (This)->lpVtbl -> QueryInterface(This,riid,ppvObject)

#define _IExecuteEvents_AddRef(This)                \
    (This)->lpVtbl -> AddRef(This)

#define _IExecuteEvents_Release(This)               \
    (This)->lpVtbl -> Release(This)


#define _IExecuteEvents_GetTypeInfoCount(This,pctinfo)        \
    (This)->lpVtbl -> GetTypeInfoCount(This,pctinfo)

#define _IExecuteEvents_GetTypeInfo(This,iTInfo,lcid,ppTInfo)
    (This)->lpVtbl -> GetTypeInfo(This,iTInfo,lcid,ppTInfo)

#define
_IExecuteEvents_GetIDsOfNames(This,riid,rgszNames,cNames,lcid,rgDispId)
\
    (This)->lpVtbl -> GetIDsOfNames(This,riid,rgszNames,cNames,lcid,rgDispId)

#define
_IExecuteEvents_Invoke(This,dispIdMember,riid,lcid,wFlags,pDispParams,pVar
Result,pExcepInfo,puArgErr)             \
    (This)->lpVtbl ->
Invoke(This,dispIdMember,riid,lcid,wFlags,pDispParams,pVarResult,pExcepInfo
,puArgErr)

#endif /* COBJMACROS */


#endif      /* C style interface */


#endif      /* ___IExecuteEvents_DISPINTERFACE_DEFINED__ */


#ifndef __IExecute_INTERFACE_DEFINED__
#define __IExecute_INTERFACE_DEFINED__

/* interface IExecute */
/* [unique][helpstring][dual][uuid][object] */


EXTERN_C const IID IID_IExecute;

#if defined(__cplusplus) && !defined(CINTERFACE)

    MIDL_INTERFACE("551AC531-AB1C-11D2-BC0C-00A0C90D2CA5")
    IExecute : public IDispatch
    {
    public:
```

```
        virtual /* [helpstring][id] */ HRESULT STDMETHODCALLTYPE
DoExecute(
            /* [in] */ BSTR szCommand,
            /* [in] */ BSTR szExecutionDtls,
            /* [in] */ ExecutionType ExecMethod,
            /* [in] */ BOOL bNoCount,
            /* [in] */ BOOL bNoExecute,
            /* [in] */ BOOL bParseOnly,
            /* [in] */ BOOL bQuotedIds,
            /* [in] */ BOOL bAnsiNulls,
            /* [in] */ BOOL bShowQP,
            /* [in] */ BOOL bStatsTime,
            /* [in] */ BOOL bStatsIO,
            /* [in] */ long lRowCount,
            /* [in] */ long lQueryTmout,
            /* [in] */ BSTR szConnection) = 0;

        virtual /* [helpstring][id][propget] */ HRESULT STDMETHODCALLTYPE
get_StepStatus(
            /* [retval][out] */ InstanceStatus __RPC_FAR *pVal) = 0;

        virtual /* [helpstring][id] */ HRESULT STDMETHODCALLTYPE Abort(
void) = 0;

        virtual /* [helpstring][id] */ HRESULT STDMETHODCALLTYPE
WriteError(
            BSTR szMsg) = 0;

        virtual /* [helpstring][id][propput] */ HRESULT STDMETHODCALLTYPE
put_OutputFile(
            /* [in] */ BSTR newVal) = 0;

        virtual /* [helpstring][id][propput] */ HRESULT STDMETHODCALLTYPE
put_ErrorFile(
            /* [in] */ BSTR newVal) = 0;

    };

#else           /* C style interface */

    typedef struct IExecuteVtbl
    {
        BEGIN_INTERFACE

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *QueryInterface )(
            IExecute __RPC_FAR * This,
            /* [in] */ REFIID riid,
            /* [iid_is][out] */ void __RPC_FAR *__RPC_FAR *ppvObject);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *AddRef )(
            IExecute __RPC_FAR * This);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *Release )(
            IExecute __RPC_FAR * This);

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *GetTypeInfoCount
)(
            IExecute __RPC_FAR * This,
            /* [out] */ UINT __RPC_FAR *pctinfo);

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *GetTypeInfo )(
            IExecute __RPC_FAR * This,
            /* [in] */ UINT iTInfo,
            /* [in] */ LCID lcid,
            /* [out] */ ITypeInfo __RPC_FAR *__RPC_FAR *ppTInfo);

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *GetIDsOfNames )(
            IExecute __RPC_FAR * This,
            /* [in] */ REFIID riid,
            /* [size_is][in] */ LPOLESTR __RPC_FAR *rgszNames,
            /* [in] */ UINT cNames,
            /* [in] */ LCID lcid,
            /* [size_is][out] */ DISPID __RPC_FAR *rgDispId);

        /* [local] */ HRESULT ( STDMETHODCALLTYPE __RPC_FAR *Invoke
)(
            IExecute __RPC_FAR * This,
            /* [in] */ DISPID dispIdMember,
            /* [in] */ REFIID riid,
            /* [in] */ LCID lcid,
            /* [in] */ WORD wFlags,
            /* [out][in] */ DISPPARAMS __RPC_FAR *pDispParams,
            /* [out] */ VARIANT __RPC_FAR *pVarResult,
            /* [out] */ EXCEPINFO __RPC_FAR *pExcepInfo,
            /* [out] */ UINT __RPC_FAR *puArgErr);

        /* [helpstring][id] */ HRESULT ( STDMETHODCALLTYPE __RPC_FAR
*DoExecute )(
            IExecute __RPC_FAR * This,
            /* [in] */ BSTR szCommand,
            /* [in] */ BSTR szExecutionDtls,
            /* [in] */ ExecutionType ExecMethod,
            /* [in] */ BOOL bNoCount,
            /* [in] */ BOOL bNoExecute,
            /* [in] */ BOOL bParseOnly,
            /* [in] */ BOOL bQuotedIds,
            /* [in] */ BOOL bAnsiNulls,
            /* [in] */ BOOL bShowQP,
            /* [in] */ BOOL bStatsTime,
            /* [in] */ BOOL bStatsIO,
            /* [in] */ long lRowCount,
            /* [in] */ long lQueryTmout,
            /* [in] */ BSTR szConnection);

        /* [helpstring][id][propget] */ HRESULT ( STDMETHODCALLTYPE
__RPC_FAR *get_StepStatus )(
            IExecute __RPC_FAR * This,
            /* [retval][out] */ InstanceStatus __RPC_FAR *pVal);

        /* [helpstring][id] */ HRESULT ( STDMETHODCALLTYPE __RPC_FAR
*Abort )(
            IExecute __RPC_FAR * This);

        /* [helpstring][id] */ HRESULT ( STDMETHODCALLTYPE __RPC_FAR
*WriteError )(
            IExecute __RPC_FAR * This,
            BSTR szMsg);

        /* [helpstring][id][propput] */ HRESULT ( STDMETHODCALLTYPE
__RPC_FAR *put_OutputFile )(
            IExecute __RPC_FAR * This,
            /* [in] */ BSTR newVal);

        /* [helpstring][id][propput] */ HRESULT ( STDMETHODCALLTYPE
__RPC_FAR *put_ErrorFile )(
            IExecute __RPC_FAR * This,
            /* [in] */ BSTR newVal);

        END_INTERFACE
    } IExecuteVtbl;

    interface IExecute
    {
        CONST_VTBL struct IExecuteVtbl __RPC_FAR *lpVtbl;
    };



#ifdef COBJMACROS


#define IExecute_QueryInterface(This,riid,ppvObject)            \
    (This)->lpVtbl -> QueryInterface(This,riid,ppvObject)

#define IExecute_AddRef(This)       \
    (This)->lpVtbl -> AddRef(This)

#define IExecute_Release(This)      \
    (This)->lpVtbl -> Release(This)


#define IExecute_GetTypeInfoCount(This,pctinfo) \
    (This)->lpVtbl -> GetTypeInfoCount(This,pctinfo)

#define IExecute_GetTypeInfo(This,iTInfo,lcid,ppTInfo)          \
    (This)->lpVtbl -> GetTypeInfo(This,iTInfo,lcid,ppTInfo)
```

```
#define IExecute_GetIDsOfNames(This,riid,rgszNames,cNames,lcid,rgDispId)
\
    (This)->lpVtbl -> GetIDsOfNames(This,riid,rgszNames,cNames,lcid,rgDispId)

#define
IExecute_Invoke(This,dispIdMember,riid,lcid,wFlags,pDispParams,pVarResult,p
ExcepInfo,puArgErr)      \
    (This)->lpVtbl ->
Invoke(This,dispIdMember,riid,lcid,wFlags,pDispParams,pVarResult,pExcepInfo
,puArgErr)


#define
IExecute_DoExecute(This,szCommand,szExecutionDtls,ExecMethod,bNoCount,
bNoExecute,bParseOnly,bQuotedIds,bAnsiNulls,bShowQP,bStatsTime,bStatsIO,
lRowCount,lQueryTmout,szConnection)                \
    (This)->lpVtbl ->
DoExecute(This,szCommand,szExecutionDtls,ExecMethod,bNoCount,bNoExecu
te,bParseOnly,bQuotedIds,bAnsiNulls,bShowQP,bStatsTime,bStatsIO,lRowCoun
t,lQueryTmout,szConnection)

#define IExecute_get_StepStatus(This,pVal)          \
    (This)->lpVtbl -> get_StepStatus(This,pVal)

#define IExecute_Abort(This)             \
    (This)->lpVtbl -> Abort(This)

#define IExecute_WriteError(This,szMsg)             \
    (This)->lpVtbl -> WriteError(This,szMsg)

#define IExecute_put_OutputFile(This,newVal)      \
    (This)->lpVtbl -> put_OutputFile(This,newVal)

#define IExecute_put_ErrorFile(This,newVal)         \
    (This)->lpVtbl -> put_ErrorFile(This,newVal)

#endif /* COBJMACROS */


#endif          /* C style interface */



/* [helpstring][id] */ HRESULT STDMETHODCALLTYPE
IExecute_DoExecute_Proxy(
    IExecute __RPC_FAR * This,
    /* [in] */ BSTR szCommand,
    /* [in] */ BSTR szExecutionDtls,
    /* [in] */ ExecutionType ExecMethod,
    /* [in] */ BOOL bNoCount,
    /* [in] */ BOOL bNoExecute,
    /* [in] */ BOOL bParseOnly,
    /* [in] */ BOOL bQuotedIds,
    /* [in] */ BOOL bAnsiNulls,
    /* [in] */ BOOL bShowQP,
    /* [in] */ BOOL bStatsTime,
    /* [in] */ BOOL bStatsIO,
    /* [in] */ long lRowCount,
    /* [in] */ long lQueryTmout,
    /* [in] */ BSTR szConnection);


void __RPC_STUB IExecute_DoExecute_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


/* [helpstring][id][propget] */ HRESULT STDMETHODCALLTYPE
IExecute_get_StepStatus_Proxy(
    IExecute __RPC_FAR * This,
    /* [retval][out] */ InstanceStatus __RPC_FAR *pVal);


void __RPC_STUB IExecute_get_StepStatus_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


/* [helpstring][id] */ HRESULT STDMETHODCALLTYPE
IExecute_Abort_Proxy(
    IExecute __RPC_FAR * This);


void __RPC_STUB IExecute_Abort_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


/* [helpstring][id] */ HRESULT STDMETHODCALLTYPE
IExecute_WriteError_Proxy(
    IExecute __RPC_FAR * This,
    BSTR szMsg);


void __RPC_STUB IExecute_WriteError_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


/* [helpstring][id][propput] */ HRESULT STDMETHODCALLTYPE
IExecute_put_OutputFile_Proxy(
    IExecute __RPC_FAR * This,
    /* [in] */ BSTR newVal);


void __RPC_STUB IExecute_put_OutputFile_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


/* [helpstring][id][propput] */ HRESULT STDMETHODCALLTYPE
IExecute_put_ErrorFile_Proxy(
    IExecute __RPC_FAR * This,
    /* [in] */ BSTR newVal);


void __RPC_STUB IExecute_put_ErrorFile_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);



#endif          /* __IExecute_INTERFACE_DEFINED__ */


EXTERN_C const CLSID CLSID_Execute;

#ifdef __cplusplus

class DECLSPEC_UUID("2EFC198E-AA8D-11D2-BC0C-00A0C90D2CA5")
Execute;
#endif
#endif /* __EXECUTEDLLLib_LIBRARY_DEFINED__ */

/* Additional Prototypes for ALL interfaces */

/* end of Additional Prototypes */

#ifdef __cplusplus
}
#endif

#endif
```

```
// ExecuteDll.idl : IDL source for ExecuteDll.dll
//

// This file will be processed by the MIDL tool to
// produce the type library (ExecuteDll.tlb) and marshalling code.

import "oaidl.idl";
import "ocidl.idl";
	typedef
	[
		uuid(0AC32070-B0DB-11d2-BC0D-00A0C90D2CA5),
		helpstring("ExecutionTypes"),
	]

	enum ExecutionType
	{
		[helpstring("Shell")]	execODBC  = 0x0001,
		[helpstring("ODBC")]	execShell  = 0x0002
	} ExecutionType;

	typedef
	[

uuid(D4A4B9B0-BAE3-11d2-BC0F-00A0C90D2CA5),
		helpstring("Run Status Values"),
	]

	enum InstanceStatus
	{
		[helpstring("Disabled")]  gintDisabled= 0x0001,
		[helpstring("Pending")]		gintPending
= 0x0002,
		[helpstring("Running")]		gintRunning
= 0x0003,
		[helpstring("Complete")] gintComplete		=
0x0004,
		[helpstring("Failed")]		gintFailed
= 0x0005,
		[helpstring("Aborted")]		gintAborted
= 0x0006
	} InstanceStatus;


[
	uuid(551AC525-AB1C-11D2-BC0C-00A0C90D2CA5),
	version(1.0),
	helpstring("ExecuteDll 1.0 Type Library")
]
library EXECUTEDLLLib
{
	importlib("stdole32.tlb");
	importlib("stdole2.tlb");

	[

uuid(551AC532-AB1C-11D2-BC0C-00A0C90D2CA5),
		helpstring("_IExecuteEvents Interface")
	]
	dispinterface _IExecuteEvents
	{
		properties:
		methods:
		[id(1), helpstring("method Start")] void Start([in]
CURRENCY StartTime);
		[id(2), helpstring("method Complete")] void
Complete([in] CURRENCY EndTime, [in] long Elapsed);
	};
	[
		object,

uuid(551AC531-AB1C-11D2-BC0C-00A0C90D2CA5),
		dual,
		helpstring("IExecute Interface"),
		pointer_default(unique)
	]
	interface IExecute : IDispatch
	{
		[id(1), helpstring("method DoExecute")] HRESULT
DoExecute([in] BSTR szCommand, [in] BSTR szExecutionDtls, [in]
ExecutionType ExecMethod, [in] BOOL bNoCount, [in] BOOL bNoExecute, [in]
BOOL bParseOnly, [in] BOOL bQuotedIds, [in] BOOL bAnsiNulls, [in] BOOL
bShowQP, [in] BOOL bStatsTime, [in] BOOL bStatsIO, [in] long lRowCount,
[in] long lQueryTmout, [in] BSTR szConnection);
		[propget, id(2), helpstring("property StepStatus")]
HRESULT StepStatus([out, retval] InstanceStatus *pVal);
		[id(3), helpstring("method Abort")] HRESULT Abort();
		[id(4), helpstring("method WriteError")] HRESULT
WriteError(BSTR szMsg);
		[propput, id(5), helpstring("property OutputFile")]
HRESULT OutputFile([in] BSTR newVal);
		[propput, id(6), helpstring("property ErrorFile")]
HRESULT ErrorFile([in] BSTR newVal);
	};
	[

uuid(2EFC198E-AA8D-11D2-BC0C-00A0C90D2CA5),
		helpstring("Execute Class")
	]
	coclass Execute
	{
		[default] interface IExecute;
		[default, source] dispinterface _IExecuteEvents;
	};
};
//Microsoft Developer Studio generated resource script.
//
#include "resource.h"

#define APSTUDIO_READONLY_SYMBOLS
////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "winres.h"

////////////////////////////////////////////////////////////////////////
#undef APSTUDIO_READONLY_SYMBOLS

////////////////////////////////////////////////////////////////////////
// English (U.S.) resources

#if !defined(AFX_RESOURCE_DLL)|| defined(AFX_TARG_ENU)
#ifdef _WIN32
LANGUAGE LANG_ENGLISH, SUBLANG_ENGLISH_US
#pragma code_page(1252)
#endif //_WIN32

#ifdef APSTUDIO_INVOKED
////////////////////////////////////////////////////////////////////////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE DISCARDABLE
BEGIN
   "resource.h\0"
END

2 TEXTINCLUDE DISCARDABLE
BEGIN
   "#include ""winres.h""\r\n"
   "\0"
END

3 TEXTINCLUDE DISCARDABLE
BEGIN
   "1 TYPELIB ""ExecuteDll.tlb""\r\n"
   "\0"
END

#endif    // APSTUDIO_INVOKED


#ifndef _MAC
////////////////////////////////////////////////////////////////////////
```

```
//
// Version
//

VS_VERSION_INFO VERSIONINFO
 FILEVERSION 1,0,0,1
 PRODUCTVERSION 1,0,0,1
 FILEFLAGSMASK 0x3fL
#ifdef _DEBUG
 FILEFLAGS 0x1L
#else
 FILEFLAGS 0x0L
#endif
 FILEOS 0x4L
 FILETYPE 0x2L
 FILESUBTYPE 0x0L
BEGIN
    BLOCK "StringFileInfo"
    BEGIN
        BLOCK "040904B0"
        BEGIN
            VALUE "CompanyName", "\0"
            VALUE "FileDescription", "ExecuteDll Module\0"
            VALUE "FileVersion", "1, 0, 0, 1\0"
            VALUE "InternalName", "ExecuteDll\0"
            VALUE "LegalCopyright", "Copyright 1999\0"
            VALUE "OriginalFilename", "ExecuteDll.DLL\0"
            VALUE "ProductName", "ExecuteDll Module\0"
            VALUE "ProductVersion", "1, 0, 0, 1\0"
            VALUE "OLESelfRegister", "\0"
        END
    END
    BLOCK "VarFileInfo"
    BEGIN
        VALUE "Translation", 0x409, 1200
    END
END

#endif    // !_MAC


/////////////////////////////////////////////////////////////////////////////
//
// REGISTRY
//

IDR_EXECUTE             REGISTRY  DISCARDABLE    "Execute.rgs"

/////////////////////////////////////////////////////////////////////////////
//
// String Table
//

STRINGTABLE DISCARDABLE
BEGIN
    IDS_PROJNAME            "ExecuteDll"
END

#endif    // English (U.S.) resources
/////////////////////////////////////////////////////////////////////////////



#ifndef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 3 resource.
//
1 TYPELIB "ExecuteDll.tlb"

/////////////////////////////////////////////////////////////////////////////
#endif    // not APSTUDIO_INVOKED
MSFT
/* this file contains the actual definitions of*/
/* the IIDs and CLSIDs */

/* link this file in with the server and any clients*/
```

```
/* File created by MIDL compiler version 5.01.0164*/
/* at Fri Sep 24 12:00:06 1999
 */
/* Compiler settings for D:\Rst\StepMaster-VB5.0\ExecuteDll\ExecuteDll.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32, ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
*/
//@@MIDL_FILE_HEADING( )
#ifdef __cplusplus
extern "C"{
#endif


#ifndef __IID_DEFINED__
#define __IID_DEFINED__

typedef struct _IID
{
    unsigned long x;
    unsigned short s1;
    unsigned short s2;
    unsigned char  c[8];
} IID;

#endif // __IID_DEFINED__

#ifndef CLSID_DEFINED
#define CLSID_DEFINED
typedef IID CLSID;
#endif // CLSID_DEFINED

const IID LIBID_EXECUTEDLLLib =
{0x551AC525,0xAB1C,0x11D2,{0xBC,0x0C,0x00,0xA0,0xC9,0x0D,0x2C,0xA
5}};


const IID DIID__IExecuteEvents =
{0x551AC532,0xAB1C,0x11D2,{0xBC,0x0C,0x00,0xA0,0xC9,0x0D,0x2C,0xA
5}};


const IID IID_IExecute =
{0x551AC531,0xAB1C,0x11D2,{0xBC,0x0C,0x00,0xA0,0xC9,0x0D,0x2C,0xA
5}};


const CLSID CLSID_Execute =
{0x2EFC198E,0xAA8D,0x11D2,{0xBC,0x0C,0x00,0xA0,0xC9,0x0D,0x2C,0xA
5}};


#ifdef __cplusplus
}
#endif
/* this ALWAYS GENERATED file contains the proxy stub code*/


/* File created by MIDL compiler version 5.01.0164*/
/* at Mon Feb 01 18:30:25 1999
 */
/* Compiler settings for D:\Rst\StepMaster\ExecuteDll\ExecuteDll.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32, ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
*/
//@@MIDL_FILE_HEADING( )

#define USE_STUBLESS_PROXY


/* verify that the <rpcproxy.h> version is high enough to compile thisfile*/
#ifndef __REDQ_RPCPROXY_H_VERSION__
#define __REQUIRED_RPCPROXY_H_VERSION__ 440
#endif


#include "rpcproxy.h"
#ifndef __RPCPROXY_H_VERSION__
```

```
#error this stub requires an updated version of <rpcproxy.h>
#endif // __RPCPROXY_H_VERSION__


#include "ExecuteDll.h"

#define TYPE_FORMAT_STRING_SIZE   37
#define PROC_FORMAT_STRING_SIZE   59

typedef struct _MIDL_TYPE_FORMAT_STRING
    {
    short          Pad;
    unsigned char  Format[ TYPE_FORMAT_STRING_SIZE ];
    } MIDL_TYPE_FORMAT_STRING;

typedef struct _MIDL_PROC_FORMAT_STRING
    {
    short          Pad;
    unsigned char  Format[ PROC_FORMAT_STRING_SIZE ];
    } MIDL_PROC_FORMAT_STRING;


extern const MIDL_TYPE_FORMAT_STRING __MIDL_TypeFormatString;
extern const MIDL_PROC_FORMAT_STRING __MIDL_ProcFormatString;


/* Standard interface: __MIDL_itf_ExecuteDll_0000, ver. 0.0,

GUID={0x00000000,0x0000,0x0000,{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x
00}} */


/* Object interface: IUnknown, ver. 0.0,

GUID={0x00000000,0x0000,0x0000,{0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x
46}} */


/* Object interface: IDispatch, ver. 0.0,

GUID={0x00020400,0x0000,0x0000,{0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x
46}} */


/* Object interface: IExecute, ver. 0.0,

GUID={0x551AC531,0xAB1C,0x11D2,{0xBC,0x0C,0x00,0xA0,0xC9,0x0D,0x
2C,0xA5}} */


extern const MIDL_STUB_DESC Object_StubDesc;


extern const MIDL_SERVER_INFO IExecute_ServerInfo;

#pragma code_seg(".orpc")
extern const USER_MARSHAL_ROUTINE_QUADRUPLE
UserMarshalRoutines[1];

static const MIDL_STUB_DESC Object_StubDesc =
    {
    0,
    NdrOleAllocate,
    NdrOleFree,
    0,
    0,
    0,
    0,
    0,
    __MIDL_TypeFormatString.Format,
    1, /* -error bounds_check flag */
    0x20000, /* Ndr library version */
    0,
    0x50100a4, /* MIDL Version 5.1.164 */
    0,
    UserMarshalRoutines,
    0,  /* notify & notify_flag routine table */
    1,  /* Flags */
    0,  /* Reserved3 */
    0,  /* Reserved4 */
    0   /* Reserved5 */
    };

static const unsigned short IExecute_FormatStringOffsetTable[]=
    {
    (unsigned short) -1,
    (unsigned short) -1,
    (unsigned short) -1,
    (unsigned short) -1,
    0
    };

static const MIDL_SERVER_INFO IExecute_ServerInfo =
    {
    &Object_StubDesc,
    0,
    __MIDL_ProcFormatString.Format,
    &IExecute_FormatStringOffsetTable[-3],
    0,
    0,
    0,
    0
    };

static const MIDL_STUBLESS_PROXY_INFO IExecute_ProxyInfo =
    {
    &Object_StubDesc,
    __MIDL_ProcFormatString.Format,
    &IExecute_FormatStringOffsetTable[-3],
    0,
    0,
    0
    };

CINTERFACE_PROXY_VTABLE(8) _IExecuteProxyVtbl =
{
    &IExecute_ProxyInfo,
    &IID_IExecute,
    IUnknown_QueryInterface_Proxy,
    IUnknown_AddRef_Proxy,
    IUnknown_Release_Proxy ,
    0 /* (void *)-1 /* IDispatch::GetTypeInfoCount */ ,
    0 /* (void *)-1 /* IDispatch::GetTypeInfo */ ,
    0 /* (void *)-1 /* IDispatch::GetIDsOfNames */ ,
    0 /* IDispatch_Invoke_Proxy */ ,
    (void *)-1 /* IExecute::DoExecute */
};


static const PRPC_STUB_FUNCTION IExecute_table[] =
{
    STUB_FORWARDING_FUNCTION,
    STUB_FORWARDING_FUNCTION,
    STUB_FORWARDING_FUNCTION,
    STUB_FORWARDING_FUNCTION,
    NdrStubCall2
};

CInterfaceStubVtbl _IExecuteStubVtbl =
{
    &IID_IExecute,
    &IExecute_ServerInfo,
    8,
    &IExecute_table[-3],
    CStdStubBuffer_DELEGATING_METHODS
};

#pragma data_seg(".rdata")

static const USER_MARSHAL_ROUTINE_QUADRUPLE
UserMarshalRoutines[1] =
        {

            {
            BSTR_UserSize
            ,BSTR_UserMarshal
```

```
        ,BSTR_UserUnmarshal
        ,BSTR_UserFree
        }

    };


#if !defined(__RPC_WIN32__)
#error  Invalid build platform for this stub.
#endif

#if !(TARGET_IS_NT40_OR_LATER)
#error You need a Windows NT 4.0 or later to run this stub because it uses these features:
#error   -Oif or -Oicf, [wire_marshal] or [user_marshal] attribute, more than 32 methods in the interface.
#error However, your C/C++ compilation flags indicate you intend to run this app on earlier systems.
#error This app will die there with the RPC_X_WRONG_STUB_VERSION error.
#endif


static const MIDL_PROC_FORMAT_STRING __MIDL_ProcFormatString=
    {
        0,
        {

            /* Procedure DoExecute */

                                    0x33,               /* FC_AUTO_HANDLE */
                                    0x6c,               /* Old Flags:  object, Oi2 */
/*  2 */    NdrFcLong( 0x0 ),       /* 0 */
/*  6 */    NdrFcShort( 0x7 ),      /* 7 */
#ifndef _ALPHA_
/*  8 */    NdrFcShort( 0x20 ),     /* x86, MIPS, PPC Stack size/offset = 32 */
#else
                                    NdrFcShort( 0x40 ),     /* Alpha Stack size/offset = 64 */
#endif
/* 10 */    NdrFcShort( 0x6 ),      /* 6 */
/* 12 */    NdrFcShort( 0x8 ),      /* 8 */
/* 14 */    0x6,                    /* Oi2 Flags:  clt must size, has return, */
                                    0x7,                /* 7 */

            /* Parameter szCommand */

/* 16 */    NdrFcShort( 0x8b ),     /* Flags:  must size, must free, in, by val, */
#ifndef _ALPHA_
/* 18 */    NdrFcShort( 0x4 ),      /* x86, MIPS, PPC Stack size/offset = 4 */
#else
                                    NdrFcShort( 0x8 ),      /* Alpha Stack size/offset = 8 */
#endif
/* 20 */    NdrFcShort( 0x1a ),     /* Type Offset=26 */

            /* Parameter szOutputFile */

/* 22 */    NdrFcShort( 0x8b ),     /* Flags:  must size, must free, in, by val, */
#ifndef _ALPHA_
/* 24 */    NdrFcShort( 0x8 ),      /* x86, MIPS, PPC Stack size/offset = 8 */
#else
                                    NdrFcShort( 0x10 ),     /* Alpha Stack size/offset = 16 */
#endif
/* 26 */    NdrFcShort( 0x1a ),     /* Type Offset=26 */

            /* Parameter szLogFile */

/* 28 */    NdrFcShort( 0x8b ),     /* Flags:  must size, must free, in, by val, */
#ifndef _ALPHA_
/* 30 */    NdrFcShort( 0xc ),      /* x86, MIPS, PPC Stack size/offset = 12 */
#else
                                    NdrFcShort( 0x18 ),     /* Alpha Stack size/offset = 24 */
#endif
/* 32 */    NdrFcShort( 0x1a ),     /* Type Offset=26 */

            /* Parameter szErrorFile */

/* 34 */    NdrFcShort( 0x8b ),     /* Flags:  must size, must free, in, by val, */
#ifndef _ALPHA_
/* 36 */    NdrFcShort( 0x10 ),     /* x86, MIPS, PPC Stack size/offset = 16 */
#else
                                    NdrFcShort( 0x20 ),     /* Alpha Stack size/offset = 32 */
#endif
/* 38 */    NdrFcShort( 0x1a ),     /* Type Offset=26 */

            /* Parameter szExecutionDtls */

/* 40 */    NdrFcShort( 0x8b ),     /* Flags:  must size, must free, in, by val, */
#ifndef _ALPHA_
/* 42 */    NdrFcShort( 0x14 ),     /* x86, MIPS, PPC Stack size/offset = 20 */
#else
                                    NdrFcShort( 0x28 ),     /* Alpha Stack size/offset = 40 */
#endif
/* 44 */    NdrFcShort( 0x1a ),     /* Type Offset=26 */

            /* Parameter ExecMethod */

/* 46 */    NdrFcShort( 0x48 ),     /* Flags:  in, base type, */
#ifndef _ALPHA_
/* 48 */    NdrFcShort( 0x18 ),     /* x86, MIPS, PPC Stack size/offset = 24 */
#else
                                    NdrFcShort( 0x30 ),     /* Alpha Stack size/offset = 48 */
#endif
/* 50 */    0xd,                    /* FC_ENUM16 */
                                    0x0,                /* 0 */

            /* Return value */

/* 52 */    NdrFcShort( 0x70 ),     /* Flags:  out, return, base type, */
#ifndef _ALPHA_
/* 54 */    NdrFcShort( 0x1c ),     /* x86, MIPS, PPC Stack size/offset = 28 */
#else
                                    NdrFcShort( 0x38 ),     /* Alpha Stack size/offset = 56 */
#endif
/* 56 */    0x8,                    /* FC_LONG */
                                    0x0,                /* 0 */

                                    0x0
        }
    };

static const MIDL_TYPE_FORMAT_STRING __MIDL_TypeFormatString=
    {
        0,
        {
                                    NdrFcShort( 0x0 ),      /* 0 */
/*  2 */
                                    0x12, 0x0,  /* FC_UP */
/*  4 */    NdrFcShort( 0xc ),      /* Offset= 12 (16) */
/*  6 */
                                    0x1b,               /* FC_CARRAY */
                                    0x1,                /* 1 */
/*  8 */    NdrFcShort( 0x2 ),      /* 2 */
/* 10 */    0x9,                    /* Corr desc: FC_ULONG */
```

```
                                0x0,                    /*  */
/* 12 */        NdrFcShort( 0xfffc ),   /* -4 */
/* 14 */        0x6,                    /* FC_SHORT */
                                0x5b,                   /* FC_END */
/* 16 */
                                0x17,                   /*
FC_CSTRUCT */
                                0x3,                    /* 3 */
/* 18 */        NdrFcShort( 0x8 ),      /* 8 */
/* 20 */        NdrFcShort( 0xfffffff2),/* Offset= -14 (6) */
/* 22 */        0x8,                    /* FC_LONG */
                                0x8,                    /* FC_LONG */
/* 24 */        0x5c,                   /* FC_PAD */
                                0x5b,                   /* FC_END */
/* 26 */        0xb4,                   /* FC_USER_MARSHAL */
                                0x83,                   /* 131 */
/* 28 */        NdrFcShort( 0x0 ),      /* 0 */
/* 30 */        NdrFcShort( 0x4 ),      /* 4 */
/* 32 */        NdrFcShort( 0x0 ),      /* 0 */
/* 34 */        NdrFcShort( 0xffffffe0),/* Offset= -32 (2) */

                                0x0
      }
   };

const CInterfaceProxyVtbl * _ExecuteDll_ProxyVtblList[]=
{
   ( CInterfaceProxyVtbl *) &_IExecuteProxyVtbl,
   0
};

const CInterfaceStubVtbl * _ExecuteDll_StubVtblList[]=
{
   ( CInterfaceStubVtbl *) &_IExecuteStubVtbl,
   0
};

PCInterfaceName const _ExecuteDll_InterfaceNamesList[]=
{
   "IExecute",
   0
};

const IID *  _ExecuteDll_BaseIIDList[]=
{
   &IID_IDispatch,
   0
};


#define _ExecuteDll_CHECK_IID(n) IID_GENERIC_CHECK_IID(
_ExecuteDll, pIID, n)

int __stdcall _ExecuteDll_IID_Lookup( const IID * pIID, int * pIndex )
{

   if(!_ExecuteDll_CHECK_IID(0))
      {
      *pIndex = 0;
      return 1;
      }

   return 0;
}

const ExtendedProxyFileInfo ExecuteDll_ProxyFileInfo=
{
   (PCInterfaceProxyVtblList *) & _ExecuteDll_ProxyVtblList,
   (PCInterfaceStubVtblList *) & _ExecuteDll_StubVtblList,
   (const PCInterfaceName * ) & _ExecuteDll_InterfaceNamesList,
   (const IID ** ) & _ExecuteDll_BaseIIDList,
   & _ExecuteDll_IID_Lookup,
   1,
   2,
   0, /* table of [async_uuid] interfaces */
   0, /* Filler1 */
   0, /* Filler2 */
   0  /* Filler3 */
};
#ifndef _EXECUTEDLLCP_H_
#define _EXECUTEDLLCP_H_
LIBRARY      "ExecuteDllPS"

DESCRIPTION  'Proxy/Stub DLL'

EXPORTS
                DllGetClassObject       @1              PRIVATE
                DllCanUnloadNow         @2              PRIVATE
                GetProxyDllInfo         @3              PRIVATE
                DllRegisterServer                       @4              PRIVATE
                DllUnregisterServer                     @5              PRIVATE




template <class T>
class CProxy_IExecuteEvents : public IConnectionPointImpl<T,
&DIID__IExecuteEvents, CComDynamicUnkArray>
{
                //Warning this class may be recreated by the wizard.
public:
                VOID Fire_Start(CY StartTime)
                {
                                T* pT = static_cast<T*>(this);
                                int nConnectionIndex;
                                CComVariant* pvars = new CComVariant[1];
                                int nConnections = m_vec.GetSize();

                                for (nConnectionIndex = 0; nConnectionIndex <
nConnections; nConnectionIndex++)
                                {
                                                pT->Lock();
                                                CComPtr<IUnknown> sp =
m_vec.GetAt(nConnectionIndex);
                                                pT->Unlock();
                                                IDispatch* pDispatch =
reinterpret_cast<IDispatch*>(sp.p);
                                                if (pDispatch != NULL)
                                                {
                                                                pvars[0] = StartTime;
                                                                DISPPARAMS disp = {
pvars, NULL, 1, 0 };
                                                                pDispatch->Invoke(0x1,
IID_NULL, LOCALE_USER_DEFAULT, DISPATCH_METHOD, &disp,
NULL, NULL, NULL);
                                                }
                                }
                                delete[] pvars;

                }
                VOID Fire_Complete(CY EndTime, LONG Elapsed)
                {
                                T* pT = static_cast<T*>(this);
                                int nConnectionIndex;
                                CComVariant* pvars = new CComVariant[2];
                                int nConnections = m_vec.GetSize();

                                for (nConnectionIndex = 0; nConnectionIndex <
nConnections; nConnectionIndex++)
                                {
                                                pT->Lock();
                                                CComPtr<IUnknown> sp =
m_vec.GetAt(nConnectionIndex);
                                                pT->Unlock();
                                                IDispatch* pDispatch =
reinterpret_cast<IDispatch*>(sp.p);
                                                if (pDispatch != NULL)
                                                {
                                                                pvars[1] = EndTime;
                                                                pvars[0] = Elapsed;
                                                                DISPPARAMS disp = {
pvars, NULL, 2, 0 };
```

```
                                                pDispatch->Invoke(0x2,
IID_NULL, LOCALE_USER_DEFAULT, DISPATCH_METHOD, &disp,
NULL, NULL, NULL);
                                        }
                                }
                                delete[] pvars;

                        }
};
#endif
ExecuteDllps.dll: dlldata.obj ExecuteDll_p.obj ExecuteDll_i.obj
                        link /dll /out:ExecuteDllps.dll /def:ExecuteDllps.def /entry:DllMain
dlldata.obj ExecuteDll_p.obj ExecuteDll_i.obj \
                        kernel32.lib rpcndr.lib rpcns4.lib rpcrt4.lib oleaut32.lib
uuid.lib \

.c.obj:
                        cl /c /Ox /DWIN32 /D_WIN32_WINNT=0x0400
/DREGISTER_PROXY_DLL \
                        $<

clean:
                        @del ExecuteDllps.dll
                        @del ExecuteDllps.lib
                        @del ExecuteDllps.exp
                        @del dlldata.obj
                        @del ExecuteDll_p.obj
                        @del ExecuteDll_i.obj

//{{NO_DEPENDENCIES}}
// Microsoft Developer Studio generated include file.
// Used by ExecuteDll.rc
//
#define IDS_PROJNAME                100
#define IDR_EXECUTE                 101
#define IDR_LOG                     102
#define IDR_EXECUTESHELL            103

#define SM_SYSTEM_ERROR             0x0201

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE    201
#define _APS_NEXT_COMMAND_VALUE     32768
#define _APS_NEXT_CONTROL_VALUE     201
#define _APS_NEXT_SYMED_VALUE       104
#endif
#endif
#pragma once

// ODBC-specific includes
#define DBNTWIN32
#include <sqltypes.h>
#include <sql.h>
#include <sqlext.h>

#define CONNECTION_NAME_LEN 256     // connection name length

typedef struct _SM_Connection_Info
{
        char        szConnectionName[CONNECTION_NAME_LEN];
        HDBC        hdbc;
        BOOL        bInUse;
} SM_Connection_Info;
// stdafx.cpp : source file that includes just the standard includes
//  stdafx.pch will be the pre-compiled header
//  stdafx.obj will contain the pre-compiled type information

#include "stdafx.h"

#ifdef _ATL_STATIC_REGISTRY
#include <statreg.h>
#include <statreg.cpp>
#endif

#include <atlimpl.cpp>
// stdafx.h : include file for standard system include files,
//      or project specific include files that are used frequently,
//      but are changed infrequently

#if
!defined(AFX_STDAFX_H__551AC528_AB1C_11D2_BC0C_00A0C90D2CA5
__INCLUDED_)
#define

AFX_STDAFX_H__551AC528_AB1C_11D2_BC0C_00A0C90D2CA5__INCL
UDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#define STRICT
#ifndef _WIN32_WINNT
#define _WIN32_WINNT 0x0400
#endif
#define _ATL_APARTMENT_THREADED

#include <atlbase.h>
//You may derive a class from CComModule and use it if you want to override
//something, but do not change the name of _Module
extern CComModule _Module;
#include <atlcom.h>

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the
previous line.

#endif //
!defined(AFX_STDAFX_H__551AC528_AB1C_11D2_BC0C_00A0C90D2CA5
__INCLUDED)
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cArrConstraints"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cArrConstraints.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Implements an array of cConstraint objects.
'              Type-safe wrapper around cNodeCollections.
'              Also contains additional functions that determine all the
'              constraints for a step, all constraints in a workspace,
'              validation functions, etc.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mcarrConstraints As cNodeCollections

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cArrConstraints."
Public Sub SaveWspConstraints(ByVal lngWorkspace As Long)
    ' Calls a procedure to commit all changes to the constraints
    ' in the passed in workspace.

    Call mcarrConstraints.Save(lngWorkspace)

End Sub
Public Property Set ConstraintDB(vdata As Database)

    Set mcarrConstraints.NodeDB = vdata

End Property
Public Property Get ConstraintDB() As Database
```

```
    Set ConstraintDB = mcarrConstraints.NodeDB

End Property

Public Sub Modify(cConsToUpdate As cConstraint)

    ' Modify the constraint record
    Call mcarrConstraints.Modify(cConsToUpdate)

End Sub
Public Sub CreateNewConstraintVersion(ByVal lngStepId As Long, _
        ByVal strNewVersion As String, _
        ByVal strOldVersion As String, _
        ByVal intStepType As Integer)

    ' Does all the processing needed to create new versions of
    ' all the constraints for a given step
    ' It inserts new constraint records in the database with
    ' the new version numbers on them
    ' It also updates the version number on all constraints
    ' for the step in the array to the new version passed in
    ' Since it handles both global and manager/worker steps,
    ' it checks for the step_id or global_step_id fields,
    ' depending on the type of step

    Dim lngIndex As Long
    Dim cUpdateConstraint As cConstraint

    On Error GoTo CreateNewConstraintVersionErr
    mstrSource = mstrModuleName & "CreateNewConstraintVersion"

    ' Update the version/global version on Constraint with the
    ' passed in step/global step id
    For lngIndex = 0 To mcarrConstraints.Count - 1
        Set cUpdateConstraint = mcarrConstraints(lngIndex)
        If intStepType = gintGlobalStep Then
            If cUpdateConstraint.GlobalStepId = lngStepId And _
                    cUpdateConstraint.IndOperation <> DeleteOp Then
                cUpdateConstraint.GlobalVersionNo = strNewVersion

                ' Set the operation to indicate an insert
                cUpdateConstraint.IndOperation = InsertOp
            End If
        Else
            If cUpdateConstraint.StepId = lngStepId And _
                    cUpdateConstraint.IndOperation <> DeleteOp Then
                cUpdateConstraint.VersionNo = strNewVersion

                ' Set the operation to indicate an insert
                cUpdateConstraint.IndOperation = InsertOp
            End If
        End If
    Next lngIndex

    Exit Sub

CreateNewConstraintVersionErr:
    LogErrors Errors
    gstrSource = mstrModuleName & "CreateNewConstraintVersion"
    On Error GoTo 0
    Err.Raise vbObjectError + errCreateNewConstraintVersionFailed, _
            mstrSource, _
            LoadResString(errCreateNewConstraintVersionFailed)

End Sub
Private Sub Class_Initialize()

    Set mcarrConstraints = New cNodeCollections
    BugMessage "cArrConstraints: Initialize event - setting Constraint count to 0"

End Sub

Private Sub Class_Terminate()

    Set mcarrConstraints = Nothing
    BugMessage "cArrConstraints: Terminate event triggered"

End Sub

Public Sub Add(ByVal cConstraintToAdd As cConstraint)

    Set cConstraintToAdd.NodeDB = mcarrConstraints.NodeDB

    ' Retrieve a unique constraint identifier
    cConstraintToAdd.ConstraintId = cConstraintToAdd.NextIdentifier

    ' Call a procedure to load the constraint record in the array
    Call mcarrConstraints.Add(cConstraintToAdd)

End Sub
Public Sub Delete(ByVal cOldConstraint As cConstraint)

    Dim lngDeleteElement As Long
    Dim cConsToDelete As cConstraint

    lngDeleteElement = QueryConstraintIndex(cOldConstraint.ConstraintId)
    Set cConsToDelete = mcarrConstraints(lngDeleteElement)

    Call mcarrConstraints.Delete(cConsToDelete.Position)

    Set cConsToDelete = Nothing

End Sub
Private Function QueryConstraintIndex(lngConstraintIdAs Long) _
        As Long

    Dim lngIndex As Integer

    ' Find the element in the array to be deleted
    For lngIndex = 0 To mcarrConstraints.Count - 1

        ' Note: The constraint id is not a primary key field in
        ' the database - there can be multiple records with the
        ' same constraint_id but for different versions of a step
        ' However, since we'll always load the constraint information
        ' for the latest version of a step, we'll have just one
        ' constraint record with a given constraint_id
        If mcarrConstraints(lngIndex).ConstraintId= lngConstraintId Then
            QueryConstraintIndex = lngIndex
            Exit Function
        End If

    Next lngIndex

    ' Raise error that Constraint has not been found
    ShowError errConstraintNotFound
    On Error GoTo 0
    Err.Raise vbObjectError + errConstraintNotFound, mstrSource, _
        LoadResString(errConstraintNotFound)

End Function

Public Function QueryConstraint(ByVal lngConstraintId As Long) _
    As cConstraint

    ' Returns a cConstraint object with the property values
    ' corresponding to the Constraint Identifier, lngConstraintId

    Dim lngQueryElement As Long

    lngQueryElement = QueryConstraintIndex(lngConstraintId)

    ' Set the return value to the queried Constraint
    Set QueryConstraint = mcarrConstraints(lngQueryElement)

End Function

Public Sub LoadConstraints(ByVal lngWorkspaceId As Long, rstStepsInWsp As
Recordset)

    ' Loads the constraints array with all the constraints
    ' for the workspace
    Dim recConstraints As Recordset
    Dim qyCons As DAO.QueryDef
    Dim strSql As String
```

```
    Dim dtStart As Date

    On Error GoTo LoadConstraintsErr
    mstrSource = mstrModuleName & "LoadConstraints"

    If rstStepsInWsp.RecordCount = 0 Then
        Exit Sub
    End If

    ' First check if the database object has been set
    If mcarrConstraints.NodeDB Is Nothing Then
        On Error GoTo 0
        Err.Raise vbObjectError + errSetDBBeforeLoad, _
                mstrSource, _
                LoadResString(errSetDBBeforeLoad)
    End If

    dtStart = Now

    ' Select based on the global step id since there might
    ' be constraints for a global step that run are executed
    ' for the workspace
    ' This method has the advantage that if the steps are queried right, everything
else follows
    strSql = "Select a.constraint_id, a.step_id, a.version_no, " & _
        " a.constraint_type, a.global_step_id, a.global_version_no," & _
        " a.sequence_no, b.workspace_id " & _
        " from step_constraints a, att_steps b " & _
        " where a.global_step_id = b.step_id " & _
        " and a.global_version_no = b.version_no " & _
        " and a.global_step_id = [g_s_id] " & _
        " and a.global_version_no = [g_ver_no] " & _
        " and b.archived_flag = [archived] "

    ' Find the highest X-component of the version number
    strSql = strSql & " AND ( a.step_id = 0 or ( cint( mid( a.version_no, 1, instr(
a.version_no, " & gstrDQ & gstrVerSeparator & gstrDQ & " ) - 1 ) ) = " & _
        " ( select max( cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) ) " & _
        " from att_steps AS d " & _
        " WHERE a.step_id = d.step_id " & _
        " and d.archived_flag = [archived] ) "

    ' Find the highest Y-component of the version number for the highest
X-component
    strSql = strSql & " AND cint( mid( a.version_no, instr( a.version_no, " &
gstrDQ & gstrVerSeparator & gstrDQ & " ) + 1 ) ) = " & _
        " ( select max( cint( mid( version_no, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) + 1 ) ) ) " & _
        " from att_steps AS y " & _
        " Where a.step_id = y.step_id " & _
        " AND cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) = " & _
        " ( select max( cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) ) " & _
        " from att_steps AS c " & _
        " WHERE y.step_id = c.step_id " & _
        " and c.archived_flag = [archived] ) ) ) ) "

    ' Order the constraints by sequence within a given step
    strSql = strSql & " order by a.sequence_no "

    Set qyCons = mcarrConstraints.NodeDB.CreateQueryDef(gstrEmptyString,
strSql)
    qyCons.Parameters("archived").Value = False

    rstStepsInWsp.MoveFirst

    While Not rstStepsInWsp.EOF

        If Not (rstStepsInWsp!global_flag)Then
            qyCons.Close
            BugMessage "Query constraints Read + load took: " & CStr(DateDiff("s",
dtStart, Now))
            Exit Sub
        End If

        qyCons.Parameters("g_s_id").Value = rstStepsInWsp!step_id
        qyCons.Parameters("g_ver_no").Value = rstStepsInWsp!version_no

        Set recConstraints = qyCons.OpenRecordset(dbOpenSnapshot)

        Call LoadRecordsetInConstraintArray(recConstraints)
        recConstraints.Close

        rstStepsInWsp.MoveNext
    Wend

    qyCons.Close
    BugMessage "Query constraints Read + load took: " & CStr(DateDiff("s",
dtStart, Now))

    Exit Sub

LoadConstraintsErr:
    LogErrors Errors
    gstrSource = mstrModuleName & "LoadConstraints"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadDataFailed, _
            mstrSource, _
            LoadResString(errLoadDataFailed)

End Sub
Public Sub UnloadStepConstraints(ByVal lngStepId As Long)

    ' Unloads all the constraints for the workspace from
    ' the constraints array

    Dim lngIndex As Long

    ' Find all constraints in the array with a matching step id
    ' It is important to step in reverse order through the array,
    ' since we delete constraint records!
    For lngIndex = mcarrConstraints.Count - 1 To 0 Step -1
        If mcarrConstraints(lngIndex).GlobalStepId= lngStepId Then

            ' Unload the constraint from the array
            Call mcarrConstraints.Unload(lngIndex)

        End If
    Next lngIndex

End Sub
Public Sub UnloadConstraint(cOldConstraint As cConstraint)
    ' Unloads the constraint from the constraints array

    Dim lngDeleteElement As Long

    lngDeleteElement = QueryConstraintIndex(cOldConstraint.ConstraintId)

    Call mcarrConstraints.Unload(lngDeleteElement)

End Sub
Private Sub LoadRecordsetInConstraintArray(ByVal recConstraints As
Recordset)
    ' Loads all the constraint records in the passed in
    ' recordset into the array

    Dim cNewConstraint As cConstraint

    On Error GoTo LoadRecordsetInConsArrayErr
    mstrSource = mstrModuleName & "LoadRecordsetInConstraintArray"

    If recConstraints.RecordCount = 0 Then
        Exit Sub
    End If

    recConstraints.MoveFirst
    While Not recConstraints.EOF
        Set cNewConstraint = New cConstraint

        ' Initialize Constraint values
        cNewConstraint.ConstraintId = CLng(ErrorOnNullField(recConstraints,
"Constraint_id"))
        cNewConstraint.StepId = CLng(ErrorOnNullField(recConstraints,
"step_id"))
```

```
        cNewConstraint.VersionNo = CStr(ErrorOnNullField(recConstraints,
"version_no"))

        cNewConstraint.GlobalStepId = CLng(ErrorOnNullField(recConstraints,
"global_step_id"))
        cNewConstraint.GlobalVersionNo = CStr(ErrorOnNullField(recConstraints,
"global_version_no"))
        cNewConstraint.SequenceNo = CInt(ErrorOnNullField(recConstraints,
"sequence_no"))

        cNewConstraint.WorkspaceId = CLng(ErrorOnNullField(recConstraints,
FLD_ID_WORKSPACE))
        cNewConstraint.ConstraintType = CInt(ErrorOnNullField(recConstraints,
"constraint_type"))

        ' Add this record to the array of Constraints
        mcarrConstraints.Load cNewConstraint

        Set cNewConstraint = Nothing
        recConstraints.MoveNext
    Wend

    Exit Sub

LoadRecordsetInConsArrayErr:
    LogErrors Errors
    gstrSource = mstrModuleName & "LoadRecordsetInConstraintArray"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadRsInArrayFailed, _
          mstrSource, _
          LoadResString(errLoadRsInArrayFailed)

End Sub

Public Function ConstraintsForStep( _
        ByVal lngStepId As Long, _
        ByVal strVersionNo As String, _
        Optional ByVal intConstraintType As ConstraintType = 0, _
        Optional ByVal blnSort As Boolean = True, _
        Optional ByVal blnGlobal As Boolean = False, _
        Optional ByVal blnGlobalConstraintsOnly As Boolean = False) _
        As Variant

    ' Returns a variant containing an array of cConstraint objects,
    ' containing all the constraints that have been defined for the
    ' given step. If the Global flag is set to true, the
    ' search will be made for all the constraints that have
    ' a matching global_step_id

    Dim lngIndex As Long
    Dim cStepConstraint() As cConstraint
    Dim lngConstraintCount As Long
    Dim cTempConstraint As cConstraint

    On Error GoTo ConstraintsForStepErr
    mstrSource = mstrModuleName & "ConstraintsForStep"

    lngConstraintCount = 0

    ' Find each element in the constraints array
    For lngIndex = 0 To mcarrConstraints.Count - 1
        ' If a constraint type has been specified then check
        ' if the constraint type for the record matches the
        ' passed in type
        Set cTempConstraint = mcarrConstraints(lngIndex)
        If Not blnGlobal Then
            If cTempConstraint.StepId = lngStepId And _
                  cTempConstraint.VersionNo = strVersionNo And _
                  cTempConstraint.IndOperation <> DeleteOp And _
                  (intConstraintType = 0 Or _
                  cTempConstraint.ConstraintType = intConstraintType) Then
                ' We have a matching constraint for the given step
                AddArrayElement cStepConstraint, _
                      cTempConstraint, lngConstraintCount
            End If
        Else
            If cTempConstraint.GlobalStepId = lngStepId And _
                  cTempConstraint.GlobalVersionNo = strVersionNo And _
                  cTempConstraint.IndOperation <> DeleteOp Then
                If blnGlobalConstraintsOnly = False Or _
                      (blnGlobalConstraintsOnly And _
                      cTempConstraint.StepId = 0 And _
                      cTempConstraint.VersionNo = gstrMinVersion) Then

                    ' We have a matching constraint for the global step
                    AddArrayElement cStepConstraint, _
                          cTempConstraint, lngConstraintCount
                End If
            End If
        End If

    Next lngIndex

    ' Set the return value of the function to the array of
    ' constraints that has been built above
    If lngConstraintCount = 0 Then
        ConstraintsForStep = Empty
    Else
        ConstraintsForStep = cStepConstraint()
    End If

    ' Sort the constraints
    If blnSort Then
        Call QuickSort(ConstraintsForStep)
    End If

    Exit Function

ConstraintsForStepErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errConstraintsForStepFailed, _
          mstrSource, _
          LoadResString(errConstraintsForStepFailed)

End Function
Private Sub AddArrayElement(ByRef arrNodes() As cConstraint, _
        ByVal objToAdd As cConstraint, _
        ByRef lngCount As Long)
    ' Adds the passed in object to the array

    ' Increase the array dimension and add the object to it
    ReDim Preserve arrNodes(lngCount)
    Set arrNodes(lngCount) = objToAdd
    lngCount = lngCount + 1

End Sub

Public Function ConstraintsForWsp( _
        ByVal lngWorkspaceId As Long, _
        Optional ByVal intConstraintType As Integer = 0, _
        Optional ByVal blnSort As Boolean = True, _
        Optional ByVal blnGlobalConstraintsOnly As Boolean = False) _
        As Variant

    ' Returns a variant containing an array of cConstraint objects,
    ' containing all the constraints that have been defined for the
    ' given workspace.

    Dim lngIndex As Long
    Dim cWspConstraint() As cConstraint
    Dim lngConstraintCount As Long
    Dim cTempConstraint As cConstraint

    On Error GoTo ConstraintsForWspErr
    mstrSource = mstrModuleName & "ConstraintsForWsp"

    lngConstraintCount = 0

    ' Find each element in the constraints array
    For lngIndex = 0 To mcarrConstraints.Count - 1
        ' If a constraint type has been specified then check
        ' if the constraint type for the record matches the
        ' passed in type
        Set cTempConstraint = mcarrConstraints(lngIndex)
        If cTempConstraint.WorkspaceId = lngWorkspaceId And _
```

```
            cTempConstraint.IndOperation <> DeleteOp And _
            (intConstraintType = 0 Or _
            cTempConstraint.ConstraintType = intConstraintType) Then

          If blnGlobalConstraintsOnly = False Or _
                (blnGlobalConstraintsOnly And _
                cTempConstraint.StepId = 0 And _
                cTempConstraint.VersionNo = gstrMinVersion) Then

              ' We have a matching constraint for the workspace
              AddArrayElement cWspConstraint, _
                    cTempConstraint, lngConstraintCount
          End If
        End If
    Next lngIndex

    ' Set the return value of the function to the array of
    ' constraints that has been built above
    If lngConstraintCount = 0 Then
        ConstraintsForWsp = Empty
    Else
        ConstraintsForWsp = cWspConstraint()
    End If

    ' Sort the constraints
    If blnSort Then
        Call QuickSort(ConstraintsForWsp)
    End If

    Exit Function

ConstraintsForWspErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errConstraintsForWspFailed, _
          mstrSource, _
          LoadResString(errConstraintsForWspFailed)

End Function
Public Function PreConstraintsForStep( _
        ByVal lngStepId As Long, _
        ByVal strVersionNo As String, _
        Optional ByVal blnSort As Boolean) As Variant

    ' Returns a variant containing an array of cConstraint objects,
    ' containing all the pre-execution constraints that have
    ' been defined for the given step_id and version

    ' Call a function that will return a variant containing
    ' all the constraints of the passed in type
    PreConstraintsForStep = ConstraintsForStep(lngStepId, _
          strVersionNo, gintPreStep, blnSort)

End Function
Public Function PostConstraintsForStep( _
        ByVal lngStepId As Long, _
        ByVal strVersionNo As String, _
        Optional ByVal blnSort As Boolean) As Variant

    ' Returns a variant containing an array of cConstraint objects,
    ' containing all the Post-execution constraints that have
    ' been defined for the given step_id and version

    ' Call a function that will return a variant containing
    ' all the constraints of the passed in type
    PostConstraintsForStep = ConstraintsForStep(lngStepId, _
          strVersionNo, gintPostStep, blnSort)

End Function
Public Function PostConstraintsForWsp( _
        ByVal lngWorkspaceId As Long, _
        Optional ByVal blnSort As Boolean) As Variant

    ' Returns a variant containing an array of cConstraint objects,
    ' containing all the Post-execution globals that have
    ' been defined for the workspace

    ' Call a function that will return a variant containing
    ' all the constraints of the passed in type
    PostConstraintsForWsp = ConstraintsForWsp(lngWorkspaceId, _
          gintPostStep, blnSort, True)

End Function
Public Function PreConstraintsForWsp( _
        ByVal lngWorkspaceId As Long, _
        Optional ByVal blnSort As Boolean) As Variant

    ' Returns a variant containing an array of cConstraint objects,
    ' containing all the Pre-execution globals that have
    ' been defined for the workspace

    ' Call a function that will return a variant containing
    ' all the constraints of the passed in type
    PreConstraintsForWsp = ConstraintsForWsp(lngWorkspaceId, _
          gintPreStep, blnSort, True)

End Function
Public Property Get ConstraintCount() As Long

    ConstraintCount = mcarrConstraints.Count

End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cArrParameters"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cArrParameters.cls
'              Microsoft  TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Implements an array of cParameter objects.
'              Type-safe wrapper around cNodeCollections.
'              Also contains additional functions to determine parameter
'              values, validation functions, etc.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mcarrParameters As cNodeCollections

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cArrParameters."

Public Property Set ParamDatabase(vdata As Database)

    Set mcarrParameters.NodeDB = vdata

End Property
Public Sub Modify(cModifiedParam As cParameter)

    ' First check if the parameter record is valid
    Call CheckDupParamName(cModifiedParam)

    Call mcarrParameters.Modify(cModifiedParam)

End Sub
Public Sub Load(ByRef cParamToAdd As cParameter)

    Call mcarrParameters.Load(cParamToAdd)

End Sub
Public Sub Add(ByRef cParamToAdd As cParameter)

    Set cParamToAdd.NodeDB = mcarrParameters.NodeDB

    ' First check if the parameter record is valid
    Call Validate(cParamToAdd)
```

```
    ' Retrieve a unique parameter identifier
    cParamToAdd.ParameterId = cParamToAdd.NextIdentifier

    Call mcarrParameters.Add(cParamToAdd)

End Sub

Public Sub Unload(lngParamToDelete As Long)

    Dim lngDeleteElement As Long

    lngDeleteElement = QueryIndex(lngParamToDelete)

    Call mcarrParameters.Unload(lngDeleteElement)

End Sub

Public Sub SaveParametersInWsp(ByVal lngWorkspace As Long)
    ' Calls a procedure to commit all changes to the parameters
    ' for the passed in workspace.

    ' Call a procedure to save all parameter records for the
    ' workspace
    Call mcarrParameters.Save(lngWorkspace)

End Sub
Public Function GetParameterValue(ByVal lngWorkspace As Long, _
        ByVal strParamName As String) As cParameter
    ' Returns the value for the passed in workspace parameter

    Dim cParamRec As cParameter
    Dim lngIndex As Long

    On Error GoTo GetParameterValueErr

    ' Find all parameters in the array with a matching workspace id
    For lngIndex = 0 To mcarrParameters.Count - 1
        Set cParamRec = mcarrParameters(lngIndex)
        If cParamRec.WorkspaceId = lngWorkspace And _
                cParamRec.ParameterName = strParamName Then

            Set GetParameterValue = cParamRec
            Exit For
        End If
    Next lngIndex

    If lngIndex > mcarrParameters.Count - 1 Then
        ' The parameter has not been defined for the workspace
        ' Raise an error
        On Error GoTo 0
        Err.Raise vbObjectError + errParamNameInvalid, _
                mstrModuleName & "GetParameterValue", _
                LoadResString(errParamNameInvalid)
    End If

    Exit Function

GetParameterValueErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "GetParameterValue"
    On Error GoTo 0
    Err.Raise vbObjectError + errGetParamValueFailed, _
            gstrSource, _
            LoadResString(errGetParamValueFailed)

End Function
Public Sub Delete(lngParamToDelete As Long)
    ' Delete the passed in parameter

    Dim lngDeleteElement As Long

    lngDeleteElement = QueryIndex(lngParamToDelete)
    Call mcarrParameters.Delete(lngDeleteElement)

End Sub
Private Function QueryIndex(lngParameterId As Long) As Long

    Dim lngIndex As Long

    ' Find the matching parameter record in the array
    For lngIndex = 0 To mcarrParameters.Count - 1
        If mcarrParameters(lngIndex).ParameterId= lngParameterId And _
                mcarrParameters(lngIndex).IndOperation<> DeleteOp Then
            QueryIndex = lngIndex
            Exit Function
        End If
    Next lngIndex

    ' Raise error that parameter has not been found
    On Error GoTo 0
    Err.Raise vbObjectError + errParamNotFound, "cArrParameters.QueryIndex",
_
        LoadResString(errParamNotFound)

End Function

Public Function QueryParameter(lngParameterId As Long) _
        As cParameter

    Dim lngQueryElement As Long

    lngQueryElement = QueryIndex(lngParameterId)

    ' Return the queried parameter object
    Set QueryParameter = mcarrParameters(lngQueryElement)

End Function
Public Property Get ParameterCount() As Long

    ParameterCount = mcarrParameters.Count

End Property
Public Property Get Item(lngIndex As Long) As cParameter
Attribute Item.VB_UserMemId = 0

    Set Item = mcarrParameters(lngIndex)

End Property

Public Sub Validate(ByVal cParamToValidate As cParameter)
    ' This procedure is necessary since the class cannot validate
    ' all the parameter properties on it's own. This is 'coz we
    ' might have created new parameters in the workspace, but not
    ' saved them to the database yet - hence the duplicate check
    ' has to be repeated in the array

    Dim lngIndex As Long
    Dim cTempParam As cParameter

    On Error GoTo ValidateErr

    ' Check if the parameter name already exists in the workspace
    For lngIndex = 0 To mcarrParameters.Count - 1
        Set cTempParam = mcarrParameters(lngIndex)
        If cTempParam.WorkspaceId = cParamToValidate.WorkspaceId And _
                cTempParam.ParameterName = cParamToValidate.ParameterName
And _
                cTempParam.IndOperation <> DeleteOp Then
            On Error GoTo 0
            Err.Raise vbObjectError + errDuplicateParameterName, _
                mstrSource, LoadResString(errDuplicateParameterName)
        End If
    Next lngIndex

    Exit Sub

ValidateErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Validate"
    On Error GoTo 0
    Err.Raise vbObjectError + errValidateFailed, _
        mstrSource, LoadResString(errValidateFailed)

End Sub
```

```
Public Sub CheckDupParamName(ByVal cParamToValidate As cParameter)

   Dim lngIndex As Long
   Dim cTempParam As cParameter

   ' Check if the parameter name already exists in the workspace
   For lngIndex = 0 To mcarrParameters.Count - 1
      Set cTempParam = mcarrParameters(lngIndex)
      If cTempParam.WorkspaceId = cParamToValidate.WorkspaceId And _
            cTempParam.ParameterName = cParamToValidate.ParameterName
And _
            cTempParam.ParameterId <> cParamToValidate.ParameterId And _
            cTempParam.IndOperation <> DeleteOp Then
         ShowError errDuplicateParameterName
         On Error GoTo 0
         Err.Raise vbObjectError + errDuplicateParameterName, _
            mstrSource, LoadResString(errDuplicateParameterName)
      End If
   Next lngIndex

End Sub

Private Sub Class_Initialize()

   'bugmessage "cArrParameters: Initialize event - setting parameter count to 0"
   Set mcarrParameters = New cNodeCollections

End Sub


Private Sub Class_Terminate()

   Set mcarrParameters = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cArrSteps"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:      cArrSteps.cls
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:    Implements an array of cStep objects.
'             Type-safe wrapper around cNodeCollections.
'             Also contains additional functions to update parent version
'             on substeps, validation functions, etc.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mcarrSteps As cNodeCollections

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cArrSteps."

Public Sub Unload(lngStepToDelete As Long)

   Dim lngDeleteElement As Long
   Dim cUnloadStep As cStep

   lngDeleteElement = QueryStepIndex(lngStepToDelete)
   Set cUnloadStep = QueryStep(lngStepToDelete)

   ' First unload all iterators for the step
   Call cUnloadStep.UnloadIterators

   ' Unload the step from the collection
   Call mcarrSteps.Unload(lngDeleteElement)

End Sub
Public Sub Modify(cModifiedStep As cStep)

   Validate cModifiedStep

   Call mcarrSteps.Modify(cModifiedStep)

End Sub
Public Sub UpdateParentVersion(ByVal lngStepId As Long, _
      ByVal strNewVersion As String, _
      ByVal strOldVersion As String, _
      ByVal intStepType As Integer)

   ' Does all the processing needed to update the parent version
   ' number on all the sub-steps for a given step
   ' It updates the parent version no in the database for all
   ' sub-steps of the passed in step id
   ' It also updates the parent version number on all sub-steps
   ' in the array to the new version passed in

   Dim lngIndex As Long
   Dim cUpdateStep As cStep

   On Error GoTo UpdateParentVersionErr

   If intStepType <> gintManagerStep Then
      ' Only a manager can have sub-steps - if the passed
      ' in step is not a manager, exit
      Exit Sub
   End If

   ' For all steps in the array
   For lngIndex = 0 To mcarrSteps.Count - 1

      Set cUpdateStep = mcarrSteps(lngIndex)

      ' If the current step is a sub-step of the passed in step
      If cUpdateStep.ParentStepId = lngStepId And _
            cUpdateStep.ParentVersionNo = strOldVersion And _
            Not cUpdateStep.ArchivedFlag Then

         ' Update the parent version number for the sub-step
         ' in the array
         cUpdateStep.ParentVersionNo = strNewVersion

         ' Update the parent version number for the sub-step
         ' in the array
         Call Modify(cUpdateStep)

      End If
   Next lngIndex

   Exit Sub

UpdateParentVersionErr:
   LogErrors Errors
   mstrSource = mstrModuleName & "UpdateParentVersion"
   On Error GoTo 0
   Err.Raise vbObjectError + errUpdateParentVersionFailed, _
         mstrSource, _
         LoadResString(errUpdateParentVersionFailed)

End Sub
Private Sub Validate(cCheckStep As cStep)

   ' Step validations that depend on other steps in the collection

   Dim lngIndex As Long

   ' Ensure that the step label is unique in the workspace
   For lngIndex = 0 To mcarrSteps.Count - 1

      ' If the current step is a sub-step of the passed in step
      If mcarrSteps(lngIndex).WorkspaceId = cCheckStep.WorkspaceId And _
            mcarrSteps(lngIndex).StepLabel = cCheckStep.StepLabel And _
            mcarrSteps(lngIndex).StepId <> cCheckStep.StepId Then
         ShowError errStepLabelUnique
```

```
            On Error GoTo 0
            Err.Raise vbObjectError + errValidateFailed, _
                mstrModuleName & "Validate", _
                LoadResString(errValidateFailed)
        End If
    Next lngIndex

End Sub

Private Sub Class_Initialize()

    BugMessage "cArrSteps: Initialize event - setting step count to 0"
    Set mcarrSteps = New cNodeCollections

End Sub

Private Sub Class_Terminate()

    BugMessage "cArrSteps: Terminate event triggered"
    Set mcarrSteps = Nothing

End Sub

Public Sub Add(ByVal cStepToAdd As cStep)

    Validate cStepToAdd

    Set cStepToAdd.NodeDB = mcarrSteps.NodeDB

    ' Retrieve a unique step identifier
    cStepToAdd.StepId = cStepToAdd.NextStepId

    ' Call a procedure to add the step record
    Call mcarrSteps.Add(cStepToAdd)

End Sub
Public Sub Load(cStepToLoad As cStep)

    Call mcarrSteps.Load(cStepToLoad)

End Sub
Public Sub SaveStepsInWsp(ByVal lngWorkspace As Long)
    ' Calls a procedure to commit all changes to the steps
    ' in the passed in workspace.

    Dim lngIndex As Integer

    ' Find all steps in the array with a matching workspace id
    ' It is important to step in reverse order through the array,
    ' since we delete step records sometimes!
    For lngIndex = mcarrSteps.Count - 1 To 0 Step -1
        If mcarrSteps(lngIndex).WorkspaceId = lngWorkspace Then

            ' Call a procedure to commit all changes to the
            ' Step record, if any
            Call CommitStep(mcarrSteps(lngIndex),lngIndex)

        End If
    Next lngIndex

End Sub
Private Sub CommitStep(ByVal cCommitStep As cStep, _
        ByVal intIndex As Integer)
    ' This procedure checks if any changes have been made to the
    ' passed in Step. If so, it calls the step methods to commit
    ' the changes.

    ' First commit all changes to the iterator records for
    ' the step
    cCommitStep.SaveIterators

    Call mcarrSteps.Commit(cCommitStep, intIndex)

End Sub
Public Sub Delete(lngStepToDelete As Long)

    Dim lngDeleteElement As Long

    lngDeleteElement = QueryStepIndex(lngStepToDelete)
    Call mcarrSteps.Delete(lngDeleteElement)

End Sub
Public Function QueryStepIndex(lngStepId As Long) As Long

    Dim lngIndex As Long

    ' Find the element in the array that corresponds to the
    ' passed in step id - note that while there will be multiple
    ' versions of a step in the database, only one version will
    ' be currently loaded in the array - meaning that the stepid
    ' is enough to uniquely identify a step
    For lngIndex = 0 To mcarrSteps.Count - 1
        If mcarrSteps(lngIndex).StepId = lngStepId Then
            QueryStepIndex = lngIndex
            Exit Function
        End If
    Next lngIndex

    ' Raise error that step has not been found
    On Error GoTo 0
    Err.Raise vbObjectError + errStepNotFound, mstrSource, _
        LoadResString(errStepNotFound)

End Function

Public Function QueryStep(ByVal lngStepId As Long) As cStep

    ' Populates the passed in cStep object with the property
    ' values corresponding to the Step Identifier, lngStepId

    Dim lngQueryElement As Integer

    lngQueryElement = QueryStepIndex(lngStepId)

    ' Initialize the passed in step object to the queried step
    Set QueryStep = mcarrSteps(lngQueryElement)

End Function
Public Property Get Item(ByVal Position As Long) As cStep
Attribute Item.VB_UserMemId = 0

    ' Returns the element at the passed in position in the array
    If Position >= 0 And Position < mcarrSteps.Count Then
        Set Item = mcarrSteps(Position)
    Else
        On Error GoTo 0
        Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
            LoadResString(errItemDoesNotExist)
    End If

End Property
Public Property Set Item(ByVal Position As Long, _
        ByVal cStepRec As cStep)

    ' Returns the element at the passed in position in the array
    If Position >= 0 And Position < mcarrSteps.Count Then
        Set mcarrSteps(Position) = cStepRec
    Else
        On Error GoTo 0
        Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
            LoadResString(errItemDoesNotExist)
    End If

End Property
Public Property Set StepDB(vdata As Database)

    Set mcarrSteps.NodeDB = vdata

End Property
Public Function SubSteps(ByVal lngStepId As Long, _
        ByVal strVersionNo As String) As Variant

    ' Returns a variant containing an array of all the substeps
    ' for the passed in step

    Dim intIndex As Integer
```

```
    Dim cSubSteps() As cStep
    Dim lngStepCount As Long
    Dim cQueryStep As cStep

    On Error GoTo SubStepsErr

    lngStepCount = 0

    Set cQueryStep = QueryStep(lngStepId)

    ' Only a manager can have sub-steps
    If cQueryStep.StepType = gintManagerStep Then

        ' For each element in the Steps array
        For intIndex = 0 To mcarrSteps.Count - 1
            ' Check if the parent step id and parent version number
            ' match the passed in step
            If mcarrSteps(intIndex).ParentStepId = lngStepId And _
                    mcarrSteps(intIndex).ParentVersionNo = strVersionNo And _
                    mcarrSteps(intIndex).IndOperation <> DeleteOp Then

                ' Increase the array dimension and add the step
                ' to it
                ReDim Preserve cSubSteps(lngStepCount)
                Set cSubSteps(lngStepCount) = mcarrSteps(intIndex)
                lngStepCount = lngStepCount + 1

            End If
        Next intIndex

    End If

    ' Set the return value of the function to the array of
    ' Steps that has been built above
    If lngStepCount = 0 Then
        SubSteps = Empty
    Else
        SubSteps = cSubSteps()
    End If

    Exit Function

SubStepsErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "SubSteps"
    On Error GoTo 0
    Err.Raise vbObjectError + errSubStepsFailed, _
            mstrSource, _
            LoadResString(errSubStepsFailed)

End Function

Public Property Get StepCount() As Integer

    StepCount = mcarrSteps.Count

End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cConnDtl"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cConnDtl.cls
'              Microsoft  TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:     Encapsulates the properties and methods of a connection.
'              Contains functions to insert, update and delete
'              connection_dtls records from the database.
'  Contact:     Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit
Option Base 0

' Local variable(s) to hold property value(s)
Public WorkspaceId As Long
Public ConnNameId As Long
Public ConnName As String
Public ConnectionString As String
Public ConnType As ConnectionType
Public Position As Long
Public NodeDB As Database

Private mintOperation As Operation

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cConnDtl."

' The cSequence class is used to generate unique Connection identifiers
Private mConnectionSeq As cSequence

' The StringSM class is used to carry out string operations
Private mFieldValue As cStringSM

Private Sub AssignParameters(qyExec As DAO.QueryDef)
    ' Assigns values to the parameters in the querydef object
    ' The parameter names are cryptic to differentiate them from the field names.
    ' When the parameter names are the same as the field names, parameters in the
where
    ' clause do not get created.

    Dim prmParam As DAO.Parameter

    On Error GoTo AssignParametersErr

    For Each prmParam In qyExec.Parameters
        Select Case prmParam.Name
            Case "[w_id]"
                prmParam.Value = WorkspaceId

            Case "[c_id]"
                prmParam.Value = ConnNameId

            Case "[c_name]"
                prmParam.Value = ConnName

            Case "[c_str]"
                prmParam.Value = ConnectionString

            Case "[c_type]"
                prmParam.Value = ConnType

            Case Else
                ' Write the parameter name that is faulty
                WriteError errInvalidParameter, mstrSource, prmParam.Name
                On Error GoTo 0
                Err.Raise errInvalidParameter, mstrModuleName &
"AssignParameters", _
                        LoadResString(errInvalidParameter)
        End Select
    Next prmParam

    Exit Sub

AssignParametersErr:

    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errAssignParametersFailed, _
        mstrModuleName & "AssignParameters",
LoadResString(errAssignParametersFailed)

End Sub

Public Function Clone() As cConnDtl

    ' Creates a copy of a given Connection
```

```
    Dim cCloneConn As cConnDtl

    On Error GoTo CloneErr

    Set cCloneConn = New cConnDtl

    ' Copy all the Connection properties to the newly created Connection
    cCloneConn.WorkspaceId = WorkspaceId
    cCloneConn.ConnNameId = ConnNameId
    cCloneConn.ConnName = ConnName
    cCloneConn.ConnectionString = ConnectionString
    cCloneConn.ConnType = ConnType
    cCloneConn.IndOperation = mintOperation
    cCloneConn.Position = Position

    ' And set the return value to the newly created Connection
    Set Clone = cCloneConn
    Set cCloneConn = Nothing

    Exit Function

CloneErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Clone"
    On Error GoTo 0
    Err.Raise vbObjectError + errCloneFailed, mstrSource,
LoadResString(errCloneFailed)

End Function
Private Sub CheckDupConnectionName()
    ' Check if the Connection name already exists in the workspace

    Dim rstConnection As Recordset
    Dim strSql As String
    Dim qy As DAO.QueryDef

    On Error GoTo CheckDupConnectionNameErr
    mstrSource = mstrModuleName & "CheckDupConnectionName"

    ' Create a recordset object to retrieve the count of all Connections
    ' for the workspace with the same name
    strSql = "Select count(*) as Connection_count " & _
        " from " & TBL_CONNECTION_DTLS & _
        " where " & FLD_ID_WORKSPACE & " = [w_id]" & _
        " and " & FLD_CONN_DTL_CONNECTION_NAME & " = [c_name]" &
_
        " and " & FLD_ID_CONN_NAME & " <> [c_id]"

    Set qy = dbsAttTool.CreateQueryDef(gstrEmptyString,strSql)
    Call AssignParameters(qy)

    Set rstConnection = qy.OpenRecordset(dbOpenForwardOnly)

    If rstConnection![Connection_count]> 0 Then
        rstConnection.Close
        qy.Close
        ShowError errDupConnDtlName
        On Error GoTo 0
        Err.Raise vbObjectError + errDupConnDtlName, _
            mstrSource, LoadResString(errDupConnDtlName)
    End If

    rstConnection.Close
    qy.Close

    Exit Sub

CheckDupConnectionNameErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "CheckDupConnectionName"
    On Error GoTo 0
    Err.Raise vbObjectError + errProgramError, _
        mstrSource, LoadResString(errProgramError)

End Sub
Public Property Let IndOperation(ByVal vdata As Operation)

    ' The valid operations are define in the cOperations
    ' class. Check if the operation is valid
    Select Case vdata
        Case QueryOp, InsertOp, UpdateOp, DeleteOp
            mintOperation = vdata

        Case Else
            BugAssert True
    End Select

End Property
Public Sub Validate()
    ' Each distinct object will have a Validate method which
    ' will check if the class properties are valid. This method
    ' will be used to check interdependant properties that
    ' cannot be validated by the let procedures.
    ' It should be called by the add and modify methods of the class

    If ConnName = gstrEmptyString Then

        ShowError errConnectionNameMandatory
        On Error GoTo 0
        ' Propogate this error back to the caller
        Err.Raise vbObjectError + errConnectionNameMandatory, _
            mstrSource, LoadResString(errConnectionNameMandatory)
    End If

    ' Raise an error if the Connection name already exists in the workspace
    Call CheckDupConnectionName

End Sub
Public Sub Add()

    Dim strInsert As String
    Dim qy As DAO.QueryDef

    On Error GoTo AddErr

    ' Validate the record before trying to insert the record
    Call Validate

    ' Create a temporary querydef object
    strInsert = "insert into " & TBL_CONNECTION_DTLS & _
            "( " & FLD_ID_WORKSPACE & _
            ", " & FLD_ID_CONN_NAME & _
            ", " & FLD_CONN_DTL_CONNECTION_NAME & _
            ", " & FLD_CONN_DTL_CONNECTION_STRING & _
            ", " & FLD_CONN_DTL_CONNECTION_TYPE & " ) " & _
            " values ( [w_id], [c_id], " & _
            " [c_name], [c_str], [c_type] ) "

    Set qy = dbsAttTool.CreateQueryDef(gstrEmptyString,strInsert)

    ' Call a procedure to assign the Connection values
    Call AssignParameters(qy)

    qy.Execute dbFailOnError
    qy.Close

    Exit Sub

AddErr:

    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errInsertFailed, _
        mstrModuleName & "Add", LoadResString(errInsertFailed)

End Sub
Public Sub Delete()

    Dim strDelete As String
    Dim qy As DAO.QueryDef

    On Error GoTo DeleteErr

    strDelete = "delete from " & TBL_CONNECTION_DTLS & _
            " where " & FLD_ID_CONN_NAME & " = [c_id]"
    Set qy = dbsAttTool.CreateQueryDef(gstrEmptyString,strDelete)
```

```
    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    qy.Close

    Exit Sub

DeleteErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteFailed, _
          mstrModuleName & "Delete", LoadResString(errDeleteFailed)

End Sub

Public Sub Modify()

    Dim strUpdate As String
    Dim qy As QueryDef

    On Error GoTo ModifyErr

    ' Validate the updated values before trying to modify thedb
    Call Validate

    ' Create a temporary querydef object with the modify string
    strUpdate = "update " & TBL_CONNECTION_DTLS & _
          " set " & FLD_ID_WORKSPACE & " = [w_id], " & _
          FLD_CONN_DTL_CONNECTION_NAME & " = [c_name], " & _
          FLD_CONN_DTL_CONNECTION_STRING & " = [c_str], " & _
          FLD_CONN_DTL_CONNECTION_TYPE & " = [c_type] " & _
          " where " & FLD_ID_CONN_NAME & " = [c_id]"
    Set qy = dbsAttTool.CreateQueryDef(gstrEmptyString,strUpdate)

    ' Call a procedure to assign the Connection values to the
    ' querydef object
    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    qy.Close

    Exit Sub

ModifyErr:

    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errModifyFailed, _
       mstrModuleName & "Modify", LoadResString(errModifyFailed)

End Sub
Public Property Get NextIdentifier() As Long

    Dim lngNextId As Long

    On Error GoTo NextIdentifierErr

    ' Retrieve the next identifier using the sequence class
    Set mConnectionSeq = New cSequence
    Set mConnectionSeq.IdDatabase = dbsAttTool
    mConnectionSeq.IdentifierColumn= FLD_ID_CONN_NAME
    lngNextId = mConnectionSeq.Identifier
    Set mConnectionSeq = Nothing

    NextIdentifier = lngNextId
    Exit Property

NextIdentifierErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errIdGetFailed, _
       mstrModuleName & "NextIdentifier", LoadResString(errIdGetFailed)

End Property
Public Property Get IndOperation() As Operation

    IndOperation = mintOperation

End Property

Private Sub Class_Initialize()

    Set mFieldValue = New cStringSM

    ' Initialize the operation indicator variable to Query
    ' It will be modified later by the collection class when
    ' inserts, updates or deletes are performed
    mintOperation = QueryOp

    ConnType = giDefaultConnType

End Sub

Private Sub Class_Terminate()

    Set mFieldValue = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cConnDtls"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cConnDtls.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Implements an array of cConnDtl objects.
'              Type-safe wrapper around cNodeCollections.
'              Also contains additional functions to determine the connection
'              string value, validation functions, etc.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mcarrConnDtls As cNodeCollections

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cConnDtls."

Public Property Set ConnDb(vdata As Database)

    Set mcarrConnDtls.NodeDB = vdata

End Property
Public Sub Modify(cModifiedConn As cConnDtl)

    ' First check if the parameter record is valid
    Call CheckDupConnName(cModifiedConn)

    Call mcarrConnDtls.Modify(cModifiedConn)

End Sub
Public Sub Load(ByRef cConnToAdd As cConnDtl)

    Call mcarrConnDtls.Load(cConnToAdd)

End Sub
Public Sub Add(ByRef cConnToAdd As cConnDtl)

    ' First check if the record is valid
    Call Validate(cConnToAdd)

    ' Retrieve a unique identifier
    cConnToAdd.ConnNameId = cConnToAdd.NextIdentifier

    Call mcarrConnDtls.Add(cConnToAdd)
```

```
End Sub

Public Sub Unload(lConnNameId As Long)

    Dim lngDeleteElement As Long

    lngDeleteElement = QueryIndex(lConnNameId)

    Call mcarrConnDtls.Unload(lngDeleteElement)

End Sub

Public Sub SaveConnDtlsInWsp(ByVal lngWorkspace As Long)
    ' Call a procedure to save all connection details records for the workspace
    Call mcarrConnDtls.Save(lngWorkspace)

End Sub
Public Function GetConnectionDtl(ByVal lngWorkspace As Long, _
        ByVal strConnectionName As String) As cConnDtl
    ' Returns the connection dtl for the passed in connection name

    Dim lngIndex As Long

    ' Find all parameters in the array with a matching workspace id
    For lngIndex = 0 To mcarrConnDtls.Count - 1
        If mcarrConnDtls(lngIndex).WorkspaceId = lngWorkspace And _
                mcarrConnDtls(lngIndex).ConnName = strConnectionName Then

            Set GetConnectionDtl = mcarrConnDtls(lngIndex)
            Exit For
        End If
    Next lngIndex

    If lngIndex > mcarrConnDtls.Count - 1 Then
        ' The parameter has not been defined for the workspace
        ' Raise an error
        On Error GoTo 0
        Err.Raise vbObjectError + errConnNameInvalid, mstrModuleName &
"GetConnection", _
                LoadResString(errConnNameInvalid)
    End If

End Function
Public Sub Delete(lConnNameId As Long)
    ' Delete the passed in parameter

    Dim lngDeleteElement As Long

    lngDeleteElement = QueryIndex(lConnNameId)
    Call mcarrConnDtls.Delete(lngDeleteElement)

End Sub
Private Function QueryIndex(lConnNameId As Long) As Long

    Dim lngIndex As Long

    ' Find the matching parameter record in the array
    For lngIndex = 0 To mcarrConnDtls.Count - 1
        If mcarrConnDtls(lngIndex).ConnNameId = lConnNameId And _
                mcarrConnDtls(lngIndex).IndOperation <> DeleteOp Then
            QueryIndex = lngIndex
            Exit Function
        End If
    Next lngIndex

    ' Raise error that parameter has not been found
    On Error GoTo 0
    Err.Raise vbObjectError + errQueryIndexFailed,
"cArrParameters.QueryIndex", _
        LoadResString(errQueryIndexFailed)

End Function

Public Function QueryConnDtl(lConnNameId As Long) As cConnDtl

    Dim lngQueryElement As Long

    lngQueryElement = QueryIndex(lConnNameId)

    ' Return the queried connection object
    Set QueryConnDtl = mcarrConnDtls(lngQueryElement)

End Function
Public Property Get Count() As Long

    Count = mcarrConnDtls.Count

End Property
Public Property Get Item(lngIndex As Long) As cConnDtl
Attribute Item.VB_UserMemId = 0

    Set Item = mcarrConnDtls(lngIndex)

End Property

Private Sub Validate(ByVal cConnToValidate As cConnDtl)
    ' This procedure is necessary since the class cannot validate
    ' all the connection_dtl properties on it's own. This is 'coz we
    ' might have created new connections in the workspace, but not
    ' saved them to the database yet - hence the duplicate check
    ' has to be repeated in the array

    Dim lngIndex As Long
    Dim cTempParam As cConnDtl

    ' Check if the parameter name already exists in the workspace
    For lngIndex = 0 To mcarrConnDtls.Count - 1
        Set cTempParam = mcarrConnDtls(lngIndex)
        If cTempParam.WorkspaceId = cConnToValidate.WorkspaceId And _
                cTempParam.ConnName = cConnToValidate.ConnName And _
                cTempParam.IndOperation <> DeleteOp Then
            On Error GoTo 0
            Err.Raise vbObjectError + errDupConnDtlName, _
                mstrSource, LoadResString(errDupConnDtlName)
        End If
    Next lngIndex

End Sub
Private Sub CheckDupConnName(ByVal cConnToValidate As cConnDtl)

    Dim lngIndex As Long
    Dim cTempParam As cConnDtl

    ' Check if the parameter name already exists in the workspace
    For lngIndex = 0 To mcarrConnDtls.Count - 1
        Set cTempParam = mcarrConnDtls(lngIndex)
        If cTempParam.WorkspaceId = cConnToValidate.WorkspaceId And _
                cTempParam.ConnName = cConnToValidate.ConnName And _
                cTempParam.ConnNameId <> cConnToValidate.ConnNameId And _
                cTempParam.IndOperation <> DeleteOp Then
            ShowError errDupConnDtlName
            On Error GoTo 0
            Err.Raise vbObjectError + errDupConnDtlName, _
                mstrSource, LoadResString(errDupConnDtlName)
        End If
    Next lngIndex

End Sub

Private Sub Class_Initialize()

    Set mcarrConnDtls = New cNodeCollections

End Sub

Private Sub Class_Terminate()

    Set mcarrConnDtls = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cConnection"
```

```
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cConnection.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Encapsulates the properties and methods of a connection string.
'              Contains functions to insert, update and delete
'              workspace_connections records from the database.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit
Option Base 0

' Local variable(s) to hold property value(s)
Public WorkspaceId As Long
Public ConnectionId As Long
Public ConnectionValue As String
Public Description As String
Public NodeDB As Database
Public Position As Long
Public NoCountDisplay As Boolean
Public NoExecute As Boolean
Public ParseQueryOnly As Boolean
Public QuotedIdentifiers As Boolean
Public AnsiNulls As Boolean
Public ShowQueryPlan As Boolean
Public ShowStatsTime As Boolean
Public ShowStatsIO As Boolean
Public ParseOdbcMsg As Boolean
Public RowCount As Long
Public TsqlBatchSeparator As String
Public QueryTimeOut As Long
Public ServerLanguage As String
Public CharacterTranslation As Boolean
Public RegionalSettings As Boolean

Private mstrConnectionName As String
Private mintOperation As Operation

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cConnection."

' The cSequence class is used to generate unique Connection identifiers
Private mConnectionSeq As cSequence

' The StringSM class is used to carry out string operations
Private mFieldValue As cStringSM

Private Sub AssignParameters(qyExec As DAO.QueryDef)
    ' Assigns values to the parameters in the querydef object
    ' The parameter names are cryptic to differentiate them from the field names.
    ' When the parameter names are the same as the field names, parameters in the
where
    ' clause do not get created.

    Dim prmParam As DAO.Parameter

    On Error GoTo AssignParametersErr

    For Each prmParam In qyExec.Parameters
        Select Case prmParam.Name
            Case "[w_id]"
                prmParam.Value = WorkspaceId

            Case "[c_id]"
                prmParam.Value = ConnectionId

            Case "[c_name]"
                prmParam.Value = mstrConnectionName

            Case "[c_value]"
                prmParam.Value = ConnectionValue

            Case "[desc]"
                prmParam.Value = Description

            Case "[no_count]"
                prmParam.Value = NoCountDisplay

            Case "[no_exec]"
                prmParam.Value = NoExecute

            Case "[parse_only]"
                prmParam.Value = ParseQueryOnly

            Case "[quoted_id]"
                prmParam.Value = QuotedIdentifiers

            Case "[a_nulls]"
                prmParam.Value = AnsiNulls

            Case "[show_qp]"
                prmParam.Value = ShowQueryPlan

            Case "[stats_tm]"
                prmParam.Value = ShowStatsTime

            Case "[stats_io]"
                prmParam.Value = ShowStatsIO

            Case "[parse_odbc]"
                prmParam.Value = ParseOdbcMsg

            Case "[row_cnt]"
                prmParam.Value = RowCount

            Case "[batch_sep]"
                prmParam.Value = TsqlBatchSeparator

            Case "[qry_tmout]"
                prmParam.Value = QueryTimeOut

            Case "[lang]"
                prmParam.Value = ServerLanguage

            Case "[char_trans]"
                prmParam.Value = CharacterTranslation

            Case "[reg_settings]"
                prmParam.Value = RegionalSettings

            Case Else
                ' Write the parameter name that is faulty
                WriteError errInvalidParameter, mstrSource, prmParam.Name
                On Error GoTo 0
                Err.Raise errInvalidParameter, mstrModuleName &
"AssignParameters", _
                    LoadResString(errInvalidParameter)
        End Select
    Next prmParam

    Exit Sub

AssignParametersErr:

    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errAssignParametersFailed, _
        mstrModuleName & "AssignParameters",
LoadResString(errAssignParametersFailed)

End Sub

Public Function Clone() As cConnection

    ' Creates a copy of a given Connection

    Dim cCloneConn As cConnection
```

```
    On Error GoTo CloneErr

    Set cCloneConn = New cConnection

    ' Copy all the Connection properties to the newly
    ' created Connection
    Set cCloneConn.NodeDB = NodeDB
    cCloneConn.WorkspaceId = WorkspaceId
    cCloneConn.ConnectionId = ConnectionId
    cCloneConn.ConnectionName = mstrConnectionName
    cCloneConn.ConnectionValue = ConnectionValue
    cCloneConn.Description = Description
    cCloneConn.IndOperation = mintOperation
    cCloneConn.Position = Position
    cCloneConn.NoCountDisplay = NoCountDisplay
    cCloneConn.NoExecute = NoExecute
    cCloneConn.ParseQueryOnly = ParseQueryOnly
    cCloneConn.QuotedIdentifiers = QuotedIdentifiers
    cCloneConn.AnsiNulls = AnsiNulls
    cCloneConn.ShowQueryPlan = ShowQueryPlan
    cCloneConn.ShowStatsTime = ShowStatsTime
    cCloneConn.ShowStatsIO = ShowStatsIO
    cCloneConn.ParseOdbcMsg = ParseOdbcMsg
    cCloneConn.RowCount = RowCount
    cCloneConn.TsqlBatchSeparator = TsqlBatchSeparator
    cCloneConn.QueryTimeOut = QueryTimeOut
    cCloneConn.ServerLanguage = ServerLanguage
    cCloneConn.CharacterTranslation = CharacterTranslation
    cCloneConn.RegionalSettings = RegionalSettings

    ' And set the return value to the newly created Connection
    Set Clone = cCloneConn
    Set cCloneConn = Nothing

    Exit Function

CloneErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Clone"
    On Error GoTo 0
    Err.Raise vbObjectError + errCloneFailed, mstrSource,
LoadResString(errCloneFailed)

End Function
Private Sub CheckDupConnectionName()
    ' Check if the Connection name already exists in the workspace

    Dim rstConnection As Recordset
    Dim strSql As String
    Dim qy As DAO.QueryDef

    On Error GoTo CheckDupConnectionNameErr
    mstrSource = mstrModuleName & "CheckDupConnectionName"

    ' Create a recordset object to retrieve the count of all Connections
    ' for the workspace with the same name
    strSql = "Select count(*) as Connection_count " & _
        " from workspace_connections " & _
        " where workspace_id = [w_id]" & _
        " and connection_name = [c_name]" & _
        " and connection_id <> [c_id]"

    Set qy = NodeDB.CreateQueryDef(gstrEmptyString, strSql)
    Call AssignParameters(qy)

    Set rstConnection = qy.OpenRecordset(dbOpenForwardOnly)

    If rstConnection![Connection_count]> 0 Then
        rstConnection.Close
        qy.Close
        ShowError errDuplicateConnectionName
        On Error GoTo 0
        Err.Raise vbObjectError + errDuplicateConnectionName, _
            mstrSource, LoadResString(errDuplicateConnectionName)
    End If

    rstConnection.Close
    qy.Close

    Exit Sub

CheckDupConnectionNameErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "CheckDupConnectionName"
    On Error GoTo 0
    Err.Raise vbObjectError + errProgramError, _
        mstrSource, LoadResString(errProgramError)

End Sub
Private Sub CheckDB()
    ' Check if the database object has been initialized

    If NodeDB Is Nothing Then
        On Error GoTo 0
        Err.Raise vbObjectError + errInvalidDB, _
            mstrModuleName & "CheckDB", LoadResString(errInvalidDB)
    End If

End Sub
Public Property Let ConnectionName(vdata As String)

    If vdata = gstrEmptyString Then

        ShowError errConnectionNameMandatory
        On Error GoTo 0
        ' Propogate this error back to the caller
        Err.Raise vbObjectError + errConnectionNameMandatory, _
            mstrSource, LoadResString(errConnectionNameMandatory)
    Else
        mstrConnectionName = vdata
    End If

End Property

Public Property Let IndOperation(ByVal vdata As Operation)

    ' The valid operations are define in the cOperations
    ' class. Check if the operation is valid
    Select Case vdata
        Case QueryOp, InsertOp, UpdateOp, DeleteOp
            mintOperation = vdata

        Case Else
            BugAssert True
    End Select

End Property
Public Sub Validate()
    ' Each distinct object will have a Validate method which
    ' will check if the class properties are valid. This method
    ' will be used to check interdependant properties that
    ' cannot be validated by the let procedures.
    ' It should be called by the add and modify methods of the class

    ' Check if the db object is valid
    Call CheckDB

    ' Raise an error if the Connection name already exists in the workspace
    Call CheckDupConnectionName

End Sub
Public Sub Add()

    Dim strInsert As String
    Dim qy As DAO.QueryDef

    On Error GoTo AddErr

    ' Validate the record before trying to insert the record
    Call Validate

    ' Create a temporary querydef object
    strInsert = "insert into workspace_connections " & _
            "( workspace_id, connection_id, " & _
            "connection_name, connection_value, " & _
            "description, no_count_display, " & _
```

```
            "no_execute, parse_query_only, " & _
            "ANSI_quoted_identifiers, ANSI_nulls, " & _
            "show_query_plan, show_stats_time, " & _
            "show_stats_io, parse_odbc_msg_prefixes, " & _
            "row_count, tsql_batch_separator, " & _
            "query_time_out, server_language, " & _
            "character_translation, regional_settings ) " & _
           " values ( [w_id], [c_id], [c_name], [c_value], " & _
           " [desc], [no_count], [no_exec], [parse_only], " & _
           " [quoted_id], [a_nulls], [show_qp], [stats_tm], " & _
           " [stats_io], [parse_odbc], [row_cnt], [batch_sep], " & _
           " [qry_tmout], [lang], [char_trans], [reg_settings] ) "

    Set qy = NodeDB.CreateQueryDef(gstrEmptyString,strInsert)

    ' Call a procedure to assign the Connection values
    Call AssignParameters(qy)

    qy.Execute dbFailOnError
    qy.Close

    Exit Sub

AddErr:

    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errInsertFailed, _
        mstrModuleName & "Add", LoadResString(errInsertFailed)

End Sub
Public Sub Delete()

    Dim strDelete As String
    Dim qy As DAO.QueryDef

    On Error GoTo DeleteErr

    ' Check if the db object is valid
    Call CheckDB

    strDelete = "delete from workspace_connections " & _
          " where connection_id = [c_id]"
    Set qy = NodeDB.CreateQueryDef(gstrEmptyString,strDelete)

    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    qy.Close

    Exit Sub

DeleteErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteFailed, _
          mstrModuleName & "Delete", LoadResString(errDeleteFailed)

End Sub

Public Sub Modify()

    Dim strUpdate As String
    Dim qy As QueryDef

    On Error GoTo ModifyErr

    ' Validate the updated values before trying to modify thedb
    Call Validate

    ' Create a temporary querydef object with the modify string
    strUpdate = "update workspace_connections " & _
          " set workspace_id = [w_id], " & _
          "connection_name = [c_name], " & _
          "connection_value = [c_value], " & _
          "description = [desc], " & _
          "no_count_display = [no_count], " & _
          "no_execute = [no_exec], " & _
          "parse_query_only = [parse_only], " & _
          "ANSI_quoted_identifiers = [quoted_id], " & _
          "ANSI_nulls = [a_nulls], " & _
          "show_query_plan = [show_qp], " & _
          "show_stats_time = [stats_tm], " & _
          "show_stats_io = [stats_io], " & _
          "parse_odbc_msg_prefixes = [parse_odbc], " & _
          "row_count = [row_cnt], " & _
          "tsql_batch_separator = [batch_sep], " & _
          "query_time_out = [qry_tmout], " & _
          "server_language = [lang], " & _
          "character_translation = [char_trans], " & _
          "regional_settings = [reg_settings] " & _
          " where connection_id = [c_id]"
    Set qy = NodeDB.CreateQueryDef(gstrEmptyString,strUpdate)

    ' Call a procedure to assign the Connection values to the
    ' querydef object
    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    qy.Close

    Exit Sub

ModifyErr:

    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errModifyFailed, _
        mstrModuleName & "Modify", LoadResString(errModifyFailed)

End Sub
Public Property Get ConnectionName() As String

    ConnectionName = mstrConnectionName

End Property

Public Property Get NextIdentifier() As Long

    Dim lngNextId As Long

    On Error GoTo NextIdentifierErr

    ' First check if the database object is valid
    Call CheckDB

    ' Retrieve the next identifier using the sequence class
    Set mConnectionSeq = New cSequence
    Set mConnectionSeq.IdDatabase = NodeDB
    mConnectionSeq.IdentifierColumn= "connection_id"
    lngNextId = mConnectionSeq.Identifier
    Set mConnectionSeq = Nothing

    NextIdentifier = lngNextId
    Exit Property

NextIdentifierErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errIdGetFailed, _
        mstrModuleName & "NextIdentifier", LoadResString(errIdGetFailed)

End Property
Public Property Get IndOperation() As Operation

    IndOperation = mintOperation

End Property

Private Sub Class_Initialize()

    Set mFieldValue = New cStringSM

    ' Initialize the operation indicator variable to Query
    ' It will be modified later by the collection class when
    ' inserts, updates or deletes are performed
```

```
    mintOperation = QueryOp

    ' Initialize connection properties to their default values
    NoCountDisplay = DEF_NO_COUNT_DISPLAY
    NoExecute = DEF_NO_EXECUTE
    ParseQueryOnly = DEF_PARSE_QUERY_ONLY
    QuotedIdentifiers = DEF_ANSI_QUOTED_IDENTIFIERS
    AnsiNulls = DEF_ANSI_NULLS
    ShowQueryPlan = DEF_SHOW_QUERY_PLAN
    ShowStatsTime = DEF_SHOW_STATS_TIME
    ShowStatsIO = DEF_SHOW_STATS_IO
    ParseOdbcMsg = DEF_PARSE_ODBC_MSG_PREFIXES
    RowCount = DEF_ROW_COUNT
    TsqlBatchSeparator = DEF_TSQL_BATCH_SEPARATOR
    QueryTimeOut = DEF_QUERY_TIME_OUT
    ServerLanguage = DEF_SERVER_LANGUAGE
    CharacterTranslation = DEF_CHARACTER_TRANSLATION
    RegionalSettings = DEF_REGIONAL_SETTINGS

End Sub

Private Sub Class_Terminate()

    Set NodeDB = Nothing
    Set mFieldValue = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cConnections"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cConnections.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Implements an array of cConnection objects.
'              Type-safe wrapper around cNodeCollections.
'              Also contains validation functions, etc.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mcarrConnections As cNodeCollections

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cConnections."

Public Property Set ConnDb(vdata As Database)

    Set mcarrConnections.NodeDB = vdata

End Property
Public Sub Modify(cModifiedConn As cConnection)

    ' First check if the parameter record is valid
    Call CheckDupConnName(cModifiedConn)

    Call mcarrConnections.Modify(cModifiedConn)

End Sub
Public Sub Load(ByRef cConnToAdd As cConnection)

    Call mcarrConnections.Load(cConnToAdd)

End Sub
Public Sub Add(ByRef cConnToAdd As cConnection)

    Set cConnToAdd.NodeDB = mcarrConnections.NodeDB

    ' First check if the record is valid
    Call Validate(cConnToAdd)

    ' Retrieve a unique identifier
    cConnToAdd.ConnectionId = cConnToAdd.NextIdentifier

    Call mcarrConnections.Add(cConnToAdd)

End Sub

Public Sub Unload(lngConnId As Long)

    Dim lngDeleteElement As Long

    lngDeleteElement = QueryIndex(lngConnId)

    Call mcarrConnections.Unload(lngDeleteElement)

End Sub

Public Sub SaveConnectionsInWsp(ByVal lngWorkspace As Long)
    ' Call a procedure to save all connection records for the workspace
    Call mcarrConnections.Save(lngWorkspace)

End Sub
Public Function GetConnection(ByVal lngWorkspace As Long, _
        ByVal strConnectionName As String) As cConnection
    ' Returns the connection string for the passed in connection name

    Dim lngIndex As Long

    ' Find all parameters in the array with a matching workspace id
    For lngIndex = 0 To mcarrConnections.Count - 1
        If mcarrConnections(lngIndex).WorkspaceId = lngWorkspace And _
            mcarrConnections(lngIndex).ConnectionName= strConnectionName
Then

            Set GetConnection = mcarrConnections(lngIndex)
            Exit For
        End If
    Next lngIndex

    If lngIndex > mcarrConnections.Count - 1 Then
        ' The parameter has not been defined for the workspace
        ' Raise an error
        On Error GoTo 0
        Err.Raise vbObjectError + errConnNameInvalid, mstrModuleName &
"GetConnection", _
            LoadResString(errConnNameInvalid)
    End If

End Function
Public Sub Delete(lngConnId As Long)
    ' Delete the passed in parameter

    Dim lngDeleteElement As Long

    lngDeleteElement = QueryIndex(lngConnId)
    Call mcarrConnections.Delete(lngDeleteElement)

End Sub
Private Function QueryIndex(lngConnId As Long) As Long

    Dim lngIndex As Long

    ' Find the matching parameter record in the array
    For lngIndex = 0 To mcarrConnections.Count - 1
        If mcarrConnections(lngIndex).ConnectionId= lngConnId And _
            mcarrConnections(lngIndex).IndOperation<> DeleteOp Then
            QueryIndex = lngIndex
            Exit Function
        End If
    Next lngIndex

    ' Raise error that parameter has not been found
    On Error GoTo 0
    Err.Raise vbObjectError + errQueryIndexFailed,
"cArrParameters.QueryIndex", _
```

```
        LoadResString(errQueryIndexFailed)

End Function

Public Function QueryConnection(lngConnId As Long) As cConnection

    Dim lngQueryElement As Long

    lngQueryElement = QueryIndex(lngConnId)

    ' Return the queried connection object
    Set QueryConnection = mcarrConnections(lngQueryElement)

End Function
Public Property Get Count() As Long

    Count = mcarrConnections.Count

End Property
Public Property Get Item(lngIndex As Long) As cConnection
Attribute Item.VB_UserMemId = 0

    Set Item = mcarrConnections(lngIndex)

End Property

Public Sub Validate(ByVal cConnToValidate As cConnection)
    ' This procedure is necessary since the class cannot validate
    ' all the parameter properties on it's own. This is 'coz we
    ' might have created new parameters in the workspace, but not
    ' saved them to the database yet - hence the duplicate check
    ' has to be repeated in the array

    Dim lngIndex As Long
    Dim cTempParam As cConnection

    ' Check if the parameter name already exists in the workspace
    For lngIndex = 0 To mcarrConnections.Count - 1
        Set cTempParam = mcarrConnections(lngIndex)
        If cTempParam.WorkspaceId = cConnToValidate.WorkspaceId And _
            cTempParam.ConnectionName = cConnToValidate.ConnectionName
And _
            cTempParam.IndOperation <> DeleteOp Then
            On Error GoTo 0
            Err.Raise vbObjectError + errDuplicateConnectionName, _
                mstrSource, LoadResString(errDuplicateConnectionName)
        End If
    Next lngIndex

End Sub
Public Sub CheckDupConnName(ByVal cConnToValidate As cConnection)

    Dim lngIndex As Long
    Dim cTempParam As cConnection

    ' Check if the parameter name already exists in the workspace
    For lngIndex = 0 To mcarrConnections.Count - 1
        Set cTempParam = mcarrConnections(lngIndex)
        If cTempParam.WorkspaceId = cConnToValidate.WorkspaceId And _
            cTempParam.ConnectionName = cConnToValidate.ConnectionName
And _
            cTempParam.ConnectionId <> cConnToValidate.ConnectionId And _
            cTempParam.IndOperation <> DeleteOp Then
            ShowError errDuplicateConnectionName
            On Error GoTo 0
            Err.Raise vbObjectError + errDuplicateConnectionName, _
                mstrSource, LoadResString(errDuplicateConnectionName)
        End If
    Next lngIndex

End Sub

Private Sub Class_Initialize()

    Set mcarrConnections = New cNodeCollections

End Sub

Private Sub Class_Terminate()

    Set mcarrConnections = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cConstraint"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cConstraint.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Encapsulates the properties and methods of a constraint.
'              Contains functions to insert, update and delete
'              step_constraints records from the database.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Module level variables to store the property values
Private mlngConstraintId As Long
Private mlngStepId As Long
Private mstrVersionNo As String
Private mintConstraintType As Integer
Private mlngGlobalStepId As Long
Private mstrGlobalVersionNo As String
Private mintSequenceNo As Integer
Private mdbsConstraintDB As Database
Private mlngWorkspaceId As Integer
Private mintOperation As Operation
Private mlngPosition As Long

' The cSequence class is used to generate unique step identifiers
Private mConstraintSeq As cSequence

Private Const mstrModuleName As String = ".cConstraint."
Private mstrSource As String

Public Enum ConstraintType
    gintPreStep = 1
    gintPostStep = 2
End Enum

Private Const mstrSQ As String = """
Public Property Get WorkspaceId() As Long
    WorkspaceId = mlngWorkspaceId
End Property
Public Property Let WorkspaceId(ByVal vdata As Long)
    mlngWorkspaceId = vdata
End Property

Public Property Get IndOperation() As Operation

    IndOperation = mintOperation

End Property
Public Property Let IndOperation(ByVal vdata As Operation)

    On Error GoTo IndOperationErr
    mstrSource = mstrModuleName & "IndOperation"

    ' The valid operations are define in the cOperations
    ' class. Check if the operation is valid
    Select Case vdata
        Case QueryOp, InsertOp, UpdateOp, DeleteOp
            mintOperation = vdata

        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errInvalidOperation, _
```

```
            mstrSource, LoadResString(errInvalidOperation)
    End Select

    Exit Property

IndOperationErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "IndOperation"
    On Error GoTo 0
    Err.Raise vbObjectError + errLetOperationFailed, _
        mstrSource, LoadResString(errLetOperationFailed)

End Property

Public Function Clone() As cConstraint

    ' Creates a copy of a given constraint

    Dim cConsClone As cConstraint

    On Error GoTo CloneErr
    mstrSource = mstrModuleName & "Clone"

    Set cConsClone = New cConstraint

    ' Copy all the workspace properties to the newly
    ' created workspace
    cConsClone.ConstraintId = mlngConstraintId
    cConsClone.StepId = mlngStepId
    cConsClone.VersionNo = mstrVersionNo
    cConsClone.ConstraintType = mintConstraintType
    cConsClone.GlobalStepId = mlngGlobalStepId
    cConsClone.GlobalVersionNo = mstrGlobalVersionNo
    cConsClone.SequenceNo = mintSequenceNo
    cConsClone.WorkspaceId = mlngWorkspaceId
    cConsClone.IndOperation = mintOperation

    ' And set the return value to the newly created constraint
    Set Clone = cConsClone

    Exit Function

CloneErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Clone"
    On Error GoTo 0
    Err.Raise vbObjectError + errCloneFailed, _
          mstrSource, LoadResString(errCloneFailed)

End Function

Public Property Get SequenceNo() As Integer

    SequenceNo = mintSequenceNo

End Property

Public Property Let SequenceNo(ByVal vdata As Integer)
    mintSequenceNo = vdata
End Property

Public Sub Add()
    ' Inserts a new step constraint into the database

    Dim strInsert As String
    Dim qy As DAO.QueryDef

    On Error GoTo AddErr

    ' First check if the database object is valid
    Call CheckDB

    ' Any record validations
    Call Validate

    ' Create a temporary querydef object
    strInsert = "insert into step_constraints " & _
          "( constraint_id, step_id, version_no, " & _
          " constraint_type, global_step_id, global_version_no, sequence_no ) " & _
          " values ( [cons_id], [s_id], [ver_no], " & _
          " [cons_type], [g_step_id], [g_ver_no], " & _
          " [seq_no] )"
    Set qy = mdbsConstraintDB.CreateQueryDef(gstrEmptyString,strInsert)

    ' Call a procedure to execute the Querydef object
    Call AssignParameters(qy)

    qy.Execute dbFailOnError
    qy.Close

'    strInsert = "insert into step_constraints " & _
'        "( constraint_id, step_id, version_no, " & _
'        " constraint_type, global_step_id, global_version_no, sequence_no ) " & _
'        " values ( " & _
'        Str(mlngConstraintId) & ", " & Str(mlngStepId) & ", " & _
'        mstrSQ & mstrVersionNo & mstrSQ & ", " & Str(mintConstraintType) & ",
" & _
'        Str(mlngGlobalStepId) & ", " & mstrSQ & mstrGlobalVersionNo & mstrSQ
& ", " & _
'        Str(mintSequenceNo) & " ) "
'
'    BugMessage strInsert
'    mdbsConstraintDB.Execute strInsert, dbFailOnError
    Exit Sub

AddErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Add"
    On Error GoTo 0
    Err.Raise vbObjectError + errAddConstraintFailed, _
          mstrSource, _
          LoadResString(errAddConstraintFailed)
End Sub
Private Sub AssignParameters(qyExec As DAO.QueryDef)
    ' Assigns values to the parameters in the querydef object
    ' The parameter names are cryptic to make them different
    ' from the field names. When the parameter names are
    ' the same as the field names, parameters in the where
    ' clause do not get created.

    Dim prmParam As DAO.Parameter

    On Error GoTo AssignParametersErr
    mstrSource = mstrModuleName & "AssignParameters"

    For Each prmParam In qyExec.Parameters
        Select Case prmParam.Name
            Case "[cons_id]"
                prmParam.Value = mlngConstraintId

            Case "[s_id]"
                prmParam.Value = mlngStepId

            Case "[ver_no]"
                prmParam.Value = mstrVersionNo

            Case "[cons_type]"
                prmParam.Value = mintConstraintType

            Case "[g_step_id]"
                prmParam.Value = mlngGlobalStepId

            Case "[g_ver_no]"
                prmParam.Value = mstrGlobalVersionNo

            Case "[seq_no]"
                prmParam.Value = mintSequenceNo

            Case Else
                ' Write the parameter name that is faulty
                WriteError errInvalidParameter, mstrSource, _
                      prmParam.Name
                On Error GoTo 0
                Err.Raise errInvalidParameter, mstrSource, _
                      LoadResString(errInvalidParameter)
        End Select
```

```
    Next prmParam

    Exit Sub

AssignParametersErr:

    mstrSource = mstrModuleName & "AssignParameters"
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errAssignParametersFailed, _
        mstrSource, LoadResString(errAssignParametersFailed)

End Sub
Public Property Get NextIdentifier() As Long

    Dim lngNextId As Long

    On Error GoTo NextIdentifierErr

    ' First check if the database object is valid
    Call CheckDB

    ' Retrieve the next constraint identifier using the
    ' sequence class
    Set mConstraintSeq = New cSequence
    Set mConstraintSeq.IdDatabase = mdbsConstraintDB
    mConstraintSeq.IdentifierColumn= "constraint_id"
    lngNextId = mConstraintSeq.Identifier
    Set mConstraintSeq = Nothing

    NextIdentifier = lngNextId
    Exit Property

NextIdentifierErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "NextIdentifier"
    On Error GoTo 0
    Err.Raise vbObjectError + errStepIdGetFailed, _
        mstrSource, LoadResString(errStepIdGetFailed)

End Property

Private Sub CheckDB()
    ' Check if the database object has been initialized

    If mdbsConstraintDB Is Nothing Then
        ShowError errInvalidDB
        On Error GoTo 0
        Err.Raise vbObjectError + errInvalidDB, _
            mstrModuleName, LoadResString(errInvalidDB)
    End If

End Sub

Public Sub Delete()
    ' Deletes the step constraint record from the database

    Dim strDelete As String
    Dim qy As DAO.QueryDef

    On Error GoTo DeleteErr
    mstrSource = mstrModuleName & "Delete"

    ' There can be multiple constraints for a step,
    ' meaning that there can be multiple constraint records
    ' with the same constraint_id. Only a combination
    ' of the step_id, version and constraint_id will be
    ' unique
    strDelete = "delete from step_constraints " & _
            " where constraint_id = [cons_id]" & _
            " and step_id = [s_id] " & _
            " and version_no = [ver_no] "
    Set qy = mdbsConstraintDB.CreateQueryDef(gstrEmptyString,strDelete)

    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    qy.Close

'    strDelete = "Delete from step_constraints " & _
'            " where constraint_id = " & Str(mlngConstraintId) & _
'            " and step_id = " & Str(mlngStepId) & _
'            " and version_no = " & mstrSQ & mstrVersionNo & mstrSQ
'
'    'BugMessage strDelete
'    mdbsConstraintDB.Execute strDelete, dbFailOnError

    Exit Sub

DeleteErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Delete"
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteConstraintFailed, _
            mstrSource, _
            LoadResString(errDeleteConstraintFailed)
End Sub
Public Sub Modify()
    ' Updates the sequence no of the step constraint record
    ' in the database

    Dim strUpdate As String
    Dim qy As QueryDef

    On Error GoTo Modify

    ' First check if the database object is valid
    Call CheckDB

    ' Any record validations
    Call Validate

    ' There can be multiple constraints for a step,
    ' meaning that there can be multiple constraint records
    ' with the same constraint_id. Only a combination
    ' of the step_id, version and constraint_id will be
    ' unique
    ' Create a temporary querydef object with the modify string
    strUpdate = "Update step_constraints " & _
            " set sequence_no = [seq_no] " & _
            " where constraint_id = [cons_id] " & _
            " and step_id = [s_id] " & _
            " and version_no = [ver_no] "
    Set qy = mdbsConstraintDB.CreateQueryDef(gstrEmptyString,strUpdate)

    ' Call a procedure to assign the parameter values to the
    ' querydef object
    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    qy.Close

'    strUpdate = "Update step_constraints " & _
'            " set sequence_no = " & Str(mintSequenceNo) & _
'            " where constraint_id = " & Str(mlngConstraintId) & _
'            " and step_id = " & Str(mlngStepId) & _
'            " and version_no = " & mstrSQ & mstrVersionNo & mstrSQ
'
'    'BugMessage strUpdate
'    mdbsConstraintDB.Execute strUpdate, dbFailOnError
    Exit Sub

Modify:
    LogErrors Errors
    mstrSource = mstrModuleName & "Modify"
    On Error GoTo 0
    Err.Raise vbObjectError + errUpdateConstraintFailed, _
            mstrSource, _
            LoadResString(errUpdateConstraintFailed)
End Sub
Public Property Get Position() As Long

    Position = mlngPosition

End Property
Public Property Let Position(ByVal RHS As Long)
```

```
    mlngPosition = RHS

End Property

Public Sub Validate()
    ' Each distinct object will have a Validate method which
    ' will check if the class properties are valid. This method
    ' will be used to check interdependant properties that
    ' cannot be validated by the let procedures.
    ' It should be called by the add and modify methods of the class

    ' No validations are necessary for the constraint object

End Sub

Public Property Set NodeDB(vdata As Database)

    Set mdbsConstraintDB = vdata

End Property

Public Property Get NodeDB() As Database

    Set NodeDB = mdbsConstraintDB

End Property

Public Property Get GlobalVersionNo() As String

    GlobalVersionNo = mstrGlobalVersionNo

End Property

Public Property Let GlobalVersionNo(ByVal vdata As String)

    mstrGlobalVersionNo = vdata

End Property

Public Property Get GlobalStepId() As Long

    GlobalStepId = mlngGlobalStepId

End Property

Public Property Get ConstraintId() As Long

    ConstraintId = mlngConstraintId

End Property

Public Property Get VersionNo() As String

    VersionNo = mstrVersionNo

End Property

Public Property Get StepId() As Long

    StepId = mlngStepId

End Property

Public Property Let VersionNo(ByVal vdata As String)

    mstrVersionNo = vdata

End Property

Public Property Let StepId(ByVal vdata As Long)

    mlngStepId = vdata

End Property

Public Property Let ConstraintId(ByVal vdata As Long)
    On Error GoTo ConstraintIdErr
    mstrSource = mstrModuleName & "ConstraintId"

    If (vdata > 0) Then
        mlngConstraintId = vdata
    Else
        ' Propogate this error back to the caller
        On Error GoTo 0
        Err.Raise vbObjectError + errConstraintIdInvalid, _
            mstrSource, LoadResString(errConstraintIdInvalid)
    End If

    Exit Property

ConstraintIdErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "ConstraintId"
    On Error GoTo 0
    Err.Raise vbObjectError + errConstraintIdSetFailed, _
        mstrSource, LoadResString(errConstraintIdSetFailed)

End Property

Public Property Let GlobalStepId(ByVal vdata As Long)

    On Error GoTo GlobalStepIdErr
    mstrSource = mstrModuleName & "GlobalStepId"

    If (vdata > 0) Then
        mlngGlobalStepId = vdata
    Else
        ' Propogate this error back to the caller
        On Error GoTo 0
        Err.Raise vbObjectError + errGlobalStepIdInvalid, _
            mstrSource, LoadResString(errGlobalStepIdInvalid)
    End If

    Exit Property

GlobalStepIdErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "GlobalStepId"
    On Error GoTo 0
    Err.Raise vbObjectError + errGlobalStepIdSetFailed, _
        mstrSource, LoadResString(errGlobalStepIdSetFailed)

End Property

Public Property Let ConstraintType(ByVal vdata As ConstraintType)

    On Error GoTo ConstraintTypeErr

    ' A global step can be either a pre- or a post-execution step.
    ' These constants have been defined in the enumeration,
    ' ConstraintType, which is exposed
    Select Case vdata
        Case gintPreStep, gintPostStep
            mintConstraintType = vdata

        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errConstraintTypeInvalid, _
                mstrSource, LoadResString(errConstraintTypeInvalid)
    End Select

    Exit Property

ConstraintTypeErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "ConstraintType"
    On Error GoTo 0
    Err.Raise vbObjectError + errConstraintTypeLetFailed, _
        mstrSource, LoadResString(errConstraintTypeLetFailed)

End Property

Public Property Get ConstraintType() As ConstraintType

    ConstraintType = mintConstraintType
```

```
End Property

Private Sub Class_Initialize()

    ' Initialize the operation indicator variable to Query
    ' It will be modified later by the collection class when
    ' inserts, updates or deletes are performed
    mintOperation = QueryOp

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cFailedStep"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cFailedStep.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Properties of a step execution failure.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Public InstanceId As Long
Public StepId As Long
Public ParentStepId As Long
Public ContCriteria As ContinuationCriteria
Public EndTime As Currency
Public AskResponse As Long
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cFailedSteps"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cFailedSteps.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module encapsulates a collection of failed steps. It
'              also determines whether sub-steps of a passed in step need
'              to be skipped due to a failure.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mcFailedSteps As cVector
Public Function ExecuteSubStep(lParentStepId As Long) As Boolean
    ' Returns False if there is any condition that prevents sub-steps of the passed
    ' in instance from being executed
    Dim lIndex As Long

    ExecuteSubStep = True

    For lIndex = 0 To Count() - 1
        If mcFailedSteps(lIndex).ContCriteria = gintOnFailureCompleteSiblingsAnd
_
                lParentStepId <> mcFailedSteps(lIndex).ParentStepId Then
            ExecuteSubStep = False
            Exit For
        End If

        If mcFailedSteps(lIndex).ContCriteria = gintOnFailureAbortSiblingsAnd _
                lParentStepId = mcFailedSteps(lIndex).ParentStepId Then
            ExecuteSubStep = False
            Exit For
        End If

        If mcFailedSteps(lIndex).ContCriteria = gintOnFailureSkipSiblingsAnd _
                lParentStepId = mcFailedSteps(lIndex).ParentStepId Then
            ExecuteSubStep = False
            Exit For
        End If

        If mcFailedSteps(lIndex).ContCriteria = gintOnFailureAbort Then
            ExecuteSubStep = False
            Exit For
        End If

    Next lIndex

End Function
Public Sub Add(ByVal objItem As cFailedStep)

    mcFailedSteps.Add objItem

End Sub
Public Function Delete(ByVal lPosition As Long) As cFailedStep

    Set Delete = mcFailedSteps.Delete(lPosition)

End Function

Public Sub Clear()

    mcFailedSteps.Clear

End Sub
Public Function Count() As Long

    Count = mcFailedSteps.Count

End Function
Public Property Get Item(ByVal Position As Long) As cFailedStep
Attribute Item.VB_UserMemId = 0

    Set Item = mcFailedSteps.Item(Position)

End Property

Public Function StepFailed(lStepId As Long) As Boolean

    ' Returns True if a failure record already exists for the passed in step
    Dim lIndex As Long

    StepFailed = False

    For lIndex = 0 To Count() - 1
        If mcFailedSteps(lIndex).StepId = lStepId Then
            StepFailed = True
            Exit For
        End If
    Next lIndex

End Function

Private Sub Class_Initialize()

    Set mcFailedSteps = New cVector

End Sub

Private Sub Class_Terminate()

    Set mcFailedSteps = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cFileInfo"
Attribute VB_GlobalNameSpace = False
```

```
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cFileInfo.cls
'              Microsoft  TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:     File Properties viz. name, handle, etc.
'  Contact:     Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mstrFileName As String
Private mintFileHandle As Integer
Private mdbsNodeDb As Database ' Since it is used to form a cNodeCollection
Private mlngPosition As Long ' Since it is used to form a cNodeCollection
Public Property Get FileName() As String

    FileName = mstrFileName

End Property
Public Property Let FileName(ByVal vdata As String)

    mstrFileName = vdata

End Property
Public Property Let FileHandle(ByVal vdata As Integer)

    mintFileHandle = vdata

End Property
Public Property Set NodeDB(vdata As Database)

    Set mdbsNodeDb = vdata

End Property

Public Property Get NodeDB() As Database

    Set NodeDB = mdbsNodeDb

End Property
Public Property Get Position() As Long

    Position = mlngPosition

End Property
Public Property Let Position(ByVal vdata As Long)

    mlngPosition = vdata

End Property

Public Property Get FileHandle() As Integer

    FileHandle = mintFileHandle

End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cFileSM"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
Attribute VB_Ext_KEY = "SavedWithClassBuilder","Yes"
Attribute VB_Ext_KEY = "Top_Level","Yes"
'  FILE:       cFileSM.cls
'              Microsoft  TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:     Encapsulates functions to open a file and write to it.
'  Contact:     Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "cFileSM."
Private mstrSource As String

Private mstrFileName As String
Private mintHFile As Integer
Private mstrFileHeader As String
Private mstrProjectName As String

Public Sub CloseFile()

    ' Close the file
    If mintHFile > 0 Then
        Call CloseFileSM(mstrFileName)
        mintHFile = 0
    End If

End Sub

Public Property Let ProjectName(ByVal vdata As String)
    ' An optional field - will be appended to the file
    ' header string if specified

    Const strProjectHdr As String = "Project Name:"

    mstrProjectName = vdata
    mstrFileHeader = mstrFileHeader & _
            Space$(1) & strProjectHdr & Space$(1) & _
            gstrSQ & vdata & gstrSQ

End Property
Public Property Get ProjectName() As String

    ProjectName = mstrProjectName

End Property
Public Property Get FileName() As String

    FileName = mstrFileName

End Property
Public Property Let FileName(ByVal vdata As String)

    mstrFileName = vdata

End Property
Public Sub WriteLine(strMsg As String)

    ' Writes the passed in string to the file
    Call WriteToFile(strMsg, False)

End Sub

Public Sub WriteField(strMsg As String)

    ' Writes the passed in string to the file
    Call WriteToFile(strMsg, True)

End Sub

Private Sub WriteToFile(strMsg As String, _
        blnContinue As Boolean)
    ' Writes the passed in string to the file - the
    ' Continue flag indicates whether the next line will
    ' be continued on the same line or printed on a new one

    On Error GoTo WriteToFileErr

    ' Open the file if it hasn't been already
    If mintHFile = 0 Then

        ' If the filename has not been initialized, do not
        ' attempt to open it
```

```
      If mstrFileName <> gstrEmptyString Then

         mintHFile = OpenFileSM(mstrFileName)

         If mintHFile = 0 Then
            ' The Open File command failed for some reason
            ' No point in trying to write the file header
         Else
            ' Print a file header, if a header string has been
            ' initialized
            If mstrFileHeader <> gstrEmptyString Then
               Print #mintHFile,
               Print #mintHFile, mstrFileHeader
               Print #mintHFile,
            End If
         End If
      End If
   End If

   If mintHFile <> 0 Then
      If strMsg = gstrEmptyString Then
         Print #mintHFile,
      Else
         If blnContinue Then
            ' Write the message to the file - continue
            ' all subsequent characters on the same line
            Print #mintHFile, strMsg;
         Else
            ' Write the message to the file
            Print #mintHFile, strMsg
         End If
      End If
   Else
      ' Display the string to the user instead of
      ' trying to write it to the file
      ' This could be the project error log that we were
      ' trying to open! Play it safe and display errors - do
      ' not try to log them.
      MsgBox strMsg, vbOKOnly
   End If

   Exit Sub

WriteToFileErr:
   ' Log the error code raised by Visual Basic
   Call DisplayErrors(Errors)

   ' Display the string to the user instead of
   ' trying to write it to the file
   MsgBox strMsg, vbOKOnly

End Sub
Public Property Let FileHeader(ByVal vdata As String)

   mstrFileHeader = vdata

End Property
Public Property Get FileHeader() As String

   FileHeader = mstrFileHeader

End Property

Private Sub Class_Terminate()

   ' Close the file opened by this instance
   Call CloseFile

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cGlobalStep"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
```

```
'  FILE:       cGlobalStep.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Encapsulates the properties and methods of a global step.
'              Implements the cStep class - carries out initializations
'              and validations that are specific to global steps.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Implements cStep

' Object variable to keep the reference in
Private mcStep As cStep

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cGlobalStep."

Private Sub cStep_AddIterator(cItRecord As cIterator)

   Call mcStep.AddIterator(cItRecord)

End Sub

Private Property Let cStep_ArchivedFlag(ByVal RHS As Boolean)

   mcStep.ArchivedFlag = RHS

End Property

Private Property Get cStep_ArchivedFlag() As Boolean

   cStep_ArchivedFlag = mcStep.ArchivedFlag

End Property

Private Sub Class_Initialize()

   ' Create the object
   Set mcStep = New cStep

   ' Initialize the object with valid values for a global step
   ' The global flag should be the first field to be initialized
   ' since subsequent validations might try to check if the
   ' step being created is global
   mcStep.GlobalFlag = True
   mcStep.StepType = gintGlobalStep

   ' A global step cannot have any sub-steps associated with it
   ' Hence, it will always be at Step Level 0
   mcStep.ParentStepId = 0
   mcStep.ParentVersionNo = gstrMinVersion
   mcStep.StepLevel = 0

   ' The enabled flag must be False for all global steps
   ' Global steps can be of two types
   ' a. Those that are run globally within a workspace either
   '    before every step, after every step or during the entire
   '    run, depending on the global run method
   ' b. Those that are not run globally, but qualify to be either
   '    pre or post-execution steps for other steps in the workspace.
   '    Whether or not such a step will be executed depends on
   '    whether the step for which it is defined as a pre/post
   '    step will be executed
   mcStep.EnabledFlag = False

   mcStep.ContinuationCriteria = gintNoOption
   mcStep.DegreeParallelism = gstrGlobalParallelism

End Sub
Private Sub Class_Terminate()

   ' Remove the step object
```

```
    Set mcStep = Nothing

End Sub

Private Sub cStep_Add()

    ' Call a private procedure to see if the step text has been
    ' entered - since a global step actually executes a step, entry
    ' of the text is mandatory
    Call StepTextOrFileEntered

    ' Call the Add method of the step class to carry out the insert
    mcStep.Add

End Sub

Private Function cStep_Clone(Optional cCloneStep As cStep) As cStep

    Dim cNewGlobal As cGlobalStep

    Set cNewGlobal = New cGlobalStep
    Set cStep_Clone = mcStep.Clone(cNewGlobal)

End Function


Private Property Get cStep_ContinuationCriteria() As ContinuationCriteria

    cStep_ContinuationCriteria = mcStep.ContinuationCriteria

End Property

Private Property Let cStep_ContinuationCriteria(ByVal RHS As
ContinuationCriteria)

    ' The continuation criteria field will always be empty for a
    ' global step
    mcStep.ContinuationCriteria = 0

End Property

Private Property Let cStep_DegreeParallelism(ByVal RHS As String)

    ' Will always be zero for a global step
    mcStep.DegreeParallelism = gstrGlobalParallelism

End Property

Private Property Get cStep_DegreeParallelism() As String

    cStep_DegreeParallelism = mcStep.DegreeParallelism

End Property

Private Sub cStep_DeleteIterator(cItRecord As cIterator)

    Call mcStep.DeleteIterator(cItRecord)

End Sub

Private Sub cStep_Delete()

    mcStep.Delete

End Sub

Private Property Get cStep_EnabledFlag() As Boolean

    cStep_EnabledFlag = mcStep.EnabledFlag

End Property

Private Property Let cStep_EnabledFlag(ByVal RHS As Boolean)

    ' The enabled flag must be False for all global steps
    ' Global steps can be of two types
    ' a. Those that are run globally within a workspace either
    '    before every step, after every step or during the entire
    '    run, depending on the global run method
    ' b. Those that are not run globally, but qualify to be either
    '    pre or post-execution steps for other steps in the workspace.
    '    Whether or not such a step will be executed depends on
    '    whether the step for which it is defined as a pre/post
    '    step will be executed
    mcStep.EnabledFlag = False

End Property

Private Property Let cStep_ErrorFile(ByVal RHS As String)

    mcStep.ErrorFile = RHS

End Property

Private Property Get cStep_ErrorFile() As String

    cStep_ErrorFile = mcStep.ErrorFile

End Property

Private Property Let cStep_ExecutionMechanism(ByVal RHS As
ExecutionMethod)

    ' Whether or not the Execution Mechanism is valid will be
    ' checked by the Step class
    mcStep.ExecutionMechanism = RHS

End Property

Private Property Get cStep_ExecutionMechanism() As ExecutionMethod

    cStep_ExecutionMechanism = mcStep.ExecutionMechanism

End Property

Private Property Let cStep_FailureDetails(ByVal RHS As String)

    ' Whether or not the Failure Details are valid for the
    ' selected failure criteria will be checked by the Step class
    mcStep.FailureDetails = RHS

End Property

Private Property Get cStep_FailureDetails() As String

    cStep_FailureDetails = mcStep.FailureDetails

End Property

Private Property Get cStep_GlobalFlag() As Boolean

    cStep_GlobalFlag = mcStep.GlobalFlag

End Property

Private Property Let cStep_GlobalFlag(ByVal RHS As Boolean)

    ' Set the global flag to true
    mcStep.GlobalFlag = True

End Property

Private Function cStep_IncVersionX() As String

    cStep_IncVersionX = mcStep.IncVersionX

End Function

Private Function cStep_IncVersionY() As String

    cStep_IncVersionY = mcStep.IncVersionY

End Function


'Private Property Let cStep_GlobalRunMethod(ByVal RHS As Integer)
```

```
'
'    ' Whether or not the Global Run Method is valid for the step
'    ' will be checked by the Step class
'    mcStep.GlobalRunMethod = RHS
'
'End Property
'
'Private Property Get cStep_GlobalRunMethod() As Integer
'
'    cStep_GlobalRunMethod = mcStep.GlobalRunMethod
'
'End Property
'
Private Property Get cStep_IndOperation() As Operation

    cStep_IndOperation = mcStep.IndOperation

End Property

Private Property Let cStep_IndOperation(ByVal RHS As Operation)

    mcStep.IndOperation = RHS

End Property

Private Sub cStep_InsertIterator(cItRecord As cIterator)

    Call mcStep.InsertIterator(cItRecord)

End Sub

Private Function cStep_IsNewVersion() As Boolean
    cStep_IsNewVersion = mcStep.IsNewVersion
End Function

Private Function cStep_IteratorCount() As Long

    cStep_IteratorCount = mcStep.IteratorCount

End Function

Private Property Let cStep_IteratorName(ByVal RHS As String)

    mcStep.IteratorName = RHS

End Property

Private Property Get cStep_IteratorName() As String

    cStep_IteratorName = mcStep.IteratorName

End Property

Private Function cStep_Iterators() As Variant

    cStep_Iterators = mcStep.Iterators

End Function

Private Sub cStep_LoadIterator(cItRecord As cIterator)

    Call mcStep.LoadIterator(cItRecord)

End Sub

'Private Property Let cStep_LogFile(ByVal RHS As String)
'
'    mcStep.LogFile = RHS
'
'End Property
'
'Private Property Get cStep_LogFile() As String
'
'    cStep_LogFile = mcStep.LogFile
'
'End Property

Private Sub cStep_ModifyIterator(cItRecord As cIterator)

    Call mcStep.ModifyIterator(cItRecord)

End Sub

Private Sub cStep_Modify()

    ' Call a private procedure to see if the step text has been
    ' entered - since a global step actually executes a step,
    ' entry of the text is mandatory
    Call StepTextOrFileEntered

    ' Call the Modify method of the step class to carry out the update
    mcStep.Modify

End Sub

Private Property Get cStep_NextStepId() As Long

    cStep_NextStepId = mcStep.NextStepId

End Property

Private Property Set cStep_NodeDB(RHS As DAO.Database)

    Set mcStep.NodeDB = RHS

End Property

Private Property Get cStep_NodeDB() As DAO.Database

    Set cStep_NodeDB = mcStep.NodeDB

End Property

Private Function cStep_OldVersionNo() As String
    cStep_OldVersionNo = mcStep.OldVersionNo
End Function

Private Property Let cStep_OutputFile(ByVal RHS As String)

    mcStep.OutputFile = RHS

End Property

Private Property Get cStep_OutputFile() As String

    cStep_OutputFile = mcStep.OutputFile

End Property

Private Property Let cStep_ParentStepId(ByVal RHS As Long)

    ' A global step cannot have any sub-steps associated with it
    ' Hence, the parent step id and parent version number will be zero
    mcStep.ParentStepId = 0

End Property

Private Property Get cStep_ParentStepId() As Long

    cStep_ParentStepId = mcStep.ParentStepId

End Property

Private Property Let cStep_ParentVersionNo(ByVal RHS As String)

    ' A global step cannot have any sub-steps associated with it
    ' Hence, the parent step id and parent version number will be zero
    mcStep.ParentVersionNo = gstrMinVersion

End Property

Private Property Get cStep_ParentVersionNo() As String

    cStep_ParentVersionNo = mcStep.ParentVersionNo

End Property
```

```
Private Property Let cStep_Position(ByVal RHS As Long)
    mcStep.Position = RHS
End Property
Private Property Get cStep_Position() As Long
    cStep_Position = mcStep.Position
End Property
Private Sub cStep_RemoveIterator(cItRecord As cIterator)
    Call mcStep.RemoveIterator(cItRecord)
End Sub
Private Sub cStep_SaveIterators()
    Call mcStep.SaveIterators
End Sub
Private Property Let cStep_SequenceNo(ByVal RHS As Integer)
    mcStep.SequenceNo = RHS
End Property
Private Property Get cStep_SequenceNo() As Integer
    cStep_SequenceNo = mcStep.SequenceNo
End Property
Private Property Let cStep_StepId(ByVal RHS As Long)
    mcStep.StepId = RHS
End Property
Private Property Get cStep_StepId() As Long
    cStep_StepId = mcStep.StepId
End Property

Private Property Let cStep_StepLabel(ByVal RHS As String)
    mcStep.StepLabel = RHS
End Property
Private Property Get cStep_StepLabel() As String
    cStep_StepLabel = mcStep.StepLabel
End Property
Private Property Let cStep_StartDir(ByVal RHS As String)
    mcStep.StartDir = RHS
End Property
Private Property Get cStep_StartDir() As String
    cStep_StartDir = mcStep.StartDir
End Property
Private Property Let cStep_StepLevel(ByVal RHS As Integer)
    ' A global step cannot have any sub-steps associated with it
    ' Hence, it will always be at step level 0
    mcStep.StepLevel = 0
End Property
Private Property Get cStep_StepLevel() As Integer
    cStep_StepLevel = mcStep.StepLevel
End Property
Private Property Let cStep_StepText(ByVal RHS As String)
    mcStep.StepText = RHS
End Property
Private Property Get cStep_StepText() As String
    cStep_StepText = mcStep.StepText
End Property
Private Property Let cStep_StepTextFile(ByVal RHS As String)
    mcStep.StepTextFile = RHS
End Property
Private Property Get cStep_StepTextFile() As String
    cStep_StepTextFile = mcStep.StepTextFile
End Property
Private Property Let cStep_StepType(RHS As gintStepType)
    mcStep.StepType = gintGlobalStep
End Property
Private Property Get cStep_StepType() As gintStepType
    cStep_StepType = mcStep.StepType
End Property
Private Sub cStep_UnloadIterators()
    Call mcStep.UnloadIterators
End Sub
Private Sub cStep_UpdateIterator(cItRecord As cIterator)
    Call mcStep.UpdateIterator(cItRecord)
End Sub
Private Sub cStep_UpdateIteratorVersion()
    Call mcStep.UpdateIteratorVersion
End Sub

Private Sub cStep_Validate()
    ' The validate routines for each of the steps will
    ' carry out the specific validations for the type and
    ' call the generic validation routine

    On Error GoTo cStep_ValidateErr
    mstrSource = mstrModuleName & "cStep_Validate"

    ' Validations specific to global steps

    ' Check if the step text or a file name has been
    ' specified
    Call StepTextOrFileEntered

    ' The step level must be zero for all globals
```

```
    If mcStep.StepLevel <> 0 Then
        ShowError errStepLevelZeroForGlobal
        On Error GoTo 0
        Err.Raise vbObjectError + errValidateFailed, _
                gstrSource, _
                LoadResString(errValidateFailed)
    End If

    If mcStep.EnabledFlag Then
        ShowError errEnabledFlagFalseForGlobal
        On Error GoTo 0
        Err.Raise vbObjectError + errValidateFailed, _
                gstrSource, _
                LoadResString(errValidateFailed)
    End If

    If mcStep.DegreeParallelism > 0 Then
        ShowError errDegParallelismNullForGlobal
        On Error GoTo 0
        Err.Raise vbObjectError + errValidateFailed, _
                gstrSource, _
                LoadResString(errValidateFailed)
    End If

    If mcStep.ContinuationCriteria > 0 Then
        ShowError errContCriteriaNullForGlobal
        On Error GoTo 0
        Err.Raise vbObjectError + errValidateFailed, _
                gstrSource, _
                LoadResString(errValidateFailed)
    End If

    mcStep.Validate

    Exit Sub

cStep_ValidateErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "cStep_Validate"
    On Error GoTo 0
    Err.Raise vbObjectError + errValidateFailed, _
            mstrSource, _
            LoadResString(errValidateFailed)
End Sub
Private Sub StepTextOrFileEntered()
    ' Checks if either the step text or the name of the file containing
    ' the text has been entered
    ' If both of them are null or both of them are not null,
    ' the global step is invalid and an error is raised

    If StringEmpty(mcStep.StepText) And StringEmpty(mcStep.StepTextFile)
Then
        ShowError errStepTextAndFileNull
        On Error GoTo 0
        Err.Raise vbObjectError + errStepTextAndFileNull,_
            mstrSource, LoadResString(errStepTextAndFileNull)
    ElseIf Not StringEmpty(mcStep.StepText) And Not
StringEmpty(mcStep.StepTextFile)Then
        ShowError errStepTextOrFile
        On Error GoTo 0
        Err.Raise vbObjectError + errStepTextOrFile, _
            mstrSource, LoadResString(errStepTextOrFile)
    End If

End Sub

Private Property Let cStep_VersionNo(ByVal RHS As String)

    mcStep.VersionNo = RHS

End Property

Private Property Get cStep_VersionNo() As String

    cStep_VersionNo = mcStep.VersionNo

End Property

Private Property Let cStep_WorkspaceId(ByVal RHS As Long)

    mcStep.WorkspaceId = RHS

End Property

Private Property Get cStep_WorkspaceId() As Long

    cStep_WorkspaceId = mcStep.WorkspaceId

End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cInstance"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cInstance.cls
'              Microsoft  TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Encapsulates the properties and methods of an instance.
'              An instance is created when a step is executed for a
'              particular iterator value (if applicable) at 'run' time.
'              Contains functions to determine if an instance is running,
'              complete, and so on.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "cInstance."
Private mstrSource As String

Private mcStep As cStep
Public Key As String ' Node key for the step being executed
Public InstanceId As Long
Public ParentInstanceId As Long ' The parent instance
Private mblnNoMoreToStart As Boolean
Private mblnComplete As Boolean
Public StartTime As Currency
Public EndTime As Currency
Public ElapsedTime As Currency
Private mintStatus As InstanceStatus
Public DegreeParallelism As Integer
Private mcIterators As cRunColIt

' A collection of all the sub-steps for this step
Private mcSubSteps As cSubSteps
Public Sub UpdateStartTime(lStepId As Long, Optional ByVal StartTm As
Currency = gdtmEmpty, _
        Optional ByVal EndTm As Currency = gdtmEmpty, _
        Optional ByVal Elapsed As Currency = 0)
    ' We do not maintain start and end timestamps for the constraint
    ' of a step. Hence we check if the process that just started/
    ' terminated is the worker step that is being executed. If so,
    ' we update the start/end time and status on the instance record.

    BugAssert (StartTm <> gdtmEmpty) Or (EndTm <> gdtmEmpty), "Mandatory
parameter missing."

    ' Make sure that we are executing the actual step and not
    ' a pre or post-execution constraint
    If mcStep.StepId = lStepId Then
        If StartTm <> 0 Then
            StartTime = StartTm
            mintStatus = gintRunning
        Else
            EndTime = EndTm
            ElapsedTime = Elapsed
            mintStatus = gintComplete
        End If
```

```
    End If

End Sub
Public Function ValidForIteration(cParentInstance As cInstance, _
        ByVal intConsType As ConstraintType) As Boolean
    ' Returns true if the instance passed in is the first or
    ' last iteration for the step, depending on the constraint type

    Dim cSubStepRec As cSubStep
    Dim vntIterators As Variant

    On Error GoTo ValidForIterationErr

    If cParentInstance Is Nothing Then
        ' This will only be true for the dummy instance, which
        ' cannot have any iterators defined for it
        ValidForIteration = True
        Exit Function
    End If

    vntIterators = mcStep.Iterators

    If Not StringEmpty(mcStep.IteratorName) And Not IsEmpty(vntIterators) Then

        Set cSubStepRec = cParentInstance.QuerySubStep(mcStep.StepId)

        If intConsType = gintPreStep Then
            ' Pre-execution constraints will only be executed
            ' before the first iteration
            If cSubStepRec.LastIterator.IteratorType = gintValue Then
                ValidForIteration = (cSubStepRec.LastIterator.Sequence = _
                    gintMinIteratorSequence)
            Else
                ValidForIteration = (cSubStepRec.LastIterator.Value = _
                    cSubStepRec.LastIterator.RangeFrom)
            End If
        Else
            ' Post-execution constraints will only be executed
            ' after the last iteration - check if there are any
            ' pending iterations
            ValidForIteration = cSubStepRec.NextIteration(mcStep) Is Nothing
        End If
    Else
        ValidForIteration = True
    End If

    Exit Function

ValidForIterationErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    mstrSource = mstrModuleName & "ValidForIteration"
    Err.Raise vbObjectError + errExecInstanceFailed, _
        mstrSource, LoadResString(errExecInstanceFailed)

End Function

Public Sub CreateSubStep(cSubStepDtls As cStep, RunParams As
cArrParameters)

    Dim cNewSubStep As cSubStep

    On Error GoTo CreateSubStepErr

    Set cNewSubStep = New cSubStep

    cNewSubStep.StepId = cSubStepDtls.StepId
    cNewSubStep.TasksComplete = 0
    cNewSubStep.TasksRunning = 0

    ' Initialize the iterator for the instance
    Set cNewSubStep.LastIterator = New cRunItDetails
    Call cNewSubStep.InitializeIt(cSubStepDtls, RunParams)

    ' Add add the substep to the collection
    mcSubSteps.Add cNewSubStep

    Set cNewSubStep = Nothing

    Exit Sub

CreateSubStepErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    mstrSource = mstrModuleName & "CreateSubStep"
    Err.Raise vbObjectError + errProgramError, mstrSource, _
        LoadResString(errProgramError)

End Sub

Public Function QuerySubStep(ByVal SubStepId As Long) As cSubStep
    ' Retrieves the sub-step record for the passed in sub-step id

    Dim lngIndex As Long

    On Error GoTo QuerySubStepErr

    ' Find the sub-step node with the matching step id
    For lngIndex = 0 To mcSubSteps.Count - 1
        If mcSubSteps(lngIndex).StepId = SubStepId Then
            Set QuerySubStep = mcSubSteps(lngIndex)
            Exit For
        End If
    Next lngIndex

    Exit Function

QuerySubStepErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    mstrSource = mstrModuleName & "QuerySubStep"
    Err.Raise vbObjectError + errNavInstancesFailed, _
            mstrSource, LoadResString(errNavInstancesFailed)

End Function
Public Property Let AllStarted(ByVal vdata As Boolean)

    'bugmessage "Set All Started to " & vData & " for : " & _
        mstrKey

    mblnNoMoreToStart = vdata

End Property
Public Property Get AllStarted() As Boolean

    AllStarted = mblnNoMoreToStart

End Property
Public Property Let AllComplete(ByVal vdata As Boolean)

    'bugmessage "Set All Complete to " & vData & " for : " & _
        mstrKey

    mblnComplete = vdata

End Property

Public Property Get AllComplete() As Boolean

    AllComplete = mblnComplete

End Property

Public Sub ChildExecuted(mlngStepId As Long)
    ' This procedure is called when a sub-step executes.

    Dim lngIndex As Long

    On Error GoTo ChildExecutedErr

    BugAssert mcStep.StepType = gintManagerStep

    For lngIndex = 0 To mcSubSteps.Count - 1
```

```
      If mcSubSteps(lngIndex).StepId = mlngStepId Then
         mcSubSteps(lngIndex).TasksRunning = _
               mcSubSteps(lngIndex).TasksRunning + 1
'         BugMessage "Tasks Running for Step Id : " & _
'               CStr(mcSubSteps(lngIndex).StepId) & _
'               " Instance Id: " & InstanceId & _
'               " = " & mcSubSteps(lngIndex).TasksRunning
         Exit For
      End If
   Next lngIndex

   If lngIndex > mcSubSteps.Count - 1 Then
      ' The child step wasn't found - raise an error
      On Error GoTo 0
      Err.Raise vbObjectError + errInvalidChild, mstrModuleName, _
            LoadResString(errInvalidChild)
   End If

   Exit Sub

ChildExecutedErr:
   ' Log the error code raised by Visual Basic
   Call LogErrors(Errors)
   On Error GoTo 0
   Err.Raise vbObjectError + errInstanceOpFailed, mstrModuleName & _
"ChildExecuted", _
         LoadResString(errInstanceOpFailed)

End Sub
Public Sub ChildTerminated(mlngStepIdAs Long)
   ' This procedure is called when any sub-step process
   ' terminates. Note: The TasksComplete field will be
   ' updated only when all the instances for a sub-step
   ' complete execution.
   Dim lngIndex As Long

   On Error GoTo ChildTerminatedErr

   BugAssert mcStep.StepType = gintManagerStep

   For lngIndex = 0 To mcSubSteps.Count - 1

      If mcSubSteps(lngIndex).StepId = mlngStepId Then
         mcSubSteps(lngIndex).TasksRunning = _
               mcSubSteps(lngIndex).TasksRunning - 1
'         BugMessage "Tasks Running for Step Id : " & _
'               CStr(mcSubSteps(lngIndex).StepId) & _
'               " Instance Id: " & InstanceId & _
'               " = " & mcSubSteps(lngIndex).TasksRunning

         BugAssert mcSubSteps(lngIndex).TasksRunning >= 0, _
               "Tasks running for " & CStr(mlngStepId) & _
               " Instance Id " & InstanceId & " is less than 0."
         Exit For
      End If
   Next lngIndex

   If lngIndex > mcSubSteps.Count - 1 Then
      ' The child step wasn't found - raise an error
      On Error GoTo 0
      Err.Raise errInvalidChild, mstrModuleName & "ChildTerminated", _
            LoadResString(errInvalidChild)
   End If

   Exit Sub

ChildTerminatedErr:
   ' Log the error code raised by Visual Basic
   Call LogErrors(Errors)
   On Error GoTo 0
   mstrSource = mstrModuleName & "ChildTerminated"
   Err.Raise vbObjectError + errInstanceOpFailed, mstrSource, _
         LoadResString(errInstanceOpFailed)

End Sub
Public Sub ChildCompleted(mlngStepIdAs Long)
   ' This procedure is called when any a sub-step completes
   ' execution. Note: The TasksComplete field will be
   ' incremented.
   Dim lngIndex As Long

   On Error GoTo ChildCompletedErr

   BugAssert mcStep.StepType = gintManagerStep

   For lngIndex = 0 To mcSubSteps.Count - 1
      BugAssert mcSubSteps(lngIndex).TasksComplete >= 0, _
            "Tasks complete for " & CStr(mcSubSteps(lngIndex).StepId) & _
            " Instance Id " & InstanceId & " is less than 0."

      If mcSubSteps(lngIndex).StepId = mlngStepId Then
         mcSubSteps(lngIndex).TasksComplete = _
               mcSubSteps(lngIndex).TasksComplete + 1
'         BugMessage "Tasks Complete for Step Id : " & _
'               CStr(mcSubSteps(lngIndex).StepId) & _
'               " Instance Id: " & InstanceId & _
'               " = " & mcSubSteps(lngIndex).TasksComplete
         Exit For
      End If
   Next lngIndex

   If lngIndex > mcSubSteps.Count - 1 Then
      ' The child step wasn't found - raise an error
      On Error GoTo 0
      Err.Raise errInvalidChild, mstrModuleName, _
            LoadResString(errInvalidChild)
   End If

   Exit Sub

ChildCompletedErr:
   ' Log the error code raised by Visual Basic
   Call LogErrors(Errors)
   On Error GoTo 0
   Err.Raise vbObjectError + errInstanceOpFailed, mstrModuleName & _
"ChildCompleted", _
         LoadResString(errInstanceOpFailed)

End Sub
Public Sub ChildDeleted(mlngStepIdAs Long)
   ' This procedure is called when a sub-step needs to be re-executed
   ' Note: The TasksComplete field is decremented. We needn't worry about
   ' the TasksRunning field since no steps are currently running.
   Dim lngIndex As Long

   On Error GoTo ChildDeletedErr

   BugAssert mcStep.StepType = gintManagerStep

   For lngIndex = 0 To mcSubSteps.Count - 1

      If mcSubSteps(lngIndex).StepId = mlngStepId Then
         mcSubSteps(lngIndex).TasksRunning = _
               mcSubSteps(lngIndex).TasksRunning - 1

         BugAssert mcSubSteps(lngIndex).TasksRunning >= 0, _
               "Tasks running for " & CStr(mcSubSteps(lngIndex).StepId) & _
               " Instance Id " & InstanceId & " is less than 0."
         Exit For
      End If
   Next lngIndex

   If lngIndex > mcSubSteps.Count - 1 Then
      ' The child step wasn't found - raise an error
      On Error GoTo 0
      Err.Raise errInvalidChild, mstrModuleName, _
            LoadResString(errInvalidChild)
   End If

   Exit Sub

ChildDeletedErr:
   ' Log the error code raised by Visual Basic
   Call LogErrors(Errors)
   On Error GoTo 0
```

```
    Err.Raise vbObjectError + errInstanceOpFailed, mstrModuleName &
"ChildDeleted", _
        LoadResString(errInstanceOpFailed)

End Sub
Private Sub RaiseErrForWorker()

    If mcStep.StepType <> gintManagerStep Then
        On Error GoTo 0
        mstrSource = mstrModuleName & "RaiseErrForWorker"
        Err.Raise vbObjectError + errInvalidForWorker, _
            mstrSource, _
            LoadResString(errInvalidForWorker)
    End If

End Sub

Public Property Get Step() As cStep

    Set Step = mcStep

End Property
Public Property Get Iterators() As cRunColIt

    Set Iterators = mcIterators

End Property
Public Property Get SubSteps() As cSubSteps

    Call RaiseErrForWorker

    Set SubSteps = mcSubSteps

End Property
Public Property Set Step(cRunStep As cStep)

    Set mcStep = cRunStep

End Property

Public Property Set Iterators(cIts As cRunColIt)

    Set mcIterators = cIts

End Property
Public Property Get IsPending() As Boolean
    ' Returns true if the step has any substeps that need
    ' execution
    Dim lngIndex As Long
    Dim lngRunning As Long

    Call RaiseErrForWorker

    If Not mblnComplete And Not mblnNoMoreToStart Then
        ' Get a count of all the substeps that are already being
        ' executed
        lngRunning = 0
        For lngIndex = 0 To mcSubSteps.Count - 1
            lngRunning = lngRunning + mcSubSteps(lngIndex).TasksRunning
        Next lngIndex

        IsPending = (lngRunning < DegreeParallelism)
    Else
        ' This should be sufficient to prove that there r no
        ' more sub-steps to be executed.
        ' mblnComplete: Handles the case where all steps have
        ' been executed
        ' mblnNoMoreToStart: Handles the case where the step
        ' has a degree of parallelism greater than the total
        ' number of sub-steps available to execute
        IsPending = False
    End If

End Property
Public Property Get IsRunning() As Boolean
    ' Returns true if the any one of the substeps is still
    ' executing
    Dim lngIndex As Long

    Call RaiseErrForWorker

    IsRunning = False

    ' If a substep has no currently executing tasks and
    ' the tasks completed is greater than zero, then we can
    ' assume that it has completed execution (otherwise we
    ' would've run a new task the moment one completed!)
    For lngIndex = 0 To mcSubSteps.Count - 1
        If mcSubSteps(lngIndex).TasksRunning > 0 Then
            IsRunning = True
            Exit For
        End If
    Next lngIndex

End Property
Public Property Get TotalRunning() As Long
    ' Returns the total number of substeps that are executing
    Dim lngTotalProcesses As Long
    Dim lngIndex As Long

    Call RaiseErrForWorker

    lngTotalProcesses = 0
    For lngIndex = 0 To mcSubSteps.Count - 1
        BugAssert mcSubSteps(lngIndex).TasksRunning >= 0, _
            "Tasks running for " & CStr(mcSubSteps(lngIndex).StepId) & _
            " is less than 0."
        lngTotalProcesses = lngTotalProcesses +
mcSubSteps(lngIndex).TasksRunning
    Next lngIndex

    TotalRunning = lngTotalProcesses
End Property
Public Property Get RunningForStep(lngSubStepId As Long) As Long
    ' Returns the total number of instances of the substep
    ' that are executing
    Dim lngIndex As Long

    Call RaiseErrForWorker

    For lngIndex = 0 To mcSubSteps.Count - 1
        BugAssert mcSubSteps(lngIndex).TasksRunning >= 0, _
            "Tasks running for " & CStr(mcSubSteps(lngIndex).StepId) & _
            " is less than 0."

        If mcSubSteps(lngIndex).StepId = lngSubStepId Then
            RunningForStep = mcSubSteps(lngIndex).TasksRunning
            Exit For
        End If
    Next lngIndex

    If lngIndex > mcSubSteps.Count - 1 Then
        ' The child step wasn't found - raise an error
        On Error GoTo 0
        Err.Raise errInvalidChild, mstrSource, _
            LoadResString(errInvalidChild)
    End If

End Property

Public Property Let Status(ByVal vdata As InstanceStatus)

    mintStatus = vdata

End Property

Public Property Get Status() As InstanceStatus

    Status = mintStatus

End Property
Private Sub Class_Initialize()

    Set mcSubSteps = New cSubSteps

    mblnNoMoreToStart = False
```

```
    mblnComplete = False
    StartTime = gdtmEmpty
    EndTime = gdtmEmpty

End Sub

Private Sub Class_Terminate()

    mcSubSteps.Clear
    Set mcSubSteps = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cInstances"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cInstances.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Implements a collection of cInstance objects.
'              Type-safe wrapper around cVector.
'              Also contains additional functions to query an instance, etc.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "cInstance."
Private mstrSource As String

Private mcInstances As cVector

Public Function QueryInstance(ByVal InstanceId As Long) As cInstance
    ' Retrieves the record for the passed in instance from
    ' the collection

    Dim lngIndex As Long

    On Error GoTo QueryInstanceErr

    ' Check for valid values of the instance id
    If InstanceId > 0 Then
        ' Find the run node with the matching step id
        For lngIndex = 0 To Count() - 1
            If mcInstances(lngIndex).InstanceId = InstanceId Then
                Set QueryInstance = mcInstances(lngIndex)
                Exit For
            End If
        Next lngIndex

        If lngIndex > mcInstances.Count - 1 Then
            On Error GoTo 0
            Err.Raise vbObjectError + errQueryFailed, mstrSource, _
                LoadResString(errQueryFailed)
        End If
    Else
        On Error GoTo 0
        Err.Raise vbObjectError + errQueryFailed, mstrSource, _
            LoadResString(errQueryFailed)
    End If

    Exit Function

QueryInstanceErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    mstrSource = mstrModuleName & "QueryInstance"
    Err.Raise vbObjectError + errQueryFailed, _
        mstrSource, LoadResString(errQueryFailed)

End Function

Public Function QueryPendingInstance(ByVal ParentInstanceId As Long, _
    ByVal lngSubStepId As Long) As cInstance
    ' Retrieves a pending instance for the passed in substep
    ' and the given parent instance id.

    Dim lngIndex As Long

    On Error GoTo QueryPendingInstanceErr

    ' Find the run node with the matching step id
    For lngIndex = 0 To Count() - 1
        If mcInstances(lngIndex).ParentInstanceId = ParentInstanceId And _
            mcInstances(lngIndex).Step.StepId = lngSubStepId Then
            ' Put in a separate if condition since the IsPending
            ' property is valid only for manager steps. If the
            ' calling procedure does not pass a manager step
            ' identifier, the procedure will error out.
            If mcInstances(lngIndex).IsPending Then
                Set QueryPendingInstance = mcInstances(lngIndex)
                Exit For
            End If
        End If
    Next lngIndex

    Exit Function

QueryPendingInstanceErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    mstrSource = mstrModuleName & "QueryPendingInstance"
    Err.Raise vbObjectError + errQueryFailed, _
        mstrSource, LoadResString(errQueryFailed)

End Function
Public Function InstanceAborted(cSubStepRec As cSubStep) As Boolean

    Dim lIndex As Long

    InstanceAborted = False

    For lIndex = 0 To Count() - 1
        If mcInstances(lIndex).Step.StepId = cSubStepRec.StepId And _
            mcInstances(lIndex).Status = gintAborted Then
            InstanceAborted = True
            Exit For
        End If
    Next lIndex

End Function
Public Function CompletedInstanceExists(lParentInstance As Long, _
    cSubStepDtls As cStep) As Boolean
    ' Checks if there is a completed instance of the passed in step

    Dim lngIndex As Long

    CompletedInstanceExists = False

    If cSubStepDtls.StepType = gintManagerStep Then
        ' Find the run node with the matching step id
        For lngIndex = 0 To Count() - 1
            If mcInstances(lngIndex).ParentInstanceId = lParentInstance And _
                mcInstances(lngIndex).Step.StepId = cSubStepDtls.StepId Then
                ' Put in a separate if condition since the IsPending
                ' property is valid only for manager steps.
                BugAssert (Not mcInstances(lngIndex).IsPending),"Pending instance
exists!"

                CompletedInstanceExists = True
                Exit Function
            End If
        Next lngIndex
    End If
```

```
End Function
Public Sub Add(ByVal objItem As cInstance)

    mcInstances.Add objItem

End Sub


Public Sub Clear()

    mcInstances.Clear

End Sub


Public Function Count() As Long

    Count = mcInstances.Count

End Function


Public Function Delete(ByVal lngDelete As Long) As cInstance

    Set Delete = mcInstances.Delete(lngDelete)

End Function


Public Property Set Item(Optional ByVal Position As Long, _
        RHS As cInstance)

    If Position = -1 Then
        Position = 0
    End If
    Set mcInstances(Position) = RHS

End Property

Public Property Get Item(Optional ByVal Position As Long = -1) _
        As cInstance
Attribute Item.VB_UserMemId = 0

    If Position = -1 Then
        Position = 0
    End If
    Set Item = mcInstances.Item(Position)

End Property

Private Sub Class_Initialize()

    Set mcInstances = New cVector

End Sub


Private Sub Class_Terminate()

    Set mcInstances = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cIterator"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cIterator.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Encapsulates the properties and methods of an iterator.
'              Contains functions to insert, update and delete
'              iterator_values records from the database.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Implements cNode

' Module level variables to store the property values
Private mintType As Integer
Private mintSequenceNo As Integer
Private mstrValue As String
Private mdbsIteratorDB As Database
Private mintOperation As Integer
Private mlngPosition As Long

Private Const mstrModuleName As String = "cIterator."
Private mstrSource As String

Public Enum ValueType
    gintFrom = 1
    gintTo
    gintStep
    gintValue
End Enum
Public Property Get Value() As String

    Value = mstrValue

End Property
Public Property Let Value(ByVal vdata As String)

    mstrValue = vdata

End Property

Public Property Get IndOperation() As Operation

    IndOperation = mintOperation

End Property
Public Property Let IndOperation(ByVal vdata As Operation)

    On Error GoTo IndOperationErr
    mstrSource = mstrModuleName & "IndOperation"

    ' The valid operations are define in the cOperations
    ' class. Check if the operation is valid
    Select Case vdata
        Case QueryOp, InsertOp, UpdateOp, DeleteOp
            mintOperation = vdata

        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errInvalidOperation, _
                mstrSource, LoadResString(errInvalidOperation)
    End Select

    Exit Property

IndOperationErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "IndOperation"
    On Error GoTo 0
    Err.Raise vbObjectError + errLetOperationFailed, _
        mstrSource, LoadResString(errLetOperationFailed)

End Property

Public Function Clone() As cIterator

    ' Creates a copy of a given Iterator

    Dim cItClone As cIterator

    On Error GoTo CloneErr

    Set cItClone = New cIterator
```

```
    ' Copy all the iterator properties to the newly
    ' created object
    cItClone.IteratorType = mintType
    cItClone.SequenceNo = mintSequenceNo
    cItClone.IndOperation = mintOperation
    cItClone.Value = mstrValue

    ' And set the return value to the newly created Iterator
    Set Clone = cItClone

    Exit Function

CloneErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Clone"
    On Error GoTo 0
    Err.Raise vbObjectError + errCloneFailed, _
        mstrSource, LoadResString(errCloneFailed)

End Function
Public Property Get SequenceNo() As Integer

    SequenceNo = mintSequenceNo

End Property

Public Property Let SequenceNo(ByVal vdata As Integer)
    mintSequenceNo = vdata
End Property

Public Sub Add(ByVal lngStepId As Long, _
    strVersion As String)
    ' Inserts a new iterator values record into the database

    Dim strInsert As String
    Dim qy As DAO.QueryDef

    On Error GoTo AddIteratorErr

    ' First check if the database object is valid
    Call CheckDB

    ' Create a temporary querydef object
    strInsert = "insert into iterator_values " & _
        "( step_id, version_no, type, " & _
        " iterator_value, sequence_no ) " & _
        " values ( [st_id], [ver_no], [it_typ], " & _
        " [it_val], [seq_no] )"
    Set qy = mdbsIteratorDB.CreateQueryDef(gstrEmptyString,strInsert)

    ' Call a procedure to execute the Querydef object
    Call AssignParameters(qy, lngStepId, strVersion)

    qy.Execute dbFailOnError
    qy.Close

    Exit Sub

AddIteratorErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "AddIterator"
    On Error GoTo 0
    Err.Raise vbObjectError + errInsertIteratorFailed, _
        mstrSource, _
        LoadResString(errInsertIteratorFailed)
End Sub

Private Sub AssignParameters(qyExec As DAO.QueryDef, _
    ByVal lngStepId As Long, _
    strVersion As String)
    ' Assigns values to the parameters in the querydef object
    ' The parameter names are cryptic to make them different
    ' from the field names. When the parameter names are
    ' the same as the field names, parameters in the where
    ' clause do not get created.

    Dim prmParam As DAO.Parameter

    On Error GoTo AssignParametersErr
    mstrSource = mstrModuleName & "AssignParameters"

    For Each prmParam In qyExec.Parameters
        Select Case prmParam.Name
            Case "[st_id]"
                prmParam.Value = lngStepId

            Case "[ver_no]"
                prmParam.Value = strVersion

            Case "[it_typ]"
                prmParam.Value = mintType

            Case "[it_val]"
                prmParam.Value = mstrValue

            Case "[seq_no]"
                prmParam.Value = mintSequenceNo

            Case Else
                ' Write the parameter name that is faulty
                WriteError errInvalidParameter, mstrSource, _
                    prmParam.Name
                On Error GoTo 0
                Err.Raise errInvalidParameter, mstrSource, _
                    LoadResString(errInvalidParameter)
        End Select
    Next prmParam

    Exit Sub

AssignParametersErr:

    mstrSource = mstrModuleName & "AssignParameters"
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errAssignParametersFailed, _
        mstrSource, LoadResString(errAssignParametersFailed)

End Sub
Private Sub CheckDB()
    ' Check if the database object has been initialized

    If mdbsIteratorDB Is Nothing Then
        ShowError errInvalidDB
        On Error GoTo 0
        Err.Raise vbObjectError + errInvalidDB, _
            mstrModuleName, LoadResString(errInvalidDB)
    End If

End Sub

Public Sub Delete(ByVal lngStepId As Long, _
    strVersion As String)
    ' Deletes the step iterator record from the database

    Dim strDelete As String
    Dim qy As DAO.QueryDef

    On Error GoTo DeleteIteratorErr
    mstrSource = mstrModuleName & "DeleteIterator"

    ' There can be multiple iterators for a step.
    ' However the values that an iterator for a step can
    ' assume will be unique, meaning that a combination of
    ' the iterator_id and value will be unique.
    strDelete = "delete from iterator_values " & _
        " where step_id = [st_id]" & _
        " and version_no = [ver_no]" & _
        " and iterator_value = [it_val] "
    Set qy = mdbsIteratorDB.CreateQueryDef(gstrEmptyString,strDelete)

    Call AssignParameters(qy, lngStepId, strVersion)
    qy.Execute dbFailOnError

    qy.Close
```

```
    Exit Sub

DeleteIteratorErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "DeleteIterator"
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteIteratorFailed, _
            mstrSource, _
            LoadResString(errDeleteIteratorFailed)
End Sub
Public Sub Update(ByVal lngStepId As Long, strVersion As String)
    ' Updates the sequence no of the step iterator record
    ' in the database

    Dim strUpdate As String
    Dim qy As QueryDef

    On Error GoTo UpdateErr

    ' First check if the database object is valid
    Call CheckDB

    If mintType = gintValue Then
        ' If the iterator is of type value, only the sequence of the values can get
updated
        strUpdate = "Update iterator_values " & _
                " set sequence_no = [seq_no] " & _
                " where step_id = [st_id]" & _
                " and version_no = [ver_no]" & _
                " and iterator_value = [it_val] "
    Else
        ' If the iterator is of type range, only the values can get updated
        strUpdate = "Update iterator_values " & _
                " set iterator_value = [it_val] " & _
                " where step_id = [st_id]" & _
                " and version_no = [ver_no]" & _
                " and type = [it_typ] "
    End If

    Set qy = mdbsIteratorDB.CreateQueryDef(gstrEmptyString,strUpdate)

    ' Call a procedure to assign the parameter values to the
    ' querydef object
    Call AssignParameters(qy, lngStepId, strVersion)
    qy.Execute dbFailOnError

    qy.Close

    Exit Sub

UpdateErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Update"
    On Error GoTo 0
    Err.Raise vbObjectError + errUpdateConstraintFailed, _
            mstrSource, _
            LoadResString(errUpdateConstraintFailed)

End Sub
Public Property Set NodeDB(vdata As Database)

    Set mdbsIteratorDB = vdata

End Property

Public Property Get NodeDB() As Database

    Set NodeDB = mdbsIteratorDB

End Property
Public Property Get Position() As Long

    Position = mlngPosition

End Property
Public Property Let Position(ByVal vdata As Long)

    mlngPosition = vdata

End Property

Public Property Let IteratorType(ByVal vdata As ValueType)

    On Error GoTo TypeErr
    mstrSource = mstrModuleName & "Type"

    ' These constants have been defined in the enumeration,
    ' Type, which is exposed
    Select Case vdata
        Case gintFrom, gintTo, gintStep, gintValue
            mintType = vdata

        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errTypeInvalid, _
                mstrSource, LoadResString(errTypeInvalid)
    End Select

    Exit Property

TypeErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Type"
    On Error GoTo 0
    Err.Raise vbObjectError + errTypeInvalid, _
        mstrSource, LoadResString(errTypeInvalid)

End Property

Public Property Get IteratorType() As ValueType

    IteratorType = mintType

End Property
Public Sub Validate()

    ' No validations necessary for the iterator class

End Sub

Private Sub Class_Initialize()

    ' Initialize the operation indicator variable to Query
    ' It will be modified later by the collection class when
    ' inserts, updates or deletes are performed
    mintOperation = QueryOp

End Sub


Private Property Let cNode_IndOperation(ByVal vdata As Operation)

    On Error GoTo IndOperationErr
    mstrSource = mstrModuleName & "IndOperation"

    ' The valid operations are define in the cOperations
    ' class. Check if the operation is valid
    Select Case vdata
        Case QueryOp, InsertOp, UpdateOp, DeleteOp
            mintOperation = vdata

        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errInvalidOperation, _
                    mstrSource, LoadResString(errInvalidOperation)
    End Select

    Exit Property

IndOperationErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "IndOperation"
    On Error GoTo 0
    Err.Raise vbObjectError + errLetOperationFailed, _
        mstrSource, LoadResString(errLetOperationFailed)
```

```
End Property

Private Property Get cNode_IndOperation() As Operation

    IndOperation = mintOperation

End Property

Private Property Set cNode_NodeDB(RHS As DAO.Database)

    Set mdbsIteratorDB = RHS

End Property

Private Property Get cNode_NodeDB() As DAO.Database

    Set cNode_NodeDB = mdbsIteratorDB

End Property


Private Property Let cNode_Position(ByVal vdata As Long)

    mlngPosition = vdata

End Property


Private Property Get cNode_Position() As Long

    cNode_Position = mlngPosition

End Property


Private Sub cNode_Validate()

    ' No validations necessary for the iterator class

End Sub


Private Property Let cNode_Value(ByVal vdata As String)

    mstrValue = vdata

End Property

Private Property Get cNode_Value() As String

    Value = mstrValue

End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cManager"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
' FILE:       cManager.cls
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
' PURPOSE:    Encapsulates the properties and methods of a manager step.
'             Implements the cStep class - carries out initializations
'             and validations that are specific to manager steps.
' Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Implements cStep

' Object variable to keep the step reference in
Private mcStep As cStep

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cManager."

Private Property Let cStep_StartDir(ByVal RHS As String)

    mcStep.StartDir = RHS

End Property

Private Property Get cStep_StartDir() As String

    cStep_StartDir = mcStep.StartDir

End Property
Private Sub cStep_Delete()

    Call mcStep.Delete

End Sub

Private Property Set cStep_NodeDB(RHS As DAO.Database)

    Set mcStep.NodeDB = RHS

End Property

Private Function cStep_IncVersionY() As String

    cStep_IncVersionY = mcStep.IncVersionY

End Function
Private Function cStep_IsNewVersion() As Boolean
    cStep_IsNewVersion = mcStep.IsNewVersion
End Function
Private Function cStep_OldVersionNo() As String
    cStep_OldVersionNo = mcStep.OldVersionNo
End Function

Private Function cStep_IncVersionX() As String

    cStep_IncVersionX = mcStep.IncVersionX

End Function
Private Sub cStep_UpdateIteratorVersion()

    Call mcStep.UpdateIteratorVersion

End Sub

Private Function cStep_IteratorCount() As Long

    cStep_IteratorCount = mcStep.IteratorCount

End Function

Private Sub cStep_UnloadIterators()

    Call mcStep.UnloadIterators

End Sub

Private Sub cStep_DeleteIterator(cItRecord As cIterator)

    Call mcStep.DeleteIterator(cItRecord)

End Sub
Private Property Get cStep_IteratorName() As String

    cStep_IteratorName = mcStep.IteratorName

End Property
Private Property Let cStep_IteratorName(ByVal RHS As String)
```

```
    mcStep.IteratorName = RHS

End Property

Private Sub cStep_SaveIterators()

    Call mcStep.SaveIterators

End Sub
Private Sub cStep_LoadIterator(cItRecord As cIterator)

    Call mcStep.LoadIterator(cItRecord)

End Sub

Private Property Let cStep_Position(ByVal RHS As Long)

    mcStep.Position = RHS

End Property
Private Sub cStep_InsertIterator(cItRecord As cIterator)

    Call mcStep.InsertIterator(cItRecord)

End Sub
Private Function cStep_Iterators() As Variant

    cStep_Iterators = mcStep.Iterators

End Function
Private Sub cStep_ModifyIterator(cItRecord As cIterator)

    Call mcStep.ModifyIterator(cItRecord)

End Sub
Private Sub cStep_RemoveIterator(cItRecord As cIterator)

    Call mcStep.RemoveIterator(cItRecord)

End Sub
Private Sub cStep_UpdateIterator(cItRecord As cIterator)

    Call mcStep.UpdateIterator(cItRecord)

End Sub
Private Sub cStep_AddIterator(cItRecord As cIterator)

    Call mcStep.AddIterator(cItRecord)

End Sub

Private Property Get cStep_Position() As Long

    cStep_Position = mcStep.Position

End Property

Private Function cStep_Clone(Optional cCloneStep As cStep) As cStep

    Dim cNewManager As cManager

    Set cNewManager = New cManager
    Set cStep_Clone = mcStep.Clone(cNewManager)

End Function

Private Property Get cStep_IndOperation() As Operation

    cStep_IndOperation = mcStep.IndOperation

End Property

Private Property Let cStep_IndOperation(ByVal RHS As Operation)

    mcStep.IndOperation = RHS

End Property

Private Property Get cStep_NextStepId() As Long

    cStep_NextStepId = mcStep.NextStepId

End Property

Private Property Let cStep_OutputFile(ByVal RHS As String)

    mcStep.OutputFile = RHS

End Property

Private Property Get cStep_OutputFile() As String

    cStep_OutputFile = mcStep.OutputFile

End Property

Private Property Let cStep_ErrorFile(ByVal RHS As String)

    mcStep.ErrorFile = RHS

End Property

Private Property Get cStep_ErrorFile() As String

    cStep_ErrorFile = mcStep.ErrorFile

End Property
'Private Property Let cStep_LogFile(ByVal RHS As String)
'
'    mcStep.LogFile = RHS
'
'End Property
'
'Private Property Get cStep_LogFile() As String
'
'    cStep_LogFile = mcStep.LogFile
'
'End Property

Private Property Let cStep_ArchivedFlag(ByVal RHS As Boolean)

    mcStep.ArchivedFlag = RHS

End Property

Private Property Get cStep_ArchivedFlag() As Boolean

    cStep_ArchivedFlag = mcStep.ArchivedFlag

End Property

Private Property Get cStep_NodeDB() As DAO.Database

    Set cStep_NodeDB = mcStep.NodeDB

End Property

Private Sub Class_Initialize()

    ' Create the object
    Set mcStep = New cStep

    ' Initialize the object with valid values for a manager step
    ' The global flag should be the first field to be initialized
    ' since subsequent validations might try to check if the
    ' step being created is global
    mcStep.GlobalFlag = False
'   mcStep.GlobalRunMethod = gintNoOption
    mcStep.StepType = gintManagerStep

    ' Since the manager step does not take any action, the step
    ' text and file name will always be empty
    mcStep.StepText = gstrEmptyString
    mcStep.StepTextFile = gstrEmptyString

    ' Since the manager step does not take any action, execution
```

```
    ' properties for the step will be empty
    mcStep.ExecutionMechanism = gintNoOption
    mcStep.FailureDetails = gstrEmptyString
    mcStep.ContinuationCriteria = gintNoOption

End Sub
Private Sub Class_Terminate()

    ' Remove the step object
    Set mcStep = Nothing

End Sub
Private Sub cStep_Add()

    ' Call the Add method of the step class to carry out the insert
    mcStep.Add

End Sub
Private Property Get cStep_ContinuationCriteria() As ContinuationCriteria

    cStep_ContinuationCriteria = mcStep.ContinuationCriteria

End Property

Private Property Let cStep_ContinuationCriteria(ByVal RHS As
ContinuationCriteria)

    ' Since a manager step cannot take any action, the continuation
    ' criteria property does not apply to it
    mcStep.ContinuationCriteria = gintNoOption

End Property

Private Property Let cStep_DegreeParallelism(ByVal RHS As String)

    mcStep.DegreeParallelism = RHS

End Property

Private Property Get cStep_DegreeParallelism() As String

    cStep_DegreeParallelism = mcStep.DegreeParallelism

End Property

Private Sub cStep_DeleteStep()

    On Error GoTo cStep_DeleteStepErr
    mstrSource = mstrModuleName & "cStep_DeleteStep"

    mcStep.Delete
    Exit Sub

cStep_DeleteStepErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "cStep_DeleteStep"
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteStepFailed, _
          mstrSource, _
          LoadResString(errDeleteStepFailed)

End Sub

Private Property Get cStep_EnabledFlag() As Boolean

    cStep_EnabledFlag = mcStep.EnabledFlag

End Property

Private Property Let cStep_EnabledFlag(ByVal RHS As Boolean)

    mcStep.EnabledFlag = RHS

End Property

Private Property Let cStep_ExecutionMechanism(ByVal RHS As
ExecutionMethod)

    ' Since a manager step cannot take any action, the Execution
    ' Mechanism property does not apply to it
    mcStep.ExecutionMechanism = gintNoOption

End Property

Private Property Get cStep_ExecutionMechanism() As ExecutionMethod

    cStep_ExecutionMechanism = mcStep.ExecutionMechanism

End Property

Private Property Let cStep_FailureDetails(ByVal RHS As String)

    ' Since a manager step cannot take any action, the Failure
    ' Details property does not apply to it
    mcStep.FailureDetails = gstrEmptyString

End Property

Private Property Get cStep_FailureDetails() As String

    cStep_FailureDetails = mcStep.FailureDetails

End Property

Private Property Get cStep_GlobalFlag() As Boolean

    cStep_GlobalFlag = mcStep.GlobalFlag

End Property

Private Property Let cStep_GlobalFlag(ByVal RHS As Boolean)

    ' Set the global flag to false - this flag is initialized when
    ' an instance of the class is created. Just making sure that
    ' nobody changes the value inadvertently
    mcStep.GlobalFlag = False

End Property
Private Sub cStep_Modify()

    ' Call the Modify method of the step class to carry out the update
    mcStep.Modify

End Sub

Private Property Let cStep_ParentStepId(ByVal RHS As Long)

    mcStep.ParentStepId = RHS

End Property

Private Property Get cStep_ParentStepId() As Long

    cStep_ParentStepId = mcStep.ParentStepId

End Property

Private Property Let cStep_ParentVersionNo(ByVal RHS As String)

    mcStep.ParentVersionNo = RHS

End Property

Private Property Get cStep_ParentVersionNo() As String

    cStep_ParentVersionNo = mcStep.ParentVersionNo

End Property

Private Property Let cStep_SequenceNo(ByVal RHS As Integer)

    mcStep.SequenceNo = RHS

End Property

Private Property Get cStep_SequenceNo() As Integer
```

```
    cStep_SequenceNo = mcStep.SequenceNo

End Property

Private Property Let cStep_StepId(ByVal RHS As Long)

    mcStep.StepId = RHS

End Property

Private Property Get cStep_StepId() As Long

    cStep_StepId = mcStep.StepId

End Property


Private Property Let cStep_StepLabel(ByVal RHS As String)

    mcStep.StepLabel = RHS

End Property

Private Property Get cStep_StepLabel() As String

    cStep_StepLabel = mcStep.StepLabel

End Property


Private Property Let cStep_StepLevel(ByVal RHS As Integer)

    mcStep.StepLevel = RHS

End Property

Private Property Get cStep_StepLevel() As Integer

    cStep_StepLevel = mcStep.StepLevel

End Property

Private Property Let cStep_StepText(ByVal RHS As String)

    ' Since the manager step does not take any action, the step
    ' text and file name will always be empty
    mcStep.StepText = gstrEmptyString

End Property

Private Property Get cStep_StepText() As String

    cStep_StepText = mcStep.StepText

End Property

Private Property Let cStep_StepTextFile(ByVal RHS As String)

    ' Since the manager step does not take any action, the step
    ' text and file name will always be empty
    mcStep.StepTextFile = gstrEmptyString

End Property

Private Property Get cStep_StepTextFile() As String

    cStep_StepTextFile = mcStep.StepTextFile

End Property

Private Property Let cStep_StepType(RHS As gintStepType)

    mcStep.StepType = gintManagerStep

End Property

Private Property Get cStep_StepType() As gintStepType

    cStep_StepType = mcStep.StepType

End Property

Private Sub cStep_Validate()
    ' The validate routines for each of the steps will
    ' carry out the specific validations for the type and
    ' call the generic validation routine

    On Error GoTo cStep_ValidateErr
    mstrSource = mstrModuleName & "cStep_Validate"

    ' Validations specific to manager steps

    ' Check if the step text or a file name has been
    ' specified
    If Not StringEmpty(mcStep.StepText)Or Not
StringEmpty(mcStep.StepTextFile)Then
        ShowError errTextAndFileNullForManager
        On Error GoTo 0
        Err.Raise vbObjectError + errValidateFailed, _
            gstrSource, _
            LoadResString(errValidateFailed)
    End If

    If mcStep.ExecutionMechanism <> gintNoOption Then
        ShowError errExecutionMechanismInvalid
        On Error GoTo 0
        Err.Raise vbObjectError + errValidateFailed, _
            gstrSource, _
            LoadResString(errValidateFailed)
    End If

    If mcStep.FailureDetails <> gstrEmptyString Then
        ShowError errFailureDetailsNullForMgr
        On Error GoTo 0
        Err.Raise vbObjectError + errValidateFailed, _
            gstrSource, _
            LoadResString(errValidateFailed)
    End If

    If mcStep.ContinuationCriteria <> gintNoOption Then
        ShowError errContCriteriaInvalid
        On Error GoTo 0
        Err.Raise vbObjectError + errValidateFailed, _
            gstrSource, _
            LoadResString(errValidateFailed)
    End If

    mcStep.Validate

    Exit Sub

cStep_ValidateErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "cStep_Validate"
    On Error GoTo 0
    Err.Raise vbObjectError + errValidateFailed, _
        mstrSource, _
        LoadResString(errValidateFailed)
End Sub

Private Property Let cStep_VersionNo(ByVal RHS As String)

    mcStep.VersionNo = RHS

End Property

Private Property Get cStep_VersionNo() As String

    cStep_VersionNo = mcStep.VersionNo

End Property

Private Property Let cStep_WorkspaceId(ByVal RHS As Long)

    mcStep.WorkspaceId = RHS
```

```
End Property

Private Property Get cStep_WorkspaceId() As Long

    cStep_WorkspaceId = mcStep.WorkspaceId

End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cNode"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cNode.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:     Defines the properties that an object has to implement.
'  Contact:     Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Public Property Get IndOperation() As Operation
End Property
Public Property Let IndOperation(ByVal vdata As Operation)
End Property
Public Sub Validate()
End Sub
Public Property Get Value() As String
End Property
Public Property Let Value(ByVal vdata As String)
End Property

Public Property Get NodeDB() As Database
End Property
Public Property Set NodeDB(vdata As Database)
End Property

Public Property Get Position() As Long
End Property
Public Property Let Position(ByVal vdata As Long)
End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cNodeCollections"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cNodeCollections.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Implements an array of objects.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Node counter
Private mlngNodeCount As Long
Private mdbsNodeDb As Database
Private mcarrNodes() As Object

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cNodeCollections."

Public Property Set Item(ByVal Position As Long, _
        ByVal objNode As Object)

    ' Returns the element at the passed in position in the array
    If Position >= 0 And Position < mlngNodeCount Then
        Set mcarrNodes(Position) = objNode
    Else
        On Error GoTo 0
        Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
            LoadResString(errItemDoesNotExist)
    End If

End Property
Public Property Get Item(ByVal Position As Long) As Object
Attribute Item.VB_UserMemId = 0

    ' Returns the element at the passed in position in the array
    If Position >= 0 And Position < mlngNodeCount Then
        Set Item = mcarrNodes(Position)
    Else
        On Error GoTo 0
        Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
            LoadResString(errItemDoesNotExist)
    End If

End Property

Public Sub Commit(ByVal cSaveObj As Object, _
        ByVal lngIndex As Long)
    ' This procedure checks if any changes have been made to the
    ' passed in object. If so, it calls the corresponding method
    ' to commit the changes.

    On Error GoTo CommitErr
    mstrSource = mstrModuleName & "Commit"

    Select Case cSaveObj.IndOperation
        Case QueryOp
            ' No changes were made to the queried parameter.
            ' Do nothing

        Case InsertOp
            cSaveObj.Add
            cSaveObj.IndOperation = QueryOp

        Case UpdateOp
            cSaveObj.Modify
            cSaveObj.IndOperation = QueryOp

        Case DeleteOp
            cSaveObj.Delete
            ' Now we can remove the record from the array
            Call Unload(lngIndex)

    End Select

    Exit Sub

CommitErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Commit"
    On Error GoTo 0
    Err.Raise vbObjectError + errCommitFailed, _
            mstrSource, _
            LoadResString(errCommitFailed)

End Sub
Public Sub Save(ByVal lngWorkspace As Long)
    ' Calls a procedure to commit all changes for the passed
    ' in workspace.

    Dim lngIndex As Long

    On Error GoTo SaveErr

    ' Find all parameters in the array with a matching workspace id
    ' It is important to step backwards through the array, since
    ' we delete parameter records as we go along!
```

```
    For lngIndex = mlngNodeCount - 1 To 0 Step -1
        If mcarrNodes(lngIndex).WorkspaceId = lngWorkspace Then

            ' Call a procedure to commit all changes to the
            ' parameter record, if any
            Call Commit(mcarrNodes(lngIndex),lngIndex)

        End If
    Next lngIndex

    Exit Sub

SaveErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Save"
    On Error GoTo 0
    Err.Raise vbObjectError + errSaveFailed, _
            mstrSource, _
            LoadResString(errSaveFailed)

End Sub
Public Property Get Count() As Long

    Count = mlngNodeCount

End Property

Public Property Get NodeDB() As Database

    Set NodeDB = mdbsNodeDb

End Property
Public Property Set NodeDB(vdata As Database)

    Set mdbsNodeDb = vdata

End Property

Public Sub Load(cNodeToLoad As Object)
    ' Adds the passed in object to the array

    On Error GoTo LoadErr

    ' If this procedure is called by the add to array procedure,
    ' the database object has already been initialized
    If cNodeToLoad.NodeDB Is Nothing Then

        ' All the Nodes will be initialized with the database
        ' objects before being added to the array
        Set cNodeToLoad.NodeDB = mdbsNodeDb

    End If

    ReDim Preserve mcarrNodes(mlngNodeCount)

    ' Set the newly added element in the array to the passed in Node
    cNodeToLoad.Position = mlngNodeCount
    Set mcarrNodes(mlngNodeCount) = cNodeToLoad

    mlngNodeCount = mlngNodeCount + 1

    Exit Sub

LoadErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadFailed, mstrModuleName & "Load", _
            LoadResString(errLoadFailed)

End Sub
Public Sub Unload(lngDeletePosition As Long)
    ' Unloads the passed in object from the array

    On Error GoTo UnloadErr

    If lngDeletePosition < (mlngNodeCount - 1) Then

        ' Set the Node at the position being deleted to
        ' the last Node in the Node array
        Set mcarrNodes(lngDeletePosition) = mcarrNodes(mlngNodeCount - 1)
        mcarrNodes(lngDeletePosition).Position = lngDeletePosition
    End If

    ' Delete the last Node from the array
    mlngNodeCount = mlngNodeCount - 1
    If mlngNodeCount > 0 Then
        ReDim Preserve mcarrNodes(0 To mlngNodeCount - 1)
    Else
        ReDim mcarrNodes(0)
    End If

    Exit Sub

UnloadErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Unload"
    On Error GoTo 0
    Err.Raise vbObjectError + errUnloadFailed, _
            mstrSource, _
            LoadResString(errUnloadFailed)

End Sub
Public Sub Delete(lngDeletePosition As Long)
    ' Deletes the object at the specified position in the
    ' array

    Dim cDeleteObj As Object

    On Error GoTo DeleteErr
    mstrSource = mstrModuleName & "Delete"

    Set cDeleteObj = mcarrNodes(lngDeletePosition)

    If cDeleteObj.IndOperation = InsertOp Then
        ' If we are deleting a record that has just been inserted,
        ' blow it away
        Call Unload(lngDeletePosition)
    Else
        ' Set the operation for the deleted object to indicate a
        ' delete - we actually delete the element only at the time
        ' of a save operation
        cDeleteObj.IndOperation = DeleteOp
    End If

    Exit Sub

DeleteErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Delete"
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteFailed, _
            mstrSource, _
            LoadResString(errDeleteFailed)

End Sub
Public Sub Modify(cModifiedNode As Object)
    ' Sets the object at the passed in position to the
    ' modified object passed in

    On Error GoTo ModifyErr

    ' First check if the record is valid - all objects that
    ' use this collection class must have a Validate routine
    cModifiedNode.Validate

    ' If we are updating a record that hasn't yet been inserted,
    ' do not change the operation indicator - or we try to update
    ' a non-existant record
    If cModifiedNode.IndOperation <> InsertOp Then
        ' Set the operations to indicate an update
        cModifiedNode.IndOperation = UpdateOp
    End If

    ' Modify the object at the queried position - the Position
    ' will be maintained by this class
    Set mcarrNodes(cModifiedNode.Position) = cModifiedNode
```

```
    Exit Sub

ModifyErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Modify"
    On Error GoTo 0
    Err.Raise vbObjectError + errModifyFailed, _
        mstrSource, _
        LoadResString(errModifyFailed)

End Sub
Public Sub Add(cNodeToAdd As Object)

    On Error GoTo AddErr

    Set cNodeToAdd.NodeDB = mdbsNodeDb

    ' First check if the record is valid
    cNodeToAdd.Validate

    ' Set the operation to indicate an insert
    cNodeToAdd.IndOperation = InsertOp

    ' Call a procedure to load the record in the array
    Call Load(cNodeToAdd)

    Exit Sub

AddErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Add"
    On Error GoTo 0
    Err.Raise vbObjectError + errAddFailed, _
        mstrSource, _
        LoadResString(errAddFailed)

End Sub

Private Sub Class_Terminate()

    ReDim mcarrNodes(0)
    mlngNodeCount = 0

End Sub
Attribute VB_Name = "Common"
'  FILE:       Common.bas
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Module containing common functionality throughout
'              StepMaster
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private Const mstrModuleName As String = "Common."

' Used to separate the variable data from the constant error
' message being raised when a context-sensitive error is displayed
Private Const mintDelimiter As String = " : "
Private Const mstrFormatString = "mmddyy"

' Identifiers for the different labels that need to be loaded
' into the tree view for each workspace
Public Const mstrWorkspacePrefix = "W"
Public Const mstrParameterPrefix = "P"
Public Const mstrParamConnectionPrefix = "C"
Public Const mstrConnectionDtlPrefix = "N"
Public Const mstrParamExtensionPrefix = "E"
Public Const mstrParamBuiltInPrefix = "B"
Public Const gstrGlobalStepPrefix = "G"
Public Const gstrManagerStepPrefix = "M"
Public Const gstrWorkerStepPrefix = "S"
Public Const gstrDummyPrefix = "D"
Public Const mstrLabelPrefix = "L"
Public Const mstrInstancePrefix = "I"
Public Function LabelStep(lngWorkspaceIdentifier As Long) As String
    ' Returns the step label for the workspace identifier passed in
    ' Basically this is a wrapper around the MakeKeyValid function

    LabelStep = MakeKeyValid(gintStepLabel, gintStepLabel,
lngWorkspaceIdentifier)

End Function


Public Function JulianDateToString(dt64Bit As Currency) As String

    Dim lYear As Long
    Dim lMonth As Long
    Dim lDay As Long
    Dim lHour As Long
    Dim lMin As Long
    Dim lSec As Long
    Dim lMs As Long

    Call JulianToTime(dt64Bit, lYear, lMonth, lDay, lHour, lMin, lSec, lMs)
    JulianDateToString = Format$(lYear, gsYearFormat) & gsDateSeparator & _
        Format$(lMonth, gsDtFormat) & gsDateSeparator & _
        Format$(lDay, gsDtFormat) & gstrBlank & _
        Format$(lHour, gsTmFormat) & gsTimeSeparator & _
        Format$(lMin, gsTmFormat) & gsTimeSeparator & _
        Format$(lSec, gsTmFormat) & gsMsSeparator & _
        Format$(lMs, gsMSecondFormat)

End Function
Public Sub DeleteFile(strFile As String, Optional ByVal bCheckIfEmpty As
Boolean = False)

    ' Ensure that there is only a single file of the name before delete, since
    ' Kill supports wildcards and can potentially delete a number of files
    Dim strTemp As String

    If CheckFileExists(strFile) Then
        If bCheckIfEmpty Then
            If FileLen(strFile) = 0 Then
                Kill strFile
            End If
        Else
            Kill strFile
        End If
    End If

End Sub
Public Function CheckFileExists(strFile As String) As Boolean

    ' Returns true if the passed in file exists
    ' Raises an error if multiple files are found (filename contains awildcard)
    CheckFileExists = False

    If Not StringEmpty(Dir(strFile)) Then
        If Not StringEmpty(Dir()) Then
            On Error GoTo 0
            Err.Raise vbObjectError + errDeleteSingleFile, _
                mstrModuleName & "DeleteFile", LoadResString(errDeleteSingleFile)
        End If

        CheckFileExists = True
    End If

End Function

Public Function GetVersionString() As String
GetVersionString = "Version " & gsVersion
End Function

Function IsLabel(strKey As String) As Boolean

    ' The tree view control on frmMain can contain two types of
    ' nodes -
    ' 1. Nodes that contain data for the workspace - this could
    ' be data for the different types of steps or parameters
    ' 2. Nodes that display static data - these kind of nodes
```

```
    ' are referred to as label nodes e.g. "Global Steps" is a
    ' label node
    ' This function returns True if the passed in key corresponds
    ' to a label node

    IsLabel = InStr(strKey, mstrLabelPrefix) > 0

End Function

Function MakeKeyValid(lngIdentifier As Long, _
    intTypeOfNode As Integer, _
    Optional ByVal WorkspaceId As Long = 0, _
    Optional ByVal InstanceId As Long = 0) As String

    ' We use a numbering scheme while loading the tree view with
    ' all node data, since it needs a unique key and we want to
    ' use the key to identify the data it contains.
    ' Moreover, add a character to the beginning of the identifier
    ' so that the tree view control accepts it as a valid string,
    ' viz. "456" doesn't work, so change it to "W456"
    ' The general scheme is to concatenate a Label with the Identifier
    ' e.g A Global Step Node will have the Label, G and the Step Id
    ' concatenated to form the unique key
    ' The list of all such node types is given below
    ' 1. "W" + Workspace_Id for Workspace nodes
    ' 2. "P" + Parameter_Id for Parameter nodes
    ' 3. "M" + Step_Id for Manager Step nodes
    ' 4. "S" + Step_Id for Worker Step nodes
    ' 5. "G" + Step_Id for Global Step nodes
    ' 6. Instance_id + "I" + Step_Id for Instance nodes
    ' 7. Workspace_id + "L" + the label identifier = node type for all Label nodes
    ' Since the manager, worker and global steps are stored in the
    ' same table and the step identifiers will always be unique, we
    ' can use the same character as the prefix, but this is a
    ' convenient way to know the type of step being processed.
    ' The workspace id is appended to the label identifier to make
    ' it unique, since multiple workspaces may be open during a session
    ' Strip the prefix characters off while saving the Ids to thedb

    Dim strPrefixChar As String

    On Error GoTo MakeKeyValidErr
    gstrSource = mstrModuleName & "MakeKeyValid"

    Select Case intTypeOfNode
        Case gintWorkspace
            strPrefixChar = mstrWorkspacePrefix
        Case gintGlobalStep
            strPrefixChar = gstrGlobalStepPrefix
        Case gintManagerStep
            strPrefixChar = gstrManagerStepPrefix
        Case gintWorkerStep
            strPrefixChar = gstrWorkerStepPrefix
        Case gintRunManager, gintRunWorker
            If InstanceId = 0 Then
                On Error GoTo 0
                Err.Raise vbObjectError + errMandatoryParameterMissing, _
                    gstrSource, _
                    LoadResString(errMandatoryParameterMissing)
            End If
            ' Concatenate the instance identifier and the step
            ' identifier to form a unique key
            strPrefixChar = Trim$(Str$(InstanceId)) & mstrInstancePrefix
        Case gintParameter
            strPrefixChar = mstrParameterPrefix
        Case gintNodeParamConnection
            strPrefixChar = mstrParamConnectionPrefix
        Case gintConnectionDtl
            strPrefixChar = mstrConnectionDtlPrefix
        Case gintNodeParamExtension
            strPrefixChar = mstrParamExtensionPrefix
        Case gintNodeParamBuiltIn
            strPrefixChar = mstrParamBuiltInPrefix
        Case gintGlobalsLabel, gintParameterLabel, gintParamConnectionLabel, _
             gintConnDtlLabel, _
             gintParamExtensionLabel, gintParamBuiltInLabel,
gintGlobalStepLabel, _
             gintStepLabel
            If WorkspaceId = 0 Then
                ' The Workspace Id has to be specified for a label node
                ' Otherwise it will not be possible to generate unique label
                ' identifiers if multiple workspaces are open
                On Error GoTo 0
                Err.Raise vbObjectError + errWorkspaceIdMandatory, _
                    gstrSource, _
                    LoadResString(errWorkspaceIdMandatory)
            End If
            ' For all labels, the workspace identifier and the
            ' label prefix are concatenated to form the key
            strPrefixChar = Trim$(Str$(WorkspaceId)) & mstrLabelPrefix
        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errInvalidNodeType, _
                gstrSource, _
                LoadResString(errInvalidNodeType)
    End Select

    MakeKeyValid = strPrefixChar & Trim$(Str$(lngIdentifier))

    Exit Function

MakeKeyValidErr:
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errMakeKeyValidFailed, _
        gstrSource, _
        LoadResString(errMakeKeyValidFailed)

End Function

Function MakeIdentifierValid(strKey As String) As Long

    ' Returns the Identifier corresponding to the passed in key
    ' (Reverse of what was done in MakeKeyValid)

    On Error GoTo MakeIdentifierValidErr

    If IsLabel(strKey) Then
        ' If the key corresponds to a label node, the identifier
        ' appears to the right of the label prefix
        MakeIdentifierValid = Val(Mid(strKey, InStr(strKey, mstrLabelPrefix) + 1))
    ElseIf InStr(strKey, mstrInstancePrefix) = 0 Then
        ' For all other nodes, stripping the first character off
        ' returns a valid Id
        MakeIdentifierValid = Val(Mid(strKey, 2))
    Else
        ' Instance node - strip of all characters till the
        ' instance prefix
        MakeIdentifierValid = Val(Mid(strKey, InStr(strKey, mstrInstancePrefix) +
1))
    End If

    Exit Function

MakeIdentifierValidErr:
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errMakeIdentifierValidFailed, _
        mstrModuleName & "MakeIdentifierValid", _
        LoadResString(errMakeIdentifierValidFailed)

End Function
Public Function IsInstanceNode(strNodeKey As String) As Boolean

    ' Returns true if the passed in node key corresponds to a step instance
    IsInstanceNode = InStr(strNodeKey, mstrInstancePrefix) > 0

End Function
Public Function IsBuiltInLabel(strNodeKey As String) As Boolean

    ' Returns true if the passed in node key corresponds to a step instance
    IsBuiltInLabel = (IsLabel(strNodeKey) And _
        (MakeIdentifierValid(strNodeKey) = gintParamBuiltInLabel))

End Function
```

```
Public Sub ShowBusy()
   ' Modifies the mousepointer to indicate that the
   ' application is busy

   On Error Resume Next

   Screen.MousePointer = vbHourglass

End Sub
Public Sub ShowFree()
   ' Modifies the mousepointer to indicate that the
   ' application has finished processing and is ready
   ' to accept user input

   On Error Resume Next

   Screen.MousePointer = vbDefault

End Sub

Public Function InstrR(strMain As String, _
      strSearch As String) As Integer
   ' Finds the last occurrence of the passed in string

   Dim intPos As Integer
   Dim intPrev As Integer

   On Error GoTo InstrRErr

   intPrev = intPos
   intPos = InStr(1, strMain, strSearch)

   Do While intPos > 0
      intPrev = intPos
      intPos = InStr(intPos + 1, strMain, strSearch)
   Loop
   InstrR = intPrev

   Exit Function

InstrRErr:
   Call LogErrors(Errors)
   gstrSource = mstrModuleName & "InstrR"
   On Error GoTo 0
   Err.Raise vbObjectError + errInstrRFailed, _
      gstrSource, _
      LoadResString(errInstrRFailed)

End Function

Public Function GetDefaultDir(lWspId As Long, WspParameters As
cArrParameters) As String

   Dim sDir As String
   sDir = SubstituteParameters( _
         gstrEnvVarSeparator & PARAM_DEFAULT_DIR &
gstrEnvVarSeparator, _
         lWspId, WspParameters:=WspParameters)
   MakePathValid (sDir & gstrFileSeparator & "a.txt")
   GetDefaultDir = GetShortName(sDir)
   If StringEmpty(GetDefaultDir)Then
      GetDefaultDir = App.Path
   End If

End Function

Public Sub AddArrayElement(ByRef arrNodes() As Object, _
      ByVal objToAdd As Object, _
      ByRef lngCount As Long)
   ' Adds the passed in object to the array

   On Error GoTo AddArrayElementErr

   ' Increase the array dimension and add the object to it
   ReDim Preserve arrNodes(lngCount)
   Set arrNodes(lngCount) = objToAdd
   lngCount = lngCount + 1

   Exit Sub

AddArrayElementErr:
   LogErrors Errors
   gstrSource = mstrModuleName & "AddArrayElement"
   On Error GoTo 0
   Err.Raise vbObjectError + errAddArrayElementFailed, _
         gstrSource, _
         LoadResString(errAddArrayElementFailed)

End Sub


Public Function CheckForNullField(rstRecords As Recordset, strFieldName As
String) As String

   ' Returns an empty string if a given field is null
   On Error GoTo CheckForNullFieldErr

   If IsNull(rstRecords.Fields(strFieldName))Then
      CheckForNullField = gstrEmptyString
   Else
      CheckForNullField = rstRecords.Fields(strFieldName)
   End If
   Exit Function

CheckForNullFieldErr:
   Call LogErrors(Errors)
   On Error GoTo 0
   Err.Raise vbObjectError + errCheckForNullFieldFailed, _
         mstrModuleName & "CheckForNullField", _
         LoadResString(errCheckForNullFieldFailed)

End Function

Public Function ErrorOnNullField(rstRecords As Recordset, strFieldName As
String) As Variant

   ' If a given field is null, raises an error
   ' Else, returns the field value in a variant
   ' The calling function must convert the return value to the
   ' appropriate type
   On Error GoTo ErrorOnNullFieldErr
   gstrSource = mstrModuleName & "ErrorOnNullField"

   If IsNull(rstRecords.Fields(strFieldName))Then
      On Error GoTo 0
      Err.Raise vbObjectError + errMandatoryFieldNull, _
         gstrSource, _
         strFieldName & mintDelimiter& LoadResString(errMandatoryFieldNull)
   Else
      ErrorOnNullField = rstRecords.Fields(strFieldName)
   End If
   Exit Function

ErrorOnNullFieldErr:
   Call LogErrors(Errors)
   On Error GoTo 0
   Err.Raise vbObjectError + errUnableToCheckNull, _
      gstrSource, _
      strFieldName & mintDelimiter& LoadResString(errUnableToCheckNull)

End Function
Public Function StringEmpty(strCheckString As String) As Boolean

   StringEmpty = (strCheckString = gstrEmptyString)

End Function
Public Function GetIteratorValue(cStepIterators As cRunColIt, _
      ByVal strItName As String)

   Dim lngIndex As Long
   Dim strValue As String

   On Error GoTo GetIteratorValueErr
   gstrSource = mstrModuleName & "GetIteratorValue"

   ' Find the iterator in the Iterators collection
```

```
    For lngIndex = 0 To cStepIterators.Count - 1
        If cStepIterators(lngIndex).IteratorName = strItName Then
            strValue = cStepIterators(lngIndex).Value
            Exit For
        End If
    Next lngIndex

    If lngIndex > cStepIterators.Count - 1 Then
        ' The iterator has not been defined for the branch
        ' Raise an error
        On Error GoTo 0
        Err.Raise vbObjectError + errParamNameInvalid, _
                gstrSource, _
                LoadResString(errParamNameInvalid)
    End If

    GetIteratorValue = strValue
    Exit Function

GetIteratorValueErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "GetIteratorValue"
    On Error GoTo 0
    Err.Raise vbObjectError + errGetParamValueFailed, _
            gstrSource, _
            LoadResString(errGetParamValueFailed)

End Function
Public Function SubstituteParameters(ByVal strComString As String, _
        ByVal lngWorkspaceId As Long, _
        Optional StepIterators As cRunColIt = Nothing, _
        Optional WspParameters As cArrParameters = Nothing) As String
    ' This function substitutes all parameter names and
    ' environment variables in the passed in string with
    ' their values. It also substitutes the value for the
    ' iterators, if any.
    ' Since the syntax is to enclose parameter names and
    ' environment variables in "%", we check if a given
    ' variable is a parameter - if so, we substitute the
    ' parameter value - else we try to get the value from
    ' the environment

    Dim intPos As Integer
    Dim intEndPos As Integer
    Dim strEnvVariable As String
    Dim strValue As String
    Dim strCommand As String
    Dim cTempStr As cStringSM

    ' Initialize the return value of the function to the
    ' passed in command
    strCommand = strComString

    If WspParameters Is Nothing Then Set WspParameters = gcParameters

    Set cTempStr = New cStringSM

    intPos = InStr(strCommand, gstrEnvVarSeparator)
    Do While intPos <> 0
        If Mid(strCommand, intPos + 1, 1) = gstrEnvVarSeparator Then
            ' Wildcard character - to be substituted by a single % - later!
            intPos = intPos + 2
            If intPos > Len(strCommand) Then Exit Do
        Else
            ' Extract the environment variable from the passed
            ' in string
            intEndPos = InStr(intPos + 1, strCommand, gstrEnvVarSeparator)

            If intEndPos > 0 Then
                strEnvVariable = Mid(strCommand, intPos + 1, intEndPos - intPos - 1)
            Else
                On Error GoTo 0
                Err.Raise vbObjectError + errParamSeparatorMissing, _
                        gstrSource, _
                        LoadResString(errParamSeparatorMissing)
            End If
            strValue = gstrEmptyString
            ' Get the value of the variable and call a function
            ' to replace the variable with it's value
            strValue = GetValue(strEnvVariable, lngWorkspaceId, StepIterators,
WspParameters)
            ' The function raises an error if the variable is
            ' not found
            strCommand = cTempStr.ReplaceSubString(strCommand,_
                    gstrEnvVarSeparator & strEnvVariable & gstrEnvVarSeparator, _
                    strValue)
        End If

        intPos = InStr(intPos, strCommand, gstrEnvVarSeparator)
    Loop

    strCommand = cTempStr.ReplaceSubString(strCommand,_
            gstrEnvVarSeparator & gstrEnvVarSeparator, gstrEnvVarSeparator)

    Set cTempStr = Nothing
    SubstituteParameters = strCommand

End Function
Private Function GetValue(ByVal strParameter As String, _
        ByVal lngWorkspaceId As Long, _
        cStepIterators As cRunColIt, _
        WspParameters As cArrParameters) As String
    ' This function returns the value for the passed in
    ' parameter - it may be a workspace parameter, an
    ' environment variable or an iterator

    Dim intPos As Integer
    Dim intEndPos As Integer
    Dim strVariable As String
    Dim strValue As String
    Dim cParamRec As cParameter

    On Error GoTo GetValueErr

    ' Initialize the return value of the function to the
    ' empty
    strValue = gstrEmptyString

    intPos = InStr(strParameter, gstrEnvVarSeparator)
    If intPos > 0 Then
        ' Extract the variable from the passed in string
        intEndPos = InStr(intPos + 1, strParameter, gstrEnvVarSeparator)
        If intEndPos = 0 Then
            intEndPos = Len(strParameter)
        End If

        strVariable = Mid(strParameter, intPos + 1, intEndPos - intPos - 1)
    Else
        ' The separator charactor has not been passed in -
        ' try to find the value of the passed in parameter
        strVariable = strParameter
    End If

    If Not StringEmpty(strVariable) Then
        ' Check if this is the timestamp parameter first
        If strVariable = gstrTimeStamp Then
            strValue = Format$(Now, mstrFormatString, _
                    vbUseSystemDayOfWeek, vbUseSystem)
        Else
            ' Try to find a parameter for the workspace with
            ' the same name
            Set cParamRec = WspParameters.GetParameterValue(lngWorkspaceId, _
                    strVariable)
            If cParamRec Is Nothing Then
                If Not cStepIterators Is Nothing Then
                    ' If the string is not a parameter, then check
                    ' if it is an iterator
                    strValue = GetIteratorValue(cStepIterators, strVariable)
                End If

                If StringEmpty(strValue) Then
                    ' Neither - Check if it is an environment variable
                    strValue = Environ$(strVariable)
                    If StringEmpty(strValue) Then
```

```
                On Error  GoTo 0
                 WriteError errSubValuesFailed, _
                      OptArgs:="Invalid parameter: " & gstrSQ & strVariable &
gstrSQ
                 Err.Raise vbObjectError + errSubValuesFailed, _
                      mstrModuleName & "GetValue", _
                      LoadResString(errSubValuesFailed)& "Invalid parameter: "
& gstrSQ & strVariable & gstrSQ
               End If
            End If
         Else
            strValue = cParamRec.ParameterValue
         End If
      End If
   End If

   GetValue = strValue

   Exit Function

GetValueErr:
   If Err.Number = vbObjectError + errParamNameInvalid Then
      ' If the parameter has not been defined for the
      ' workspace then check if it is an environment
      ' variable
      Resume Next
   End If

   ' Log the error code raised by Visual Basic
   Call LogErrors(Errors)
   gstrSource = mstrModuleName & "GetValue"
   WriteError errSubValuesFailed, gstrSource, "Parameter: " & gstrSQ &
strVariable & gstrSQ
   On Error GoTo 0
   Err.Raise vbObjectError + errSubValuesFailed, _
         gstrSource, _
         LoadResString(errSubValuesFailed)& "Parameter: " & gstrSQ &
strVariable & gstrSQ

End Function
Public Function SQLFixup(strField As String) As String
   ' Returns a string that can be executed by SQL Server

   Dim cMyStr As New cStringSM
   Dim strTemp As String

   On Error GoTo SQLFixupErr

   strTemp = strField
   SQLFixup = strTemp

   ' Single-quotes have to be replaced by two single-quotes,
   ' since a single-quote is the identifier delimiter
   ' character - call a procedure to do the replace
'    SQLFixup = cMyStr.ReplaceSubString(strTemp, gstrDQ, "\" & gstrDQ)

   ' Replace pipe characters with the corresponding chr function
   ' SQLFixup = cMyStr.ReplaceSubString(strTemp, gstrDQ, gstrDQ & gstrDQ)

   Exit Function

SQLFixupErr:
   gstrSource = mstrModuleName & "SQLFixup"
   LogErrors Errors
   On Error GoTo 0
   Err.Raise vbObjectError + errMakeFieldValidFailed, _
         gstrSource, LoadResString(errMakeFieldValidFailed)

End Function
Public Function TranslateStepLabel(sLabel As String) As String
   ' Translates the passed in step label to a valid file name
   ' All characters in the label that are invalid for filenames (viz. \ / : * ? " < > |)
   ' and spaces are substituted with underscores - also ensure that the resulting
filename
   ' is not greater than 255 characters
   Dim cTempStr As New cStringSM
   TranslateStepLabel = cTempStr.ReplaceSubString(sLabel, gstrFileSeparator,
"_")
   TranslateStepLabel = cTempStr.ReplaceSubString(TranslateStepLabel,"/",
gstrUnderscore)
   TranslateStepLabel = cTempStr.ReplaceSubString(TranslateStepLabel,":",
gstrUnderscore)
   TranslateStepLabel = cTempStr.ReplaceSubString(TranslateStepLabel,"*",
gstrUnderscore)
   TranslateStepLabel = cTempStr.ReplaceSubString(TranslateStepLabel,"?",
gstrUnderscore)
   TranslateStepLabel = cTempStr.ReplaceSubString(TranslateStepLabel,
gstrDQ, gstrUnderscore)
   TranslateStepLabel = cTempStr.ReplaceSubString(TranslateStepLabel,"<",
gstrUnderscore)
   TranslateStepLabel = cTempStr.ReplaceSubString(TranslateStepLabel,">",
gstrUnderscore)
   TranslateStepLabel = cTempStr.ReplaceSubString(TranslateStepLabel,"|",
gstrUnderscore)
   TranslateStepLabel = cTempStr.ReplaceSubString(TranslateStepLabel,
gstrBlank, gstrUnderscore)

   If Len(TranslateStepLabel) > MAX_PATH Then
      TranslateStepLabel = Mid(TranslateStepLabel, 1, MAX_PATH)
   End If

End Function

Public Function TypeOfObject(ByVal objNode As Object) As Integer
   ' Determines the type of object that is passed in

   On Error GoTo TypeOfObjectErr
   gstrSource = mstrModuleName & "TypeOfObject"

   Select Case TypeName(objNode)
      Case "cWorkspace"
         TypeOfObject = gintWorkspace

      Case "cParameter"
         TypeOfObject = gintParameter

      Case "cConnection"
         TypeOfObject = gintParameterConnect

      Case "cConnDtl"
         TypeOfObject = gintConnectionDtl

      Case "cGlobalStep"
         TypeOfObject = gintGlobalStep

      Case "cManager"
         TypeOfObject = gintManagerStep

      Case "cWorker"
         TypeOfObject = gintWorkerStep

      Case "cStep"
         ' If a step record is passed in, call a function
         ' to determine the type of step
         TypeOfObject = TypeOfStep(StepClass:=objNode)

      Case Else
         WriteError errTypeOfObjectFailed, gstrSource, _
               TypeName(objNode)
         On Error GoTo 0
         Err.Raise vbObjectError + errTypeOfObjectFailed, _
               gstrSource, _
               LoadResString(errTypeOfObjectFailed)
   End Select

   Exit Function

TypeOfObjectErr:
   ' Log the error code raised by Visual Basic
   Call LogErrors(Errors)
   On Error GoTo 0
   Err.Raise vbObjectError + errTypeOfObjectFailed, _
         gstrSource, _
         LoadResString(errTypeOfObjectFailed)

End Function
```

```
Attribute VB_Name = "ConnDtlCommon"
'  FILE:      ConnDtlCommon.bas
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:   Contains functionality common across StepMaster and
'             SMRunOnly, pertaining to connections
'             Specifically, functions to load connections in an array
'             and so on.
'  Contact:   Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "ConnDtlCommon."

Public Sub LoadRSInConnDtlArray(rstConns As Recordset, cConns As
cConnDtls)

    Dim cNewConnDtl As cConnDtl

    On Error GoTo LoadRSInConnDtlArrayErr

    If rstConns.RecordCount = 0 Then
        Exit Sub
    End If

    rstConns.MoveFirst
    While Not rstConns.EOF

        Set cNewConnDtl = New cConnDtl

        ' Initialize ConnDtl values
        ' Call a procedure to raise an error if mandatory fields are null.
        cNewConnDtl.ConnNameId = ErrorOnNullField(rstConns,
FLD_ID_CONN_NAME)
        cNewConnDtl.WorkspaceId = ErrorOnNullField(rstConns,
FLD_ID_WORKSPACE)
        cNewConnDtl.ConnName = CStr(ErrorOnNullField(rstConns,
FLD_CONN_DTL_CONNECTION_NAME))
        cNewConnDtl.ConnectionString = CheckForNullField(rstConns,
FLD_CONN_DTL_CONNECTION_STRING)
        cNewConnDtl.ConnType = CheckForNullField(rstConns,
FLD_CONN_DTL_CONNECTION_TYPE)

        cConns.Load cNewConnDtl

        Set cNewConnDtl = Nothing
        rstConns.MoveNext
    Wend

    Exit Sub

LoadRSInConnDtlArrayErr:
    LogErrors Errors
    gstrSource = mstrModuleName & "LoadRSInConnDtlArray"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadRsInArrayFailed, gstrSource, _
          LoadResString(errLoadRsInArrayFailed)
End Sub
Attribute VB_Name = "ConnectionCommon"
'  FILE:      ConnnectionCommon.bas
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:   Contains functionality common across StepMaster and
'             SMRunOnly, pertaining to connection strings
'             Specifically, functions to load connections strings
'             in an array and so on.
'  Contact:   Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "ConnectionCommon."

Public Sub LoadRecordsetInConnectionArray(rstConns As Recordset, cConns As
cConnections)

    Dim cNewConnection As cConnection

    On Error GoTo LoadRecordsetInConnectionArrayErr

    If rstConns.RecordCount = 0 Then
        Exit Sub
    End If

    rstConns.MoveFirst
    While Not rstConns.EOF

        Set cNewConnection = New cConnection

        ' Initialize Connection values
        ' Call a procedure to raise an error if mandatory fields are null.
        cNewConnection.ConnectionId = ErrorOnNullField(rstConns,
"connection_id")
        cNewConnection.WorkspaceId = CStr(ErrorOnNullField(rstConns,
FLD_ID_WORKSPACE))
        cNewConnection.ConnectionName = CStr(ErrorOnNullField(rstConns,
"connection_name"))
        cNewConnection.ConnectionValue = CheckForNullField(rstConns,
"connection_value")
        cNewConnection.Description = CheckForNullField(rstConns, "description")

        cNewConnection.NoCountDisplay = CheckForNullField(rstConns,
"no_count_display")
        cNewConnection.NoExecute = CheckForNullField(rstConns, "no_execute")
        cNewConnection.ParseQueryOnly = CheckForNullField(rstConns,
"parse_query_only")
        cNewConnection.QuotedIdentifiers = CheckForNullField(rstConns,
"ANSI_quoted_identifiers")
        cNewConnection.AnsiNulls = CheckForNullField(rstConns, "ANSI_nulls")
        cNewConnection.ShowQueryPlan = CheckForNullField(rstConns,
"show_query_plan")
        cNewConnection.ShowStatsTime = CheckForNullField(rstConns,
"show_stats_time")
        cNewConnection.ShowStatsIO = CheckForNullField(rstConns,
"show_stats_io")
        cNewConnection.ParseOdbcMsg = CheckForNullField(rstConns,
"parse_odbc_msg_prefixes")
        cNewConnection.RowCount = CheckForNullField(rstConns, "row_count")
        cNewConnection.TsqlBatchSeparator = CheckForNullField(rstConns,
"tsql_batch_separator")
        cNewConnection.QueryTimeOut = CheckForNullField(rstConns,
"query_time_out")
        cNewConnection.ServerLanguage = CheckForNullField(rstConns,
"server_language")
        cNewConnection.CharacterTranslation = CheckForNullField(rstConns,
"character_translation")
        cNewConnection.RegionalSettings = CheckForNullField(rstConns,
"regional_settings")

        cConns.Load cNewConnection

        Set cNewConnection = Nothing
        rstConns.MoveNext
    Wend

    Exit Sub

LoadRecordsetInConnectionArrayErr:
    LogErrors Errors
    gstrSource = mstrModuleName & "LoadRecordsetInConnectionArray"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadRsInArrayFailed, gstrSource, _
          LoadResString(errLoadRsInArrayFailed)
End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
```

```
END
Attribute VB_Name = "cParameter"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cParameter.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Encapsulates the properties and methods of a parameter.
'              Contains functions to insert, update and delete
'              workspace_parameters records from the database.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit
Option Base 0

' Local variable(s) to hold property value(s)
Private mlngWorkspaceId As Long
Private mlngParameterId As Long
Private mstrParameterName As String
Private mstrParameterValue As String
Private mstrDescription As String
Private mintParameterType As Integer
Private mdbsStepMaster As Database
Private mintOperation As Operation
Private mlngPosition As Long

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cParameter."

' The cSequence class is used to generate unique parameter identifiers
Private mParameterSeq As cSequence

' The StringSM class is used to carry out string operations
Private mFieldValue As cStringSM

' Parameter types
Public Enum ParameterType
    gintParameterGeneric = 0
    gintParameterConnect
    gintParameterApplication
    gintParameterBuiltIn
End Enum


Private Sub AssignParameters(qyExec As DAO.QueryDef)
    ' Assigns values to the parameters in the querydef object
    ' The parameter names are cryptic to make them different
    ' from the field names. When the parameter names are
    ' the same as the field names, parameters in the where
    ' clause do not get created.

    Dim prmParam As DAO.Parameter

    On Error GoTo AssignParametersErr

    For Each prmParam In qyExec.Parameters
        Select Case prmParam.Name
            Case "[w_id]"
                prmParam.Value = mlngWorkspaceId

            Case "[p_id]"
                prmParam.Value = mlngParameterId

            Case "[p_name]"
                prmParam.Value = mstrParameterName

            Case "[p_value]"
                prmParam.Value = mstrParameterValue

            Case "[desc]"
                prmParam.Value = mstrDescription

            Case "[p_type]"
                prmParam.Value = mintParameterType

            Case Else
                ' Write the parameter name that is faulty
                WriteError errInvalidParameter, mstrSource, _
                    prmParam.Name
                On Error GoTo 0
                Err.Raise errInvalidParameter, mstrModuleName &
"AssignParameters", _
                    LoadResString(errInvalidParameter)
        End Select
    Next prmParam

'    qyExec.Parameters("w_id").Value = mlngWorkspaceId
'    qyExec.Parameters("p_id").Value = mlngParameterId
'    qyExec.Parameters("p_name").Value = mstrParameterName
'    qyExec.Parameters("p_value").Value = mstrParameterValue
'

    Exit Sub

AssignParametersErr:

    mstrSource = mstrModuleName & "AssignParameters"
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errAssignParametersFailed, _
        mstrSource, LoadResString(errAssignParametersFailed)

End Sub

Public Property Let Position(ByVal RHS As Long)

    mlngPosition = RHS

End Property

Public Property Get Position() As Long

    Position = mlngPosition

End Property

Public Function Clone() As cParameter

    ' Creates a copy of a given parameter

    Dim cCloneParam As cParameter

    On Error GoTo CloneErr
    mstrSource = mstrModuleName & "Clone"

    Set cCloneParam = New cParameter

    ' Copy all the parameter properties to the newly
    ' created parameter
    Set cCloneParam.NodeDB = mdbsStepMaster
    cCloneParam.WorkspaceId = mlngWorkspaceId
    cCloneParam.ParameterId = mlngParameterId
    cCloneParam.ParameterName = mstrParameterName
    cCloneParam.ParameterValue = mstrParameterValue
    cCloneParam.Description = mstrDescription
    cCloneParam.ParameterType = mintParameterType
    cCloneParam.IndOperation = mintOperation
    cCloneParam.Position = mlngPosition

    ' And set the return value to the newly created parameter
    Set Clone = cCloneParam
    Set cCloneParam = Nothing

    Exit Function

CloneErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Clone"
    On Error GoTo 0
```

```
    Err.Raise vbObjectError + errCloneFailed, _
        mstrSource, LoadResString(errCloneFailed)

End Function
Public Property Set NodeDB(vdata As Database)

    Set mdbsStepMaster = vdata

End Property
Public Property Get NodeDB() As Database

    Set NodeDB = mdbsStepMaster

End Property

Private Sub CheckDupParameterName()
    ' Check if the parameter name already exists in the workspace

    Dim rstParameter As Recordset
    Dim strSql As String
    Dim qy As DAO.QueryDef

    On Error GoTo CheckDupParameterNameErr
    mstrSource = mstrModuleName & "CheckDupParameterName"

    ' Create a recordset object to retrieve the count of all parameters
    ' for the workspace with the same name
    strSql = "Select count(*) as parameter_count " & _
        " from workspace_parameters " & _
        " where workspace_id = [w_id]" & _
        " and parameter_name = [p_name]" & _
        " and parameter_id <> [p_id]"

    Set qy = mdbsStepMaster.CreateQueryDef(gstrEmptyString, strSql)
    Call AssignParameters(qy)

    Set rstParameter = qy.OpenRecordset(dbOpenForwardOnly)

    If rstParameter![parameter_count] > 0 Then
        rstParameter.Close
        qy.Close
        ShowError errDuplicateParameterName
        On Error GoTo 0
        Err.Raise vbObjectError + errDuplicateParameterName, _
            mstrSource, LoadResString(errDuplicateParameterName)
    End If

    rstParameter.Close
    qy.Close

    Exit Sub

CheckDupParameterNameErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "CheckDupParameterName"
    On Error GoTo 0
    Err.Raise vbObjectError + errCheckDupParameterNameFailed, _
        mstrSource, LoadResString(errCheckDupParameterNameFailed)

End Sub
Private Sub CheckDB()
    ' Check if the database object has been initialized

    If mdbsStepMaster Is Nothing Then
        On Error GoTo 0
        Err.Raise vbObjectError + errInvalidDB, _
            mstrModuleName & "CheckDB", LoadResString(errInvalidDB)
    End If

End Sub
Public Property Let ParameterValue(vdata As String)

    mstrParameterValue = vdata

End Property
Public Property Let Description(vdata As String)

    mstrDescription = vdata

End Property
Public Property Let ParameterType(vdata As ParameterType)

    mintParameterType = vdata

End Property

Public Property Let ParameterName(vdata As String)

    If vdata = gstrEmptyString Then

        ShowError errParameterNameMandatory
        On Error GoTo 0
        ' Propogate this error back to the caller
        Err.Raise vbObjectError + errParameterNameMandatory, _
            mstrSource, LoadResString(errParameterNameMandatory)
    Else
        mstrParameterName = vdata
    End If

End Property

Public Property Let ParameterId(vdata As Long)
    mlngParameterId = vdata
End Property

Public Property Let IndOperation(ByVal vdata As Operation)

    ' The valid operations are define in the cOperations
    ' class. Check if the operation is valid
    Select Case vdata
        Case QueryOp, InsertOp, UpdateOp, DeleteOp
            mintOperation = vdata

        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errInvalidOperation, _
                mstrSource, LoadResString(errInvalidOperation)
    End Select

End Property
Public Sub Validate()
    ' Each distinct object will have a Validate method which
    ' will check if the class properties are valid. This method
    ' will be used to check interdependant properties that
    ' cannot be validated by the let procedures.
    ' It should be called by the add and modify methods of the class

    On Error GoTo ValidateErr

    ' Check if the db object is valid
    Call CheckDB

    ' Call procedure to raise an error if the parameter name
    ' already exists in the workspace -
    ' if there are duplicates, we don't know what value for the
    ' parameter to use at runtime
    Call CheckDupParameterName

    Exit Sub

ValidateErr:

    mstrSource = mstrModuleName & "Validate"
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errValidateFailed, _
        mstrSource, LoadResString(errValidateFailed)

End Sub
Public Property Let WorkspaceId(vdata As Long)

    mlngWorkspaceId = vdata

End Property

Public Sub Add()
```

```
    Dim strInsert As String
    Dim qy As DAO.QueryDef

    On Error GoTo AddErr

    ' Validate the record before trying to insert the record
    Call Validate

    ' Create a temporary querydef object
    strInsert = "insert into workspace_parameters " & _
          "( workspace_id, parameter_id, " & _
          " parameter_name, parameter_value, " & _
          " description, parameter_type ) " & _
          " values ( [w_id], [p_id], [p_name], [p_value], [desc], [p_type] ) "
    Set qy = mdbsStepMaster.CreateQueryDef(gstrEmptyString,strInsert)

    ' Call a procedure to assign the parameter values
    Call AssignParameters(qy)

    qy.Execute dbFailOnError
    qy.Close

'    strInsert = "insert into workspace_parameters " & _
'          "( workspace_id, parameter_id, " & _
'          " parameter_name, parameter_value ) " & _
'          " values ( " & _
'          Str(mlngWorkspaceId) & ", " & Str(mlngParameterId) & _
'          ", " & mFieldValue.MakeStringFieldValid(mstrParameterName) & _
'          ", " & mFieldValue.MakeStringFieldValid(mstrParameterValue) & " ) "
'    mdbsStepMaster.Execute strInsert, dbFailOnError + dbSQLPassThrough

    Exit Sub

AddErr:

    mstrSource = mstrModuleName & "Add"
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errParameterInsertFailed, _
       mstrSource, LoadResString(errParameterInsertFailed)

End Sub
Public Sub Delete()

    Dim strDelete As String
    Dim qy As DAO.QueryDef

    On Error GoTo DeleteErr

    ' Check if the db object is valid
    Call CheckDB

    strDelete = "delete from workspace_parameters " & _
          " where parameter_id = [p_id]"
    Set qy = mdbsStepMaster.CreateQueryDef(gstrEmptyString,strDelete)

    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    qy.Close

    Exit Sub

DeleteErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Delete"
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteParameterFailed, _
          mstrSource, _
          LoadResString(errDeleteParameterFailed)

End Sub

Public Sub Modify()

    Dim strUpdate As String
    Dim qy As QueryDef

    On Error GoTo ModifyErr

    ' Validate the updated values before trying to modify thedb
    Call Validate

    ' Create a temporary querydef object with the modify string
    strUpdate = "update workspace_parameters " & _
          " set workspace_id = [w_id], " & _
          "parameter_name = [p_name], " & _
          "parameter_value = [p_value], " & _
          "description = [desc], " & _
          "parameter_type = [p_type] " & _
          " where parameter_id = [p_id]"
    Set qy = mdbsStepMaster.CreateQueryDef(gstrEmptyString,strUpdate)

    ' Call a procedure to assign the parameter values to the
    ' querydef object
    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    qy.Close

'    mdbsStepMaster.Execute strUpdate, dbFailOnError
'
    Exit Sub

ModifyErr:

    mstrSource = mstrModuleName & "Modify"
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errParameterUpdateFailed, _
       mstrSource, LoadResString(errParameterUpdateFailed)

End Sub
Public Property Get ParameterName() As String

    ParameterName = mstrParameterName

End Property

Public Property Get ParameterId() As Long

    ParameterId = mlngParameterId

End Property
Public Property Get NextIdentifier() As Long

    Dim lngNextId As Long

    On Error GoTo NextIdentifierErr

    ' First check if the database object is valid
    Call CheckDB

    ' Retrieve the next identifier using the sequence class
    Set mParameterSeq = New cSequence
    Set mParameterSeq.IdDatabase = mdbsStepMaster
    mParameterSeq.IdentifierColumn= "Parameter_id"
    lngNextId = mParameterSeq.Identifier
    Set mParameterSeq = Nothing

    NextIdentifier = lngNextId
    Exit Property

NextIdentifierErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "NextIdentifier"
    On Error GoTo 0
    Err.Raise vbObjectError + errIdGetFailed, _
       mstrSource, LoadResString(errIdGetFailed)

End Property
Public Property Get IndOperation() As Operation

    IndOperation = mintOperation
```

```
End Property

Public Property Get WorkspaceId() As Long

    WorkspaceId = mlngWorkspaceId

End Property

Public Property Get ParameterValue() As String

    ParameterValue = mstrParameterValue

End Property
Public Property Get Description() As String

    Description = mstrDescription

End Property
Public Property Get ParameterType() As ParameterType

    ParameterType = mintParameterType

End Property


Private Sub Class_Initialize()

    Set mFieldValue = New cStringSM

    ' Initialize the operation indicator variable to Query
    ' It will be modified later by the collection class when
    ' inserts, updates or deletes are performed
    mintOperation = QueryOp

End Sub

Private Sub Class_Terminate()

    Set mdbsStepMaster = Nothing
    Set mFieldValue = Nothing

End Sub

VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cRunColIt"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cRunColIt.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module implements a stack of Iterator nodes.
'              Ensures that only cRunItNode objects are stored in the stack.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "cRunColIt."
Private mstrSource As String

Private mcIterators As cStack
Public Sub Clear()

    mcIterators.Clear

End Sub

Private Sub Class_Initialize()

    Set mcIterators = New cStack

End Sub

Private Sub Class_Terminate()

    Set mcIterators = Nothing

End Sub


Public Function Value(strItName As String) As String

    Dim lngIndex As Long

    For lngIndex = 0 To mcIterators.Count - 1
        If mcIterators(lngIndex).IteratorName = strItName Then
            Value = mcIterators(lngIndex).Value
            Exit For
        End If
    Next lngIndex

End Function

Public Property Get Item(ByVal Position As Long) As cRunItNode
Attribute Item.VB_UserMemId = 0

    Set Item = mcIterators(Position)

End Property


Public Function Count() As Long

    Count = mcIterators.Count

End Function

Public Function Pop() As cRunItNode

    Set Pop = mcIterators.Pop

End Function

Public Sub Push(objToPush As cRunItNode)

    Call mcIterators.Push(objToPush)

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cRunItDetails"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cRunItDetails.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module encapsulates the properties of iterator values
'              that are used by the step being executed at runtime.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'

Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "cRunItDetails."
Private mstrSource As String

Private mstrIteratorName As String
Private mintType As ValueType
```

```
Private mlngSequence As Long
Private mlngFrom As Long
Private mlngTo As Long
Private mlngStep As Long
Private mstrValue As String

Public Property Get RangeTo() As Long

    RangeTo = mlngTo

End Property
Public Property Let RangeTo(ByVal vdata As Long)

    mlngTo = vdata

End Property

Public Property Get RangeFrom() As Long

    RangeFrom = mlngFrom

End Property
Public Property Get Sequence() As Long

    Sequence = mlngSequence

End Property

Public Property Get RangeStep() As Long

    RangeStep = mlngStep

End Property
Public Property Let RangeStep(vdata As Long)

    mlngStep = vdata

End Property

Public Property Let RangeFrom(ByVal vdata As Long)

    mlngFrom = vdata

End Property
Public Property Let Sequence(ByVal vdata As Long)

    mlngSequence = vdata

End Property

Public Property Get IteratorType() As ValueType

    IteratorType = mintType

End Property
Public Property Let IteratorType(ByVal vdata As ValueType)

    On Error GoTo TypeErr
    mstrSource = mstrModuleName & "Type"

    ' These constants have been defined in the enumeration,
    ' Type, which is exposed
    Select Case vdata
        Case gintFrom, gintTo, gintStep, gintValue
            mintType = vdata

        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errTypeInvalid, _
                mstrSource, LoadResString(errTypeInvalid)
    End Select

    Exit Property

TypeErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Type"
    On Error GoTo 0
    Err.Raise vbObjectError + errTypeInvalid, _
        mstrSource, LoadResString(errTypeInvalid)

End Property
Private Sub IsList()

    If mintType <> gintValue Then
        On Error GoTo 0
        Err.Raise vbObjectError + errInvalidProperty, mstrSource, _
            LoadResString(errInvalidProperty)
    End If

End Sub
Private Sub IsRange()

    If mintType = gintValue Then
        On Error GoTo 0
        Err.Raise vbObjectError + errInvalidProperty, mstrSource, _
            LoadResString(errInvalidProperty)
    End If

End Sub

Public Property Get Value() As String

    Value = mstrValue

End Property
Public Property Let Value(vdata As String)

    mstrValue = vdata

End Property


Public Property Get IteratorName() As String

    IteratorName = mstrIteratorName

End Property
Public Property Let IteratorName(ByVal vdata As String)

    mstrIteratorName = vdata

End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cRunStep"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cRunStep.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This class executes the step that is assigned to the
'              ExecuteStep property. It executes the pre-execution constraints
'              in sequence and then the step itself. At the end it executes
'              the post-execution constraints. Since these steps should always
'              be executed in sequence, each step is only fired on the
'              completion of the previous step.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "cRunStep."
Private mstrSource As String

' Local variable(s) to hold property value(s)
Private mcStep As cStep
Private mcGlobals As cArrSteps
```

```
Private mcvntWspPreCons As Variant
Private mcvntWspPostCons As Variant
Private mcvntPreCons As Variant
Private mcvntPostCons As Variant
Private mcIterators As cRunColIt
Private mlngInstanceId As Long ' Identifier for the current instance
Private mlngIndex As Long ' Index value for the current instance
Private mstrCommand As String ' The command string
Private msRunStepDtl As String ' Step text/file name that will go into the
run_step_details table
Private mblnAbort As Boolean   ' Set to True when the user aborts the run
Private msOutputFile As String
Private msErrorFile As String
Private miStatus As InstanceStatus
Private mcVBErr As cVBErrorsSM
Public WspParameters As cArrParameters
Public WspConnections As cConnections
Public WspConnDtls As cConnDtls

Private WithEvents mcTermProcess As cTermProcess
Attribute mcTermProcess.VB_VarHelpID = -1
Public RunId As Long
Public CreateInputFiles As Boolean
Private msOutputDir As String

' Object that will execute the step
Private WithEvents mcExecObj As EXECUTEDLLLib.Execute
Attribute mcExecObj.VB_VarHelpID = -1

' Holds the step that is currently being executed (constraint or
' worker step)
Private mcExecStep As cStep

Private Const msCompareExe As String = "\diff.exe"

Private Enum NextNodeType
   mintWspPreConstraint = 1
   mintPreConstraint
   mintStep
   mintWspPostConstraint
   mintPostConstraint
End Enum

' Public events to notify the calling function of the
' start and end time for each step
Public Event StepStart(cStepRecord As cStep, _
      dtmStartTime As Currency, InstanceId As Long)
Public Event StepComplete(cStepRecord As cStep, _
      dtmEndTime As Currency, InstanceId As Long, Status As InstanceStatus)
Public Event ProcessStart(cStepRecord As cStep, _
      strCommand As String, dtmStartTime As Currency, _
      InstanceId As Long)
Public Event ProcessComplete(cStepRecord As cStep, _
      dtmStartTime As Currency, InstanceId As Long, lElapsed As Long)

Private Function AppendDiffErrors(sDiffFile As String)
   ' The file containing the errors generated by the diff utility is passed in
   ' These errors are appended to the error file for the step

   Dim sTemp As String
   Dim InputFile As Integer

   If Not StringEmpty(sDiffFile)Then

      InputFile = FreeFile
      Open sDiffFile For Input Access Read As InputFile

      Do While Not EOF(InputFile)        ' Loop until end of file.
         Line Input #InputFile, sTemp    ' Read line into variable.
         mcVBErr.LogMessage sTemp
      Loop

      Close InputFile
   End If

End Function

Private Sub CreateStepTextFile()
   ' Creates a file containing the step text being executed
   On Error GoTo CreateStepTextFileErr

   Dim sInputFile As String

   If mcExecStep.ExecutionMechanism = gintExecuteShell Then
      sInputFile = GetOutputFile(gsCmdFileSuffix)
   Else
      sInputFile = GetOutputFile(gsSqlFileSuffix)
   End If

   ' Generate a file containing the step text being executed
   If Not StringEmpty(mcExecStep.StepTextFile)Or
mcExecStep.ExecutionMechanism = gintExecuteShell Then
      FileCopy mstrCommand, sInputFile
   Else
      Call WriteCommandToFile(mstrCommand,sInputFile)
   End If

   Exit Sub

CreateStepTextFileErr:

   mcVBErr.LogVBErrors

End Sub
Private Function GetOutputFile(strFileExt As String) As String
   ' This function generates the output file name for the step currently being
executed
   ' The value of the built-in parameter'DefaultDir' is appended with the run
identifier
   ' for the file location
   ' The step label is used for the file name and a combination of alliterator values
   ' for the step is used to make the output files unique for each instance
   Dim sFile As String
   Dim sIt As String
   Dim lIt As Long

   On Error GoTo GetOutputFileErr

   sFile = SubstituteParametersIfPossible(mcExecStep.StepLabel)

   sFile = TranslateStepLabel(sFile)

   If mcExecStep Is mcStep Then
      ' Use iterators that have been defined for the worker or any of it's managers
      ' to make the error/log file unique for this instance
      For lIt = mcIterators.Count - 1 To 0 Step -1
         sIt = sIt & gsExtSeparator & mcIterators(lIt).Value
      Next lIt
   End If
   sIt = sIt & strFileExt

   ' Ensure that the length of the complete path does not exceed 255 characters
   If Len(msOutputDir) + Len(sFile) + Len(sIt) > MAX_PATH Then
      sFile = Mid(sFile, 1, MAX_PATH - Len(sIt) - Len(msOutputDir))
   End If
   GetOutputFile = msOutputDir & sFile & sIt
   Exit Function

GetOutputFileErr:

   ' Does not make sense to log error to the error file yet. Write to the project
   ' log and return the step label as default
   GetOutputFile = mcExecStep.StepLabel & gsExtSeparator & strFileExt

End Function


Private Sub HandleExecutionError()

   On Error GoTo HandleExecutionError

   ' Log the error code raised by Visual Basic
   miStatus = gintFailed
   mcVBErr.LogVBErrors
   Call mcVBErr.WriteError(errExecuteStepFailed,_
```

```
        OptArgs:="Continuation criteria for the step is: " &
gsContCriteria(mcStep.ContinuationCriteria))

HandleExecutionError:

   ' Logging failed - return

End Sub

Public Property Get Index() As Long

   Index = mlngIndex

End Property
Public Property Let Index(ByVal vdata As Long)

   mlngIndex = vdata

End Property
Private Function InitializeExecStatus() As InstanceStatus
   Dim sCompareFile As String

   On Error GoTo InitializeExecStatusErr

   InitializeExecStatus = mcExecObj.StepStatus

   If InitializeExecStatus = gintComplete Then
      If Not StringEmpty(mcExecStep.FailureDetails)Then
         ' Compare output to determine whether the step failed
         sCompareFile = GetShortName(SubstituteParameters( _
               mcExecStep.FailureDetails, mcExecStep.WorkspaceId, mcIterators,
_
               WspParameters))
         InitializeExecStatus = IIf(CompareOutput(sCompareFile, msOutputFile),
gintComplete, gintFailed)
      End If
   End If

   Exit Function

InitializeExecStatusErr:
   mcVBErr.LogVBErrors
'   Call LogErrors(Errors)
   InitializeExecStatus = mcExecObj.StepStatus

End Function
Private Function CompareOutput(sCompareFile As String, sOutputFile As String)
As Boolean

   Dim sCmpOutput As String
   Dim sDiffOutput As String

   On Error GoTo CompareOutputErr

   ' Create temporary files to store the file compare output and
   ' the errors generated by the compare function
   sCmpOutput = CreateTempFile()
   sDiffOutput = CreateTempFile()

   ' Run the compare utility and redirect it's output and errors
   SyncShell ("cmd /c " & _
         GetShortName(App.Path & msCompareExe) & gstrBlank & _
         sCompareFile & gstrBlank & sOutputFile & _
         " > " & sCmpOutput & " 2> " & sDiffOutput)

   If FileLen(sDiffOutput)> 0 Then
      ' The compare generated errors - append error msgs to the error file
      Call AppendDiffErrors(sDiffOutput)
      CompareOutput = False
   Else
      CompareOutput = (FileLen(sCmpOutput) = 0)
   End If

   If Not CompareOutput Then
      mcVBErr.WriteError errDiffFailed
   End If

   ' Delete the temporary files used to store the output of the compare and
   ' the errors generated by the compare
   Kill sDiffOutput
   Kill sCmpOutput

   Exit Function

CompareOutputErr:
   mcVBErr.LogVBErrors
   CompareOutput = False

End Function
Public Property Get InstanceId() As Long

   InstanceId = mlngInstanceId

End Property
Public Property Let InstanceId(ByVal vdata As Long)

   mlngInstanceId = vdata

End Property

Private Function ExecuteConstraint(vntConstraints As Variant, _
      ByRef intLoopIndex As Integer) As Boolean

   ' Returns True if there is a constraint in the passed in
   ' array that remains to be executed

   If IsArray(vntConstraints) And Not IsEmpty(vntConstraints)Then
      ExecuteConstraint = (LBound(vntConstraints) <= intLoopIndex) And
(intLoopIndex <= UBound(vntConstraints))
   Else
      ExecuteConstraint = False
   End If

End Function
Private Function NextStep() As cStep

   ' Determines which is the next step to be executed - it could
   ' be either a pre-execution step, the worker step itself
   ' or a post-execution step

   Dim cConsRec As cConstraint
   Dim cNextStepRec As cStep
   Dim vntStepConstraints As Variant

   ' Static variable to remember exactly where we are in the
   ' processing
   Static intIndex As Integer
   Static intNextStepType As NextNodeType

   On Error GoTo NextStepErr

   If mblnAbort = True Then
      ' The user has aborted the run - do not run any more
      ' processes for the step
      Set NextStep = Nothing
      Exit Function
   End If

   If intNextStepType = 0 Then
      ' First time through this function - set the Index and
      ' node type to initial values
      intNextStepType = mintWspPreConstraint
      intIndex = 0
      RaiseEvent StepStart(mcStep, Determine64BitTime(), mlngInstanceId)
   End If

   Do
      Select Case intNextStepType
         Case mintWspPreConstraint
            vntStepConstraints = mcvntWspPreCons

         Case mintPreConstraint
            vntStepConstraints = mcvntPreCons

         Case mintStep
            ' CONS:
```

```
            If mcStep.StepType = gintWorkerStep Then
               Set cNextStepRec = mcStep
            End If

         Case mintWspPostConstraint
            vntStepConstraints = mcvntWspPostCons

         Case mintPostConstraint
            vntStepConstraints = mcvntPostCons

      End Select

      If intNextStepType <> mintStep Then
         ' Check if there is a constraint to be executed
         If ExecuteConstraint(vntStepConstraints, intIndex) Then
            ' Get the corresponding step record to be executed
            ' Query the global step record for the current
            ' constraint
            Set cConsRec = vntStepConstraints(intIndex)

            Set cNextStepRec = mcGlobals.QueryStep(cConsRec.GlobalStepId)
            intIndex = intIndex + 1
         Else
            If intNextStepType = mintPostConstraint Then
               ' No more stuff to be executed for the step
               ' Raise a Done event
               Set cNextStepRec = Nothing

               ' Set the next step type to an invalid value
               intNextStepType = -1
            Else
               Call NextType(intNextStepType, intIndex)
            End If
         End If
      Else
         ' Increment the step type so we look at the post-
         ' execution steps the next time through
         Call NextType(intNextStepType, intIndex)
      End If

   Loop Until (Not cNextStepRec Is Nothing) Or _
         intNextStepType = -1

   Set NextStep = cNextStepRec

   Exit Function

NextStepErr:
   ' Log the error code raised by Visual Basic
   Call LogErrors(Errors)
   On Error GoTo 0
   mstrSource = mstrModuleName & "NextStep"
   Err.Raise vbObjectError + errNextStepFailed, mstrSource, _
      LoadResString(errNextStepFailed)
End Function
Public Sub Execute()
   ' This procedure is the method that executes the step that
   ' is assigned to the ExecuteStep property. It call a procedure
   ' to determine the next step to be executed.
   ' Then it initializes all the properties of the cExecuteSM object
   ' and calls it's run method to execute it.
   Dim cConn As cConnection
   Dim cRunConnDtl As cConnDtl

   On Error GoTo ExecuteErr

   ' If this procedure is called after a step has completed,
   ' we would have to check if we created any temporary files
   ' while executing that step
   If Not mcExecStep Is Nothing Then
      If Not StringEmpty(mcExecStep.StepTextFile)Or
mcExecStep.ExecutionMechanism= gintExecuteShell Then
         ' Remove the temporary file that we created while
         ' running this command
         Kill mstrCommand
      End If

      Call StepCompleted

      ' The VB errors class stores a reference to the Execute class since it uses
      ' a method of the class to write errors to the error log. Hence,
      ' release all references to the Execute object before destroying it.
      Set mcVBErr.ErrorFile = Nothing
      Set mcExecObj = Nothing

      ' Delete empty output and error files (generated by shell commands)
      ' (Can be done only after cleaning up cExecObj)
      Call DeleteEmptyOutputFiles
   Else
      ' First time through - initialize the location of output files
      msOutputDir = GetDefaultDir(mcStep.WorkspaceId, WspParameters)
      msOutputDir = msOutputDir & gstrFileSeparator & Trim(Str(RunId)) &
gstrFileSeparator
      ' Dummy file since the function expects a file name
      MakePathValid (msOutputDir & "a.txt")
   End If

   ' Call a procedure to determine the next step to be executed
   ' - could be a constraint or the step itself
   ' Initialize a module-level variable to the step being
   ' executed
   Set mcExecStep = NextStep
   If mcExecStep Is Nothing Then
      RaiseEvent StepComplete(mcStep, Determine64BitTime(), mlngInstanceId,
miStatus)
      ' No more stuff to execute
      Exit Sub
   End If

   Dim sStartDir As String

   Set mcExecObj = New EXECUTEDLLLib.Execute

   ' The VB errors class uses the WriteError method of the Execute class to write
   ' all VB errors to the error file for the step (this prevents a clash when the
   ' VB errors and Execution errors have to be written to the same log). Hence,
store
   ' a reference to the Execute object in mcVBErr
   msErrorFile = GetOutputFile(gsErrorFileSuffix)
   mcExecObj.ErrorFile = msErrorFile
   Call DeleteFile(msErrorFile, bCheckIfEmpty:=False)
   Set mcVBErr.ErrorFile = mcExecObj

   If mcExecStep.ExecutionMechanism = gintExecuteShell Then
      sStartDir = Trim$(GetShortName(SubstituteParameters(_
            mcExecStep.StartDir, mcExecStep.WorkspaceId, mcIterators,
WspParameters:=WspParameters)))
      ' Dummy connection object
      Set cConn = New cConnection
      Set cRunConnDtl = New cConnDtl
   Else
      ' Find the connection string value and substitute parameter values in it
      Set cRunConnDtl =
WspConnDtls.GetConnectionDtl(mcExecStep.WorkspaceId,
mcExecStep.StartDir)
      Set cConn = WspConnections.GetConnection(mcExecStep.WorkspaceId,
cRunConnDtl.ConnectionString)
      sStartDir = Trim$(SubstituteParameters(cConn.ConnectionValue,_
            mcExecStep.WorkspaceId, mcIterators,
WspParameters:=WspParameters))
   End If

   msOutputFile = GetOutputFile(gsOutputFileSuffix)
   Call DeleteFile(msOutputFile, bCheckIfEmpty:=False)
   mcExecObj.OutputFile = msOutputFile
'   mcExecObj.LogFile = GetShortName(SubstituteParameters(_
'         mcExecStep.LogFile, mcExecStep.WorkspaceId, mcIterators,
WspParameters:=WspParameters))
   If mcExecStep.ExecutionMechanism = gintExecuteODBC And _
         cRunConnDtl.ConnType = ConnTypeDynamic Then
      Call mcExecObj.DoExecute(BuildCommandString(),sStartDir,
mcExecStep.ExecutionMechanism,_
            cConn.NoCountDisplay, cConn.NoExecute, cConn.ParseQueryOnly,
cConn.QuotedIdentifiers, _
            cConn.AnsiNulls, cConn.ShowQueryPlan, cConn.ShowStatsTime,
cConn.ShowStatsIO, _
```

```
            cConn.RowCount, cConn.QueryTimeOut, gstrEmptyString)
    Else
        Call mcExecObj.DoExecute(BuildCommandString(),sStartDir,
mcExecStep.ExecutionMechanism, _
            cConn.NoCountDisplay, cConn.NoExecute, cConn.ParseQueryOnly,
cConn.QuotedIdentifiers, _
            cConn.AnsiNulls, cConn.ShowQueryPlan, cConn.ShowStatsTime,
cConn.ShowStatsIO, _
            cConn.RowCount, cConn.QueryTimeOut, mcExecStep.StartDir)
    End If

    Exit Sub

ExecuteErr:
    Call HandleExecutionError

    ' We can assume that if we are in this function, a StepStart event has been
triggered already.
    RaiseEvent StepComplete(mcStep, Determine64BitTime(), mlngInstanceId,
miStatus)

End Sub
Private Function BuildCommandString()As String
    ' Process text to be executed - either from the text
    ' field or read it from a file.
    ' This function will always return the command text forODBC commands
    ' and a file name for Shell commands
    Dim sFile As String
    Dim sCommand As String
    Dim sTemp As String

    On Error GoTo BuildCommandStringErr

    If Not StringEmpty(mcExecStep.StepTextFile)Then
        ' Substitute parameter values and environment variables
        ' in the filename
        msRunStepDtl = SubstituteParameters(mcExecStep.StepTextFile,_
            mcExecStep.WorkspaceId, mcIterators,
WspParameters:=WspParameters)

        sFile = GetShortName(msRunStepDtl)

        mstrCommand = SubstituteParametersInText(sFile,
mcExecStep.WorkspaceId)

        If mcExecStep.ExecutionMechanism= gintExecuteODBC Then
            ' Read the contents of the file and pass it to ODBC
            BuildCommandString= ReadCommandFromFile(mstrCommand)
        Else
            BuildCommandString= mstrCommand
        End If
    Else
        ' Substitute parameter values and environment variables
        ' in the step text
        msRunStepDtl = SubstituteParameters(mcExecStep.StepText,_
            mcExecStep.WorkspaceId, mcIterators,
WspParameters:=WspParameters)
        mstrCommand = msRunStepDtl

        If mcExecStep.ExecutionMechanism= gintExecuteShell Then
            ' Write the command to a temp file (enables us to execute multiple
            ' commands via the command interpreter)
            mstrCommand = WriteCommandToFile(msRunStepDtl)
            BuildCommandString= mstrCommand
        Else
            BuildCommandString= SQLFixup(msRunStepDtl)
        End If
    End If

    If CreateInputFiles Then
        Call CreateStepTextFile
    End If

    Exit Function

BuildCommandStringErr:
    ' Log the error code raised by the Execute procedure
'    Call LogErrors(Errors)
    mcVBErr.LogVBErrors

    On Error GoTo 0
    mstrSource = mstrModuleName & "Execute"
    Err.Raise vbObjectError + errExecuteStepFailed, mstrSource, _
        LoadResString(errExecuteStepFailed)& mstrCommand

End Function
Public Sub Abort()

    On Error GoTo AbortErr

    ' Setting the Abort flag to True will ensure that we
    ' don't execute any more processes for this step
    mblnAbort = True

    If Not mcExecObj Is Nothing Then
        mcExecObj.Abort
    Else
        ' We are not in the middle of execution yet
    End If

    Exit Sub

AbortErr:
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errProgramError, _
        mstrModuleName & "Abort", _
        LoadResString(errProgramError)

End Sub
Private Sub NextType(ByRef StepType As NextNodeType, _
        ByRef Position As Integer)

    StepType = StepType + 1
    Position = 0

End Sub
Private Sub StepCompleted()

    On Error GoTo StepCompletedErr

    If Not mcExecStep Is Nothing Then
        If mcExecStep Is mcStep Then
            miStatus = InitializeExecStatus
            If miStatus = gintFailed Then
                ' Create input files if the step failed execution and one hasn't been
created already
                If Not CreateInputFiles Then CreateStepTextFile
                Call mcVBErr.WriteError(errExecuteStepFailed,_
                    OptArgs:="Continuation criteria for the step is: " &
gsContCriteria(mcStep.ContinuationCriteria))
            End If
        End If
    End If

    Exit Sub

StepCompletedErr:
    ' Log the error code raised by Visual Basic
    miStatus = gintFailed
    mcVBErr.LogVBErrors
    Call mcVBErr.WriteError(errExecuteStepFailed,_
        OptArgs:="Continuation criteria for the step is: " &
gsContCriteria(mcStep.ContinuationCriteria))

End Sub
Private Sub DeleteEmptyOutputFiles()

    On Error GoTo DeleteEmptyOutputFilesErr

    ' Delete empty output and error files
    If Not mcExecStep Is Nothing Then
        Call DeleteFile(msErrorFile, bCheckIfEmpty:=True)
        Call DeleteFile(msOutputFile, bCheckIfEmpty:=True)
    End If
```

```
    Exit Sub

DeleteEmptyOutputFilesErr:
    ' Not a critical error - continue

End Sub
Private Function ReadCommandFromFile(strFileNameAs String) As String

    ' Returns the contents of the passed in file

    Dim sCommand As String
    Dim sTemp As String
    Dim InputFile As Integer

    On Error GoTo ReadCommandFromFileErr

    If Not StringEmpty(strFileName)Then

        InputFile = FreeFile
        Open strFileName For Input Access Read As InputFile

        Line Input #InputFile, sCommand ' Read line into variable.

        Do While Not EOF(InputFile)' Loop until end of file.
            Line Input #InputFile, sTemp ' Read line into variable.
            sCommand = sCommand & vbCrLf & sTemp
        Loop

        Close InputFile
    End If

    ReadCommandFromFile= sCommand

    Exit Function

ReadCommandFromFileErr:

    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "ReadCommandFromFile"
    On Error GoTo 0
    Err.Raise vbObjectError + errSubValuesFailed, _
            gstrSource, _
            LoadResString(errSubValuesFailed)

End Function
Private Function SubstituteParametersIfPossible(strLabel As String)

    On Error GoTo SubstituteParametersIfPossibleErr

    SubstituteParametersIfPossible = SubstituteParameters(strLabel, _
            mcExecStep.WorkspaceId, mcIterators,
WspParameters:=WspParameters)
    Exit Function

SubstituteParametersIfPossibleErr:
    SubstituteParametersIfPossible = strLabel

End Function
Private Function SubstituteParametersInText(strFileNameAs String, _
        lngWorkspace As Long) As String

    ' Reads each line in the passed in file, substitutes parameter
    ' values in the line and writes out the modified line to a
    ' temporary file that we create. The temporary file will be
    ' removed once the step completes execution.
    ' Returns the name of the newly created temporary file.

    Dim strTempFile As String
    Dim strTemp As String
    Dim strOutput As String
    Dim InputFile As Integer
    Dim OutputFile As Integer

    On Error GoTo SubstituteParametersInTextErr

    strTempFile = CreateTempFile()

    If Not StringEmpty(strFileName)Then

        InputFile = FreeFile
        Open strFileName For Input Access Read As InputFile

        OutputFile = FreeFile
        Open strTempFile For Output Access Write As OutputFile

        Do While Not EOF(InputFile)' Loop until end of file.
            Line Input #InputFile, strTemp ' Read line into variable.
            strOutput = SubstituteParameters(strTemp, lngWorkspace, mcIterators,
WspParameters:=WspParameters)

            If mcExecStep.ExecutionMechanism= gintExecuteODBC Then strOutput
= SQLFixup(strOutput)

            Print #OutputFile, strOutput
            BugMessage strOutput
        Loop

    End If

    Close InputFile
    Close OutputFile

    SubstituteParametersInText = strTempFile

    Exit Function

SubstituteParametersInTextErr:

    ' Log the error code raised by Visual Basic
'   Call LogErrors(Errors)
    mcVBErr.LogVBErrors
    mstrSource = mstrModuleName & "SubstituteParametersInText"
    On Error GoTo 0
    Err.Raise vbObjectError + errSubValuesFailed, _
            gstrSource, _
            LoadResString(errSubValuesFailed)

End Function

Private Function WriteCommandToFile(sCommandAs String, Optional sFile As
String = gstrEmptyString) As String

    ' Writes the command text to a temporary file
    ' Returns the name of the temporary file

    Dim OutputFile As Integer

    On Error GoTo WriteCommandToFileErr

    If StringEmpty(sFile)Then
        sFile = CreateTempFile()
    End If

    OutputFile = FreeFile
    Open sFile For Output Access Write As OutputFile

    Print #OutputFile, sCommand

    Close OutputFile

    WriteCommandToFile= sFile

    Exit Function

WriteCommandToFileErr:

    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "WriteCommandToFile"
    On Error GoTo 0
    Err.Raise vbObjectError + errSubValuesFailed, _
            gstrSource, _
            LoadResString(errSubValuesFailed)

End Function
```

```
Public Property Get WspPreCons() As Variant
    WspPreCons = mcvntWspPreCons
End Property
Public Property Let WspPreCons(ByVal vdata As Variant)
    mcvntWspPreCons = vdata
End Property

Public Property Get WspPostCons() As Variant
    WspPostCons = mcvntWspPostCons
End Property
Public Property Let WspPostCons(ByVal vdata As Variant)
    mcvntWspPostCons = vdata
End Property

Public Property Get PreCons() As Variant
    PreCons = mcvntPreCons
End Property
Public Property Let PreCons(ByVal vdata As Variant)
    mcvntPreCons = vdata
End Property

Public Property Get PostCons() As Variant
    PostCons = mcvntPostCons
End Property
Public Property Let PostCons(ByVal vdata As Variant)
    mcvntPostCons = vdata
End Property

Public Property Set Globals(cRunSteps As cArrSteps)

    Set mcGlobals = cRunSteps

End Property
Public Property Set ExecuteStep(cRunStep As cStep)

    Set mcStep = cRunStep

End Property
Public Property Get Globals() As cArrSteps

    Set Globals = mcGlobals

End Property
Public Property Get ExecuteStep() As cStep

    Set ExecuteStep = mcStep

End Property
Public Property Set Iterators(vdata As cRunColIt)

    Set mcIterators = vdata

End Property
Private Sub Class_Initialize()

    ' Initialize the Abort flag to False
    mblnAbort = False
    Set mcVBErr = New cVBErrorsSM
    Set mcTermProcess = New cTermProcess

End Sub

Private Sub Class_Terminate()

    On Error GoTo Class_TerminateErr

    Set mcExecObj = Nothing
    Set mcVBErr = Nothing
    Set mcTermProcess = Nothing

    Exit Sub

Class_TerminateErr:
    Call LogErrors(Errors)

End Sub

Private Sub mcExecObj_Start(ByVal StartTime As Currency)
    ' Raise an event indicating that the step has begun execution
    RaiseEvent ProcessStart(mcExecStep, msRunStepDtl, StartTime,
mlngInstanceId)
End Sub

Private Sub mcExecObj_Complete(ByVal EndTime As Currency, ByVal Elapsed
As Long)

    On Error GoTo mcExecObj_CompleteErr

    Debug.Print Elapsed
    RaiseEvent ProcessComplete(mcExecStep, EndTime, mlngInstanceId,
Elapsed)
    mcTermProcess.ProcessTerminated

    Exit Sub

mcExecObj_CompleteErr:
    Call LogErrors(Errors)

End Sub

Private Sub mcTermProcess_TermProcessExists()

    On Error GoTo TermProcessExistsErr

    ' Call a procedure to execute the next step, if any
    Call Execute

    Exit Sub

TermProcessExistsErr:
    ' Log the error code raised by the Execute procedure
    Call LogErrors(Errors)

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cRunWorkspace"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cRunWorkspace.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This class loads all the information necessary to
'              execute a workspace and calls cRunInst to execute the workspace.
'              It also propagates Step start and complete and
'              Run start and complete events.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "cRunWorkspace."
Private mstrSource As String

Private mcRunSteps As cArrSteps
Private mcRunParams As cArrParameters
Private mcRunConstraints As cArrConstraints
Private mcRunConnections As cConnections
Private mcRunConnDtls As cConnDtls
Private mcvntWspPreCons As Variant
Private mcvntWspPostCons As Variant
Private mdbsLoadDb As Database
Private mlngRunId As Long
Private mlngWorkspaceId As Long
Private mField As cStringSM
Public CreateInputFiles As Boolean
```

```
Private WithEvents mcRun As cRunInst
Attribute mcRun.VB_VarHelpID = -1

Public Event RunStart(dtmStartTime As Currency, strWspLog As String)
Public Event RunComplete(dtmEndTime As Currency)
Public Event StepStart(cStepRecord As cStep, dtmStartTime As Currency,
lngInstanceId As Long, _
    sPath As String, sIts As String)
Public Event StepComplete(cStepRecord As cStep, dtmEndTime As Currency,
lngInstanceId As Long)
Public Event ProcessStart(cStepRecord As cStep, strCommand As String, _
    dtmStartTime As Currency, lngInstanceId As Long)
Public Event ProcessComplete(cStepRecord As cStep, dtmEndTime As Currency,
lngInstanceId As Long)
Public Function InstancesForStep(lngStepId As Long, iStatus As InstanceStatus)
As cInstances
  ' Returns an array of all the instances for a step

  If mcRun Is Nothing Then
    Set InstancesForStep = Nothing
  Else
    Set InstancesForStep = mcRun.InstancesForStep(lngStepId, iStatus)
  End If

End Function
Private Sub InsertRunDetail(cStepRecord As cStep, _
    strCommand As String, dtmStartTime As Currency, _
    lngInstanceId As Long, lParentInstanceId As Long, sItValue As String)
  ' Inserts a new run detail record into the database

  Dim strInsert As String
  Dim qy As QueryDef

  On Error GoTo InsertRunDetailErr
  mstrSource = mstrModuleName & "InsertRunDetail"

  strInsert = "insert into run_step_details " & _
      "( run_id, step_id, version_no, instance_id, parent_instance_id, " & _
      " command, start_time, iterator_value ) " & _
      " values ( "

#If USE_JET Then

  strInsert = strInsert & " [r_id], [s_id], [ver_no], [i_id], [p_i_id], " & _
      " [com], [s_date], [it_val] )"

  Set qy = mdbsLoadDb.CreateQueryDef( _
      gstrEmptyString, strInsert)

  ' Call a procedure to assign the Querydef parameters
  Call AssignParameters(qy, StartTime:=dtmStartTime,_
      StepId:=cStepRecord.StepId, _
      Version:=cStepRecord.VersionNo, _
      InstanceId:=lngInstanceId, _
      Command:=strCommand)

  qy.Execute dbFailOnError
  qy.Close

#Else

  strInsert = strInsert & Str(mlngRunId) _
    & ", " & Str(cStepRecord.StepId) _
    & ", " & mField.MakeStringFieldValid(cStepRecord.VersionNo)_
    & ", " & Str(lngInstanceId) _
    & ", " & Str(lParentInstanceId) _
    & ", " & mField.MakeStringFieldValid(strCommand)_
    & ", " & Str(dtmStartTime) _
    & ", " & mField.MakeStringFieldValid(sItValue)

  strInsert = strInsert & " ) "

  mdbsLoadDb.Execute strInsert, dbFailOnError


#End If
  Exit Sub

InsertRunDetailErr:
  LogErrors Errors
  mstrSource = mstrModuleName & "InsertRunDetail"
  On Error GoTo 0
  Err.Raise vbObjectError + errUpdateRunDataFailed, _
      mstrSource, _
      LoadResString(errUpdateRunDataFailed)

End Sub
Private Sub UpdateRunDetail(cStepRecord As cStep, _
    dtmEndTime As Currency, lngInstanceId As Long, lElapsed As Long)
  ' Updates the run detail record in the database

  Dim strUpdate As String
  Dim qy As QueryDef

  On Error GoTo UpdateRunDetailErr

  strUpdate = "update run_step_details " & _
      " set end_time = [e_date], elapsed_time = [elapsed] " & _
      " where run_id = [r_id] " & _
      " and step_id = [s_id] " & _
      " and version_no = [ver_no] " & _
      " and instance_id = [i_id] "

  Set qy = mdbsLoadDb.CreateQueryDef( _
      gstrEmptyString, strUpdate)

  ' Call a procedure to assign the Querydef parameters
  Call AssignParameters(qy, EndTime:=dtmEndTime,_
      StepId:=cStepRecord.StepId, _
      Version:=cStepRecord.VersionNo, _
      InstanceId:=lngInstanceId, Elapsed:=lElapsed)

  qy.Execute dbFailOnError
  qy.Close

  Exit Sub

UpdateRunDetailErr:
  LogErrors Errors
  mstrSource = mstrModuleName & "UpdateRunDetail"
  On Error GoTo 0
  Err.Raise vbObjectError + errUpdateRunDataFailed, _
      mstrSource, _
      LoadResString(errUpdateRunDataFailed)

End Sub
Private Function InsertRunHeader(dtmStartTime As Currency) As Long
  ' Inserts a new run header record into the database
  ' and returns the id for the run

  Dim strInsert As String
  Dim qy As QueryDef

  On Error GoTo InsertRunHeaderErr

  strInsert = "insert into run_header " & _
    "( run_id, workspace_id, start_time ) " & _
    " values ( " & _
    " [r_id], [w_id], [s_date] )"

  Set qy = mdbsLoadDb.CreateQueryDef( _
      gstrEmptyString, strInsert)

  ' Call a procedure to execute the Querydef object
  Call AssignParameters(qy, StartTime:=dtmStartTime)

  qy.Execute dbFailOnError
  qy.Close

  InsertRunHeader = mlngRunId
  Exit Function

InsertRunHeaderErr:
  LogErrors Errors
```

```
    mstrSource = mstrModuleName & "InsertRunHeader"
    On Error GoTo 0
    Err.Raise vbObjectError + errUpdateRunDataFailed, _
          mstrSource, _
          LoadResString(errUpdateRunDataFailed)

End Function
Private Sub InsertRunParameters(dtmStartTime As Currency)
    ' Inserts a new run header record into the database
    ' and returns the id for the run

    Dim strInsert As String
    Dim qy As QueryDef
    Dim cParamRec As cParameter
    Dim lngIndex As Long

    On Error GoTo InsertRunParametersErr

    strInsert = "insert into run_parameters " & _
       "( run_id, parameter_name, parameter_value ) " & _
       " values ( " & _
       " [r_id], [p_name], [p_value] )"

    Set qy = mdbsLoadDb.CreateQueryDef( _
          gstrEmptyString, strInsert)
    qy.Parameters("r_id").Value = mlngRunId

    For lngIndex = 0 To mcRunParams.ParameterCount - 1
       Set cParamRec = mcRunParams(lngIndex)

       qy.Parameters("p_name").Value = cParamRec.ParameterName
       qy.Parameters("p_value").Value = cParamRec.ParameterValue
       qy.Execute dbFailOnError

    Next lngIndex

    qy.Close

    Exit Sub

InsertRunParametersErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "InsertRunParameters"
    On Error GoTo 0
    Err.Raise vbObjectError + errUpdateRunDataFailed, _
          mstrSource, _
          LoadResString(errUpdateRunDataFailed)

End Sub
Private Sub AssignParameters(qyExec As DAO.QueryDef, _
       Optional StartTime As Currency = 0, _
       Optional EndTime As Currency = 0, _
       Optional StepId As Long = 0, _
       Optional Version As String = gstrEmptyString, _
       Optional InstanceId As Long = 0, _
       Optional ParentInstanceId As Long = 0, _
       Optional Command As String = gstrEmptyString, _
       Optional Elapsed As Long = 0, _
       Optional ItValue As String = gstrEmptyString)
    ' Assigns values to the parameters in the querydef object

    Dim prmParam As DAO.Parameter

    On Error GoTo AssignParametersErr
    mstrSource = mstrModuleName & "AssignParameters"

    For Each prmParam In qyExec.Parameters
       Select Case prmParam.Name
          Case "[w_id]"
             prmParam.Value = mlngWorkspaceId

          Case "[r_id]"
             prmParam.Value = mlngRunId

          Case "[s_id]"
             BugAssert StepId <> 0
             prmParam.Value = StepId

          Case "[ver_no]"
             BugAssert Not StringEmpty(Version)
             prmParam.Value = Version

          Case "[i_id]"
             BugAssert InstanceId <> 0
             prmParam.Value = InstanceId

          Case "[p_i_id]"
             prmParam.Value = ParentInstanceId

          Case "[com]"
             BugAssert Not StringEmpty(Command)
             prmParam.Value = Command

          Case "[s_date]"
             BugAssert StartTime <> 0
             prmParam.Value = StartTime

          Case "[e_date]"
             BugAssert EndTime <> 0
             prmParam.Value = EndTime

          Case "[elapsed]"
             prmParam.Value = Elapsed

          Case "[it_val]"
             prmParam.Value = ItValue

          Case Else
             ' Write the parameter name that is faulty
             WriteError errInvalidParameter, mstrSource, _
                   prmParam.Name
             On Error GoTo 0
             Err.Raise errInvalidParameter, mstrSource, _
                   LoadResString(errInvalidParameter)
       End Select
    Next prmParam

    Exit Sub

AssignParametersErr:

    mstrSource = mstrModuleName & "AssignParameters"
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errAssignParametersFailed, _
          mstrSource, LoadResString(errAssignParametersFailed)

End Sub
Private Sub RunStartProcessing(dtmStartTime As Currency)

    On Error GoTo RunStartProcessingErr

    ' Insert the run header into the database
    Call InsertRunHeader(dtmStartTime)

    ' Insert the run parameters into the database
    Call InsertRunParameters(dtmStartTime)

    Exit Sub

RunStartProcessingErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "RunStartProcessing"
    ShowError errUpdateRunDataFailed
    WriteError errUpdateRunDataFailed, mstrSource

End Sub
Private Sub ProcessStartProcessing(cStepRecord As cStep, _
       strCommand As String, dtmStartTime As Currency, lngInstanceId As Long,
_
       lParentInstanceId As Long, sItValue As String)

    On Error GoTo ProcessStartProcessingErr

    ' Insert the run detail into the database
```

```
    Call InsertRunDetail(cStepRecord, strCommand, dtmStartTime, lngInstanceId,
_
            lParentInstanceId, sItValue)

    Exit Sub

ProcessStartProcessingErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "ProcessStartProcessing"
    ShowError errUpdateRunDataFailed
    WriteError errUpdateRunDataFailed, mstrSource

End Sub
Private Sub StepStartProcessing(cStepRecord As cStep, dtmStartTime As
Currency, _
        lngInstanceId As Long, lParentInstanceId As Long, sItValue As String)

    On Error GoTo StepStartProcessingErr

    ' Since ProcessStart events won't be triggered for manager steps
    If cStepRecord.StepType = gintManagerStep Then
        ' Insert the run detail into the database
        Call InsertRunDetail(cStepRecord, cStepRecord.StepLabel, _
                dtmStartTime, lngInstanceId, lParentInstanceId, sItValue)
    End If

    Exit Sub

StepStartProcessingErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "StepStartProcessing"
    ShowError errUpdateRunDataFailed

End Sub
Private Sub ProcessCompleteProcessing(cStepRecord As cStep, _
        dtmStartTime As Currency, lngInstanceId As Long, lElapsed As Long)

    On Error GoTo ProcessCompleteProcessingErr

    ' Insert the run detail into the database
    Call UpdateRunDetail(cStepRecord, dtmStartTime, lngInstanceId, lElapsed)

    Exit Sub

ProcessCompleteProcessingErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "ProcessCompleteProcessing"
    ShowError errUpdateRunDataFailed

End Sub
Private Sub StepCompleteProcessing(cStepRecord As cStep, _
        dtmEndTime As Currency, lngInstanceId As Long, lElapsed As Long)

    On Error GoTo StepCompleteProcessingErr

    ' Since ProcessComplete events won't be triggered for manager steps
    If cStepRecord.StepType = gintManagerStep Then
        ' Update the run detail in the database
        Call UpdateRunDetail(cStepRecord, dtmEndTime, lngInstanceId, lElapsed)
    End If

    Exit Sub

StepCompleteProcessingErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    ShowError errUpdateRunDataFailed

End Sub
Private Sub RunCompleteProcessing(dtmEndTimeAs Currency)

    On Error GoTo RunCompleteProcessingErr

    ' Update the header record with the end time for the run
    Call UpdateRunHeader(dtmEndTime)

    Exit Sub

RunCompleteProcessingErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    ShowError errUpdateRunDataFailed

End Sub

Private Sub UpdateRunHeader(ByVal dtmEndTime As Currency)
    ' Updates the run header record with the end date

    Dim strUpdate As String
    Dim qy As QueryDef

    On Error GoTo UpdateRunHeaderErr

    strUpdate = "update run_header " & _
            " set end_time = [e_date] " & _
            " where run_id = [r_id] "

    Set qy = mdbsLoadDb.CreateQueryDef( _
            gstrEmptyString, strUpdate)

    ' Call a procedure to execute the Querydef object
    Call AssignParameters(qy, EndTime:=dtmEndTime)

    qy.Execute dbFailOnError
    qy.Close

    Exit Sub

UpdateRunHeaderErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "UpdateRunHeader"
    On Error GoTo 0
    Err.Raise vbObjectError + errUpdateRunDataFailed, _
            mstrSource, _
            LoadResString(errUpdateRunDataFailed)

End Sub

Public Property Let WorkspaceId(ByVal vdata As Long)
    mlngWorkspaceId = vdata
End Property
Public Property Get WorkspaceId() As Long
    WorkspaceId = mlngWorkspaceId
End Property
Public Sub RunWorkspace()

    Dim cRunSeq As cSequence

    On Error GoTo RunWorkspaceErr

    ' Call a procedure to load the module-level structures
    ' with all the step and parameter data for the run
    If LoadRunData = False Then
        ' Error handled by the function already
        Exit Sub
    End If

    ' Retrieve the next identifier using the sequence class
    Set cRunSeq = New cSequence
    Set cRunSeq.IdDatabase = dbsAttTool
    cRunSeq.IdentifierColumn= "run_id"
    mlngRunId = cRunSeq.Identifier
    Set cRunSeq = Nothing

    Set mcRun.Constraints = mcRunConstraints
    mcRun.WspPreExecution = mcvntWspPreCons
    mcRun.WspPostExecution = mcvntWspPostCons

    Set mcRun.Steps = mcRunSteps
    Set mcRun.Parameters = mcRunParams
    Set mcRun.RunConnections = mcRunConnections
    Set mcRun.RunConnDtls = mcRunConnDtls
```

```
    mcRun.WspId = mlngWorkspaceId
    mcRun.RootKey = LabelStep(mlngWorkspaceId)
    mcRun.RunId = mlngRunId
    mcRun.CreateInputFiles = CreateInputFiles

    mcRun.Run

    Exit Sub

RunWorkspaceErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)

End Sub
Public Property Get LoadDb() As Database

    Set LoadDb = mdbsLoadDb

End Property
Public Property Set LoadDb(vdata As Database)

    Set mdbsLoadDb = vdata

End Property
Private Function LoadRunData() As Boolean

    ' Loads the step, parameter and constraint arrays
    ' with all the data for the workspace. Returns False
    ' if a failure occurs

    Dim strWorkspaceName As String
    Dim recWspSteps As Recordset
    Dim qySteps As DAO.QueryDef
    Dim recWspParams As Recordset
    Dim qyParams As DAO.QueryDef
    Dim recWspConns As Recordset
    Dim qyConns As DAO.QueryDef
    Dim recWspConnDtls As Recordset
    Dim qyConnDtls As DAO.QueryDef

    On Error GoTo LoadRunDataErr

    Set mcRunSteps.StepDB = mdbsLoadDb
    Set mcRunParams.ParamDatabase = mdbsLoadDb
    Set mcRunConstraints.ConstraintDB = mdbsLoadDb
    Set mcRunConnections.ConnDb = mdbsLoadDb
    Set mcRunConnDtls.ConnDb = mdbsLoadDb

    ' Read all the step and parameter data for the workspace
    Call ReadWorkspaceData(mlngWorkspaceId, mcRunSteps, _
          mcRunParams, mcRunConstraints, mcRunConnections, mcRunConnDtls,
_
          recWspSteps, qySteps, recWspParams, qyParams, recWspConns,
qyConns, _
          recWspConnDtls, qyConnDtls)

    ' Load all the pre- and post-execution constraints that
    ' have been defined for the workspace
    mcvntWspPreCons = mcRunConstraints.ConstraintsForWsp(_
       mlngWorkspaceId, _
       gintPreStep, _
       blnSort:=True, _
       blnGlobalConstraintsOnly:=True)
    mcvntWspPostCons = mcRunConstraints.ConstraintsForWsp(_
       mlngWorkspaceId, _
       gintPostStep, _
       blnSort:=True, _
       blnGlobalConstraintsOnly:=True)

    On Error Resume Next
    recWspSteps.Close
    qySteps.Close
    recWspParams.Close
    qyParams.Close
    recWspConns.Close
    qyConns.Close

    LoadRunData = True

    Exit Function

LoadRunDataErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    ShowError errLoadRunDataFailed
    LoadRunData = False

End Function
Public Sub StopRun()

    On Error GoTo StopRunErr

    If mcRun Is Nothing Then
       ' We haven't been the run yet, so do nothing
    Else
       mcRun.StopRun
    End If

    Exit Sub

StopRunErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    ' Errors would have been displayed by the called process

End Sub
Public Sub AbortRun()

    On Error GoTo AbortRunErr

    If mcRun Is Nothing Then
       ' We haven't been the run yet, so do nothing
    Else
       mcRun.Abort
    End If

    Exit Sub

AbortRunErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    ' Errors would have been displayed by the called process

End Sub

Private Sub Class_Initialize()

    ' Create instances of the step, parameter and constraint arrays
    Set mcRunSteps = New cArrSteps
    Set mcRunParams = New cArrParameters
    Set mcRunConstraints = New cArrConstraints
    Set mcRunConnections = New cConnections
    Set mcRunConnDtls = New cConnDtls
    Set mcRun = New cRunInst
    Set mField = New cStringSM

End Sub
Private Sub Class_Terminate()

    On Error GoTo UnLoadRunDataErr

    ' Clears the step, parameter and constraint arrays
    Set mcRunSteps = Nothing
    Set mcRunParams = Nothing
    Set mcRunConstraints = Nothing
    Set mcRunConnections = Nothing
    Set mcRunConnDtls = Nothing

    Set mcRun = Nothing
    Set mdbsLoadDb = Nothing
    Set mField = Nothing

    Exit Sub

UnLoadRunDataErr:
    ' Log the error code raised by Visual Basic
```

```
    Call LogErrors(Errors)
    ' Not a critical error - continue
    Resume Next

End Sub

Private Sub mcRun_ProcessComplete(cStepRecord As cStep, _
        dtmEndTime As Currency, lngInstanceId As Long, lElapsed As Long)

    RaiseEvent ProcessComplete(cStepRecord, dtmEndTime, lngInstanceId)
    Call ProcessCompleteProcessing(cStepRecord, dtmEndTime, lngInstanceId,
lElapsed)

End Sub

Private Sub mcRun_ProcessStart(cStepRecord As cStep, _
        strCommand As String, dtmStartTime As Currency, lngInstanceId As Long,
_
        lParentInstanceId As Long, sItValue As String)

    RaiseEvent ProcessStart(cStepRecord, strCommand, dtmStartTime,
lngInstanceId)
    Call ProcessStartProcessing(cStepRecord, strCommand, dtmStartTime,
lngInstanceId, _
            lParentInstanceId, sItValue)

End Sub

Private Sub mcRun_RunComplete(dtmEndTimeAs Currency)

    Debug.Print "Run ended at: " & CStr(dtmEndTime)
    Call RunCompleteProcessing(dtmEndTime)

    RaiseEvent RunComplete(dtmEndTime)

End Sub
Private Sub mcRun_RunStart(dtmStartTimeAs Currency, strWspLog As String)

    RaiseEvent RunStart(dtmStartTime, strWspLog)
    Debug.Print "Run started at: " & CStr(dtmStartTime)

    Call RunStartProcessing(dtmStartTime)

End Sub
Private Sub mcRun_StepComplete(cStepRecord As cStep, _
        dtmEndTime As Currency, lngInstanceId As Long, lElapsed As Long)

    RaiseEvent StepComplete(cStepRecord, dtmEndTime, lngInstanceId)
'    BugMessage "Step: " & cStepRecord.StepLabel & " has completed!"

    Call StepCompleteProcessing(cStepRecord, dtmEndTime, lngInstanceId,
lElapsed)

End Sub
Private Sub mcRun_StepStart(cStepRecord As cStep, dtmStartTime As Currency,
_
        lngInstanceId As Long, lParentInstanceId As Long, sPath As String, sIts As
String, sItValue As String)

    RaiseEvent StepStart(cStepRecord, dtmStartTime, lngInstanceId, sPath, sIts)
    'bugmessage "Step: " & cStepRecord.StepLabel & " has started."

    Call StepStartProcessing(cStepRecord, dtmStartTime, lngInstanceId,
lParentInstanceId, sItValue)

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cSequence"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cSequence.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This class uses the att_identifiers table to generate unique
'              identifiers.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mlngIdentifier As Long
Private mstrIdentifierColumn As String
Private mrecIdentifiers As Recordset
Private mdbsDatabase As Database

Private Const mstrEmptyString = ""

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cSequence."

Private Sub CreateIdRecord()
    ' Creates a record with all identifiers having an initial value of 1

    Dim sSql As String
    Dim pId As DAO.Parameter
    Dim qyId As DAO.QueryDef

    sSql = "insert into att_identifiers (" & _
           " workspace_id, parameter_id, step_id, " & _
           " constraint_id, run_id, connection_id ) values ( " & _
           "[w_id], [p_id], [s_id], [c_id], [r_id], [conn_id] )"
    Set qyId = mdbsDatabase.CreateQueryDef(gstrEmptyString, sSql)
    For Each pId In qyId.Parameters
        pId.Value = glMinId
    Next pId
    qyId.Execute dbFailOnError
    qyId.Close

End Sub
Private Sub CreateIdRecordset()

    Dim strSql As String

    ' Initialize the recordset with all identifiers
    strSql = "select * from att_identifiers"
    Set mrecIdentifiers = mdbsDatabase.OpenRecordset(strSql,
dbOpenForwardOnly)

    If mrecIdentifiers.RecordCount = 0 Then
        CreateIdRecord
        Set mrecIdentifiers = mdbsDatabase.OpenRecordset(strSql,
dbOpenForwardOnly)
    End If

    BugAssert mrecIdentifiers.RecordCount <> 0

End Sub

Public Property Set IdDatabase(vdata As Database)

    Set mdbsDatabase = vdata

End Property

Public Property Let IdentifierColumn(vdataAs String)

    Dim intIndex As Integer

    On Error GoTo IdentifierColumnErr

    ' Initialize the return value to an empty string
    mstrIdentifierColumn = mstrEmptyString
    Call CreateIdRecordset

    For intIndex = 0 To mrecIdentifiers.Fields.Count - 1

        If LCase(Trim(mrecIdentifiers.Fields(intIndex).Name)) = _
```

```
            LCase(Trim(vdata)) Then

          ' Valid column name
          mstrIdentifierColumn= vdata
          Exit Property
        End If

    Next intIndex

    BugAssert True, "Invalid column name!"

    Exit Property

IdentifierColumnErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "IdentifierColumn"
    On Error GoTo 0
    Err.Raise vbObjectError + errIdentifierColumnFailed,_
          mstrSource, _
          LoadResString(errIdentifierColumnFailed)

End Property
Public Property Get Identifier() As Long
    Dim strSql As String

    On Error GoTo GetIdentifierErr

    BugAssert mstrIdentifierColumn<> mstrEmptyString

    ' Increment the identifier column by 1
    strSql = "update att_identifiers " & _
       " set " & mstrIdentifierColumn& _
       " = " & mstrIdentifierColumn& " + 1"
    mdbsDatabase.Execute strSql, dbFailOnError

    ' Refresh the recordset with identifier values
    Call CreateIdRecordset

    mlngIdentifier= mrecIdentifiers.Fields(mstrIdentifierColumn).Value

    Identifier = mlngIdentifier

    Exit Property

GetIdentifierErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Identifier"
    On Error GoTo 0
    Err.Raise vbObjectError + errGetIdentifierFailed, _
          mstrSource, _
          LoadResString(errGetIdentifierFailed)

End Property
Private Sub Class_Terminate()

    mrecIdentifiers.Close

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cStack"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cStack.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:     This class implements a stack of objects.
'  Contact:     Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "cStack."
Private mstrSource As String

Private mcVector As cVector
Private mlngCount As Long
Public Property Get Item(ByVal Position As Long) As Object
Attribute Item.VB_UserMemId = 0

    Set Item = mcVector(Position)

End Property

Public Sub Push(objToPush As Object)

    mcVector.Add objToPush

End Sub
Public Sub Clear()

    mcVector.Clear

End Sub

Public Function Pop() As Object

    If mcVector.Count > 0 Then
       Set Pop =  mcVector.Delete(mcVector.Count - 1)
    Else
       Set Pop = Nothing
    End If

End Function
Public Function Count() As Long

    Count = mcVector.Count

End Function


Private Sub Class_Initialize()

    Set mcVector = New cVector

End Sub


Private Sub Class_Terminate()

    Set mcVector = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cStep"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
Attribute VB_Ext_KEY = "SavedWithClassBuilder","Yes"
Attribute VB_Ext_KEY = "Top_Level","Yes"
'  FILE:       cStep.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:     Encapsulates the properties and methods of a step.
'              Contains functions to insert, update and delete
'              att_steps records from the database.
'  Contact:     Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Local variable(s) to hold property value(s)
```

```
Private mlngStepId As Long
Private mstrVersionNo As String
Private mstrStepLabel As String
Private mstrStepTextFile As String
Private mstrStepText As String
Private mstrStartDir As String
Private mlngWorkspaceId As Integer
Private mlngParentStepId As Integer
Private mstrParentVersionNo As String
Private mintSequenceNo As Integer
Private mintStepLevel As Integer
Private mblnEnabledFlag As Boolean
Private mstrDegreeParallelism As String
Private mintExecutionMechanism As Integer
Private mstrFailureDetails As String
Private mintContinuationCriteria As Integer
Private mblnGlobalFlag As Boolean
Private mblnArchivedFlag As Boolean
Private mstrOutputFile As String
'Private mstrLogFile As String
Private mstrErrorFile As String
Private mdbsDatabase As Database
Private mintStepType As Integer
Private mintOperation As Operation
Private mlngPosition As Long
Private mstrIteratorName As String
Private mcIterators As cNodeCollections
Private mbIsNewVersion As Boolean
Private msOldVersion As String

' The following constants are used throughout the project to
' indicate the different options selected by the user
' The options are presented to the user as control arrays of
' option buttons. These constants have to be in sync with the
' indexes of the option buttons.
' All the control arrays have an lbound of 1. The value 0 is
' used to indicate that the property being represented by the
' control array is not valid for the step
' Public enums are used since we cannot expose public constants
' in class modules. gintNoOption is applicable to all enums,
' but declared in the Execution method enum, since we cannot
' declare it more than once.

' Is here as a comment
' Has been defined in public.bas with the other object types
'Public Enum gintStepType
'    gintGlobalStep = 3
'    gintManagerStep
'    gintWorkerStep
'End Enum

' Execution Method options
Public Enum ExecutionMethod
    gintNoOption = 0
    gintExecuteODBC
    gintExecuteShell
End Enum

' Failure criteria options
Public Enum FailureCriteria
    gintFailureODBC = 1
    gintFailureTextCompare
End Enum

' Continuation criteria options
' Note: Update the initialization of gsContCriteria in Initialize() if the
' continuation criteria are modified
Public Enum ContinuationCriteria
    gintOnFailureAbort = 1
    gintOnFailureContinue
    gintOnFailureCompleteSiblings
    gintOnFailureAbortSiblings
    gintOnFailureSkipSiblings
    gintOnFailureAsk
End Enum

' The initial version #
Private Const mstrMinVersion As String = "0.0"

' End of constants for option button control arrays
' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cStep."

' The cSequence class is used to generate unique step identifiers
Private mStepSeq As cSequence

' The StringSM class is used to carry out string operations
Private mFieldValue As cStringSM
Private Sub NewVersion()

mbIsNewVersion = True
msOldVersion = mstrVersionNo

End Sub
Public Function IsNewVersion() As Boolean
    IsNewVersion = mbIsNewVersion
End Function

Public Function OldVersionNo() As String
    OldVersionNo = msOldVersion
End Function

Public Sub SaveIterators()
    ' This procedure checks if any changes have been made
    ' to the iterators for the step. If so, it calls the
    ' methods of the iterator class to commit the changes
    Dim cItRec As cIterator
    Dim lngIndex As Long

    On Error GoTo SaveIteratorsErr

    For lngIndex = 0 To mcIterators.Count - 1
        Set cItRec = mcIterators(lngIndex)

        Select Case cItRec.IndOperation
            Case QueryOp
                ' No changes were made to the queried Step.
                ' Do nothing

            Case InsertOp
                cItRec.Add mlngStepId, mstrVersionNo
                cItRec.IndOperation = QueryOp

            Case UpdateOp
                cItRec.Update mlngStepId, mstrVersionNo
                cItRec.IndOperation = QueryOp

            Case DeleteOp
                cItRec.Delete mlngStepId, mstrVersionNo
                ' Remove the record from the collection
                mcIterators.Delete lngIndex

        End Select
    Next lngIndex

    Exit Sub

SaveIteratorsErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "SaveIterators"
    On Error GoTo 0
    Err.Raise vbObjectError + errSaveFailed, _
          mstrSource, _
          LoadResString(errSaveFailed)

End Sub
Public Property Get IndOperation() As Operation

    IndOperation = mintOperation

End Property
Public Property Let IndOperation(ByVal vdata As Operation)
```

```
    BugAssert vdata = QueryOp Or vdata = InsertOp Or vdata = UpdateOp Or
vdata = DeleteOp, "Invalid operation"
    mintOperation = vdata

End Property

Public Function Iterators() As Variant
    ' Returns a variant containing all the iterators that
    ' have been defined for the step

    Dim cStepIterators() As cIterator
    Dim cTempIt As cIterator
    Dim lngIndex As Long
    Dim lngItCount As Long

    On Error GoTo IteratorsErr

    lngItCount = 0
    For lngIndex = 0 To mcIterators.Count - 1
        ' Increase the array dimension and add the constraint
        ' to it
        Set cTempIt = mcIterators(lngIndex)

        If cTempIt.IndOperation <> DeleteOp Then
            ReDim Preserve cStepIterators(lngItCount)
            Set cStepIterators(lngItCount) = cTempIt
            lngItCount = lngItCount + 1
        End If

    Next lngIndex

    If lngItCount = 0 Then
        Iterators = Empty
    Else
        Iterators = cStepIterators()
    End If

    Call QuickSort(Iterators)

    Exit Function

IteratorsErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errIteratorsFailed, _
          mstrModuleName & "Iterators", _
          LoadResString(errIteratorsFailed)

End Function
Public Function IteratorCount() As Long
    ' Returns a count of all the iterators for the step

    Dim lngItCount As Long
    Dim lngIndex As Long
    Dim cTempIt As cIterator

    On Error GoTo IteratorsErr

    lngItCount = 0
    For lngIndex = 0 To mcIterators.Count - 1

        If mcIterators(lngIndex).IndOperation <> DeleteOp Then
            lngItCount = lngItCount + 1
        End If

    Next lngIndex

    IteratorCount = lngItCount

    Exit Function

IteratorsErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errIteratorsFailed, _
          mstrSource, _
          LoadResString(errIteratorsFailed)

End Function
Public Sub Validate()
    ' Each distinct object will have a Validate method which
    ' will check if the class properties are valid. This method
    ' will be used to check interdependant properties that
    ' cannot be validated by the let procedures.
    ' It should be called by the add and modify methods of the class

    ' Check if the step label has been specified
    If StringEmpty(mstrStepLabel)Then
        ShowError errStepLabelMandatory
        On Error GoTo 0
        Err.Raise vbObjectError + errValidateFailed, _
              "Validate", LoadResString(errValidateFailed)
    End If

    If Not IsStringEmpty(mstrStepText)And Not IsStringEmpty(mstrStepTextFile)
Then
        ShowError errStepTextOrFile
        On Error GoTo 0
        Err.Raise vbObjectError + errStepTextOrFile, _
            "Validate", LoadResString(errStepTextOrFile)
    End If

End Sub

Public Function IncVersionY() As String
    ' The version number for a step is stored in the x.y
    ' format where x is the parent component and y is the
    ' child component of the step. This function will increment
    ' the y component of the step by 1

    On Error GoTo IncVersionYErr

    ' Store the old version number for the step
    Call NewVersion

    mstrVersionNo = Trim$(Str$(GetX(mstrVersionNo)))& gstrVerSeparator & _
          Trim$(Str(GetY(mstrVersionNo)+ 1))
    IncVersionY = mstrVersionNo

    Exit Function

IncVersionYErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "IncVersionY"
    On Error GoTo 0
    Err.Raise vbObjectError + errIncVersionYFailed, _
          gstrSource, _
          LoadResString(errIncVersionYFailed)

End Function
Public Function IncVersionX() As String
    ' The version number for a step is stored in the x.y
    ' format where x is the parent component and y is the
    ' child component of the step. This function will increment
    ' the y component of the step by 1 and reset the x component
    ' to 0

    On Error GoTo IncVersionXErr

    ' Store the old version number for the step
    Call NewVersion

    mstrVersionNo = Trim$(Str$(GetX(mstrVersionNo)+ 1)) & gstrVerSeparator
& "0"
    IncVersionX = mstrVersionNo

    Exit Function

IncVersionXErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "IncVersionX"
    On Error GoTo 0
    Err.Raise vbObjectError + errIncVersionXFailed, _
          gstrSource, _
```

```
        LoadResString(errIncVersionXFailed)

End Function


Private Function GetY(strVersion As String) As Long
    ' The version number for a step is stored in the x.y
    ' format where x is the parent component and y is the
    ' child component of the step. Given an argument of type
    ' x.y, it returns y

    ' Truncate the fractional part to get the parent component
    ' of the version number (x.y)
    GetY = Val(Mid(strVersion, InStr(strVersion, gstrVerSeparator) + 1))

End Function
Private Function GetX(strVersion As String) As Long
    ' The version number for a step is stored in the x.y
    ' format where x is the parent component and y is the
    ' child component of the step. Given an argument of type
    ' x.y, it returns x

    ' Truncate the fractional part to get the parent component
    ' of the version number (x.y)
    GetX = Val(Left(strVersion, InStr(strVersion, gstrVerSeparator) - 1))

End Function

Public Function Clone(Optional cCloneStep As cStep) As cStep

    ' Creates a copy of a given step

    Dim lngIndex As Long
    Dim cItRec As cIterator
    Dim cItClone As cIterator

    On Error GoTo CloneErr

    If cCloneStep Is Nothing Then
        Set cCloneStep = New cStep
    End If

    ' Copy all the step properties to the newly created step
    ' Initialize the global flag first since subsequent
    ' validations might depend on it
    cCloneStep.GlobalFlag = mblnGlobalFlag
'   cCloneStep.GlobalRunMethod = mintGlobalRunMethod

    cCloneStep.StepType = mintStepType
    cCloneStep.StepId = mlngStepId
    cCloneStep.VersionNo = mstrVersionNo
    cCloneStep.StepLabel = mstrStepLabel
    cCloneStep.StepTextFile = mstrStepTextFile
    cCloneStep.StepText = mstrStepText
    cCloneStep.StartDir = mstrStartDir
    cCloneStep.WorkspaceId = mlngWorkspaceId
    cCloneStep.ParentStepId = mlngParentStepId
    cCloneStep.ParentVersionNo = mstrParentVersionNo
    cCloneStep.StepLevel = mintStepLevel
    cCloneStep.SequenceNo = mintSequenceNo
    cCloneStep.EnabledFlag = mblnEnabledFlag
    cCloneStep.DegreeParallelism = mstrDegreeParallelism
    cCloneStep.ExecutionMechanism = mintExecutionMechanism
    cCloneStep.FailureDetails = mstrFailureDetails
    cCloneStep.ContinuationCriteria = mintContinuationCriteria
    cCloneStep.ArchivedFlag = mblnArchivedFlag
    cCloneStep.OutputFile = mstrOutputFile
'   cCloneStep.LogFile = mstrLogFile
    cCloneStep.ErrorFile = mstrErrorFile
    cCloneStep.IteratorName = mstrIteratorName

    cCloneStep.IndOperation = mintOperation
    cCloneStep.Position = mlngPosition

    Set cCloneStep.NodeDB = mdbsDatabase

    ' Clone all the iterators for the step
    For lngIndex = 0 To mcIterators.Count - 1
        Set cItRec = mcIterators(lngIndex)
        Set cItClone = cItRec.Clone
        cCloneStep.LoadIterator cItClone
    Next lngIndex

    ' And set the return value to the newly created step
    Set Clone = cCloneStep

    Exit Function

CloneErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Clone"
    On Error GoTo 0
    Err.Raise vbObjectError + errCloneFailed, _
          mstrSource, LoadResString(errCloneFailed)

End Function
'End Sub
'
Public Property Let OutputFile(ByVal vdata As String)

    mstrOutputFile = vdata

End Property

Public Property Get OutputFile() As String

    OutputFile = mstrOutputFile

End Property

'Public Property Let LogFile(ByVal vdata As String)
'
'    mstrLogFile = vdata
'
'End Property
'
'Public Property Get LogFile() As String
'
'    LogFile = mstrLogFile
'
'End Property

Public Property Let ErrorFile(ByVal vdata As String)

    mstrErrorFile = vdata

End Property
Public Property Let IteratorName(ByVal vdata As String)

    mstrIteratorName = vdata

End Property

Public Property Get ErrorFile() As String

    ErrorFile = mstrErrorFile

End Property
Public Property Get IteratorName() As String

    IteratorName = mstrIteratorName

End Property

Public Property Set NodeDB(vdata As Database)

    Set mdbsDatabase = vdata
    Set mcIterators.NodeDB = vdata

End Property
Public Property Get NodeDB() As Database

    Set NodeDB = mdbsDatabase

End Property
```

```
Private Function IsStringEmpty(strToCheck As String) As Boolean

    IsStringEmpty = (strToCheck = gstrEmptyString)

End Function

Public Property Let EnabledFlag(ByVal vdata As Boolean)

    ' The enabled flag must be False for all global steps.
    ' This check must be made by the global step class. Only
    ' generic step validations will be carried out by this
    ' class
    mblnEnabledFlag = vdata

End Property

Public Property Let GlobalFlag(ByVal vdata As Boolean)

    mblnGlobalFlag = vdata

End Property
Public Property Get EnabledFlag() As Boolean

    EnabledFlag = mblnEnabledFlag

End Property

Public Property Let ArchivedFlag(ByVal vdata As Boolean)

    mblnArchivedFlag = vdata

End Property

Public Property Get ArchivedFlag() As Boolean

    ArchivedFlag = mblnArchivedFlag

End Property

Public Property Get GlobalFlag() As Boolean

    GlobalFlag = mblnGlobalFlag

End Property

Public Sub Add()
    ' Inserts a step record into the database - it initializes
    ' the necessary properties for the step and calls InsertStepRec
    ' to do the database work

    On Error GoTo AddErr

    ' A new record would have the deleted_flag turned off!
    mblnArchivedFlag = False

    Call InsertStepRec

    ' If a new version of a step has been created, reset the old version info, since
    ' it's already been saved to the db
    If IsNewVersion() Then
        mbIsNewVersion = False
        msOldVersion = gstrEmptyString
    End If

    Exit Sub

AddErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errAddStepFailed, _
          mstrModuleName & "Add", LoadResString(errAddStepFailed)
End Sub
Private Sub InsertStepRec()
    ' Inserts a step record into the database
    ' It first generates the insert statement using the different
    ' step properties and then executes it

    Dim strInsert As String
    Dim qy As DAO.QueryDef

    On Error GoTo InsertStepRecErr

    ' First check if the database object is valid
    Call CheckDB

    ' Check if the step record is valid
    Call Validate

    If IsNewVersion() Then
        Call UpdOldVersionsArchFlg
    End If

    ' Create a temporary querydef object
    strInsert = "insert into att_steps " & _
        "( workspace_id, step_id, version_no, " & _
        " step_label, step_file_name, step_text, start_directory, " & _
        " parent_step_id, parent_version_no, sequence_no, " & _
        " enabled_flag, step_level, " & _
        " degree_parallelism, execution_mechanism, " & _
        " failure_details, " & _
        " continuation_criteria, global_flag, " & _
        " archived_flag, " & _
        " output_file_name, error_file_name, " & _
        " iterator_name ) values ( "

    ' log_file_name,

#If USE_JET Then

    strInsert = strInsert & " [w_id], [s_id], [ver_no], " & _
        " [s_label], [s_file_name], [s_text], [s_start_dir], " & _
        " [p_step_id], [p_version_no], [seq_no], " & _
        " [enabled], [s_level], [deg_parallelism], " & _
        " [exec_mechanism], [fail_dtls], " & _
        " [cont_criteria], [global], [archived], " & _
        " [output_file], [error_file], " & _
        " [it_name] ) "

    ' [log_file],

    Set qy = mdbsDatabase.CreateQueryDef(gstrEmptyString, strInsert)

    ' Call a procedure to execute the Querydef object
    Call AssignParameters(qy)

    qy.Execute dbFailOnError
    qy.Close
#Else

        strInsert = strInsert & Str(mlngWorkspaceId) & ", " & Str(mlngStepId) & _
        ", " & mFieldValue.MakeStringFieldValid(mstrVersionNo)

    ' For fields that may be null, call a function to determine
    ' the string to be appended to the insert statement
    strInsert = strInsert & ", " &
mFieldValue.MakeStringFieldValid(mstrStepLabel)
    strInsert = strInsert & ", " &
mFieldValue.MakeStringFieldValid(mstrStepTextFile)
    strInsert = strInsert & ", " & mFieldValue.MakeStringFieldValid(mstrStepText)
    strInsert = strInsert & ", " & mFieldValue.MakeStringFieldValid(mstrStartDir)

    strInsert = strInsert & ", " & Str(mlngParentStepId) & _
        ", " & mFieldValue.MakeStringFieldValid(mstrParentVersionNo) & _
        ", " & Str(mintSequenceNo) & _
        ", " & Str(mblnEnabledFlag) & ", " & Str(mintStepLevel)

    strInsert = strInsert & ", " &
mFieldValue.MakeStringFieldValid(mstrDegreeParallelism)
    strInsert = strInsert & ", " & Str(mintExecutionMechanism)

    strInsert = strInsert & ", " &
mFieldValue.MakeStringFieldValid(mstrFailureDetails) & _
        ", " & Str(mintContinuationCriteria) & _
        ", " & Str(mblnGlobalFlag) & _
        ", " & Str(mblnArchivedFlag)
```

```
    strInsert = strInsert & ", " &
mFieldValue.MakeStringFieldValid(mstrOutputFile)
'    strInsert = strInsert & ", " & mFieldValue.MakeStringFieldValid(mstrLogFile)
    strInsert = strInsert & ", " & mFieldValue.MakeStringFieldValid(mstrErrorFile)
    strInsert = strInsert & ", " &
mFieldValue.MakeStringFieldValid(mstrIteratorName)

    strInsert = strInsert & " ) "

    BugMessage strInsert
    mdbsDatabase.Execute strInsert, dbFailOnError

#End If

    Exit Sub

InsertStepRecErr:
    mstrSource = mstrModuleName & "InsertStepRec"
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errInsertStepFailed, _
          mstrSource, LoadResString(errInsertStepFailed)
End Sub
Private Sub UpdOldVersionsArchFlg()
    ' Updates the archived flag on all old version for the step to True

    Dim sUpdate As String
    Dim qy As DAO.QueryDef

    On Error GoTo UpdOldVersionsArchFlgErr
    mstrSource = mstrModuleName & "UpdOldVersionsArchFlg"

#If USE_JET Then

    sUpdate = "update att_steps " & _
       " set archived_flag = True "

    ' Append the Where clause
    sUpdate = sUpdate & " where step_id = [s_id] " & _
       " and version_no <> [ver_no]"

    Set qy = mdbsDatabase.CreateQueryDef(gstrEmptyString,sUpdate)

    ' Call a procedure to execute the Querydef object
    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    If qy.RecordsAffected = 0 Then
       On Error GoTo 0
       Err.Raise vbObjectError + errModifyStepFailed, _
            mstrSource, LoadResString(errModifyStepFailed)
    End If

    qy.Close

#Else

    sUpdate = "update att_steps " & _
       " set archived_flag = True "

    sUpdate = sUpdate & " where step_id = " & Str(mlngStepId) & _
       " and version_no <> " &
mFieldValue.MakeStringFieldValid(mstrVersionNo)

    BugMessage sUpdate
    mdbsDatabase.Execute sUpdate, dbFailOnError
#End If

    Exit Sub

UpdOldVersionsArchFlgErr:
    mstrSource = mstrModuleName & "UpdOldVersionsArchFlg"
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errModifyStepFailed, _
          mstrSource, LoadResString(errModifyStepFailed)
End Sub
Public Sub InsertIterator(cItRecord As cIterator)
    ' Inserts the iterator record into the database

    Call cItRecord.Add(mlngStepId, mstrVersionNo)

End Sub
Public Sub UpdateIterator(cItRecord As cIterator)
    ' Updates the iterator record in the database

    Call cItRecord.Update(mlngStepId, mstrVersionNo)

End Sub
Public Sub UpdateIteratorVersion()
    ' Updates the iterator record in the database

    Dim lngIndex As Long
    Dim cTempIt As cIterator

    On Error GoTo UpdateIteratorVersionErr

    For lngIndex = 0 To mcIterators.Count - 1
       ' Increase the array dimension and add the constraint
       ' to it
       Set cTempIt = mcIterators(lngIndex)

       If cTempIt.IndOperation <> DeleteOp Then
          ' Set the operation to indicate an insert
          cTempIt.IndOperation = InsertOp
       End If

    Next lngIndex

    Exit Sub

UpdateIteratorVersionErr:
    mstrSource = mstrModuleName & "UpdateIteratorVersion"
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errUpdateFailed, _
          mstrSource, LoadResString(errUpdateFailed)

End Sub
Public Sub AddIterator(cItRecord As cIterator)
    ' Adds the iterator record to the collection of iterators
    ' for the step

    Call mcIterators.Add(cItRecord)

End Sub
Public Sub LoadIterator(cItRecord As cIterator)
    ' Adds the iterator record to the collection of iterators
    ' for the step

    Call mcIterators.Load(cItRecord)

End Sub
Public Sub UnloadIterators()
    ' Unloads all iterator records for the step

    Dim lngIndex As Long

    For lngIndex = mcIterators.Count - 1 To 0 Step -1
       ' Calls the collection method to unload the node
       ' from the array
       mcIterators.Unload lngIndex
    Next lngIndex

End Sub
Public Sub ModifyIterator(cItRecord As cIterator)
    ' Modifies the iterator record in the collection

    Call mcIterators.Modify(cItRecord)

End Sub
Public Sub DeleteIterator(cItRecord As cIterator)
    ' Deletes the iterator record from the database

    Call cItRecord.Delete(mlngStepId, mstrVersionNo)
```

```
End Sub
Public Sub RemoveIterator(cItRecord As cIterator)
    ' Marks the iterator record in the collection to
    ' indicate a delete

    Call mcIterators.Delete(cItRecord.Position)

End Sub


Private Sub AssignParameters(qyExec As DAO.QueryDef)
    ' Assigns values to the parameters in the querydef object
    ' The parameter names are cryptic to make them different
    ' from the actual field names. When the parameter names
    ' are the same as the field names, parameters in the
    ' where clause do not get created.

    Dim prmParam As DAO.Parameter

    On Error GoTo AssignParametersErr
    mstrSource = mstrModuleName & "AssignParameters"

    For Each prmParam In qyExec.Parameters
        Select Case prmParam.Name
            Case "[w_id]"
                prmParam.Value = mlngWorkspaceId
            Case "[s_id]"
                prmParam.Value = mlngStepId
            Case "[ver_no]"
                prmParam.Value = mstrVersionNo
            Case "[s_label]"
                prmParam.Value = mstrStepLabel
            Case "[s_file_name]"
                prmParam.Value = mstrStepTextFile
            Case "[s_text]"
                prmParam.Value = mstrStepText
            Case "[s_start_dir]"
                prmParam.Value = mstrStartDir
            Case "[p_step_id]"
                prmParam.Value = mlngParentStepId
            Case "[p_version_no]"
                prmParam.Value = mstrParentVersionNo
            Case "[seq_no]"
                prmParam.Value = mintSequenceNo
            Case "[enabled]"
                prmParam.Value = mblnEnabledFlag
            Case "[s_level]"
                prmParam.Value = mintStepLevel
            Case "[deg_parallelism]"
                prmParam.Value = mstrDegreeParallelism
            Case "[exec_mechanism]"
                prmParam.Value = mintExecutionMechanism
            Case "[fail_dtls]"
                prmParam.Value = mstrFailureDetails
            Case "[cont_criteria]"
                prmParam.Value = mintContinuationCriteria
            Case "[global]"
                prmParam.Value = mblnGlobalFlag
            Case "[archived]"
                prmParam.Value = mblnArchivedFlag
            Case "[output_file]"
                prmParam.Value = mstrOutputFile
'           Case "[log_file]"
'               prmParam.Value = mstrLogFile
            Case "[error_file]"
                prmParam.Value = mstrErrorFile
            Case "[it_name]"
                prmParam.Value = mstrIteratorName
            Case Else
                ' Write the parameter name that is faulty
                WriteError errInvalidParameter, mstrSource, _
                    prmParam.Name
                On Error GoTo 0
                Err.Raise errInvalidParameter, mstrSource, _
                    LoadResString(errInvalidParameter)
        End Select
    Next prmParam

    If qyExec.Parameters("s_id") = 0 Or
StringEmpty(qyExec.Parameters("ver_no"))Then
        WriteError errInvalidParameter, mstrSource
        On Error GoTo 0
        Err.Raise errInvalidParameter, mstrSource,
LoadResString(errInvalidParameter)
    End If

    Exit Sub

AssignParametersErr:

    mstrSource = mstrModuleName & "AssignParameters"
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errAssignParametersFailed, _
        mstrSource, LoadResString(errAssignParametersFailed)

End Sub

Public Sub Modify()

    Dim strUpdate As String
    Dim qy As QueryDef

    On Error GoTo ModifyErr
    mstrSource = mstrModuleName & "Modify"

    ' Check if the database object is valid
    Call CheckDB

    ' Check if the step record is valid
    Call Validate

    ' The step_id and version_no will never be updated -
    ' whenever a step is modified a copy of the old step will
    ' be created with an incremented version_no

#If USE_JET Then

    strUpdate = "update att_steps " & _
        " set step_label = [s_label] " & _
        " , step_file_name = [s_file_name] " & _
        " , step_text = [s_text] " & _
        " , start_directory = [s_start_dir] " & _
        " , workspace_id = [w_id] " & _
        " , parent_step_id = [p_step_id] " & _
        " , parent_version_no = [p_version_no] " & _
        " , sequence_no = [seq_no] " & _
        " , step_level = [s_level] " & _
        " , enabled_flag = [enabled] " & _
        " , degree_parallelism = [deg_parallelism] " & _
        " , execution_mechanism = [exec_mechanism] " & _
        " , failure_details = [fail_dtls] " & _
        " , continuation_criteria = [cont_criteria] " & _
        " , global_flag = [global] " & _
        " , archived_flag = [archived] " & _
        " , output_file_name = [output_file] " & _
        " , error_file_name = [error_file] " & _
        " , iterator_name = [it_name] "

        ' " , log_file_name = [log_file] " & _

    ' Append the Where clause
    strUpdate = strUpdate & " where step_id = [s_id] " & _
        " and version_no = [ver_no]"

    Set qy = mdbsDatabase.CreateQueryDef(gstrEmptyString, strUpdate)

    ' Call a procedure to execute the Querydef object
    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    If qy.RecordsAffected = 0 Then
        On Error GoTo 0
        Err.Raise vbObjectError + errModifyStepFailed, _
            mstrSource, LoadResString(errModifyStepFailed)
    End If
```

```
    qy.Close

#Else

    strUpdate = "update att_steps " & _
        " set step_label = "

    ' For fields that may be null, call a function to determine
    ' the string to be appended to the update statement
    strUpdate = strUpdate & mFieldValue.MakeStringFieldValid(mstrStepLabel)

    strUpdate = strUpdate & " , step_file_name = " &
mFieldValue.MakeStringFieldValid(mstrStepTextFile)
    strUpdate = strUpdate & " , step_text = " &
mFieldValue.MakeStringFieldValid(mstrStepText)
    strUpdate = strUpdate & " , start_directory = " &
mFieldValue.MakeStringFieldValid(mstrStartDir)

    strUpdate = strUpdate & " , workspace_id = " & Str(mlngWorkspaceId) & _
        " , parent_step_id = " & Str(mlngParentStepId) & _
        " , parent_version_no = " &
mFieldValue.MakeStringFieldValid(mstrParentVersionNo)& _
        " , sequence_no = " & Str(mintSequenceNo) & _
        " , step_level = " & Str(mintStepLevel) & _
        " , enabled_flag = " & Str(mblnEnabledFlag)& _
        " , degree_parallelism = " &
mFieldValue.MakeStringFieldValid(mstrDegreeParallelism)& _
        " , execution_mechanism = " & Str(mintExecutionMechanism)& _
        " , failure_details = " &
mFieldValue.MakeStringFieldValid(mstrFailureDetails)& _
        " , continuation_criteria = " & Str(mintContinuationCriteria)& _
        " , global_flag = " & Str(mblnGlobalFlag)& _
        " , archived_flag = " & Str(mblnArchivedFlag)& _
        " , output_file_name = " &
mFieldValue.MakeStringFieldValid(mstrOutputFile)& _
        " , error_file_name = " & mFieldValue.MakeStringFieldValid(mstrErrorFile)
& _
        " , iterator_name = " &
mFieldValue.MakeStringFieldValid(mstrIteratorName)

'       " , log_file_name = " & mFieldValue.MakeStringFieldValid(mstrLogFile)&
_

    strUpdate = strUpdate & " where step_id = " & Str(mlngStepId) & _
        " and version_no = " & mFieldValue.MakeStringFieldValid(mstrVersionNo)

    BugMessage strUpdate
    mdbsDatabase.Execute strUpdate, dbFailOnError
#End If

    Exit Sub

ModifyErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Modify"
    On Error GoTo 0
    Err.Raise vbObjectError + errModifyStepFailed, _
        mstrSource, LoadResString(errModifyStepFailed)
End Sub
Private Sub CheckDB()
    ' Check if the database object has been initialized

    If mdbsDatabase Is Nothing Then
        ShowError errInvalidDB
        On Error GoTo 0
        Err.Raise vbObjectError + errInvalidDB, _
            mstrModuleName, LoadResString(errInvalidDB)
    End If

End Sub

Public Sub Delete()

    Dim strDelete As String
    Dim qy As DAO.QueryDef

    On Error GoTo DeleteErr

    Call CheckDB

    strDelete = "delete from att_steps " & _
        " where step_id = [s_id] " & _
        " and version_no = [ver_no] "
'   mdbsDatabase.Execute strDelete, dbFailOnError
    Set qy = mdbsDatabase.CreateQueryDef(gstrEmptyString, strDelete)

    Call AssignParameters(qy)
    qy.Execute dbFailOnError

    qy.Close

    Exit Sub

DeleteErr:
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteStepFailed, _
        mstrModuleName & "Delete", LoadResString(errDeleteStepFailed)
End Sub
Public Property Get DegreeParallelism() As String

    DegreeParallelism = mstrDegreeParallelism

End Property
Public Property Get Position() As Long

    Position = mlngPosition

End Property

Public Property Let DegreeParallelism(ByVal vdata As String)

    ' The degree of parallelism must be zero for all global steps
    ' This check must be made by the global step class. Only
    ' generic step validations will be carried out by this
    ' class
    mstrDegreeParallelism = vdata

End Property


Public Property Let ExecutionMechanism(ByVal vdata As ExecutionMethod)

    BugAssert vdata = gintExecuteODBC Or vdata = gintExecuteShell Or vdata =
gintNoOption, _
        "Execution mechanism invalid"
    mintExecutionMechanism = vdata

End Property

Public Property Let FailureDetails(ByVal vdata As String)

    mstrFailureDetails = vdata

End Property
Public Property Let SequenceNo(ByVal vdata As Integer)
    mintSequenceNo = vdata
End Property

Public Property Let Position(ByVal vdata As Long)
    mlngPosition = vdata
End Property

Public Property Let ParentStepId(ByVal vdata As Long)
    mlngParentStepId = vdata
End Property

Public Property Get SequenceNo() As Integer

    SequenceNo = mintSequenceNo

End Property

Public Property Get StepLevel() As Integer
    StepLevel = mintStepLevel
```

```
End Property

Public Property Get ParentVersionNo() As String
   ParentVersionNo = mstrParentVersionNo
End Property

Public Property Let ParentVersionNo(ByVal vdata As String)
   mstrParentVersionNo = vdata
End Property

Public Property Get ParentStepId() As Long
   ParentStepId = mlngParentStepId
End Property

Public Property Let WorkspaceId(ByVal vdata As Long)
   mlngWorkspaceId = vdata
End Property

Public Property Let VersionNo(ByVal vdata As String)
   ' The version number of a step is stored in the x.y format where
   ' x represents a change to the step as a result of modifications
   ' to any of the step properties
   ' y represents a change to the step as a result of modifications
   ' to the sub-steps associated with it. Hence the y-component
   ' of the version will be incremented when a sub-step is added,
   ' modified or deleted
   ' x will be referred to throughout this code as the parent
   ' component of the version and y will be referred to as the
   ' child component of the version
   ' The version information for a step is maintained by the
   ' calling function

   mstrVersionNo = vdata

End Property

Public Property Get StepType() As gintStepType

   On Error GoTo StepTypeErr

   If mintStepType = 0 Then
      ' The step type variable has not been initialized -
      If mblnGlobalFlag Then
         mintStepType = gintGlobalStep
      ElseIf IsStringEmpty(mstrStepText) And _
            IsStringEmpty(mstrStepTextFile) Then
         mintStepType = gintManagerStep
      Else
         mintStepType = gintWorkerStep
      End If
   End If

   StepType = mintStepType

   Exit Property

StepTypeErr:
   LogErrors Errors
   mstrSource = mstrModuleName & "StepType"
   On Error GoTo 0
   Err.Raise vbObjectError + errGetStepTypeFailed, _
         mstrSource, _
         LoadResString(errGetStepTypeFailed)

End Property

Public Property Let StepType(vdata As gintStepType)

   On Error GoTo StepTypeErr

   Select Case vdata
      Case gintGlobalStep, gintManagerStep, gintWorkerStep
         mintStepType = vdata

      Case Else
         On Error GoTo 0
         Err.Raise vbObjectError + errStepTypeInvalid, _
               mstrModuleName & "StepType",
LoadResString(errStepTypeInvalid)
   End Select
   Exit Property

StepTypeErr:
   LogErrors Errors
   mstrSource = mstrModuleName & "StepType"
   On Error GoTo 0
   Err.Raise vbObjectError + errLetStepTypeFailed, _
         mstrSource, _
         LoadResString(errLetStepTypeFailed)

End Property

Public Property Get WorkspaceId() As Long
   WorkspaceId = mlngWorkspaceId
End Property

Public Property Get ContinuationCriteria() As ContinuationCriteria

   ContinuationCriteria = mintContinuationCriteria

End Property

Public Property Let ContinuationCriteria(ByVal vdata As ContinuationCriteria)

   ' The Continuation criteria must be null for all global steps
   ' and non-null for all manager and worker steps
   ' These checks will have to be made by the corresponding
   ' classes - only generic step validations will be made
   ' by this class
   BugAssert vdata = gintOnFailureAbortSiblingsOr vdata =
gintOnFailureCompleteSiblings_
         Or vdata = gintOnFailureSkipSiblingsOr vdata = gintOnFailureAbort_
         Or vdata = gintOnFailureContinueOr vdata = gintOnFailureAsk_
         Or vdata = gintNoOption, _
         "Invalid continuation criteria"
   mintContinuationCriteria = vdata

End Property
Public Property Get ExecutionMechanism() As ExecutionMethod

   ExecutionMechanism = mintExecutionMechanism

End Property


Public Property Get FailureDetails() As String

   FailureDetails = mstrFailureDetails

End Property

Public Property Let StepText(ByVal vdata As String)
   ' Has to be null for manager steps
   ' The check will have to be made by the user interface or
   ' by the manager step class
   mstrStepText = vdata
End Property
Public Property Let StepLevel(ByVal vdata As Integer)

   ' The step level must be zero for all global steps
   ' This check must be made in the global step class
   mintStepLevel = vdata

End Property
Public Property Get StepText() As String
   StepText = mstrStepText
End Property

Public Property Let StepTextFile(ByVal vdata As String)
   ' Has to be null for manager steps
   ' The check will have to be made by the user interface and
   ' by the manager step class
   mstrStepTextFile = vdata
End Property
```

```
Public Property Get StepTextFile() As String
    StepTextFile = mstrStepTextFile
End Property

Public Property Let StepLabel(ByVal vdata As String)
    ' Cannot be null for manager steps
    ' But this check cannot be made here since we do not know
    ' at this point if the step being created is a manager
    ' or a worker step
    ' The check will have to be made by the user interface and
    ' by the manager step class
    mstrStepLabel = vdata
End Property

Public Property Get StepLabel() As String
    StepLabel = mstrStepLabel
End Property

Public Property Let StartDir(ByVal vdata As String)
    mstrStartDir = vdata
End Property

Public Property Get StartDir() As String
    StartDir = mstrStartDir
End Property

Public Property Get VersionNo() As String
    ' The version number of a step is stored in the x.y format where
    ' x represents a change to the step as a result of modifications
    ' to any of the step properties
    ' y represents a change to the step as a result of modifications
    ' to the sub-steps associated with it. Hence the y-component
    ' of the version will be incremented when a sub-step is added,
    ' modified or deleted
    ' x will be referred to throughout this code as the parent
    ' component of the version and y will be referred to as the
    ' child component of the version
    ' The version information for a step is maintained by the
    ' calling function

    VersionNo = mstrVersionNo

End Property

Public Property Get StepId() As Long

    StepId = mlngStepId

End Property
Public Property Get NextStepId() As Long

    Dim lngNextId As Long

    On Error GoTo NextStepIdErr

    ' First check if the database object is valid
    Call CheckDB

    ' Retrieve the next identifier using the sequence class
    Set mStepSeq = New cSequence
    Set mStepSeq.IdDatabase = mdbsDatabase
    mStepSeq.IdentifierColumn= "step_id"
    lngNextId = mStepSeq.Identifier
    Set mStepSeq = Nothing

    NextStepId = lngNextId
    Exit Property

NextStepIdErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "NextStepId"
    On Error GoTo 0
    Err.Raise vbObjectError + errStepIdGetFailed, _
        mstrSource, LoadResString(errStepIdGetFailed)

End Property
Public Property Let StepId(ByVal vdata As Long)
    mlngStepId = vdata

End Property

Private Sub Class_Initialize()

    ' Initialize the operation indicator variable to Query
    ' It will be modified later by the collection class when
    ' inserts, updates or deletes are performed
    mintOperation = QueryOp
    mbIsNewVersion = False
    msOldVersion = gstrEmptyString

    Set mFieldValue = New cStringSM
    Set mcIterators = New cNodeCollections

End Sub

Private Sub Class_Terminate()

    Set mFieldValue = Nothing
    Set mcIterators = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cStepTree"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cStepTree.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Implements step navigation functions such as determining
'              the child of a step and so on.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "cStepTree."
Private mstrSource As String

Public StepRecords As cArrSteps
Public Property Get HasChild(Optional ByVal StepKey As String, _
      Optional ByVal StepId As Long = 0) As Boolean

    Dim lTemp As Long

    HasChild = False
    StepId = GetStepId(StepKey, StepId)

    For lTemp = 0 To StepRecords.StepCount - 1
       If StepRecords(lTemp).StepType <> gintGlobalStep And
StepRecords(lTemp).ParentStepId = StepId Then
          HasChild = True
          Exit For
       End If
    Next lTemp

End Property
Public Property Get ChildStep(Optional ByVal StepKey As String, _
      Optional ByVal StepId As Long = 0) As cStep

    Dim lTemp As Long

    Set ChildStep = Nothing
    StepId = GetStepId(StepKey, StepId)

    For lTemp = 0 To StepRecords.StepCount - 1
```

```
      If StepRecords(lTemp).StepType <> gintGlobalStep And
StepRecords(lTemp).ParentStepId = StepId And _
            StepRecords(lTemp).SequenceNo = gintMinSequenceNo Then
         Set ChildStep = StepRecords(lTemp)
         Exit For
      End If
   Next lTemp

End Property
Public Property Get NextStep(Optional ByVal StepKey As String, _
      Optional ByVal StepId As Long = 0) As cStep

   Dim lTemp As Long
   Dim cChildStep As cStep

   Set NextStep = Nothing
   StepId = GetStepId(StepKey, StepId)
   Set cChildStep = StepRecords.QueryStep(StepId)

   For lTemp = 0 To StepRecords.StepCount - 1
      If StepRecords(lTemp).StepType <> gintGlobalStep And _
            StepRecords(lTemp).ParentStepId = cChildStep.ParentStepId And _
            StepRecords(lTemp).SequenceNo = cChildStep.SequenceNo + 1 Then
         Set NextStep = StepRecords(lTemp)
         Exit For
      End If
   Next lTemp

End Property
Private Function GetStepId(Optional ByVal StepKey As String, _
      Optional ByVal StepId As Long = 0) As Long
   If StepId = 0 Then
      If StringEmpty(StepKey) Then
         Err.Raise vbObjectError + errMandatoryParameterMissing, _
               mstrModuleName & "GetStepId",
LoadResString(errMandatoryParameterMissing)
      Else
         GetStepId = IIf(IsLabel(StepKey), 0, MakeIdentifierValid(StepKey))
      End If
   Else
      GetStepId = StepId
   End If
End Function
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cStringSM"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cStringSM.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module contains common procedures that can be used
'              to manipulate strings
'              It is called StringSM, since String is a Visual Basic keyword
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cStringSM."

Private mstrText As String

Private Const mstrNullValue = "null"
Private Const mstrSQ = "'"
Private Const mstrEnvVarSeparator = "%"
Public Function InsertEnvVariables( _
      Optional ByVal strComString As String) As String
   ' This function replaces all environment variables in
   ' the passed in string with their values - they are
   ' enclosed by "%"

   Dim intPos As Integer
   Dim intEndPos As Integer
   Dim strEnvVariable As String
   Dim strValue As String
   Dim strCommand As String

   On Error GoTo InsertEnvVariablesErr
   mstrSource = mstrModuleName & "InsertEnvVariables"

   ' Initialize the return value of the function to the
   ' passed in command
   If IsStringEmpty(strComString)Then
      strCommand = mstrText
   Else
      strCommand = strComString
   End If

   intPos = InStr(strCommand, mstrEnvVarSeparator)
   Do While intPos <> 0
      ' Extract the environment variable from the passed
      ' in string
      intEndPos = InStr(intPos + 1, strCommand, mstrEnvVarSeparator)
      strEnvVariable = Mid(strCommand, intPos + 1, intEndPos - intPos - 1)

      ' Get the value of the variable and call a function
      ' to replace the variable with it's value
      strValue = Environ$(strEnvVariable)
      strCommand = ReplaceSubString(strCommand, _
            mstrEnvVarSeparator & strEnvVariable & mstrEnvVarSeparator, _
            strValue)

      intPos = InStr(strCommand, mstrEnvVarSeparator)
   Loop

   InsertEnvVariables = strCommand
   Exit Function

InsertEnvVariablesErr:
   ' Log the error code raised by Visual Basic
   Call LogErrors(Errors)
   ' Return an empty string
   InsertEnvVariables = gstrEmptyString

End Function
Public Function MakeStringFieldValid( _
      Optional strField As String = gstrEmptyString) As String
   ' Returns a string that can be appended to any insert
   ' or modify (sql) statement
   ' If an argument is not passed to this function, the
   ' default text property is used

   Dim strTemp As String

   On Error GoTo MakeStringFieldValidErr

   If IsStringEmpty(strField)Then
      strTemp = mstrText
   Else
      strTemp = strField
   End If

   ' It checks whether the text is empty
   ' If so, it returns the string, "null"
   If IsStringEmpty(strTemp)Then
      MakeStringFieldValid = mstrNullValue
   Else
      ' Single-quotes have to be replaced by two single-quotes,
      ' since a single-quote is the identifier delimiter
      ' character - call a procedure to do the replace
      strTemp = ReplaceSubString(strTemp, mstrSQ, mstrSQ & mstrSQ)

      ' Replace pipe characters with the corresponding chr function
      strTemp = ReplaceSubString(strTemp, "|", "' & Chr(124) & "')

      ' Enclose the string in single quotes
```

```
        MakeStringFieldValid = mstrSQ & strTemp & mstrSQ

    End If

    Exit Function

MakeStringFieldValidErr:
    mstrSource = mstrModuleName & "MakeStringFieldValid"
    LogErrors Errors
    On Error GoTo 0
    Err.Raise vbObjectError + errMakeFieldValidFailed, _
          mstrSource, LoadResString(errMakeFieldValidFailed)

End Function
Public Function MakeDateFieldValid( _
        Optional dtmField As Date = gdtmEmpty) As String
    ' Returns a string that can be appended to any insert
    ' or modify (sql) statement

    ' Enclose the date in single quotes
    MakeDateFieldValid = mstrSQ & dtmField & mstrSQ

End Function

Private Function IsStringEmpty(strToCheck As String) As Boolean

    If strToCheck = gstrEmptyString Then
        IsStringEmpty = True
    Else
        IsStringEmpty = False
    End If

End Function
Public Function ReplaceSubString(ByVal MainString As String, _
        ByVal ReplaceString As String, _
        ByVal ReplaceWith As String) As String

    ' Replaces all occurrences of ReplaceString in MainString with ReplaceWith

    Dim intPos As Integer
    Dim strTemp As String

    On Error GoTo ReplaceSubStringErr

    strTemp = MainString

    intPos = InStr(strTemp, ReplaceString)
    Do While intPos <> 0
        strTemp = Left(strTemp, intPos - 1) & ReplaceWith & _
              Mid(strTemp, intPos + Len(ReplaceString))
        intPos = InStr(intPos + Len(ReplaceString) + 1, strTemp, ReplaceString)
    Loop
    ReplaceSubString = strTemp

    Exit Function

ReplaceSubStringErr:
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "ReplaceSubString"
    On Error GoTo 0
    Err.Raise vbObjectError + errParseStringFailed, _
        mstrSource, _
        LoadResString(errParseStringFailed)

End Function

Public Property Get Text() As String
Attribute Text.VB_UserMemId = 0
    Text = mstrText
End Property

Public Property Let Text(ByVal vdata As String)
    mstrText = vdata
End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cSubStep"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cSubStep.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module encapsulates the properties of sub-steps
'              that are used during the execution of a workspace.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "cSubStep"

Private mlngStepId As Long
Private mintRunning As Integer ' Number of running tasks
Private mintComplete As Integer ' Number of completed tasks
' The last iterator for this sub-step
Private mcLastIterator As cRunItDetails

Public Function NewIteration(cStepRec As cStep) As cIterator
    ' Calls a procedure to determine the next iterator value
    ' for the passed in step - returns the value to be used
    ' in the iteration.
    ' It updates the instance node with the new iteration
    ' for the step.

    Dim cItRec As cIterator

    On Error GoTo NewIterationErr

    ' Call a function that will populate an iterator record
    ' with the iterator values
    Set cItRec = NextIteration(cStepRec)

    ' Initialize the run node with the new iterator
    ' values
    If Not mcLastIterator Is Nothing Then
        If cItRec Is Nothing Then
            mcLastIterator.Value = gstrEmptyString
        Else
            mcLastIterator.Value = cItRec.Value

            ' And if the iterator is a list of values, then update
            ' the sequence number as well
            If mcLastIterator.IteratorType = gintValue Then
                mcLastIterator.Sequence = cItRec.SequenceNo
            End If
        End If
    End If

    Set NewIteration = cItRec
    Set cItRec = Nothing

    Exit Function

NewIterationErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errIterateFailed, mstrModuleName, _
        LoadResString(errIterateFailed)

End Function
Public Function NextIteration(cStepRec As cStep) As cIterator

    ' Retrieves the next iterator value for the passed in step -
    ' returns an iterator record with the new iterator values

    Dim cItRec As cIterator
    Dim vntIterators As Variant
```

```
    Dim lngValue As String

    On Error GoTo NextIterationErr

    vntIterators = cStepRec.Iterators

    If Not mcLastIterator Is Nothing Then
'       ' This procedure depends on the fact that the iterator type
'       ' hasn't been initialized - it may well have been, though
'       ' Try to modify the check later.
'       If mcLastIterator.IteratorType = 0 Then
'          ' The iterator details have not been initialized on the
'          ' run node for the step - call a procedure to carry out
'          ' the initialization
'          BugMessage "Initialize later happens!!!"
'          Call InitializeIt(cStepRec, RunParams, vntIterators)
'       End If
'
'       ' mcLastIterator will be set to Nothing if no iterators
'       ' have been defined for the step
'       If Not mcLastIterator Is Nothing Then

          ' The run node contains the iterator details
          ' Get the next value for the iterator
          If mcLastIterator.IteratorType = gintValue Then
             ' Find the next iterator that appears in the list of
             ' iterator values
             Set cItRec = NextInSequence(vntIterators, mcLastIterator.Sequence)
          Else
             lngValue = CLng(Trim$(mcLastIterator.Value))
             ' Determine whether the new iterator value falls in the
             ' range between From and To
             If (mcLastIterator.RangeStep > 0 And _
                   (mcLastIterator.RangeFrom <= mcLastIterator.RangeTo) And _
                   (mcLastIterator.RangeStep + lngValue) <=
mcLastIterator.RangeTo) Or _
                   (mcLastIterator.RangeStep < 0 And _
                   (mcLastIterator.RangeFrom >= mcLastIterator.RangeTo) And _
                   (mcLastIterator.RangeStep + lngValue) >=
mcLastIterator.RangeTo) Then
                Set cItRec = New cIterator
                cItRec.Value = Trim$(CStr(mcLastIterator.RangeStep + lngValue))
             Else
                Set cItRec = Nothing
             End If
          End If

'       End If
    Else
       Set cItRec = Nothing
    End If

    Set NextIteration = cItRec
    Exit Function

NextIterationErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errIterateFailed, mstrModuleName, _
       LoadResString(errIterateFailed)

End Function
Public Sub InitializeIt(cPendingStep As cStep, _
       ColParameters As cArrParameters, _
       Optional vntIterators As Variant)

    ' Initializes the LastIteration structure with the iterator details for the
    ' passed in step

    On Error GoTo InitializeItErr

    If IsMissing(vntIterators) Then
       vntIterators = cPendingStep.Iterators
    End If

    If IsArray(vntIterators) And Not IsEmpty(vntIterators) Then
       mcLastIterator.IteratorName = cPendingStep.IteratorName
       If vntIterators(LBound(vntIterators)).IteratorType = _
             gintValue Then
          mcLastIterator.IteratorType = gintValue
          ' Since the sequence numbers begin at 0
          mcLastIterator.Sequence = gintMinIteratorSequence - 1
       Else
          mcLastIterator.IteratorType = gintFrom
          Call InitializeItRange(vntIterators, cPendingStep.WorkspaceId, _
                ColParameters)
       End If
    Else
       Set mcLastIterator = Nothing
    End If

    Exit Sub

InitializeItErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errIterateFailed, mstrModuleName, _
       LoadResString(errIterateFailed)

End Sub

Private Sub InitializeItRange(vntIterators As Variant, ByVal lWorkspace As
Long, _
       ColParameters As cArrParameters)

    ' Initializes the LastIteration structure for range iterators from the
    ' passed in variant containing the iterator records

    Dim lngIndex As Long
    Dim cItRec As cIterator

    On Error GoTo InitializeItRangeErr

    If IsArray(vntIterators) And Not IsEmpty(vntIterators) Then

       ' Check if the iterator range has been completely initialized
       RangeComplete (vntIterators)

       ' Initialize the Run node with the values for the From,
       ' To and Step boundaries
       For lngIndex = LBound(vntIterators) To UBound(vntIterators)
          Set cItRec = vntIterators(lngIndex)
          Select Case cItRec.IteratorType
             Case gintFrom
                mcLastIterator.RangeFrom = SubstituteParameters(cItRec.Value,
lWorkspace, WspParameters:=ColParameters)
             Case gintTo
                mcLastIterator.RangeTo = SubstituteParameters(cItRec.Value,
lWorkspace, WspParameters:=ColParameters)
             Case gintStep
                mcLastIterator.RangeStep = SubstituteParameters(cItRec.Value,
lWorkspace, WspParameters:=ColParameters)
             Case Else
                On Error GoTo 0
                Err.Raise vbObjectError + errTypeInvalid, mstrModuleName, _
                      LoadResString(errTypeInvalid)
          End Select
       Next lngIndex

       mcLastIterator.Value = Trim$(CStr(mcLastIterator.RangeFrom-
mcLastIterator.RangeStep))
    End If

    Exit Sub

InitializeItRangeErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errIterateFailed, mstrModuleName, _
       LoadResString(errIterateFailed)

End Sub
Private Function NextInSequence(vntIterators As Variant, _
```

```
    lngOldSequence As Long) As cIterator

  Dim lngIndex As Long
  Dim cItRec As cIterator

  On Error GoTo NextInSequenceErr

  If IsArray(vntIterators) And Not IsEmpty(vntIterators) Then
    For lngIndex = LBound(vntIterators) To UBound(vntIterators)
      Set cItRec = vntIterators(lngIndex)
      If cItRec.IteratorType <> gintValue Then
        On Error GoTo 0
        Err.Raise vbObjectError + errTypeInvalid, mstrModuleName, _
            LoadResString(errTypeInvalid)
      End If
      If cItRec.SequenceNo = lngOldSequence + 1 Then
        Exit For
      End If

    Next lngIndex

    If cItRec.SequenceNo <> lngOldSequence + 1 Then
      Set cItRec = Nothing
    End If
  Else
    Set cItRec = Nothing
  End If

  Set NextInSequence = cItRec

  Exit Function

NextInSequenceErr:
  ' Log the error code raised by Visual Basic
  Call LogErrors(Errors)
  On Error GoTo 0
  Err.Raise vbObjectError + errIterateFailed, mstrModuleName, _
    LoadResString(errIterateFailed)

End Function

Public Property Get LastIterator() As cRunItDetails

  Set LastIterator = mcLastIterator

End Property
Public Property Set LastIterator(vdata As cRunItDetails)

  Set mcLastIterator = vdata

End Property

Public Property Get TasksRunning() As Integer

  TasksRunning = mintRunning

End Property

Public Property Let TasksRunning(ByVal vdata As Integer)

  mintRunning = vdata

End Property

Public Property Get TasksComplete() As Integer

  TasksComplete = mintComplete

End Property
Public Property Let TasksComplete(ByVal vdata As Integer)

  mintComplete = vdata

End Property
Public Property Get StepId() As Long

  StepId = mlngStepId

End Property
Public Property Let StepId(ByVal vdata As Long)

  mlngStepId = vdata

End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cSubSteps"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cSubSteps.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module provides a type-safe wrapper around cVector to
'              implement a collection of cSubStep objects.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mcSubSteps As cVector

Public Sub Add(ByVal objItem As cSubStep)

  mcSubSteps.Add objItem

End Sub


Public Sub Clear()

  mcSubSteps.Clear

End Sub


Public Function Count() As Long

  Count = mcSubSteps.Count

End Function


Public Function Delete(ByVal lngDelete As Long) As cSubStep

  Set Delete = mcSubSteps.Delete(lngDelete)

End Function


Public Property Get Item(ByVal Position As Long) As cSubStep
Attribute Item.VB_UserMemId = 0

  Set Item = mcSubSteps.Item(Position)

End Property

Private Sub Class_Initialize()

  Set mcSubSteps = New cVector

End Sub

Private Sub Class_Terminate()

  Set mcSubSteps = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
```

```
END
Attribute VB_Name = "cTermProcess"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
' FILE:       cTermProcess.cls
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
' PURPOSE:    This module raises an event if a completed step exists.
' Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private WithEvents moTimer As cTimerSM
Attribute moTimer.VB_VarHelpID = -1
Private bTermProcessExists As Boolean
Public Event TermProcessExists()

Public Sub ProcessTerminated()

   bTermProcessExists = True
   moTimer.Enabled = True

End Sub

Private Sub Class_Initialize()

   bTermProcessExists = False

   Set moTimer = New cTimerSM
   moTimer.Enabled = False

End Sub

Private Sub Class_Terminate()

   Set moTimer = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cTermStep"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
' FILE:       cTermStep.cls
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
' PURPOSE:    This module encapsulates the properties of steps that
'             have completed execution such as status and time of completion.
' Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Public TimeComplete As Currency
Public Index As Long
Public InstanceId As Long
Public ExecutionStatus As InstanceStatus



Private Sub moTimer_Timer()

   On Error GoTo moTimer_TimerErr

   If bTermProcessExists Then
      RaiseEvent TermProcessExists
   End If

   moTimer.Enabled = False
   bTermProcessExists = False

   Exit Sub

moTimer_TimerErr:
   LogErrors Errors

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cTermSteps"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
' FILE:       cTermSteps.cls
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
' PURPOSE:    This module provides a type-safe wrapper around cVector to
'             implement a collection of cTermStep objects. Raises an
'             event if a step that has completed execution exists.
' Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Private mcTermSteps As cVector
Private WithEvents moTimer As cTimerSM
Attribute moTimer.VB_VarHelpID = -1
Public Event TermStepExists(cStepDetails As cTermStep)

Public Sub Add(ByVal citem As cTermStep)

   Call mcTermSteps.Add(citem)
   moTimer.Enabled = True

End Sub

Public Sub Clear()

   mcTermSteps.Clear

End Sub

Public Function Delete()

   Call mcTermSteps.Delete(0)
   ' Disable the timer if there are no more pending events
   If mcTermSteps.Count = 0 Then moTimer.Enabled = False

End Function

Public Property Get Item(ByVal Position As Long) As cTermStep

   Set Item = mcTermSteps(Position)

End Property

Public Function Count() As Long

   Count = mcTermSteps.Count

End Function

Private Sub Class_Initialize()

   Set mcTermSteps = New cVector

   Set moTimer = New cTimerSM
   moTimer.Enabled = False

End Sub
```

```
Private Sub Class_Terminate()

   Set mcTermSteps = Nothing
   Set moTimer = Nothing

End Sub

Private Sub moTimer_Timer()

   On Error GoTo moTimer_TimerErr

   If mcTermSteps.Count > 0 Then
      ' Since items are appended to the end of the array
      RaiseEvent TermStepExists(mcTermSteps(0))
   Else
      moTimer.Enabled = False
   End If
   Exit Sub

moTimer_TimerErr:
   LogErrors Errors

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cTimerSM"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cTimer.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module implements a timer.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'

Option Explicit

Public Event Timer()

Private Const mnDefaultInterval As Long = 1

Private mnTimerID As Long
Private mnInterval As Long
Private mfEnabled As Boolean

Public Property Get Interval() As Long
   Interval = mnInterval
End Property
Public Property Let Interval(Value As Long)
   If mnInterval <> Value Then
      mnInterval = Value
      If mfEnabled Then
         SetInterval mnInterval, mnTimerID
      End If
   End If
End Property

Public Property Get Enabled() As Boolean
   Enabled = mfEnabled
End Property
Public Property Let Enabled(Value As Boolean)
   If mfEnabled <> Value Then
      If Value Then
         mnTimerID = StartTimer(mnInterval)
         If mnTimerID <> 0 Then
            mfEnabled = True
            'Storing Me in the global would add a reference to Me, which
            '  would prevent Me from being released, which in turn would
            '  prevent my Class_Terminate code from running. To prevent
            '  this, I store a "soft reference" - the collection holds a
            '  pointer to me without incrementing my reference count.
            gcTimerObjects.Add ObjPtr(Me), Str$(mnTimerID)
         End If
      Else
         StopTimer mnTimerID
         mfEnabled = False
         gcTimerObjects.Remove Str$(mnTimerID)
      End If
   End If
End Property

Private Sub Class_Initialize()
   If gcTimerObjects Is Nothing Then Set gcTimerObjects = New Collection
   mnInterval = mnDefaultInterval
End Sub

Private Sub Class_Terminate()
   Enabled = False
End Sub

Friend Sub Tick()
   RaiseEvent Timer
End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cVBErrorsSM"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
Attribute VB_Ext_KEY = "SavedWithClassBuilder","Yes"
Attribute VB_Ext_KEY = "Top_Level","Yes"
'  FILE:       cVBErrors.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module encapsulates the handling of Visual Basic errors.
'              This module does not do any error handling - any error handler
'              will erase the errors object!
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' The Execute class exposes a method, WriteError through which we can write to
the
' error log that is currently being used by the Execute object. Store a reference to
' Execute object locally.
Private mcExecObjRef As EXECUTEDLLLib.Execute
Public Sub WriteError(ByVal ErrorCode As errErrorConstants, _
      Optional ByVal ErrorSource As String = gstrEmptyString, _
      Optional ByVal OptArgs As String = gstrEmptyString)

   Dim sError As String

   sError = "StepMaster Error:" & ErrorCode & vbCrLf &
LoadResString(ErrorCode) & vbCrLf

   If Not StringEmpty(ErrorSource)Then
      sError = sError & "(Source: " & ErrorSource & ")" & vbCrLf
   End If
   sError = sError & OptArgs

   Call LogMessage(sError)

End Sub
Private Function InitErrorString() As String
   ' Initializes a string with all the properties of the
   ' Err object

   Dim strError As String
   Dim errCode As Long

   If Err.Number = 0 Then
      InitErrorString = gstrEmptyString
```

```
    Else
    With Err
        If Err.Number > vbObjectError And Err.Number < (vbObjectError + 65536)
Then
            errCode = .Number - vbObjectError
        Else
            errCode = .Number
        End If
        strError = "Error #: " & errCode & vbCrLf
        strError = strError & "Description: " & .Description & vbCrLf
        strError = strError & "Source: " & Err.Source & vbCrLf
    End With

    Debug.Print strError
    InitErrorString = strError
    End If

End Function
Public Sub LogVBErrors()

    Dim strErr As String

    strErr = InitErrorString

    On Error GoTo LogVBErrorsErr

    If Not StringEmpty(strErr) Then
        ' Write an error using the WriteError method of the Execute object.
        If Not mcExecObjRef Is Nothing Then
            mcExecObjRef.WriteError strErr
        Else
            WriteMessage strErr
        End If
    End If

    Err.Clear

    Exit Sub

LogVBErrorsErr:
    Call LogErrors(Errors)
    ' Since write to the error file for the step has failed, write to the project log
    Call WriteMessage(strErr)

End Sub
Public Sub DisplayErrors()

    Dim strErr As String

    strErr = InitErrorString

    If Not StringEmpty(strErr) Then
        ' Display the error message
        MsgBox strErr
    End If

    Err.Clear

End Sub
Public Sub LogMessage(strMsg As String)

    On Error GoTo LogMessageErr

    ' Write an error using the WriteError method of the Execute object.
    If Not mcExecObjRef Is Nothing Then
        mcExecObjRef.WriteError strMsg
    Else
        WriteMessage strMsg
    End If

    Exit Sub

LogMessageErr:
    Call LogErrors(Errors)
    ' Since write to the error file for the step has failed, write to the project log
    Call WriteMessage(strMsg)

End Sub
Public Property Set ErrorFile(vdata As EXECUTEDLLLib.Execute)

    Set mcExecObjRef = vdata

End Property
Private Sub Class_Terminate()

    Set mcExecObjRef = Nothing

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cVector"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cVector.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This class implements an array of objects.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cVector."

' Array counter
Private mlngCount As Long
Private mcarrItems() As Object

Public Sub Add(ByVal objItem As Object)
    ' Adds the passed in Object variable to the array

    On Error GoTo AddErr

    ReDim Preserve mcarrItems(mlngCount)

    ' Set the newly added element in the array to the
    ' passed in variable
    Set mcarrItems(mlngCount) = objItem
    mlngCount = mlngCount + 1

    Exit Sub

AddErr:
    LogErrors Errors
    gstrSource = mstrModuleName & "Add"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadInArrayFailed, _
          mstrSource, _
          LoadResString(errLoadInArrayFailed)

End Sub
Public Sub Clear()

    ' Clear the array
    ReDim mcarrItems(0)
    mlngCount = 0

End Sub

Public Function Delete(ByVal lngDelete As Long) As Object

    Dim lngIndex As Long

    On Error GoTo DeleteErr

    If lngDelete < (mlngCount - 1) Then
```

```
        ' We want to maintain the order of all items in the
        ' array - so move all remaining elements in the array
        ' up by 1
        For lngIndex = lngDelete To mlngCount - 2
            MoveDown lngIndex
        Next lngIndex

    End If

    ' Return the deleted node
    Set Delete = mcarrItems(mlngCount - 1)

    ' Delete the last Node from the array
    mlngCount = mlngCount - 1
    If mlngCount > 0 Then
        ReDim Preserve mcarrItems(0 To mlngCount - 1)
    Else
        ReDim mcarrItems(0)
    End If

    Exit Function

DeleteErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Delete"
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteArrayElementFailed, _
        mstrSource, _
        LoadResString(errDeleteArrayElementFailed)

End Function
Public Property Get Item(ByVal Position As Long) As Object
Attribute Item.VB_UserMemId = 0

    ' Returns the element at the passed in position in the array
    If Position >= 0 And Position < mlngCount Then
        Set Item = mcarrItems(Position)
    Else
        On Error GoTo 0
        Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
            LoadResString(errItemDoesNotExist)
    End If

End Property
Public Property Set Item(ByVal Position As Long, _
    ByVal Value As Object)

    ' Returns the element at the passed in position in the array
    If Position >= 0 Then
        ' If the passed in position is outside the array
        ' bounds, then resize the array
        If Position >= mlngCount Then
            ReDim Preserve mcarrItems(Position)
            mlngCount = Position + 1
        End If

        ' Set the newly added element in the array to the
        ' passed in variable
        Set mcarrItems(Position) = Value
    Else
        On Error GoTo 0
        Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
            LoadResString(errItemDoesNotExist)
    End If

End Property
Public Sub MoveUp(ByVal Position As Long)
    ' Moves the element at the passed in position up by 1

    Dim cTemp As Object

    If Position > 0 And Position < mlngCount Then
        Set cTemp = mcarrItems(Position)

        Set mcarrItems(Position) = mcarrItems(Position - 1)
        Set mcarrItems(Position - 1) = cTemp
    End If

End Sub
Public Sub MoveDown(ByVal Position As Long)
    ' Moves the element at the passed in position down by 1

    Dim cTemp As Object

    If Position >= 0 And Position < mlngCount - 1 Then
        Set cTemp = mcarrItems(Position)

        Set mcarrItems(Position) = mcarrItems(Position + 1)
        Set mcarrItems(Position + 1) = cTemp
    End If

End Sub

Public Function Count() As Long

    Count = mlngCount

End Function


Private Sub Class_Initialize()

    mlngCount = 0

End Sub


Private Sub Class_Terminate()

    Call Clear

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cVectorLng"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cVectorLng.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This class implements an array of longs.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cVectorLng."

' Array counter
Private mlngCount As Long
Private mcarrItems() As Long

Public Sub Add(ByVal lngItem As Long)
    ' Adds the passed in long variable to the array

    On Error GoTo AddErr

    ReDim Preserve mcarrItems(mlngCount)

    ' Set the newly added element in the array to the
    ' passed in variable
    mcarrItems(mlngCount) = lngItem
    mlngCount = mlngCount + 1

    Exit Sub

AddErr:
```

```
    LogErrors Errors
    gstrSource = mstrModuleName & "Add"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadInArrayFailed, _
          mstrSource, _
          LoadResString(errLoadInArrayFailed)

End Sub
Public Sub Clear()

    ' Clear the array
    ReDim mcarrItems(0)

End Sub

Public Sub Delete(Optional ByVal Position As Long = -1, _
      Optional ByVal Item As Long = -1)
    ' The user can opt to delete either a specific item in
    ' the list or the item at a specified position. If no
    ' parameters are passed in, we delete the element at
    ' position 0!

    Dim lngDelete As Long
    Dim lngIndex As Long

    On Error GoTo DeleteErr

    If Position = -1 Then
       ' Since we can never store an element at position -1,
       ' we can be sure that the user is trying to delete
       ' a given item
       lngDelete = Find(Item)
    Else
       lngDelete = Position
    End If

    If lngDelete < (mlngCount - 1) Then

       ' We want to maintain the order of all items in the
       ' array - so move all remaining elements in the array
       ' up by 1
       For lngIndex = lngDelete To mlngCount - 2
          MoveDown lngIndex
       Next lngIndex

    End If

    ' Delete the last Node from the array
    mlngCount = mlngCount - 1
    If mlngCount > 0 Then
       ReDim Preserve mcarrItems(0 To mlngCount - 1)
    Else
       ReDim mcarrItems(0)
    End If

    Exit Sub

DeleteErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Delete"
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteArrayElementFailed, _
          mstrSource, _
          LoadResString(errDeleteArrayElementFailed)

End Sub
Public Function Find(ByVal Item As Long) As Long

    ' Returns the position at which the passed in value occurs
    ' in the array

    Dim lngIndex As Long

    On Error GoTo FindErr

    ' Find the element in the array to be deleted
    For lngIndex = 0 To mlngCount - 1
       If mcarrItems(lngIndex) = Item Then
          Find = lngIndex
          Exit Function
       End If

    Next lngIndex

    Find = -1

    Exit Function

FindErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Find"
    On Error GoTo 0
    Err.Raise vbObjectError + errItemNotFound, mstrSource, _
       LoadResString(errItemNotFound)

End Function
Public Property Get Item(ByVal Position As Long) As Long
Attribute Item.VB_UserMemId = 0

    ' Returns the element at the passed in position in the array
    If Position >= 0 And Position < mlngCount Then
       Item = mcarrItems(Position)
    Else
       On Error GoTo 0
       Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
          LoadResString(errItemDoesNotExist)
    End If

End Property
Public Property Let Item(ByVal Position As Long, _
       ByVal Value As Long)

    ' Returns the element at the passed in position in the array
    If Position >= 0 Then
       ' If the passed in position is outside the array
       ' bounds, then resize the array
       If Position >= mlngCount Then
          ReDim Preserve mcarrItems(Position)
          mlngCount = Position + 1
       End If

       ' Set the newly added element in the array to the
       ' passed in variable
       mcarrItems(Position) = Value
    Else
       On Error GoTo 0
       Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
          LoadResString(errItemDoesNotExist)
    End If

End Property
Public Sub MoveUp(ByVal Position As Long)
    ' Moves the element at the passed in position up by 1

    Dim lngTemp As Long

    If Position > 0 And Position < mlngCount Then
       lngTemp = mcarrItems(Position)

       mcarrItems(Position) = mcarrItems(Position - 1)
       mcarrItems(Position - 1) = lngTemp
    End If

End Sub
Public Sub MoveDown(ByVal Position As Long)
    ' Moves the element at the passed in position down by 1

    Dim lngTemp As Long

    If Position >= 0 And Position < mlngCount - 1 Then
       lngTemp = mcarrItems(Position)

       mcarrItems(Position) = mcarrItems(Position + 1)
       mcarrItems(Position + 1) = lngTemp
    End If
```

```
End Sub

Public Function Count() As Long

    Count = mlngCount

End Function


Private Sub Class_Initialize()

    mlngCount = 0

End Sub


Private Sub Class_Terminate()

    Call Clear

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cVectorStr"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cVectorStr.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:     This class implements an array of strings.
'  Contact:     Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cVectorStr."

' Array counter
Private mlngCount As Long
Private mcarrItems() As String

Public Sub Add(ByVal strItem As String)
    ' Adds the passed in string variable to the array

    On Error GoTo AddErr

    ReDim Preserve mcarrItems(mlngCount)

    ' Set the newly added element in the array to the
    ' passed in variable
    mcarrItems(mlngCount)= strItem
    mlngCount = mlngCount + 1

    Exit Sub

AddErr:
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "Add"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadInArrayFailed, _
          mstrSource, _
          LoadResString(errLoadInArrayFailed)

End Sub
Public Sub Clear()

    ' Clear the array
    ReDim mcarrItems(0)

End Sub

Public Sub Delete(Optional ByVal Position As Long = -1, _
       Optional ByVal Item As String = -1)
    ' The user can opt to delete either a specific item in
    ' the list or the item at a specified position. If no
    ' parameters are passed in, we delete the element at
    ' position 0!

    Dim lngDelete As Long
    Dim lngIndex As Long

    On Error GoTo DeleteErr
    mstrSource = mstrModuleName & "Delete"

    If Position = -1 Then
       ' Since we can never store an element at position -1,
       ' we can be sure that the user is trying to delete
       ' a given item
       lngDelete = Find(Item)
    Else
       lngDelete = Position
    End If

    If lngDelete < (mlngCount - 1) Then

       ' We want to maintain the order of all items in the
       ' array - so move all remaining elements in the array
       ' up by 1
       For lngIndex = lngDelete To mlngCount - 2
          MoveDown lngIndex
       Next lngIndex

    End If

    ' Delete the last Node from the array
    mlngCount = mlngCount - 1
    If mlngCount > 0 Then
       ReDim Preserve mcarrItems(0 To mlngCount - 1)
    Else
       ReDim mcarrItems(0)
    End If

    Exit Sub

DeleteErr:
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "Delete"
    On Error GoTo 0
    Err.Raise vbObjectError + errDeleteArrayElementFailed, _
          mstrSource, _
          LoadResString(errDeleteArrayElementFailed)

End Sub
Public Function Find(ByVal Item As String) As Long

    ' Returns the position at which the passed in value occurs
    ' in the array

    Dim lngIndex As Long

    On Error GoTo FindErr
    mstrSource = mstrModuleName & "Find"

    ' Find the element in the array to be deleted
    For lngIndex = 0 To mlngCount - 1

       If mcarrItems(lngIndex)= Item Then
          Find = lngIndex
          Exit Function
       End If

    Next lngIndex

    Find = -1

    Exit Function
```

```
FindErr:
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "Find"
    On Error GoTo 0
    Err.Raise vbObjectError + errItemNotFound, mstrSource, _
        LoadResString(errItemNotFound)

End Function
Public Property Get Item(ByVal Position As Long) As String
Attribute Item.VB_UserMemId = 0

    ' Returns the element at the passed in position in the array
    If Position >= 0 And Position < mlngCount Then
        Item = mcarrItems(Position)
    Else
        On Error GoTo 0
        Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
            LoadResString(errItemDoesNotExist)
    End If

End Property
Public Property Let Item(ByVal Position As Long, _
        ByVal Value As String)

    ' Returns the element at the passed in position in the array
    If Position >= 0 Then
        ' If the passed in position is outside the array
        ' bounds, then resize the array
        If Position >= mlngCount Then
            ReDim Preserve mcarrItems(Position)
            mlngCount = Position + 1
        End If

        ' Set the newly added element in the array to the
        ' passed in variable
        mcarrItems(Position) = Value
    Else
        On Error GoTo 0
        Err.Raise vbObjectError + errItemDoesNotExist, mstrSource, _
            LoadResString(errItemDoesNotExist)
    End If

End Property
Public Sub MoveUp(ByVal Position As Long)
    ' Moves the element at the passed in position up by 1

    Dim strTemp As String

    If Position > 0 And Position < mlngCount Then
        strTemp = mcarrItems(Position)

        mcarrItems(Position) = mcarrItems(Position - 1)
        mcarrItems(Position - 1) = strTemp
    End If

End Sub
Public Sub MoveDown(ByVal Position As Long)
    ' Moves the element at the passed in position down by 1

    Dim strTemp As String

    If Position >= 0 And Position < mlngCount - 1 Then
        strTemp = mcarrItems(Position)

        mcarrItems(Position) = mcarrItems(Position + 1)
        mcarrItems(Position + 1) = strTemp
    End If

End Sub

Public Function Count() As Long

    Count = mlngCount

End Function

Private Sub Class_Initialize()

    mlngCount = 0

End Sub
Private Sub Class_Terminate()

    Call Clear

End Sub
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cWorker"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cWorker.cls
'              Microsoft  TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Encapsulates the properties and methods of a worker step.
'              Implements the cStep class - carries out initializations
'              and validations that are specific to worker steps.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Implements cStep

' Object variable to keep the step reference in
Private mcStep As cStep

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cWorker."
Private Property Let cStep_StartDir(ByVal RHS As String)

    mcStep.StartDir = RHS

End Property

Private Property Get cStep_StartDir() As String

    cStep_StartDir = mcStep.StartDir

End Property

Private Property Set cStep_NodeDB(RHS As DAO.Database)

    Set mcStep.NodeDB = RHS

End Property

Private Property Get cStep_NodeDB() As DAO.Database

    Set cStep_NodeDB = mcStep.NodeDB

End Property

Private Function cStep_IncVersionY() As String

    cStep_IncVersionY = mcStep.IncVersionY

End Function
Private Function cStep_IsNewVersion() As Boolean
    cStep_IsNewVersion = mcStep.IsNewVersion
End Function
Private Function cStep_OldVersionNo() As String
    cStep_OldVersionNo = mcStep.OldVersionNo
End Function

Private Function cStep_IncVersionX() As String
```

```
    cStep_IncVersionX = mcStep.IncVersionX

End Function
Private Sub cStep_UpdateIteratorVersion()

    Call mcStep.UpdateIteratorVersion

End Sub

Private Function cStep_IteratorCount() As Long

    cStep_IteratorCount = mcStep.IteratorCount

End Function

Private Sub cStep_UnloadIterators()

    Call mcStep.UnloadIterators

End Sub

Private Sub cStep_SaveIterators()

    Call mcStep.SaveIterators

End Sub
Private Property Get cStep_IteratorName() As String

    cStep_IteratorName = mcStep.IteratorName

End Property
Private Property Let cStep_IteratorName(ByVal RHS As String)

    mcStep.IteratorName = RHS

End Property

Private Sub cStep_LoadIterator(cItRecord As cIterator)

    Call mcStep.LoadIterator(cItRecord)

End Sub
Private Sub cStep_DeleteIterator(cItRecord As cIterator)

    Call mcStep.DeleteIterator(cItRecord)

End Sub

Private Sub cStep_InsertIterator(cItRecord As cIterator)

    Call mcStep.InsertIterator(cItRecord)

End Sub
Private Function cStep_Iterators() As Variant

    cStep_Iterators = mcStep.Iterators

End Function
Private Sub cStep_ModifyIterator(cItRecord As cIterator)

    Call mcStep.ModifyIterator(cItRecord)

End Sub
Private Sub cStep_RemoveIterator(cItRecord As cIterator)

    Call mcStep.RemoveIterator(cItRecord)

End Sub
Private Sub cStep_UpdateIterator(cItRecord As cIterator)

    Call mcStep.UpdateIterator(cItRecord)

End Sub
Private Sub cStep_AddIterator(cItRecord As cIterator)

    Call mcStep.AddIterator(cItRecord)

End Sub

Private Property Let cStep_Position(ByVal RHS As Long)

    mcStep.Position = RHS

End Property

Private Property Get cStep_Position() As Long

    cStep_Position = mcStep.Position

End Property

Private Function cStep_Clone(Optional cCloneStep As cStep) As cStep

    Dim cNewWorker As cWorker

    Set cNewWorker = New cWorker
    Set cStep_Clone = mcStep.Clone(cNewWorker)

End Function

Private Sub StepTextOrFileEntered()
    ' Checks if either the step text or the name of the file containing
    ' the text has been entered
    ' If both of them are null or both of them are not null,
    ' the worker step is invalid and an error is raised
    If StringEmpty(mcStep.StepText) And StringEmpty(mcStep.StepTextFile)
Then
        ShowError errStepTextAndFileNull
        On Error GoTo 0
        Err.Raise vbObjectError + errStepTextAndFileNull,_
            mstrSource, LoadResString(errStepTextAndFileNull)
    End If

End Sub

Private Property Get cStep_IndOperation() As Operation

    cStep_IndOperation = mcStep.IndOperation

End Property

Private Property Let cStep_IndOperation(ByVal RHS As Operation)

    mcStep.IndOperation = RHS

End Property

Private Property Get cStep_NextStepId() As Long

    cStep_NextStepId = mcStep.NextStepId

End Property

Private Property Let cStep_OutputFile(ByVal RHS As String)

    mcStep.OutputFile = RHS

End Property

Private Property Get cStep_OutputFile() As String

    cStep_OutputFile = mcStep.OutputFile

End Property

Private Property Let cStep_ErrorFile(ByVal RHS As String)

    mcStep.ErrorFile = RHS

End Property

Private Property Get cStep_ErrorFile() As String

    cStep_ErrorFile = mcStep.ErrorFile
```

```
End Property
'Private Property Let cStep_LogFile(ByVal RHS As String)
'
'    mcStep.LogFile = RHS
'
'End Property
'
'Private Property Get cStep_LogFile() As String
'
'    cStep_LogFile = mcStep.LogFile
'
'End Property

Private Property Let cStep_ArchivedFlag(ByVal RHS As Boolean)

    mcStep.ArchivedFlag = RHS

End Property

Private Property Get cStep_ArchivedFlag() As Boolean

    cStep_ArchivedFlag = mcStep.ArchivedFlag

End Property

Private Sub Class_Initialize()

    ' Create the object
    Set mcStep = New cStep

    ' Initialize the object with valid values for a Worker step
    ' The global flag should be the first field to be initialized
    ' since subsequent validations might try to check if the
    ' step being created is global
    mcStep.GlobalFlag = False
'    mcStep.GlobalRunMethod = gintNoOption
    mcStep.StepType = gintWorkerStep

End Sub
Private Sub Class_Terminate()

    ' Remove the step object
    Set mcStep = Nothing

End Sub
Private Sub cStep_Add()

    ' Call a private procedure to see if the step text has been
    ' entered - since a worker step actually executes a step, entry
    ' of the text is mandatory
    Call StepTextOrFileEntered

    ' Call the Add method of the step class to carry out the insert
    mcStep.Add

End Sub

Private Property Get cStep_ContinuationCriteria() As ContinuationCriteria

    cStep_ContinuationCriteria = mcStep.ContinuationCriteria

End Property

Private Property Let cStep_ContinuationCriteria(ByVal RHS As
ContinuationCriteria)

    ' The Continuation criteria must be non-null for all worker steps.
    ' Check if the Continuation Criteria is valid
    Select Case RHS
        Case gintOnFailureAbortSiblings,gintOnFailureCompleteSiblings,_
             gintOnFailureSkipSiblings,gintOnFailureAbort,_
             gintOnFailureContinue,gintOnFailureAsk
            mcStep.ContinuationCriteria = RHS

        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errContCriteriaInvalid,_
                mstrModuleName, LoadResString(errContCriteriaInvalid)
    End Select

End Property

Private Property Let cStep_DegreeParallelism(ByVal RHS As String)

    mcStep.DegreeParallelism = RHS

End Property

Private Property Get cStep_DegreeParallelism() As String

    cStep_DegreeParallelism = mcStep.DegreeParallelism

End Property

Private Sub cStep_Delete()

    mcStep.Delete

End Sub

Private Property Get cStep_EnabledFlag() As Boolean

    cStep_EnabledFlag = mcStep.EnabledFlag

End Property

Private Property Let cStep_EnabledFlag(ByVal RHS As Boolean)

    mcStep.EnabledFlag = RHS

End Property

Private Property Let cStep_ExecutionMechanism(ByVal RHS As
ExecutionMethod)

    On Error GoTo ExecutionMechanismErr
    mstrSource = mstrModuleName & "cStep_ExecutionMechanism"

    Select Case RHS
        Case gintExecuteShell, gintExecuteODBC
            mcStep.ExecutionMechanism = RHS

        Case Else
            On Error GoTo 0
            Err.Raise vbObjectError + errExecutionMechanismInvalid,_
                mstrSource, LoadResString(errExecutionMechanismInvalid)
    End Select

    Exit Property

ExecutionMechanismErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "cStep_ExecutionMechanism"
    On Error GoTo 0
    Err.Raise vbObjectError + errExecutionMechanismLetFailed,_
        mstrSource, LoadResString(errExecutionMechanismLetFailed)

End Property

Private Property Get cStep_ExecutionMechanism() As ExecutionMethod

    cStep_ExecutionMechanism = mcStep.ExecutionMechanism

End Property

Private Property Let cStep_FailureDetails(ByVal RHS As String)

    mcStep.FailureDetails = RHS

End Property

Private Property Get cStep_FailureDetails() As String

    cStep_FailureDetails = mcStep.FailureDetails

End Property
```

```
Private Property Get cStep_GlobalFlag() As Boolean

    cStep_GlobalFlag = mcStep.GlobalFlag

End Property

Private Property Let cStep_GlobalFlag(ByVal RHS As Boolean)

    ' Set the global flag to false - this flag is initialized when
    ' an instance of the class is created. Just making sure that
    ' nobody changes the value inadvertently
    mcStep.GlobalFlag = False

End Property
Private Sub cStep_Modify()

    ' Call a private procedure to see if the step text has been
    ' entered - since a worker step actually executes a step, entry
    ' of the text is mandatory
    Call StepTextOrFileEntered

    ' Call the Modify method of the step class to carry out the update
    mcStep.Modify

End Sub

Private Property Let cStep_ParentStepId(ByVal RHS As Long)

    mcStep.ParentStepId = RHS

End Property

Private Property Get cStep_ParentStepId() As Long

    cStep_ParentStepId = mcStep.ParentStepId

End Property

Private Property Let cStep_ParentVersionNo(ByVal RHS As String)

    mcStep.ParentVersionNo = RHS

End Property

Private Property Get cStep_ParentVersionNo() As String

    cStep_ParentVersionNo = mcStep.ParentVersionNo

End Property

Private Property Let cStep_SequenceNo(ByVal RHS As Integer)

    mcStep.SequenceNo = RHS

End Property

Private Property Get cStep_SequenceNo() As Integer

    cStep_SequenceNo = mcStep.SequenceNo

End Property

Private Property Let cStep_StepId(ByVal RHS As Long)

    mcStep.StepId = RHS

End Property

Private Property Get cStep_StepId() As Long

    cStep_StepId = mcStep.StepId

End Property


Private Property Let cStep_StepLabel(ByVal RHS As String)

    mcStep.StepLabel = RHS

End Property

Private Property Get cStep_StepLabel() As String

    cStep_StepLabel = mcStep.StepLabel

End Property


Private Property Let cStep_StepLevel(ByVal RHS As Integer)

    mcStep.StepLevel = RHS

End Property

Private Property Get cStep_StepLevel() As Integer

    cStep_StepLevel = mcStep.StepLevel

End Property

Private Property Let cStep_StepText(ByVal RHS As String)

    mcStep.StepText = RHS

End Property

Private Property Get cStep_StepText() As String

    cStep_StepText = mcStep.StepText

End Property

Private Property Let cStep_StepTextFile(ByVal RHS As String)

    mcStep.StepTextFile = RHS

End Property

Private Property Get cStep_StepTextFile() As String

    cStep_StepTextFile = mcStep.StepTextFile

End Property

Private Property Let cStep_StepType(RHS As gintStepType)

    mcStep.StepType = gintWorkerStep

End Property

Private Property Get cStep_StepType() As gintStepType

    cStep_StepType = mcStep.StepType

End Property

Private Sub cStep_Validate()
    ' The validate routines for each of the steps will
    ' carry out the specific validations for the type and
    ' call the generic validation routine

    On Error GoTo cStep_ValidateErr

    ' Validations specific to worker steps

    ' Check if the step text or a file name has been
    ' specified
    Call StepTextOrFileEntered

    mcStep.Validate

    Exit Sub

cStep_ValidateErr:
    LogErrors Errors
```

```
    mstrSource = mstrModuleName & "cStep_Validate"
    On Error GoTo 0
    Err.Raise vbObjectError + errValidateFailed, _
          mstrSource, _
          LoadResString(errValidateFailed)
End Sub

Private Property Let cStep_VersionNo(ByVal RHS As String)

    mcStep.VersionNo = RHS

End Property

Private Property Get cStep_VersionNo() As String

    cStep_VersionNo = mcStep.VersionNo

End Property

Private Property Let cStep_WorkspaceId(ByVal RHS As Long)

    mcStep.WorkspaceId = RHS

End Property

Private Property Get cStep_WorkspaceId() As Long

    cStep_WorkspaceId = mcStep.WorkspaceId

End Property
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "cWorkspace"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
'  FILE:       cWorkspace.cls
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Encapsulates the properties and methods of a workspace.
'              Contains functions to insert, update and delete
'              att_workspaces records from the database.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Local variable(s) to hold property value(s)
Private mlngWorkspaceId As Long
Private mstrWorkspaceName As String
Private mblnArchivedFlag As Boolean
Private mdbsStepMaster As Database

' Used to indicate the source module name when errors
' are raised by this class
Private mstrSource As String
Private Const mstrModuleName As String = "cWorkspace."

' The cSequence class is used to generate unique workspace identifiers
Private mWorkspaceSeq As cSequence

' The StringSM class is used to carry out string operations
Private mFieldValue As cStringSM

Public Function Clone() As cWorkspace

    ' Creates a copy of a given workspace

    Dim cCloneWsp As cWorkspace

    On Error GoTo CloneErr

    Set cCloneWsp = New cWorkspace

    ' Copy all the workspace properties to the newly
    ' created workspace
    cCloneWsp.WorkspaceId = mlngWorkspaceId
    cCloneWsp.WorkspaceName = mstrWorkspaceName
    cCloneWsp.ArchivedFlag = mblnArchivedFlag

    ' And set the return value to the newly created workspace
    Set Clone = cCloneWsp

    Exit Function

CloneErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "Clone"
    On Error GoTo 0
    Err.Raise vbObjectError + errCloneFailed, _
          mstrSource, LoadResString(errCloneFailed)

End Function

Public Property Let ArchivedFlag(ByVal vdata As Boolean)

    mblnArchivedFlag = vdata

End Property

Public Property Get ArchivedFlag() As Boolean

    ArchivedFlag = mblnArchivedFlag

End Property

Public Property Set WorkDatabase(vdata As Database)

    Set mdbsStepMaster = vdata

End Property

Private Sub WorkspaceNameDuplicate()
    ' Check if the workspace name already exists in the workspace

    Dim rstWorkspace As Recordset
    Dim strSql As String
    Dim qy As DAO.QueryDef

    On Error GoTo WorkspaceNameDuplicateErr
    mstrSource = mstrModuleName & "WorkspaceNameDuplicate"

    ' Create a recordset to retrieve the count of records
    ' having the same workspace name
    strSql = " Select count(*) as workspace_count " & _
       " from att_workspaces " & _
       " where workspace_name = [w_name] " & _
       " and workspace_id <> [w_id] "
    Set qy = mdbsStepMaster.CreateQueryDef(gstrEmptyString, strSql)

    ' Call a procedure to assign the parameter values
    Call AssignParameters(qy)

    Set rstWorkspace = qy.OpenRecordset(dbOpenForwardOnly)

'       mFieldValue.MakeStringFieldValid(mstrWorkspaceName) & _
'       " and workspace_id <> " & _
'       Str(mlngWorkspaceId)
'
'    Set rstWorkspace = mdbsStepMaster.OpenRecordset( _
'       strSQL, dbOpenForwardOnly)

    If rstWorkspace![workspace_count] > 0 Then
       rstWorkspace.Close
       qy.Close
       ShowError errDuplicateWorkspaceName
       On Error GoTo 0
       Err.Raise vbObjectError + errDuplicateWorkspaceName, _
          mstrSource, LoadResString(errDuplicateWorkspaceName)
    End If
    rstWorkspace.Close
```

```
    qy.Close

    Exit Sub

WorkspaceNameDuplicateErr:
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "WorkspaceNameDuplicate"
    On Error GoTo 0
    Err.Raise vbObjectError + errWorkspaceNameDuplicateFailed, _
        mstrSource, LoadResString(errWorkspaceNameDuplicateFailed)

End Sub
Public Property Let WorkspaceName(vdata As String)

    On Error GoTo WorkspaceNameErr
    mstrSource = mstrModuleName & "WorkspaceName"

    If vdata = gstrEmptyString Then

        On Error GoTo 0
        ' Propogate this error back to the caller
        Err.Raise vbObjectError + errWorkspaceNameMandatory, _
            mstrSource, LoadResString(errWorkspaceNameMandatory)
    Else
        mstrWorkspaceName = vdata
    End If
    Exit Property

WorkspaceNameErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "WorkspaceName"
    On Error GoTo 0
    Err.Raise vbObjectError + errWorkspaceNameSetFailed, _
        mstrSource, LoadResString(errWorkspaceNameSetFailed)

End Property

Public Property Let WorkspaceId(vdata As Long)

    On Error GoTo WorkspaceIdErr
    mstrSource = mstrModuleName & "WorkspaceId"

    If (vdata > 0) Then
        mlngWorkspaceId = vdata
    Else
        ' Propogate this error back to the caller
        On Error GoTo 0
        Err.Raise vbObjectError + errWorkspaceIdInvalid, _
            mstrSource, LoadResString(errWorkspaceIdInvalid)
    End If

    Exit Property

WorkspaceIdErr:
    LogErrors Errors
    mstrSource = mstrModuleName & "WorkspaceId"
    On Error GoTo 0
    Err.Raise vbObjectError + errWorkspaceIdSetFailed, _
        mstrSource, LoadResString(errWorkspaceIdSetFailed)

End Property

Public Sub AddWorkspace()

    Dim strInsert As String
    Dim qy As DAO.QueryDef

    On Error GoTo AddWorkspaceErr

    ' Retrieve the next identifier using the sequence class
    Set mWorkspaceSeq = New cSequence
    Set mWorkspaceSeq.IdDatabase = mdbsStepMaster
    mWorkspaceSeq.IdentifierColumn = FLD_ID_WORKSPACE
    mlngWorkspaceId = mWorkspaceSeq.Identifier
    Set mWorkspaceSeq = Nothing

    ' Call procedure to raise an error if the Workspace name
    ' already exists in the db
    Call WorkspaceNameDuplicate

    ' A new record will have the archived_flag turned off
    mblnArchivedFlag = False

    ' Create a temporary querydef object
    strInsert = "insert into att_workspaces " & _
            "( workspace_id, workspace_name, " & _
            " archived_flag ) " & _
            " values ( [w_id], [w_name], [archived] ) "
    Set qy = mdbsStepMaster.CreateQueryDef(gstrEmptyString, strInsert)

    ' Call a procedure to assign the parameter values
    Call AssignParameters(qy)

    qy.Execute dbFailOnError
    qy.Close

'   strInsert = "insert into att_workspaces " & _
'           "( workspace_id, workspace_name, " & _
'           " archived_flag ) " & _
'           " values ( " & _
'           Str(mlngWorkspaceId) & _
'           ", " & mFieldValue.MakeStringFieldValid(mstrWorkspaceName)& _
'           ", " & Str(mblnArchivedFlag)& _
'           " ) "
'   mdbsStepMaster.Execute strInsert, dbFailOnError
'
    Exit Sub

AddWorkspaceErr:

    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "AddWorkspace"
    On Error GoTo 0
    Err.Raise vbObjectError + errWorkspaceInsertFailed, _
        mstrSource, LoadResString(errWorkspaceInsertFailed)

End Sub
Private Sub AssignParameters(qyExec As DAO.QueryDef)
    ' Assigns values to the parameters in the querydef object
    ' The parameter names are cryptic to make them different
    ' from the field names. When the parameter names are
    ' the same as the field names, parameters in the where
    ' clause do not get created.

    Dim prmParam As DAO.Parameter

    On Error GoTo AssignParametersErr
    mstrSource = mstrModuleName & "AssignParameters"

    For Each prmParam In qyExec.Parameters
        Select Case prmParam.Name
            Case "[w_id]"
                prmParam.Value = mlngWorkspaceId

            Case "[w_name]"
                prmParam.Value = mstrWorkspaceName

            Case "[archived]"
                prmParam.Value = mblnArchivedFlag

            Case Else
                ' Write the parameter name that is faulty
                WriteError errInvalidParameter, mstrSource, _
                    prmParam.Name
                On Error GoTo 0
                Err.Raise errInvalidParameter, mstrSource, _
                    LoadResString(errInvalidParameter)
        End Select
    Next prmParam

    Exit Sub

AssignParametersErr:

    mstrSource = mstrModuleName & "AssignParameters"
    Call LogErrors(Errors)
```

```
    On Error GoTo 0
    Err.Raise vbObjectError + errAssignParametersFailed, _
        mstrSource, LoadResString(errAssignParametersFailed)

End Sub
Public Sub DeleteWorkspace()

    Dim strDelete As String
    Dim qy As DAO.QueryDef

    On Error GoTo DeleteWorkspaceErr

    strDelete = "delete from att_workspaces " & _
        " where workspace_id = [w_id]"
    Set qy = mdbsStepMaster.CreateQueryDef(gstrEmptyString, strDelete)

    ' Call a procedure to assign the parameter values
    Call AssignParameters(qy)

    qy.Execute dbFailOnError
    qy.Close

'    mdbsStepMaster.Execute strDelete, dbFailOnError
'        " where workspace_id = " & _
'        Str(mlngWorkspaceId)

    Exit Sub

DeleteWorkspaceErr:
    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "DeleteWorkspace"
    On Error GoTo 0
    Err.Raise vbObjectError + errWorkspaceDeleteFailed, _
        mstrSource, LoadResString(errWorkspaceDeleteFailed)
End Sub

Public Sub ModifyWorkspace()

    Dim strUpdate As String
    Dim qy As DAO.QueryDef

    On Error GoTo ModifyWorkspaceErr

    ' Call procedure to raise an error if the Workspace name
    ' already exists in the db
    Call WorkspaceNameDuplicate

    strUpdate = "update att_workspaces " & _
        " set workspace_name = [w_name] " & _
        ", archived_flag = [archived] " & _
        " where workspace_id = [w_id] "
    Set qy = mdbsStepMaster.CreateQueryDef(gstrEmptyString, strUpdate)

    ' Call a procedure to assign the parameter values
    Call AssignParameters(qy)

    qy.Execute dbFailOnError
    qy.Close

'    strUpdate = "update att_workspaces " & _
'        " set workspace_name = " & _
'        mFieldValue.MakeStringFieldValid(mstrWorkspaceName)& _
'        ", archived_flag = " & _
'        Str(mblnArchivedFlag)& _
'        " where workspace_id = " & _
'        Str(mlngWorkspaceId)
'
'    mdbsStepMaster.Execute strUpdate, dbFailOnError
'
    Exit Sub

ModifyWorkspaceErr:

    Call LogErrors(Errors)
    mstrSource = mstrModuleName & "ModifyWorkspace"
    On Error GoTo 0
    Err.Raise vbObjectError + errWorkspaceUpdateFailed, _
        mstrSource, LoadResString(errWorkspaceUpdateFailed)

End Sub
Public Property Get WorkspaceName() As String

    WorkspaceName = mstrWorkspaceName

End Property

Public Property Get WorkspaceId() As Long

    WorkspaceId = mlngWorkspaceId

End Property

Private Sub Class_Initialize()

    ' Each function will append it's own name to this
    ' variable
    mstrSource = "cWorkspace."

    Set mFieldValue = New cStringSM

End Sub

Private Sub Class_Terminate()

    Set mdbsStepMaster = Nothing
    Set mFieldValue = Nothing

End Sub
Attribute VB_Name = "DatabaseSM"
'  FILE:       DatabaseSM.bas
'              Microsoft  TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Contains all the database initialization/cleanup
'              procedures for the project. Also contains upgrade
'              database upgrade functions.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
' This module is called DatabaseSM, since Database is a standard
' Visual Basic object and we want to avoid any confusion with it.

Option Explicit

Public wrkJet As Workspace
Public dbsAttTool As Database
Public gblnDbOpen As Boolean
Public gRunEngine As rdoEngine

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "DatabaseSM."
Public Const gsDefDBFileExt As String = ".stp"
Private Const msDefDBFile As String = "\SMData" & gsDefDBFileExt

Private Const merrFileNotFound As Integer = 3024
Private Const merrDaoTableMissing As Integer = 3078

Private Const STEPMASTER_SETTINGS_VAL_NAME_DBFILE As String =
"WorkspaceFile"

Public Const DEF_NO_COUNT_DISPLAY         As Boolean = False
Public Const DEF_NO_EXECUTE               As Boolean = False
Public Const DEF_PARSE_QUERY_ONLY         As Boolean = False
Public Const DEF_ANSI_QUOTED_IDENTIFIERS  As Boolean = False
Public Const DEF_ANSI_NULLS               As Boolean = True
Public Const DEF_SHOW_QUERY_PLAN          As Boolean = False
Public Const DEF_SHOW_STATS_TIME          As Boolean = False
Public Const DEF_SHOW_STATS_IO            As Boolean = False
Public Const DEF_PARSE_ODBC_MSG_PREFIXES  As Boolean = True
Public Const DEF_ROW_COUNT                As Long = 0
Public Const DEF_TSQL_BATCH_SEPARATOR     As String = "GO"
Public Const DEF_QUERY_TIME_OUT           As Long = 0
Public Const DEF_SERVER_LANGUAGE          As String = "(Default)"
Public Const DEF_CHARACTER_TRANSLATION    As Boolean = True
```

```
Public Const DEF_REGIONAL_SETTINGS        As Boolean = False

Public Const PARAM_DEFAULT_DIR            As String = "DEFAULT_DIR"
Public Const PARAM_DEFAULT_DIR_DESC       As String = "Default
destination directory " & _
            "for all output and error files. If it is blank, the StepMaster installation
directory will be used."

Public Const CONNECTION_STRINGS_TO_NAME_SUFFIX As String =
"_NAME"

Private Const TBL_RUN_STEP_HDR As String = "run_header"
Private Const TBL_RUN_STEP_DTLS As String = "run_step_details"
Public Const TBL_CONNECTION_DTLS As String = "connection_dtls"
Public Const TBL_CONNECTION_STRINGS As String =
"workspace_connections"
Public Const TBL_STEPS As String = "att_steps"

Public Const FLD_ID_CONN_NAME As String = "connection_name_id"
Public Const FLD_ID_WORKSPACE As String = "workspace_id"
Public Const FLD_ID_STEP As String = "step_id"

Public Const FLD_CONN_DTL_CONNECTION_NAME As String =
"connection_name"
Public Const FLD_CONN_DTL_CONNECTION_STRING As String =
"connection_string_name"
Public Const FLD_CONN_DTL_CONNECTION_TYPE As String =
"connection_type"

Public Const FLD_CONN_STR_CONNECTION_NAME As String =
"connection_name"

Public Const FLD_STEPS_EXEC_MECHANISM As String =
"execution_mechanism"
Public Const FLD_STEPS_EXEC_DTL As String = "start_directory"
Public Const FLD_STEPS_VERSION_NO As String = "version_no"

Public Const DATA_TYPE_CURRENCY As String = "CURRENCY"
Public Const DATA_TYPE_LONG As String = "Long"
Public Const DATA_TYPE_INTEGER As String = "INTEGER"
Public Const DATA_TYPE_TEXT255 As String = "Text(255)"

Public Sub InitRunEngine()

   Set gRunEngine = New rdoEngine
   gRunEngine.rdoDefaultCursorDriver= rdUseServer

End Sub

Public Function DefaultDBFile() As String
   DefaultDBFile = GetSetting(App.Title, "Settings",
STEPMASTER_SETTINGS_VAL_NAME_DBFILE, App.Path &
msDefDBFile)
End Function

Public Sub CloseDatabase()

   Dim dbsInstance As Database
   Dim recInstance As Recordset

   On Error GoTo CloseDatabaseErr

   ' Close all open recordsets and databases in the workspace
   For Each dbsInstance In wrkJet.Databases

      For Each recInstance In dbsAttTool.Recordsets
         recInstance.Close
      Next recInstance
      dbsInstance.Close

   Next dbsInstance

   Set dbsAttTool = Nothing

   gblnDbOpen = False
   wrkJet.Close

   Exit Sub

CloseDatabaseErr:

   Call LogErrors(Errors)
   Resume Next

End Sub

Private Function NoDbChanges(sVerTo As String, sVerFrom As String) As
Boolean

   If sVerTo = gsVersion242 And sVerFrom = gsVersion241 Then
      NoDbChanges = True
   ElseIf sVerTo = gsVersion242 And sVerFrom = gsVersion24 Then
      NoDbChanges = True
   Else
      NoDbChanges = False
   End If

End Function

Public Function SMOpenDatabase(Optional strDbName As String =
gstrEmptyString) As Boolean
   Dim sVersion As String
   Dim bOpeningDb As Boolean ' This flag is used to check if OpenDatabase
failed

   On Error GoTo OpenDatabaseErr

   bOpeningDb = False
   SMOpenDatabase = False

   ' Create Microsoft Jet Workspace object.
   If Not gblnDbOpen Then
      Set wrkJet = CreateWorkspace("att_tool_workspace_setup", "admin",
gstrEmptyString, dbUseJet)
   End If

   ' Prompt the user for the database file if it is not passed in
   If StringEmpty(strDbName) Then
      strDbName = BrowseDBFile
      If StringEmpty(strDbName) Then
         Exit Function
      End If
   End If

   Do
      If gblnDbOpen Then
#If Not RUN_ONLY Then
         CloseOpenWorkspaces
#End If
         Set wrkJet = CreateWorkspace("att_tool_workspace_setup", "admin",
gstrEmptyString, dbUseJet)
      End If

      ' Toggle the bOpeningDb flag around the OpenDatabase method - the value
      ' of this flag will be checked by the error handler to determine if it is
      ' the OpenDatabase that failed.
      BugMessage "DB File: " & strDbName

      bOpeningDb = True
      ' Open the database for exclusive use
      Set dbsAttTool = wrkJet.OpenDatabase(strDbName, Options:=True)
      bOpeningDb = False

      If dbsAttTool Is Nothing Then
         ' If the file is not present in the directory, display
         ' an error and ask the user to enter a new path
         Call ShowError(errOpenDbFailed, OptArgs:=strDbName)

         strDbName = BrowseDBFile
      Else
         sVersion = DBVersion(dbsAttTool)

         ' Make sure the application and db version numbers match
         If sVersion = gsVersion Then
            Call InitializeData(strDbName)
            gblnDbOpen = True
```

```
            SMOpenDatabase = True
         Else
            If UpgradeDb(wrkJet, dbsAttTool, gsVersion, sVersion) Then
               Call InitializeData(strDbName)
               gblnDbOpen = True
               SMOpenDatabase = True
            Else
               dbsAttTool.Close
               Set dbsAttTool = Nothing

               ShowError errVersionMismatch, _
                     OptArgs:=" Please install Version '" & gsVersion & "' of the
workspace definition file."
               strDbName = BrowseDBFile
            End If
         End If
      End If
   Loop While gblnDbOpen = False And Not StringEmpty(strDbName)

   Exit Function

OpenDatabaseErr:
   Call DisplayErrors(Errors)

   ' If the OpenDatabase failed, continue
   If bOpeningDb Then
      Resume Next
   End If

   Call ShowError(errOpenDbFailed, OptArgs:=strDbName)

End Function
Private Sub InitializeData(sDb As String)

   Set gcParameters = New cArrParameters
   Set gcParameters.ParamDatabase = dbsAttTool

   Set gcSteps = New cArrSteps
   Set gcSteps.StepDB = dbsAttTool

   Set gcConstraints = New cArrConstraints
   Set gcConstraints.ConstraintDB = dbsAttTool

   Set gcConnections = New cConnections
   Set gcConnections.ConnDb = dbsAttTool

   Set gcConnDtls = New cConnDtls
   Set gcConnDtls.ConnDb = dbsAttTool

   ' Disable the error handler since this is not a critical step
   On Error GoTo 0
   SaveSetting App.Title, "Settings",
STEPMASTER_SETTINGS_VAL_NAME_DBFILE,sDb
End Sub
Private Sub UpdateContinuationCriteria(dbFile As DAO.Database)

   Dim qyTemp As DAO.QueryDef
   Dim sBuf As String

   On Error GoTo UpdateContinuationCriteriaErr

   sBuf = "Since this version of the executable incorporates failure processing, "
& _
         "the upgrade will update the On Failure field for each of the steps " & _
         "to 'Continue' to be compatible with the existing behaviour. " & _
         "Proceed?"
   If Not Confirm(Buttons:=vbYesNo, strMessage:=sBuf, strTitle:="Upgrade
database") Then
      Exit Sub
   End If

   ' Create a recordset object to retrieve all steps for
   ' the given workspace
   sBuf = " update att_steps a " & _
      " set continuation_criteria = " & CStr(gintOnFailureContinue)& _
      " where archived_flag = [archived] "

   ' Find the highest X-component of the version number
   sBuf = sBuf & " AND cint( mid( version_no, 1, instr( version_no, " & gstrDQ
& gstrVerSeparator & gstrDQ & " ) - 1 ) ) = " & _
      " ( select max( cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) ) " & _
      " from att_steps AS d " & _
      " WHERE a.step_id = d.step_id ) "

   ' Find the highest Y-component of the version number for the highest
X-component
   sBuf = sBuf & " AND cint( mid( version_no, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) + 1 ) ) = " & _
      " ( select max( cint( mid( version_no, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) + 1 ) ) ) " & _
      " from att_steps AS b " & _
      " Where a.step_id = b.step_id " & _
      " AND cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) = " & _
      " ( select max( cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) ) " & _
      " from att_steps AS c " & _
      " WHERE a.step_id = c.step_id ) ) "

   ' Create a temporary Querydef object
   Set qyTemp = dbFile.CreateQueryDef(gstrEmptyString,sBuf)
   qyTemp.Parameters("archived").Value = False

   qyTemp.Execute dbFailOnError
   qyTemp.Close

   Exit Sub

UpdateContinuationCriteriaErr:
   Call LogErrors(Errors)
   Err.Raise vbObjectError + errModifyStepFailed, mstrModuleName, _
         LoadResString(errModifyStepFailed)

End Sub

Private Sub UpdateDbDtls(dbFile As Database, sNewVersion As String)

   Dim sSql As String
   Dim cTemp As New cStringSM

   On Error GoTo UpdateDbDtlsErr

   sSql = "update db_details " & _
         " set db_version = " & cTemp.MakeStringFieldValid(sNewVersion)

   dbFile.Execute sSql, dbFailOnError

   Exit Sub

UpdateDbDtlsErr:
   Call LogErrors(Errors)
   Err.Raise vbObjectError + errUpgradeFailed, mstrModuleName, _
         LoadResString(errUpgradeFailed)

End Sub

Private Sub Upgrade10to21(UpgradeWsp As DAO.Workspace, dbFile As
Database, sVersion As String)

   Dim sSql As String

   On Error GoTo Upgrade10to21Err

   Call UpdateDbDtls(dbFile, sVersion)

   Call UpdateContinuationCriteria(dbFile)

   Exit Sub

Upgrade10to21Err:
   UpgradeWsp.Rollback
   Call LogErrors(Errors)
   Err.Raise vbObjectError + errUpgradeFailed, mstrModuleName, _
         LoadResString(errUpgradeFailed)
```

```
End Sub
Private Sub Upgrade21to23(UpgradeWsp As DAO.Workspace, dbFile As
Database, sVersion As String)

    Dim sBuf As String
    Dim cTempStr As New cStringSM

    On Error GoTo Upgrade21to23Err

    ' Add a parameter type field and a description field to the parameter table
    sBuf = "alter table workspace_parameters " & _
           " add column description TEXT(255) "
    dbFile.Execute sBuf, dbFailOnError

    sBuf = "alter table workspace_parameters " & _
           " add column parameter_type INTEGER "
    dbFile.Execute sBuf, dbFailOnError

    ' Initialize the parameter type on all parameters to indicate generic parameters
    sBuf = "update workspace_parameters " & _
           " set parameter_type = " & CStr(gintParameterGeneric)
    dbFile.Execute sBuf, dbFailOnError

    sBuf = "Release 2.3 onwards, connection string parameters will be " & _
           "displayed in a separate node. After this upgrade, all connection " & _
           "string parameters will appear under the Globals/Connection Strings " & _
           "node in the workspace. "
    Call MsgBox(sBuf, vbOKOnly + vbApplicationModal, "Upgrade database")

    ' Update the parameter type on all parameters that look like db connection
strings
    sBuf = "update workspace_parameters " & _
           " set parameter_type = " & CStr(gintParameterConnect) & _
           " where UCase(parameter_value) like '*DRIVER*' " & _
           " or UCase(parameter_value) like '*DSN*'"
    dbFile.Execute sBuf, dbFailOnError

    ' Add an elapsed time field to the run_step_details table - this field is
    ' needed to store the elapsed time in milliseconds.
    sBuf = "alter table run_step_details " & _
           " add column elapsed_time LONG "
    dbFile.Execute sBuf, dbFailOnError

    ' The failure_details field has some data for the case when an ODBC failure
    ' threshold was specified. Since that's no longer relevant, update the
failure_details
    ' field for records with failure_criteria = gintFailureODBC to empty.
    ' failure_criteria = gintFailureODBC = 1
    sBuf = "update att_steps " & _
           " set failure_details = " &
cTempStr.MakeStringFieldValid(gstrEmptyString) & _
           " where failure_criteria = '1'"
    dbFile.Execute sBuf, dbFailOnError

    Call UpdateDbDtls(dbFile, sVersion)

    UpgradeWsp.CommitTrans

    On Error GoTo DropColumnErr

    UpgradeWsp.BeginTrans

    ' This ddl cannot be in the same transaction as the failure_details update
    ' But we can do this in a separate transaction since we do not expect this
    ' statement to fail - AND, it doesn't matter if this transaction fails
    ' Drop the failure_criteria column from the att_steps table
    sBuf = "alter table att_steps " & _
           " drop column failure_criteria "
    dbFile.Execute sBuf, dbFailOnError

    Exit Sub

DropColumnErr:
    Call LogErrors(Errors)
    ShowError errDeleteColumnFailed
    Exit Sub

Upgrade21to23Err:
    UpgradeWsp.Rollback
    Call LogErrors(Errors)
    Err.Raise vbObjectError + errUpgradeFailed, mstrModuleName, _
          LoadResString(errUpgradeFailed)

End Sub
Private Sub Upgrade23to24(UpgradeWsp As DAO.Workspace, dbFile As
Database, sVersion As String)

    Dim sBuf As String
    Dim cTempStr As New cStringSM
    Dim lId As Long
    Dim rTemp As DAO.Recordset
    Dim rParam As DAO.Recordset
    Dim cTempSeq As cSequence

    On Error GoTo Upgrade23to24Err

    ' Add a new table for connection properties
    sBuf = CreateConnectionsTableScript()
    ' TODO: Not sure of column sizes for row count, tsql_batch_separator and
server_language
    dbFile.Execute sBuf, dbFailOnError

    ' Move all connection parameters from the parameter table to the connections
tables
    ' Insert default values for the newly added connection properties
    sBuf = "select * from workspace_parameters " & _
           "where parameter_type = " & CStr(gintParameterConnect)
    Set rTemp = dbFile.OpenRecordset(sBuf, dbOpenSnapshot)
    lId = 1
    If rTemp.RecordCount <> 0 Then
        rTemp.MoveFirst

        While Not rTemp.EOF
            sBuf = "insert into workspace_connections " & _
              "( workspace_id, connection_id, " & _
              "connection_name, connection_value, " & _
              "description, no_count_display, " & _
              "no_execute, parse_query_only, " & _
              "ANSI_quoted_identifiers, ANSI_nulls, " & _
              "show_query_plan, show_stats_time, " & _
              "show_stats_io, parse_odbc_msg_prefixes, " & _
              "row_count, tsql_batch_separator, " & _
              "query_time_out, server_language, " & _
              "character_translation, regional_settings ) " & _
             " values ( " & _
             Str(rTemp!workspace_id) & ", " & Str(lId) & ", " & _
             cTempStr.MakeStringFieldValid(""& rTemp!parameter_name) & ", " &
_
             cTempStr.MakeStringFieldValid(""& rTemp!parameter_value) & ", " &
_
             cTempStr.MakeStringFieldValid(""& rTemp!Description) & ", " & _
             Str(DEF_NO_COUNT_DISPLAY) & ", " & _
             Str(DEF_NO_EXECUTE) & ", " & Str(DEF_PARSE_QUERY_ONLY)
& ", " & _
             Str(DEF_ANSI_QUOTED_IDENTIFIERS) & ", " &
Str(DEF_ANSI_NULLS) & ", " & _
             Str(DEF_SHOW_QUERY_PLAN) & ", " &
Str(DEF_SHOW_STATS_TIME) & ", " & _
             Str(DEF_SHOW_STATS_IO) & ", " &
Str(DEF_PARSE_ODBC_MSG_PREFIXES) & ", " & _
             Str(DEF_ROW_COUNT) & ", " &
cTempStr.MakeStringFieldValid(DEF_TSQL_BATCH_SEPARATOR) & ", " &
_
             Str(DEF_QUERY_TIME_OUT) & ", " &
cTempStr.MakeStringFieldValid(DEF_SERVER_LANGUAGE) & ", " & _
             Str(DEF_CHARACTER_TRANSLATION) & ", " &
Str(DEF_REGIONAL_SETTINGS) & _
             " ) "
             dbFile.Execute sBuf, dbFailOnError

             lId = lId + 1
             rTemp.MoveNext
        Wend
    End If
    rTemp.Close
```

```
    ' Add an identifier column for the connection_id field
    sBuf = "alter table att_identifiers " & _
           " add column connection_id long "
    dbFile.Execute sBuf, dbFailOnError

    ' Initialize the value of the connection identifier, initialized above
    sBuf = "update att_identifiers " & _
           " set connection_id = " & Str(lId)
    dbFile.Execute sBuf, dbFailOnError

    ' Delete all connection strings from the parameter table
    sBuf = "delete from workspace_parameters " & _
           "where parameter_type = " & CStr(gintParameterConnect)
    dbFile.Execute sBuf, dbFailOnError

    ' Create the built-in parameter, default directory, for each workspace in the db
    Set cTempSeq = New cSequence
    Set cTempSeq.IdDatabase = dbFile
    cTempSeq.IdentifierColumn= "parameter_id"

    sBuf = "select * from att_workspaces "
    Set rTemp = dbFile.OpenRecordset(sBuf, dbOpenSnapshot)
    If rTemp.RecordCount <> 0 Then
       rTemp.MoveFirst

       While Not rTemp.EOF
          sBuf = "select * from workspace_parameters " & _
                 " where workspace_id = " & Str(rTemp!workspace_id) & _
                 " and parameter_name = " &
cTempStr.MakeStringFieldValid(PARAM_DEFAULT_DIR)
          Set rParam = dbFile.OpenRecordset(sBuf, dbOpenSnapshot)
          If rParam.RecordCount <> 0 Then
             rParam.MoveFirst
             ' Since the parameter already exists, change it to a built-in type
             sBuf = "update workspace_parameters " & _
                    " set parameter_type = " & CStr(gintParameterBuiltIn) & _
                    " where workspace_id = " & Str(rTemp!workspace_id) & _
                    " and parameter_id = " & Str(rParam!parameter_id)
          Else
             ' Else, insert a parameter record
             lId = cTempSeq.Identifier
             sBuf = "insert into workspace_parameters " & _
                    "( workspace_id, parameter_id, " & _
                    " parameter_name, parameter_value, " & _
                    " description, parameter_type ) " & _
                    " values ( " & _
                    Str(rTemp!workspace_id) & ", " & Str(lId) & ", " & _
                    cTempStr.MakeStringFieldValid(PARAM_DEFAULT_DIR) & ",
" & _
                    cTempStr.MakeStringFieldValid(gstrEmptyString) & ", " & _

cTempStr.MakeStringFieldValid(PARAM_DEFAULT_DIR_DESC) & ", " & _
                    CStr(gintParameterBuiltIn) & _
                    " ) "
          End If
          dbFile.Execute sBuf, dbFailOnError
          rParam.Close

          rTemp.MoveNext
       Wend
    End If
    rTemp.Close

    Call UpdateDbDtls(dbFile, sVersion)

    Exit Sub

Upgrade23to24Err:
    UpgradeWsp.Rollback
    Call LogErrors(Errors)
    Err.Raise vbObjectError + errUpgradeFailed, mstrModuleName, _
          LoadResString(errUpgradeFailed)

End Sub
Private Sub Upgrade243to25(UpgradeWsp As DAO.Workspace, dbFile As
Database, sVersion As String)

    Dim sBuf As String
    Dim qy As DAO.QueryDef
    Dim rTemp As DAO.Recordset
    Dim lId As Long
    Dim cTempStr As New cStringSM

    On Error GoTo Upgrade243to25Err

    sBuf = "Release " & gsVersion25 & " onwards, new 'Connections' must be
created for all " & _
           "connection strings. " & vbCrLf & vbCrLf & _
           "Connections will appear under the Globals/Connections " & _
           "node in the workspace. " & vbCrLf & _
           "A list of all 'Connections' (instead of 'Connection Strings') " & _
           "in the workspace will be displayed in the 'Connections' field for " & _
           "ODBC steps on the Step definition screen. " & vbCrLf & vbCrLf & _
           "Each Connection can be marked as static or dynamic. " & vbCrLf & _
           "Dynamic connections will be created when a step starts execution and "
& _
           "closed once the step completes. " & vbCrLf & _
           "Static connections will be kept open till the run completes." & vbCrLf &
vbCrLf & _
           "Currently dynamic 'Connections' have been created for all existing
'Connection Strings' " & _
           "with the suffix " & CONNECTION_STRINGS_TO_NAME_SUFFIX
    Call MsgBox(sBuf, vbOKOnly + vbApplicationModal, "Upgrade database")

    ' Add a new table for the connection name entity
    ' This table has been added in order to satisfy the TPC-H requirement that
    ' all the queries in a stream need to be executed on a single connection.
    sBuf = CreateConnectionDtlsTableScript()
    dbFile.Execute sBuf, dbFailOnError

    ' Add an identifier column for the connection_name_id field
    sBuf = "alter table att_identifiers " & _
           " add column " & FLD_ID_CONN_NAME & " long "
    dbFile.Execute sBuf, dbFailOnError

    Call UpdateDbDtls(dbFile, sVersion)

    ' insert connection_dtl records for each of the connection strings
    sBuf = "select * from " & TBL_CONNECTION_STRINGS
    Set rTemp = dbFile.OpenRecordset(sBuf, dbOpenSnapshot)

    sBuf = "insert into " & TBL_CONNECTION_DTLS & _
           "( " & FLD_ID_WORKSPACE & _
           ", " & FLD_ID_CONN_NAME & _
           ", " & FLD_CONN_DTL_CONNECTION_NAME & _
           ", " & FLD_CONN_DTL_CONNECTION_STRING & _
           ", " & FLD_CONN_DTL_CONNECTION_TYPE & " ) " & _
           " values ( [w_id], [c_id], [c_name], [c_str], [c_type] ) "
    Set qy = dbFile.CreateQueryDef("", sBuf)

    lId = glMinId
    If rTemp.RecordCount <> 0 Then
       rTemp.MoveFirst

       While Not rTemp.EOF
          qy.Parameters("w_id").Value = rTemp.Fields(FLD_ID_WORKSPACE)
          qy.Parameters("c_id").Value = lId
          qy.Parameters("c_name").Value =
rTemp.Fields(FLD_CONN_STR_CONNECTION_NAME) &
CONNECTION_STRINGS_TO_NAME_SUFFIX
          qy.Parameters("c_str").Value =
rTemp.Fields(FLD_CONN_STR_CONNECTION_NAME)
          qy.Parameters("c_type").Value = ConnTypeDynamic

          qy.Execute dbFailOnError

          lId = lId + 1
          rTemp.MoveNext
       Wend
    End If
    qy.Close
    rTemp.Close

    ' Initialize the value of the connection_name_id
    sBuf = "update att_identifiers " & _
           " set " & FLD_ID_CONN_NAME & " = " & Str(lId)
```

```
    dbFile.Execute sBuf, dbFailOnError

    ' Update the start_directory field in att_steps to point to the newly
    ' created connections
    Call ReadStepsInWorkspace(rTemp, qy, glInvalidId, dbLoad:=dbFile, _
          bSelectArchivedRecords:=False)

    sBuf = "update " & TBL_STEPS & _
       " set " & FLD_STEPS_EXEC_DTL & " = [c_name] " & _
       " where " & FLD_ID_STEP & " = [s_id] " & _
       " and " & FLD_STEPS_VERSION_NO & " = [ver_no] "
    Set qy = dbFile.CreateQueryDef("", sBuf)

    If rTemp.RecordCount <> 0 Then
       rTemp.MoveFirst

       While Not rTemp.EOF
          If rTemp.Fields(FLD_STEPS_EXEC_MECHANISM).Value=
gintExecuteODBC Then
             If Not (StringEmpty(""& rTemp.Fields(FLD_STEPS_EXEC_DTL)))
Then
                sBuf = rTemp.Fields(FLD_STEPS_EXEC_DTL)
               ' Strip the enclosing "%" characters
                sBuf = Mid(sBuf, 2, Len(sBuf) - 2) &
CONNECTION_STRINGS_TO_NAME_SUFFIX

                qy.Parameters("c_name").Value = sBuf
                qy.Parameters("s_id").Value = rTemp.Fields(FLD_ID_STEP)
                qy.Parameters("ver_no").Value =
rTemp.Fields(FLD_STEPS_VERSION_NO)

                qy.Execute dbFailOnError
             End If
          End If
          rTemp.MoveNext
       Wend
    End If

    qy.Close
    rTemp.Close

    Exit Sub

Upgrade243to25Err:
    UpgradeWsp.Rollback
    Call LogErrors(Errors)
    Err.Raise vbObjectError + errUpgradeFailed, mstrModuleName, _
          LoadResString(errUpgradeFailed)

End Sub
Private Sub Upgrade242to243(UpgradeWsp As DAO.Workspace, dbFile As
Database, sVersion As String)

    Dim sBuf As String
    Dim cTempStr As New cStringSM
    Dim iResponse As Integer

    On Error GoTo DeleteHistoryErr

    Call DeleteRunHistory(dbFile)

    On Error GoTo Upgrade242to243Err

    UpgradeWsp.CommitTrans

    UpgradeWsp.BeginTrans

    ' Add a parameter type field and a description field to the parameter table
    sBuf = "alter table run_step_details " & _
          " add column parent_instance_id LONG "

    dbFile.Execute sBuf, dbFailOnError

    sBuf = "alter table run_step_details " & _
          " add column iterator_value TEXT(255) "

    dbFile.Execute sBuf, dbFailOnError

    Call AlterFieldType(dbFile, TBL_RUN_STEP_DTLS, "start_time",
DATA_TYPE_CURRENCY)
    Call AlterFieldType(dbFile, TBL_RUN_STEP_DTLS, "end_time",
DATA_TYPE_CURRENCY)
    Call AlterFieldType(dbFile, TBL_RUN_STEP_HDR, "start_time",
DATA_TYPE_CURRENCY)
    Call AlterFieldType(dbFile, TBL_RUN_STEP_HDR, "end_time",
DATA_TYPE_CURRENCY)

    Call UpdateDbDtls(dbFile, sVersion)

    Exit Sub

DeleteHistoryErr:
    ' This is not a critical error - continue with upgrade
    Call LogErrors(Errors)
    Resume Next

Upgrade242to243Err:
    UpgradeWsp.Rollback
    Call LogErrors(Errors)
    Err.Raise vbObjectError + errUpgradeFailed, mstrModuleName, _
          LoadResString(errUpgradeFailed)

End Sub
'********************************************************************
' The AlterFieldType Sub procedure requires three string
' parameters. The first string specifies the name of the table
' containing the field to be changed. The second string specifies
' the name of the field to be changed. The third string specifies
' the new data type for the field.
'********************************************************************

Private Sub AlterFieldType(dbFile As Database, TblName As String, FieldName
As String, _
                NewDataType As String)
    Dim qdf As DAO.QueryDef
    Dim sSql As String

    ' Add a temporary field to the table.
    sSql = "ALTER TABLE [" & TblName & _
          "] ADD COLUMN AlterTempField " & NewDataType
    Set qdf = dbFile.CreateQueryDef("", sSql)
    qdf.Execute

    ' Copy the data from old field into the new field.
    qdf.SQL = "UPDATE DISTINCTROW [" & TblName & "] SET
AlterTempField = [" & FieldName & "]"
    qdf.Execute

    ' Delete the old field.
    qdf.SQL = "ALTER TABLE [" & TblName & "] DROP COLUMN [" &
FieldName & "]"
    qdf.Execute

    ' Rename the temporary field to the old field's name.
    dbFile.TableDefs("[" & TblName & "]").Fields("AlterTempField").Name=
FieldName
    dbFile.TableDefs.Refresh

    ' Clean up.
End Sub
Private Sub Upgrade01to21(UpgradeWsp As DAO.Workspace, dbFile As
DAO.Database, sVersion As String)
    Dim sSql As String

    On Error GoTo Upgrade01to21Err

    sSql = "Create table db_details (" & _
          "db_version              Text(50) " & _
    ");"

    dbFile.Execute sSql, dbFailOnError

    sSql = "insert into db_details " & _
          "( db_version ) values ( '" & sVersion & "' ) "

    dbFile.Execute sSql, dbFailOnError
```

```
    Call UpdateContinuationCriteria(dbFile)

    Exit Sub

Upgrade01to21Err:
    Call LogErrors(Errors)
    UpgradeWsp.Rollback
    Err.Raise vbObjectError + errUpgradeFailed, mstrModuleName, _
          LoadResString(errUpgradeFailed)

End Sub
Private Function UpgradeDb(UpgradeWsp As DAO.Workspace, dbFile As
Database, _
       sVerTo As String, sVerFrom As String) As Boolean

    Dim sMsg As String

    On Error GoTo UpgradeDbErr

    UpgradeDb = False
    If Not ValidUpgrade(sVerTo, sVerFrom) Then Exit Function

    If NoDbChanges(sVerTo, sVerFrom) Then
       UpgradeDb = True
       Exit Function
    End If

    sMsg = "The database needs to be upgraded from Version " & sVerFrom & _
          " to Version " & sVerTo & "." & vbCrLf & _
          "Proceed?"
    If Not Confirm(Buttons:=vbYesNo, strMessage:=sMsg, strTitle:="Upgrade
database") Then
       Exit Function
    End If

    UpgradeWsp.BeginTrans

    Select Case sVerFrom
       Case gsVersion243
          Call Upgrade243to25(UpgradeWsp, dbFile, gsVersion25)

       Case gsVersion24, gsVersion241, gsVersion242
          sMsg = "After this upgrade, the run history for previous runs will no
longer be available. " & _
                "Continue?"
          If Not Confirm(Buttons:=vbYesNo, strMessage:=sMsg,
strTitle:="Upgrade database") Then
             UpgradeWsp.CommitTrans
             Exit Function
          End If
          Call Upgrade242to243(UpgradeWsp, dbFile, gsVersion243)
          Call Upgrade243to25(UpgradeWsp, dbFile, gsVersion25)

       Case gsVersion23
          Call Upgrade23to24(UpgradeWsp, dbFile, gsVersion24)
          Call Upgrade242to243(UpgradeWsp, dbFile, gsVersion242)
          Call Upgrade243to25(UpgradeWsp, dbFile, gsVersion25)

       Case gsVersion21
          Call Upgrade21to23(UpgradeWsp, dbFile, gsVersion23)
          Call Upgrade23to24(UpgradeWsp, dbFile, gsVersion24)
          Call Upgrade242to243(UpgradeWsp, dbFile, gsVersion242)
          Call Upgrade243to25(UpgradeWsp, dbFile, gsVersion25)

       Case gsVersion10
          Call Upgrade10to21(UpgradeWsp, dbFile, gsVersion21)
          Call Upgrade21to23(UpgradeWsp, dbFile, gsVersion23)
          Call Upgrade23to24(UpgradeWsp, dbFile, gsVersion24)
          Call Upgrade242to243(UpgradeWsp, dbFile, gsVersion242)
          Call Upgrade243to25(UpgradeWsp, dbFile, gsVersion25)

       Case gsVersion01
          Call Upgrade01to21(UpgradeWsp, dbFile, gsVersion21)
          Call Upgrade21to23(UpgradeWsp, dbFile, gsVersion23)
          Call Upgrade23to24(UpgradeWsp, dbFile, gsVersion24)
          Call Upgrade242to243(UpgradeWsp, dbFile, gsVersion242)
          Call Upgrade243to25(UpgradeWsp, dbFile, gsVersion25)
    End Select

    UpgradeWsp.CommitTrans

    UpgradeDb = True
    Exit Function

UpgradeDbErr:
    Call LogErrors(Errors)
    ShowError errUpgradeFailed

End Function
Private Function DBVersion(TestDb As Database) As String
    ' Retrieves the database version
    Dim rVersion As Recordset

    On Error GoTo DBVersionErr

    Set rVersion = TestDb.OpenRecordset("Select db_version from db_details ", _
          dbOpenForwardOnly)

    BugAssert rVersion.RecordCount <> 0
    DBVersion = rVersion!db_version

    rVersion.Close
    Exit Function

DBVersionErr:
    If Err.Number = merrDaoTableMissing Then
       DBVersion = gsVersion01
    Else
       LogErrors Errors
       Err.Raise vbObjectError + errUpgradeFailed, mstrModuleName, _
             LoadResString(errUpgradeFailed)
    End If

End Function

Private Function ValidUpgrade(sVerTo As String, sVerFrom As String) As
Boolean

    If sVerTo = gsVersion And sVerFrom = gsVersion243 Then
       ValidUpgrade = True
    ElseIf sVerTo = gsVersion And sVerFrom = gsVersion242 Then
       ValidUpgrade = True
    ElseIf sVerTo = gsVersion And sVerFrom = gsVersion241 Then
       ValidUpgrade = True
    ElseIf sVerTo = gsVersion And sVerFrom = gsVersion24 Then
       ValidUpgrade = True
    ElseIf sVerTo = gsVersion And sVerFrom = gsVersion23 Then
       ValidUpgrade = True
    ElseIf sVerTo = gsVersion And sVerFrom = gsVersion21 Then
       ValidUpgrade = True
    ElseIf sVerTo = gsVersion And sVerFrom = gsVersion10 Then
       ValidUpgrade = True
    ElseIf sVerTo = gsVersion And sVerFrom = gsVersion01 Then
       ValidUpgrade = True
    Else
       ValidUpgrade = False
    End If

End Function
Attribute VB_Name = "DebugSM"
'  FILE:      DebugSM.bas
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:    Contains all the functions that carry out error/debug
'              processing for the project.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
' Most of the functions in this module that manipulate the
' error object do not have an On Error GoTo statement - this
' is because it will clear the passed in error object - let
' the calling functions handle the errors raised by this
```

```
' module, if any
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "DebugSM."

Private mcLogFile As cFileSM
Private mcErrorFile As cFileSM

Private Const FORMAT_MESSAGE_FROM_SYSTEM = &H1000
Private Const FORMAT_MESSAGE_IGNORE_INSERTS = &H200
Private Const pNull = 0

Declare Function FormatMessage Lib "kernel32" Alias "FormatMessageA"
(ByVal dwFlags As Long, lpSource As Any, ByVal dwMessageId As Long,
ByVal dwLanguageId As Long, ByVal lpbuffer As String, ByVal nSize As Long,
Arguments As Long) As Long
Public Function Confirm(Optional lngMessageCode As conConfirmMsgCodes, _
    Optional lngTitleCode As conConfirmMsgTitleCodes, _
    Optional TitleParameter As String, _
    Optional ByVal Buttons As Integer = -1, _
    Optional strMessage As String = gstrEmptyString, _
    Optional strTitle As String = gstrEmptyString) _
    As Boolean
  ' Displays a confirmation message corresponding to the
  ' passed in message code. Returns True if the user says
  ' Ok and False otherwise

  Dim intResponse As Integer
  Dim intButtonStyle As Integer

  On Error GoTo ConfirmErr

  Confirm = False

  ' If the buttons style hasn't been specified, set the
  ' default style to display OK and Cancel buttons
  If Buttons = -1 Then
    intButtonStyle = vbOKCancel
  Else
    intButtonStyle = Buttons
  End If

  ' Find the message string for the passed in code
  If StringEmpty(strMessage) Then
    strMessage = Trim$(LoadResString(lngMessageCode))
  End If

  If StringEmpty(strTitle) Then
    strTitle = Trim$(LoadResString(lngTitleCode))
  End If

  If Not StringEmpty(TitleParameter) Then
    strTitle = strTitle & Chr$(vbKeySpace) & _
        gstrSQ & TitleParameter & gstrSQ
  End If

  ' Display the confirmation message with the Cancel button
  ' set to the default - assume that we are confirming
  ' potentially dangerous operations!
  intResponse = MsgBox(strMessage, _
      intButtonStyle + vbQuestion + vbApplicationModal, _
      strTitle)

  ' Translate the user response into a True/False return code
  If intButtonStyle = vbOKCancel Then
    If intResponse = vbOK Then
      Confirm = True
    Else
      Confirm = False
    End If
  Else
    If intResponse = vbYes Then
      Confirm = True
    Else
      Confirm = False
    End If
  End If

  Exit Function

ConfirmErr:
  ' Log the error code raised by Visual Basic
  Call LogErrors(Errors)
  On Error GoTo 0
  gstrSource = mstrModuleName & "Confirm"
  Err.Raise vbObjectError + errConfirmFailed, _
      gstrSource, _
      LoadResString(errConfirmFailed)

End Function
Public Sub LogSystemError()
  Dim eErrCode As Long

  eErrCode = GetLastError()
  If eErrCode <> 0 Then
    WriteToFile "System Error: " & eErrCode & vbCrLf & ApiError(eErrCode),
_
        blnError:=True
  End If

End Sub
Public Function ApiError(ByVal e As Long) As String

  Dim s As String
  Dim c As Long

  s = String(256, 0)
  c = FormatMessage(FORMAT_MESSAGE_FROM_SYSTEM Or _
      FORMAT_MESSAGE_IGNORE_INSERTS, _
      pNull, e, 0&, s, Len(s), ByVal pNull)
  If c Then ApiError = e & ": " & Left$(s, c)

End Function

' Output flags determine output destination of BugAsserts and messages
#Const afLogfile = 1
#Const afMsgBox = 2
#Const afDebugWin = 4
#Const afAppLog = 8

' Display appropriate error message, and then stop
' program.  These errors should NOT be possible in
' shipping product.
Sub BugAssert(ByVal fExpression As Boolean, _
      Optional sExpression As String)
#If afDebug Then
  If fExpression Then Exit Sub
  BugMessage "BugAssert failed: " & sExpression
  Stop
#End If
End Sub

Sub BugMessage(sMsg As String)

#If afDebug And afLogfile Then
  ' Since we are writing log messages, the error flag is turned off
  Call WriteToFile(sMsg, False)
#End If
#If afDebug And afMsgBox Then
  MsgBox sMsg
#End If
#If afDebug And afDebugWin Then
  Debug.Print sMsg
#End If
#If afDebug And afAppLog Then
  App.LogEvent sMsg
#End If

End Sub
Public Function ProjectLogFile() As String

  ProjectLogFile = mcLogFile.FileName

End Function
```

```
Public Function ProjectErrorFile() As String

    ProjectErrorFile = mcErrorFile.FileName

End Function

Private Sub WriteToFile(sMsg As String, Optional ByVal blnError As Boolean)

    ' Calls procedures to write the passed in message to the log -
    ' The blnError flag is used to indicate that the message
    ' should be logged to the error file - by default the log
    ' file is used

    Dim mcFileObj As cFileSM
    Dim strFileName As String
    Dim strFileHdr As String

    On Error GoTo WriteToFileErr

    If blnError Then
        If mcErrorFile Is Nothing Then
            Set mcErrorFile = New cFileSM
        End If
        Set mcFileObj = mcErrorFile
    Else
        If mcLogFile Is Nothing Then
            Set mcLogFile = New cFileSM
        End If
        Set mcFileObj = mcLogFile
    End If

    If StringEmpty(mcFileObj.FileName)Then
        If blnError Then
            strFileName = gstrProjectPath & "\" & App.EXEName & ".ERR"
            strFileHdr = "Stepmaster Errors"
        Else
            strFileName = gstrProjectPath & "\" & App.EXEName & ".DBG"
            strFileHdr = "Stepmaster Log"
        End If

        mcFileObj.FileName = strFileName
        mcFileObj.WriteLine strFileHdr
        mcFileObj.WriteLine "Log start time : " & Now
    End If

    mcFileObj.WriteLine sMsg

    Exit Sub

WriteToFileErr:
    ' Display the error code raised by Visual Basic
    Call DisplayErrors(Errors)
    ' An error message would've been displayed by the called
    ' procedures

End Sub
Public Sub WriteMessage(sMsg As String)

    Call WriteToFile(sMsg, True)

End Sub

Sub BugTerm()
#If afDebug And afLogfile Then
    ' Close log file
    mcLogFile.CloseFile
#End If
End Sub

Public Sub ShowError(ByVal ErrorCode As errErrorConstants, _
        Optional ByVal ErrorSource As String = gstrEmptyString, _
        Optional ByVal OptArgs As String = gstrEmptyString, _
        Optional ByVal DoWriteError As Boolean = True)

    If DoWriteError Then
        ' Call a procedure to write the error to a log file
        Call WriteError(ErrorCode, ErrorSource, OptArgs)
    End If

    ' Re-initialize the values of the Error object before
    ' displaying the error to the user
    Call InitErrObject(ErrorCode, ErrorSource, OptArgs)

    Call DisplayErrors(Errors)

    Err.Clear

End Sub
Public Sub WriteError(ByVal ErrorCode As errErrorConstants, _
        Optional ByVal ErrorSource As String = gstrEmptyString, _
        Optional ByVal OptArgs As String = gstrEmptyString)

    ' Initialize the values of the Error object before
    ' calling the log function
    Call InitErrObject(ErrorCode, ErrorSource, OptArgs)

    Call LogErrors(Errors)

    Err.Clear

End Sub
Private Sub InitErrObject(ByVal ErrorCode As errErrorConstants, _
        Optional ByVal ErrorSource As String = gstrEmptyString, _
        Optional ByVal OptArgs As String = gstrEmptyString)

    Dim lngError As Long

    lngError = IIf(ErrorCode > vbObjectError And ErrorCode < vbObjectError +
65535, _
            ErrorCode - vbObjectError, ErrorCode)
    Err.Number = lngError + vbObjectError
    Err.Description = LoadResString(lngError) & OptArgs
    Err.Source = App.EXEName & ErrorSource

End Sub
Public Sub ShowMessage(ByVal MessageCode As errErrorConstants, _
        Optional ByVal OptArgs As String)

    Dim strMessage As String

    On Error GoTo ShowMessageErr

    strMessage = LoadResString(MessageCode) & OptArgs

    ' Write the error to a log file
    BugMessage strMessage

    MsgBox strMessage, vbOKOnly

    Exit Sub

ShowMessageErr:
    ' Log the error and exit
    Call DisplayErrors(Errors)

End Sub

Public Sub DisplayErrors(myErrCollection As Errors)
    Dim strError As String
    Dim errLoop As Error
    Dim errCode As Long

    ' Enumerate Errors collection and display properties of
    ' each Error object.
    If Err.Number <> 0 Then
        If Err.Number > vbObjectError And Err.Number < (vbObjectError + 65536)
Then
            errCode = Err.Number - vbObjectError
        Else
            errCode = Err.Number
        End If
        strError = "Error # " & Str(errCode) & " was generated by " _
            & Err.Source & Chr(13) & Err.Description
        MsgBox strError, , "Error", Err.HelpFile, Err.HelpContext
    Else
        For Each errLoop In myErrCollection
```

```
        With errLoop
          If Err.Number > vbObjectError And Err.Number < (vbObjectError +
65536) Then
            errCode = .Number - vbObjectError
          Else
            errCode = .Number
          End If
          strError = "Error #" & errCode & vbCrLf
          strError = strError & "    " & .Description & vbCrLf
          strError = strError & _
            "    (Source: " & .Source & ")" & vbCrLf
'         strError = strError & _
'           "Press F1 to see topic " & .HelpContext & vbCrLf
'         strError = strError & _
'           "    in the file " & .HelpFile & "."
        End With

        MsgBox strError
      Next
   End If

End Sub
Public Sub LogErrors(myErrCollection As Errors)
   Dim cColErrors As cVectorStr
   Dim strError As String
   Dim errLoop As Error
   Dim errCode As Long
   Dim lngIndex As Long

   Set cColErrors = New cVectorStr

   ' Enumerate Errors collection and display properties of
   ' each Error object.
   If Err.Number <> 0 Then
      If Err.Number > vbObjectError And Err.Number < (vbObjectError + 65536)
Then
        errCode = Err.Number - vbObjectError
      Else
        errCode = Err.Number
      End If
      strError = "Error # " & Str(errCode) & " was generated by " _
        & Err.Source & vbCrLf & Err.Description

      cColErrors.Add strError
   End If

   ' Log all database errors, if any
   For Each errLoop In myErrCollection
      With errLoop
        If Err.Number > vbObjectError And Err.Number < (vbObjectError +
65536) Then
          errCode = .Number - vbObjectError
        Else
          errCode = .Number
        End If
        strError = "Error #" & errCode & vbCrLf
        strError = strError & "    " & .Description & vbCrLf
        strError = strError & _
          "    (Source: " & .Source & ")" & vbCrLf
      End With

      cColErrors.Add strError
   Next

   ' We can have a error handler now that we have stored all
   ' errors away safely! - having an error handler before
   ' enumerating all the errors would have cleared the error
   ' collection
   On Error GoTo LogErrorsErr
   gstrSource = mstrModuleName & "LogErrors"

   For lngIndex = 0 To cColErrors.Count - 1
      strError = cColErrors(lngIndex)
      Debug.Print strError
      Call WriteToFile(strError, True)
   Next lngIndex

   Set cColErrors = Nothing
   Exit Sub

LogErrorsErr:
   ' Display the error code raised by Visual Basic
   DisplayErrors Errors
   On Error GoTo 0
   ShowError errUnableToWriteError, DoWriteError:=False

End Sub
// Execute.cpp : Implementation of CExecute
#include "stdafx.h"

#include "ExecuteDll.h"
#include "SMExecute.h"
#include "Execute.h"

extern SQLHENV henv;

extern SM_Connection_Info               *p_Connections;
                // Pointer to open connections
extern int
iConnectionCount;                                       // Number of open
connections
extern CRITICAL_SECTION                         hConnections;
                        // Critical section to serialize access to available
connections

#ifdef _TPCH_AUDIT
                extern FILE *pfLogFile;
                                                        // Log file containing
timestamps
                extern CRITICAL_SECTION hLogFileWrite;
                        // Handle to critical section
#endif

////////////////////////////////////////////////////////////////////////
// CExecute

char * CExecute::m_szOdbcOps[] = {
                        "SQLAllocHandle",
                        "SQLDriverConnect",
                        "SQLExecDirect",
                        "SQLSetStmtAttr",
                        "SQLCancel",
                        "SQLNumResultCols",
                        "SQLDescribeCol",
                        "SQLColAttribute",
                        "SQLFetch",
                        "SQLGetData",
                        "SQLRowCount",
                        "SQLMoreResults"
                };

STDMETHODIMP CExecute::InterfaceSupportsErrorInfo(REFIIDriid)
{
                static const IID* arr[] =
                {
                        &IID_IExecute
                };
                for (int i=0; i < sizeof(arr) / sizeof(arr[0]); i++)
                {
                        if (InlineIsEqualGUID(*arr[i],riid))
                                return S_OK;
                }
                return S_FALSE;
}

STDMETHODIMP CExecute::put_OutputFile(BSTRnewVal)
{
                assert(m_pOutputFile);
                m_OutputFile = newVal;

                HRESULT   hr = m_pOutputFile->put_FileName(newVal);
                if FAILED(hr)
                {
                        m_pOutputFile->Release();
                        m_pOutputFile = NULL;
```

```
		}
		return hr;
}

//DEL STDMETHODIMP CExecute::put_LogFile(BSTR newVal)
//DEL {
//DEL 	assert(m_pLogFile);
//DEL
//DEL 	m_pLogFile->put_FileName(newVal);
//DEL 	return S_OK;
//DEL }

STDMETHODIMP CExecute::put_ErrorFile(BSTR newVal)
{
		assert(m_pErrorFile);
		m_ErrorFile = newVal;

		HRESULT	hr = m_pErrorFile->put_FileName(newVal);
		if FAILED(hr)
		{
			m_pErrorFile->Release();
			m_pErrorFile = NULL;
		}
		return hr;
}

STDMETHODIMP CExecute::DoExecute(BSTR szCommand, BSTR
szExecutionDtls, ExecutionType ExecMethod, \

		BOOL bNoCount, BOOL bNoExecute, BOOL bParseOnly, BOOL
bQuotedIds, \

		BOOL bAnsiNulls, BOOL bShowQP, BOOL bStatsTime, BOOL
bStatsIO, \

		long lRowCount, long lQueryTmout, BSTR szConnection)
{
  HANDLE		hThrd;
  DWORD		tid;

  _CrtSetReportFile(_CRT_WARN, _CRTDBG_FILE_STDOUT);

		m_szCommand = szCommand;
		m_szExecDtls = szExecutionDtls;

		m_ExecMthd = ExecMethod;
		if (m_ExecMthd == execODBC)
  {
    m_bNoCount = bNoCount;
			m_bNoExecute = bNoExecute;
			m_bParseOnly = bParseOnly;
			m_bQuotedIds = bQuotedIds;
			m_bAnsiNulls = bAnsiNulls;
			m_bShowQP = bShowQP;
			m_bStatsTime = bStatsTime;
			m_bStatsIO = bStatsIO;
			m_lRowCount = lRowCount;
			m_lQueryTmout = lQueryTmout;
			m_szConnection = szConnection;
  }

		if((hThrd = CreateThread( 0, 0,
(LPTHREAD_START_ROUTINE)ExecutionThread,
				this, 0, &tid)) == NULL)
			return(RaiseSystemError());

		CloseHandle(hThrd);

		return S_OK;
}

STDMETHODIMP CExecute::Abort()
{
		if (m_ExecMthd == execShell)
			return(AbortShell());
		else
			return(AbortODBC());
}

void ExecutionThread(LPVOID lpParameter)
{

		CExecute	*MyExecute = (CExecute*)lpParameter;

		MyExecute->m_tElapsedTime = 0;

		GetLocalTime(&MyExecute->m_tStartTime);
		MyExecute->PostMessage(WM_TASK_START, 0, 0);

#ifdef _TPCH_AUDIT
		char			szBuffer[MAXLOGCMDBUF];
		char			szFmt[MAXBUFLEN];

		sprintf(szFmt, "Start Step: '%%.%ds' at '%d/%d/%d
%d:%d:%d:%d'\n",
			MAXLOGCMDLEN,
			MyExecute->m_tStartTime.wMonth,
MyExecute->m_tStartTime.wDay,
			MyExecute->m_tStartTime.wYear,
MyExecute->m_tStartTime.wHour,
			MyExecute->m_tStartTime.wMinute,
MyExecute->m_tStartTime.wSecond,
			MyExecute->m_tStartTime.wMilliseconds);
		if (MyExecute->m_ExecMthd == execShell)

WriteFileToTpchLog((LPSTR)MyExecute->m_szCommand,szFmt);
		else
		{
			sprintf(szBuffer, szFmt,
(LPSTR)MyExecute->m_szCommand);
			WriteToTpchLog(szBuffer);
		}
#endif

		// Initialize the run status for the step to running. The completion
status for
		// the step will be initialized by the Shell and ODBC execution
functions.
  MyExecute->m_StepStatus = gintRunning;

		if (MyExecute->m_ExecMthd == execShell)
			MyExecute->m_tElapsedTime =
MyExecute->ExecuteShell(MyExecute);
		else
			MyExecute->m_tElapsedTime =
MyExecute->ExecuteODBC(MyExecute);

		// Close the output, log and error files
		if (MyExecute->m_pOutputFile)
			MyExecute->m_pOutputFile->Release();
		MyExecute->m_pOutputFile = NULL;

		MyExecute->m_ExecTime = NULL;

		GetLocalTime(&MyExecute->m_tEndTime);

#ifdef _TPCH_AUDIT
		sprintf(szFmt, "Complete Step: '%%.%ds' at '%d/%d/%d
%d:%d:%d:%d'\n",
			MAXLOGCMDLEN,
			MyExecute->m_tEndTime.wMonth,
MyExecute->m_tEndTime.wDay,
			MyExecute->m_tEndTime.wYear,
MyExecute->m_tEndTime.wHour,
			MyExecute->m_tEndTime.wMinute,
MyExecute->m_tEndTime.wSecond,
			MyExecute->m_tEndTime.wMilliseconds);
		if (MyExecute->m_ExecMthd == execShell)

WriteFileToTpchLog((LPSTR)MyExecute->m_szCommand,szFmt);
		else
		{
			sprintf(szBuffer, szFmt,
(LPSTR)MyExecute->m_szCommand);
			WriteToTpchLog(szBuffer);
		}
```

```
#endif

        MyExecute->PostMessage(WM_TASK_FINISH, 0, 0);

        return;
}

#ifdef _TPCH_AUDIT

void WriteFileToTpchLog(LPSTR szFile, LPSTR szFmt)
{
        // Reads a maximum of MAXLOGCMDBUF characters from the
command file and writes it to the log
        FILE    *fpCmd;
        int             iRead;
        char    szBuf[MAXLOGCMDBUF];
        char    szCmd[MAXLOGCMDLEN];

        if ( pfLogFile != NULL )
        {
                if ( (fpCmd = fopen(szFile, FILE_ACCESS_READ)) !=
NULL)
                {
                        iRead = fread(szCmd, sizeof(char),
sizeof(szCmd) / sizeof(char), fpCmd);
                        if (iRead < MAXLOGCMDLEN)
                                szCmd[iRead] = '\0';
                        else
                                szCmd[MAXLOGCMDLEN
- 1] = '\0';
                        sprintf(szBuf, szFmt, szCmd);
                        WriteToTpchLog(szBuf);
                        fclose(fpCmd);
                }
        }
}

void WriteToTpchLog(char *szMsg)
{
        if (pfLogFile != NULL)
        {
                EnterCriticalSection(&hLogFileWrite);
                fprintf(pfLogFile,szMsg);
                LeaveCriticalSection(&hLogFileWrite);
        }

        return;
}
#endif

TC_TIME CExecute::ExecuteShell(CExecute *p)
{
        STARTUPINFOA                            Start;
        PROCESS_INFORMATION             proc;
        DWORD
exitCode;
        TC_TIME
tElapsed = 0;
        _bstr_t
szCommand("cmd /c ");
        LPSTR
szStartDir;
        CURRENCY
Elapsed;

        szCommand += p->m_szCommand;

        // Redirect output and error information
        szCommand += " > " + m_OutputFile + " 2> " + m_ErrorFile;

        // Initialize the STARTUPINFO structure:
        memset(&Start, 0, sizeof(STARTUPINFOA));
        Start.cb                = sizeof(Start);
        Start.dwFlags           = STARTF_USESHOWWINDOW;
        Start.wShowWindow       = SW_SHOWMINNOACTIVE;

        memset(&proc, 0, sizeof(PROCESS_INFORMATION));

        szStartDir = strcmp((LPCTSTR)m_szExecDtls, "") == 0 ? NULL :
(LPSTR)m_szExecDtls;

        p->m_ExecTime->Start();

        // Start the shelled application:
        if (!CreateProcessA( NULL, (LPSTR)szCommand, NULL, NULL,
FALSE,
                        NORMAL_PRIORITY_CLASS, NULL, szStartDir,
&Start, &proc ))
        {
                m_StepStatus = gintFailed;
                LogSystemError(p->m_pErrorFile);

                p->m_ExecTime->Stop(&Elapsed);
                return((TC_TIME)Elapsed.int64);
        }

        m_hHandle = proc.hProcess;
        // Give the process time to execute and finish
        WaitForSingleObject(m_hHandle,INFINITE);
        p->m_ExecTime->Stop(&Elapsed);

        if (!GetExitCodeProcess(m_hHandle,&exitCode))
        {
                m_StepStatus = gintFailed;
                LogSystemError(p->m_pErrorFile);
        }
        else
                m_StepStatus = gintComplete;

        // Close all open handles to the shelled process
        CloseHandle(m_hHandle);

        return((TC_TIME)Elapsed.int64);
}

STDMETHODIMP CExecute::AbortShell()
{
        if (m_hHandle != SQL_NULL_HSTMT)
                if (!TerminateProcess(m_hHandle,0))
                        return(RaiseSystemError());

        return(S_OK);
}

TC_TIME CExecute::ExecuteODBC(CExecute *p)
{
        TC_TIME                                 tElapsed = 0;
        HDBC                                    m_hdbc;
        SQLRETURN                               rc;
        LPSTR                                   szCmd;
        CURRENCY                                Elapsed;
        BOOL                                    bDoConnect =
FALSE;

        // ODBC specific initialization
        m_hdbc = SQL_NULL_HDBC;

        // Allocate a new connection if we are creating a dynamic connection
or if
        // the named connection doesn't exist
        if (!InitializeConnection(&m_hdbc,&bDoConnect))
                return(tElapsed);

        if (bDoConnect)
        {
                // Allocate connection handle, open a connection and
set connection attributes.
#ifdef _DEBUG
                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLDriverConnect.\n");
#endif

                if (m_bAbort)
                        return(tElapsed);
```

```
                        // Connect to the server using the passed in connection
string
                        rc = SQLDriverConnect(m_hdbc, NULL,
                                (unsigned char
*)(LPSTR)p->m_szExecDtls, SQL_NTS,
                                NULL, 0, NULL,
SQL_DRIVER_NOPROMPT);
                        if (rc != SQL_SUCCESS)
                        {
                                if (!HandleODBCError(rc,
SQL_HANDLE_DBC, m_hdbc, &m_hdbc, SMSQLDriverConnect))
                                        return(tElapsed);
                        }
                }

#ifdef _DEBUG
                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLAllocHandle for hdbc.\n");
#endif

                if (!m_bAbort && (rc = SQLAllocHandle(SQL_HANDLE_STMT,
m_hdbc, &m_hHandle)) != SQL_SUCCESS)
                {
                        if (!HandleODBCError(rc, SQL_HANDLE_DBC,
m_hdbc, &m_hdbc, SMSQLAllocHandle))
                                return(tElapsed);
                }

                // Set connection attributes if any have been modified from the
default values
                if (m_lRowCount > 0)
                {
                        char
szConnOptions[512];

                        sprintf(szConnOptions, "SET ROWCOUNT %d ",
m_lRowCount);
                        if (!SetConnectionOption(szConnOptions,&m_hdbc))
                                return(tElapsed);
                }

                if (m_bQuotedIds)
                {
                        if (!SetConnectionOption("SET
QUOTED_IDENTIFIER ON ", &m_hdbc))
                                return(tElapsed);
                }

                if (!m_bAnsiNulls)
                {
                        if (!SetConnectionOption("SET
ANSI_NULL_DFLT_OFF ON ", &m_hdbc))
                                return(tElapsed);
                }
                if (!m_bAbort && m_lQueryTmout > 0)
                {
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLSetStmtAttr.\n");
#endif

                        // Set the query timeout on the statement handle
                        rc = SQLSetStmtAttr(m_hHandle,
SQL_ATTR_QUERY_TIMEOUT, &m_lQueryTmout,
                                SQL_IS_UINTEGER);

                        if (!HandleODBCError(rc, SQL_HANDLE_STMT,
m_hHandle, &m_hdbc, SMSQLSetStmtAttr))
                                return(tElapsed);
                }

                if (m_bNoExecute)
                {
                        if (!SetConnectionOption("SET NOEXEC ON ",
&m_hdbc))
                                return(tElapsed);
                }
                else if (m_bParseOnly)
                {
                        if (!SetConnectionOption("SET PARSEONLY ON ",
&m_hdbc))
                                return(tElapsed);
                }
                else if (m_bShowQP)
                {
                        // Important to ensure that this is the last connection
attributes being set -
                        // otherwise showplans are generated for all remaining
SET statements
                        if (!SetConnectionOption("SET SHOWPLAN_TEXT
ON ", &m_hdbc))
                                return(tElapsed);
                }
                else
                {
                        if (m_bNoCount)
                        {
                                if (!SetConnectionOption("SET
NOCOUNT ON ", &m_hdbc))
                                        return(tElapsed);
                        }

                        if (m_bStatsIO)
                        {
                                if (!SetConnectionOption("SET
STATISTICS IO ON ", &m_hdbc))
                                        return(tElapsed);
                        }

                        // Important to ensure that this is the last connection
attributes being set -
                        // otherwise timing statistics are generated for all
remaining SET statements
                        if (m_bStatsTime)
                        {
                                if (!SetConnectionOption("SET
STATISTICS TIME ON ", &m_hdbc))
                                        return(tElapsed);
                        }
                }

                m_szCmd = (LPSTR)p->m_szCommand;
                p->m_ExecTime->Start();

                while ((szCmd = NextCmdInBatch((LPSTR)p->m_szCommand))!=
NULL && !m_bAbort)
                {
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLExecDirect.\n");
#endif

                        // Execute the ODBC command
                        rc = SQLExecDirect(m_hHandle,(unsigned char
*)szCmd, SQL_NTS);
                        if (!HandleODBCError(rc, SQL_HANDLE_STMT,
m_hHandle, &m_hdbc, SMSQLExecDirect))
                                return(tElapsed);

                        free(szCmd);

                        // Call a procedure to log the results to the output file
                        ProcessResultsets();
                }
                p->m_ExecTime->Stop(&Elapsed);

                ResetConnectionProperties(&m_hdbc);

                ODBCCleanup(&m_hdbc, &m_hHandle);

                if (m_StepStatus != gintFailed)
                        m_StepStatus = gintComplete;

                return((DWORD)Elapsed.int64);
}
```

```
BOOL CExecute::InitializeConnection(HDBC*phdbc, BOOL *pbDoConnect)
{
	SQLRETURN rc;

	*pbDoConnect = TRUE;

	if (IsDynamicConnection())
	{
#ifdef _DEBUG
		_CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLAllocHandle for m_hdbc.\n");
#endif

		if (!m_bAbort && (rc =
SQLAllocHandle(SQL_HANDLE_DBC,henv, phdbc)) != SQL_SUCCESS)
		{
			if (!HandleODBCError(rc,
SQL_HANDLE_ENV, henv, phdbc, SMSQLAllocHandle))
				return FALSE;
		}
		return TRUE;
	}

	EnterCriticalSection(&hConnections);
	// Returns the connection handle if the connection,m_szConnection,
exists
	for (m_iConnectionIndex = iConnectionCount - 1;
m_iConnectionIndex >= 0; m_iConnectionIndex--)
	{
		if (!strcmp( (p_Connections +
m_iConnectionIndex)->szConnectionName,(LPSTR)m_szConnection))
		{
			if (!(p_Connections +
m_iConnectionIndex)->bInUse)
			{
				*phdbc = (p_Connections +
m_iConnectionIndex)->hdbc;
				(p_Connections +
m_iConnectionIndex)->bInUse= TRUE;

				*pbDoConnect = FALSE;
				break;
			}
			else
			{

LeaveCriticalSection(&hConnections);

				m_StepStatus = gintFailed;
				_bstr_t
temp(SM_ERR_CONN_IN_USE);
				if (m_pErrorFile)

m_pErrorFile->WriteLine((BSTR)temp);

				return FALSE;
			}
		}
	}

	if (m_iConnectionIndex < 0)
	{
		// Connection was not found. Allocate connection
handle and add it to list of
		// available connections.
#ifdef _DEBUG
		_CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLAllocHandle for m_hdbc.\n");
#endif

		if (!m_bAbort && (rc =
SQLAllocHandle(SQL_HANDLE_DBC,henv, phdbc)) != SQL_SUCCESS)
		{
			if (!HandleODBCError(rc,
SQL_HANDLE_ENV, henv, phdbc, SMSQLAllocHandle))
				return FALSE;
		}

		m_iConnectionIndex = iConnectionCount++;

		p_Connections = (SM_Connection_Info
*)realloc(p_Connections, iConnectionCount * sizeof(SM_Connection_Info));

		strcpy((p_Connections +
m_iConnectionIndex)->szConnectionName,(LPSTR)m_szConnection);
		(p_Connections + m_iConnectionIndex)->hdbc=
*phdbc;
		(p_Connections + m_iConnectionIndex)->bInUse=
TRUE;
	}

	LeaveCriticalSection(&hConnections);

	return TRUE;
}

void CExecute::ResetConnectionUsage()
{
	if(m_iConnectionIndex >= 0 && m_iConnectionIndex <
iConnectionCount)
	{
		EnterCriticalSection(&hConnections);
		(p_Connections + m_iConnectionIndex)->bInUse=
FALSE;
		LeaveCriticalSection(&hConnections);
	}

	return;
}

BOOL CExecute::ResetConnectionProperties(HDBC *p_hdbc)
{
	SQLRETURN rc;

	// Reset connection attributes if any have been modified from the
default values

	if (m_bNoExecute)
	{
		if (!SetConnectionOption("SETNOEXEC OFF ",
p_hdbc))
			return FALSE;
	}
	else if (m_bParseOnly)
	{
		if (!SetConnectionOption("SETPARSEONLY OFF ",
p_hdbc))
			return FALSE;
	}
	else if (m_bShowQP)
	{
		// Reset connection attributes in reverse order
		if (!SetConnectionOption("SETSHOWPLAN_TEXT
OFF ", p_hdbc))
			return FALSE;
	}
	else
	{
		// Reset connection attributes in reverse order
		if (m_bStatsTime)
		{
			if (!SetConnectionOption("SET
STATISTICS TIME OFF ", p_hdbc))
				return FALSE;
		}

		if (m_bNoCount)
		{
			if (!SetConnectionOption("SET
NOCOUNT OFF ", p_hdbc))
				return FALSE;
		}

		if (m_bStatsIO)
		{
```

```
                        if (!SetConnectionOption("SET
STATISTICS IO OFF ", p_hdbc))
                                return FALSE;
                }
        }

        if (m_lRowCount > 0)
        {
                char
szConnOptions[512];

                sprintf(szConnOptions, "SET ROWCOUNT 0 ");
                if (!SetConnectionOption(szConnOptions, p_hdbc))
                        return FALSE;
        }

        if (m_bQuotedIds)
        {
                if (!SetConnectionOption("SET
QUOTED_IDENTIFIER OFF ", p_hdbc))
                        return FALSE;
        }



        if (!m_bAnsiNulls)

        {

                if (!SetConnectionOption("SET
ANSI_NULL_DFLT_OFFOFF ", p_hdbc))

                        return FALSE;

        }



        if (m_lQueryTmout > 0)

        {

                SQLUINTEGER                     lQueryTmout =
0;



#ifdef _DEBUG

                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLSetStmtAttr.\n");

#endif



                // Set the query timeout on the statement handle

                rc = SQLSetStmtAttr(m_hHandle,
SQL_ATTR_QUERY_TIMEOUT, &lQueryTmout,
                        SQL_IS_UINTEGER);

                if (!HandleODBCError(rc, SQL_HANDLE_STMT,
m_hHandle, p_hdbc, SMSQLSetStmtAttr))
                        return FALSE;
        }

        return TRUE;
}

LPSTR CExecute::NextCmdInBatch(LPSTR szBatch)
{
        LPSTR   szCmd, szSeparator, szStart;
        char    szNext;

        szStart = m_szCmd;

        while ( (szSeparator = strstr(szStart, CMD_SEPARATOR)) !=
NULL)
        {
                szNext = *(szSeparator +
strlen(CMD_SEPARATOR));
                if ( szNext == '\n' || szNext == '\r' || szNext == '\0')
                        break;
                else
                        szStart = szSeparator +
strlen(CMD_SEPARATOR);
        }

        if (!szSeparator)
        {
                // No more GO's
                if (strlen(m_szCmd) > 0)
                {
                        szCmd =
(LPSTR)malloc(strlen(m_szCmd)+ 1);
                        strcpy(szCmd, m_szCmd);
                        m_szCmd += strlen(m_szCmd);
                }
                else
                        szCmd = NULL;
        }
        else if (szSeparator - m_szCmd > 0)
        {
                // Strip the succeeding newline
                szCmd = (LPSTR)malloc(szSeparator - m_szCmd);
                strncpy(szCmd, m_szCmd, szSeparator - m_szCmd -
1);
                *(szCmd + (szSeparator - m_szCmd - 1)) = '\0';
                m_szCmd += szSeparator - m_szCmd +
strlen(CMD_SEPARATOR);
                if ( szNext == '\n' || szNext == '\r')
                        m_szCmd += 1;
        }
        else
                szCmd = NULL;

        return(szCmd);
}

BOOL CExecute::SetConnectionOption(LPSTR szConn, HDBC *pHdbc)
{
        // Executes the passed in connection options 'set' statement. Returns
True if it succeeded
        char                    szConnOptions[512];
        SQLRETURN               rc;

        sprintf(szConnOptions, szConn);

#ifdef _DEBUG
        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLExecDirect for connection option.\n");
#endif

        if (m_bAbort)
                return FALSE;

        rc = SQLExecDirect(m_hHandle, (unsigned char *)szConnOptions,
SQL_NTS);
        if (rc != SQL_SUCCESS)
                LogODBCErrors(rc, SQL_HANDLE_STMT,
m_hHandle, SMSQLExecDirect);

        if (!SQL_SUCCEEDED(rc))
        {
                ODBCCleanup(pHdbc, &m_hHandle);
                return FALSE;
        }

        return TRUE;
}

BOOL CExecute::HandleODBCError(SQLRETURN rc, SWORD fHandleType,
SQLHANDLE handle, HDBC *pHdbc, OdbcOperations OdbcOp)
{
        if (rc != SQL_SUCCESS)
  {
                LogODBCErrors(rc, fHandleType, handle, OdbcOp);
```

```
                        if (!SQL_SUCCEEDED(rc))
                        {
                                ODBCCleanup(pHdbc, &m_hHandle);
                                return FALSE;
                        }
   }
        return TRUE;
}

STDMETHODIMP CExecute::AbortODBC()
{
        m_bAbort = TRUE;

        if (m_hHandle != SQL_NULL_HSTMT)
        {
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLCancel.\n");
#endif

                        SQLRETURN rc = SQLCancel(m_hHandle);
                        if (rc != SQL_SUCCESS)
                                LogODBCErrors(rc,
SQL_HANDLE_STMT, m_hHandle, SMSQLCancel);
        }

        return(S_OK);
}

void CExecute::ProcessResultsets()
{
   SQLSMALLINT *CTypeArray, *CScaleArray;
   SQLINTEGER  *ColLenArray, *DispLenArray;
        SQLSMALLINT iColNameLen, SQLType, iColNull, i, NumCols =
0;
        SQLINTEGER              iDispLen, iRowCount, LenOrInd;
   SQLRETURN            rc;
        char                    szColName[MAX_DATA_LEN + 1];
   void                 *DataPtr;


        if (!m_pOutputFile|| m_bAbort)
                        return;

        do
        {
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLNumResultCols.\n");
#endif

                        // Determine the number of result set columns.
                        rc = SQLNumResultCols(m_hHandle,&NumCols);
                        if (rc != SQL_SUCCESS)
                        {
                                LogODBCErrors(rc,
SQL_HANDLE_STMT, m_hHandle, SMSQLNumResultCols);
                                if (!SQL_SUCCEEDED(rc))
                                        break;
                        }

                        if (NumCols > 0)
                        {

                                // Allocate arrays to hold the C type, scale,
column and display length of the data
                                CTypeArray = (SQLSMALLINT *)
malloc(NumCols * sizeof(SQLSMALLINT));
                                CScaleArray = (SQLSMALLINT *)
malloc(NumCols * sizeof(SQLSMALLINT));
                                ColLenArray = (SQLINTEGER *)
malloc(NumCols * sizeof(SQLINTEGER));
                                DispLenArray = (SQLINTEGER *)
malloc(NumCols * sizeof(SQLINTEGER));

                                for (i = 0; i < NumCols && !m_bAbort;
i++)
                                {
#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLDescribeCol.\n");
#endif

                                        // Get the column description,
include the SQL type
                                        rc =
SQLDescribeCol(m_hHandle, ((SQLUSMALLINT) i)+1,
                                                (unsigned char
*)szColName, sizeof(szColName), &iColNameLen,
                                                &SQLType,
(unsigned long *)&ColLenArray[i], &CScaleArray[i], &iColNull);
                                        if (rc != SQL_SUCCESS)
                                        {

LogODBCErrors(rc, SQL_HANDLE_STMT, m_hHandle, SMSQLDescribeCol);
                                                if
(!SQL_SUCCEEDED(rc))

return;
                                        }

#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLColAttribute.\n");
#endif

                                        if (m_bAbort)
                                                return;

                                        rc =
SQLColAttribute(m_hHandle, ((SQLUSMALLINT) i)+1,
SQL_DESC_DISPLAY_SIZE, NULL, 0, NULL, &iDispLen);
                                        if (rc != SQL_SUCCESS)
                                        {

LogODBCErrors(rc, SQL_HANDLE_STMT, m_hHandle, SMSQLColAttribute);
                                                if
(!SQL_SUCCEEDED(rc))

return;
                                        }

                                        // GetDefaultCType contains
a switch statement that returns the default C type
                                        // for each SQL type.
                                        CTypeArray[i] =
GetDefaultCType(SQLType);
                                        if ( (CTypeArray[i] ==
SQL_C_CHAR || CTypeArray[i] == SQL_C_BINARY) && ColLenArray[i] >
MAX_DATA_LEN)
                                        {
                                                ColLenArray[i]
= MAX_DATA_LEN;
                                                iDispLen =
MAX_DATA_LEN;
                                        }

                                        DispLenArray[i] =
max(iColNameLen, iDispLen);
                                        DispLenArray[i] =
max(DispLenArray[i], sizeof(S_NULL));

                                        // Print the column names in
the header
                                        PrintData(szColName,
SQL_C_CHAR, DispLenArray[i], 0, m_pOutputFile);

                                        // Add a byte for the
null-termination character
                                        ColLenArray[i] += 1;
                                        ColLenArray[i] =
ALIGNBUF(ColLenArray[i]);
                                }
                                m_pOutputFile->WriteLine(NULL);
```

```
                                        // Underline each column name
                                        for (i = 0; i < NumCols; i++)
                                        {
                                                memset(szColName, '-',
DispLenArray[i]);
                                                *(szColName +
DispLenArray[i]) = '\0';
                                                PrintData(szColName,
SQL_C_CHAR, DispLenArray[i], 0, m_pOutputFile);
                                        }
                                        m_pOutputFile->WriteLine(NULL);

                                        // Retrieve and print each row. PrintData
accepts a pointer to the data, its C type,
                                        // and its byte length/indicator.
#ifdef _DEBUG
                                        _CrtDbgReport(_CRT_WARN, NULL, 0,
NULL, "Executing SQLFetch.\n");
#endif

                                        while (!m_bAbort && (rc =
SQLFetch(m_hHandle)) != SQL_NO_DATA)
                                        {
                                                if (!SQL_SUCCEEDED(rc))
                                                {

LogODBCErrors(rc, SQL_HANDLE_STMT, m_hHandle, SMSQLFetch);
                                                        break;
                                                }

                                                for (i = 0; i < NumCols; i++)
                                                {
                                                        // Allocate the
data buffer.
                                                        DataPtr =
malloc(ColLenArray[i]);

#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing SQLGetData.\n");
#endif
                                                        while
(!m_bAbort && (rc=SQLGetData(m_hHandle, i + 1, CTypeArray[i],

DataPtr, ColLenArray[i], &LenOrInd)) != SQL_NO_DATA)
                                                        {
                                                                if
(!SQL_SUCCEEDED(rc))
                                                                {

LogODBCErrors(rc, SQL_HANDLE_STMT, m_hHandle, SMSQLGetData);

if (!SQL_SUCCEEDED(rc))

        return;
                                                                }


                                                                if
(LenOrInd == SQL_NULL_DATA)

PrintData(S_NULL, SQL_C_CHAR, DispLenArray[i], 0, m_pOutputFile);
                                                                else
                                                                {

PrintData((SQLCHAR *)DataPtr, CTypeArray[i], DispLenArray[i],

        CScaleArray[i], m_pOutputFile);

// Currently  printing a maximum ofMAX_DATA_LEN chars.

break;
                                                                }
                                                        }

                                                        free(DataPtr);
                                                }
m_pOutputFile->WriteLine(NULL);
                                        }
                                        m_pOutputFile->WriteLine(NULL);

                                        free(CTypeArray);
                                        free(CScaleArray);
                                        free(ColLenArray);
                                        free(DispLenArray);
                                }

                                // Write io statistics, if applicable
                                LogODBCErrors(rc, SQL_HANDLE_STMT,
m_hHandle, SMSQLFetch);

#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLRowCount.\n");
#endif

                                if (m_bAbort)
                                        break;

                                // action (insert, update, delete) query
                                rc = SQLRowCount(m_hHandle, &iRowCount);
                                if (rc != SQL_SUCCESS)
                                {
                                        LogODBCErrors(rc,
SQL_HANDLE_STMT, m_hHandle, SMSQLRowCount);
                                        if (!SQL_SUCCEEDED(rc))
                                                break;
                                }

                                if (!m_bNoCount && iRowCount != -1)
                                {
                                        sprintf(szColName, "(%d row(s)
affected)", iRowCount);
                                        _bstr_t temp(szColName);
                                        m_pOutputFile->WriteLine((BSTR)temp);
                                        m_pOutputFile->WriteLine(NULL);
                                }

#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLFreeStmt.\n");
#endif

                                if (m_bAbort)
                                        break;

                                SQLFreeStmt(m_hHandle, SQL_UNBIND);

#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLMoreResults.\n");
#endif

                                if (m_bAbort)
                                        break;

                                // Process the next resultset. This function returns
'success with info' even
                                // if there is no other resultset and there are statistics
messages to be printed.
                                // Hence the check for -1 rows before printing.
                                rc=SQLMoreResults(m_hHandle);
                                if (rc != SQL_SUCCESS)
                                {
                                        LogODBCErrors(rc,
SQL_HANDLE_STMT, m_hHandle, SMSQLMoreResults);

                                        if (!SQL_SUCCEEDED(rc))
                                                break;
                                }

        } while (rc != SQL_NO_DATA);

                return;
```

```
}

void CExecute::PrintData(void *vData, SQLSMALLINT CType, SQLINTEGER
IndPtr, SQLSMALLINT iScale, ISMLog *pOutput)
{
	// PrintData accepts a pointer to the data, its C type,
	// and its byte length/indicator. It contains a switch statement that
casts and prints
	// the data according to its type.

	char		*s;
	char		fmt[MAXBUFLEN];
	int			j = 0;
	SQLINTEGER	iColLen = IndPtr + 1;

	assert(iColLen);
	s = (LPSTR)malloc(iColLen+ 1);

	if (s)
	{
		if (vData)
		{
			switch(CType)
			{
			case SQL_C_CHAR:
			case SQL_C_WCHAR:
			case SQL_C_TYPE_DATE:
			case SQL_C_TYPE_TIME:
			case SQL_C_TYPE_TIMESTAMP:
			case SQL_C_INTERVAL_YEAR:
			case SQL_C_INTERVAL_MONTH:
			case
SQL_C_INTERVAL_YEAR_TO_MONTH:
			case SQL_C_INTERVAL_DAY:
			case SQL_C_INTERVAL_HOUR:
			case SQL_C_INTERVAL_MINUTE:
			case SQL_C_INTERVAL_SECOND:
			case
SQL_C_INTERVAL_DAY_TO_HOUR:
			case
SQL_C_INTERVAL_DAY_TO_MINUTE:
			case
SQL_C_INTERVAL_DAY_TO_SECOND:
			case
SQL_C_INTERVAL_HOUR_TO_MINUTE:
			case
SQL_C_INTERVAL_HOUR_TO_SECOND:
			case
SQL_C_INTERVAL_MINUTE_TO_SECOND:
			case SQL_C_BINARY:
				sprintf(fmt, "%%.%ds",
iColLen);
				j = sprintf(s, fmt, (char
*)vData);
				break;

			case SQL_C_SHORT:
				j = sprintf(s, "%d", *(short
*)vData);
				break;

			case SQL_C_LONG:
				j = sprintf(s, "%ld", *(long
*)vData);
				break;

			case SQL_C_UBIGINT:
				j = sprintf(s, "%I64d",
*(__int64 *)vData);
				break;

			case SQL_C_FLOAT:
				sprintf(fmt, "%%.0%df",
iScale);
				j = sprintf(s, fmt, *(float
*)vData);
				break;

			case SQL_C_DOUBLE:
			case SQL_C_NUMERIC:
				sprintf(fmt, "%%.0%df",
iScale);
				j = sprintf(s, fmt, *(double
*)vData);
				break;

			default:
				j = sprintf(s, "%s", vData);
				break;
			}
		}

		// Strip off terminating null character and pad the string
with blanks
		if (iColLen - j > 0)
			memset(s + j, ' ', iColLen - j);

		*(s + iColLen) = '\0';

		// Write the field to the output file
		_bstr_t temp(s);
		pOutput->WriteField((BSTR)temp);
		free(s);
	}

	return;
}

SQLSMALLINT CExecute::GetDefaultCType(SQLINTEGER SQLType)
{
	// GetDefaultCType returns the C type for the passed in SQL
datatype.

  switch(SQLType)
  {
  case SQL_CHAR:
	case SQL_VARCHAR:
	case SQL_LONGVARCHAR:
	case SQL_WCHAR:
	case SQL_WVARCHAR:
	case SQL_WLONGVARCHAR:
	  return(SQL_C_CHAR);

  case SQL_TINYINT:
		return(SQL_C_CHAR);

  case SQL_SMALLINT:
		return(SQL_C_SHORT);

  case SQL_INTEGER:
		return(SQL_C_LONG);

  case SQL_BIGINT:
		return(SQL_C_UBIGINT);

  case SQL_REAL:
		return(SQL_C_FLOAT);

  case SQL_FLOAT:
  case SQL_DOUBLE:
//	case SQL_DECIMAL:
		return(SQL_C_DOUBLE);

	case SQL_DECIMAL:
		return(SQL_C_CHAR);

	case SQL_BIT:
		return(SQL_C_CHAR);

  case SQL_BINARY:
	case SQL_VARBINARY:
	case SQL_LONGVARBINARY:
		return(SQL_C_CHAR);
//		return(SQL_C_BINARY);

  case SQL_TYPE_DATE:
```

```
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_TYPE_DATE);

   case SQL_TYPE_TIME:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_TYPE_TIME);

   case SQL_TYPE_TIMESTAMP:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_TYPE_TIMESTAMP);

            case SQL_NUMERIC:
        return(SQL_C_FLOAT);

   case SQL_INTERVAL_YEAR:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_YEAR);

   case SQL_INTERVAL_MONTH:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_MONTH);

   case SQL_INTERVAL_YEAR_TO_MONTH:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_YEAR_TO_MONTH);

   case SQL_INTERVAL_DAY:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_DAY);

   case SQL_INTERVAL_HOUR:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_HOUR);

   case SQL_INTERVAL_MINUTE:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_MINUTE);

   case SQL_INTERVAL_SECOND:
                                        return(SQL_C_CHAR);
//                                      return(SQL_C_INTERVAL_SECOND);

   case SQL_INTERVAL_DAY_TO_HOUR:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_DAY_TO_HOUR);

   case SQL_INTERVAL_DAY_TO_MINUTE:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_DAY_TO_MINUTE);

   case SQL_INTERVAL_DAY_TO_SECOND:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_DAY_TO_SECOND);

   case SQL_INTERVAL_HOUR_TO_MINUTE:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_HOUR_TO_MINUTE);

   case SQL_INTERVAL_HOUR_TO_SECOND:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_HOUR_TO_SECOND);

   case SQL_INTERVAL_MINUTE_TO_SECOND:
                                        return(SQL_C_CHAR);
//
return(SQL_C_INTERVAL_MINUTE_TO_SECOND);

            default:
                        assert(TRUE);
                        return(SQL_C_CHAR);
                        break;
   }
}

/*
   FUNCTION: LogODBCErrors(SQLRETURN rc, SWORD fHandleType,
SQLHANDLE handle)
   COMMENTS: Formats ODBC errors or warnings and logs them. Also
initializes the
                                        completion status for the step to failure, if
an ODBC error has occurred.
*/

void CExecute::LogODBCErrors(SQLRETURN nResult, SWORD fHandleType,
SQLHANDLE handle, OdbcOperations FailedOp)
{
            // Messages returned by the server (e.g. Print statements) will be
logged to the output file
            // ODBC warnings will be logged to the log file
            // All other ODBC errors will be logged to the error file.

            UCHAR               szErrState[SQL_SQLSTATE_SIZE+1];
            // SQL Error State string
            UCHAR
szErrText[SQL_MAX_MESSAGE_LENGTH+1];   // SQL Error Text string
            char
szBuffer[SQL_SQLSTATE_SIZE+SQL_MAX_MESSAGE_LENGTH+MAXBU
FLEN+1] = "";


// formatted Error text Buffer
            SWORD               wErrMsgLen;
                                                    // Error message
length
            SQLINTEGER          dwErrCode;
                                        // Native Error code
            SQLRETURN           nErrResult;
                                        // Return Code from
SQLGetDiagRec
            SWORD               sMsgNum = 1;
                                        // Error sequence number
            _bstr_t             temp;

            if (IsErrorReturn(nResult))
            {
                        sprintf(szBuffer, "ODBC Operation: '%s' returned error
code: %d",
                                    m_szOdbcOps[FailedOp], nResult);
                        temp = szBuffer;
                        m_pErrorFile->WriteLine((BSTR)temp);
                        m_StepStatus = gintFailed;
            }

            if (handle == SQL_NULL_HSTMT)
                        return;

#ifdef _DEBUG
            _CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLGetDiagRec.\n");
#endif

            // call SQLGetDiagRec function with proper ODBC handles,
repeatedly until
            // function returns SQL_NO_DATA.
            while (!m_bAbort && (nErrResult = SQLGetDiagRec(fHandleType,
handle, sMsgNum++,
                        szErrState, &dwErrCode, szErrText,
SQL_MAX_MESSAGE_LENGTH-1, &wErrMsgLen))
                        != SQL_NO_DATA)
            {
                        if (!SQL_SUCCEEDED(nErrResult))
                                    break;

                        if (m_pOutputFile && IsServerMessage(dwErrCode,
szErrText))
                        {
                                    wsprintf(szBuffer,
SM_SQLMSG_FORMAT, (LPSTR)szErrText);
                                    temp = szBuffer;
                                    m_pOutputFile->WriteLine((BSTR)temp);
```

```
                        }
                        else if (IsODBCWarning(szErrState) && dwErrCode
!= 5701 && dwErrCode != 5703)
                        {
                                // Suppress warnings - 'Changed database
context to...' and 'Changed language setting to...'
                                wsprintf(szBuffer,
SM_SQLMSG_FORMAT, ParseOdbcMsgPrefixes((LPCSTR)szErrText));
                                temp = szBuffer;
                                m_pOutputFile->WriteLine((BSTR)temp);
                        }
                        else if (m_pErrorFile &&
!IsODBCWarning(szErrState))
                        {
                                wsprintf(szBuffer,
SM_SQLERR_FORMAT, (LPSTR)szErrState, dwErrCode, (LPSTR)szErrText);
                                temp = szBuffer;
                                m_pErrorFile->WriteLine((BSTR)temp);
                        }
                }
}

/*
   FUNCTION: ODBCCleanup(HDBC *hdbc, HSTMT *hstmt)
   COMMENTS: Cleanup of all ODBC structures
*/

void CExecute::ODBCCleanup(HDBC *hdbc, HSTMT *hstmt)
{
                SQLRETURN               lReturn;

#ifdef _DEBUG
                _CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
ODBCCleanup.\n");
#endif

                if (*hstmt != SQL_NULL_HSTMT)
                {
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLCloseCursor.\n");
#endif
                        SQLCloseCursor(hstmt);
#ifdef _DEBUG
                        _CrtDbgReport(_CRT_WARN, NULL, 0, NULL,
"Executing SQLFreeHandle for hstmt.\n");
#endif
                        SQLFreeHandle(SQL_HANDLE_STMT,hstmt);
                        *hstmt = SQL_NULL_HSTMT;
                }

                // Cleanup connection if it is a dynamic connection
                if (IsDynamicConnection())
                {
                        if (*hdbc != SQL_NULL_HDBC)
                        {
#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0,
NULL, "Executing SQLDisconnect.\n");
#endif
                                lReturn = SQLDisconnect(*hdbc);
#ifdef _DEBUG
                                _CrtDbgReport(_CRT_WARN, NULL, 0,
NULL, "Executing SQLFreeHandle for hdbc.\n");
#endif
                                SQLFreeHandle(SQL_HANDLE_DBC,
hdbc);
                                *hdbc = SQL_NULL_HDBC;
                        }
                }
                else
                        ResetConnectionUsage();

                return;
}

// Wrapper function that raises an error if a Windows Api fails
STDMETHODIMP CExecute::RaiseSystemError(void)
{
                char s[MAXBUFLEN];

                GetSystemError(s);
                return Error(s, 0, NULL, GUID_NULL);
}

// Wrapper function that logs the error raised by an Api function to the passed in
file
void CExecute::LogSystemError(ISMLog*pFile)
{
                if (pFile)
                {
                        char s[MAXBUFLEN];
                        GetSystemError(s);

                        _bstr_t temp(s);
                        pFile->WriteLine((BSTR)temp);
                }
}

// Populates the passed in string with the last Windows Api error that occurred
void CExecute::GetSystemError(LPSTR s)
{
                long c;
                DWORD e;

                e = GetLastError();

                c = sprintf(s, "Error code: %ld. ", e);
                c = FormatMessage(FORMAT_MESSAGE_FROM_SYSTEM |
FORMAT_MESSAGE_IGNORE_INSERTS,
                        NULL, e, 0, s + c, MAXBUFLEN - c, NULL);

                return;
}

STDMETHODIMP CExecute::get_StepStatus(InstanceStatus *pVal)
{
                *pVal = m_StepStatus;
                return S_OK;
}

STDMETHODIMP CExecute::WriteError(BSTR szMsg)
{
                if (m_pErrorFile)
                        return(m_pErrorFile->WriteLine(szMsg));

                return S_OK;
}
// Execute.h : Declaration of the CExecute

#ifndef __EXECUTE_H_
#define __EXECUTE_H_

#include <atlwin.h>
#include <comdef.h>
#include <stdio.h>
#include "resource.h"       // main symbols
#include "ExecuteDllCP.h"
#include "..\LogWriter\LogWriter.h"
#include "..\LogWriter\SMLog.h"
#include "..\common\SMTime\SMTime.h"
#include "..\common\SMTime\SMTimer.h"

// ODBC-specific includes
#define DBNTWIN32
#include <sqltypes.h>
#include <sql.h>
#include <sqlext.h>

/////////////////////////////////////////////////////////////////////////////
// CExecute

#define WM_TASK_START  (WM_USER + 101)
#define WM_TASK_FINISH  (WM_USER + 102)
```

```
#define SM_SQLERR_FORMAT                "SQL Error State:%s, Native
Error Code: %ld\r\nODBC Error: %s"

                                                                        //
format for ODBC error messages
#define SM_SQLWARN_FORMAT               SM_SQLERR_FORMAT
// format for ODBC warnings
#define SM_SQLMSG_FORMAT                "%s"
        // format for messages from the server

#define SM_SQL_STATE_WARNING            "01000"
#define SM_MSG_SERVER
"[Microsoft][ODBC SQL Server Driver][SQL Server]"

#define SM_ERR_CONN_IN_USE              "StepMaster Error:
Connection is already in use."

#define CMD_SEPARATOR                                   "\nGO"

#define INV_ARRAY_INDEX    -1                           // invalid index
into an array

#define MAXBUFLEN          256      // display buffer size
#define MAXLOGCMDLEN       256      // maximum characters in command
that will be

                                    // printed to log
#define MAXLOGCMDBUF       512      // maximum characters in command
that will be

                                    // printed to log
#define MAX_DATA_LEN       4000     // maximum buffer size for
variable-length data types

                                    // viz. character and binary fields
#define FILE_ACCESS_READ   "r"                          // Open file for
read access

#define ALIGNSIZE 4
#define S_NULL             "NULL"
#define ALIGNBUF(Length) Length % ALIGNSIZE ? \
   Length + ALIGNSIZE - (Length % ALIGNSIZE) : Length

class ATL_NO_VTABLE CExecute :
        public CWindowImpl<CExecute>,
        public CComObjectRootEx<CComSingleThreadModel>,
        public CComCoClass<CExecute, &CLSID_Execute>,
        public IConnectionPointContainerImpl<CExecute>,
        public ISupportErrorInfo,
        public IDispatchImpl<IExecute, &IID_IExecute,
&LIBID_EXECUTEDLLLib>,
        public CProxy_IExecuteEvents< CExecute >
{
public:
        CExecute()
        {
                m_pErrorFile = NULL;
                //m_pLogFile = NULL;
                m_pOutputFile = NULL;

                // Initialize the elapsed time for the step
                m_tElapsedTime = 0;

                // Initialize the run status for the step
                m_StepStatus = gintPending;

                m_hHandle = SQL_NULL_HSTMT;
                m_bAbort = FALSE;

                m_iConnectionIndex = INV_ARRAY_INDEX;
        }

        ~CExecute()
        {
        }

public:
        DECLARE_WND_CLASS("Execute")

                BEGIN_MSG_MAP(CExecute)
                MESSAGE_HANDLER(WM_TASK_FINISH,
OnTaskFinished)
                MESSAGE_HANDLER(WM_TASK_START,
OnTaskStarted)
                END_MSG_MAP()
public:

        LRESULT OnTaskStarted(UINT uMsg, WPARAM wParam,
                LPARAM lParam, BOOL& bHandled)
        {
                CURRENCY        CStartTime =
Get64BitTime(&m_tStartTime);

                Fire_Start(CStartTime);
                return 0;
        }

        LRESULT OnTaskFinished(UINT uMsg, WPARAM wParam,
                LPARAM lParam, BOOL& bHandled)
        {
                CURRENCY        CEndTime =
Get64BitTime(&m_tEndTime);

                Fire_Complete(CEndTime,(long)m_tElapsedTime);
                return 0;
        }

        HRESULT FinalConstruct()
        {
                HRESULT         hr;
                RECT            rect;

                rect.left=0;
                rect.right=100;
                rect.top=0;
                rect.bottom=100;

                HWND hwnd = Create( NULL, rect,
"ExecuteWindow", WS_POPUP);

                if (!hwnd)
                        return
HRESULT_FROM_WIN32(GetLastError());

                hr = CoCreateInstance(CLSID_SMLog, NULL,
CLSCTX_INPROC,
                        IID_ISMLog, (void **)&m_pErrorFile);
                if FAILED(hr)
                        return(hr);
                m_pErrorFile->put_Append(TRUE);

                //hr = CoCreateInstance(CLSID_SMLog, NULL,
CLSCTX_INPROC,
                //      IID_ISMLog, (void **)&m_pLogFile);
                //if FAILED(hr)
                //      return(hr);

                hr = CoCreateInstance(CLSID_SMLog, NULL,
CLSCTX_INPROC,
                        IID_ISMLog, (void **)&m_pOutputFile);
                if FAILED(hr)
                        return(hr);
                m_pOutputFile->put_Append(TRUE);

                hr = CoCreateInstance(CLSID_SMTimer, NULL,
CLSCTX_INPROC,
                        IID_ISMTimer, (void **)&m_ExecTime);
                if FAILED(hr)
                        return(hr);

                return S_OK;
        }

        void FinalRelease()
        {
                if (m_hWnd != NULL)
```

```
                DestroyWindow();

        // Close the log and error files
        if (m_pErrorFile)
                m_pErrorFile->Release();
        m_pErrorFile = NULL;

        //if (m_pLogFile)
        //      m_pLogFile->Release();
        //m_pLogFile = NULL;

        if (m_ExecTime)
                m_ExecTime->Release();
        m_ExecTime = NULL;
    }

DECLARE_REGISTRY_RESOURCEID(IDR_EXECUTE)

DECLARE_PROTECT_FINAL_CONSTRUCT()

BEGIN_COM_MAP(CExecute)
    COM_INTERFACE_ENTRY(IExecute)
    COM_INTERFACE_ENTRY(ISupportErrorInfo)
    COM_INTERFACE_ENTRY(IDispatch)
    COM_INTERFACE_ENTRY(IConnectionPointContainer)
    COM_INTERFACE_ENTRY_IMPL(IConnectionPointContainer)
END_COM_MAP()
BEGIN_CONNECTION_POINT_MAP(CExecute)
CONNECTION_POINT_ENTRY(DIID__IExecuteEvents)
END_CONNECTION_POINT_MAP()

// ISupportsErrorInfo
STDMETHOD(InterfaceSupportsErrorInfo)(REFIIDriid);

// IExecute
public:
    STDMETHOD(put_ErrorFile)(/*[in]*/BSTR newVal);
    STDMETHOD(put_OutputFile)(/*[in]*/BSTR newVal);
    STDMETHOD(WriteError)(BSTR szMsg);
    STDMETHOD(Abort)();
    STDMETHOD(get_StepStatus)(/*[out, retval]*/ InstanceStatus
*pVal);
    STDMETHOD(DoExecute)(/*[in]*/BSTR szCommand, /*[in]*/
BSTR szExecutionDtls, /*[in]*/ ExecutionType ExecMethod,
                /*[in]*/BOOL bNoCount, /*[in]*/BOOL bNoExecute,
/*[in]*/BOOL bParseOnly,
                /*[in]*/BOOL bQuotedIds, /*[in]*/BOOL bAnsiNulls,
/*[in]*/BOOL bShowQP,
                /*[in]*/BOOL bStatsTime, /*[in]*/BOOL bStatsIO,
/*[in]*/long lRowCount,
                /*[in]*/long lQueryTmout, /*[in]*/BSTR
szConnection);

    TC_TIME                 ExecuteShell(CExecute *p);
    TC_TIME                 ExecuteODBC(CExecute *p);
    STDMETHODIMP    AbortShell();
    STDMETHODIMP    AbortODBC();

    _bstr_t                 m_szCommand;
    _bstr_t                 m_szExecDtls;
    _bstr_t                 m_szConnection;
    DWORD                   m_lMode;
    //DATE                  m_CurTime;
    SYSTEMTIME              m_tStartTime;
    SYSTEMTIME              m_tEndTime;
    TC_TIME                 m_tElapsedTime;
    ISMLog                  *m_pErrorFile;
    //ILog                  *m_pLogFile;
    ISMLog                  *m_pOutputFile;
    ISMTimer                *m_ExecTime;
    ExecutionType           m_ExecMthd;
    InstanceStatus          m_StepStatus;
    HANDLE                  m_hHandle;              //
Process handle for shell commands and

                // Statement handle for ODBC commands
    LPSTR                   m_szCmd;

private:
    typedef enum OdbcOperations
    {
        SMSQLAllocHandle,
        SMSQLDriverConnect,
        SMSQLExecDirect,
        SMSQLSetStmtAttr,
        SMSQLCancel,
        SMSQLNumResultCols,
        SMSQLDescribeCol,
        SMSQLColAttribute,
        SMSQLFetch,
        SMSQLGetData,
        SMSQLRowCount,
        SMSQLMoreResults
    };

    LPSTR                   NextCmdInBatch(LPSTR
szBatch);
    void                    ProcessResultsets();
    SQLSMALLINT
GetDefaultCType(SQLINTEGER SQLType);
    void                    PrintData(void *vData,
SQLSMALLINT CType, SQLINTEGER IndPtr, SQLSMALLINT iScale,
ISMLog *pOutput);
    void
LogODBCErrors(SQLRETURN nResult, SWORD fHandleType, SQLHANDLE
handle, OdbcOperations FailedOp);
    void                    ODBCCleanup(HDBC
*hdbc, HSTMT *hstmt);
    STDMETHODIMP    RaiseSystemError(void);
    void                    LogSystemError(ISMLog
*pFile);
    void                    GetSystemError(LPSTR s);
    BOOL                    SetConnectionOption(LPSTR
szConn, HDBC *pHdbc);
    BOOL
ResetConnectionProperties(HDBC *p_hdbc);
    BOOL
HandleODBCError(SQLRETURN rc, SWORD fHandleType, SQLHANDLE
handle, HDBC *pHdbc, OdbcOperations OdbcOp);

    BOOL                    InitializeConnection(HDBC
*phdbc, BOOL *pbDoConnect);
    void                    ResetConnectionUsage();

    int
m_iConnectionIndex;

    static char             *m_szOdbcOps[];

    BOOL                    m_bNoCount,
m_bNoExecute, m_bParseOnly, m_bQuotedIds, m_bAnsiNulls, \
                                    m_bShowQP,
m_bStatsTime, m_bStatsIO;
    long                    m_lRowCount;
    SQLUINTEGER             m_lQueryTmout;
    _bstr_t                 m_ErrorFile, m_OutputFile;
    BOOL                    m_bAbort;

private:
inline BOOL IsServerMessage(SQLINTEGER lNativeError, UCHAR *szErr){
    return( (strstr((LPCTSTR)szErr, SM_MSG_SERVER) != NULL) ?
(lNativeError == 0) : FALSE); }
inline BOOL IsODBCWarning(UCHAR *szSqlState){
return(strcmp((LPCSTR)szSqlState, SM_SQL_STATE_WARNING) == 0);}
inline BOOL IsErrorReturn(SQLRETURN iRetCode){
    return( (iRetCode != SQL_SUCCESS) && (iRetCode !=
SQL_SUCCESS_WITH_INFO) && (iRetCode != SQL_NO_DATA) );}
inline LPCSTR ParseOdbcMsgPrefixes(LPCSTR szMsg){ char *pDest;
    return( (pDest = strstr(szMsg, SM_MSG_SERVER)) == NULL ?
szMsg : pDest + strlen(SM_MSG_SERVER));}
inline BOOL IsDynamicConnection(){ return(!strcmp((LPSTR)m_szConnection,
""));}
};

void                    ExecutionThread(LPVOID lpParameter);
```

```
#ifdef _TPCH_AUDIT
		void 		WriteFileToTpchLog(LPSTR
szFile, LPSTR szFmt);
		void 		WriteToTpchLog(char
*szMsg);
#endif

#endif //__EXECUTE_H_
// ExecuteDll.cpp : Implementation of DLL Exports.


// Note: Proxy/Stub Information
//      To build a separate proxy/stub DLL,
//      run nmake -f ExecuteDllps.mk in the project directory.

#include "stdafx.h"
#include "resource.h"
#include <initguid.h>

#include "..\LogWriter\LogWriter.h"
#include "..\LogWriter\LogWriter_i.c"

#include "..\common\SMTime\SMTime.h"
#include "..\common\SMTime\SMTime_i.c"

#include "ExecuteDll.h"
#include "SMExecute.h"

#include "ExecuteDll_i.c"
#include "Execute.h"

CComModule _Module;

BEGIN_OBJECT_MAP(ObjectMap)
OBJECT_ENTRY(CLSID_Execute, CExecute)
END_OBJECT_MAP()

SQLHENV henv = NULL;
		// ODBC environment handle

static char szCaption[] = "StepMaster";
		// Message box caption

CRITICAL_SECTION hConnections;
		// Critical section to serialize access to available
connections
SM_Connection_Info	*p_Connections = NULL;
		// Pointer to open connections
int
iConnectionCount = 0;		// Number of
open connections

#ifdef _TPCH_AUDIT
		FILE *pfLogFile = NULL;
		// Log file containing
timestamps
		CRITICAL_SECTION hLogFileWrite;
			// Critical section to serialize writes to log

		static char szFileOpenModeAppend[] = "a+";
// Log file open mode

		static char szEnvVarLogFile[] = "TPCH_LOG_FILE";		//
Environment variable - initialized to


// log file name if timing information


// is to be logged
#endif

void ShowODBCErrors(SWORD fHandleType, SQLHANDLE handle);
void CloseOpenConnections();

////////////////////////////////////////////////////////////////////////
// DLL Entry Point

extern "C"
BOOL WINAPI DllMain(HINSTANCE hInstance, DWORD dwReason,
LPVOID /*lpReserved*/)
{

  if (dwReason == DLL_PROCESS_ATTACH)
  {
    _Module.Init(ObjectMap, hInstance, &LIBID_EXECUTEDLLLib);
    DisableThreadLibraryCalls(hInstance);

#ifdef _TPCH_AUDIT
			char szMsg[MAXBUFLEN];
			LPSTR szLogFileName = getenv(szEnvVarLogFile);

			if (szLogFileName == NULL)
			{
				sprintf(szMsg, "The environment variable
'%s' does not exist. "
					"Step timing information will
not be written to a log.", szEnvVarLogFile);
				MessageBox(NULL, szMsg, szCaption,
MB_OK);
			}
			else
			{
				if ( (pfLogFile = fopen(szLogFileName,
szFileOpenModeAppend)) == NULL )
				{
					sprintf(szMsg, "The file '%s'
does not exist. "
						"Step timing
information will not be written to log.",szLogFileName);
					MessageBox(NULL, szMsg,
szCaption, MB_OK);
				}
				else

InitializeCriticalSection(&hLogFileWrite);
			}
#endif

			InitializeCriticalSection(&hConnections);

			p_Connections = NULL;
			iConnectionCount = 0;

			if (!SQL_SUCCEEDED(SQLSetEnvAttr(NULL,
SQL_ATTR_CONNECTION_POOLING,
(SQLPOINTER)SQL_CP_ONE_PER_HENV, 0)))
				ShowODBCErrors(SQL_HANDLE_ENV,
henv);

			if
(!SQL_SUCCEEDED(SQLAllocHandle(SQL_HANDLE_ENV,
SQL_NULL_HANDLE, &henv)))
				return FALSE;
			/*
			SQLUINTEGER CpMatch;
			if (!SQL_SUCCEEDED(SQLGetEnvAttr(henv,
SQL_ATTR_CP_MATCH, &CpMatch, 0, NULL)))
				ShowODBCErrors(SQL_HANDLE_ENV,
henv);

			if (!SQL_SUCCEEDED(SQLSetEnvAttr(henv,
SQL_ATTR_CP_MATCH, (SQLPOINTER)SQL_CP_STRICT_MATCH,
SQL_IS_INTEGER)))
				ShowODBCErrors(SQL_HANDLE_ENV,
henv);
				*/

			if (!SQL_SUCCEEDED(SQLSetEnvAttr(henv,
SQL_ATTR_ODBC_VERSION, (LPVOID)SQL_OV_ODBC3, 0)))
				ShowODBCErrors(SQL_HANDLE_ENV,
henv);

  }
  else if (dwReason == DLL_PROCESS_DETACH)
		{
```

```
#ifdef _TPCH_AUDIT
                        if (pfLogFile != NULL)
                        {
                                fclose(pfLogFile);

                                DeleteCriticalSection(&hLogFileWrite);
                        }
#endif

                        CloseOpenConnections();

                        if (henv != NULL)
                                SQLFreeEnv(henv);

                        DeleteCriticalSection(&hConnections);

        _Module.Term();
                }
   return TRUE;    // ok
}

void ShowODBCErrors(SWORD fHandleType, SQLHANDLE handle)
{
                UCHAR                   szErrState[SQL_SQLSTATE_SIZE+1];
                // SQL Error State string
                UCHAR
szErrText[SQL_MAX_MESSAGE_LENGTH+1];    // SQL Error Text string
                char
szBuffer[SQL_SQLSTATE_SIZE+SQL_MAX_MESSAGE_LENGTH+MAXBU
FLEN+1] = "";


// formatted Error text Buffer
                SWORD                   wErrMsgLen;
                                                        // Error message
length
                SQLINTEGER              dwErrCode;
                                        // Native Error code
                SQLRETURN               nErrResult;
                                        // Return Code from
SQLGetDiagRec
                SWORD                   sMsgNum = 1;
                                        // Error sequence number

                // call SQLGetDiagRec function with proper ODBC handles,
repeatedly until
                // function returns SQL_NO_DATA.
                while ((nErrResult = SQLGetDiagRec(fHandleType, handle,
sMsgNum++,
                        szErrState, &dwErrCode, szErrText,
SQL_MAX_MESSAGE_LENGTH-1, &wErrMsgLen))
                        != SQL_NO_DATA)
                {
                        if (!SQL_SUCCEEDED(nErrResult))
                                break;

                        wsprintf(szBuffer, SM_SQLERR_FORMAT,
(LPSTR)szErrState, dwErrCode, (LPSTR)szErrText);

                        MessageBox(NULL, szBuffer, szCaption, MB_OK);
                }
}

void CloseOpenConnections()
{
                // Closes all open connections

                if (p_Connections)
                {
                        for (int iConnIndex = iConnectionCount - 1;
iConnIndex >= 0; iConnIndex--)
                        {
                                if ((p_Connections + iConnIndex)->hdbc
!= SQL_NULL_HDBC)
                                {
#ifdef _DEBUG
_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing
SQLDisconnect.\n");
#endif

SQLDisconnect((p_Connections + iConnIndex)->hdbc);
#ifdef _DEBUG

_CrtDbgReport(_CRT_WARN, NULL, 0, NULL, "Executing SQLFreeHandle
for hdbc.\n");
#endif

SQLFreeHandle(SQL_HANDLE_DBC, (p_Connections + iConnIndex)->hdbc );
                                                        (p_Connections +
iConnIndex)->hdbc = SQL_NULL_HDBC;
                                        }

                        }

                        free(p_Connections);
                }
                p_Connections = NULL;

                return;
}

////////////////////////////////////////////////////////////////////////
// Used to determine whether the DLL can be unloaded by OLE

STDAPI DllCanUnloadNow(void)
{
   return (_Module.GetLockCount()==0) ? S_OK : S_FALSE;
}

////////////////////////////////////////////////////////////////////////
// Returns a class factory to create an object of the requested type

STDAPI DllGetClassObject(REFCLSID rclsid, REFIID riid, LPVOID* ppv)
{
   return _Module.GetClassObject(rclsid, riid, ppv);
}

////////////////////////////////////////////////////////////////////////
// DllRegisterServer - Adds entries to the system registry

STDAPI DllRegisterServer(void)
{
   // registers object, typelib and all interfaces in typelib
   return _Module.RegisterServer(TRUE);
}

////////////////////////////////////////////////////////////////////////
// DllUnregisterServer - Removes entries from the system registry

STDAPI DllUnregisterServer(void)
{
   return _Module.UnregisterServer(TRUE);
}
; ExecuteDll.def : Declares the module parameters.

LIBRARY      "ExecuteDll.DLL"

EXPORTS
                DllCanUnloadNow     @1 PRIVATE
                DllGetClassObject   @2 PRIVATE
                DllRegisterServer   @3 PRIVATE
                DllUnregisterServer         @4 PRIVATE
// ExecuteDll.idl : IDL source for ExecuteDll.dll
//

// This file will be processed by the MIDL tool to
// produce the type library (ExecuteDll.tlb) and marshalling code.

import "oaidl.idl";
import "ocidl.idl";
                typedef
                [
                                uuid(0AC32070-B0DB-11d2-BC0D-00A0C90D2CA5),
```

```
                helpstring("ExecutionTypes"),
        ]

        enum ExecutionType
        {
                [helpstring("Shell")]   execODBC  = 0x0001,
                [helpstring("ODBC")]    execShell = 0x0002
        } ExecutionType;

        typedef
        [

uuid(D4A4B9B0-BAE3-11d2-BC0F-00A0C90D2CA5),
                helpstring("Run Status Values"),
        ]

        enum InstanceStatus
        {
                [helpstring("Disabled")] gintDisabled= 0x0001,
                [helpstring("Pending")]          gintPending
= 0x0002,
                [helpstring("Running")]          gintRunning
= 0x0003,
                [helpstring("Complete")] gintComplete          =
0x0004,
                [helpstring("Failed")]           gintFailed
= 0x0005,
                [helpstring("Aborted")]          gintAborted
= 0x0006
        } InstanceStatus;


[
        uuid(551AC525-AB1C-11D2-BC0C-00A0C90D2CA5),
        version(1.0),
        helpstring("ExecuteDll 1.0 Type Library")
]
library EXECUTEDLLLib
{
        importlib("stdole32.tlb");
        importlib("stdole2.tlb");

        [

uuid(551AC532-AB1C-11D2-BC0C-00A0C90D2CA5),
                helpstring("_IExecuteEvents Interface")
        ]
        dispinterface _IExecuteEvents
        {
                properties:
                methods:
                [id(1), helpstring("method Start")] void Start([in]
CURRENCY StartTime);
                [id(2), helpstring("method Complete")] void
Complete([in] CURRENCY EndTime, [in] long Elapsed);
        };
        [
                object,

uuid(551AC531-AB1C-11D2-BC0C-00A0C90D2CA5),
                dual,
                helpstring("IExecute Interface"),
                pointer_default(unique)
        ]
        interface IExecute : IDispatch
        {
                [id(1), helpstring("method DoExecute")] HRESULT
DoExecute([in] BSTR szCommand, [in] BSTR szExecutionDtls, [in]
ExecutionType ExecMethod, [in] BOOL bNoCount, [in] BOOL bNoExecute, [in]
BOOL bParseOnly, [in] BOOL bQuotedIds, [in] BOOL bAnsiNulls, [in] BOOL
bShowQP, [in] BOOL bStatsTime, [in] BOOL bStatsIO, [in] long lRowCount,
[in] long lQueryTmout, [in] BSTR szConnection);
                [propget, id(2), helpstring("property StepStatus")]
HRESULT StepStatus([out, retval] InstanceStatus *pVal);
                [id(3), helpstring("method Abort")] HRESULT Abort();
                [id(4), helpstring("method WriteError")] HRESULT
WriteError(BSTR szMsg);
                [propput, id(5), helpstring("property OutputFile")]
HRESULT OutputFile([in] BSTR newVal);
                [propput, id(6), helpstring("property ErrorFile")]
HRESULT ErrorFile([in] BSTR newVal);
        };
        [

uuid(2EFC198E-AA8D-11D2-BC0C-00A0C90D2CA5),
                helpstring("Execute Class")
        ]
        coclass Execute
        {
                [default] interface IExecute;
                [default, source] dispinterface _IExecuteEvents;
        };
};
#ifndef _EXECUTEDLLCP_H_
#define _EXECUTEDLLCP_H_




template <class T>
class CProxy_IExecuteEvents : public IConnectionPointImpl<T,
&DIID__IExecuteEvents, CComDynamicUnkArray>
{
        //Warning this class may be recreated by the wizard.
public:
        VOID Fire_Start(CY StartTime)
        {
                T* pT = static_cast<T*>(this);
                int nConnectionIndex;
                CComVariant* pvars = new CComVariant[1];
                int nConnections = m_vec.GetSize();

                for (nConnectionIndex = 0; nConnectionIndex <
nConnections; nConnectionIndex++)
                {
                        pT->Lock();
                        CComPtr<IUnknown> sp =
m_vec.GetAt(nConnectionIndex);
                        pT->Unlock();
                        IDispatch* pDispatch =
reinterpret_cast<IDispatch*>(sp.p);
                        if (pDispatch != NULL)
                        {
                                pvars[0] = StartTime;
                                DISPPARAMS disp = {
pvars, NULL, 1, 0 };
                                pDispatch->Invoke(0x1,
IID_NULL, LOCALE_USER_DEFAULT, DISPATCH_METHOD, &disp,
NULL, NULL, NULL);
                        }
                }
                delete[] pvars;

        }
        VOID Fire_Complete(CY EndTime, LONG Elapsed)
        {
                T* pT = static_cast<T*>(this);
                int nConnectionIndex;
                CComVariant* pvars = new CComVariant[2];
                int nConnections = m_vec.GetSize();

                for (nConnectionIndex = 0; nConnectionIndex <
nConnections; nConnectionIndex++)
                {
                        pT->Lock();
                        CComPtr<IUnknown> sp =
m_vec.GetAt(nConnectionIndex);
                        pT->Unlock();
                        IDispatch* pDispatch =
reinterpret_cast<IDispatch*>(sp.p);
                        if (pDispatch != NULL)
                        {
```

```
                                        pvars[1] = EndTime;
                                        pvars[0] = Elapsed;
                                        DISPPARAMS disp = {
pvars, NULL, 2, 0 };
                                        pDispatch->Invoke(0x2,
IID_NULL, LOCALE_USER_DEFAULT, DISPATCH_METHOD, &disp,
NULL, NULL, NULL);
                                    }
                                }
                                delete[] pvars;

                        }
};
#endif
Attribute VB_Name = "FileCommon"
'  FILE:      FileCommon.bas
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:    This module contains common functionality to display
'             the File Open dialog.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "FileCommon."

Private Enum EOpenFile
    OFN_OVERWRITEPROMPT = &H2
    OFN_HIDEREADONLY = &H4
    OFN_FILEMUSTEXIST = &H1000
    OFN_EXPLORER = &H80000
End Enum

' The locations for the different output files are presented to
' the user in a list box. These constants are used while loading the
' data and while reading the data from the list box.
' These constants also represent the different file types that are
' displayed to the user in File Open dialogs
Public Enum gFileTypes
    gintOutputFile = 0
'    gintLogFile = 1
    gintErrorFile
    gintStepTextFile
    gintOutputCompareFile
    gintDBFile
    gintDBFileNew
    gintImportFile
    gintExportFile
End Enum

Public Const gsSqlFileSuffix= ".sql"
Public Const gsCmdFileSuffix= ".cmd"

Public Const gsOutputFileSuffix= ".out"
Public Const gstrLogFileSuffix= ".log"
Public Const gsErrorFileSuffix= ".err"
Public Function BrowseDBFile() As String
    ' Prompts the user for a database file with the workspace information
    ' Call CallFileDialog to display the open file dialog
    BrowseDBFile = CallFileDialog(gintDBFile)

End Function
Public Function CallFileDialog(intFileTypeAs Integer, _
        Optional ByVal strDefaultFile As String = gstrEmptyString) As String
    ' This function initializes the values of the filter property,
    ' the dialog title and flags for the File Open dialog depending
    ' on the FileType passed in
    ' It then calls ShowFileOpenDialog to set these properties and
    ' display the File Open dialog to the user

    ' All the properties used by the File Open dialog are defined
    ' as constants in this function and passed to ShowFileOpenDialog
    ' as parameters. So if any of the dialog properties need to be
    ' modified, these constants are what need to be changed
    Const s_DLG_TITLE_OPEN = "Open"
    Const s_DLG_TITLE_NEW = "New"
    Const s_DLG_TITLE_IMPORT = "Import From"
    Const s_DLG_TITLE_EXPORT = "Export To"

    Const mlng_FILE_STEP_TEXT_FLAGS= OFN_EXPLORER Or
OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY
    Const mlng_FILE_OUTPUT_COMPARE_FLAGS=
mlng_FILE_STEP_TEXT_FLAGS
    Const mlng_FILE_DB_FLAGS= mlng_FILE_STEP_TEXT_FLAGS
    Const mlng_FILE_OUTPUT_FLAGS= OFN_EXPLORER Or
OFN_HIDEREADONLY Or OFN_OVERWRITEPROMPT
    Const mlng_FILE_LOG_FLAGS= mlng_FILE_OUTPUT_FLAGS
    Const mlng_FILE_ERROR_FLAGS= mlng_FILE_OUTPUT_FLAGS
    Const mlng_FILE_DB_NEW_FLAGS= mlng_FILE_OUTPUT_FLAGS

    Const mstr_FILE_ALL_FILTER= "|All Files (*.*)|*.*"
    Const mstr_FILE_STEP_TEXT_FILTER= "Query Files (*" & gsSqlFileSuffix
& _
        ")|*" & gsSqlFileSuffix & "|Command Script Files (*" & gsCmdFileSuffix
& _
        ")|*" & gsCmdFileSuffix
    Const mstr_FILE_OUTPUT_COMPARE_FILTER= "Text Files (*.txt)|*.txt"
    Const mstr_FILE_OUTPUT_FILTER= "Output Files (*.out)|*.out"
    Const mstr_FILE_LOG_FILTER= "Log Files (*.log)|*.log"
    Const mstr_FILE_ERROR_FILTER= "Error Files (*.err)|*.err"
    Const mstr_FILE_DB_FILTER= "Stepmaster Workspace Files (*" &
gsDefDBFileExt & ")|*" & gsDefDBFileExt

    Dim strFileName As String

    On Error GoTo CallFileDialogErr

    Select Case intFileType
        Case gintStepTextFile
            strFileName = ShowFileOpenDialog(_
                mstr_FILE_STEP_TEXT_FILTER& mstr_FILE_ALL_FILTER,_
                s_DLG_TITLE_OPEN, _
                mlng_FILE_STEP_TEXT_FLAGS,_
                strDefaultFile)

        Case gintOutputCompareFile
            strFileName = ShowFileOpenDialog(_
                mstr_FILE_OUTPUT_COMPARE_FILTER&
mstr_FILE_ALL_FILTER,_
                s_DLG_TITLE_OPEN, _
                mlng_FILE_OUTPUT_COMPARE_FLAGS,_
                strDefaultFile)

        Case gintOutputFile
            strFileName = ShowFileOpenDialog(_
                mstr_FILE_OUTPUT_FILTER& mstr_FILE_ALL_FILTER,_
                s_DLG_TITLE_OPEN, _
                mlng_FILE_OUTPUT_FLAGS,_
                strDefaultFile)

'        Case gintLogFile
'            strFileName = ShowFileOpenDialog(_
'                mstr_FILE_LOG_FILTER& mstr_FILE_ALL_FILTER,_
'                s_DLG_TITLE_OPEN, _
'                mlng_FILE_LOG_FLAGS,_
'                strDefaultFile)
'
        Case gintErrorFile
            strFileName = ShowFileOpenDialog(_
                mstr_FILE_ERROR_FILTER& mstr_FILE_ALL_FILTER,_
                s_DLG_TITLE_OPEN, _
                mlng_FILE_ERROR_FLAGS,_
                strDefaultFile)

        Case gintDBFile
            strFileName = ShowFileOpenDialog(_
                mstr_FILE_DB_FILTER& mstr_FILE_ALL_FILTER,_
                s_DLG_TITLE_OPEN, _
                mlng_FILE_DB_FLAGS,_
                strDefaultFile)
```

```
        Case gintDBFileNew
            strFileName = ShowFileOpenDialog(_
                mstr_FILE_DB_FILTER& mstr_FILE_ALL_FILTER,_
                s_DLG_TITLE_NEW, _
                mlng_FILE_DB_NEW_FLAGS,_
                strDefaultFile)

        Case gintImportFile
            strFileName = ShowFileOpenDialog(_
                mstr_FILE_DB_FILTER& mstr_FILE_ALL_FILTER,_
                s_DLG_TITLE_IMPORT, _
                mlng_FILE_DB_FLAGS,_
                strDefaultFile)

        Case gintExportFile
            strFileName = ShowFileOpenDialog(_
                mstr_FILE_DB_FILTER& mstr_FILE_ALL_FILTER,_
                s_DLG_TITLE_EXPORT, _
                mlng_FILE_DB_FLAGS,_
                strDefaultFile)

        Case Else
            BugAssert True, "Incorrect file type passed in."
            ' Default processing will be for the output file
            strFileName = ShowFileOpenDialog(_
                mstr_FILE_OUTPUT_FILTER& mstr_FILE_ALL_FILTER,_
                s_DLG_TITLE_OPEN, _
                mlng_FILE_OUTPUT_FLAGS,_
                strDefaultFile)

    End Select
    CallFileDialog = strFileName

    Exit Function

CallFileDialogErr:
    CallFileDialog = gstrEmptyString
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "CallFileDialog"
    Call ShowError(errBrowseFailed)

End Function
Attribute VB_Name = "IteratorCommon"
'  FILE:      IteratorCommon.bas
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:    Contains functionality common across StepMaster and
'              SMRunOnly, pertaining to iterators
'             Specifically, functions to read iterators records
'             in the workspace, load them in an array and so on.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "IteratorCommon."

Public Const gintMinIteratorSequence As Integer = 0

Public Sub RangeComplete(vntIterators As Variant)
    ' This is a debug procedure
    ' Checks if the from, to and step values are present in
    ' the array

    Dim bReset As Byte
    Dim bShift As Byte
    Dim lngIndex As Long

    ' Set the three lowest order bits to 1
    bReset = 7

    BugAssert IsArray(vntIterators) And Not IsEmpty(vntIterators), _
        "Iterators not specified!"

    For lngIndex = LBound(vntIterators) To _
            UBound(vntIterators)
        bShift = 1
        bShift = bShift * (2 ^ (vntIterators(lngIndex).IteratorType - 1))

        bReset = bReset Xor bShift
    Next lngIndex

    ' Assert that all the elements are present
    BugAssert bReset = 0, "Range not completely specified!"

End Sub
Public Sub LoadIteratorsForWsp(cStepsCol As cArrSteps, _
        ByVal lngWorkspaceId As Long, rstStepsInWsp As Recordset)
    ' Initializes the step records in with all the iterator
    ' values for each step

    Dim recIterators As Recordset

    On Error GoTo LoadIteratorsForWspErr

#If QUERY_ALL Then
    Dim dtStart As Date

    dtStart = Now
    Set recIterators = ReadWspIterators(lngWorkspaceId)

    Call LoadIteratorsArray(cStepsCol, recIterators)

    recIterators.Close

    BugMessage "QueryAll Read + load took: " & CStr(DateDiff("s", dtStart,
Now))

#Else
    Dim dtStart As Date
    Dim qyIt As DAO.QueryDef
    Dim sSql As String

    dtStart = Now
    If rstStepsInWsp.RecordCount = 0 Then
        Exit Sub
    End If

    ' This method has the advantage that if the steps are queried right, everything
else follows
    sSql = "Select step_id, version_no, type, iterator_value, " & _
        " sequence_no " & _
        " from iterator_values " & _
        " where step_id = [s_id] " & _
        " and version_no = [ver_no] "

    ' Order the iterators by sequence within a step
    sSql = sSql & " order by sequence_no "

    Set qyIt = dbsAttTool.CreateQueryDef(gstrEmptyString,sSql)
    rstStepsInWsp.MoveFirst

    While Not rstStepsInWsp.EOF

        qyIt.Parameters("s_id").Value = rstStepsInWsp!step_id
        qyIt.Parameters("ver_no").Value = rstStepsInWsp!version_no

        Set recIterators = qyIt.OpenRecordset(dbOpenSnapshot)

        Call LoadIteratorsArray(cStepsCol, recIterators)
        recIterators.Close

        rstStepsInWsp.MoveNext
    Wend

    qyIt.Close

    BugMessage "Query step at a time Read + load took: " & CStr(DateDiff("s",
dtStart, Now))

#End If
```

```
    Exit Sub

LoadIteratorsForWspErr:
    LogErrors Errors
    gstrSource = mstrModuleName & "LoadIteratorsForWsp"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadRsInArrayFailed, _
          gstrSource, _
          LoadResString(errLoadRsInArrayFailed)

End Sub
Private Function ReadWspIterators(ByVal lngWorkspaceId As Long) As
Recordset

    ' This function will return a recordset that is populated
    ' with the iterators for all the steps in a given workspace

    Dim recIterators As Recordset
    Dim qyIt As DAO.QueryDef
    Dim strSql As String

    On Error GoTo ReadWspIteratorsErr
    gstrSource = mstrModuleName & "ReadWspIterators"

    strSql = "Select i.step_id, i.version_no, " & _
       " i.type, i.iterator_value, " & _
       " i.sequence_no " & _
       " from iterator_values i, att_steps a " & _
       " where i.step_id = a.step_id " & _
       " and i.version_no = a.version_no " & _
       " and a.workspace_id = [w_id] " & _
       " and a.archived_flag = [archived] "

    ' Find the highest X-component of the version number
    strSql = strSql & " AND cint( mid( a.version_no, 1, instr( a.version_no, " &
gstrDQ & gstrVerSeparator & gstrDQ & " ) - 1 ) ) = " & _
       " ( select max( cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) ) " & _
       " from att_steps AS d " & _
       " WHERE a.step_id = d.step_id ) "

    ' Find the highest Y-component of the version number for the highest
X-component
    strSql = strSql & " AND cint( mid( a.version_no, instr( a.version_no, " &
gstrDQ & gstrVerSeparator & gstrDQ & " ) + 1 ) ) = " & _
       " ( select max( cint( mid( version_no, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) + 1 ) ) ) " & _
       " from att_steps AS b " & _
       " Where a.step_id = b.step_id " & _
       " AND cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) = " & _
       " ( select max( cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) ) " & _
       " from att_steps AS c " & _
       " WHERE a.step_id = c.step_id ) ) "


    ' Order the iterators by sequence within a step
    strSql = strSql & " order by i.step_id, i.sequence_no "

    Set qyIt = dbsAttTool.CreateQueryDef(gstrEmptyString, strSql)
    qyIt.Parameters("w_id").Value = lngWorkspaceId
    qyIt.Parameters("archived").Value = False

    Set recIterators = qyIt.OpenRecordset(dbOpenSnapshot)

    qyIt.Close
    Set ReadWspIterators = recIterators

    Exit Function

ReadWspIteratorsErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errReadDataFailed, _
          gstrSource, LoadResString(errReadDataFailed)

End Function
Private Sub LoadIteratorsArray(cStepsCol As cArrSteps, _
       recIterators As Recordset)
    ' Initializes the step records with the iterators for
    ' the step

    Dim cNewIt As cIterator
    Dim cStepRec As cStep
    Dim lngStepId As Long

    On Error GoTo LoadIteratorsArrayErr
    gstrSource = mstrModuleName & "LoadIteratorsArray"

    If recIterators.RecordCount = 0 Then
       Exit Sub
    End If

    recIterators.MoveFirst
    While Not recIterators.EOF
       Set cNewIt = New cIterator

       lngStepId = CLng(ErrorOnNullField(recIterators,"step_id"))
       If Not cStepRec Is Nothing Then
          If cStepRec.StepId <> lngStepId Then
             Set cStepRec = cStepsCol.QueryStep(lngStepId)
          End If
       Else
          Set cStepRec = cStepsCol.QueryStep(lngStepId)
       End If

       ' Initialize iterator values
       cNewIt.IteratorType = CInt(ErrorOnNullField(recIterators,"type"))
       cNewIt.Value = CStr(ErrorOnNullField(recIterators,"iterator_value"))
       cNewIt.SequenceNo = CInt(ErrorOnNullField(recIterators,"sequence_no"))

       ' Add this record to the array of iterators
       cStepRec.LoadIterator cNewIt

       Set cNewIt = Nothing
       recIterators.MoveNext
    Wend

    Exit Sub

LoadIteratorsArrayErr:
    LogErrors Errors
    gstrSource = mstrModuleName & "LoadIteratorsArray"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadRsInArrayFailed, _
          gstrSource, _
          LoadResString(errLoadRsInArrayFailed)

End Sub
Attribute VB_Name = "MsgConfirm"
'  FILE:       MsgConfirm.bas
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    Contains constants for confirmation messages that
'              will be displayed by StepMaster
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' A public enum containing the codes for all the confirmation
' messages that will be used by the project - each of the codes
' has the prefix, con
Public Enum conConfirmMsgCodes
    conWspDelete = 2000
    conSave
    conStopRun
    conSaveConnect
    conSaveDB
End Enum
```

```
' A public enum containing the titles for all the confirmation
' messages that will be used by the project - each of the codes
' has the prefix, cont - most confirmation message codes will
' have a corresponding title code in here
Public Enum conConfirmMsgTitleCodes
    contWspDelete = 3000
    contSave
    contStopRun
    contSaveConnect
    contSaveDB
End Enum
Attribute VB_Name = "OpenFiles"
'  FILE:       OpenFiles.bas
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module holds a list of all files that have been
'              opened by the project. This module is needed since there
'              is no way to share static data between different instances
'              of a class.
'              Many procedure in this module do not do any error handling -
'              this is 'coz it is also used by procedures that log error
'              messages and any error handler will erase the collection
'              of errors!

'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit


' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = ".OpenFiles."

Private mOpenFiles As cNodeCollections

Private Const mstrTempDir As String = "\Temp\"

' The maximum number of temporary files that we can create in a
' session
Private Const mlngMaxFileIndex As Long = 999999
Private Const mstrFileIndexFormat As String = "000000"
Private Const mstrTempFilePrefix As String = "SM"
Private Const mstrTempFileSuffix As String = ".cmd"

Private Const merrFileNotFound As Long = 76
Private Function GetFileHandle(strFileName) As cFileInfo

    Dim lngIndex As Long
    Dim blnFileOpen As Boolean

    If Not mOpenFiles Is Nothing Then

        blnFileOpen = False
        For lngIndex = 0 To mOpenFiles.Count - 1
            If mOpenFiles(lngIndex).FileName= strFileName Then
                blnFileOpen = True
                Exit For
            End If
        Next lngIndex

        If blnFileOpen Then
            Set GetFileHandle = mOpenFiles(lngIndex)
        Else
            Set GetFileHandle = Nothing
        End If
    Else
        Set GetFileHandle = Nothing
    End If

End Function

Private Function GetTempFileDir() As String

    Dim strTempFileDir As String

    On Error GoTo GetTempFileDirErr

    strTempFileDir = gstrProjectPath & mstrTempDir

    If StringEmpty(Dir$(strTempFileDir,vbDirectory)) Then
        MkDir strTempFileDir
    End If

    GetTempFileDir = strTempFileDir

    Exit Function

GetTempFileDirErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "GetTempFileDir"
    On Error GoTo 0
    Err.Raise vbObjectError + errProgramError, gstrSource, _
          LoadResString(errProgramError)

End Function
Public Function MakePathValid(strFileName As String) As String
    ' Checks if the passed in file path is valid

    Dim strFileDir As String
    Dim strTempDir As String
    Dim strTempFile As String
    Dim intPos As Integer
    Dim intStart As Integer

    On Error GoTo MakePathValidErr
    gstrSource = mstrModuleName & "MakePathValid"

    strTempFile = strFileName
    intPos = InstrR(strFileName, gstrFileSeparator)

    If intPos > 0 Then
        strFileDir = Left$(strTempFile, intPos - 1)
        If StringEmpty(Dir$(strFileDir,vbDirectory)) Then
            ' Loop through the entire path starting at the root
            ' since Mkdir can create only one level of sub-directory
            ' at a time
            intStart = InStr(strFileDir, gstrFileSeparator)

            Do While strTempDir <> strFileDir

                If intStart > 0 Then
                    strTempDir = Left$(strFileDir, intStart - 1)
                Else
                    strTempDir = strFileDir
                End If

                If StringEmpty(Dir$(strTempDir,vbDirectory)) Then
                    ' If the specified directory doesn't exist, try to
                    ' create it.
                    MkDir strTempDir
                Else
                    ' The directory exists - go to it's sub-directory
                End If
                intStart = InStr(intStart + 1, strFileDir, gstrFileSeparator)
            Loop

            ' Sanity check
            If StringEmpty(Dir$(strFileDir,vbDirectory)) Then
                ' We were unable to create the file directory
                ShowError errCreateDirectoryFailed, gstrSource, _
                      strFileDir, DoWriteError:=False
                MakePathValid = gstrEmptyString
            Else
                MakePathValid = strTempFile
            End If
        Else
            ' The specified directory exists - we should be able
            ' to create the output file in it
            MakePathValid = strTempFile
        End If
    Else
        ' The user has only specified a filename - VB will try
```

```
        ' to create it in the current directory
        MakePathValid = strTempFile
    End If

    Exit Function

MakePathValidErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "MakePathValid"
    ' Log the filename for debug
    Call WriteError(errInvalidFile, gstrSource, strTempFile)
    On Error GoTo 0
    Err.Raise vbObjectError + errProgramError, gstrSource, _
        LoadResString(errProgramError)

End Function
Public Function OpenFileSM(strFileName As String) As Integer
    Dim intHFile As Integer
    Dim NewFileInfo As cFileInfo

    On Error GoTo OpenFileSMErr
    gstrSource = mstrModuleName & "OpenFileSM"

    If StringEmpty(strFileName)Then
        On Error GoTo 0
        Err.Raise vbObjectError + errInvalidFile, gstrSource, _
            LoadResString(errInvalidFile)
    End If

    If mOpenFiles Is Nothing Then
        Set mOpenFiles = New cNodeCollections
    End If

    Set NewFileInfo = GetFileHandle(strFileName)

    If NewFileInfo Is Nothing Then
        ' The file has not been opened yet

        ' If the filename has not been initialized, do not
        ' attempt to open it
        strFileName = MakePathValid(strFileName)

        If strFileName <> gstrEmptyString Then
            intHFile = FreeFile
            Open strFileName For Output Shared As intHFile

            Set NewFileInfo = New cFileInfo
            NewFileInfo.FileHandle = intHFile
            NewFileInfo.FileName = strFileName
            mOpenFiles.Load NewFileInfo
        Else
            ' Either the directory was invalid or s'thing failed
            ' Display the error to the user instead of trying
            ' to log to the file
            ShowError errInvalidFile, gstrSource, strFileName, _
                DoWriteError:=False
            intHFile = 0
        End If
    Else
        intHFile = NewFileInfo.FileHandle
    End If

    OpenFileSM = intHFile

    Exit Function

OpenFileSMErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    ' The Open command failed for some reason - write an error
    ' and let the calling function handle the error
    ShowError errInvalidFile, gstrSource, strFileName, _
        DoWriteError:=False
    OpenFileSM = 0

End Function
Public Function CreateTempFile() As String
    Dim strTempFileDir As String
    Dim strTempFileName As String

    Static lngLastFileIndex As Long

    On Error GoTo CreateTempFileErr

    strTempFileDir = GetTempFileDir()

    Do
        If lngLastFileIndex = mlngMaxFileIndex Then
            On Error GoTo 0
            Err.Raise vbObjectError + errMaxTempFiles, gstrSource, _
                LoadResString(errMaxTempFiles)
        End If

        lngLastFileIndex = lngLastFileIndex + 1
        strTempFileName = mstrTempFilePrefix & _
            Format$(lngLastFileIndex, mstrFileIndexFormat) & _
            mstrTempFileSuffix

        If Not StringEmpty(Dir$(strTempFileDir & strTempFileName)) Then
            ' Remove any files left over from a previous run,
            ' if they still exist
            Kill strTempFileDir & strTempFileName
        End If

    ' Looping in case the file delete doesn't go through for
    ' some reason
    Loop While Not StringEmpty(Dir$(strTempFileDir & strTempFileName))

    CreateTempFile = GetShortName(strTempFileDir)
    CreateTempFile = CreateTempFile & strTempFileName

    Exit Function

CreateTempFileErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    gstrSource = gstrSource & "CreateTempFile"
    On Error GoTo 0
    Err.Raise vbObjectError + errProgramError, gstrSource, _
        LoadResString(errProgramError)

End Function
Public Sub CloseFileSM(strFileName As String)
    Dim FileToClose As cFileInfo

    If Not mOpenFiles Is Nothing Then

        ' Get the handle to the open file, if it exists
        Set FileToClose = GetFileHandle(strFileName)

        If Not FileToClose Is Nothing Then
            Close FileToClose.FileHandle

            ' Remove the file info from the collection of open files
            mOpenFiles.Unload FileToClose.Position
        End If
    End If

End Sub
Public Sub CloseOpenFiles()
    Dim lIndex As Long

    If Not mOpenFiles Is Nothing Then
        For lIndex = mOpenFiles.Count - 1 To 0
            CloseFileSM (mOpenFiles(lIndex).FileName)
        Next lIndex
    End If

End Sub
Attribute VB_Name = "ParameterCommon"
'  FILE:      ParameterCommon.bas
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
```

```
'
'
'  PURPOSE:    Contains functionality common across StepMaster and
'              SMRunOnly, pertaining to parameters
'              Specifically, functions to load parameter records
'              in an array.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "ParameterCommon."

Public Sub LoadRecordsetInParameterArray(rstWorkSpaceParameters As
Recordset, _
    cParamCol As cArrParameters)

    Dim cNewParameter As cParameter

    On Error GoTo LoadRecordsetInParameterArrayErr

    If rstWorkSpaceParameters.RecordCount = 0 Then
        Exit Sub
    End If

    rstWorkSpaceParameters.MoveFirst
    While Not rstWorkSpaceParameters.EOF

        Set cNewParameter = New cParameter

        ' Initialize parameter values
        cNewParameter.ParameterId = rstWorkSpaceParameters.Fields(0)

        ' Call a procedure to raise an error if mandatory fields are
        ' null.
        cNewParameter.ParameterName = CStr( _
                ErrorOnNullField(rstWorkSpaceParameters, "parameter_name"))
        cNewParameter.ParameterValue = CheckForNullField(_
                rstWorkSpaceParameters, "parameter_value")
        cNewParameter.WorkspaceId = CStr( _
                ErrorOnNullField(rstWorkSpaceParameters,
FLD_ID_WORKSPACE))
        cNewParameter.ParameterType = CStr( _
                ErrorOnNullField(rstWorkSpaceParameters, "parameter_type"))
        cNewParameter.Description = CheckForNullField(_
                rstWorkSpaceParameters, "description")

        cParamCol.Load cNewParameter

        Set cNewParameter = Nothing
        rstWorkSpaceParameters.MoveNext
    Wend

    Exit Sub

LoadRecordsetInParameterArrayErr:
    LogErrors Errors
    gstrSource = mstrModuleName & "LoadRecordsetInParameterArray"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadRsInArrayFailed, gstrSource, _
            LoadResString(errLoadRsInArrayFailed)
End Sub
Attribute VB_Name = "Public"
'  FILE:       Public.bas
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module contains all the public constants for this project
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Public Const gsVersion01 As String = "0.1"
Public Const gsVersion10 As String = "1.0"
Public Const gsVersion21 As String = "2.1"
Public Const gsVersion23 As String = "2.3"
Public Const gsVersion24 As String = "2.4"
Public Const gsVersion241 As String = "2.4.1"
Public Const gsVersion242 As String = "2.4.2"
Public Const gsVersion243 As String = "2.4.3"
Public Const gsVersion25 As String = "2.5"
Public Const gsVersion As String = gsVersion25

' The same form is used for the creation of new nodes and
' updates to existing nodes (where each node can be a parameter,
' global step, etc.) A tag is set on each flag is used to indicate
' whether it is being called in the insert or update mode. The
' constants for these modes are defined below
Public Const gstrInsertMode = "Insert"
Public Const gstrUpdateMode = "Update"
Public Const gstrPropertiesMode = "View"

Public Const gstrEmptyString = ""
Public Const gstrSQ = "'"
Public Const gstrDQ = """"
Public Const gstrVerSeparator = "."
Public Const gstrBlank = " "

' Constants used to indicate type of node being processed
' The constants for the different objects correspond to the
' indexes in the menu control arrays (for both the main andpopup
' menus) that are used to create new objects. That way we can
' use the index passed in by the click event to determine the
' type of node being processed
Public Const gintWorkspace = 1

' Decided to leave it here after some debate over whether it
' actually belongs in the cStep class definition
Public Enum gintStepType
    gintGlobalStep = 3
    gintManagerStep
    gintWorkerStep
End Enum

Public Const gintRunManager = 6
Public Const gintRunWorker = 7

Public Enum gintParameterNodeType
    gintParameter = 8
    gintNodeParamConnection
    gintNodeParamExtension
    gintNodeParamBuiltIn
End Enum

' Leave some constants free for newer types of parameters (?)
Public Const gintConnectionDtl = 15

Public Enum gintLabelNodeType
    gintGlobalsLabel = 21
    gintParameterLabel
    gintParamConnectionLabel
    gintParamExtensionLabel
    gintParamBuiltInLabel
    gintConnDtlLabel
    gintGlobalStepLabel
    gintStepLabel
End Enum

Public Enum ConnectionType
    ConnTypeStatic = 1
    ConnTypeDynamic
End Enum

Public Const giDefaultConnType As Integer = ConnTypeStatic

' The constants defined below are used to identify the different
' tabs. If any more step properties and thereby tabs are added
' to the tabbed dialog on the Step Properties form, they should
' be defined here and accessed in the code only using these
' pre-defined constants
' Note: These constants will mainly be used by the functions that
' initialize, customize and display the Step Properties form
Public Const gintDefinition = 0
```

```
Public Const gintExecution = 1
Public Const gintMgrDefinition = 2
Public Const gintPreExecutionSteps = 3
Public Const gintPostExecutionSteps = 4
Public Const gintFileLocations = 5

' These constants correspond to the index values in the imagelist
' associated with the tree view control. The imagelist contains
' the icons that will be displayed for each node.
Public Enum TreeImages
    gintImageWorkspaceClosed = 1
    gintImageWorkspaceOpen
    gintImageLabelClosed
    gintImageLabelOpen
    gintImageManagerClosed
    gintImageManagerOpen
    gintImageWorkerClosed
    gintImageWorkerOpen
    gintImageGlobalClosed
    gintImageGlobalOpen
    gintImageParameter
    gintImageDrag
    gintImageDrop
    gintImageRun
    gintImagePending
    gintImageStop
    gintImageDisabled
    gintImageAborted
    gintImageFailed
End Enum

' These constants correspond to the index values in the imagelist
' associated with the list view control. The imagelist contains
' the icons that will be displayed for each node.
Public Enum SMListImages
    gintImageWorkspace = 1
    gintImageLabel
    gintImageManager
    gintImageWorker
    gintImageGlobal
    gintImageParameterList
End Enum

' Public variable used to indicate the name of the function
' that raises an error
Public gstrSource As String

' Public instances of the different collections
Public gcParameters As cArrParameters
Public gcSteps As cArrSteps
Public gcConstraints As cArrConstraints
Public gcConnections As cConnections
Public gcConnDtls As cConnDtls

' Public constants for the index values of the different toolbar
' options. Will be used while dynamically enabling/disabling
' these options.
Public Const tbNew = 1
Public Const tbOpen = 2
Public Const tbSave = 3

Public Const tbCut = 5
Public Const tbCopy = 6
Public Const tbPaste = 7
Public Const tbDelete = 8

Public Const tbProperties = 10
Public Const tbRun = 11
Public Const tbStop = 12

' The initial version #
Public Const gstrMinVersion As String = "0.0"
Public Const gstrGlobalParallelism As String = "0"
Public Const gintMinParallelism As Integer = 1
Public Const gintMaxParallelism As Integer = 100

' Constant for the minimum identifier, used for all identifier, viz.
' step, workspace, etc.
Public Const glMinId As Long = 1
Public Const glInvalidId As Long = -1

' A parameter that has a special meaning to Stepmaster
' The system time will be substituted wherever it occurs
' (typically as a part of the error, log ... file names
Public Const gstrTimeStamp As String = "TIMESTAMP"
Public Const gstrEnvVarSeparator = "%"
Public Const gstrFileSeparator = "\"
Public Const gstrUnderscore = "_"

' Constants used by date and time formatting functions
Public Const gsTimeSeparator = ":"
Public Const gsDateSeparator = "-"
Public Const gsMsSeparator = "."
Public Const gsDtFormat = "00"
Public Const gsYearFormat = "0000"
Public Const gsTmFormat = "00"
Public Const gsMSecondFormat = "000"

' Default nothing value for a date variable
Public Const gdtmEmpty As Currency = 0

Public Const FMT_WSP_LOG_FILE As String = "yyyymmdd-hhnnss"

Public gsContCriteria() As String
' Note: Update the initialization of gsExecutionStatus in Initialize() if the
' InstanceStatus values are modified - also the boundary checks
Public gsExecutionStatus() As String

Public Const gsConnTypeStatic As String = "Static"
Public Const gsConnTypeDynamic As String = "Dynamic"

#If RUN_ONLY Then
Public Const gsCaptionRunWsp As String = "Run Workspace"
#End If

' Valid operations on a cNode object
Public Enum Operation
    QueryOp = 1
    InsertOp = 2
    UpdateOp = 3
    DeleteOp = 4
End Enum
Attribute VB_Name = "RunCommon"
'  FILE:      RunCommon.bas
'             Microsoft  TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:    Contains common functions that are used during the execution
'              of a workspace.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = ".RunCommon."


Public Function GetInstanceItValue(cInstanceRec As cInstance) As String

    ' Returns the iterator value for the instance, if an
    ' iterator has been defined for it
    Dim cStepIt As cRunColIt
    Dim cRunIterator As cRunItNode

    On Error GoTo GetInstanceItValueErr

    ' Since we create a dummy instance for Disabled and Pending steps,
    ' doesn't make sense to look at their iterators
    If cInstanceRec.Status <> gintDisabled And cInstanceRec.Status <>
gintPending Then
        Set cStepIt = cInstanceRec.Iterators

        If Not StringEmpty(cInstanceRec.Step.IteratorName) Then
```

```
        BugAssert cStepIt.Count > 0, "Iterator Count is greater " & _
            "than zero for a step that has an iterator defined."
        Set cRunIterator = cStepIt(0)
        BugAssert cRunIterator.IteratorName = cInstanceRec.Step.IteratorName,
_
            "The first iterator in the collection is the " & _
            "one that has been defined for the step."
        If cRunIterator.IteratorName = cInstanceRec.Step.IteratorName Then
          GetInstanceItValue = cRunIterator.Value
        Else
          GetInstanceItValue = gstrEmptyString
        End If
      Else
        GetInstanceItValue = gstrEmptyString
      End If
    Else
      GetInstanceItValue = gstrEmptyString
    End If

    Exit Function

GetInstanceItValueErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    gstrSource = mstrModuleName & "GetInstanceItValue"
    Err.Raise vbObjectError + errProgramError, gstrSource, _
        LoadResString(errProgramError)

End Function
Attribute VB_Name = "RunInstHelper"
'  FILE:      RunInstHelper.bas
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:    This module contains helper procedures that are called by
'              cRunInst.cls
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this class
Private Const mstrModuleName As String = "RunInstHelper."

' Should be equal to the number of steps defined in cRunInst.cls
Public Const glngNumConcurrentProcesses As Long = 16
Public Const gintBitsPerByte = 8
Public Function AnyStepRunning(cFreeSteps As cVectorLng, arrFree() As Byte)
As Boolean

    Dim lngIndex As Long
    Dim intPosInByte As Integer
    Dim lngTemp As Long

    ' Check if there are any running instances to wait for
    If cFreeSteps.Count <> glngNumConcurrentProcesses Then

      ' For every free step, reset the corresponding element
      ' in the byte array to 0
      For lngIndex = 0 To cFreeSteps.Count - 1

        lngTemp = cFreeSteps(lngIndex) \ gintBitsPerByte
        intPosInByte = cFreeSteps(lngIndex) Mod gintBitsPerByte

        arrFree(lngTemp) = arrFree(lngTemp) Xor 2 ^ intPosInByte
      Next lngIndex

      AnyStepRunning = False

      ' Check if we have a non-zero bit in the byte array
      For lngIndex = LBound(arrFree) To UBound(arrFree) Step 1
        If arrFree(lngIndex) <> 0 Then
          ' We are waiting for a step to complete
          AnyStepRunning = True
          Exit For
        End If
      Next lngIndex

    Else
      AnyStepRunning = False
    End If

End Function


Public Function OrderConstraints(vntTempCons() As Variant, _
      intConsType As ConstraintType) As Variant
    ' Returns a variant containing all the constraint records in the order
    ' in which they should be executed

    Dim vntTemp As Variant
    Dim lngOuter As Long
    Dim lngInner As Long
    Dim cTempConstraint As cConstraint
    Dim cConstraints() As cConstraint
    Dim lngConsCount As Long
    Dim lngLbound As Long
    Dim lngUbound As Long
    Dim lngStep As Long

    On Error GoTo OrderConstraintsErr

    If intConsType = gintPreStep Then
      ' Since we are travelling up and we need to execute the constraints
      ' for the top-level steps first, reverse the order that they
      ' have been stored in the array
      lngLbound = UBound(vntTempCons)
      lngUbound = LBound(vntTempCons)
      lngStep = -1
    Else
      lngLbound = LBound(vntTempCons)
      lngUbound = UBound(vntTempCons)
      lngStep = 1
    End If

    lngConsCount = 0

    For lngOuter = lngLbound To lngUbound Step lngStep
      vntTemp = vntTempCons(lngOuter)

      If Not IsEmpty(vntTemp) Then
        ' Each of the elements is an array
        For lngInner = LBound(vntTemp) To UBound(vntTemp) Step 1
          If Not IsEmpty(vntTemp(lngInner)) Then
            Set cTempConstraint = vntTemp(lngInner)

            If Not cTempConstraint Is Nothing Then
              ReDim Preserve cConstraints(lngConsCount)
              Set cConstraints(lngConsCount) = cTempConstraint
              lngConsCount = lngConsCount + 1
            End If
          End If
        Next lngInner
      End If
    Next lngOuter

    ' Set the return value of the function to the array of
    ' constraints that has been built above
    If lngConsCount = 0 Then
      OrderConstraints = Empty
    Else
      OrderConstraints = cConstraints()
    End If

    Exit Function

OrderConstraintsErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errExecInstanceFailed, _
        mstrModuleName, LoadResString(errExecInstanceFailed)
```

```
End Function
Attribute VB_Name = "ShellSM"
' FILE:      ShellSM.bas
'           Microsoft TPC-H Kit Ver. 1.00
'           Copyright Microsoft, 1999
'           All Rights Reserved
'
'
' PURPOSE:   This module contains a function that creates a process and
'           waits for it to complete.
' Contact:   Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Public Function SyncShell(CommandLine As String, Optional Timeout As Long,
_
   Optional WaitForInputIdle As Boolean) As Boolean

   Dim proc As PROCESS_INFORMATION
   Dim Start As STARTUPINFO
   Dim ret As Long
   Dim nMilliseconds As Long

   BugMessage "Executing: " & CommandLine
   If Timeout > 0 Then
      nMilliseconds = Timeout
   Else
      nMilliseconds = INFINITE
   End If

   'Initialize the STARTUPINFO structure:
   Start.cb = Len(Start)
   Start.dwFlags = STARTF_USESHOWWINDOW
   Start.wShowWindow = SW_SHOWMINNOACTIVE

   'Start the shelled application:
   CreateProcessA 0&, CommandLine, 0&, 0&, 1&, _
      NORMAL_PRIORITY_CLASS, 0&, 0&, Start, proc

   If WaitForInputIdle Then
      'Wait for the shelled application to finish setting up its UI:
      ret = InputIdle(proc.hProcess, nMilliseconds)
   Else
      'Wait for the shelled application to terminate:
      ret = WaitForSingleObject(proc.hProcess, nMilliseconds)
   End If

   CloseHandle proc.hProcess

   'Return True if the application finished. Otherwise it timed out or erred.
   SyncShell = (ret = WAIT_OBJECT_0)
End Function
Attribute VB_Name = "SMErr"
' FILE:      SMErr.bas
'           Microsoft TPC-H Kit Ver. 1.00
'           Copyright Microsoft, 1999
'           All Rights Reserved
'
'
' PURPOSE:   This module contains error code for all the errors that are
'           raised by StepMaster.
' Contact:   Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' A public enum containing the codes for all the error
' messages that will be displayed by the project - each
' of the codes has the prefix, err
Public Enum errErrorConstants
   errParameterIdInvalid = 1000
   errParameterNameMandatory
   errParameterInsertFailed
   errStepLabelOrTextOrFileRequired
   errMandatoryNodeTextMissing
   errParameterUpdateFailed
   errDupConnDtlName
   errDummy14
   errContCriteriaMandatory
   errContCriteriaNullForGlobal
   errContCriteriaInvalid = 1010
   errParamSeparatorMissing
   errStepTextOrTextFileMandatory
   errStepTextOrFile
   errEnabledFlagFalseForGlobal
   errEnabledFlagLetFailed
   errDegParallelismNullForGlobal
   errInvalidDegParallelism
   errExecutionMechanismInvalid
   errExecutionMechanismLetFailed
   errStepLevelNull = 1020
   errStepLevelZeroForGlobal
   errStepLevelLetFailed
   errRowCountNumeric
   errConnNameInvalid
   errTimeoutNumeric
   errResetConnPropertiesFailed
   errFailureThresholdNumeric
   errConnectionUpdateFailed
   errGlobalRunMethodMandatory
   errGlobalRunMethodNull = 1030
   errGlobalRunMethodInvalid
   errGlobalRunMethodLetFailed
   errNoTrueOption
   errGetOptionFailed
   errSetEnabled
   errStepLabelTextAndFileNull
   errInvalidNodeType
   errSetOptionFailed
   errQueryParameterFailed
   errParamNotFound = 1040
   errQueryStepFailed
   errStepNotFound
   errReadFromScreenFailed
   errCopyPropertiesToFormFailed
   errMandatoryFieldNull
   errUnableToCheckNull
   errUpgradeFailed
   errFindStepSequenceFailed
   errFindParentStepIdFailed
   errCircularReference = 1050
   errAddStepFailed
   errModifyStepFailed
   errDeleteColumnFailed
   errFindPositionFailed
   errDeleteStepFailed
   errRunExistsForStepFailed
   errCreateNewParentFailed
   errInsertNewStepVersionFailed
   errCreateNewStepFailed
   errDuplicateParameterName = 1060
   errCheckDupParameterNameFailed
   errConstraintTypeInvalid
   errConstraintTypeLetFailed
   errAddConstraintFailed
   errWorkspaceIdMandatory
   errInvalidWorkspaceData
   errGetWorkspaceDetailsFailed
   errNoWorkspaceLoaded
   errWorkspaceAlreadyOpen
   errDuplicateWorkspaceName = 1070
   errWorkspaceNameMandatory
   errWorkspaceNameSetFailed
   errWorkspaceIdInvalid
   errWorkspaceIdSetFailed
   errWorkspaceInsertFailed
   errWorkspaceDeleteFailed
   errWorkspaceUpdateFailed
   errInvalidFile
   errCheckWorkspaceOpenFailed
   errWriteFailed = 1080
   errDeleteParameterRecordFailed
   errUnableToLogOutput
   errDeleteDBRecordFailed
   errRunExistsForWorkspaceFailed
   errClearHistoryFailed
   errCreateDBFailed
```

errImportWspFailed
errStepModifyFailed
errStepDeleteFailed
errDummy3 = 1090
errUnableToGetWorkspace
errInvalidNode
errUnableToRemoveSubtree
errSetFileNameFailed
errStepTypeInvalid
errObjectMandatory
errBuiltInUpdateOnly
errInvalidStep
errTypeOfStepFailed
errGetParentKeyFailed = 1100
errLabelTextAndFileCheckFailed
errStepTextAndFileNull
errTextOrFileCheckFailed
errParentStepManager
errDeleteSubStepsFailed
errWorkspaceNameDuplicateFailed
errNewConstraintVersionFailed
errDeleteStepConstraintsFailed
errOldVersionMandatory
errLoadConstraintsInListFailed = 1110
errLoadGlobalStepsFailed
errDeleteConstraintFailed
errUpdateConstraintFailed
errConstraintIdInvalid
errConstraintIdSetFailed
errGlobalStepIdInvalid
errGlobalStepIdSetFailed
errUpdateVersionFailed
errQueryAdjacentConsFailed
errConstraintNotFound = 1120
errQueryConstraintFailed
errSetDBBeforeLoad
errLoadDataFailed
errLoadRsInArrayFailed
errConstraintsForStepFailed
errPreConstraintsForStepFailed
errPostConstraintsForStepFailed
errExecuteConstraintMethodFailed
errIdOrKeyMandatory
errInListFailed = 1130
errUnableToWriteChanges
errQuickSortFailed
errCheckParentValidFailed
errLogErrorFailed
errCopyListFailed
errConnected
errVersionMismatch
errStepNodeFailed
errWorkspaceSelectedFailed
errIdentifierSelectedFailed = 1140
errCheckForNullFieldFailed
errInstanceInUse
errSetVisiblePropertyFailed
errExportWspFailed
errMakeKeyValidFailed
errDummy16
errRunApplicationFailed
errStepLabelUnique
errDeleteSingleFile
errMakeIdentifierValidFailed = 1150
errTypeOfNodeFailed
errConstraintCommandFailed
errOpenDbFailed
errInsertNewConstraintsFailed
errLoadPostExecutionStepsFailed
errLoadPreExecutionStepsFailed
errCreateNewNodeFailed
errDeleteNodeFailed
errDisplayPopupFailed
errDisplayPropertiesFailed = 1160
errUnableToCreateNewObject
errDiffFailed
errLoadWorkspaceFailed
errTerminateProcessFailed
errCompareFailed
errCreateConnectionFailed
errShowFormFailed
errAbortFailed
errDeleteParameterFailed
errUpdateViewFailed = 1170
errParameterNewFailed
errCopyNodeFailed
errCutNodeFailed
errCheckObjectValidFailed
errDeleteViewNodeFailed
errMainFailed
errNewStepFailed
errProcessStepModifyFailed
errCustomizeStepFormFailed
errInitializeStepFormFailed= 1180
errInsertStepFailed
errIncVersionYFailed
errIncVersionXFailed
errShowCreateStepFormFailed
errShowStepFormFailed
errStepNewFailed
errUnableToApplyChanges
errUnableToCommitChanges
errGetStepNodeTextFailed
errSelectGlobalRunMethodFailed = 1190
errConnectionNameMandatory
errUpdateStepFailed
errBrowseFailed
errDummy4
errDummy1
errDummy2
errDummy
errUnableToPreviewFile
errCopyWorkspaceFailed
errCopyParameterFailed = 1200
errGetStepTypeAndPositionFailed
errCopyStepFailed
errMandatoryParameterMissing
errDeleteWorkspaceRecordsFailed
errCreateDirectoryFailed
errConfirmDeleteOrMoveFailed
errCreateWorkspaceFailed
errTypeOfObjectFailed
errCreateNodeFailed
errCreateParameterFailed = 1210
errInsertParameterFailed
errCreateStepFailed
errNoConstraintsCreated
errCopyFailed
errCloneFailed
errCloneGlobalFailed
errCloneWorkerFailed
errCloneManagerFailed
errLetStepTypeFailed
errUnableToCloseWorkspace = 1220
errUnableToModifyWorkspace
errUnableToCreateWorkspace
errAddArrayElementFailed
errUpdateSequenceFailed
errCannotCopySubSteps
errSubStepsFailed
errModifyInArrayFailed
errUpdateParentVersionFailed
errGetNodeTextFailed
errAddToArrayFailed = 1230
errDeleteFromArrayFailed
errQueryIndexFailed
errCreateNewConstraintVersionFailed
errGetRootNodeFailed
errPopulateWspDetailsFailed
errLoadRsInTreeFailed
errAddNodeToTreeFailed
errMaxTempFiles
errMoveFailed
errRootNodeKeyInvalid = 1240
errNextNodeFailed
errBranchWillMove

```
errMoveBranchInvalid
errCreateIdRecordsetFailed
errIdentifierColumnFailed
errGetIdentifierFailed
errGetStepTypeFailed
errUpdateConstraintSeqFailed
errDelParamsInWspFailed
errDuplicateConnectionName = 1250
errOpenWorkspaceFailed
errShowWorkspaceNewFailed
errShowWorkspaceModifyFailed
errPopulateListFailed
errExploreNodeFailed
errInitializeListNodeFailed
errMakeListColumnsFailed
errRefreshViewFailed
errExploreFailed
errCollapseNodeFailed = 1260
errUnableToProcessListViewClick
errSetEnabledForStepFailed
errDisplayStepFormFailed
errSetEnabledPropertyFailed
errInvalidDB
errDeleteConnectionFailed
errInvalidOperation
errLetOperationFailed
errIdGetFailed
errCommitFailed = 1270
errSaveParametersInWspFailed
errDeleteArrayElementFailed
errSaveWorkspaceFailed
errInitializeFailed
errLoadInArrayFailed
errSaveStepsInWspFailed
errCommitStepFailed
errStepIdGetFailed
errUnloadFromArrayFailed
errValidateFailed = 1280
errTextEnteredFailed
errStepLabelMandatory
errTextAndFileNullForManager
errFailureDetailsNullForMgr
errSetTabOrderFailed
errSaveWspConstraintsFailed
errCommitConstraintFailed
errUnloadStepConstraintsFailed
errUnableToModifyMenu
errConfirmFailed = 1290
errInitSubItemsFailed
errUpdateListNodeFailed
errAddNodeFailed
errLoadListNodeFailed
errAddListNodeFailed
errExecutionFailed
errSetListViewStyleFailed
errSetCheckedFailed
errGetCheckedFailed
errUnableToProcessListViewDblClick = 1300
errDefaultPosition
errShellFailed
errOpenFileFailed
errSetTBar97Failed
errConnectFailed
errApiFailed
errRegEntryInvalid
errParseStringFailed
errConstraintsForWspFailed
errPostConstraintsForWspFailed = 1310
errPreConstraintsForWspFailed
errLoadWspPostExecStepsFailed
errLoadWspPreExecStepsFailed
errLoadConstraintsOnFormFailed
errQueryFailed
errPasteNodeFailed
errShowAllWorkspacesFailed
errMakeFieldValidFailed
errInitializeTree
errRootNodeFailed = 1320
   errDirectionInvalid
   errUnableToDetListProperty
   errUnableToGetListData
   errItemNotFound
   errItemDoesNotExist
   errParamNameInvalid
   errGetParamValueFailed
   errSubValuesFailed
   errStringOpFailed
   errReadWorkspaceDataFailed = 1330
   errUpdateRunDataFailed
   errProgramError
   errUnableToOpenFile
   errLoadRunDataFailed
   errExecuteODBCCommandFailed
   errRunWorkspaceFailed
   errExecuteStepFailed
   errUnableToWriteError
   errRunStepFailed
   errSaveChanges = 1340
   errDragDropFailed
   errInvalidParameter
   errAssignParametersFailed
   errLoadLabelsInTreeFailed
   errInstrRFailed
   errInsertIteratorFailed
   errDeleteIteratorFailed
   errTypeInvalid
   errLoadFailed
   errDeleteFailed = 1350
   errModifyFailed
   errIteratorsFailed
   errInsertFailed
   errUpdateFailed
   errDuplicateIterator
   errSaveFailed
   errReadDataFailed
   errUnloadFailed
   errAddFailed
   errExecuteBranchFailed = 1360
   errRangeNumeric
   errRangeInvalid
   errNextStepFailed
   errUpdateDisplayFailed
   errDateToStringFailed
   errGetElapsedTimeFailed
   errMaxProcessesExceeded
   errInvalidProperty
   errInvalidChild
   errCreateInstanceFailed = 1370
   errInvalidForWorker
   errInstanceOpFailed
   errNavInstancesFailed
   errIterateFailed
   errExecInstanceFailed
   errDupIterator
End Enum

#pragma once

// ODBC-specific includes
#define DBNTWIN32
#include <sqltypes.h>
#include <sql.h>
#include <sqlext.h>

#define CONNECTION_NAME_LEN 256        // connection name length

typedef struct _SM_Connection_Info
{
        char        szConnectionName[CONNECTION_NAME_LEN];
        HDBC        hdbc;
        BOOL        bInUse;
} SM_Connection_Info;
// SMTime.cpp : Implementation of DLL Exports.


// Note: Proxy/Stub Information
```

```
//      To build a separate proxy/stub DLL,
//      run nmake -f SMTimeps.mk in the project directory.

#include "stdafx.h"
#include "resource.h"
#include <initguid.h>
#include "SMTime.h"

#include "SMTime_i.c"
#include "SMTimer.h"


CComModule _Module;

BEGIN_OBJECT_MAP(ObjectMap)
OBJECT_ENTRY(CLSID_SMTimer, CSMTimer)
END_OBJECT_MAP()

/////////////////////////////////////////////////////////////////////////////
// DLL Entry Point

extern "C"
BOOL WINAPI DllMain(HINSTANCE hInstance, DWORD dwReason,
LPVOID /*lpReserved*/)
{
   if (dwReason == DLL_PROCESS_ATTACH)
   {
      _Module.Init(ObjectMap, hInstance, &LIBID_SMTIMELib);
      DisableThreadLibraryCalls(hInstance);
   }
   else if (dwReason == DLL_PROCESS_DETACH)
      _Module.Term();
   return TRUE;    // ok
}

/////////////////////////////////////////////////////////////////////////////
// Used to determine whether the DLL can be unloaded by OLE

STDAPI DllCanUnloadNow(void)
{
   return (_Module.GetLockCount()==0) ? S_OK : S_FALSE;
}

/////////////////////////////////////////////////////////////////////////////
// Returns a class factory to create an object of the requested type

STDAPI DllGetClassObject(REFCLSID rclsid, REFIID riid, LPVOID* ppv)
{
   return _Module.GetClassObject(rclsid, riid, ppv);
}

/////////////////////////////////////////////////////////////////////////////
// DllRegisterServer - Adds entries to the system registry

STDAPI DllRegisterServer(void)
{
   // registers object, typelib and all interfaces in typelib
   return _Module.RegisterServer(TRUE);
}

/////////////////////////////////////////////////////////////////////////////
// DllUnregisterServer - Removes entries from the system registry

STDAPI DllUnregisterServer(void)
{
   return _Module.UnregisterServer(TRUE);
}
; SMTime.def : Declares the module parameters.

LIBRARY      "SMTime.DLL"

EXPORTS
                DllCanUnloadNow     @1 PRIVATE
                DllGetClassObject   @2 PRIVATE
                DllRegisterServer   @3 PRIVATE
                DllUnregisterServer         @4 PRIVATE
                Get64BitTime                @5
                SMTime_JulianToTime   @6
```

```
/* this ALWAYS GENERATED file contains the definitions for the interfaces*/


/* File created by MIDL compiler version 5.01.0164*/
/* at Fri Sep 24 11:59:48 1999
 */
/* Compiler settings for
D:\Rst\StepMaster-VB5.0\COMMON\SMTime\SMTime.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32, ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
*/
//@@MIDL_FILE_HEADING( )


/* verify that the <rpcndr.h> version is high enough to compile thisfile*/
#ifndef __REQUIRED_RPCNDR_H_VERSION__
#define __REQUIRED_RPCNDR_H_VERSION__ 440
#endif

#include "rpc.h"
#include "rpcndr.h"

#ifndef __RPCNDR_H_VERSION__
#error this stub requires an updated version of <rpcndr.h>
#endif // __RPCNDR_H_VERSION__

#ifndef COM_NO_WINDOWS_H
#include "windows.h"
#include "ole2.h"
#endif /*COM_NO_WINDOWS_H*/

#ifndef __SMTime_h__
#define __SMTime_h__

#ifdef __cplusplus
extern "C"{
#endif

/* Forward Declarations */

#ifndef __ISMTimer_FWD_DEFINED__
#define __ISMTimer_FWD_DEFINED__
typedef interface ISMTimer ISMTimer;
#endif          /* __ISMTimer_FWD_DEFINED__ */


#ifndef __SMTimer_FWD_DEFINED__
#define __SMTimer_FWD_DEFINED__

#ifdef __cplusplus
typedef class SMTimer SMTimer;
#else
typedef struct SMTimer SMTimer;
#endif /* __cplusplus */

#endif          /* __SMTimer_FWD_DEFINED__ */


/* header files for imported files */
#include "oaidl.h"
#include "ocidl.h"

void __RPC_FAR * __RPC_USER MIDL_user_allocate(size_t);
void __RPC_USER MIDL_user_free( void __RPC_FAR * );

#ifndef __ISMTimer_INTERFACE_DEFINED__
#define __ISMTimer_INTERFACE_DEFINED__

/* interface ISMTimer */
/* [unique][helpstring][dual][uuid][object]*/


EXTERN_C const IID IID_ISMTimer;

#if defined(__cplusplus) && !defined(CINTERFACE)

    MIDL_INTERFACE("1A6D0AE4-8528-453B-B8E3-8DAD1F0561B7")
    ISMTimer : public IDispatch
```

```
    {
    public:
        virtual /* [helpstring][id] */ HRESULT STDMETHODCALLTYPE Start(
void) = 0;

        virtual /* [helpstring][id] */ HRESULT STDMETHODCALLTYPE Stop(
            CURRENCY __RPC_FAR *pElapsedTime) = 0;

        virtual /* [helpstring][id][propget] */ HRESULT STDMETHODCALLTYPE
get_Running(
            /* [retval][out] */ BOOL __RPC_FAR *pVal) = 0;

    };

#else           /* C style interface */

    typedef struct ISMTimerVtbl
    {
        BEGIN_INTERFACE

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *QueryInterface )(
            ISMTimer __RPC_FAR * This,
            /* [in] */ REFIID riid,
            /* [iid_is][out] */ void __RPC_FAR *__RPC_FAR *ppvObject);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *AddRef )(
            ISMTimer __RPC_FAR * This);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *Release )(
            ISMTimer __RPC_FAR * This);

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *GetTypeInfoCount
)(
            ISMTimer __RPC_FAR * This,
            /* [out] */ UINT __RPC_FAR *pctinfo);

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *GetTypeInfo )(
            ISMTimer __RPC_FAR * This,
            /* [in] */ UINT iTInfo,
            /* [in] */ LCID lcid,
            /* [out] */ ITypeInfo __RPC_FAR *__RPC_FAR *ppTInfo);

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *GetIDsOfNames )(
            ISMTimer __RPC_FAR * This,
            /* [in] */ REFIID riid,
            /* [size_is][in] */ LPOLESTR __RPC_FAR *rgszNames,
            /* [in] */ UINT cNames,
            /* [in] */ LCID lcid,
            /* [size_is][out] */ DISPID __RPC_FAR *rgDispId);

        /* [local] */ HRESULT ( STDMETHODCALLTYPE __RPC_FAR *Invoke
)(
            ISMTimer __RPC_FAR * This,
            /* [in] */ DISPID dispIdMember,
            /* [in] */ REFIID riid,
            /* [in] */ LCID lcid,
            /* [in] */ WORD wFlags,
            /* [out][in] */ DISPPARAMS __RPC_FAR *pDispParams,
            /* [out] */ VARIANT __RPC_FAR *pVarResult,
            /* [out] */ EXCEPINFO __RPC_FAR *pExcepInfo,
            /* [out] */ UINT __RPC_FAR *puArgErr);

        /* [helpstring][id] */ HRESULT ( STDMETHODCALLTYPE __RPC_FAR
*Start )(
            ISMTimer __RPC_FAR * This);

        /* [helpstring][id] */ HRESULT ( STDMETHODCALLTYPE __RPC_FAR
*Stop )(
            ISMTimer __RPC_FAR * This,
            CURRENCY __RPC_FAR *pElapsedTime);

        /* [helpstring][id][propget] */ HRESULT ( STDMETHODCALLTYPE
__RPC_FAR *get_Running )(
            ISMTimer __RPC_FAR * This,
            /* [retval][out] */ BOOL __RPC_FAR *pVal);

        END_INTERFACE
    } ISMTimerVtbl;

    interface ISMTimer
    {
        CONST_VTBL struct ISMTimerVtbl __RPC_FAR *lpVtbl;
    };



#ifdef COBJMACROS


#define ISMTimer_QueryInterface(This,riid,ppvObject)            \
    (This)->lpVtbl -> QueryInterface(This,riid,ppvObject)

#define ISMTimer_AddRef(This)       \
    (This)->lpVtbl -> AddRef(This)

#define ISMTimer_Release(This)      \
    (This)->lpVtbl -> Release(This)


#define ISMTimer_GetTypeInfoCount(This,pctinfo)             \
    (This)->lpVtbl -> GetTypeInfoCount(This,pctinfo)

#define ISMTimer_GetTypeInfo(This,iTInfo,lcid,ppTInfo)      \
    (This)->lpVtbl -> GetTypeInfo(This,iTInfo,lcid,ppTInfo)

#define ISMTimer_GetIDsOfNames(This,riid,rgszNames,cNames,lcid,rgDispId)
\
    (This)->lpVtbl -> GetIDsOfNames(This,riid,rgszNames,cNames,lcid,rgDispId)

#define
ISMTimer_Invoke(This,dispIdMember,riid,lcid,wFlags,pDispParams,pVarResult,
pExcepInfo,puArgErr)    \
    (This)->lpVtbl ->
Invoke(This,dispIdMember,riid,lcid,wFlags,pDispParams,pVarResult,pExcepInfo
,puArgErr)


#define ISMTimer_Start(This)            \
    (This)->lpVtbl -> Start(This)

#define ISMTimer_Stop(This,pElapsedTime)        \
    (This)->lpVtbl -> Stop(This,pElapsedTime)

#define ISMTimer_get_Running(This,pVal)         \
    (This)->lpVtbl -> get_Running(This,pVal)

#endif /* COBJMACROS */


#endif          /* C style interface */



/* [helpstring][id] */ HRESULT STDMETHODCALLTYPE
ISMTimer_Start_Proxy(
    ISMTimer __RPC_FAR * This);


void __RPC_STUB ISMTimer_Start_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


/* [helpstring][id] */ HRESULT STDMETHODCALLTYPE
ISMTimer_Stop_Proxy(
    ISMTimer __RPC_FAR * This,
    CURRENCY __RPC_FAR *pElapsedTime);


void __RPC_STUB ISMTimer_Stop_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);
```

```
/* [helpstring][id][propget] */ HRESULT STDMETHODCALLTYPE
ISMTimer_get_Running_Proxy(
    ISMTimer __RPC_FAR * This,
    /* [retval][out] */ BOOL __RPC_FAR *pVal);


void __RPC_STUB ISMTimer_get_Running_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);



#endif          /* __ISMTimer_INTERFACE_DEFINED__ */



#ifndef __SMTIMELib_LIBRARY_DEFINED__
#define __SMTIMELib_LIBRARY_DEFINED__

/* library SMTIMELib */
/* [helpstring][version][uuid] */


EXTERN_C const IID LIBID_SMTIMELib;


#ifndef __StepMasterTimeFunctions_MODULE_DEFINED__
#define __StepMasterTimeFunctions_MODULE_DEFINED__


/* module StepMasterTimeFunctions */
/* [dllname][version][helpstring] */

/* [entry][helpstring] */ CURRENCY __stdcall Get64BitTime(
    /* [in] */ LPSYSTEMTIME lpInitTime);

/* [entry][helpstring] */ void __stdcall JulianToTime(
    /* [in] */ CURRENCY julianTS,
    /* [out][in] */ int __RPC_FAR *yr,
    /* [out][in] */ int __RPC_FAR *mm,
    /* [out][in] */ int __RPC_FAR *dd,
    /* [out][in] */ int __RPC_FAR *hh,
    /* [out][in] */ int __RPC_FAR *mi,
    /* [out][in] */ int __RPC_FAR *ss,
    /* [out][in] */ int __RPC_FAR *ms);

#endif /* __StepMasterTimeFunctions_MODULE_DEFINED__ */

EXTERN_C const CLSID CLSID_SMTimer;

#ifdef __cplusplus

class DECLSPEC_UUID("27BAB71B-89E1-4A78-8854-FDFFBDC8037E")
SMTimer;
#endif
#endif /* __SMTIMELib_LIBRARY_DEFINED__ */

/* Additional Prototypes for ALL interfaces */

/* end of Additional Prototypes */

#ifdef __cplusplus
}
#endif

#endif
// SMTime.idl : IDL source for SMTime.dll
//

// This file will be processed by the MIDL tool to
// produce the type library (SMTime.tlb) and marshalling code.

import "oaidl.idl";
import "ocidl.idl";
        [
                object,
                uuid(1A6D0AE4-8528-453B-B8E3-8DAD1F0561B7),
                dual,
                helpstring("ISMTimer Interface"),
                pointer_default(unique)
        ]
        interface ISMTimer : IDispatch
        {
                [id(1), helpstring("method Start")] HRESULT Start();
                [id(2), helpstring("method Stop")] HRESULT
Stop(CURRENCY *pElapsedTime);
                [propget, id(3), helpstring("property Running")]
HRESULT Running([out, retval] BOOL *pVal);
        };

[
        uuid(1B31AB30-D7C1-41DB-B654-C9FA1A7D267F),
        version(1.0),
        helpstring("SMTime 1.0 Type Library")
]
library SMTIMELib
{
        importlib("stdole32.tlb");
        importlib("stdole2.tlb");

        // Now define the module that will "declare" your C functions.
        [
        helpstring("Functions exported by SMTime.dll"),
        version(1.0),
        dllname("SMTime.dll")
        ]
        module StepMasterTimeFunctions
        {

                [
                        // Add a description for your function that
the developer can
                        // read in the VB Object Browser.
                        helpstring("Returns the time in 64 bits."),
                        // Specify the actual DLL entry point for
the function. Notice
                        // the entry field is like the Alias keyword
in a VB Declare
                        // statement -- it allows you to specify a
more friendly name
                        // for your exported functions.
                        entry("SMTime_Get64BitTime")
                ]
                // The [in], [out], and [in, out] keywords tell the
Automation
                // client which direction parameters need to be passed.
Some
                // calls can be optimized if a function only needs a
parameter
                // to be passed one-way.
                CURRENCY __stdcall Get64BitTime([in]
LPSYSTEMTIME lpInitTime);
                [
                        helpstring("Converts the Julian time into
it's components."),
                        entry("SMTime_JulianToTime")
                ]
                void __stdcall JulianToTime([in] CURRENCY
julianTS, [in, out] int *yr, [in, out] int* mm, [in, out] int* dd, [in, out] int *hh, [in,
out] int *mi, [in, out] int *ss, [in, out] int *ms);

        } // End of Module

        [
                uuid(27BAB71B-89E1-4A78-8854-FDFFBDC8037E),
                helpstring("SMTimer Class")
        ]
        coclass SMTimer
        {
                [default] interface ISMTimer;
        };
};
```

```
// SMTimer.cpp : Implementation of CSMTimer
#include "stdafx.h"
#include "SMTime.h"
#include "SMTimer.h"

/////////////////////////////////////////////////////////////////////////////
// CSMTimer

/////////////////////////////////////////////////////////////////////
// Construction/Destruction
/////////////////////////////////////////////////////////////////////

CSMTimer::~CSMTimer()
{

}

STDMETHODIMP CSMTimer::Start()
{
	// Starts the timer
	assert(!m_bInProcess);
	m_bInProcess = TRUE;

	m_lStartTime = MyTickCount();

	return S_OK;
}

STDMETHODIMP CSMTimer::Stop(CURRENCY *pElapsedTime)
{
	TC_TIME		lEndTime = MyTickCount();

	// Stops the timer and returns the elapsed time
	assert(m_bInProcess);
	m_bInProcess = FALSE;

	pElapsedTime->int64 = lEndTime - m_lStartTime;

	return S_OK;
}

TC_TIME CSMTimer::MyTickCount(void)
{
	TC_TIME	currentTC;
	LARGE_INTEGER	l;
	__int64 count;

	//The purpose of this function is to prevent the 49 day wrapping
effect of the
	//system API GetTickCount(). This function essentially provides a
monotonically
	//increasing timer value which is milliseconds from class
instantiation.

	if ( m_bCountUnavailable )
	{
		count = (__int64)GetTickCount();
		currentTC = (TC_TIME)(count-m_baseTC);
	}
	else
	{
		QueryPerformanceCounter(&l);
		count = (__int64)l.HighPart << 32 | (__int64)l.LowPart;
		currentTC = (TC_TIME)(((count-m_baseTC) * 1000) /
m_Timerfreq);
	}

	return currentTC;
}

STDMETHODIMP CSMTimer::get_Running(BOOL*pVal)
{
	*pVal = m_bInProcess;

	return S_OK;
}

CURRENCY __stdcall Get64BitTime(LPSYSTEMTIME lpInitTime)
{
	__int64	ms_day, ms_hour, ms_minute, ms_seconds,
ms_milliseconds, ms_total;
	int			day;
	SYSTEMTIME		tim;
	CURRENCY		tmReturn;

	if ( lpInitTime )
		memcpy(&tim, lpInitTime, sizeof(SYSTEMTIME));
	else
		GetLocalTime(&tim);
	day = JulianDay((int)tim.wYear, (int)tim.wMonth, (int)tim.wDay);

	ms_day			= (__int64)day * (__int64)(24
* 1000 * 60 * 60);
	ms_hour			= (__int64)tim.wHour	*
(__int64)(1000 * 3600);
	ms_minute		= (__int64)tim.wMinute * (1000 * 60);
	ms_seconds		= (__int64)(tim.wSecond * 1000);
	ms_milliseconds		= (__int64)tim.wMilliseconds;

	ms_total = ms_day + ms_hour + ms_minute + ms_seconds +
ms_milliseconds;
	tmReturn.int64 = ms_total;

	return tmReturn;
}

// JulianDay computes the number of days since Jan 1, 1900.
// This function is valid for dates from 1-Jan-1900 to 1-Jan-2100.
// 1-Jan-1900 = 0
int JulianDay( int yr, int mm, int dd )
{
	// MonthArray contains cumulative days for months in a non
leap-year
	int MonthArray[12] = { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273,
304, 334};
	int j1, j2;

	// compute day of year (j1)
	j1 = MonthArray[mm-1]+ dd - 1;
	// adjust day of year if this is a leap year and it is after February
	if ((yr % 4)==0 && (yr != 1900) && (mm > 2))
		j1++;
	// compute number of days from 1/1/1900 to beginning of present
year
	j2 = (yr-1900)*365 + (yr-1901)/4;
	return j1+j2;

}

// Breaks up the Julian Time into it's sub-components
void __stdcall SMTime_JulianToTime(CURRENCY CurJulian, int* yr, int* mm,
int* dd, int *hh, int *mi, int *ss, int *ms )
{
	int julianDay, msLeft;
	JULIAN_TIME			julianTS = CurJulian.int64;

	*ms = julianTS % 1000;

	julianTS /= 1000;

	julianDay = (int)(julianTS / ( 60 * 60 * 24 ));

	JulianToCalendar(julianDay, yr, mm, dd );

	msLeft = (int)(julianTS - (julianDay * (__int64)( 60 * 60 * 24 )));

	*hh = msLeft / (60 * 60);
	msLeft = msLeft - *hh * 3600;
	*mi = msLeft / (60);
	*ss = msLeft % 60;
}

// JulianToCalendar converts a day index (from the JulianDay function) to
// its corresponding calendar value (mm/dd/yr).  The valid range for days
// is { 0 .. 73049 } for dates from 1-Jan-1900 to 1-Jan-2100.
void JulianToCalendar( int day, int* yr, int* mm, int* dd )
```

```
{
	int y, m, d;
	// month array contains days of months for months in a non leap-year
	int month[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

	// compute year from days
	if (day < 365)
		y = 1900;
	else
		y = 1901 + ((day-365)/1461)*4 +
(4*((day-365)%1461)+3)/1461;

	// adjust February if this year is a leap year
	if ((y % 4)==0 && (y != 1900))
		month[1] = 29;
	else
		month[1] = 28;

	d = day - JulianDay( y, 1, 1 ) + 1;
	m = 1;

	while (d > month[m-1])
	{
		d = d - month[m-1];
		m++;
	}

	*yr = y;
	*mm = m;
	*dd = d;
}
// SMTimer.h : Declaration of the CSMTimer

#ifndef __SMTIMER_H_
#define __SMTIMER_H_

#include "resource.h"	// main symbols

#include "assert.h"

#define	MAX_JULIAN_TIME		0x7FFFFFFFFFFFFFFF
#define JULIAN_TIME	__int64
#define	TC_TIME		DWORD

#ifdef SMTIMER
#define DLL_LINK	__declspec( dllexport )
#else
#define DLL_LINK	__declspec( dllimport )
#endif

#ifdef __cplusplus
extern "C"
{
#endif
//DLL_LINK		CURRENCY		__stdcall
SMTime_Get64BitTime(LPSYSTEMTIME lpInitTime);
int				JulianDay( int yr, int mm, int dd );
void			JulianToCalendar( int day, int* yr, int* mm, int* dd );
#ifdef __cplusplus
}
#endif


/////////////////////////////////////////////////////////////////////////////
// CSMTimer
class ATL_NO_VTABLE CSMTimer :
	public CComObjectRootEx<CComSingleThreadModel>,
	public CComCoClass<CSMTimer, &CLSID_SMTimer>,
	public IDispatchImpl<ISMTimer, &IID_ISMTimer,
&LIBID_SMTIMELib>
{
public:
	CSMTimer()
	{
		LARGE_INTEGER	l;

		if ( !QueryPerformanceFrequency(&l))
		{
			m_baseTC = (__int64)GetTickCount();
			m_bCountUnavailable = TRUE;
		}
		else
		{
			m_bCountUnavailable = FALSE;

			m_Timerfreq = (__int64)l.HighPart << 32 |
(__int64)l.LowPart;
			QueryPerformanceCounter(&l);
			m_baseTC = (__int64)l.HighPart << 32 |
(__int64)l.LowPart;
		}
		m_bInProcess = FALSE;
	}

DECLARE_REGISTRY_RESOURCEID(IDR_SMTIMER)

DECLARE_PROTECT_FINAL_CONSTRUCT()

BEGIN_COM_MAP(CSMTimer)
	COM_INTERFACE_ENTRY(ISMTimer)
	COM_INTERFACE_ENTRY(IDispatch)
//	COM_INTERFACE_ENTRY2(IDispatch, ISMTimer)
END_COM_MAP()

// ISMTimer
public:
	STDMETHOD(get_Running)(/*[out,retval]*/ BOOL *pVal);
	STDMETHOD(Stop)(CURRENCY *pElapsedTime);
	STDMETHOD(Start)();
	virtual ~CSMTimer();

private:
	__int64			m_baseTC;
	__int64			m_Timerfreq;
	BOOL			m_bCountUnavailable;
	TC_TIME			m_lStartTime;
	BOOL			m_bInProcess;

	TC_TIME			MyTickCount(void);
};

#endif //__SMTIMER_H_
Attribute VB_Name = "SortSM"
'  FILE:      SortSM.bas
'           Microsoft  TPC-H Kit Ver. 1.00
'           Copyright Microsoft, 1999
'           All Rights Reserved
'
'
'  PURPOSE:    This module contains an implementation ofQuickSort.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Comment out for case-sensitive sorts
Option Compare Text

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "SortSM."

Private Function Compare(ByVal vntToCompare1 As Variant, _
      ByVal vntToCompare2 As Variant) As Integer

   On Error GoTo CompareErr

   Compare = 0

   If vntToCompare1.SequenceNo < vntToCompare2.SequenceNo Then
      Compare = -1
   ElseIf vntToCompare1.SequenceNo > vntToCompare2.SequenceNo Then
      Compare = 1
   End If

   Exit Function
```

```
CompareErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errCompareFailed, _
            gstrSource, _
            LoadResString(errCompareFailed)

End Function

Private Sub Swap(ByRef vntToSwap1 As Variant, _
        ByRef vntToSwap2 As Variant)

    Dim vntTemp As Variant

    On Error GoTo SwapErr

    If IsObject(vntToSwap1) And IsObject(vntToSwap2) Then
        Set vntTemp = vntToSwap1
        Set vntToSwap1 = vntToSwap2
        Set vntToSwap2 = vntTemp
    Else
        vntTemp = vntToSwap1
        vntToSwap1 = vntToSwap2
        vntToSwap2 = vntTemp
    End If

    Exit Sub

SwapErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errQuickSortFailed, mstrModuleName & "Swap", _
        LoadResString(errQuickSortFailed)

End Sub

Public Sub QuickSort(vntArray As Variant, _
        Optional ByVal intLBound As Integer, _
        Optional ByVal intUBound As Integer)
    ' Sorts a variant array using Quicksort

    Dim i As Integer
    Dim j As Integer
    Dim vntMid As Variant

    On Error GoTo QuickSortErr

    If IsEmpty(vntArray) Or _
            Not IsArray(vntArray) Then
        Exit Sub
    End If

    ' Set default boundary values for first time through
    If intLBound = 0 And intUBound = 0 Then
        intLBound = LBound(vntArray)
        intUBound = UBound(vntArray)
    End If

    ' BugMessage "Sorting elements " & Str(intLBound) & " and " &
Str(intUBound)

    If intLBound > intUBound Then
        Exit Sub
    End If

    ' Only two elements in this subdivision; exchange if they
    ' are out of order and end recursive calls
    If (intUBound - intLBound) = 1 Then
        If Compare(vntArray(intLBound), vntArray(intUBound)) > 0 Then
            Call Swap(vntArray(intLBound), vntArray(intUBound))
        End If
        Exit Sub
    End If

    ' Set the pivot point
    Set vntMid = vntArray(intUBound)
    i = intLBound
    j = intUBound

    Do
        ' Move in from both sides towards pivot element
        Do While (i < j) And Compare(vntArray(i), vntMid) <= 0
            i = i + 1
        Loop

        Do While (j > i) And Compare(vntArray(j), vntMid) >= 0
            j = j - 1
        Loop

        If i < j Then
            Call Swap(vntArray(i), vntArray(j))
        End If
    Loop While i < j

    ' Since i has been adjusted, swap element i with element,
    ' intUBound
    Call Swap(vntArray(i), vntArray(intUBound))

    ' Recursively call sort array - pass smaller subdivision
    ' first to conserve stack space
    If (i - intLBound) < (intUBound - 1) Then
        ' Recursively sort with adjusted values for upper and
        ' lower bounds
        Call QuickSort(vntArray, intLBound, i - 1)
        Call QuickSort(vntArray, i + 1, intUBound)
    Else
        Call QuickSort(vntArray, i + 1, intUBound)
        Call QuickSort(vntArray, intLBound, i - 1)
    End If
    Exit Sub

QuickSortErr:
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errQuickSortFailed, mstrModuleName &
"QuickSort", _
        LoadResString(errQuickSortFailed)

End Sub
Attribute VB_Name = "Startup"
'  FILE:       Startup.bas
'              Microsoft TPC-H Kit Ver. 1.00
'              Copyright Microsoft, 1999
'              All Rights Reserved
'
'
'  PURPOSE:    This module contains startup and cleanup functions for the
project.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Public Const LISTVIEW_BUTTON = 14

Public gstrProjectPath As String

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "Startup."

Private Sub Initialize()

    On Error GoTo InitializeErr

    ReDim gsContCriteria(gintOnFailureAbortTo gintOnFailureAsk) As String
    gsContCriteria(gintOnFailureAbort)= "Abort"
    gsContCriteria(gintOnFailureContinue)= "Continue"
    gsContCriteria(gintOnFailureCompleteSiblings)= "Execute sibling steps and
stop"
    gsContCriteria(gintOnFailureAbortSiblings)= "Abort sibling steps and execute
next parent"
    gsContCriteria(gintOnFailureSkipSiblings)= "Skip sibling steps and execute
next parent"
    gsContCriteria(gintOnFailureAsk)= "Ask"
```

```
    ReDim gsExecutionStatus(gintDisabledTo gintAborted) As String
    gsExecutionStatus(gintDisabled)= "Disabled"
    gsExecutionStatus(gintPending)= "Pending"
    gsExecutionStatus(gintRunning)= "Running"
    gsExecutionStatus(gintComplete)= "Complete"
    gsExecutionStatus(gintFailed)= "Failed"
    gsExecutionStatus(gintAborted)= "Stopped"

#If Not RUN_ONLY Then
    ' Call a procedure to change the style of the toolbar
    ' on the Step Properties form
    Call SetTBar97(frmSteps.tblConstraintCommands)

#End If

    Call InitRunEngine

    Exit Sub

InitializeErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "Initialize"
    Call ShowError(errInitializeFailed)

End Sub
Sub Main()

    On Error GoTo MainErr

    ' Mousepointer should indicate busy
    Call ShowBusy

    ' Display the Splash screen while we carry out some initialization
    frmSplash.Show
    frmSplash.Refresh

    gstrProjectPath = App.Path

    ' Open the database
    If OpenDBFile() = False Then
        Unload frmSplash
        Exit Sub
    End If

#If Not RUN_ONLY Then
    Load frmMain

    ' Enable the Stop Run menu options only when a workspace is
    ' actually running
    Call EnableStop(False)

    ' Clear all application extension menu items
    Call ClearToolsMenu
#End If

    Call Initialize

    ' Mousepointer - ready to accept user input
    Call ShowFree

    ' Unload the Splash screen and display the main form
    Unload frmSplash

#If RUN_ONLY Then
    frmWorkspaceOpen.Caption = gsCaptionRunWsp

    Call ShowWorkspacesInDb(dbsAttTool)
#Else
    frmMain.Show
#End If

    Exit Sub

MainErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    Call ShowFree
    Call ShowError(errMainFailed)

End Sub
Private Function OpenDBFile() As Boolean
    Dim sDb As String

    On Error GoTo OpenDBFileErr

#If RUN_ONLY Then
    ' Always use the registry setting for the run_only mode
    sDb = DefaultDBFile()
#Else
    ' Check if the user has specified the workspace defn. file to open on the
command line
    ' Else, use the registry setting
    sDb = IIf(StringEmpty(Command),DefaultDBFile(), Command)

    If Len(sDb) > 0 Then
        ' Trim off the enclosing double-quotes if any
        If Mid(sDb, 1, 1) = gstrDQ Then
            If Len(sDb) > 1 Then
                sDb = Mid(sDb, 2)
            Else
                sDb = gstrEmptyString
            End If
        End If
    End If

    If Len(sDb) > 0 Then
        If Mid(sDb, Len(sDb), 1) = gstrDQ Then
            If Len(sDb) > 1 Then
                sDb = Mid(sDb, 1, Len(sDb) - 1)
            Else
                sDb = gstrEmptyString
            End If
        End If
    End If
#End If

    ' Open the database
    OpenDBFile = SMOpenDatabase(sDb)

    Exit Function

OpenDBFileErr:
    Call LogErrors(Errors)
    OpenDBFile = False

End Function
Public Sub Cleanup()

    On Error GoTo CleanupErr

    ' Set the mousepointer to indicate Busy
    Call ShowBusy

#If Not RUN_ONLY Then
    ' Close all open workspaces - will also prompt for unsaved
    ' changes
    Call CloseOpenWorkspaces
#End If

    ' Close all open files
    Call CloseOpenFiles

    ' Reset the mousepointer
    Call ShowFree

    Exit Sub

CleanupErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    Resume Next

End Sub
Attribute VB_Name = "StepCommon"
```

```
' FILE:      StepCommon.bas
'           Microsoft TPC-H Kit Ver. 1.00
'           Copyright Microsoft, 1999
'           All Rights Reserved
'
'
' PURPOSE:    Contains functionality common across StepMaster and
'           SMRunOnly, pertaining to steps
'           Specifically, functions to load iterators records
'           in an array, determine the type of step, etc.
' Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "StepCommon."

' Step property constants
Private Const mintMinFailureThreshold As Integer = 1
Public Const gintMinSequenceNo As Integer = 1
Public Const gintMinLevel As Integer = 0
Public Function ValidateParallelism(sParallelism As String, lWorkspace As Long,
_
      Optional ParamsInWsp As cArrParameters = Nothing) As String
   ' Returns the degree of parallelism for the step if the user input is valid
   Dim sTemp As String

   On Error GoTo ValidateParallelismErr
   gstrSource = mstrModuleName & "ValidateParallelism"

   sTemp = SubstituteParameters(Trim$(sParallelism),lWorkspace,
WspParameters:=ParamsInWsp)

   If Not IsNumeric(sTemp) Then
      ShowError errInvalidDegParallelism
      On Error GoTo 0
      Err.Raise vbObjectError + errInvalidDegParallelism, gstrSource, _
            LoadResString(errInvalidDegParallelism)
   Else
      If (CInt(sTemp) < gintMinParallelism) Or (CInt(sTemp) >
gintMaxParallelism) Then
         ShowError errInvalidDegParallelism
         On Error GoTo 0
         Err.Raise vbObjectError + errInvalidDegParallelism, gstrSource, _
               LoadResString(errInvalidDegParallelism)
      Else
         ValidateParallelism = Trim$(sParallelism)
      End If
   End If

   Exit Function

ValidateParallelismErr:
   ' Log the error code raised by Visual Basic
   gstrSource = mstrModuleName & "ValidateParallelism"
   If Err.Number = vbObjectError + errSubValuesFailed Then
      ShowError errInvalidDegParallelism
   End If

   Call LogErrors(Errors)
   On Error GoTo 0
   Err.Raise vbObjectError + errInvalidDegParallelism, gstrSource, _
      LoadResString(errInvalidDegParallelism)

End Function

Public Function IsGlobal( _
      Optional ByVal StepClass As cStep = Nothing, _
      Optional ByVal StepRecord As Recordset = Nothing, _
      Optional ByVal StepKey As String = gstrEmptyString, _
      Optional ByVal StepId As Long = 0, _
      Optional StepForm As Form = Nothing) As Boolean

   ' This function contains all the possible checks for whether
   ' a step is global - The check that will be made depends on
   ' the parameter passed in
   Dim cStepRecord As cStep

   If Not StepClass Is Nothing Then
      IsGlobal = StepClass.GlobalFlag
      Exit Function
   End If

   If Not StepRecord Is Nothing Then
      IsGlobal = StepRecord![global_flag]
      Exit Function
   End If

   If Not StringEmpty(StepKey) Then
      IsGlobal = InStr(StepKey, gstrGlobalStepPrefix) > 0
      Exit Function
   End If

   If StepId <> 0 Then
      Set cStepRecord = gcSteps.QueryStep(StepId)
      IsGlobal = cStepRecord.GlobalFlag
      Set cStepRecord = Nothing
      Exit Function
   End If

   If Not StepForm Is Nothing Then
      IsGlobal = (StepForm.lblStepType.Caption = Str(gintGlobalStep))
      Exit Function
   End If

   ' Not a single object was passed in! - raise an error
   On Error GoTo 0
   Err.Raise vbObjectError + errObjectMandatory, _
         mstrModuleName & "IsGlobal", _
         LoadResString(errObjectMandatory)

End Function
Public Function TypeOfStep(Optional ByVal StepClass As cStep = Nothing, _
      Optional ByVal StepRecord As Recordset = Nothing, _
      Optional ByVal StepKey As String = gstrEmptyString, _
      Optional ByVal StepId As Long = 0, _
      Optional StepForm As Form = Nothing) As Integer
   ' Calls functions to determine the type of step
   ' The check that will be made depends on the parameter passed in

   On Error GoTo TypeOfStepErr

   ' Make the check whether a step is global first - both
   ' worker and global steps have the step text or file name
   ' not null - but only the global step will have the global
   ' flag set
   If IsGlobal(StepClass, StepRecord, StepKey, StepId, StepForm) Then
      TypeOfStep = gintGlobalStep
   ElseIf IsManager(StepClass, StepRecord, StepKey, StepId, StepForm) Then
      TypeOfStep = gintManagerStep
   ElseIf IsWorker(StepClass, StepRecord, StepKey, StepId, StepForm) Then
      TypeOfStep = gintWorkerStep
   Else
      On Error GoTo 0
      Err.Raise vbObjectError + errInvalidStep, _
            mstrModuleName & "TypeOfStep", _
            LoadResString(errInvalidStep)
   End If

   Exit Function

TypeOfStepErr:
   ' Log the error code raised by Visual Basic
   Call LogErrors(Errors)
   On Error GoTo 0
   Err.Raise vbObjectError + errTypeOfStepFailed, _
         mstrModuleName & "TypeOfStep", _
         LoadResString(errTypeOfStepFailed)

End Function

Public Function IsStep(intNodeType As Integer) As Boolean
   ' Returns true if the node type corresponds to a global, manager
   ' or worker step
```

```
    IsStep = (intNodeType = gintGlobalStep) Or (intNodeType =
gintManagerStep) Or _
        (intNodeType = gintWorkerStep)

End Function

Public Function IsManager(Optional ByVal StepClass As cStep = Nothing, _
        Optional ByVal StepRecord As Recordset = Nothing, _
        Optional ByVal StepKey As String = gstrEmptyString, _
        Optional ByVal StepId As Long = 0, _
        Optional StepForm As Form = Nothing) As Boolean

    ' This function contains all the possible checks for whether
    ' a step is a manager step - The check that will be made depends
    ' on the parameter passed in

    Dim cStepRecord As cStep

    If Not StepClass Is Nothing Then
        IsManager = (StepClass.StepType = gintManagerStep)
        Exit Function
    End If

    If Not StepRecord Is Nothing Then
        IsManager = (IsNull(StepRecord![step_text]) And
IsNull(StepRecord![step_file_name]))
        Exit Function
    End If

    If Not StringEmpty(StepKey) Then
        IsManager = (InStr(StepKey, gstrManagerStepPrefix) > 0)
        Exit Function
    End If

    If StepId <> 0 Then
        Set cStepRecord = gcSteps.QueryStep(StepId)
        IsManager = (cStepRecord.StepType = gintManagerStep)
        Set cStepRecord = Nothing
        Exit Function
    End If

    If Not StepForm Is Nothing Then
        IsManager = (StepForm.lblStepType.Caption = Str(gintManagerStep))
        Exit Function
    End If

    ' Not a single object was passed in! - raise an error
    On Error GoTo 0
    Err.Raise vbObjectError + errObjectMandatory, _
            "Step.IsManager", _
            LoadResString(errObjectMandatory)

End Function
Public Function IsWorker( _
        Optional ByVal StepClass As cStep = Nothing, _
        Optional ByVal StepRecord As Recordset = Nothing, _
        Optional ByVal StepKey As String = gstrEmptyString, _
        Optional ByVal StepId As Long = 0, _
        Optional StepForm As Form = Nothing) As Boolean

    ' This function contains all the possible checks for whether
    ' a step is a Worker step - The check that will be made depends
    ' on the parameter passed in

    Dim cStepRecord As cStep

    If Not StepClass Is Nothing Then
        IsWorker = (StepClass.StepType = gintWorkerStep)
        Exit Function
    End If

    If Not StepRecord Is Nothing Then
        IsWorker = (Not StepRecord![global_flag] And _
            (Not IsNull(StepRecord![step_text]) Or Not
IsNull(StepRecord![step_file_name])))
        Exit Function
    End If

    If Not StringEmpty(StepKey) Then
        IsWorker = InStr(StepKey, gstrWorkerStepPrefix) > 0
        Exit Function
    End If

    If StepId <> 0 Then
        Set cStepRecord = gcSteps.QueryStep(StepId)
        IsWorker = (cStepRecord.StepType = gintWorkerStep)
        Set cStepRecord = Nothing
        Exit Function
    End If

    If Not StepForm Is Nothing Then
        IsWorker = (StepForm.lblStepType.Caption = Str(gintWorkerStep))
        Exit Function
    End If

    ' Not a single object was passed in! - raise an error
    On Error GoTo 0
    Err.Raise vbObjectError + errObjectMandatory, _
            "Step.IsWorker", _
            LoadResString(errObjectMandatory)

End Function
Public Function GetStepNodeText(ByVal cStepNode As cStep) As String

    On Error GoTo GetStepNodeTextErr

    ' Returns the string that will be displayed as the text
    ' in the tree view node to the user
    If StringEmpty(cStepNode.StepLabel) Then

        If StringEmpty(cStepNode.StepTextFile) Then

            If StringEmpty(cStepNode.StepText) Then
                ' This should never happen
                On Error GoTo 0
                Err.Raise vbObjectError + errStepLabelTextAndFileNull,_
                    gstrSource, _
                    LoadResString(errStepLabelTextAndFileNull)
            Else
                GetStepNodeText = cStepNode.StepText
            End If
        Else
            GetStepNodeText = cStepNode.StepTextFile
        End If
    Else
        GetStepNodeText = cStepNode.StepLabel
    End If

    Exit Function

GetStepNodeTextErr:
    ' Log the error code raised by Visual Basic
    Call LogErrors(Errors)
    On Error GoTo 0
    Err.Raise vbObjectError + errGetStepNodeTextFailed, _
            gstrSource, _
            LoadResString(errGetStepNodeTextFailed)

End Function

Public Function LoadRecordsetInStepsArray(rstSteps As Recordset, _
    cStepCol As cArrSteps) As Boolean

    Dim cNewStep As cStep
    Dim cNewGlobal As cGlobalStep
    Dim cNewManager As cManager
    Dim cNewWorker As cWorker

    On Error GoTo LoadRecordsetInStepsArrayErr

    If rstSteps.RecordCount = 0 Then
        Exit Function
    End If

    rstSteps.MoveFirst
    While Not rstSteps.EOF
```

```
        ' For fields that should not be null, a procedure is first
        ' called to raise an error if the field is null

        Set cNewStep = New cStep

        cNewStep.StepType = TypeOfStep(StepRecord:=rstSteps)

        If cNewStep.StepType = gintGlobalStep Then
            Set cNewGlobal = New cGlobalStep
            Set cNewStep = cNewGlobal
        ElseIf cNewStep.StepType = gintManagerStep Then
            Set cNewManager = New cManager
            Set cNewStep = cNewManager
        Else
            Set cNewWorker = New cWorker
            Set cNewStep = cNewWorker
        End If

        ' Initialize the global flag first, since subsequent
        ' validations might depend on whether the step is global
        cNewStep.GlobalFlag = CBool(ErrorOnNullField(rstSteps,"global_flag"))

        ' Initialize step values
        cNewStep.StepId = CLng(ErrorOnNullField(rstSteps,"step_id"))
        cNewStep.VersionNo = CStr(ErrorOnNullField(rstSteps,"version_no"))

        cNewStep.StepLabel = CheckForNullField(rstSteps,"step_label")
        cNewStep.StepTextFile = CheckForNullField(rstSteps,"step_file_name")
        cNewStep.StepText = CheckForNullField(rstSteps,"step_text")
        cNewStep.StartDir = CheckForNullField(rstSteps,"start_directory")

        cNewStep.WorkspaceId = CLng(ErrorOnNullField(rstSteps,
FLD_ID_WORKSPACE))
        cNewStep.ParentStepId = CLng(ErrorOnNullField(rstSteps,
"parent_step_id"))
        cNewStep.ParentVersionNo = CStr(ErrorOnNullField(rstSteps,
"parent_version_no"))

        cNewStep.SequenceNo = CInt(ErrorOnNullField(rstSteps,"sequence_no"))
        cNewStep.StepLevel = CInt(ErrorOnNullField(rstSteps,"step_level"))
        cNewStep.EnabledFlag = CBool(ErrorOnNullField(rstSteps,
"enabled_flag"))

        ' Initialize the execution details for the step
        cNewStep.DegreeParallelism = CheckForNullField(rstSteps,
"degree_parallelism")
        cNewStep.ExecutionMechanism = CInt(ErrorOnNullField(rstSteps,
"execution_mechanism"))
        cNewStep.FailureDetails = CheckForNullField(rstSteps,"failure_details")
        cNewStep.ContinuationCriteria = CInt(ErrorOnNullField(rstSteps,
"continuation_criteria"))

        ' Initialize the output file locations for the step
        cNewStep.OutputFile = CheckForNullField(rstSteps,"output_file_name")
        ' cNewStep.LogFile = CheckForNullField(rstSteps,"log_file_name")
        cNewStep.ErrorFile = CheckForNullField(rstSteps,"error_file_name")

        ' Initialize the iterator name for the step, if any
        cNewStep.IteratorName = CheckForNullField(rstSteps,"iterator_name")

        ' Add this record to the array of steps
        cStepCol.Load cNewStep

        Set cNewStep = Nothing
        rstSteps.MoveNext
    Wend

    Exit Function

LoadRecordsetInStepsArrayErr:

    LogErrors Errors
    gstrSource = mstrModuleName & "LoadRecordsetInStepsArray"
    On Error GoTo 0
    Err.Raise vbObjectError + errLoadRsInArrayFailed, gstrSource, _
            LoadResString(errLoadRsInArrayFailed)

End Function
```

```
Attribute VB_Name = "TimerSM"
' FILE:      TimerSM.bas
'            Microsoft TPC-H Kit Ver. 1.00
'            Copyright Microsoft, 1999
'            All Rights Reserved
'
'
' PURPOSE:    This module contains wrapper functions for Timer APIs.
' Contact:    Reshma Tharamal (reshmat@microsoft.com)
'

Option Explicit

Private Declare Function SetTimer Lib "user32" (ByVal hWnd As Long, _
    ByVal nIDEvent As Long, ByVal uElapse As Long, ByVal lpTimerFunc As
Long) _
    As Long
Private Declare Function KillTimer Lib "user32" (ByVal hWnd As Long, _
    ByVal nIDEvent As Long) As Long
Private Declare Sub CopyMemory Lib "kernel32" Alias "RtlMoveMemory" ( _
    pDest As Any, pSource As Any, ByVal ByteLen As Long)

Public gcTimerObjects As Collection

Private Sub TimerProc(ByVal lHwnd As Long, ByVal lMsg As Long, _
    ByVal lTimerID As Long, ByVal lTime As Long)

    Dim nPtr As Long
    Dim oTimerObject As cTimerSM

    'Create a Timer object from the pointer
    nPtr = gcTimerObjects.Item(Str$(lTimerID))
    CopyMemory oTimerObject, nPtr, 4
    'Call a method which will fire the Timer event
    oTimerObject.Tick
    'Get rid of the Timer object so that VB will not try to release it
    CopyMemory oTimerObject, 0&, 4
End Sub

Public Function StartTimer(lInterval As Long) As Long
    StartTimer = SetTimer(0, 0, lInterval, AddressOf TimerProc)
End Function

Public Sub StopTimer(lTimerID As Long)
    KillTimer 0, lTimerID
End Sub

Public Sub SetInterval(lInterval As Long, lTimerID As Long)
    SetTimer 0, lTimerID, lInterval, AddressOf TimerProc
End Sub
Attribute VB_Name = "ToolsCommon"
' FILE:      ToolsCommon.bas
'            Microsoft TPC-H Kit Ver. 1.00
'            Copyright Microsoft, 1999
'            All Rights Reserved
'
'
' PURPOSE:    Contains functions to remove run history and initialize
'            table creation scripts
' Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

Public sCreateTables() As String

Public Const gsExtSeparator As String = "."

Public Sub DeleteRunHistory(dbFile As DAO.Database)
    ' Delete all run history records from the database, viz. the records in
    ' run_header, run_step_details and run_parameters

    Dim sDelete As String

    On Error GoTo DeleteRunHistoryErr

    sDelete = "delete from run_header "
    dbFile.Execute sDelete, dbFailOnError
```

```
    sDelete = "delete from run_step_details "
    dbFile.Execute sDelete, dbFailOnError

    sDelete = "delete from run_parameters "
    dbFile.Execute sDelete, dbFailOnError

    sDelete = "update att_identifiers " & _
        " set run_id = " & CStr(glMinId)
    dbFile.Execute sDelete, dbFailOnError

    Exit Sub

DeleteRunHistoryErr:
    LogErrors Errors
    Err.Raise vbObjectError + errDeleteDBRecordFailed, "DeleteRunHistory", _
          LoadResString(errDeleteDBRecordFailed)

End Sub

Public Function CreateConnectionsTableScript() As String
    ' Returns the table creation script for the workspace_connections table

    Call InitCreateSQLArray
    CreateConnectionsTableScript = sCreateTables(10)
    ReDim sCreateTables(0)

End Function
Public Function CreateConnectionDtlsTableScript() As String
    ' Returns the table creation script for the connection_dtls table

    Call InitCreateSQLArray
    CreateConnectionDtlsTableScript = sCreateTables(11)
    ReDim sCreateTables(0)

End Function

Public Sub InitCreateSQLArray()

ReDim sCreateTables(0 To 11)

sCreateTables(0) = "Create table att_identifiers (" & _
        "workspace_id           Long, " & _
        "parameter_id           Long, " & _
        "step_id                Long, " & _
        "constraint_id          Long, " & _
        "run_id                 Long, " & _
        "connection_id          Long " & _
");"

sCreateTables(1) = "Create table att_steps (step_id   Long, " & _
        "version_no             Text(255), " & _
        "step_label             Text(255), " & _
        "step_file_name         Text(255), " & _
        "step_text              Memo, " & _
        "start_directory        Text(255), " & _
        "workspace_id           Long, " & _
        "parent_step_id         Long, " & _
        "parent_version_no      Text(255), " & _
        "step_level             Long, " & _
        "sequence_no            Integer, " & _
        "enabled_flag           Bit, " & _
        "degree_parallelism     Text(255), " & _
        "execution_mechanism    Text(50), " & _
        "failure_details        Text(255), " & _
        "continuation_criteria  Text(50), " & _
        "global_flag            Long, " & _
        "archived_flag          Bit, " & _
        "output_file_name       Text(255), " & _
        "error_file_name        Text(255), " & _
        "iterator_name          Text(255), " & _
       "CONSTRAINT pk_steps PRIMARY KEY (step_id, version_no) " & _
");"

'       "log_file_name          Text(255), " & _

sCreateTables(2) = "Create table att_workspaces (" & _
        "workspace_id           Long, " & _
        "workspace_name         Text(255), " & _
        "archived_flag          Bit, " & _
       "CONSTRAINT pk_workspaces PRIMARY KEY (workspace_id) " & _
");"

sCreateTables(3) = "Create table iterator_values (" & _
        "step_id                Long, " & _
        "version_no             Text(255), " & _
        "type                   Integer, " & _
        "iterator_value         Text(255), " & _
        "sequence_no            Integer " & _
");"

sCreateTables(4) = "Create table run_header (" & _
        "run_id                 Long, " & _
        "workspace_id           Long, " & _
        "start_time             Currency, " & _
        "end_time               Currency, " & _
       "CONSTRAINT pk_run_header PRIMARY KEY (run_id) " & _
");"

sCreateTables(5) = "Create table run_parameters (" & _
        "run_id                 Long, " & _
        "parameter_name         Text(255), " & _
        "parameter_value        Text(255) " & _
");"

sCreateTables(6) = "Create table run_step_details (" & _
        "run_id                 Long, " & _
        "step_id                Long, " & _
        "version_no             Text(255), " & _
        "instance_id            Long, " & _
        "parent_instance_id     Long, " & _
        "command                Memo, " & _
        "iterator_value         Text(255), " & _
        "start_time             Currency, " & _
        "end_time               Currency, " & _
        "elapsed_time           Long " & _
");"

sCreateTables(7) = "Create table step_constraints (" & _
        "constraint_id          Long, " & _
        "step_id                Long, " & _
        "version_no             Text(255), " & _
        "constraint_type        Integer, " & _
        "global_step_id         Long, " & _
        "global_version_no      Text(255), " & _
        "sequence_no            Integer " & _
");"

sCreateTables(8) = "Create table workspace_parameters (" & _
        "workspace_id           Long, " & _
        "parameter_id           Long, " & _
        "parameter_name         Text(255), " & _
        "parameter_value        Text(255), " & _
        "description            Text(255), " & _
        "parameter_type         Integer, " & _
       "CONSTRAINT pk_parameters PRIMARY KEY (parameter_id) " & _
");"

sCreateTables(9) = "Create table db_details (" & _
        "db_version             Text(50) " & _
");"

sCreateTables(10) = "Create table " & TBL_CONNECTION_STRINGS & " ("
& _
        "workspace_id           Long, " & _
        "connection_id          Long, " & _
        "connection_name        Text(255), " & _
        "connection_value       Text(255), " & _
        "description            Text(255), " & _
        "no_count_display       Bit, " & _
        "no_execute             Bit, " & _
        "parse_query_only       Bit, " & _
        "ANSI_quoted_identifiers  Bit, " & _
        "ANSI_nulls             Bit, " & _
        "show_query_plan        Bit, " & _
        "show_stats_time        Bit, " & _
        "show_stats_io          Bit, " & _
```

```
        "parse_odbc_msg_prefixes   Bit, " & _
        "row_count                 long, " & _
        "tsql_batch_separator      Text(255), " & _
        "query_time_out            long, " & _
        "server_language           Text(255), " & _
        "character_translation     Bit, " & _
        "regional_settings         Bit, " & _
        "CONSTRAINT pk_connections PRIMARY KEY (connection_id) " & _
");"

' This table has been added in order to satisfy the TPC-H requirement that
' all the queries in a stream need to be executed on a single connection.
' Specify a connection for each odbc step. If the connection is of type,
' static, it should be kept open till the step execution is complete.
sCreateTables(11) = "Create table " & TBL_CONNECTION_DTLS & " (" & _
        FLD_ID_WORKSPACE & gstrBlank & DATA_TYPE_LONG & ", " & _
        FLD_ID_CONN_NAME & gstrBlank & DATA_TYPE_LONG & ", " & _
        FLD_CONN_DTL_CONNECTION_NAME & gstrBlank &
DATA_TYPE_TEXT255 & ", " & _
        FLD_CONN_DTL_CONNECTION_STRING & gstrBlank &
DATA_TYPE_TEXT255 & ", " & _
        FLD_CONN_DTL_CONNECTION_TYPE & gstrBlank &
DATA_TYPE_INTEGER & ", " & _
        "CONSTRAINT pk_connection_name PRIMARY KEY (" &
FLD_ID_CONN_NAME & ") " & _
");"

End Sub
Attribute VB_Name = "WindowsApiCommon"
'  FILE:      WindowsApiCommon.bas
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:    This module contains functions that are wrappers around the
'              Windows API and are used by both StepMaster and SMRunOnly.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "WindowsApiCommon."

Public Type PROCESS_INFORMATION
   hProcess    As Long
   hThread     As Long
   dwProcessID As Long
   dwThreadID  As Long
End Type

' Used by GetShortName to return the short file name for a given file
Private Declare Function GetShortPathName Lib "kernel32" _
     Alias "GetShortPathNameA" (ByVal lpszLongPath As String, _
     ByVal lpszShortPath As String, ByVal cchBuffer As Long) As Long

Public Declare Function GetExitCodeProcess Lib "kernel32" ( _
   ByVal hProcess As Long, lpExitCode As Long) As Long
Public Declare Function TerminateProcess Lib "kernel32" ( _
      hProcess As Long, uExitCode As Long) As Long
Public Declare Function CloseHandle Lib "kernel32" ( _
   ByVal hObject As Long) As Long

Public Const NORMAL_PRIORITY_CLASS     As Long = &H20&
Public Const INFINITE                  As Long = -1&

Public Const STATUS_WAIT_0             As Long = &H0
Public Const STATUS_ABANDONED_WAIT_0   As Long = &H80
Public Const STATUS_USER_APC           As Long = &HC0
Public Const STATUS_TIMEOUT            As Long = &H102
Public Const STATUS_PENDING            As Long = &H103

Public Const WAIT_FAILED               As Long = &HFFFFFFFF
Public Const WAIT_OBJECT_0             As Long = STATUS_WAIT_0
Public Const WAIT_TIMEOUT              As Long = STATUS_TIMEOUT

Public Const WAIT_ABANDONED            As Long =
STATUS_ABANDONED_WAIT_0
Public Const WAIT_ABANDONED_0          As Long =
STATUS_ABANDONED_WAIT_0

Public Const WAIT_IO_COMPLETION        As Long = STATUS_USER_APC
Public Const STILL_ACTIVE              As Long = STATUS_PENDING

Public Const PROCESS_QUERY_INFORMATION  As Long = &H400
Public Const STANDARD_RIGHTS_REQUIRED   As Long = &HF0000

'--------------------------------------------------------------------------------
'Declarations for shelling:

Public Type STARTUPINFO
   cb              As Long
   lpReserved      As String
   lpDesktop       As String
   lpTitle         As String
   dwX             As Long
   dwY             As Long
   dwXSize         As Long
   dwYSize         As Long
   dwXCountChars   As Long
   dwYCountChars   As Long
   dwFillAttribute As Long
   dwFlags         As Long
   wShowWindow     As Integer
   cbReserved2     As Integer
   lpReserved2     As Long
   hStdInput       As Long
   hStdOutput      As Long
   hStdError       As Long
End Type

Public Declare Function WaitForSingleObject Lib "kernel32" ( _
   ByVal hProcess As Long, ByVal dwMilliseconds As Long) As Long

Public Declare Function InputIdle Lib "user32" Alias "WaitForInputIdle" ( _
   ByVal hProcess As Long, ByVal dwMilliseconds As Long) As Long

Public Declare Function CreateProcessA Lib "kernel32" ( _
   ByVal lpApplicationName As Long, ByVal lpCommandLine As String, _
   ByVal lpProcessAttributes As Long, ByVal lpThreadAttributes As Long, _
   ByVal bInheritHandles As Long, ByVal dwCreationFlags As Long, _
   ByVal lpEnvironment As Long, ByVal lpCurrentDirectory As Long, _
   lpStartupInfo As STARTUPINFO, lpProcessInformation As _
   PROCESS_INFORMATION) As Long

Public Declare Function GetLastError Lib "kernel32" () As Long

Private Type OPENFILENAME
   lStructSize As Long
   hwndOwner As Long
   hInstance As Long
   lpstrFilter As String
   lpstrCustomFilter As String
   nMaxCustFilter As Long
   nFilterIndex As Long
   lpstrFile As String
   nMaxFile As Long
   lpstrFileTitle As String
   nMaxFileTitle As Long
   lpstrInitialDir As String
   lpstrTitle As String
   Flags As Long
   nFileOffset As Integer
   nFileExtension As Integer
   lpstrDefExt As String
   lCustData As Long
   lpfnHook As Long
   lpTemplateName As Long
End Type

Private Declare Function GetOpenFileName Lib "COMDLG32" _
      Alias "GetOpenFileNameA" (file As OPENFILENAME) As Long

Private Declare Function lstrlen Lib "kernel32" (lpsz As String) As Long
```

```
Public Const MAX_PATH = 255

' Used when creating a process
Public Const SW_SHOWMINNOACTIVE = 7
Public Const STARTF_USESHOWWINDOW = &H1

Public Const MB_YESNOCANCEL = &H3&
Public Const MB_ABORTRETRYIGNORE = &H2&
Public Const MB_OK = &H0&

Public Const MB_APPLMODAL = &H0&

Public Const MB_ICONQUESTION = &H20&
Public Const MB_ICONEXCLAMATION = &H30&

Public Const IDABORT = 3
Public Const IDRETRY = 4
Public Const IDIGNORE = 5
Public Const IDYES = 6
Public Const IDNO = 7
Public Const IDCANCEL = 2

Private Declare Function MessageBox Lib "user32" Alias "MessageBoxA" ( _
      ByVal hWnd As Long, ByVal lpText As String, _
      ByVal lpCaption As String, ByVal wType As Long) As Long

Private Type SYSTEMTIME
      wYear As Integer
      wMonth As Integer
      wDayOfWeek As Integer
      wDay As Integer
      wHour As Integer
      wMinute As Integer
      wSecond As Integer
      wMilliseconds As Integer
End Type

Private Declare Function Get64BitTime Lib "smtime.dll" ( _
      ByVal lpInitTime As Any) As Currency

Public Function ShowMessageBox(hWnd As Long, strText As String, _
      strTitle As String, wType As Integer) As Long
   ' Using the Windows MessageBox Api since the VB Msgbox function
suppresses
   ' all events
   ShowMessageBox = MessageBox(hWnd, ByVal strText, ByVal strTitle,
wType)

   If ShowMessageBox = 0 Then
      LogSystemError
      Err.Raise vbObjectError + errConfirmFailed, App.EXEName, _
            LoadResString(errConfirmFailed)
   End If

End Function
Public Function ShowFileOpenDialog(ByVal strFilter As String, _
      ByVal strDialogTitle As String, ByVal lngFlags As Long, _
      Optional ByVal strOldFile As String = gstrEmptyString) As String
   ' Returns the file name selected by the user
   Dim strInitDir As String
   Dim intPos As Integer
   Dim opfile As OPENFILENAME
   Dim sFile As String

   On Error GoTo ShowFileOpenDialogErr

   If Not StringEmpty(strOldFile)Then
      intPos = InstrR(strOldFile, gstrFileSeparator)
      If intPos > 0 Then
         strInitDir = Left$(strOldFile, intPos - 1)
      End If
   End If

   With opfile
      .lStructSize = Len(opfile)
      .Flags = lngFlags
      .lpstrInitialDir = strInitDir
      .lpstrTitle = strDialogTitle
      .lpstrFilter = MakeWindowsFilter(strFilter)
      sFile = strOldFile & String$(MAX_PATH - Len(strOldFile), 0)
      .lpstrFile = sFile
      .nMaxFile = MAX_PATH
   End With

   If GetOpenFileName(opfile) Then
      ShowFileOpenDialog = Left$(opfile.lpstrFile, InStr(opfile.lpstrFile,
vbNullChar) - 1)
   Else
      ShowFileOpenDialog = strOldFile
   End If

   Exit Function

ShowFileOpenDialogErr:
   Call LogErrors(Errors)
   ' Reset the selection to the passed in file, if any
   ShowFileOpenDialog = strOldFile

End Function

Private Function MakeWindowsFilter(sFilter As String) As String

   Dim s As String, ch As String, iTemp As Integer

   On Error GoTo MakeWindowsFilterErr

   ' To make Windows-style filter, replace | and : with nulls
   For iTemp = 1 To Len(sFilter)
      ch = Mid$(sFilter, iTemp, 1)
      If ch = "|" Then
         s = s & vbNullChar
      Else
         s = s & ch
      End If
   Next iTemp

   ' Put double null at end
   s = s & vbNullChar & vbNullChar
   MakeWindowsFilter = s

   Exit Function

MakeWindowsFilterErr:
   Call LogErrors(Errors)
   gstrSource = mstrModuleName & "MakeWindowsFilter"
   On Error GoTo 0
   Err.Raise vbObjectError + errApiFailed, gstrSource, _
         LoadResString(errApiFailed)

End Function


Public Function GetShortName(ByVal sLongFileName As String) As String
   ' Returns the short name for the passed in file - will only work
   ' if the passed in path/file exists

   Dim lRetVal As Long, sShortPathName As String, iLen As Integer
   Dim sLongFile As String
   Dim sDir As String
   Dim sFile As String
   Dim intPos As Integer

   On Error GoTo GetShortNameErr

   sFile = gstrEmptyString
   sLongFile = MakePathValid(sLongFileName)
   If StringEmpty(Dir$(sLongFile,vbNormal + vbDirectory)) Then
      ' The passed in path is a file that does not exist - since
      ' the GetShortPathName api does not work on non-existent files
      ' on Win2K, use the directory as an argument to the api and
      ' then append the file
      intPos = InstrR(sLongFile, gstrFileSeparator)
      sDir = Mid$(sLongFile, 1, intPos - 1)
      sFile = Right(sLongFile, Len(sLongFile) - intPos + 1)
      sLongFile = sDir
```

```
    End If

    'Set up buffer area for API function call return
    sShortPathName = Space(MAX_PATH)
    iLen = Len(sShortPathName)

    'Call the function
    lRetVal = GetShortPathName(sLongFile, sShortPathName, iLen)
    If lRetVal = 0 Then
       Call LogSystemError
    End If

    GetShortName = IIf(lRetVal = 0, sLongFile, Left(sShortPathName, lRetVal))
    If Not StringEmpty(sFile) Then
       GetShortName = GetShortName & sFile
    End If

    Exit Function

GetShortNameErr:
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "GetShortName"
    On Error GoTo 0
    Err.Raise vbObjectError + errApiFailed, gstrSource, _
          LoadResString(errApiFailed)

End Function
Public Function Determine64BitTime() As Currency

    Determine64BitTime = Get64BitTime(ByVal 0&)

End Function
Attribute VB_Name = "WorkspaceCommon"
'  FILE:      WorkspaceCommon.bas
'             Microsoft TPC-H Kit Ver. 1.00
'             Copyright Microsoft, 1999
'             All Rights Reserved
'
'
'  PURPOSE:    Contains functionality common across StepMaster and
'              SMRunOnly, pertaining to workspaces
'             Specifically, functions to read workspace records from
'             the database and so on.
'  Contact:    Reshma Tharamal (reshmat@microsoft.com)
'
Option Explicit

' Used to indicate the source module name when errors
' are raised by this module
Private Const mstrModuleName As String = "WorkspaceCommon."


Public Function GetWorkspaceDetails( _
    Optional ByVal WorkspaceId As Long, _
    Optional WorkspaceName As String = gstrEmptyString _
    ) As Variant
    ' Depending on the passed in parameter, it returns
    ' either the workspace name or the workspace identifier
    ' in a variant. The calling function must convert the
    ' return value to the appropriate type

    Dim rstWorkspace As Recordset
    Dim qyWsp As DAO.QueryDef
    Dim strSql As String
    Dim cTempStr As cStringSM

    On Error GoTo GetWorkspaceDetailsErr
    gstrSource = mstrModuleName & "GetWorkspaceDetails"

    If WorkspaceId = 0 And _
          WorkspaceName = gstrEmptyString Then
       On Error GoTo 0
       Err.Raise vbObjectError + errMandatoryParameterMissing, _
             gstrSource, _
             LoadResString(errMandatoryParameterMissing)
    End If

    Set cTempStr = New cStringSM

    If WorkspaceId = 0 Then
       strSql = " Select workspace_id from att_workspaces " & _
             " where workspace_name = [w_name] "
       Set qyWsp = dbsAttTool.CreateQueryDef(gstrEmptyString, strSql)
       qyWsp.Parameters("w_name").Value = WorkspaceName
    Else
       strSql = " Select workspace_name from att_workspaces " & _
             " where workspace_id = [w_id] "
       Set qyWsp = dbsAttTool.CreateQueryDef(gstrEmptyString, strSql)
       qyWsp.Parameters("w_id").Value = WorkspaceId
    End If

    Set cTempStr = Nothing

    Set rstWorkspace = qyWsp.OpenRecordset(dbOpenForwardOnly)

    If rstWorkspace.RecordCount <> 0 Then
       GetWorkspaceDetails = rstWorkspace.Fields(0)
    Else
       rstWorkspace.Close
       qyWsp.Close
       On Error GoTo 0
       Err.Raise vbObjectError + errInvalidWorkspaceData, _
             gstrSource, _
             LoadResString(errInvalidWorkspaceData)
    End If

    rstWorkspace.Close
    qyWsp.Close
    Exit Function

GetWorkspaceDetailsErr:
    Call LogErrors(Errors)
    gstrSource = mstrModuleName & "GetWorkspaceDetails"
    On Error GoTo 0
    Err.Raise vbObjectError + errGetWorkspaceDetailsFailed, _
          gstrSource, _
          LoadResString(errGetWorkspaceDetailsFailed)

End Function

Public Sub ReadStepsInWorkspace(rstStepsInWorkSpace As Recordset, _
       qySteps As DAO.QueryDef, _
       Optional lngWorkspaceId As Long = glInvalidId, _
       Optional dbLoad As DAO.Database = Nothing, _
       Optional ByVal bSelectArchivedRecords As Boolean = False)

    ' This function will populate the passed in recordset with
    ' all the steps for a given workspace (if one is passed in, else all workspaces)

    Dim strSql As String

    On Error GoTo ReadStepsInWorkspaceErr

    ' Create a recordset object to retrieve all steps for
    ' the given workspace
    strSql = "Select step_id, step_label, step_file_name, step_text, " & _
       " start_directory, version_no, workspace_id, " & _
       " parent_step_id, parent_version_no, " & _
       " sequence_no, step_level, " & _
       " enabled_flag, degree_parallelism, " & _
       " execution_mechanism, " & _
       " failure_details, continuation_criteria, " & _
       " global_flag, archived_flag, " & _
       " output_file_name, " & _
       " error_file_name, iterator_name " & _
       " from att_steps a " & _
       " where "

       ' log_file_name,

    If lngWorkspaceId <> glInvalidId Then
       strSql = strSql & " workspace_id = [w_id] AND "
    End If

    If Not bSelectArchivedRecords Then
       strSql = strSql & " archived_flag = [archived] AND "
```

```
    End If

    ' Find the highest X-component of the version number
    strSql = strSql & " cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) = " & _
        " ( select max( cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) ) " & _
        " from att_steps AS d " & _
        " WHERE a.step_id = d.step_id ) "

    ' Find the highest Y-component of the version number for the highest
X-component
    strSql = strSql & " AND cint( mid( version_no, instr( version_no, " & gstrDQ
& gstrVerSeparator & gstrDQ & " ) + 1 ) ) = " & _
        " ( select max( cint( mid( version_no, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) + 1 ) ) ) " & _
        " from att_steps AS b " & _
        " Where a.step_id = b.step_id " & _
        " AND cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) = " & _
        " ( select max( cint( mid( version_no, 1, instr( version_no, " & gstrDQ &
gstrVerSeparator & gstrDQ & " ) - 1 ) ) ) " & _
        " from att_steps AS c " & _
        " WHERE a.step_id = c.step_id ) ) "

    ' Append the order clause as follows
    ' First, separate all global/non-global steps
    ' Order the worker and manager steps by step_level to
    ' ensure that the parent steps are populated before
    ' any sub-steps within it
    ' Further ordering by parent_step_id and sequence_no
    ' ensures that all the children within a parent are
    ' selected in the necessary order
    strSql = strSql & " order by global_flag, step_level, " & _
        " parent_step_id, sequence_no "

    If dbLoad Is Nothing Then Set dbLoad = dbsAttTool

    ' Create a temporary Querydef object
    Set qySteps = dbLoad.CreateQueryDef(gstrEmptyString, strSql)

    ' Initialize the parameter values
    If lngWorkspaceId <> glInvalidId Then
        qySteps.Parameters("w_id").Value = lngWorkspaceId
    End If

    If Not bSelectArchivedRecords Then
        qySteps.Parameters("archived").Value = False
    End If

    Set rstStepsInWorkSpace = qySteps.OpenRecordset(dbOpenSnapshot)

    Exit Sub

ReadStepsInWorkspaceErr:

    LogErrors Errors
    gstrSource = mstrModuleName & "ReadStepsInWorkspace"
    On Error GoTo 0
    Err.Raise vbObjectError + errReadWorkspaceDataFailed, _
          gstrSource, _
          LoadResString(errReadWorkspaceDataFailed)

End Sub
Public Sub ReadWorkspaces(dbLoad As Database, rstWsp As Recordset, _
        qyWsp As DAO.QueryDef, _
        Optional ByVal bSelectArchivedRecords As Boolean = False)

    ' This function will populate the passed in recordset with all workspace records

    Dim strSql As String

    On Error GoTo ReadWorkspacesErr

    ' Create a recordset object containing all the workspaces
    ' (that haven't been archived) in the database
    strSql = " Select workspace_id, workspace_name, archived_flag " & _
          " from att_workspaces "

    If Not bSelectArchivedRecords Then
        strSql = strSql & " where archived_flag = [archived]"
    End If
    strSql = strSql & " order by workspace_name"

    Set qyWsp = dbLoad.CreateQueryDef(gstrEmptyString, strSql)
    If Not bSelectArchivedRecords Then
        qyWsp.Parameters("archived").Value = False
    End If

    Set rstWsp = qyWsp.OpenRecordset(dbOpenForwardOnly)

    Exit Sub

ReadWorkspacesErr:

    LogErrors Errors
    gstrSource = mstrModuleName & "ReadWorkspaces"
    On Error GoTo 0
    Err.Raise vbObjectError + errReadWorkspaceDataFailed, _
          gstrSource, _
          LoadResString(errReadWorkspaceDataFailed)

End Sub
Public Sub ShowWorkspacesInDb(dbLoad As Database)

    Dim recWorkspaces As Recordset
    Dim qryAllWsp As QueryDef

    On Error GoTo ShowWorkspacesInDbErr

    ' Set the mousepointer to indicate Busy
    Call ShowBusy

    Load frmWorkspaceOpen

    Call ReadWorkspaces(dbLoad, recWorkspaces, qryAllWsp)

    frmWorkspaceOpen.lstWorkspaces.Clear

    ' Load all the workspaces into the listbox
    If recWorkspaces.RecordCount <> 0 Then
        Do
            ' Add the workspace name to the list and store
            ' the corresponding workspace id as the ItemData
            ' property of the item.
            ' The workspace id will be used for all further
            ' processing of the workspace
            frmWorkspaceOpen.lstWorkspaces.AddItem
recWorkspaces![workspace_name]

frmWorkspaceOpen.lstWorkspaces.ItemData(frmWorkspaceOpen.lstWorkspaces
.NewIndex) = _
                recWorkspaces![workspace_id]
            recWorkspaces.MoveNext

        Loop Until recWorkspaces.EOF
    End If
    recWorkspaces.Close
    qryAllWsp.Close

    ' Reset the mousepointer
    ShowFree

    #If RUN_ONLY Then
        frmWorkspaceOpen.Show vbModal
    #Else
        frmWorkspaceOpen.Show vbModal, frmMain
    #End If

    Exit Sub

ShowWorkspacesInDbErr:
    LogErrors Errors
    Call ShowFree
    Err.Raise vbObjectError + errProgramError, mstrModuleName &
"ShowWorkspacesInDb", _
```

```
        LoadResString(errProgramError)

End Sub
Private Sub ReadWorkspaceParameters(lngWorkspaceId As Long, _
   rstWorkSpaceParameters As Recordset, _
   qyWspParams As DAO.QueryDef)

   ' Will populate the recordset with all the parameters for
   ' a given workspace

   Dim strSql As String

   On Error GoTo ReadWorkspaceParametersErr

   strSql = "Select parameter_id, parameter_name, " & _
         " parameter_value, workspace_id, parameter_type, description " & _
         " from workspace_parameters " & _
         " where workspace_id = [w_id] " & _
         " order by parameter_name, parameter_value "

   ' Create a temporary Querydef object and initialize
   ' it's parameter values
   Set qyWspParams = dbsAttTool.CreateQueryDef(gstrEmptyString,strSql)
   qyWspParams.Parameters("w_id").Value = lngWorkspaceId

   Set rstWorkSpaceParameters =
qyWspParams.OpenRecordset(dbOpenSnapshot)

   Exit Sub

ReadWorkspaceParametersErr:

   LogErrors Errors
   gstrSource = mstrModuleName & "ReadWorkspaceParameters"
   On Error GoTo 0
   Err.Raise vbObjectError + errReadWorkspaceDataFailed, _
         gstrSource, _
         LoadResString(errReadWorkspaceDataFailed)

End Sub
Private Sub ReadConnections(lngWorkspaceId As Long, rstConns As Recordset,
_
   qyConns As DAO.QueryDef)

   ' Will populate the recordset with all the parameters for
   ' a given workspace

   Dim strSql As String

   On Error GoTo ReadWorkspaceParametersErr

   strSql = "Select connection_id, " & _
         " connection_name, connection_value, workspace_id, description, " & _
         " no_count_display, no_execute, parse_query_only,
ANSI_quoted_identifiers, " & _
         " ANSI_nulls, show_query_plan, show_stats_time, show_stats_io, " & _
         " parse_odbc_msg_prefixes, row_count, tsql_batch_separator,
query_time_out, " & _
         " server_language, character_translation, regional_settings " & _
         " from workspace_connections " & _
         " where workspace_id = [w_id] " & _
         " order by connection_name, connection_value "

   ' Create a temporary Querydef object and initialize
   ' it's parameter values
   Set qyConns = dbsAttTool.CreateQueryDef(gstrEmptyString,strSql)
   qyConns.Parameters("w_id").Value = lngWorkspaceId

   Set rstConns = qyConns.OpenRecordset(dbOpenSnapshot)

   Exit Sub

ReadWorkspaceParametersErr:

   LogErrors Errors
   On Error GoTo 0
   Err.Raise vbObjectError + errReadWorkspaceDataFailed, _
         mstrModuleName & "ReadConnections",
LoadResString(errReadWorkspaceDataFailed)

End Sub
Private Sub ReadConnectionDtls(lngWorkspaceId As Long, rstConns As
Recordset, _
   qyConns As DAO.QueryDef)

   ' Will populate the recordset with all the connection_dtls records for
   ' a given workspace

   Dim strSql As String

   On Error GoTo ReadWorkspaceParametersErr

   strSql = "Select " & FLD_ID_CONN_NAME & ", " & _
         FLD_CONN_DTL_CONNECTION_NAME & ", " & _
         FLD_CONN_DTL_CONNECTION_STRING & ", " & _
         FLD_ID_WORKSPACE & ", " & _
         FLD_CONN_DTL_CONNECTION_TYPE & _
         " from " & TBL_CONNECTION_DTLS & _
         " where " & FLD_ID_WORKSPACE & " = [w_id] " & _
         " order by " & FLD_CONN_DTL_CONNECTION_NAME

   ' Create a temporary Querydef object and initialize
   ' it's parameter values
   Set qyConns = dbsAttTool.CreateQueryDef(gstrEmptyString,strSql)
   qyConns.Parameters("w_id").Value = lngWorkspaceId

   Set rstConns = qyConns.OpenRecordset(dbOpenSnapshot)

   Exit Sub

ReadWorkspaceParametersErr:

   LogErrors Errors
   On Error GoTo 0
   Err.Raise vbObjectError + errReadWorkspaceDataFailed, _
         mstrModuleName & "ReadConnectionDtls",
LoadResString(errReadWorkspaceDataFailed)

End Sub

Public Sub ReadWorkspaceData(lngWorkspaceId As Long, _
      cStepsCol As cArrSteps, _
      cParamsCol As cArrParameters, _
      cConsCol As cArrConstraints, _
      cConns As cConnections, _
      cConnDetails As cConnDtls, _
      rstStepsInWsp As Recordset, _
      qyStepsInWsp As DAO.QueryDef, _
      rstParamsInWsp As Recordset, _
      qyParamsInWsp As DAO.QueryDef, _
      rstConns As Recordset, _
      qyConns As DAO.QueryDef, _
      rstConnDtls As Recordset, _
      qyConnDtls As DAO.QueryDef)
   ' Loads the passed in structures with all the data for
   ' the workspace. It also initializes the recordsets
   ' with the step and parameter records for the workspace.

   On Error GoTo ReadWorkspaceDataErr

   ShowBusy

   Call ReadStepsInWorkspace(rstStepsInWsp, qyStepsInWsp, lngWorkspaceId)

   ' Load all the steps in the array
   LoadRecordsetInStepsArray rstStepsInWsp, cStepsCol

   ' Initialize the steps with all the iterator
   ' records for each step
   Call LoadIteratorsForWsp(cStepsCol, lngWorkspaceId, rstStepsInWsp)

   ReadWorkspaceParameters lngWorkspaceId, rstParamsInWsp,
qyParamsInWsp

   ' Load all the workspace parameters in the array
```

```
    LoadRecordsetInParameterArray rstParamsInWsp, cParamsCol

    ' Read and load connection strings
    ReadConnections lngWorkspaceId, rstConns, qyConns

    LoadRecordsetInConnectionArray rstConns, cConns

    ' Read and load connection information
    ReadConnectionDtls lngWorkspaceId, rstConnDtls, qyConnDtls

    LoadRSInConnDtlArray rstConnDtls, cConnDetails

    ' Finally, load the step constraints collection class with
    ' all the constraints for the steps in the workspace
    cConsCol.LoadConstraints lngWorkspaceId, rstStepsInWsp

    ShowFree
    Exit Sub

ReadWorkspaceDataErr:
    ' Log the error code raised by Visual Basic
    ShowFree
    Call LogErrors(Errors)
    On Error GoTo 0
    gstrSource = mstrModuleName & "ReadWorkspaceData"
    Err.Raise vbObjectError + errReadWorkspaceDataFailed, _
          gstrSource, _
          LoadResString(errReadWorkspaceDataFailed)

End Sub
```

Appendix H: Price Quotations

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399

Tel 425 882 8080
Fax 425 936 7329
http://www.microsoft.com/

Microsoft

March 15, 2001

IBM
Tricia Thomas
D23U/B060/Office E127
3039 Cornwallis Road
Research Triangle Park, NC 27709

Tricia:

Here is the information you requested regarding pricing for several Microsoft products to be used in conjunction with your TPC-H benchmark testing.

All pricing shown is in US Dollars ($).

Part Number	Description	Price
810-00652	**SQL Server 2000 Enterprise Edition** *Server license only - No CALs* *Discount schedule: Open Program - No Level*	$ 5,549
359-00532	**SQL Server 2000 Client License** *50 Client Licenses @ $146.00 each* *Discount schedule: Open Program - No Level*	$ 7,300
C10-00475	**Windows 2000 Advanced Server** *Server license only - No CALs* *Discount schedule: Open Program - No Level*	$ 2,399
659-00390	**Visual Studio Professional 6.0 Win32**	$ 1,079
	5-year maintenance for above software *($2095 per year)*	$ 10,475

All products are currently orderable through Microsoft's normal distribution channels.

This quote is valid for the next 90 days.

If we can be of any further assistance, please contact Jamie Reding at
(425) 703-0510 or jamiere@microsoft.com.

Reference ID: Pupbt0106026198
Please include this Reference ID in any correspondence regarding this price quote.

Mylex/IBM Extreme RAID 2000 Quotation

Ms. Chris King, March 1, 2001
IBM Netfinity Performance Group

Dear Ms. King,

Mylex is pleased to submit the following quotation for ExtremeRAID 2000 controller.

==========

Mylex P/N :	**Description**
E2000-4-32NB	(4 external +2 internal chnl, 32MB cache, no BBU)

Suggested Retail Price: $1,872.00 ea

Notes: Above price is based on FOB, ex factory, Fremont, California and firm for 90 days.

Lead time: 45 Days ARO

Product is covered by 5 year warranty.

Failed product will be repaired or replaced within 7 days.

Regards,

Robert Kelly - Director, Strategic Sales

Cc: Steve Page - Director PCI Marketing