TPC Benchmark™ C
Full Disclosure Report
For
Acer Incorporated AcerAltos 21000
Using
Microsoft SQL Server 7.0 Enterprise Edition
And
Microsoft Windows NT Server 4.0 Enterprise Edition

First Edition
Submitted for Review
May 17, 1999

First Printing, May 17, 1999

Acer Inc. believes that the information included in this document is accurate as of the publication date. The information in this document is subject to change without notice. Furthermore, Acer Inc. is not responsible for any errors contained within this document.

The pricing information given in this FDR is accurate as of the publication date, May 17, 1999. However, Acer Inc. provides no warranty of the pricing information in this document.

Benchmark results are highly dependent upon workload, specific application requirements, and system design and implementation. Relative system performance will vary as a result for these and other factors. Therefore, TPC Benchmark C should not be used as a substitute for a specific customer application benchmark when critical capacity planning and/or product evaluation decisions are contemplated.

All performance data contained in this report were obtained in a rigorously controlled environment. Results obtained in other operating environments may vary significantly. Acer Inc. does not warrant or represent that a user can or will achieve similar performance expressed in transactions per minute (tpmC) or normalized price/performance ($/tpmC). No warranty of system performance or price/performance is expressed or implied in this report.

AcerAltos is a trademark of the Acer Inc.

Microsoft, Windows NT and SQL Server for Windows NT are registered trademarks of Microsoft Corporation.

TPC Benchmark, TPC-C and tpmC are registered trademarks of the Transaction Processing Performance Council.

Intel and Pentium is a registered trademark of Intel.

Other product names mentioned in this document may be trademarks and/or registered trademarks of their respective companies.

Abstract

Overview

This report documents the methodology and results of the TPC Benchmark™ C test conducted on the Acer Inc. AcerAltos 21000. The tests were run in a client/server configuration using 3 AcerAltos 9100B as clients. The operating system used for the benchmark was Microsoft Windows NT Server 4.0 Enterprise Edition for the server and Windows NT Server 4.0 on the clients. The database was the Microsoft SQL Server 7.0 Enterprise Edition. Tuxedo 6.4 Core Functional Services (CFS) provided the database connection queues. All tests were done in compliance with Revision 3.4 of the Transaction Processing Council's TPC Benchmark ™ C Standard Specification. Two standard TPC Benchmark™ C metrics, transactions per second (tpmC) and price per tpmC ($/tpmC) are reported and referred to in this document. The results from the tests are summarized below.

Hardware	Software	Total System Cost	tpmC	$/tpmC	Availability Date
Acer Inc. AcerAltos 21000	Microsoft Windows NT Server 4.0 Enterprise Edition. Microsoft SQL Server 7.0 Enterprise Edition Tuxedo 6.4 CFS	**$387,010**	**23,235.57**	**$16.66**	HW: May 17, 1999 SW: May 17, 1999

AUDITOR

The results of the benchmark and test methodology used to produce the results were audited by Tom Sawyer of Performance Metrics, Inc. and have fully met the TPC-C rev 3.4 specifications.

Additional copies of this Full Disclosure Report can be obtained from either the Transaction Processing Performance Council or Acer Inc. at the following address:

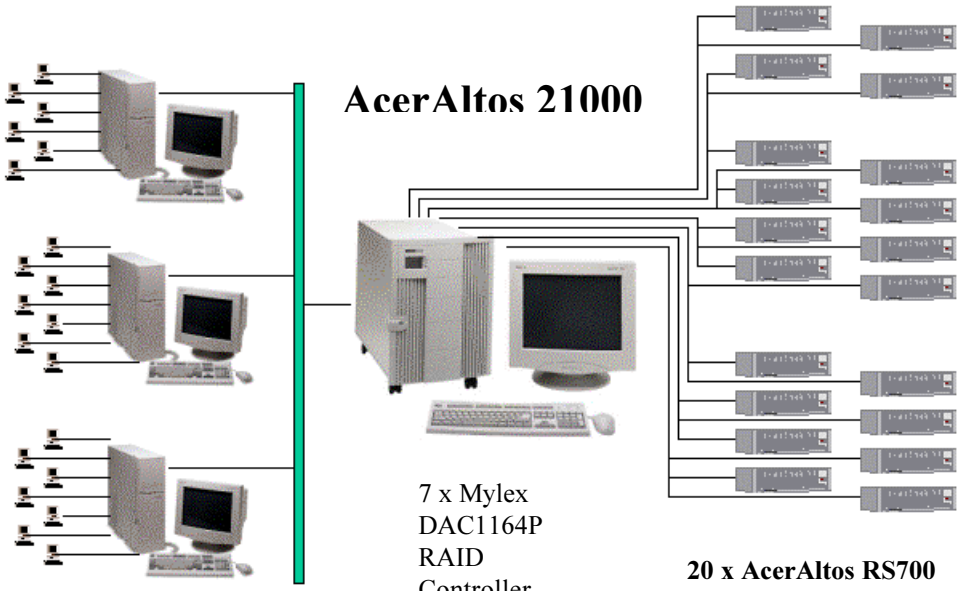
Transaction Processing Performance Council (TPC)
c/o Shanley Public Relations
777 North First Street, Suite 600
San Jose, CA 95112, USA
Phone: (408) 295-8894, fax 295-9768

or

Acer Inc.
21F, 88, Sec. 1, Hsin Tai Wu Rd.,
Hsichih, Taipei Hsien 221,
Taiwan, R.O.C.
Phone: (02)8691-1530, fax 8691-2379

ACER	AcerAltos 21000 **Client/Server w/ 3 AcerAltos 9100B Front Ends**		TPC-C Rev 3.4 Report Date May 17, 1999
Total System Cost	TPC-C Throughput	Price/Performance	Availability Date
$387,010	**23,235.57 tpmC**	**$16.66**	May 17, 1999

Processors	Database Manager	Operating System	Other Software	Number of Users
4 × Pentium III Xeon Processors 500 MHz 2MB L2 cache	Microsoft SQL Server 7.0 Enterprise Edition	Microsoft Windows NT Server 4.0 Enterprise Edition	Microsoft Internet Information Server Microsoft Visual C++ Tuxedo 6.4 CFS	18,800



System Component	Server		Each Clients	
Processors Cache	4	Pentium III Xeon 500 MHz 2 MB	1	Pentium II 450 MHz 512 KB
Memory	1	4096 MB	1	512 MB
Disk Controllers	7	Mylex DAC1164P		Adaptec On-Board AIC-7880P
Disk Drives	193 8	9 GB 18 GB	1	4 GB
Total Storage		1881 GB		4 GB
Other	1	CD-ROM	1	CD-ROM

AcerAltos 21000 Client/Server

TPC-C REV3.4 EXECUTIVE SUMMARY

Description	Part Number	Third Party	Unit Price	Qty	Extended Price	5 yr. Maint Price
Server Hardware						
Altos 21000 w/ CD-Rom 40x, keyboard, mouse, 0 Proc , 0MB Mem, 8 hot-swap drive bays, Three 430W power supplies, APC smart UPS 1000	91.AB704.201	1	$9,524	1	$9,524	1,429
Pentium III Xeon 500MHz/2MB Cache	91.AB711,009	1	$5,576	4	$22,304	3,346
256MB EDO DIMM	71.84208.060	1	$545	16	$8,720	1,308
9GB 10,000rpm Ultra2 SCSI HDD	56.02961.121	1	$680	1	$680	102
DAT Tape Drive	91.AB359.001	1	$1,065	1	$1,065	160
14" AcerView Monitor	91.74302.004	1	$155	1	$155	23
				Subtotal	$42,448	6,367
Server Storage						
Mylex DAC1164P Controller w/ 32MB EDO		Mylex 2	$1,895	7	$13,265	$700
9GB 10,000rpm Ultra2 SCSI HDD	56.02961.121	1	$680	192	$130,560	$19,584
18GB 10,000rpm Ultra2 SCSI HDD	56.02A04.061	1	$1,219	8	$9,752	$1,463
RS700-RM112 Storage Box	81.54A28.501	1	$1,000	18	$18,000	$2,700
RS700-RM101 Storage Box	91.54A28.014	1	$1,100	2	$2,200	$330
External SCSI Cable	50.AB600.001	1	$65	20	$1,300	$195
				Subtotal	$175,077	$24,972
Server Software						
MS Windows NT Server 4.0 Enterprise Edtion, incl 25 CALs*		Microsoft 3	$3,999	1	$3,999	$0
MS SQL Server, Enterprise Edition 7.0, unlimited user license*		Microsoft 3	$28,999	1	$28,999	$10,475
				Subtotal	$32,998	$10,475
Client Hardware						
AcerAltos 9100B,1 Pentium II 450MHz, 4GB Ultra Wide SCSI HD, 32X SCSI CD-ROM, Keyboard, Mouse, 420W power supply	AA9100B-9450	1	$4,084	3	$12,252	$1,838
Pentium II 450MHz Processor Upgrade Kit	91.AB009.007	1	$784	3	$2,352	$353
128MB SDRAM	91.AB658.001	1	$286	12	$3,432	$515
14" AcerView color monitor	91.74302.004	1	$155	3	$465	$70
Intel Pro/100+ Dual Port Server Adapter	91.AB023.004	1	$225	12	$2,700	$405
				Subtotal	$21,201	$3,180
Client Software						
MS Windows NT Server 4.0, incl 5 CALs*		Microsoft 3	$809	3	$2,427	$0
MS Visual C++ 32-bit Edition(subscription)*		Microsoft 3	$499	1	$499	$0
TUXEDO Core Functional Services 6.4 for NT		BEA 4	$3,000	3	$9,000	$7,200
				Subtotal	$11,926	$7,200
User Connectivity						
Acer Netxus 100Base Ethernet Hub 8 port**	91.22009.00	1	$195	3	$585	spared
Acer Nexsus 10Base Ethernet Hub 8 port**	90.80516.41	1	$29	2596	$75,284	spared
				Subtotal	$75,869	$0
				Discounts	-$24,703	$0
				Total	$334,816	$52,194

Notes

* All Microsoft maintenance is covered by the maintenance cost of Microsoft SQL Server.

** 10% or minimum 2 spares are added in place of onsite service (product have a five year return-to-vendor warranty)

Pricing: 1=Acer, 2=Mylex, 3=Microsoft, 4=Bea

Five-Year Cost of Ownership:	**$387,010**
tpmC Rating:	**23235.57**
$/tpmC	**$16.66**

The benchmark results and test methodology were audited by Tom Sawyer of Performance Metrics, Inc.

Price 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 assumption about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed compoments. For complete details, see the pricing section of the TPC benchmarks specifications. If you find that stated prices are not available according to these term, please inform the TPC at pricing @tpc.org. Thank you.

MQTh, computed Maximum Qualified Throughput 23,235.57tpmC
% throughput difference, reported & reproducibility runs 0.1%

Response Times (90th percentile | Average | Maximum) in seconds

- Neworder	0.98	0.56	4.77
- Payment	0.85	0.43	4.75
- Order Status	0.91	0.50	4.68
- Delivery (interactive portion)	0.30	0.31	2.05
- Delivery (deferred portion)	1.0	0.599	5.157
- Stock-Level	3.03	2.30	7.51
- Menu	0.32	0.21	2.08

Transaction Mix, in percent of total transactions

- New-Order	44.94 %
- Payment	43.00 %
- Order-Status	4.03 %
- Delivery	4.01 %
- Stock-Level	4.01 %

Keying/Think Times (in seconds),	Min		Average		Max	
- New-Order	18.00	0.00	18.01	12.06	18.03	120.50
- Payment	3.00	0.00	3.01	12.03	3.03	120.50
- Order-Status	2.00	0.00	2.01	10.12	2.03	100.49
- Delivery	2.00	0.00	2.01	5.04	2.03	50.50
- Stock-Level	2.00	0.00	2.01	5.07	2.03	49.93

Test Duration

- Ramp-up time	30 minutes
- Measurement interval	30 minutes
- Number of checkpoints in measured interval	1
- Checkpoint interval	30 minutes
- Number of transactions (all types) completed in measurement interval	1,550,963

Introduction

Document Structure

The contents of this report are determined by the TPC Benchmark C Standard Specification Revision 3.4, written and approved by the Transaction Processing Performance Council (TPC). The format of this report is based on this specification. Most sections of this report begins with the relevant specification requirements printed in italic type, immediately followed by the detail in plain type of how Acer Inc. complied with the specification. Where extensive listings are required (such as listing of code), a note is included which references an appendix containing the listing.

Benchmark Overview

TPC Benchmark™ C (TPC-C) is an OLTP workload. It is a mixture of read-only and update intensive transactions that simulate the activities found in complex OLTP application environments. It does so by exercising a breadth of system components associated with such environments, which are characterized by:

- The simultaneous execution of multiple transaction types that span a breadth of complexity
- On-line and deferred transaction execution modes
- Multiple on-line terminal sessions
- Moderate system and application execution time
- Significant disk input/output
- Transaction integrity (ACID properties)
- Non-uniform distribution of data access through primary and secondary keys
- Databases consisting of many tables with a wide variety of sizes, attributes, and relationships
- Contention on data access and update

The performance metric reported by TPC-C is a "business throughput" measuring the number of orders processed per minute. Multiple transactions are used to simulate the business activity of processing an order, and each transaction is subject to a response time constraint.

The performance metric for this benchmark is expressed in transactions-per-minute-C (tpmC). To be compliant with the TPC-C standard, all references to tpmC results must include the tpmC rate, the associated price-per-tpmC, and the availability date of the priced configuration.

Although these specifications express implementation in terms of a relational data model with conventional locking scheme, the database may be implemented using any commercially available database management system (DBMS), database server, file system, or other data repository that provides a functionally equivalent implementation. The terms "table", "row", and "column" are used in this document only as examples of logical data structures.

TPC-C uses terminology and metrics that are similar to other benchmarks, originated by the TPC or others. Such similarity in terminology does not in any way imply that TPC-C results are comparable to other benchmarks. The only benchmark results comparable to TPC-C are other TPC-C results conformant with the same revision.

Despite the fact that this benchmark offers a rich environment that emulates many OLTP applications, this benchmark does not reflect the entire range of OLTP requirements. In addition, the extent to which a customer can achieve the results reported by a vendor is highly dependent on how closely TPC-C approximates the customer application. The relative performance of systems derived from this benchmark

does not necessarily hold for other workloads or environments. Extrapolations to any other environment are not recommended.

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

System Overview

The hardware configuration used in this TPC-C test is a AcerAltos 21000 Server driven by 3 AcerAltos 9100B clients. The clients and server are networked together via a 100BaseT hub. Five remote terminal emulators (RTE), AcerAltos 1100, emulated 18,800 users executing the standard TPC-C workload. The RTEs were connected to the three clients. Microsoft Windows NT Server 4.0 Enterprise Edition was the operating system used on the server and Windows NT Server 4.0 Standard Edition on the clients. Microsoft SQL Server 7.0 Enterprise Edition was the database on the server machine.

Each of the clients has two 450MHz Pentium II processors with 512 KB of L2 cache, 512 MB of RAM, one 4 GB SCSI hard disk, and four Intel PRO100/+ Dual-Port Server Adapters. On each client, one port of the Intel Ethernet adapters was connected to the server through a 100BaseT hub. The other ports of Intel Ethernet adapters provided 6 to 7 network segments to the RTE machines, with 940 emulated users per segment.

General Items

Test Sponsor

A statement identifying the sponsor of the Benchmark and any other companies who have participated.

Acer Inc. was the test sponsor of this TPC Benchmark™ C.

Application Code and Definition Statements

The application program must be disclosed. This includes, but is not limited to, the code implementing the five transactions and the terminal input/output functions.

The Section 3.0 entitled Clause 3 Related Items contains a brief discussion of the database design and loading. The database definition statements, distribution across disk drives, loading scripts, and tables are provided in Appendix B-Database Design.

The program that implements the TPC Benchmark C translation and collects appropriate transaction statistics is referred to as the Remote Terminal Emulator (RTE) or Driver program. The Driver program is discussed in Section 7.0.

Parameter Settings

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

- *Database options*
- *Recover/commit options*
- *Consistency/locking options*
- *System parameter, application parameters, and configuration parameters.*

This requirement can be satisfied by providing a full listing of all parameters and options.

Appendix C contains all the database and operating system parameters used in this benchmark. Appendix C contains all the hardware configuration details.

Configuration Diagrams

Diagrams of both the measured priced system must be provided, accompanied by a description of the differences.

Figure 1 and 2 respectively show the measured and priced full client/server configurations. The SUT in the measured system and the priced system are the same.

Figure 1: Measured Configuration

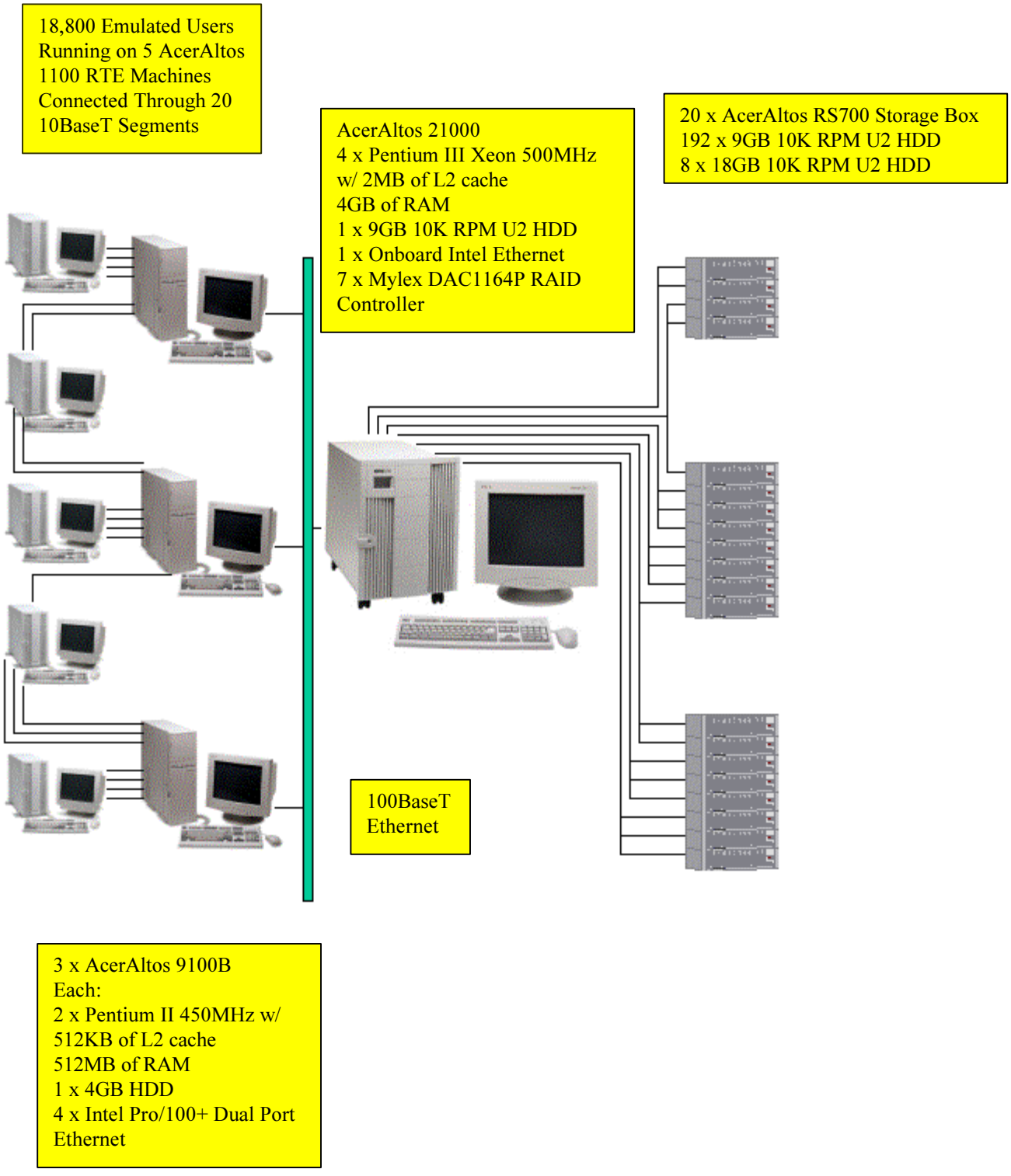
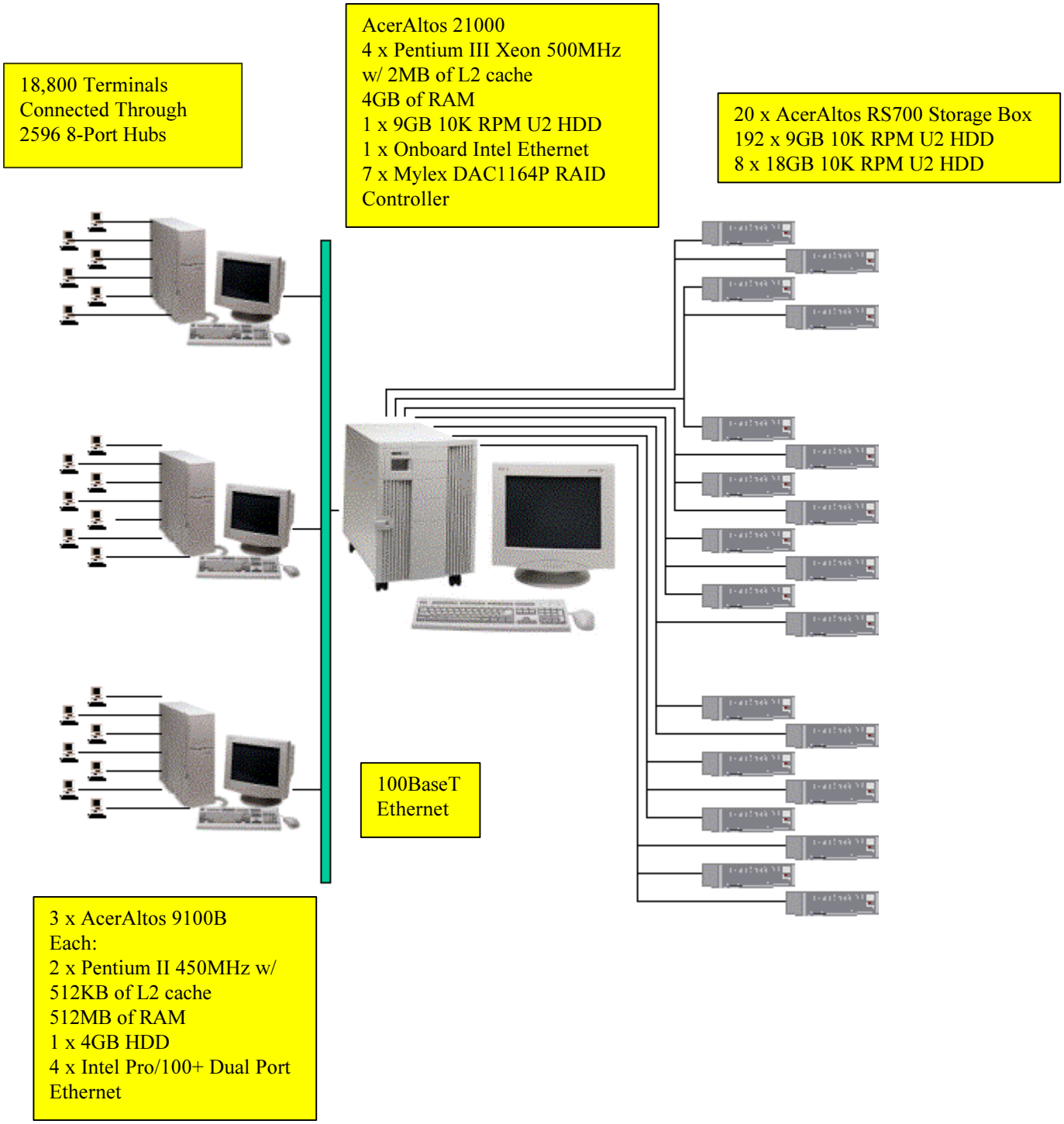


Figure 2: Priced Configuration



Clause 1 -- Logical Database Design Related Items

Table Definitions

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

Appendix B contains the code used to define and load the database tables.

Physical Organization of the Database

The physical organization of tables and indices, within the database, must be disclosed. (8.1.2.2)

The measured configuration used 201 disk drives, 193-9GB drives and 8-18GB drives. The organization is shown in Table 5: Data Distribution and in Appendix B.

Insert and Delete Operations

It must be ascertained that insert and/or delete operations to any of the tables can occur concurrently with the TPC-C transaction mix. Furthermore, any restriction in the SUT database implementation that precludes inserts beyond the limits defined in Clause 1.4.11 must be disclosed. This includes the maximum number of rows that can be inserted and the maximum key value for these new rows. (8.1.2.3)

Insert and delete functionality was fully operational during the benchmark.

Horizontal and Vertical Partitioning

While there are a few restrictions placed upon horizontal or vertical partitioning of tables and rows in the TPC-C benchmark (see Clause 1.6), any such partitioning must be disclosed. (8.1.2.4)

Partitioning was not used in this benchmark.

Replication

Replication of tables, if used, must be disclosed (see Clause 1.4.6). (8.1.2.5)

Replication was not used in this benchmark.

Table Attributes

Additional and/or duplicated attributes in any table must be disclosed along with a statement on the impact on performance (see Clause 1.4.7). (8.1.2.6)

No additional attributes were used in this benchmark.

Clause 2 -- Transaction and Terminal Profiles Related Items

Random Number Generation

The method of verification for the random number generation must be described. (8.1.3.1)

The seeds and offsets for the random number generator were collected and verified to be different for each driver. The auditor selected samples of the generated numbers from the database. The samples were verified to have no discernible patterns.

Screen Layout

The actual layouts of the terminal input/output screens must be disclosed. (8.1.3.2)

The screen layouts are based on those in Clauses 2.4.3, 2.5.3, 2.6.3, 2.7.3, and 2.8.3 of the TPC-C Standard Specification. There are some differences based on the fact that this is a WEB client implementation.

Terminal Verification

The method used to verify that the emulated terminals provide all the features described in Clause 2.2.2.4 must be explained. Although not specifically priced, the type and model of the terminals used for the demonstration in 8.1.3.3 must be disclosed and commercially available (including supporting software and maintenance). (8.1.3.3)

The terminal features were verified by allowing the auditor to manually execute each of the five transaction types, using Microsoft Internet Explorer version 3.0.

Intelligent Terminals

Any usage of presentation managers or intelligent terminals must be explained. (8.1.3.4)
***Comment 1**: The intent of this clause is to describe any special manipulations performed by a local terminal or workstation to off-load work from the SUT. This includes, but is not limited to: screen presentations, message bundling, and local storage of TPC-C rows.*
***Comment 2**: This disclosure also requires that all data manipulation functions performed by the local terminal to provide navigational aids for transaction(s) must also be described. Within this disclosure, the purpose of such additional function(s) must be explained.*

Application code involved in the manipulation of data was run on the client. Screen manipulation commands in the form of HTML were downloaded to the WEB browser which handled input and output presentation graphics. A listing of this code is included in Appendix A. Microsoft Internet Information Service assisted in the processing and presentation of this data.

Transaction Profiles

The percentage of home and remote order-lines in the New-Order transactions must be disclosed. (8.1.3.5)
The percentage of New-Order transactions that were rolled back as a result of an unused item number must be disclosed. (8.1.3.6)
The number of items per orders entered by New-Order transactions must be disclosed. (8.1.3.7)
The percentage of home and remote Payment transactions must be disclosed. (8.1.3.8)
The percentage of Payment and Order-Status transactions that used non-primary key (C_LAST) access to the database must be disclosed. (8.1.3.9)
The percentage of Delivery transactions that were skipped as a result of an insufficient number of rows in the NEW-ORDER table must be disclosed. 8.1.3.10)

Table 1: Transaction Statistics

Transaction	Function	Value
New Order	Home Warehouse Items	99.00%
	Remote Warehouse Items	1.00.%
	Rolled Back Transactions	1.02%
	Average Lines Per Order	10.00%
Payment	Home Warehouse	85.05%
	Remote Warehouse	14.95%
	Non-Primary Key Access	60.04%
Order Status	Non-Primary Key Access	60.17%
Delivery	Skipped Transactions	0

Transaction Mix

The mix (i.e., percentages) of transaction types seen by the SUT must be disclosed. (8.1.3.11)

Table 2: Transaction Mix

Transaction	Percentage
New Order	44.94%
Payment	43.00%
Order Status	4.01%
Delivery	4.03%
Stock Level	4.01%

Deferred Delivery Mechanism

The queuing mechanism used to defer the execution of the Delivery transaction must be disclosed. (8.1.3.12)

The client application submits delivery transactions to asynchronous Tuxedo servers, or queues, running on the client machines. There were multiple delivery servers with single execution threads running on each client machine. These delivery servers were responsible for processing deliveries queued to Tuxedo and submitting them to the database server.

The source code is listed in Appendix A.

Clause 3 -- Transaction and System Properties Related Items

ACID Tests

The results of the ACID tests must be disclosed along with a description of how the ACID requirements were met. This includes disclosing which case was followed for the execution of Isolation Test 7. (8.1.4.1)

All ACID property tests were successful. The executions are described below.

Atomicity

The system under test must guarantee that the database 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.

Completed Transactions

A row was selected in a script from the warehouse, district and customer tables, and the balances noted. A payment transaction was started with the same warehouse, district and customer identifiers and a known amount. The payment transaction was committed and the rows were verified to contain correctly updated balances.

Aborted Transactions

A row was selected in a script from the warehouse, district and customer tables, and the balances noted. A payment transaction was started with the same warehouse, district and customer identifiers and a known amount. The payment transaction was rolled back and the rows were verified to contain the original balances.

Consistency

Consistency is the property of the application that requires any execution of a database transaction to take the database from one consistent state to another, assuming that the database is initially in a consistent state.

Consistency conditions one through four were tested using a shell script to issue queries to the database. The results of the queries verified that the database was consistent for all four tests. A run was executed under full load lasting over ten (10) minutes and included a checkpoint. The shell script was executed again. The result of the same queries verified that the database remained consistent after the run.

Isolation

Sufficient conditions must be enabled at either the system or application level to ensure the required isolation defined above (clause 3.4.1) is obtained.

Isolation tests one through nine were executed using shell scripts to issue queries to the database. Each script included timestamps to demonstrate the concurrency of operations. The results of the queries were captured to files. The captured files were verified by the auditor to demonstrate the required isolation had been met.

For Isolation test seven, case A was followed.

Durability

The tested system must guarantee durability: the ability to preserve the effects of committed transaction and insure database consistency after recovery from any one of the failures listed in Clause 3.5.3.

Durable Media Failure

Durability from media failure was demonstrated on a 10 warehouse database having similar characteristics to the fully scaled database. The standard driving mechanism was used to generate the transaction load of 100 users for the loss of log and loss of data tests. The fully scaled database under full load would also have passed the following test.

Loss of Log and Loss of Data

The loss of log and loss of data tests were performed on a 10 warehouse system. The standard driving mechanism was used to generate the transaction load of 100 users for the test. To demonstrate recovery from a permanent failure of durable medium containing TPC-C tables, the following steps were executed:

1. A 10 warehouse database was built in a manner similar to the full size database.
2. A backup was taken of the database using Microsoft SQL Server facilities.
3. A sum of d_next_o_id was taken and noted.
4. 100 users were logged into the database and proceeded to run transactions.
5. One mirrored log drive was removed from the system. The system continued running.
6. One disk drive was removed from a disk array containing database files.
7. SQL Server reported errors and transactions failed.
8. SQL Server was restarted and a dump of the transaction log was taken.
9. The tpcc database was dropped.
10. The system was shut down and the faulty disk drive was replaced.
11. The 10 warehouse database was restored from backup.
12. The saved transaction log was loaded and transactions rolled forward.
13. A sum of d_next_o_id was taken and noted.
14. The number of transactions reported in the database and in the RTE log was compared.

Instantaneous Interruption and Loss of Memory

Because loss of power erases the contents of memory, the instantaneous interruption and the loss of memory tests were combined into a single test. This test was executed on a fully scaled database of 1,880 warehouses under a full load of 18,800 users. The following steps were executed:

1. A sum of d_next_o_id was taken and noted.
2. 18,800 users were logged into the database and proceeded to run transactions.
3. System power was terminated.
4. System power was reapplied and NT restarted.
5. SQL Server was started and recovery occured automatically.
6. A sum of d_next_o_id was taken and noted.
7. The number of transactions reported in the database and in the RTE log was compared.

Clause 4 -- Scaling and Database Population Related Items

Table Cardinality

The cardinality (e.g., the number of rows) of each table, as it existed at the start of the benchmark run (see Clause 4.2), must be disclosed. If the database was over-scaled and inactive rows of the WAREHOUSE table were deleted (see Clause 4.2.2), the cardinality of the WAREHOUSE table as initially configured and the number of rows deleted must be disclosed. (8.1.5.1)

Table 3: Table Cardinality

Table	Cardinality as Benchmarked
Warehouse	1900
District	19000
Customer	57000000
History	57000000
Orders	57000000
New Order	17100000
Order Line	570001928
Stock	190000000
Item	100000

Constant Values

The following values were used as constant value inputs to the NURand function for this benchmark.

Table 4: Constant Values

Function	Constant C Value
C_LAST (Build)	123
C_LAST (Run)	208

Data Distribution

The distribution of tables and logs across all media must be explicitly depicted for the tested and priced systems. (8.1.5.2)

Table 5 : Data Distribution

Controller	Drives	Partition	Size	Use
RAID Controller 1	9GB	C: & D:	4095/4581MB	OS, SQL and MSSQL70_tpcc_root
	18GB	E:	45000 MB	MSSQL70_tpcc_log
RAID Controller 2	9GB	F:	17950 MB	MSSQL70_cs1
		L:	9500 MB	MSSQL70_misc1
		R:	60001 MB	Tpccback1
RAID Controller 3	9GB	G:	17950 MB	MSSQL70_cs2
		M:	9500 MB	MSSQL70_misc2
		S:	60001 MB	Tpccback2
RAID Controller 4	9GB	H:	17950 MB	MSSQL70_cs3
		N:	9500 MB	MSSQL70_misc3
		T:	60001 MB	Tpccback3
RAID Controller 5	9GB	I:	17950 MB	MSSQL70_cs4
		O:	9500 MB	MSSQL70_misc4
		U:	60001 MB	Tpccback4
RAID Controller 6	9GB	J:	17950 MB	MSSQL70_cs5
		P:	9500 MB	MSSQL70_misc5
RAID Controller 7	9GB	K:	17950 MB	MSSQL70_cs6
		Q:	9500 MB	MSSQL70_misc6

***Comment**: Detailed diagrams for layout of database files on disks can widely vary, and it is difficult to provide exact guideline suitable for all implementations. The intent is to provide sufficient detail to allow independent reconstruction of the test database. The two figures below are examples of database layout descriptions and are not intended to depict or imply any optimal layout for the TPC-C database.*
8.1.5.3 A statement must be provided that describes:

1. *The data model implemented by the DBMS used (e.g., relational, network, hierarchical)*
2. *The database interface (e.g., embedded, call level) and access language (e.g., SQL, DL/1, COBOL read/write) used to implement the TPC-C transactions. If more than one interface/access language is used to implement TPC-C, each interface/access language must be described and a list of which interface/access language is used with which transaction type must be disclosed.*

Microsoft SQL Server 7.0 is a relational DBMS.

The interface used was Microsoft SQL Server stored procedures accessed with Remote Procedure Calls embedded in C code using the Microsoft DBLIB interface.

Partition Mapping

The mapping of database partitions/replications must be explicitly described.
***Comment**: The intent is to provide sufficient detail about partitioning and replication to allow*

independent reconstruction of the test database. (8.1.5.4)
An description of a database partitioning scheme is presented below as an example. The nomenclature of this example was outlined using the CUSTOMER table (in Clause 8.1.2.1), and has been extended to use the ORDER and ORDER_LINE tables as well.

The database was not replicated.

180 Day Space Calculation

Details of the 180 day space computations along with proof that the database is configured to sustain 8 hours of growth for the dynamic tables (Order, Order-Line, and History) must be disclosed (see Clause 4.2.3). (8.1.5.5)

1. To calculate the space required to sustain the database log for 8 hours of growth at steady state, the following steps were followed:
2. The free space on the log file was queried using *DBCC sqlperf(logspace)*.
3. Transactions were run against the database with a full load of users.
4. The free space was again queried using *DBCC sqlperf(logspace)*.
5. The space used was calculated as the difference between the first and second query.
6. The number of NEW-ORDERS was verified from an RTE report covering the entire run.
7. The space used was divided by the number of NEW-ORDERS giving a spaceused per NEW-ORDER transaction.
8. The space used per transaction was multiplied by the measured tpmC rate times 480 minutes.

The results of the above steps yielded a requirement of 113GB(including mirror) to sustain the log for 8 hours. Space available on the transaction log volume was 138GB (including mirror), indicating that enough storage was configured to sustain 8 hours of growth.
The same methodology was used to compute growth requirements for dynamic tables Order, Order-Line and History.

The details of the 180-day space requirement is shown in Appendix D.

Clause 5 -- Performance Metrics and Response Time Related Items

Measured tpmC

Measured tpmC must be reported. (8.1.6.1)

Measured tpmC	**23,235.57 tpmC**
Price per tpmC	**$16.66 / tpmC**

Response Times

Ninetieth percentile, maximum and average response times must be reported for all transaction types as well as for the Menu response time. (8.1.6.2)

Table 5: Transaction Response Times

Transaction	Average	Maximum	90%
New Order	0.56	4.77	0.98
Payment	0.43	4.75	0.85
Order Status	0.50	4.68	0.91
Interactive Delivery	0.31	2.05	0.30
Deferred Delivery	0.599	5.157	1.0
Stock Level	2.30	7.51	3.03
Menu	0.21	2.08	0.32

Think Times & Key Times

The minimum, the average, and the maximum keying and think times must be reported for each transaction type. (8.1.6.3)

Table 6: Transaction Key Times

Transaction	Average	Max	Min
New Order	18.01	18.03	18.00
Payment	3.01	3.03	3.00
Order Status	2.01	2.03	2.00
Delivery	2.01	2.03	2.00
Stock Level	2.01	2.03	2.00

Table 7: Transaction Think Times

Transaction	Average	Max	Min
New Order	12.06	120.50	0.00
Payment	12.03	120.50	0.00
Order Status	10.12	100.49	0.00
Delivery	5.04	50.50	0.00
Stock Level	5.07	49.93	0.00

Response Time Distribution Curves

Response Time frequency distribution curves (see Clause 5.6.1) must be reported for each transaction type. (8.1.6.4)

Figure 3: New Order Response Time Distribution

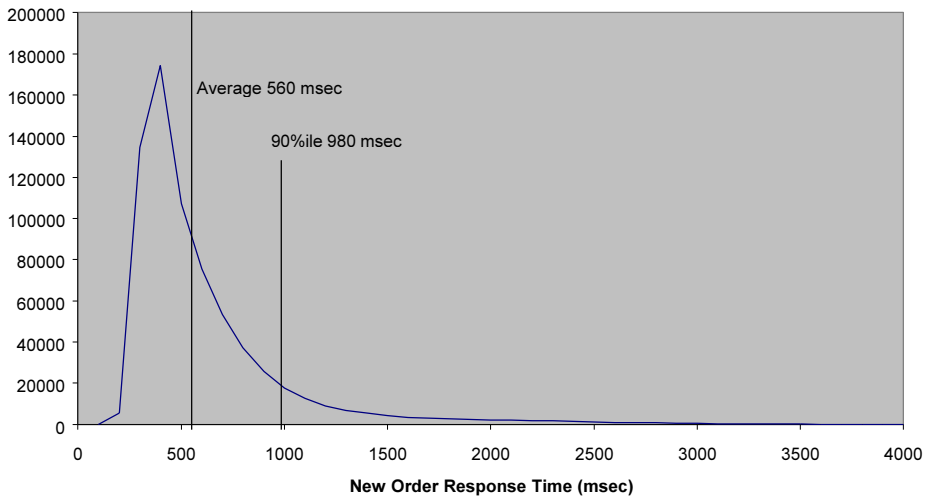


Figure 4: Payment Response Time Distribution

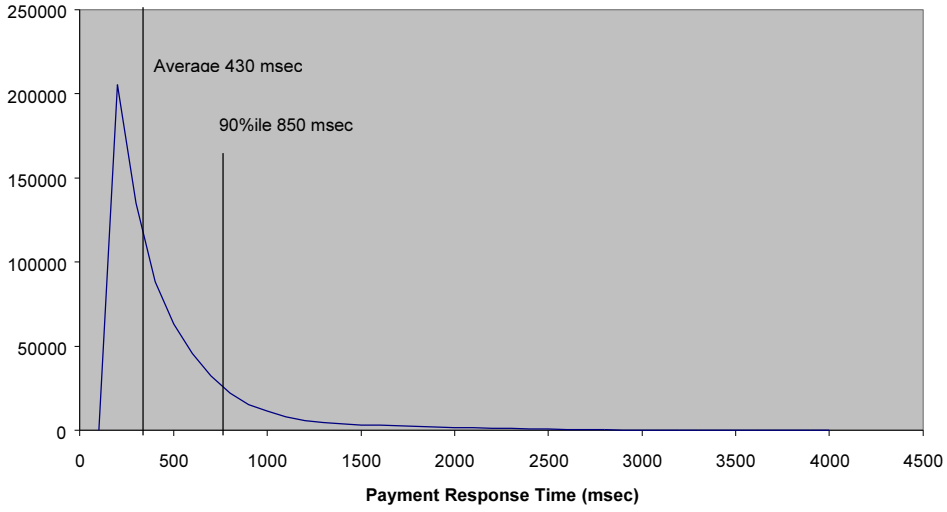


Figure 5: Order Status Response Time Distribution

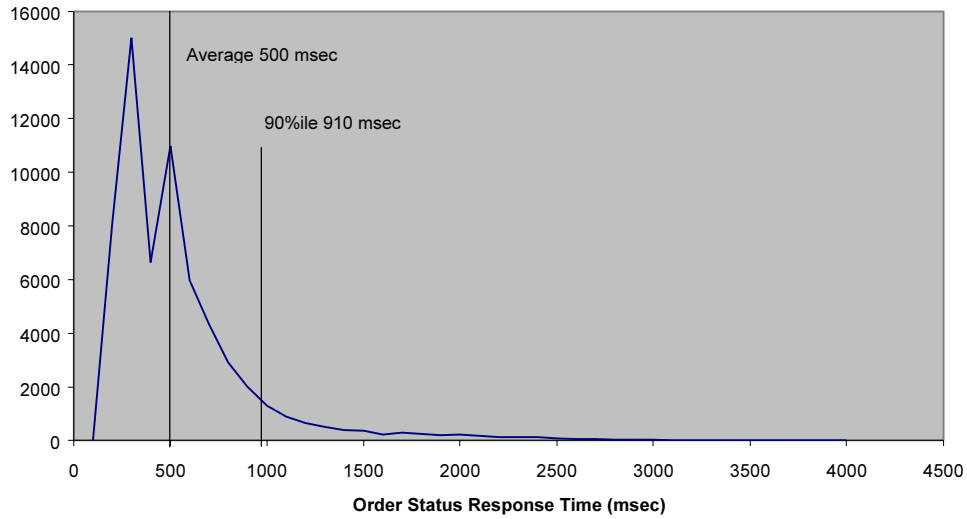


Figure 6: Delivery Response Time Distribution

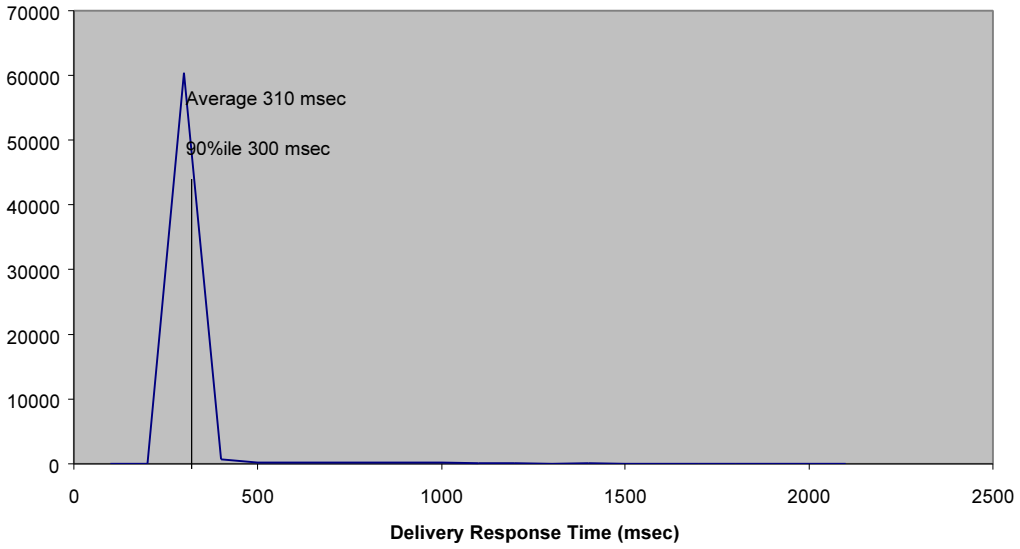
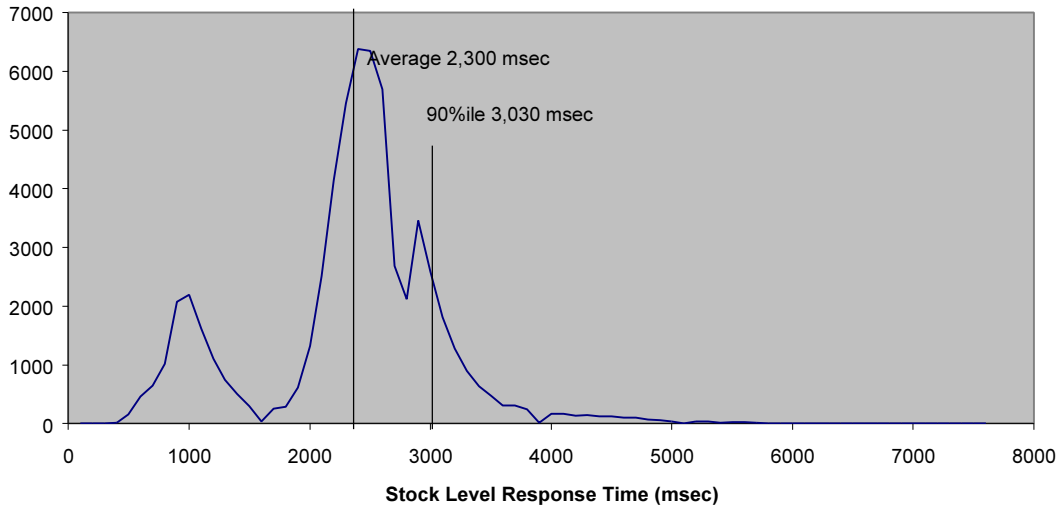


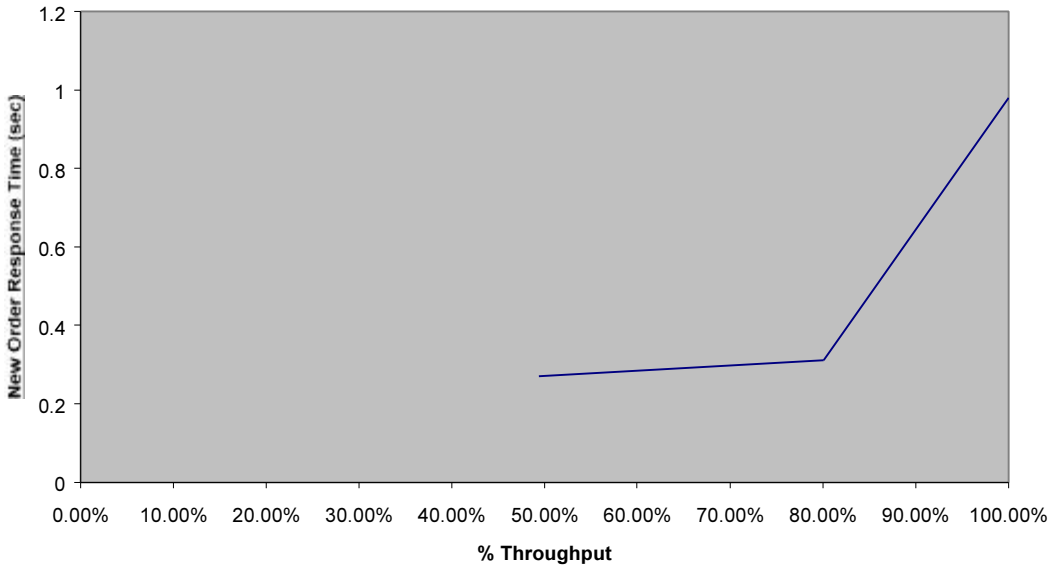
Figure 7: Stock Level Response Time Distribution



New-Order Response Time vs. Throughput Graph

The performance curve for response times versus throughput (see Clause 5.6.2) must be reported for the New-Order transaction. (8.1.6.5)

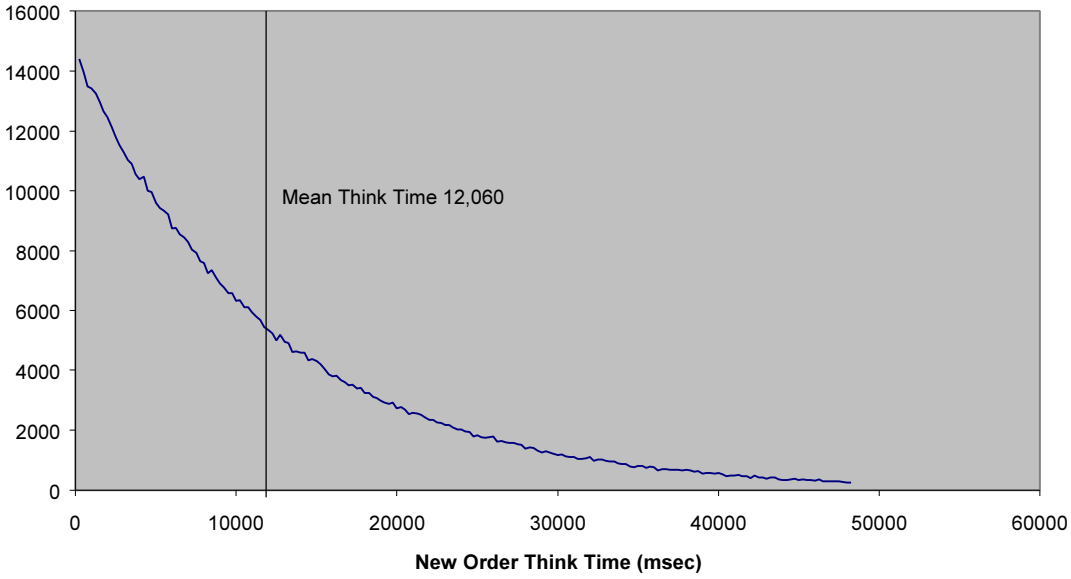
Figure 8: New Order Response Time vs. Throughput



New-Order Think Time Distribution Graph

Think Time frequency distribution curves (see Clause 5.6.3) must be reported for the New-Order transaction. (8.1.6.6)

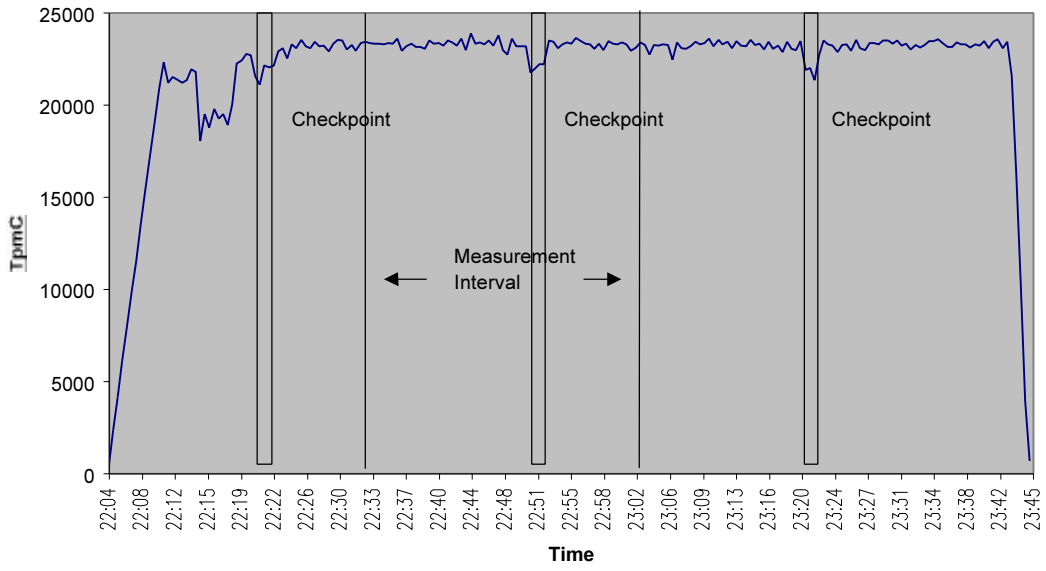
Figure 9: New Order Think Time Distribution



Steady-State Graph

A graph of throughput versus elapsed time (see Clause 5.6.5) must be reported for the New-Order transaction. (8.1.6.8)

Figure 10: New Order Throughput vs. Time



Steady-State Methodology

The method used to determine that the SUT had reached a steady state prior to commencing the measurement interval (see Clause 5.5) must be described. (8.1.6.9)

Steady state was determined using the RTE. Steady state was further confirmed by the throughput data collected during the run and graphed in Figure 10.

Work Performed During Steady State

A description of how the work normally performed during a sustained test (for example checkpointing, writing redo/undo log records, etc.), actually occurred during the measurement interval must be reported. (8.1.6.10)

The RTE generated the required input data to choose a transaction from the menu. This data was timestamped. The response for the requested transaction was verified and timestamped in the RTE log files.

The RTE generated the required input data for the chosen transaction. It waited to complete the minimum required key time before transmitting the input screen. The transmission was timestamped. The return of the screen with the required response data was timestamped. The difference between these two timestamps was the response time for that transaction and was logged in the RTE log.

The RTE then waited the required think time interval before repeating the process starting at selecting another transaction from the menu.

The RTE transmissions were sent to application processes running on the client machines through Ethernet LANs. These client application processes handled all screen I/O as well as all requests to the database on the server. The applications communicated with the database server over another Ethernet LAN using Microsoft SQL Server DBLIB library and RPC calls.

To perform checkpoints at specific intervals, we set SQL Server *recovery interval* to the maximum allowable value and wrote a script to schedule multiple checkpoints at specific intervals. By setting the TRACE FLAG #3502, SQL Server logged the checkpoint beginning and ending time in the ERRORLOG file. The script included a wait time between each checkpoint equal to the measurement interval, which was 30 minutes. The checkpoint script was started manually after the RTE had all users logged in and sending transactions.

At each checkpoint, Microsoft SQL Server wrote to disk all memory pages that had been updated but not yet physically written to disk. Upon completion of the checkpoint, Microsoft SQL Server wrote a special record to the recovery log to indicate that all disk operations had been satisfied to this point. The positioning of the checkpoint was verified to be clear of the guard zones and is depicted on the graph in Figure 10.

Reproducibility Methodology

A description of the method used to determine the reproducibility of the measurement results must be reported. (8.1.6.11)

Multiple performance runs were tested and verified. All compliant tests were within 1% of the reported measurement result.

Measurement Interval

A statement of the duration of the measurement interval for the reported Maximum Qualified Throughput (tpmC) must be included. (8.1.6.12)

The measurement interval was 30 minutes.

Transaction Mix

8.1.6.13 The method of regulation of the transaction mix (e.g., card decks or weighted random distribution) must be described. If weighted distribution is used and the RTE adjusts the weights associated with each transaction type, the maximum adjustments to the weight from the initial value must be disclosed. (8.1.6.13)

The RTE was given a weighted random distribution, which was not adjusted during the run.

The percentage of the total mix for each transaction type must be disclosed. (8.1.6.14)

Table 8: Transaction Mix

Transaction	Percentage
New Order	44.94%
Payment	43.00%
Order Status	4.01%
Delivery	4.03%
Stock Level	4.01%

Other Metrics

The percentage of New-Order transactions rolled back as a result of invalid item number must be disclosed. (8.1.6.15)
The average number of order-lines entered per New-Order transaction must be disclosed. (8.1.6.16)
The percentage of remote order-lines entered per New-Order transaction must be disclosed. (8.1.6.17)
The percentage of remote Payment transactions must be disclosed. (8.1.6.18)
The percentage of customer selections by customer last name in the Payment and Order-Status transactions must be disclosed. (8.1.6.19)
The percentage of Delivery transactions skipped due to there being fewer than necessary orders in the New-Order table must be disclosed. (8.1.6.20)

Table 9: Transaction Statistics

Transaction	Function	Value
New Order	Home Warehouse Items	99.00%
	Remote Warehouse Items	1.00.%
	Rolled Back Transactions	1.02%
	Average Lines Per Order	10.00%
Payment	Home Warehouse	85.05%
	Remote Warehouse	14.95%
	Non-Primary Key Access	60.04%
Order Status	Non-Primary Key Access	60.17%
Delivery	Skipped Transactions	0

Checkpoints

The number of checkpoints in the Measurement Interval, the time in seconds from the start of the Measurement Interval to the first checkpoint and the Checkpoint Interval must be disclosed. (8.1.6.21)

The first checkpoint was started 12 minutes into the rampup. The second checkpoint was started 17 minutes into the measurement interval. The checkpoint in the measurement interval lasted 1 minutes 48 seconds. The measurement interval contains the second checkpoint, and is clear of the guard zones.

Clause 6 -- SUT, Driver, and Communication Definition Related Items

RTE Parameters

The RTE input parameters, code fragments, functions, etc. used to generate each transaction input field must be disclosed. (8.1.7.1)
***Comment**: The intent is to demonstrate the RTE was configured to generate transaction input data as specified in Clause 2.*

The RTE used was the Microsoft BenchCraft RTE System. The RTE parameters are in Appendix C.

Emulated Components

It must be demonstrated that the functionality and performance of the components being emulated in the Driver System are equivalent to that of the priced system. The results of the test described in Clause 6.6.3.4 must be disclosed. (8.1.7.2)

No components were emulated.

Benchmarked and Targeted System Configuration Diagrams

A complete functional diagram of both the benchmark configuration and the configuration of the proposed (target) system must be disclosed. A detailed list of all software and hardware functionality being performed on the Driver System, and its interface to the SUT must be disclosed (see Clause 6.6.3.6). (8.1.7.3)

The driver system performed the data generation and communication to the client through the standard web browser (HTTP) protocol. It also captured and timestamped the SUT output data for post-processing of the reported metrics. No other functionality was included on the driver system.

Figures 1 & 2 of this report contain detailed diagrams of both the benchmark configuration and the priced configuration.

Network Configuration

The network configurations of both the tested services and the proposed (target) services which are being represented and a thorough explanation of exactly which parts of the proposed configuration are being replaced with the Driver System must be disclosed (see Clause 6.6.4). (8.1.7.4)

The network configuration of the benchmarked and priced configurations were the same.

Network Bandwidth

The bandwidth of the network(s) used in the tested/priced configuration must be disclosed. (8.1.7.5)

The bandwidth of the tested and priced networks were as follows:
- 10 BaseT (10 Mbit/sec) network segments between the RTE/Emulated Users and the Clients.
- 100 BaseT(100 Mbit/sec) between the Clients and Server.

Operator Intervention

If the configuration requires operator intervention (see Clause 6.6.6), the mechanism and the frequency of this intervention must be disclosed. (8.1.7.6)

This configuration does not require any operator intervention to sustain eight hours of the reported throughput.

Clause 7 -- Pricing Related Items

Hardware and Software List

A detailed list of hardware and software used in the priced system must be reported. Each separately orderable item must have vendor part number, description, and release/revision level, and either general availability status or committed delivery date. If package-pricing is used, vendor part number of the package and a description uniquely identifying each of the components of the package must be disclosed.

Pricing source(s) and effective date(s) of price(s) must also be reported. (8.1.8.1)

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 used must be disclosed. (8.1.8.2)

The details of the hardware and software are reported in the front of this report as part of the executive summary. All third party quotations are included at the end of this report as Appendix E.

Included in the price is a 6% discount for large purchases.

Availability Date

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, the reported availability date for the priced system must be the date at which all components are committed to be available. (8.1.8.3)

Hardware Availability Date: May 17, 1999
Software Availability Date: May 17, 1999

Measured TpmC

A statement of the measured tpmC, as well as the respective calculations for 5-year pricing, price/performance (price/tpmC), and the availability date must be included. (8.1.8.4)

Maximum Qualified Throughput: **23,235.57 tpmC**
Price Performance Metric: **$16.66 per tpmC**

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. (8.1.8.5)

This system is priced for the United States of America.

Usage Pricing

For any usage pricing, the sponsor must disclose (8.1.8.6):

- *Usage level at which the component was priced.*
- *A statement of the company policy allowing such pricing.*

***Comment**: Usage pricing may include, but is not limited to, the operating system and database management software.*

The component pricing based on usage is shown below:

- 3 Microsoft Windows NT 4.0 Licenses
- 1 Microsoft Windows NT 4.0 Enterprise Edition license
- 1 Microsoft SQL Server, Enterprise Edition, v.7.00 License
- 1 Microsoft Visual C++
- 3 BEA Tuxedo Core Functional Services 6.4 for NT
- 5 years support for all components

System Pricing

System pricing should include subtotals for the following components: Server Hardware, Server Software, Client Hardware, Client Software, and Network Components used for terminal connection (see Clause 7.2.2.3). Clause 6.1 describes the Server and Client components. An example of the standard pricing sheet is shown in Appendix B. (8.1.8.7)
System pricing must include line item indication where non-sponsoring companies' brands are used. System pricing must also include line item indication of third party pricing. See example in Appendix B. (8.1.8.8)
***Comment:** By standardizing the pricing spreadsheet and adding subtotals the value of the FDR and executive summary will be enhanced. This will allow the reader to more easily compare results and determine pricing.*

The details of the hardware and software are reported in the front of this report as part of the executive summary. All third party quotations are included at the end of this report as Appendix E.

Clause 9 -- Audit Related Items

Auditor

The auditor's name, address, phone number, and a copy of the auditor's attestation letter indicating compliance must be included in the Full Disclosure Report. (8.1.9.1)
A review of the pricing model is required to ensure that all components required are priced (see Clause 9.2.8). The auditor is not required to review the final Full Disclosure Report or the final pricing prior to issuing the attestation letter. (8.1.9.2)

This TPC-C benchmark was audited by Tom Sawyer of Performance Metrics, Inc.

Availability of the Full Disclosure Report

The Full Disclosure Report must be readily available to the public at a reasonable charge, similar to the charges for similar documents by the test sponsor. The report must be made available when results are made public. In order to use the phrase "TPC Benchmark™ C", the Full Disclosure Report must have been submitted to the TPC Administrator as well as written permission obtained to distribute same.

Requests for this TPC Benchmark C Full Disclosure Report should be sent to:

Transaction Processing Performance Council
c/o Shanley Public Relations
777 North First Street, Suite 6000
San Jose, CA 95112-6311

or:

Acer Inc.
7F, 88, Sec. 1, Hsin Tai Wu Rd.,
Hsichih, Taipei Hsien 221,
Taiwan, R.O.C.

May 15, 1999

Mr. Duncan Liu
Director
Acer Inc.
21F, 88, Sec. 1, Hsin Tai Wu Rd.,
Hsichih, Taipei Hsien 221,
Taiwan, ROC

I have verified the TPC Benchmark™ C client/server for the following configuration:

Platform:	AcerAltos 21000
Database Manager:	Microsoft SQL Server 7.0 Enterprise Edition
Operating System:	Miscrosoft NT Server 4.0 Enterprise Edition
Transaction Monitor:	Tuxedo 6.4

Server: AcerAltos 21000				
CPU's	Memory	Disks	90% Response	tpmC
4 Pentium III Xeon @ 500 MHz	Main: 4 GB Cache: 2MB each	8 @ 18 GB 193 @ 9.1GB	**0.98 sec**	**23,235.57**
3 Clients: AcerAltos9100B				
2 Pentium II @ 450 MHz	Main: 512 MB Cache: 512K	1 @ 4.5 GB	na	na

In my opinion, these performance results were produced in compliance with the TPC requirements for the benchmark. The following attributes of the benchmark were given special attention:

- The transactions were correctly implemented.
- The database files were properly sized and populated.
- The database was properly scaled with 1,900 warehouses. The measurement used 1,880 warehouses. I verified that d_next_o_id and w_ytd did not change for the unused warehouses.
- The ACID properties were met.
- The loss-of-log and loss-of-data tests were performed on a database scaled to 10 warehouses.
- Input data was generated according to the specified percentages.
- Eight hours of mirrored log space was present on the tested system.
- Eight hours of growth space for the dynamic tables was present on the tested system.
- The data for the 180 day space calculation was verified.
- The steady state portion of the test was 30 minutes.
- One checkpoint was taken before the measured interval.
- One checkpoint was taken during the measured interval.

- The checkpoints were verified to be clear of the guard zone.
- The system pricing was checked for major components and maintenance.
- Third party quotes were verified for compliance.

Auditor Notes:

Sincerely,

Tom Sawyer
Auditor

Appendix A – Source Code

WEB Client

Makefile

```
!IF "$(CFG)" == ""
CFG=Release
!MESSAGE No configuration specified.  Defaulting to
Debug
!ENDIF

!IF "$(SQL_LOC)" == ""
SQL_LOC=D:\mssql7\DevTools
!MESSAGE No SQL_LOC specified.  Defaulting to
D:\MSSQL7\DevTools
!ENDIF

!IF "$(TUXDIR)" == ""
TUXDIR    = F:\TUXEDO
!MESSAGE No TUXDIR specified.  Defaulting to F:\TUXEDO
!ENDIF

!IF "$(CFG)" != "Release" && "$(CFG)" != "Debug"
!MESSAGE Invalid configuration "$(CFG)" specified.
!MESSAGE You can specify a configuration when running
NMAKE on this makefile
!MESSAGE by defining the macro CFG on the command line.
For example:
!MESSAGE
!MESSAGE NMAKE CFG="Debug"
!MESSAGE
!MESSAGE Possible choices for configuration are:
!MESSAGE
!MESSAGE "Release"
!MESSAGE "Debug"
!MESSAGE
!ERROR An invalid configuration is specified.
!ENDIF

OUTDIR          = .
SRCDIR          = .\Src
OBJDIR          = .\Objs
DBLIB           = $(SQL_LOC)
DBLIBINC        = $(DBLIB)\include
DBLIBDIR        = $(DBLIB)\lib

!IF "$(CFG)" != "Debug"
LDEBUG       =
CDEBUG       =
LDEBUG_RG    =
CDEBUG_RG    =
DEBUG        =
FLAGS        = /D "WIN32" /D "_WINDOWS" /D
"_TMSTHREADS"
#FLAGS        = /D "WIN32" /D "_WINDOWS" /D
"_TMSTHREADS" /D "LOCAL_ALLOC"
CFLAGS       = /D "WIN32" /D "_WINDOWS" /D
"_TMSTHREADS"
OPT          = /Ot
!ELSE
LDEBUG       = /debug /pdb:$(OBJDIR)\tpcc.pdb
CDEBUG       = /Zi /Yd /Fd$(OBJDIR)\tpcc.pdb
LDEBUG_RG    = /debug /pdb:$(OBJDIR)\install.pdb
CDEBUG_RG    = /Zi /Yd /Fd$(OBJDIR)\install.pdb
FLAGS        = /D "WIN32" /D "_WINDOWS" /D
"_TMSTHREADS"
#FLAGS        = /D "WIN32" /D "_WINDOWS" /D
"_TMSTHREADS" /D "LOCAL_ALLOC"
CFLAGS       = /D "WIN32" /D "_WINDOWS" /D
"_TMSTHREADS"
OPT          = /Od
!ENDIF


LINK32_LIBS     = user32.lib msacm32.lib advapi32.lib
$(DBLIBDIR)\ntwdblib.lib
TUX_LIBS  = $(TUXDIR)\lib\libtux.lib
$(TUXDIR)\lib\libbuft.lib $(TUXDIR)\lib\libtux2.lib  \
$(TUXDIR)\lib\libfml.lib $(TUXDIR)\lib\libfml32.lib
$(TUXDIR)\lib\libgp.lib
OTHER_LIBS          = wsock32.lib kernel32.lib
gdi32.lib comdlg32.lib winspool.lib
LINK32_OBJS     = "$(OBJDIR)\tpcc.obj"
LINK32_DEF      = "$(SRCDIR)\tpcc.def"

LINK32_FLAGS    = /nologo /subsystem:windows /dll
/incremental:no $(LDEBUG) /def:"$(LINK32_DEF)"
/out:"$(OUTDIR)\tpcc.dll"

LINK32_LIBS_RG  = user32.lib gdi32.lib advapi32.lib
version.lib comctl32.lib
LINK32_OBJS_RG  = "$(OBJDIR)\install.obj"
"$(OBJDIR)\install.res"
LINK32_FLAGS_RG = /nologo /subsystem:windows
/incremental:no $(LDEBUG_RG) /out:$(OUTDIR)\install.exe

ALL:    $(OBJDIR)\. $(OUTDIR)\. $(OUTDIR)\tpcc.dll
$(OUTDIR)\multi_trans.exe \
 $(OUTDIR)\Delivery.exe $(OUTDIR)\Delirpt.exe

$(OBJDIR)\.:
          if not exist $(OBJDIR) md $(OBJDIR)

$(OUTDIR)\.:
          if not exist $(OUTDIR) md $(OUTDIR)

"$(OBJDIR)\tpcc.obj":   "$(SRCDIR)\tpcc.c"
"$(SRCDIR)\tpcc.h"
          cl.exe /nologo /MT /W3 $(CDEBUG) $(OPT) /I
$(DBLIBINC) /I $(TUXDIR)\include $(FLAGS)
/Fo$(OBJDIR)\tpcc.obj /c "$(SRCDIR)\tpcc.c"

$(OBJDIR)\tpcc.res: $(SRCDIR)\tpcc.rc
          rc.exe /l 0x409 /fo $(OBJDIR)\tpcc.res
$(FLAGS) $(SRCDIR)\tpcc.rc

$(OUTDIR)\tpcc.dll:     $(LINK32_OBJS) $(LINK32_DEF)
          link.exe   $(LINK32_FLAGS) $(LINK32_OBJS)
$(LINK32_LIBS) $(TUX_LIBS) $(OTHER_LIBS)

$(OUTDIR)\multi_trans.exe: $(SRCDIR)\multi_trans.c
          $(TUXDIR)\bin\buildserver /f
$(SRCDIR)\multi_trans.c /o $(OUTDIR)\multi_trans.exe \
/s T1,T2,T3,T4,T5,T6,T7,T8,T9,T10 /l "$(LINK32_LIBS) /I
$(DBLIBINC)"
          del multi_trans.obj


$(OUTDIR)\Delivery.exe: $(SRCDIR)\delivery.c
          $(TUXDIR)\bin\buildserver /f
$(SRCDIR)\delivery.c /o $(OUTDIR)\Delivery.exe \
/s DELIVERY /l "$(LINK32_LIBS) /I  $(DBLIBINC)"
          del delivery.obj

$(OUTDIR)\Delirpt.exe: $(SRCDIR)\delirpt.c
          cl.exe $(SRCDIR)\delirpt.c /o
"$(OUTDIR)\Delirpt.exe"
          del delirpt.obj
```

Delivery.c

```
/*         FILE:               DELIVERY.C
 *
 *         Based on: Microsoft TPC-C Kit Ver. 3.00.000
 *
 *                                        Copyright
Microsoft, 1996
 *                                        Copyright
Performance Tuning Corporation, 1997
 *
 *         PURPOSE:  New Order Tuxedo Server.
 *         Author:             Philip Durr
 *
          philipdu@Microsoft.com
 *
 *  MODIFIED        Changed for modularity and to allow
for the Tuxedo TM
 *
 *  Author:                    Edward Whalen
 *                                        Performance
Tuning Corporation
 *
          ewhalen@perftuning.com
 */

#include <windows.h>
#include <process.h>
#include <stdio.h>
#include <stdarg.h>
#include <malloc.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <sys\timeb.h>
#include <io.h>
```

```
#define DBNTWIN32
#include <sqlfront.h>
#include <sqldb.h>

#include "trans.h"
                    //tpckit transaction header
contains defination of structures specific to TPC-C
#include "httpext.h"
                    //ISAPI DLL information header

#include "tpcc.h"
                    //this dlls specific structure,
value e.t. header.

#include <tmenv.h>
#include <xa.h>
#include <atmi.h>

extern BOOL         bLog;                          //=
FALSE;
extern BOOL         bFlush;
                                        //Flush
delivery log info when written.
BOOL      verbose                       = FALSE;
BOOL      bError              = FALSE;

extern int                    iThreads; //        = 5;
extern int                    iMaxWareHouses;//   =
500;
extern int                    iDelayMs;           //=
100;
short     iMaxConnections     = (short)1;
short     iDeadlockRetry      = (short)3;

DBPROCESS *pdbproc;

char
          szServer[32];
          //SQL server name
char
          szDatabase[32];
          //tpcc database name
char
          szUser[32];
                    //user name
char
          szPassword[32];
          //user password
int spId;

TERM Term;

static char         szTpccLogPath[256]; //path to html
log file if logging turned on in registry.
static char szErrorLogPath[256];        //path to error
log file.

static    CRITICAL_SECTION
          CriticalSection;
static    CRITICAL_SECTION
          ErrorLogCriticalSection;
static    EXTENSION_CONTROL_BLOCK       *gpECB = NULL;
static    int
                    bTpccExit;          //exit delivery
disconnect loop as dll exiting.

void TMLog();
BOOL SQLInit();
extern void UtilStrCpy();
extern void UtilStrCpy();
extern BOOL SQLOpenConnection();
extern BOOL SQLCloseConnection();



//BOOL
          bDone;
                              //delivery executable
termination request flag
BOOL
          bFlush;
                              //Flush delivery log info
when written.

#define ERR_CANNOT_CREATE_THREAD                  1000
          //Cannot create thread.
#define ERR_DBGETDATA_FAILED                      1001
          //Get data failed.
#define   ERR_REGISTRY_NOT_SETUP
          1002      //Registry not setup for tpcc.
#define   ERR_CANNOT_ACCESS_DELIVERY_FN 1003
          //Cannot access ReadDelivery cache.
#define   ERR_CANNOT_ACCESS_REGISTRY              1004
          //Cannot access registry key TPCC.
#define   ERR_CANNOT_CREATE_RESULTS_FILE          1005
          //Cannot create results file.

FILE      *fpLog;

/* FUNCTION: tpsvrinit ( int argc, char *argv[])
 *
 * PURPOSE:          Initialize the Server to Database
connection.
 *
 * RETURNS:                    int       0
          Success
 *                                                   -1
          Failure
 *
 *
 * COMMENTS:         None
 *
 */

int tpsvrinit ( int argc, char *argv[] )
{

          if ( GetParameters(argc, argv) )
          {
                    PrintParameters();
                    return -1;
          }

          if ( verbose )
                    TMLog("TPSVRINIT: Delivery: Server
%s, Database %s, User %s, Password %s, Flush %d.",
                              szServer, szDatabase,
szUser, szPassword, bFlush);

          if ( ! SQLInit() )
          {
                    TMLog( "DELIVERY: SQLInit Failed"
);
                    return -1;
          }

          if ( SQLOpenConnection ( NULL, 0, 0,
&pdbproc, szServer, szDatabase, szUser, szPassword,
szDatabase, &spId))
          {
                    TMLog ( "DELIVERY:
SQLOpenConnection Failed" );
                    dbexit();
                    return -1;
          }

          OpenLogFile();

          return 0;
}

/* FUNCTION: tpsvrdone ( void )
 *
 * PURPOSE:          Close the Server to Database
connection.
 *
 * RETURNS:                    int       0
          Success
 *                                                   -1
          Failure
 *
 *
 * COMMENTS:         None
 *
 */

void tpsvrdone ( void )
{
          SQLCloseConnection( NULL, pdbproc);
          dbexit();
}

/* FUNCTION: DELIVERY ( TPSVCINFO *rqst )
 *
 * PURPOSE:          Process a Delivery request.
 *
 * RETURNS:                    int       0
          Success
 *                                                   -1
          Failure
 *
 *
 * COMMENTS:         None
 *
 */
```

```
void DELIVERY ( TPSVCINFO *rqst )
{
          PECBINFO pECBInfo = dbgetuserdata(pdbproc);
          DELIVERY_DATA *dp;
          int size = rqst->len;

          if (verbose)
                    TMLog(" DELIVERY: Begin
transaction");


          dp = (DELIVERY_DATA *) rqst->data;

          if (verbose )
          {
                    TMLog(" DELIVERY: w_id %d ", dp-
>w_id);
                    TMLog(" DELIVERY: d_id %d ", dp-
>o_carrier_id);
          }

          bError = FALSE;

          dp->retval = SQLDelivery( pdbproc, dp,
iDeadlockRetry);

          if (bError == TRUE)
                    dp->retval = -1;


          tpreturn( TPSUCCESS, 0, rqst->data, size, 0);
}


/* FUNCTION: void CalculateElapsedTime(int *pElapsed,
LPSYSTEMTIME lpBegin, LPSYSTEMTIME lpEnd)
 *
 * PURPOSE:         This function calculates the
elapsed time a delivery transaction took.
 *
 * ARGUMENTS:       int
          *pElapsed pointer to int variable to receive
calculated elapsed
 *

          time in milliseconds.
 *                                        LPSYSTEMTIME
          lpBegin             Pointer to system time
structure containing
 *

          transaction beginning time.
 *                                        LPSYSTEMTIME
          lpEnd               Pointer to system time
structure containing
 *

          transaction ending time.
 * RETURNS:                   None
 *
 * COMMENTS:        None
 *
 */

static void CalculateElapsedTime(int *pElapsed,
LPSYSTEMTIME lpBegin, LPSYSTEMTIME lpEnd)
{
          int                         beginSeconds;
          int                         endSeconds;


          beginSeconds = (lpBegin->wHour * 3600000) +
(lpBegin->wMinute * 60000) + (lpBegin->wSecond * 1000)
+ lpBegin->wMilliseconds;
          endSeconds = (lpEnd->wHour * 3600000) +
(lpEnd->wMinute * 60000) + (lpEnd->wSecond * 1000) +
lpEnd->wMilliseconds;
          *pElapsed = endSeconds - beginSeconds;

          //check for day boundry, this will function
for 24 hour period however it will not work over 48
hours.
          if ( *pElapsed < 0 )
                    *pElapsed = *pElapsed + (24 * 60 *
60 * 1000);

          return;
}

/* FUNCTION: int SQLDelivery(DBPROCESS *dbproc,
DELIVERY *pDelivery, short deadlock_retry )
 *
 * PURPOSE:         This function processes the
delivery transaction.
 *
 * ARGUMENTS:       DELIVERY            *pDelivery
                    Pointer to delivery transaction
structure
 *
 * RETURNS:                   int
          ERR_DBGETDATA_FAILED
          Delivery get data operation failed.
 *
          ERR_SUCCESS
                    Delivery successful, no error
 *
 *
 * COMMENTS:        None
 *
 */

static int SQLDelivery(DBPROCESS *dbproc, DELIVERY_DATA
*pDelivery, short deadlock_retry)
{
          RETCODE   rc;
          int                 i;
          int                 deadlock_count;
          BYTE      *pData;
          SYSTEMTIME          trans_end;
          //delivery transaction finished time
          int       elapsed;
          //delivery transaction time

          deadlock_count = 0;


          // Start new delivery
          while ( TRUE )
          {
                    if (dbrpcinit(dbproc,
"tpcc_delivery", 0) == SUCCEED)
                    {
                              dbrpcparam(dbproc, NULL,
0, SQLINT2, -1, -1, (BYTE *)&pDelivery->w_id);
                              dbrpcparam(dbproc, NULL,
0, SQLINT1, -1, -1, (BYTE *)&pDelivery->o_carrier_id);

                              if (dbrpcexec(dbproc) ==
SUCCEED)
                              {
                                        while (((rc =
dbresults(dbproc)) != NO_MORE_RESULTS) && (rc != FAIL))
                                        {
                                                  while
(((rc = dbnextrow(dbproc)) != NO_MORE_ROWS) && (rc !=
FAIL))
                                                  {

          for (i=0;i<10;i++)

          {

                    if(pData=dbdata(dbproc, i+1))

                              pDelivery->o_id[i] =
*((DBINT *)pData);

                    else

                              pDelivery->o_id[i] = 0;

          }
                                                  }
                                        }
                              }
                    }
                    if ( !SQLDetectDeadlock(dbproc) )
                              break;
                    deadlock_count++;
                    Sleep(10 * deadlock_count);
          }
          GetLocalTime(&trans_end);

          CalculateElapsedTime(&elapsed, &pDelivery-
>queue_time, &trans_end);

          fprintf(fpLog,
"%2.2d/%2.2d/%2.2d,%2.2d:%2.2d:%2.2d:%3.3d,%2.2d:%2.2d:
%2.2d:%3.3d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d\r\n"
,
                    trans_end.wYear - 1900,
trans_end.wMonth, trans_end.wDay,
                    pDelivery->queue_time.wHour,
pDelivery->queue_time.wMinute,
```

```
                    pDelivery->queue_time.wSecond,
pDelivery->queue_time.wMilliseconds,
                    trans_end.wHour, trans_end.wMinute,
trans_end.wSecond, trans_end.wMilliseconds,
                    elapsed,
                    pDelivery->w_id, pDelivery-
>o_carrier_id,
                    pDelivery->o_id[0], pDelivery-
>o_id[1], pDelivery->o_id[2], pDelivery->o_id[3],
                    pDelivery->o_id[4], pDelivery-
>o_id[5], pDelivery->o_id[6], pDelivery->o_id[7],
                    pDelivery->o_id[8], pDelivery-
>o_id[9] );

          if ( bFlush )
                    fflush(fpLog);


          return ERR_SUCCESS;
}

/* FUNCTION: BOOL SQLDetectDeadlock(DBPROCESS *dbproc)
 *
 * PURPOSE:         This function is used to check for
deadlock conditions.
 *
 * ARGUMENTS:       DBPROCESS           *dbproc
          DBPROCESS to check
 *
 * RETURNS:                   BOOL      FALSE
                              No lock condition present
 *
          TRUE                                      Lock
condition detected
 *
 * COMMENTS:        None
 *
 */

static BOOL SQLDetectDeadlock(DBPROCESS *dbproc)
{
          if (*((BOOL *) dbgetuserdata(dbproc)) ==
TRUE)
    {
                    *((BOOL *) dbgetuserdata(dbproc)) =
FALSE;
                    return TRUE;
          }
          return FALSE;
}

/* FUNCTION: int OpenLogFile(void)
 *
 * PURPOSE:         This function opens the delivery
log file for use.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                   int
          ERR_REGISTRY_NOT_SETUP
          Registry not setup.
 *
          ERR_CANNOT_CREATE_RESULTS_FILE
          Cannot create results log file.
 *
          ERR_SUCCESS
                              Log file successfully
opened
 *
 *
 * COMMENTS:        None
 *
 */

static int OpenLogFile(void)
{
          HKEY      hKey;
          BOOL      bRc;
          BYTE      szTmp[256];
          char      szKey[256];
          char      szLogPath[256];
          DWORD     size;
          DWORD     sv;
          int                 len;
          char      *ptr;

          szLogPath[0] =  0;
          bRc = TRUE;
          if ( RegOpenKeyEx(HKEY_LOCAL_MACHINE,
"SYSTEM\\CurrentControlSet\\Services\\W3SVC\\Parameters
\\Virtual Roots", 0, KEY_ALL_ACCESS, &hKey) ==
ERROR_SUCCESS )
          {
                    sv = sizeof(szKey);
                    size = sizeof(szTmp);

                    if ( RegEnumValue(hKey, 0, szKey,
&sv, NULL, NULL, szTmp, &size) == ERROR_SUCCESS )
                    {
                              strcpy(szLogPath, szTmp);
                              bRc = FALSE;
                    }
                    RegCloseKey(hKey);
          }

          if ( bRc )
                    return ERR_REGISTRY_NOT_SETUP;

          if ( (ptr = strchr(szLogPath, ',')) )
                    *ptr = 0;

          len = strlen(szLogPath);
          if ( szLogPath[len-1] != '\\' )
          {
                    szLogPath[len] = '\\';
                    szLogPath[len+1] = 0;
          }
          strcat(szLogPath, "delilog.");

          fpLog = fopen(szLogPath, "ab");

          if ( !fpLog )
                    return
ERR_CANNOT_CREATE_RESULTS_FILE;

          return ERR_SUCCESS;
}

/*
 *        Common Code for all Servers
 */

 /* FUNCTION: BOOL SQLInit()
 *
 * PURPOSE:         This function initializes SQL
Server for later use.
 *
 *
 * RETURNS:                   BOOL      FALSE     if
successfull
 *
          TRUE      if an error occurs and connection
cannot be established.
 *
 * COMMENTS:        None
 *
 */
BOOL SQLInit ()
{
          dbinit();

          if ( dbgetmaxprocs() < iMaxConnections )
          {
                    if ( dbsetmaxprocs(iMaxConnections)
== FAIL )
                    {
                              //set for fail error
message when HttpExtensionProc() is called because
                              //at this point we don't
have a pECB so no way to show error message.
                              iMaxConnections = -1;
                    }
          }

          // install error and message handlers
          dbmsghandle((DBMSGHANDLE_PROC)msg_handler);
          dberrhandle((DBERRHANDLE_PROC)err_handler);

          return TRUE;
}


/* FUNCTION: BOOL
SQLOpenConnection(EXTENSION_CONTROL_BLOCK
          *pECB, int iTermId, int iSyncId, DBPROCESS
**dbproc, char *server, char *database, char *user,
char *password, char *app, int *spid, long *pack_size)
 *
 * PURPOSE:         This function opens the sql
connection for use.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                 int
                              iTermId
          terminal id of browser
```

```
 *                                      int
                              iSyncId             sync
id of browser
 *                                      DBPROCESS
         **dbproc  pointer to returned DBPROCESS
 *                                      char
                   *server             SQL server name
 *                                      char
                   *database SQL server database
 *                                      char
                   *user               user name
 *                                      char
                   *password user password
 *                                      char
                   *app                pointer to
returned application array
 *                                      int
                              *spid
         pointer to returned spid
 *                                      long
                   *pack_size          pointer to
returned default pack size
 *
 * RETURNS:                   BOOL      FALSE     if
successfull
 *
         TRUE      if an error occurs
 *
 * COMMENTS:        None
 *
 */


#ifdef USE_ODBC
         static BOOL
SQLOpenConnection(EXTENSION_CONTROL_BLOCK *pECB, int
iTermId, int iSyncId, DBPROCESS **dbproc, char *server,
char *database, char *user, char *password, char *app,
int *spid, long *pack_size)
         {

                   RETCODE   rc;
                   char      buffer[30];

                   *dbproc = (DBPROCESS
*)malloc(sizeof(DBPROCESS));
                   if ( !*dbproc )
                             return TRUE;

                   //set pECB data into dbproc
                   (*dbproc)->bDeadlock = FALSE;
                   (*dbproc)->bFailed = FALSE;
                   (*dbproc)->pECB = pECB;
                   (*dbproc)->iTermId = iTermId;
                   (*dbproc)->iSyncId = iSyncId;


                   if ( SQLAllocConnect(henv,
&(*dbproc)->hdbc) == SQL_ERROR )
                             return TRUE;

                   if ( SQLSetConnectOption((*dbproc)-
>hdbc, SQL_PACKET_SIZE, pack_size) == SQL_ERROR )
                             return TRUE;

                   rc = SQLConnect((*dbproc)->hdbc,
server, SQL_NTS, user, SQL_NTS, password, SQL_NTS);
                   if (rc != SQL_SUCCESS && rc !=
SQL_SUCCESS_WITH_INFO)
                             return TRUE;
                   rc = SQLAllocStmt((*dbproc)->hdbc,
&(*dbproc)->hstmt);
                   if (rc == SQL_ERROR)
                             return TRUE;

                   sprintf(buffer,"use %s", Client-
>database);

                   rc = SQLExecDirect((*dbproc)-
>hstmt, buffer, SQL_NTS);
                   if (rc != SQL_SUCCESS && rc !=
SQL_SUCCESS_WITH_INFO)
                             return TRUE;

                   SQLFreeStmt((*dbproc)->hstmt,
SQL_CLOSE);
                   sprintf(buffer,"set nocount on");
                   rc = SQLExecDirect((*dbproc)-
>hstmt, buffer, SQL_NTS);
                   if (rc != SQL_SUCCESS && rc !=
SQL_SUCCESS_WITH_INFO)
                             return TRUE;

                   SQLFreeStmt((*dbproc)->hstmt,
SQL_CLOSE);

                   sprintf(buffer,"select @@spid");

                   rc = SQLExecDirect((*dbproc)-
>hstmt, buffer, SQL_NTS);
                   if (rc != SQL_SUCCESS && rc !=
SQL_SUCCESS_WITH_INFO)
                             return TRUE;

                   if ( SQLBindCol((*dbproc)->hstmt,
1, SQL_C_SSHORT, &(*dbproc)->spid, 0, NULL) ==
SQL_ERROR )
                             return TRUE;


                   if ( SQLFetch((*dbproc)->hstmt) ==
SQL_ERROR )
                             return TRUE;

                   SQLFreeStmt((*dbproc)->hstmt,
SQL_CLOSE);

                   return FALSE;
         }

#else

         static BOOL
SQLOpenConnection(EXTENSION_CONTROL_BLOCK *pECB, int
iTermId, int iSyncId, DBPROCESS **dbproc, char *server,
char *database, char *user, char *password, char *app,
int *spid)
         {
                   LOGINREC  *login;
                   PECBINFO  pEcbInfo;

                   //set local msg proc for login
record
                   //attach pECB record

                   //this is necessary as dblib
provides no way to pass user data in a login structure.
So until
                   //there is an allocated dbproc we
need to use a static which means that the login attempt
must
                   //be serialized.

                   gpECB = pECB;

                   login = dblogin();
                   if ( !*user )
                             DBSETLUSER(login, "sa");
                   else
                             DBSETLUSER(login, user);

                   DBSETLPWD(login,  password);
                   DBSETLHOST(login, app);

//                 Do not set the packet size.  Use
the size set up in SQL Server.
                   DBSETLPACKET(login, (unsigned
short)DEFCLPACKSIZE);

//                 This can potentially cut down on
data conversion
                   DBSETLVERSION(login, DBVER60);

                   if ((*dbproc = dbopen(login, server
)) == NULL)
                             return TRUE;

                   //set pECB data into dbproc
                   pEcbInfo =
(PECBINFO)malloc(sizeof(ECBINFO));
                   pEcbInfo->bDeadlock = FALSE;
                   pEcbInfo->pECB = pECB;
                   pEcbInfo->iTermId = iTermId;
                   pEcbInfo->iSyncId = iSyncId;
                   dbsetuserdata(*dbproc, pEcbInfo);

                   // Use the the right database
                   dbuse(*dbproc, database);

                   dbcmd(*dbproc, "select @@spid");

                   dbsqlexec(*dbproc);
                   while (dbresults(*dbproc) !=
NO_MORE_RESULTS)
                   {
```

```
                                dbbind(*dbproc, 1,
SMALLBIND, (DBINT) 0, (BYTE *) spid);
                                while (dbnextrow(*dbproc)
!= NO_MORE_ROWS)
                                ;
                        }
                        dbcmd(*dbproc, "set nocount on");

                        dbsqlexec(*dbproc);
                        while (dbresults(*dbproc) !=
NO_MORE_RESULTS)
                        {
                                while (dbnextrow(*dbproc)
!= NO_MORE_ROWS)
                                ;
                        }

                        //rollback transaction on abort
                        dbcmd(*dbproc, "set XACT_ABORT
ON");

                        dbsqlexec(*dbproc);
                        while (dbresults(*dbproc) !=
NO_MORE_RESULTS)
                        {
                                while (dbnextrow(*dbproc)
!= NO_MORE_ROWS)
                                ;
                        }

                        return FALSE;
                }

#endif

/* FUNCTION: BOOL
SQLCloseConnection(EXTENSION_CONTROL_BLOCK *pECB,
DBPROCESS *dbproc)
 *
 * PURPOSE:             This function closes the sql
connection.
 *
 * ARGUMENTS:           EXTENSION_CONTROL_BLOCK       *pECB
                passed in structure pointer from inetsrv.
 *                                      DBPROCESS
                *dbproc   pointer to DBPROCESS
 *
 * RETURNS:                     BOOL      FALSE     if
successful
 *
                TRUE      if an error occurs
 *
 * COMMENTS:            None
 *
 */

#ifdef USE_ODBC
                static BOOL
SQLCloseConnection(EXTENSION_CONTROL_BLOCK *pECB,
DBPROCESS *dbproc)
                {
                        if ( dbproc )
                        {
                                SQLFreeStmt(dbproc-
>hstmt, SQL_DROP);
                                SQLDisconnect(dbproc-
>hdbc);
                                SQLFreeConnect(dbproc-
>hdbc);
                                free(dbproc);
                                dbproc = NULL;
                        }
                        return FALSE;
                }
#else
                static BOOL
SQLCloseConnection(EXTENSION_CONTROL_BLOCK *pECB,
DBPROCESS *dbproc)
                {
                        if (dbclose(dbproc) == FAIL)
                                return TRUE;
                        return FALSE;
                }
#endif

 //Lifted from HP FDR since they did such a nice job
void TMLog( char *format, ... )
{
                va_list args;
                char buf[4096];
                int len;
                va_start( args, format );
                _strtime( buf );
                strcat( buf," ");
                len = strlen( buf );
                (void)_vsnprintf( buf+ len, sizeof( buf) -
len - 1, format, args);
                buf[sizeof( buf )- 1]= '\0';
                va_end( args );
                userlog( buf );
}

/* FUNCTION: void UtilStrCpy(char *pDest, char *pSrc,
int n)
 *
 * PURPOSE:             This function copies n characters
from string pSrc to pDst and places a
 *                              null character at the end
of the destination string.
 *
 * ARGUMENTS:           char
                *pDest    destination string pointer
 *                                      char
                *pSrc     source string pointer
 *                                      int
                n
                number of characters to copy
 *
 * RETURNS:                     None
 *
 * COMMENTS:            Unlike strncpy this function
ensures that the result string is
 *                                      always null
terminated.
 *
 */

static void UtilStrCpy(char *pDest, char *pSrc, int n)
{
    strncpy(pDest, pSrc, n);
    pDest[n] = '\0';

                return;
}

/* FUNCTION: int err_handler(DBPROCESS *dbproc, int
severity, int dberr, int oserr, char *dberrstr, char
*oserrstr)
 *
 * PURPOSE:             This function handles DB-Library
errors
 *
 * ARGUMENTS:           DBPROCESS             *dbproc
                        DBPROCESS id pointer
 *                                      int
                                severity
                severity of error
 *                                      int
                                dberr
                error id
 *                                      int
                                oserr
                operating system specific error code
 *                                      char
                        *dberrstr             printable error
description of dberr
 *                                      char
                        *oserrstr             printable error
description of oserr
 *
 * RETURNS:                     int
                        INT_CONTINUE          continue if
error is SQLETIME else INT_CANCEL action
 *
 * COMMENTS:            None
 *
 */


int err_handler(DBPROCESS *dbproc, int severity, int
dberr, int oserr, char *dberrstr, char *oserrstr)
{
                PECBINFO
                pEcbInfo;
                EXTENSION_CONTROL_BLOCK       *pECB;
                FILE
                *fp;
                SYSTEMTIME
                systemTime;
                char
                szTmp[256];
                int
                        iTermId;
                int
                        iSyncId;
```

```
        pEcbInfo = NULL;

        if ((dbproc == NULL) || (DBDEAD(dbproc)))
        {
                TMLog("DBPROC is invalid");
                return INT_CANCEL;
        }

        if ( !(pEcbInfo =
(PECBINFO)dbgetuserdata(dbproc)) )
        {
                pECB = gpECB;
                iTermId = 0;
                iSyncId = 0;
        }
        else
        {
                pECB = pEcbInfo->pECB;
                iTermId = pEcbInfo->iTermId;
                iSyncId = pEcbInfo->iSyncId;
        }

        if ( pEcbInfo && pEcbInfo->bFailed )
        {
                bError == FALSE;
                return INT_CANCEL;
        }

        if ( oserr != DBNOERR )
        {
                TMLog("DBLIB Error %s", oserrstr);
                if ( pEcbInfo )
                {
                        pEcbInfo->bFailed = TRUE;
                        bError = TRUE;
                }

                GetLocalTime(&systemTime);
                fp = fopen(szErrorLogPath, "ab");

                sprintf(szTmp, "ErrorHandler:
DBLIB(%d): %s", oserr, oserrstr);

                TMLog("%2.2d/%2.2d/%2.2d
%2.2d:%2.2d:%2.2d\r\n\r\n%s\r\n\r\n",
                        systemTime.wYear,
systemTime.wMonth, systemTime.wDay,
                        systemTime.wHour,
systemTime.wMinute, systemTime.wSecond,
                        szTmp);

                fclose(fp);
        }

        return INT_CANCEL;
}


/* FUNCTION: int msg_handler(DBPROCESS *dbproc, DBINT
msgno, int msgstate, int severity, char *msgtext)
 *
 * PURPOSE:         This function handles DB-Library
SQL Server error messages
 *
 * ARGUMENTS:       DBPROCESS           *dbproc
                    DBPROCESS id pointer
 *                                      DBINT
                    msgno
        message number
 *                                      int
                            msgstate
        message state
 *                                      int
                            severity
        message severity
 *                                      char
                    *msgtext            printable
message description
 *
 * RETURNS:                 int
                    INT_CONTINUE        continue if
error is SQLETIME else INT_CANCEL action
 *
                            INT_CANCEL
        cancel operation
 *
 * COMMENTS:        This function also sets the dead
lock dbproc variable if necessary.
 *
 */

int msg_handler(DBPROCESS *dbproc, DBINT msgno, int
msgstate, int severity, char *msgtext)
{
        PECBINFO
        pEcbInfo;
        EXTENSION_CONTROL_BLOCK       *pECB;
FILE                                            *fp;
SYSTEMTIME
        systemTime;
        char
        szTmp[256];
        int
                iTermId;
        int
                iSyncId;

        if ( !(pEcbInfo =
(PECBINFO)dbgetuserdata(dbproc)) )
        {
                pECB = gpECB;
                iTermId = 0;
                iSyncId = 0;
        }
        else
        {
                pECB = pEcbInfo->pECB;
                iTermId = pEcbInfo->iTermId;
                iSyncId = pEcbInfo->iSyncId;
        }


        if ( (msgno == 5701) || (msgno == 2528) ||
(msgno == 5703) || (msgno == 6006) )
                return INT_CONTINUE;

        // deadlock message
        if (msgno == 1205)
        {
                // set the deadlock indicator
                if ( pEcbInfo )
                        pEcbInfo->bDeadlock =
TRUE;
                else
                        TMLog("Error,
dbgetuserdata returned NULL.");
                return INT_CONTINUE;
        }
        if ( pEcbInfo && pEcbInfo->bFailed )
        {
                TMLog("SQL Error ");
                return INT_CANCEL;
        }

        if (msgno == 0)
                return INT_CONTINUE;
        else
        {
                TMLog("MsgHandler: SQL Error %s",
msgtext);

                if ( pEcbInfo )
                        pEcbInfo->bFailed = TRUE;

                bError = TRUE;

                sprintf(szTmp, "Error: SQLSVR(%d):
%s", msgno, msgtext);

                TMLog("%2.2d/%2.2d/%2.2d
%2.2d:%2.2d:%2.2d\r\n\r\n%s\r\n\r\n",
                        systemTime.wYear,
systemTime.wMonth, systemTime.wDay,
                        systemTime.wHour,
systemTime.wMinute, systemTime.wSecond,
                        szTmp);
        }
        return INT_CANCEL;
}


/* FUNCTION: BOOL GetParameters(int argc, char *argv[])
 *
 * PURPOSE:         This function parses the command
line passed in to the delivery executable, initializing
 *                          and filling in global
variable parameters.
 *
 * ARGUMENTS:       int                 argc
        number of command line arguments passed to
delivery
 *                                      char
        *argv[]   array of command line argument
pointers
 *
```

```
 * RETURNS:                   BOOL      FALSE
          parameter read successful
 *
          TRUE      user has requested parameter
information screen be displayed.
 *
 * COMMENTS:        None
 *
 */

static BOOL GetParameters(int argc, char *argv[])
{
          int i;

          szServer[0]                    = 0;
          szPassword[0]       = 0;
          bFlush                         = FALSE;
          strcpy(szDatabase, "tpcc");
          strcpy(szUser, "sa");

          for(i=0; i<argc; i++)
          {
                    if ( argv[i][0] == '-' ||
argv[i][0] == '/' )
                    {
                              switch(argv[i][1])
                              {
                                        case 'S':
                                        case 's':

          strcpy(szServer, argv[i]+2);

          break;
                                        case 'V':
                                        case 'v':

          verbose = TRUE;

          break;
                                        case 'F':
                                        case 'f':

          bFlush = TRUE;      //turn on delilog flush
when written.

          break;
                                        case '?':

          return TRUE;
                              }
                    }
          }
          return FALSE;
}

/* FUNCTION: void PrintParameters(void)
 *
 * PURPOSE:         This function displays the
supported command line flags.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                   None
 *
 * COMMENTS:        None
 *
 */

static void PrintParameters(void)
{
          TMLog("Performance Tuning Corporation Tuxedo
Kit");
          TMLog("   www.perftuning.com   (281) 251-3495
");
          TMLog("Delivery: -S Server [-v (verbose)] [-F
(Flush delilog)]" );
          TMLog("Delivery: Server %s Flush %d.",
szServer, bFlush);

}
```

Httpext.h

```
/********
*
*  Copyright (c) 1995  Process Software Corporation
*
*  Copyright (c) 1995  Microsoft Corporation
*
*
*  Module Name  : HttpExt.h
*
*  Abstract :
*
*     This module contains  the structure definitions
and prototypes for the
*     version 1.0 HTTP Server Extension interface.
*
*****************/

#ifndef _HTTPEXT_H_
#define _HTTPEXT_H_

#include <windows.h>

#ifdef __cplusplus
extern "C" {
#endif

#define   HSE_VERSION_MAJOR           1      // major
version of this spec
#define   HSE_VERSION_MINOR           0      // minor
version of this spec
#define   HSE_LOG_BUFFER_LEN         80
#define   HSE_MAX_EXT_DLL_NAME_LEN  256

typedef   LPVOID  HCONN;

// the following are the status codes returned by the
Extension DLL

#define   HSE_STATUS_SUCCESS                        1
#define   HSE_STATUS_SUCCESS_AND_KEEP_CONN          2
#define   HSE_STATUS_PENDING                        3
#define   HSE_STATUS_ERROR                          4

// The following are the values to request services
with the ServerSupportFunction.
//  Values from 0 to 1000 are reserved for future
versions of the interface

#define   HSE_REQ_BASE                              0
#define   HSE_REQ_SEND_URL_REDIRECT_RESP            (
HSE_REQ_BASE + 1 )
#define   HSE_REQ_SEND_URL                          (
HSE_REQ_BASE + 2 )
#define   HSE_REQ_SEND_RESPONSE_HEADER              (
HSE_REQ_BASE + 3 )
#define   HSE_REQ_DONE_WITH_SESSION                 (
HSE_REQ_BASE + 4 )
#define   HSE_REQ_END_RESERVED                      1000

//
//  These are Microsoft specific extensions
//

#define   HSE_REQ_MAP_URL_TO_PATH
(HSE_REQ_END_RESERVED+1)
#define   HSE_REQ_GET_SSPI_INFO
(HSE_REQ_END_RESERVED+2)


//
// passed to GetExtensionVersion
//

typedef struct   _HSE_VERSION_INFO {

    DWORD  dwExtensionVersion;
    CHAR   lpszExtensionDesc[HSE_MAX_EXT_DLL_NAME_LEN];

} HSE_VERSION_INFO, *LPHSE_VERSION_INFO;

//
// passed to extension procedure on a new request
//
typedef struct _EXTENSION_CONTROL_BLOCK {

    DWORD     cbSize;                 // size of this
struct.
    DWORD     dwVersion;              // version info
of this spec
    HCONN     ConnID;                 // Context number
not to be modified!
    DWORD     dwHttpStatusCode;       // HTTP Status
code
    CHAR      lpszLogData[HSE_LOG_BUFFER_LEN];// null
terminated log info specific to this Extension DLL

    LPSTR     lpszMethod;             // REQUEST_METHOD
    LPSTR     lpszQueryString;        // QUERY_STRING
    LPSTR     lpszPathInfo;           // PATH_INFO
```

```
    LPSTR     lpszPathTranslated;     //
PATH_TRANSLATED

    DWORD     cbTotalBytes;           // Total bytes
indicated from client
    DWORD     cbAvailable;            // Available
number of bytes
    LPBYTE    lpbData;                // pointer to
cbAvailable bytes

    LPSTR     lpszContentType;        // Content type
of client data

    BOOL (WINAPI * GetServerVariable) ( HCONN
hConn,
                                        LPSTR
lpszVariableName,




LPVOID      lpvBuffer,
                                        LPDWORD
lpdwSize );

    BOOL (WINAPI * WriteClient)  ( HCONN      ConnID,
                                   LPVOID     Buffer,
                                   LPDWORD
lpdwBytes,
                                   DWORD
dwReserved );

    BOOL (WINAPI * ReadClient)  ( HCONN      ConnID,
                                  LPVOID     lpvBuffer,
                                  LPDWORD    lpdwSize
);

    BOOL (WINAPI * ServerSupportFunction)( HCONN
hConn,
                                           DWORD
dwHSERRequest,
                                           LPVOID
lpvBuffer,
                                           LPDWORD
lpdwSize,
                                           LPDWORD
lpdwDataType );

} EXTENSION_CONTROL_BLOCK, *LPEXTENSION_CONTROL_BLOCK;

//
//  these are the prototypes that must be exported from
the extension DLL
//

BOOL  WINAPI   GetExtensionVersion( HSE_VERSION_INFO
*pVer );
DWORD WINAPI   HttpExtensionProc(
EXTENSION_CONTROL_BLOCK *pECB );

// the following type declarations is for the server
side

typedef BOOL  (WINAPI * PFN_GETEXTENSIONVERSION)(
HSE_VERSION_INFO  *pVer );
typedef DWORD (WINAPI * PFN_HTTPEXTENSIONPROC )(
EXTENSION_CONTROL_BLOCK *pECB );

#ifdef __cplusplus
}
#endif

#endif  // end definition _HTTPEXT_H_
```

Resource.h

```
//{{NO_DEPENDENCIES}}
// Microsoft Developer Studio generated include file.
// Used by TPCC.rc
//

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE        101
#define _APS_NEXT_COMMAND_VALUE         40001
#define _APS_NEXT_CONTROL_VALUE         1000
#define _APS_NEXT_SYMED_VALUE           101
#endif
#endif
```

Tpcc.h

```
/*        FILE:               TPCC.H
 *                                    Microsoft TPC-C
Kit Ver. 3.00.001
 *                                    Audited
08/23/96, By Francois Raab
 *
 *                                    Copyright
Microsoft, 1996
 *
 *        PURPOSE:  Header file for ISAPI TPCC.DLL,
defines structures and functions used in the isapi
tpcc.dll.
 *        Author:             Philip Durr
 *
          philipdu@Microsoft.com
 */

#define LOCAL_ALLOC 1

//VERSION RESOURCE DEFINES
#define _APS_NEXT_RESOURCE_VALUE
                    101
#define _APS_NEXT_COMMAND_VALUE
                                    40001
#define _APS_NEXT_CONTROL_VALUE
                                    1000
#define _APS_NEXT_SYMED_VALUE
                    101

#define TP_MAX_RETRIES
                                    50

#define ERR_BAD_ITEM_ID
                                    1
          //expected abort record in txnRecord
#define ERR_TYPE_DELIVERY_POST
                                    2
          //expected delivery post failed
#define   ERR_TYPE_WEBDLL
                                    3
          //tpcc web generated error
#define   ERR_TYPE_SQL
                                    4
          //sql server generated error
#define   ERR_TYPE_DBLIB
                                    5
          //dblib generated error
#define   ERR_TYPE_ODBC
                                    6
          //odbc generated error
#define ERR_TYPE_SOCKET
                                    7
          //error on communication socket client rte
only
#define ERR_TYPE_DEADLOCK
                                    8
          //dblib and odbc only deadlock condition
#define ERR_TYPE_TUXEDO
                                    9
          //tuxedo error

#define   ERR_SUCCESS
                                                    1000
          //"Success, no error.
#define   ERR_COMMAND_UNDEFINED
                                    1001      //"Command
undefined.
#define   ERR_NOT_IMPLEMENTED_YET
                                    1002      //"Not
Implemented Yet.
#define   ERR_CANNOT_INIT_TERMINAL
                    1003      //"Cannot initialize
client connection.
#define   ERR_OUT_OF_MEMORY
                                    1004      //"insufficient
memory.
#define   ERR_NEW_ORDER_NOT_PROCESSED
                    1005      //"Cannot process new
Order form.
#define   ERR_PAYMENT_NOT_PROCESSED
                    1006      //"Cannot process payment
form.
```

```
#define   ERR_NO_SERVER_SPECIFIED
                              1007      //"No Server
name specified.
#define   ERR_ORDER_STATUS_NOT_PROCESSED
                    1008      //"Cannot process order
status form.
#define   ERR_W_ID_INVALID
                              1009      //"Invalid
Warehouse ID.
#define   ERR_CAN_NOT_SET_MAX_CONNECTIONS
                    1010      //"Insufficient memory to
allocate # connections.
#define   ERR_NOSUCH_CUSTOMER
                              1011      //"No such
customer.
#define   ERR_D_ID_INVALID
                              1012      //"Invalid
District ID Must be 1 to 10.
#define   ERR_MAX_CONNECT_PARAM
                              1013      //"Max client
connections exceeded, run install to increase.
#define   ERR_INVALID_SYNC_CONNECTION
                    1014      //"Invalid Terminal Sync
ID.
#define   ERR_INVALID_TERMID
                              1015      //"Invalid
Terminal ID.
#define   ERR_PAYMENT_INVALID_CUSTOMER
          1016      //"Payment Form, No such Customer.
#define   ERR_SQL_OPEN_CONNECTION
                              1017
          //"SQLOpenConnection API Failed.
#define   ERR_STOCKLEVEL_MISSING_THRESHOLD_KEY    1018
          //"Stock Level missing Threshold key "TT*".
#define   ERR_STOCKLEVEL_THRESHOLD_INVALID
          1019      //"Stock Level Threshold invalid
data type range = 1 - 99.
#define   ERR_STOCKLEVEL_THRESHOLD_RANGE
                    1020      //"Stock Level Threshold
out of range, range must be 1 - 99.
#define   ERR_STOCKLEVEL_NOT_PROCESSED
          1021      //"Stock Level not processed.
#define   ERR_NEWORDER_FORM_MISSING_DID
          1022      //"New Order missing District key
"DID*".
#define   ERR_NEWORDER_DISTRICT_INVALID
          1023      //"New Order District ID Invalid
range 1 - 10.
#define   ERR_NEWORDER_DISTRICT_RANGE
                    1024      //"New Order District ID
out of Range. Range = 1 - 10.
#define   ERR_NEWORDER_CUSTOMER_KEY
                    1025      //"New Order missing
Customer key "CID*".
#define   ERR_NEWORDER_CUSTOMER_INVALID
          1026      //"New Order customer id invalid
data type, range = 1 to 3000.
#define   ERR_NEWORDER_CUSTOMER_RANGE
                    1027      //"New Order customer id
out of range, range = 1 to 3000.
#define   ERR_NEWORDER_MISSING_IID_KEY
          1028      //"New Order missing Item Id key
"IID*".
#define   ERR_NEWORDER_ITEM_BLANK_LINES
          1029      //"New Order blank order lines all
orders must be continuous.
#define   ERR_NEWORDER_ITEMID_INVALID
                    1030      //"New Order Item Id is
wrong data type, must be numeric.
#define   ERR_NEWORDER_MISSING_SUPPW_KEY
                    1031      //"New Order missing
Supp_W key "SP##*".
#define   ERR_NEWORDER_SUPPW_INVALID
                    1032      //"New Order Supp_W
invalid data type must be numeric.
#define   ERR_NEWORDER_MISSING_QTY_KEY
          1033      //"New Order Missing Qty key
"Qty##*".
#define   ERR_NEWORDER_QTY_INVALID
                    1034      //"New Order Qty invalid
must be numeric range 1 - 99.
#define   ERR_NEWORDER_SUPPW_RANGE
                    1035      //"New Order Supp_W value
out of range range = 1 - Max Warehouses.
#define   ERR_NEWORDER_ITEMID_RANGE
                    1036      //"New Order Item Id is
out of range. Range = 1 to 999999.
#define   ERR_NEWORDER_QTY_RANGE
                              1037      //"New Order
Qty is out of range. Range = 1 to 99.
#define   ERR_PAYMENT_DISTRICT_INVALID
          1038      //"Payment District ID is invalid
must be 1 - 10.
#define   ERR_NEWORDER_SUPPW_WITHOUT_ITEMID
          1039      //"New Order Supp_W field entered
without a corrisponding Item_Id.
#define   ERR_NEWORDER_QTY_WITHOUT_ITEMID
                    1040      //"New Order Qty entered
without a corrisponding Item_Id.
#define   ERR_NEWORDER_NOITEMS_ENTERED
          1041      //"New Order Blank Items between
items, items must be continuous.
#define   ERR_PAYMENT_MISSING_DID_KEY
                    1042      //"Payment missing
District Key "DID*".
#define   ERR_PAYMENT_DISTRICT_RANGE
                    1043      //"Payment District Out
of range, range = 1 - 10.
#define   ERR_PAYMENT_MISSING_CID_KEY
                    1044      //"Payment missing
Customer Key "CID*".
#define   ERR_PAYMENT_CUSTOMER_INVALID
          1045      //"Payment Customer data type
invalid, must be numeric.
#define   ERR_PAYMENT_MISSING_CLT
                              1046      //"Payment
missing Customer Last Name Key "CLT*".
#define   ERR_PAYMENT_LAST_NAME_TO_LONG
          1047      //"Payment Customer last name
longer than 16 characters.
#define   ERR_PAYMENT_CUSTOMER_RANGE
                    1048      //"Payment Customer ID
out of range, must be 1 to 3000.
#define   ERR_PAYMENT_CID_AND_CLT
                              1049      //"Payment
Customer ID and Last Name entered must be one or other.
#define   ERR_PAYMENT_MISSING_CDI_KEY
                    1050      //"Payment missing
Customer district key "CDI*".
#define   ERR_PAYMENT_CDI_INVALID
                              1051      //"Payment
Customer district invalid must be numeric.
#define   ERR_PAYMENT_CDI_RANGE
                              1052      //"Payment
Customer district out of range must be 1 - 10.
#define   ERR_PAYMENT_MISSING_CWI_KEY
                    1053      //"Payment missing
Customer Warehouse key "CWI*".
#define   ERR_PAYMENT_CWI_INVALID
                              1054      //"Payment
Customer Warehouse invalid must be numeric.
#define   ERR_PAYMENT_CWI_RANGE
                              1055      //"Payment
Customer Warehouse out of range, 1 to Max Warehouses.
#define   ERR_PAYMENT_MISSING_HAM_KEY
                    1056      //"Payment missing Amount
key "HAM*".
#define   ERR_PAYMENT_HAM_INVALID
                              1057      //"Payment
Amount invalid data type must be numeric.
#define   ERR_PAYMENT_HAM_RANGE
                              1058      //"Payment
Amount out of range, 0 - 9999.99.
#define   ERR_ORDERSTATUS_MISSING_DID_KEY
                    1059      //"Order Status missing
District key "DID*".
#define   ERR_ORDERSTATUS_DID_INVALID
                    1060      //"Order Status District
invalid, value must be numeric 1 - 10.
#define   ERR_ORDERSTATUS_DID_RANGE
                    1061      //"Order Status District
out of range must be 1 - 10.
#define   ERR_ORDERSTATUS_MISSING_CID_KEY
                    1062      //"Order Status missing
Customer key "CID*".
#define   ERR_ORDERSTATUS_MISSING_CLT_KEY
                    1063      //"Order Status missing
Customer Last Name key "CLT*".
#define   ERR_ORDERSTATUS_CLT_RANGE
                    1064      //"Order Status Customer
last name longer than 16 characters.
#define   ERR_ORDERSTATUS_CID_INVALID
                    1065      //"Order Status Customer
ID invalid, range must be numeric 1 - 3000.
#define   ERR_ORDERSTATUS_CID_RANGE
                    1066      //"Order Status Customer
ID out of range must be 1 - 3000.
#define   ERR_ORDERSTATUS_CID_AND_CLT
                    1067      //"Order Status Customer
ID and LastName entered must be only one."
#define   ERR_DELIVERY_MISSING_OCD_KEY
          1068      //"Delivery missing Carrier ID key
\"OCD*\".
#define   ERR_DELIVERY_CARRIER_INVALID
          1069      //"Delivery Carrier ID invalid must
be numeric 1 - 10.
```

```
#define   ERR_DELIVERY_CARRIER_ID_RANGE
          1070      //"Delivery Carrier ID out of range
must be 1 - 10.
#define ERR_PAYMENT_MISSING_CLT_KEY
                    1071      //"Payment missing
Customer Last Name key "CLT*".
#define ERR_TPINIT_BAD                              5001
          //"Bad TPINIT"
#define ERR_TPALLOC_BAD                             5002
          //"Bad TPALLOC"
#define ERR_TPCALL_BAD                              5003
          //"Bad TPCALL"

//note that the welcome form must be processed first as
terminal ids assigned here, once the
//terminal id is assigned then the forms can be
processed in any order.
#define WELCOME_FORM
                              1
          //beginning form no term id assigned, form id
#define MAIN_MENU_FORM
                              2
          //term id assigned main menu form id
#define NEW_ORDER_FORM
                              3                   //new
order form id
#define PAYMENT_FORM
                              4
          //payment form id
#define DELIVERY_FORM
                              5
          //delivery form id
#define ORDER_STATUS_FORM
                    6                   //order status
id
#define STOCK_LEVEL_FORM
                    7                   //stock level
form id

//This macro is used to prevent the compiler error
unused formal parameter
#define UNUSEDPARAM(x) (x = x)

//error message structure used in ErrorMessage API
typedef struct _SERRORMSG
{
          int                 iError;
                    //error id of message
          char      szMsg[80];
          //message to sent to browser
} SERRORMSG;

//This structure is used for posting delivery
transactions
typedef struct _DELIVERY_TRANSACTION
{
          SYSTEMTIME          queue;
          //time delivery transaction queued
          short               w_id;
          //delivery warehouse
          short               o_carrier_id;
          //carrier id
} DELIVERY_TRANSACTION;

#ifdef USE_ODBC
          typedef struct _DBPROCESS
          {
                    HDBC      hdbc;
                    HSTMT     hstmt;
                    int                 spid;
                    void      *uPtr;
          } DBPROCESS, *PDBPROCESS;

          //dblib error message return values
          #define INT_EXIT        0
          #define INT_CONTINUE    1
          #define INT_CANCEL      2
#endif

//This structure defines the data necessary to keep
distinct for each terminal or client connection.
typedef struct _CLIENTDATA
{
          int                           inUse;
                                        //in use flag
allows client entries to be reused
          int                           w_id;
                                        //warehouse id
assigned at welcome form
          int                           d_id;
                                        //district id
assigned at welcome form

          PDBPROCESS          dbproc;
                                        //dblib connection
pointer
          int                           spid;
                                        //spid assigned
from dblib
          int                           iSyncId;
                                        //syncronization id
          int                           iTickCount;
                                        //time of last
access;
          int                           iTermId;
                                        //terminal id of http
stream connection

          char
          szBuffer[4096];     //form buffer each HTML
form is built for a client in here

          NEW_ORDER_DATA                NewOrderData;
                    //new order form data
          PAYMENT_DATA                  PaymentData;
                    //payment form data
          ORDER_STATUS_DATA   OrderStatusData;
          //order status form data
          DELIVERY_DATA                 DeliveryData;
                    //delivery form data
          STOCK_LEVEL_DATA    StockLevelData;
          //stock level form data

#ifndef LOCAL_ALLOC
          NEW_ORDER_DATA
          *NewOrderDataPtr;
          PAYMENT_DATA
          *PaymentDataPtr;
          ORDER_STATUS_DATA   *OrderStatusDataPtr;
          DELIVERY_DATA
          *DeliveryDataPtr;
          STOCK_LEVEL_DATA    *StockLevelDataPtr;
#endif // LOCAL_ALLOC

} CLIENTDATA;

typedef CLIENTDATA *PCLIENTDATA;
          //pointer to client structure

//This structure is used to define the operational
interface for terminal id support
typedef struct _TERM
{
          int                           iAvailable;

          //total allocated terminal array entries
          int                           iNext;

          //next available terminal array element
          int                           iMasterSyncId;

          //syncronization id
          BOOL                bInit;

          //structure has been initialized flag
          CLIENTDATA          *pClientData;
                                        //pointer to
allocated client data
          void                (*Init)(void);
                                        //API to
initialize this structure
          int
          (*Allocate)(void);
          //API to allocate a new terminal entry array
id returned
          void                (*Restore)(void);
                                        //API to free terminal
data
          int
          (*Add)(EXTENSION_CONTROL_BLOCK *pECB, char
*pQueryString);     //API to add a terminal id to
array, this context will


                              //be passed from the browser to the
tpcc.dll in the


                              //TERMID= key in the HTTP string.
          void
          (*Delete)(EXTENSION_CONTROL_BLOCK *pECB, int
```

```
id);      //API to free resources used by a terminal
array entry
} TERM;

typedef TERM *PTERM;
                                        //pointer to
terminal structure type

//this structure allows the EXTENSION CONTROL BLOCK to
be passed to the msg and error handlers.
typedef struct _ECBINFO
{
          int
                    iTermId;  //terminal id
          int
                    iSyncId;  //browser sync id
          BOOL
          bDeadlock;          //deadlock condition flag
          BOOL
          bFailed;  //cleared before sql transaction,
set in err handlers if an error occurs
          EXTENSION_CONTROL_BLOCK *pECB;
          //inetsrv current connection structure
information
} ECBINFO, *PECBINFO;


//function prototypes

BOOL APIENTRY DllMain(HANDLE hModule, DWORD
ul_reason_for_call, LPVOID lpReserved);
static void DeliveryDisconnect(void *ptr);
static BOOL IsValidTermId(int TermId);
BOOL ProcessQueryString(EXTENSION_CONTROL_BLOCK *pECB,
int *pCmd, int *pFormId, int *pTermId, int *pSyncId);
void NewOrderForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void PaymentForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void DeliveryForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void OrderStatusForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void StockLevelForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void Exitcmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void SubmitCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void BeginCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void ProcessCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void ClearCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void MenuCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId);
void NumberOfConnectionsCmd(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId);
static void WriteZString(EXTENSION_CONTROL_BLOCK *pECB,
char *szStr);
static void h_printf(EXTENSION_CONTROL_BLOCK *pECB,
char *format, ...);
void LogTuxError(int TpRc, char *ErrMessage);
void ErrorMessage(EXTENSION_CONTROL_BLOCK *pECB, int
iError, int iErrorType, char *szMsg, int iTermId, int
iSyncId);
static BOOL GetKeyValue(char *pQueryString, char *pKey,
char *pValue, int iMax);
static void TermInit(void);
int err_handler(DBPROCESS *dbproc, int severity, int
dberr, int oserr, char *dberrstr, char *oserrstr);
int msg_handler(DBPROCESS *dbproc, DBINT msgno, int
msgstate, int severity, char *msgtext);
static void TermRestore(void);
static int TermAllocate(void);
static int TermAdd(EXTENSION_CONTROL_BLOCK *pECB, char
*pQueryString);
static void TermDelete(EXTENSION_CONTROL_BLOCK *pECB,
int id);
BOOL Init(EXTENSION_CONTROL_BLOCK *pECB, int iTermId,
int iSyncId, char *szServer, char *szUser, char
*szPassword, char *szDatabase);
static BOOL Close(EXTENSION_CONTROL_BLOCK *pECB, int
iTermId, int iSyncId);
static BOOL SQLOpenConnection(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId, DBPROCESS **dbproc,
char *server, char *database, char *user, char
*password, char *app, int *spid);
static BOOL SQLCloseConnection(EXTENSION_CONTROL_BLOCK
*pECB, DBPROCESS *dbproc);
static BOOL SQLStockLevel(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId, DBPROCESS *dbproc,
STOCK_LEVEL_DATA *pStockLevel, short deadlock_retry);
static int SQLNewOrder(EXTENSION_CONTROL_BLOCK
          *pECB, int iTermId, int iSyncId, DBPROCESS
*dbproc, NEW_ORDER_DATA  *pNewOrder, short
deadlock_retry);
static int SQLPayment(EXTENSION_CONTROL_BLOCK *pECB,
int iTermId, int iSyncId, DBPROCESS *dbproc,
PAYMENT_DATA *pPayment, short deadlock_retry);
static int SQLOrderStatus(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId, DBPROCESS *dbproc,
ORDER_STATUS_DATA *pOrderStatus, short deadlock_retry);
static int SQLDelivery(DBPROCESS *dbproc, DELIVERY_DATA
*pDelivery, short deadlock_retry);
static void WriteLog(DELIVERY_DATA *pDelivery,
SYSTEMTIME *trans_end );
static void CalculateElapsedTime(int *pElapsed,
LPSYSTEMTIME lpBegin, LPSYSTEMTIME lpEnd);
BOOL SQLDetectDeadlock(DBPROCESS *dbproc);
static void FormatString(char *szDest, char *szPic,
char *szSrc);
static char *MakeStockLevelForm(int iTermId, int
iSyncId, BOOL bInput);
static char *MakeMainMenuForm(int iTermId, int
iSyncId);
static char *MakeWelcomeForm(void);
static char *MakeNewOrderForm(int iTermId, int iSyncId,
BOOL Rollback, BOOL bInput, BOOL bValid);
static char *MakePaymentForm(int iTermId, int iSyncId,
BOOL bInput);
static char *MakeOrderStatusForm(int iTermId, int
iSyncId, BOOL bInput);
static char *MakeDeliveryForm(int iTermId, int iSyncId,
BOOL bInput, BOOL bSuccess);
static void UtilStrCpy(char * pDest, char * pSrc, int
n);
static void ProcessNewOrderForm(EXTENSION_CONTROL_BLOCK
          *pECB, int iTermId, int iSyncId);
static void ProcessPaymentForm(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId);
static void
ProcessOrderStatusForm(EXTENSION_CONTROL_BLOCK *pECB,
int iTermId, int iSyncId);
static void ProcessDeliveryForm(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId);
static void
ProcessStockLevelForm(EXTENSION_CONTROL_BLOCK *pECB,
int iTermId, int iSyncId);
static int GetNewOrderData(LPSTR lpszQueryString,
NEW_ORDER_DATA *pNewOrderData);
static int GetPaymentData(LPSTR lpszQueryString,
PAYMENT_DATA *pPaymentData);
static int GetOrderStatusData(LPSTR lpszQueryString,
ORDER_STATUS_DATA *pOrderStatusData);
static BOOL ReadRegistrySettings(void);
static BOOL PostDeliveryInfo(short w_id, short
o_carrier_id);
static BOOL IsNumeric(char *ptr);
static void FormatHTMLString(char *szBuff, char *szStr,
int iLen);
static void         PrintParameters(void);
static BOOL GetParameters(int argc, char *argv[]);
static int OpenLogFile(void);

#ifdef USE_ODBC
          void dbsetuserdata(PDBPROCESS dbproc, void
*uPtr);
          void *dbgetuserdata(PDBPROCESS dbproc);
          void BindParameter(PDBPROCESS dbproc, UWORD
ipar, SWORD fCType, SWORD fSqlType, UDWORD cbColDef,
SWORD ibScale, PTR rgbValue, SDWORD cbValueMax);
          void ODBCError(PDBPROCESS dbproc);
          BOOL ExecuteStatement(PDBPROCESS dbproc, char
*szStatement);
          BOOL BindColumn(PDBPROCESS dbproc,
SQLUSMALLINT icol, SQLSMALLINT fCType, SQLPOINTER
rgbValue, SQLINTEGER cbValueMax);
          BOOL GetResults(PDBPROCESS dbproc);
          BOOL MoreResults(PDBPROCESS  dbproc);
          BOOL ReopenConnection(PDBPROCESS dbproc);
#endif
```

Trans.h

```
/*        FILE:               TRANS.H
 *                                        Microsoft TPC-C
Kit Ver. 3.00.000
 *                                        Audited
08/23/96  By Francois Raab
```

```
 *        PURPOSE:  Header file for ISAPI TPCC.DLL,
defines structures and functions used in the isapi
tpcc.dll.
 *
 *                                    Copyright
Microsoft inc. 1996, All Rights Reserved
 *
 *        Author:             PhilipDu, from tpcc.h by
DamienL
 *
          DamienL@Microsoft.com
 *
          philipdu@Microsoft.com
 */

#ifndef _INC_TRANS

          #define _INC_TRANS

          #ifdef USE_ODBC
                    #ifndef TIMESTAMP_STRUCT
                              #include <sqltypes.h>
                    #endif
          #else
                    #ifndef _INC_SQLFRONT
                              #include <sqlfront.h>
                    #endif
          #endif

          #ifndef DBINT
                    typedef long DBINT;
          #endif

          #define   DEFCLPACKSIZE
                    4096
          #define   DEADLOCKWAIT
                    10

          // String length constants
          #define SERVER_NAME_LEN           20
          #define DATABASE_NAME_LEN         20
          #define USER_NAME_LEN             20
          #define PASSWORD_LEN              20
          #define TABLE_NAME_LEN            20
          #define I_DATA_LEN                50
          #define I_NAME_LEN                24
          #define BRAND_LEN                  1
          #define LAST_NAME_LEN             16
          #define W_NAME_LEN                10
          #define ADDRESS_LEN               20
          #define STATE_LEN                  2
          #define ZIP_LEN                    9
          #define S_DIST_LEN                24
          #define S_DATA_LEN                50
          #define D_NAME_LEN                10
          #define FIRST_NAME_LEN            16
          #define MIDDLE_NAME_LEN            2
          #define PHONE_LEN                 16
          #define DATETIME_LEN              30
          #define CREDIT_LEN                 2
          #define C_DATA_LEN               250
          #define H_DATA_LEN                24
          #define DIST_INFO_LEN             24
          #define MAX_OL_NEW_ORDER_ITEMS     15
          #define MAX_OL_ORDER_STATUS_ITEMS  15
          #define STATUS_LEN                25
          #define OL_DIST_INFO_LEN                  24

          // transaction structures

          typedef struct
          {
                    short
          ol_supply_w_id;
                    long
          ol_i_id;
                    char
          ol_i_name[I_NAME_LEN+1];
                    short
          ol_quantity;
                    char
          ol_brand_generic[BRAND_LEN+1];
                    double
          ol_i_price;
                    double
          ol_amount;
                    short
          ol_stock;
                    short
          num_warehouses;
          } OL_NEW_ORDER_DATA;

          typedef struct
          {
                    short               w_id;
                    short               d_id;
                    long                c_id;
                    short               o_ol_cnt;
                    char
          c_last[LAST_NAME_LEN+1];
                    char
          c_credit[CREDIT_LEN+1];
                    double              c_discount;
                    double              w_tax;
                    double              d_tax;
                    long                          o_id;
                    short
          o_commit_flag;
          #ifdef USE_ODBC
                    TIMESTAMP_STRUCT    o_entry_d;
          #else
                    DBDATEREC           o_entry_d;
          #endif
                    short               o_all_local;
                    double              total_amount;
                    long                num_deadlocks;
                    int
                    retval;
                    int
                    error;
                    char
          execution_status[STATUS_LEN];
                    OL_NEW_ORDER_DATA
          Ol[MAX_OL_NEW_ORDER_ITEMS];
          } NEW_ORDER_DATA;

          typedef struct
          {
                    short
           w_id;
                    short
           d_id;
                    long
           c_id;
                    short
           c_d_id;
                    short
           c_w_id;
                    double
           h_amount;
          #ifdef USE_ODBC
                    TIMESTAMP_STRUCT     h_date;
          #else
                    DBDATEREC
h_date;
          #endif
                    char
           w_street_1[ADDRESS_LEN+1];
                    char
           w_street_2[ADDRESS_LEN+1];
                    char
           w_city[ADDRESS_LEN+1];
                    char
           w_state[STATE_LEN+1];
                    char
           w_zip[ZIP_LEN+1];
                    char
           d_street_1[ADDRESS_LEN+1];
                    char
           d_street_2[ADDRESS_LEN+1];
                    char
           d_city[ADDRESS_LEN+1];
                    char
           d_state[STATE_LEN+1];
                    char
d_zip[ZIP_LEN+1];
                    char
           c_first[FIRST_NAME_LEN+1];
                    char
           c_middle[MIDDLE_NAME_LEN + 1];
                    char
c_last[LAST_NAME_LEN+1];
                    char
           c_street_1[ADDRESS_LEN+1];
                    char
           c_street_2[ADDRESS_LEN+1];
                    char
           c_city[ADDRESS_LEN+1];
                    char
           c_state[STATE_LEN+1];
                    char
           c_zip[ZIP_LEN+1];
                    char
           c_phone[PHONE_LEN+1];
          #ifdef USE_ODBC
                    TIMESTAMP_STRUCT     c_since;
```

```
          #else
                    DBDATEREC
c_since;
          #endif
                    char
           c_credit[CREDIT_LEN+1];
                    double
           c_credit_lim;
                    double
           c_discount;
                    double
           c_balance;
                    char
           c_data[200+1];
                    long
           num_deadlocks;
                    int
                     retval;
                    int
                     error;
                    char
           execution_status[STATUS_LEN];
          } PAYMENT_DATA;

          typedef struct
          {
                    long
          ol_i_id;
                    short
          ol_supply_w_id;
                    short
          ol_quantity;
                    double
          ol_amount;
          #ifdef USE_ODBC
                    TIMESTAMP_STRUCT    ol_delivery_d;
          #else
                    DBDATEREC
          ol_delivery_d;
          #endif
          } OL_ORDER_STATUS_DATA;

          typedef struct
          {
                    short               w_id;
                    short               d_id;
                    long                c_id;
                    char
c_first[FIRST_NAME_LEN+1];
                    char
c_middle[MIDDLE_NAME_LEN+1];
                    char
c_last[LAST_NAME_LEN+1];
                    double              c_balance;
                    long                o_id;
          #ifdef USE_ODBC
                    TIMESTAMP_STRUCT    o_entry_d;
          #else
                    DBDATEREC           o_entry_d;
          #endif
                    short               o_carrier_id;
                    OL_ORDER_STATUS_DATA
OlOrderStatusData[MAX_OL_ORDER_STATUS_ITEMS];
                    short               o_ol_cnt;

                    long                num_deadlocks;
                    int
                     retval;
                    //int
                     error;
                    char
execution_status[STATUS_LEN];
          } ORDER_STATUS_DATA;

          typedef struct
          {
                    long
          o_id;
          } DEL_ITEM;

          typedef struct
          {
                    short                         w_id;
                    short
          o_carrier_id;
                    SYSTEMTIME
          queue_time;
                    long
          num_deadlocks;
                    long
          o_id[10];
                    int
                    retval;
                    int
                    error;
                    char
          execution_status[STATUS_LEN];
          } DELIVERY_DATA;

          typedef struct
          {
                    short
          w_id;
                    short
          d_id;
                    short
          thresh_hold;
                    long
          low_stock;
                    long
          num_deadlocks;
                    int
                    retval;
                    int
                    error;
                    char
          execution_status[STATUS_LEN];
          } STOCK_LEVEL_DATA;

          typedef struct
          {
                    NEW_ORDER_DATA
          NewOrderData;               //new order
form data
                    PAYMENT_DATA
          PaymentData;                //payment form
data
                    ORDER_STATUS_DATA
          OrderStatusData;    //order status form data
                    DELIVERY_DATA
          DeliveryData;               //delivery form
data
                    STOCK_LEVEL_DATA    StockLevelData;
                    //stock level form data
          } TUX_DATA;

#endif
```

Tpcc.c

```
/*        FILE:               TPCC.C
 *
 *        Based on: Microsoft TPC-C Kit Ver. 3.00.000
 *
 *                                  Copyright
Microsoft, 1996
 *                                  Copyright
Performance Tuning Corporation, 1997
 *
 *        PURPOSE:  TPC-C main program.
 *        Author:             Philip Durr
 *
          philipdu@Microsoft.com
 *
 *  MODIFIED        Changed for modularity and to allow
for the Tuxedo TM
 *
 *  Author:                   Edward Whalen
 *                                  Performance
Tuning Corporation
 *
          ewhalen@perftuning.com
 *
*/
#define LOCAL_ALLOC 1
#define MULTI_TRANS 1

#include <windows.h>
#include <process.h>
#include <stdio.h>
#include <stdarg.h>
#include <malloc.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <sys\timeb.h>
#include <io.h>

#define DBNTWIN32
#include <sqlfront.h>
#include <sqldb.h>
```

Appendix A-Application Code

```
#include "trans.h"
                    //tpckit transaction header
contains definations of structures specific to TPC-C
#include "httpext.h"
                    //ISAPI DLL information header

#include "tpcc.h"
                    //this dlls specific structure,
value e.t. header.

#include <tmenv.h>
#include <xa.h>
#include <atmi.h>

static TPINIT       *tpinf;
static DWORD        TLSIsTpInitedKey;
static DWORD        TLSNewOrderKey;
static DWORD        TLSPaymentKey;
static DWORD        TLSOrderStatusKey;
static DWORD        TLSDeliveryKey;
static DWORD        TLSStockLevelKey;
static int                  ThrTpInit();

char
qname[10][4]={"T1","T2","T3","T4","T5","T6","T7","T8","
T9","T10"};
char      szServer[32]        = "EDW";
          //global variables used with this DLL
char      szUser[32]                  = "sa";
char      szPassword[32]      = "";
char      szDatabase[32]      = "tpcc";

char
szIID[16][8]={"IID00*","IID01","IID02","IID03","IID04",
"IID05","IID06","IID07","IID08","IID09","IID10","IID11"
,"IID12","IID13","IID14","IID15"};

char
szSP[16][8]={"SP00*","SP01","SP02","SP03","SP04","SP05"
,"SP06","SP07","SP08","SP09","SP10","SP11","SP12","SP13
","SP14","SP15"};

char
szQty[16][8]={"Qty00*","Qty01","Qty02","Qty03","Qty04",
"Qty05","Qty06","Qty07","Qty08","Qty09","Qty10","Qty11"
,"Qty12","Qty13","Qty14","Qty15"};

BOOL      bLog                        = FALSE;
BOOL      dLog                        = FALSE;

int       iThreads            = 5;
int       iMaxWareHouses      = 500;
int       iDelayMs            = 100;
short     iDeadlockRetry      = (short)3;
short     iMaxConnections = (short)25;
int                 iErrVal                     = 0;

char buffer[256];

//allowable client command strings i.e. CMD=command
char *szCmds[] =
{
          "..NewOrder..", "..Payment..",
"..Delivery..", "..Order-Status..", "..Stock-Level..",
"..Exit..",
          "Submit", "Begin", "Process", "Menu",
"Clear", "Users", ""
};

//defined command string functions, called via
CMD=command http string from html client.

void (*DoCmd[])(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId) =
{
          NewOrderForm,
          PaymentForm,
          DeliveryForm,
          OrderStatusForm,
          StockLevelForm,
          Exitcmd,
          SubmitCmd,
          BeginCmd,
          ProcessCmd,
          MenuCmd,
          ClearCmd
};


//Terminal client id structure and interface defination
TERM      Term = { 0, 0, 0, FALSE, NULL, TermInit,
TermAllocate, TermRestore, TermAdd, TermDelete };

//welcome to tpc-c html form buffer, this is first form
client sees.
static char         *szWelcomeForm =    "<HTML>"

                              "<HEAD><TITLE>Welcome To
TPC-C</TITLE></HEAD><BODY>"

                              "Please Identify your
Warehouse and District for this session.<BR>"

                              "<FORM
ACTION=\"tpcc.dll\" METHOD=\"GET\">"

                              "<INPUT TYPE=\"hidden\"
NAME=\"STATUSID\" VALUE=\"0\">"

                              "<INPUT TYPE=\"hidden\"
NAME=\"ERROR\" VALUE=\"0\">"

                              "<INPUT TYPE=\"hidden\"
NAME=\"FORMID\" VALUE=\"1\">"

                              "<INPUT TYPE=\"hidden\"
NAME=\"TERMID\" VALUE=\"-2\">"

                              "<INPUT TYPE=\"hidden\"
NAME=\"SYNCID\" VALUE=\"0\">"

                              "Warehouse ID <INPUT
NAME=\"w_id\" SIZE=4><BR>"

                              "District ID <INPUT
NAME=\"d_id\" SIZE=2><BR>"

                              "<HR>"

                              "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"Submit\">"

                              "</FORM><BODY>"

                              "</HTML>";


static char         szTpccLogPath[256]; //path to html
log file if logging turned on in registry.
static char szErrorLogPath[256];        //path to error
log file.

static    CRITICAL_SECTION
          CriticalSection;
static    CRITICAL_SECTION
          ErrorLogCriticalSection;

static    EXTENSION_CONTROL_BLOCK       *gpECB;
static    int
                    bTpccExit;          //exit delivery
disconnect loop as dll exiting.

/* FUNCTION: BOOL APIENTRY DllMain(HANDLE hModule,
DWORD ul_reason_for_call, LPVOID lpReserved)
 *
 * PURPOSE:         This function is the entry point
for the DLL this implementation is baised on the
 *                              fact that
DLL_PROCESS_ATTACH is only called from the inet service
once. Connections
 *                              are sent to this function
as thread attachments.
 *
 * ARGUMENTS:       HANDLE    hModule
                    module handle
 *                                        DWORD
          ul_reason_for_call  reason for call
 *                                        LPVOID
          lpReserved
          reserved for future use
 *
 * RETURNS:                   BOOL      FALSE
                              errors occured in
initialization
 *
          TRUE                                        DLL
successfully initialized
 *
 * COMMENTS:        None
 *
 */

BOOL APIENTRY DllMain(HANDLE hModule, DWORD
ul_reason_for_call, LPVOID lpReserved)
{
```

```
          int
          i;
          static SECURITY_ATTRIBUTES    sa;
          static PSECURITY_DESCRIPTOR pSD;

          switch( ul_reason_for_call )
          {
                    case DLL_PROCESS_ATTACH:
                              if (
ReadRegistrySettings() )
                              {

          MessageBox(NULL, "Cannot Find TPCC Key in
registry (run install.exe).", "Init", MB_OK |
MB_ICONSTOP);
                                        return FALSE;
                              }


          InitializeCriticalSection(&CriticalSection);

          InitializeCriticalSection(&ErrorLogCriticalSe
ction);

                              (*Term.Init)();
                              if ( !(*Term.Allocate)()
)
                              {

          MessageBox(NULL, "Error Trm.Allocate().",
"Init", MB_OK | MB_ICONSTOP);
                                        return FALSE;
                              }
                              for(i=Term.iNext;
i<Term.iAvailable; i++)

          Term.pClientData[i].inUse = 0;
                              Term.pClientData[0].inUse
= 1;

                              TLSIsTpInitedKey =
TlsAlloc(); // check for failure later
                              TLSNewOrderKey =
TlsAlloc();
                              TLSPaymentKey =
TlsAlloc();
                              TLSOrderStatusKey =
TlsAlloc();
                              TLSDeliveryKey =
TlsAlloc();
                              TLSStockLevelKey =
TlsAlloc();
                              // assumption:value
inited to 0
                              break;
                    case DLL_THREAD_ATTACH:
                              break;
                    case DLL_THREAD_DETACH:
                              break;
                    case DLL_PROCESS_DETACH:
                              if ( pSD )
                                        free( pSD );

                              bTpccExit = TRUE;

                              (*Term.Restore)();


          DeleteCriticalSection(&CriticalSection);

          DeleteCriticalSection(&ErrorLogCriticalSectio
n);


          TlsFree(TLSIsTpInitedKey);
                              TlsFree(TLSNewOrderKey);
                              TlsFree(TLSPaymentKey);

          TlsFree(TLSOrderStatusKey);
                              TlsFree(TLSDeliveryKey);

          TlsFree(TLSStockLevelKey);
                              break;
          }
          return TRUE;
}


/* FUNCTION: BOOL WINAPI
GetExtensionVersion(HSE_VERSION_INFO *pVer)
 *
 * PURPOSE:         This function is called by the inet
service when the DLL is first loaded.
 *
 * ARGUMENTS:       HSE_VERSION_INFO    *pVer
          passed in structure in which to place
expected version number.
 *
 * RETURNS:                   TRUE      inet service
expected return value.
 *
 * COMMENTS:        None
 *
 */

BOOL WINAPI GetExtensionVersion(HSE_VERSION_INFO *pVer)
{

          pVer->dwExtensionVersion =
MAKELONG(HSE_VERSION_MINOR, HSE_VERSION_MAJOR);
          lstrcpyn(pVer->lpszExtensionDesc, "TPC-C
Server.", HSE_MAX_EXT_DLL_NAME_LEN);

          return TRUE;
}

/* FUNCTION: DWORD WINAPI
HttpExtensionProc(EXTENSION_CONTROL_BLOCK *pECB)
 *
 * PURPOSE:         This function is the main entry
point for the TPCC DLL. The internet service
 *                           calls this function
passing in the http string.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
          structure pointer to passed in internet
 *
                    service information.
 *
 * RETURNS:                   DWORD
          HSE_STATUS_SUCCESS
                    connection can be dropped if error
 *
          HSE_STATUS_SUCCESS_AND_KEEP_CONN        keep
connect valid comment sent
 *
 * COMMENTS:        None
 *
 */

DWORD WINAPI HttpExtensionProc(EXTENSION_CONTROL_BLOCK
*pECB)
{
          int iCmd, FormId, TermId, iSyncId;
          FILE *fp;

          static BOOL bReadRegistry = FALSE;

          if ( iMaxConnections == -1 )
          {
                    ErrorMessage(pECB,
ERR_CAN_NOT_SET_MAX_CONNECTIONS, ERR_TYPE_WEBDLL, NULL,
-1, -1);
                    return HSE_STATUS_SUCCESS;
          }

          //if registry setting is for html logging
then show http string passed in.
          if ( bLog )
          {
                    SYSTEMTIME          systemTime;

                    fp = fopen(szTpccLogPath, "ab");

                    GetLocalTime(&systemTime);

                    fprintf(fp, "* QUERY *
%2.2d/%2.2d/%2.2d %2.2d:%2.2d:%2.2d\r\n\r\n%s\r\n\r\n",
                              systemTime.wYear,
systemTime.wMonth, systemTime.wDay,
                              systemTime.wHour,
systemTime.wMinute, systemTime.wSecond,
                              pECB->lpszQueryString);
                    fclose(fp);
          }


          //process http query
          if ( !ProcessQueryString(pECB, &iCmd,
&FormId, &TermId, &iSyncId) )
          {
                    if ( TermId < 0 )
                              ErrorMessage(pECB,
ERR_INVALID_TERMID, ERR_TYPE_WEBDLL, NULL, TermId,
iSyncId);
```

```
                    else
                              ErrorMessage(pECB,
ERR_COMMAND_UNDEFINED, ERR_TYPE_WEBDLL, NULL, TermId,
iSyncId);
                    return
HSE_STATUS_SUCCESS_AND_KEEP_CONN;
          }

          if ( TermId != 0 )
          {
                    if ( !IsValidTermId(TermId) )
                    {
                              ErrorMessage(pECB,
ERR_INVALID_TERMID, ERR_TYPE_WEBDLL, NULL, TermId,
iSyncId);
                              return
HSE_STATUS_SUCCESS_AND_KEEP_CONN;
                    }

                    //must have a valid syncid here
since termid is valid
                    if ( iSyncId < 1 || iSyncId !=
Term.pClientData[TermId].iSyncId )
                    {
                              ErrorMessage(pECB,
ERR_INVALID_SYNC_CONNECTION, ERR_TYPE_WEBDLL, NULL,
TermId, iSyncId);
                              return
HSE_STATUS_SUCCESS_AND_KEEP_CONN;
                    }

          }

          //set use time
          Term.pClientData[TermId].iTickCount =
GetTickCount();

          //go execute http: command
          (*DoCmd[iCmd])(pECB, FormId, TermId,
iSyncId);

          //finish up and keep connection
          return HSE_STATUS_SUCCESS_AND_KEEP_CONN;
}

/* FUNCTION: static BOOL IsValidTermId(int TermId)
 *
 * PURPOSE:         This function checks to see of the
passed in terminal id is valid.
 *
 * ARGUMENTS:       int
          TermId
          client terminal id
 *
 * RETURNS:                  BOOL      FALSE

          Terminal ID Invalid
 *
          TRUE
                    Terminal ID valid
 *
 * COMMENTS:        None
 *
 */

static BOOL IsValidTermId(int TermId)
{
          return (BOOL) ( TermId > 0 && TermId <=
Term.iAvailable && Term.pClientData[TermId].inUse );
}

/* FUNCTION: BOOL
ProcessQueryString(EXTENSION_CONTROL_BLOCK *pECB, int
*pCmd, int *pFormId, int *pTermId, int *pSyncId)
 *
 * PURPOSE:         This function extracts the relevent
information out of the http command passed in from
 *                              the browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
                    structure pointer to passed in
internet
 *
                              service information.
 *                                  int
                              *pCmd
                    returned command id
 *                                  int
                              *pFormId
          returned active form client browser is on
 *                                  int
                              *pTermId
          returned client terminal id
 *
 * RETURNS:                  BOOL      FALSE

          success
 *
          TRUE
                    command passed in is invalid
 *
 * COMMENTS:        If this is the initial connection
i.e. client is at welcome screen then
 *                                  there will not
be a terminal id or current form id if this is the case
 *                                  then the
pTermid and pFormid return values are undefined.
 */

BOOL ProcessQueryString(EXTENSION_CONTROL_BLOCK *pECB,
int *pCmd, int *pFormId, int *pTermId, int *pSyncId)
{
          char *ptr;
          char szBuffer[25];
          char szTmp[25];
          char *dest = szBuffer;
          int i;

          if ( (ptr = strstr(pECB->lpszQueryString,
"FORMID=")) )
                    *pFormId = *(ptr+7) & 0x0F;

          if ( (ptr = strstr(pECB->lpszQueryString,
"TERMID=")) )
          {
                    *pTermId = atoi((ptr+7));
                    if ( *pTermId == 0 )
          //terminal id 0 used internally
                              *pTermId = -1;
                    if ( *pTermId == -2 )
          //login screen
                              *pTermId = 0;
          }
          else
                    *pTermId = 0;


          if ( (ptr = strstr(pECB->lpszQueryString,
"SYNCID=")) )
                    *pSyncId = atoi((ptr+7));
          else
                    *pSyncId = 0;

          if ( !(ptr = strstr(pECB->lpszQueryString,
"CMD=")) )
          {

                    ptr = szBuffer;
                    if ( !stricmp(szBuffer, "Default")
)
                              strcpy(szBuffer,
"CMD=Begin");
                    switch( *pFormId )
                    {
                              case WELCOME_FORM:

          strcpy(szBuffer, "CMD=Submit");
                                        break;
                              case MAIN_MENU_FORM:

          strcpy(szBuffer, "CMD=NewOrder");
                                        break;
                              case NEW_ORDER_FORM:
                              case PAYMENT_FORM:
                              case DELIVERY_FORM:
                              case ORDER_STATUS_FORM:
                              case STOCK_LEVEL_FORM:
                                        if (
!(*pTermId) )

          return FALSE;
                                        if (
GetKeyValue(pECB->lpszQueryString, "PI*", szTmp,
sizeof(szTmp)) )

          strcpy(szBuffer, "CMD=Process");
                                        else
                                        {

          strcpy(szBuffer, "CMD=");

          strcat(szBuffer, szCmds[*pFormId -
NEW_ORDER_FORM]);
```

```
                                        }
                                        break;
                                default:
                                        return FALSE;
                        }
                }

                ptr += 4;

                while( *ptr && *ptr != '&' )
                        *dest++ = *ptr++;
                *dest = 0;

                for(i=0; szCmds[i][0]; i++)
                {
                        if ( !strcmp(szCmds[i], szBuffer) )
                        {
                                *pCmd = i;
                                return TRUE;
                        }
                }
                return FALSE;
}

/* FUNCTION: void NewOrderForm(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:         This function wraps the
functionality needed for the TPC-C New Order Form.
 *
 * ARGUMENTS:       int
                                iFormId
                unused
 *                                      int

                iTermId             id of calling browser,
i.e. TERMID= from http command line
 *
                EXTENSION_CONTROL_BLOCK       *pECB
                structure pointer to passed in internet
 *

                                service information.
 *
 * RETURNS:                     None
 *
 * COMMENTS:        None
 *
 */

void NewOrderForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
                WriteZString(pECB, MakeNewOrderForm(iTermId,
iSyncId, FALSE, TRUE, FALSE));

                UNUSEDPARAM(iFormId);

                return;
}

/* FUNCTION: void PaymentForm(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:         This function wraps the
functionality needed for the TPC-C Payment Form.
 *
 * ARGUMENTS:       int
                                iFormId
                unused
 *                                      int

                iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                      int

                iSyncId             sync id of calling
browser
 *
                EXTENSION_CONTROL_BLOCK       *pECB
                structure pointer to passed in internet
 *

                                service information.
 * RETURNS:                     None
 *
 * COMMENTS:        None
 *
 */

void PaymentForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
                WriteZString(pECB, MakePaymentForm(iTermId,
iSyncId, TRUE) );

                UNUSEDPARAM(iFormId);

}

/* FUNCTION: void DeliveryForm(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:         This function wraps the
functionality needed for the TPC-C Delivery Form.
 *
 * ARGUMENTS:       int
                                iFormId
                unused
 *                                      int

                iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                      int

                iSyncId             sync id of calling
browser
 *
                EXTENSION_CONTROL_BLOCK       *pECB
                structure pointer to passed in internet
 *

                                service information.
 * RETURNS:                     None
 *
 * COMMENTS:        None
 *
 */

void DeliveryForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
//              WriteZString(pECB, MakeDeliveryForm(iTermId,
iSyncId, TRUE) );
                WriteZString(pECB, MakeDeliveryForm(iTermId,
iSyncId, TRUE, TRUE) );

                UNUSEDPARAM(iFormId);
}

/* FUNCTION: void
OrderStatusForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:         This function wraps the
functionality needed for the TPC-C Order Status Form.
 *
 * ARGUMENTS:       int
                                iFormId
                unused
 *                                      int

                iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                      int

                iSyncId             sync id of calling
borwser
 *
                EXTENSION_CONTROL_BLOCK       *pECB
                structure pointer to passed in internet
 *

                                service information.
 * RETURNS:                     None
 *
 * COMMENTS:        None
 *
 */

void OrderStatusForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
                WriteZString(pECB,
MakeOrderStatusForm(iTermId, iSyncId, TRUE) );

                UNUSEDPARAM(iFormId);

}

/* FUNCTION: void
StockLevelForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:         This function wraps the
functionality needed for the TPC-C Stock Level Form.
```

```
 *
 * ARGUMENTS:       int
                                iFormId
          unused
 *                                      int

          iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                      int

          iSyncId             sync id of calling
browser
 *
          EXTENSION_CONTROL_BLOCK       *pECB
          structure pointer to passed in internet
 *

                                service information.
 * RETURNS:                   None
 *
 * COMMENTS:        None
 *
 */

void StockLevelForm(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
          WriteZString(pECB,
MakeStockLevelForm(iTermId, iSyncId, TRUE) );

          return;
}

/* FUNCTION: void Exitcmd(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:         This function removes a terminal id
from use, the allocated structure however remains
 *                               valid so the next request
for a new client will not require a new memory
allocation.
 *
 * ARGUMENTS:       int
                                iFormId
          unused
 *                                      int

          iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                      int

          iSyncId             sync id of calling
browser
 *
          EXTENSION_CONTROL_BLOCK       *pECB
          structure pointer to passed in internet
 *

                                service information.
 * RETURNS:                   None
 *
 * COMMENTS:        None
 *
 */

void Exitcmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
          (*Term.Delete)(pECB, iTermId);

          WriteZString(pECB, MakeWelcomeForm() );

          UNUSEDPARAM(iFormId);
          UNUSEDPARAM(iSyncId);

          return;
}

/* FUNCTION: void SubmitCmd(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:         This function allocated a new
terminal id in the Term structure array.
 *
 * ARGUMENTS:       int
                                iFormId
          unused
 *                                      int

          iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                      int

          iSyncId             sync id of calling
browser
 *
          EXTENSION_CONTROL_BLOCK       *pECB
          structure pointer to passed in internet
 *

                                service information.
 * RETURNS:                   None
 *
 * COMMENTS:        A terminal id can be allocated but
still be invalid if the requested warehouse number
 *                                      is outside the
range specified in the registry. This then will force
the client id
 *                                      to be invalid
and an error message sent to the users browser.
 */

void SubmitCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
          int       iCurrent;

          if ( (iCurrent = (*Term.Add)(pECB, pECB-
>lpszQueryString)) < 0 )
          {
                    ErrorMessage(pECB,
ERR_CANNOT_INIT_TERMINAL, ERR_TYPE_WEBDLL, NULL,
iCurrent, iSyncId);
                    return;
          }

          if ( Term.pClientData[iCurrent].w_id >
iMaxWareHouses || Term.pClientData[iCurrent].w_id < 1 )
          {
                    ErrorMessage(pECB,
ERR_W_ID_INVALID, ERR_TYPE_WEBDLL, NULL, iCurrent,
iSyncId);
                    (*Term.Delete)(pECB, iCurrent);
                    return;
          }
          if ( Term.pClientData[iCurrent].d_id < 1 ||
Term.pClientData[iCurrent].d_id > 10 )
          {
                    ErrorMessage(pECB,
ERR_D_ID_INVALID, ERR_TYPE_WEBDLL, NULL, iCurrent,
iSyncId);
                    (*Term.Delete)(pECB, iCurrent);
                    return;
          }

          WriteZString(pECB, MakeMainMenuForm(iCurrent,
Term.pClientData[iCurrent].iSyncId) );

          return;
}

/* FUNCTION: void BeginCmd(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:         This function is the first command
executed. It is executed with the command
 *                               CMD=Begin?Server=xxx from
the http command line.
 *
 * ARGUMENTS:       int
                                iFormId
          unused
 *                                      int

          iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                      int

          iSyncId             sync id of calling
browser
 *
          EXTENSION_CONTROL_BLOCK       *pECB
          structure pointer to passed in internet
 *

                                service information.
 * RETURNS:                   None
 *
 * COMMENTS:        SQL server must be specified,
however the user and password parameters are optional.
 *                                      The complete
command line is CMD=Begin&Server=server&User=sa&Psw=&.
The & are used
 *                                      to separate
parameters which is internet browser standard.
 */
```

```
void BeginCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
          LPSTR pQueryString;

          pQueryString = pECB->lpszQueryString;

          WriteZString(pECB, MakeWelcomeForm() );

          UNUSEDPARAM(iFormId);

          return;
}

/* FUNCTION: void ProcessCmd(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:          This function process the passed in
http command
 *
 * ARGUMENTS:       int
                              iFormId
          unused
 *                                    int

          iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                    int

          iSyncId             sync id of calling
browser
 *
          EXTENSION_CONTROL_BLOCK       *pECB
          structure pointer to passed in internet
 *

                              service information.
 * RETURNS:                  None
 *
 * COMMENTS:        None
 *
 */

void ProcessCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
          switch( iFormId )
          {
                    case WELCOME_FORM:
                              return;
                    case MAIN_MENU_FORM:
                              return;
                    case NEW_ORDER_FORM:
                              ProcessNewOrderForm(pECB,
iTermId, iSyncId);
                              return;
                    case PAYMENT_FORM:
                              ProcessPaymentForm(pECB,
iTermId, iSyncId);
                              return;
                    case DELIVERY_FORM:
                              ProcessDeliveryForm(pECB,
iTermId, iSyncId);
                              return;
                    case ORDER_STATUS_FORM:

          ProcessOrderStatusForm(pECB, iTermId,
iSyncId);
                              return;
                    case STOCK_LEVEL_FORM:

          ProcessStockLevelForm(pECB, iTermId,
iSyncId);
                              return;
          }
}

/* FUNCTION: void ClearCmd(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:          This function frees all currently
logged in terminal ids.
 *
 * ARGUMENTS:       int
                              iFormId
          unused
 *                                    int

          iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                    int

          iSyncId             sync id of calling
browser
 *
          EXTENSION_CONTROL_BLOCK       *pECB
          structure pointer to passed in internet
 *

                              service information.
 * RETURNS:                  None
 *
 * COMMENTS:        Use this function with caution, it
may cause unpredictable results
 *                                    if existing
browsers attempt to use the web client with out
 *                                    beginning at
the login screen for each client.
 */

void ClearCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
          int i;

          EnterCriticalSection(&CriticalSection);

          for(i=0; i<Term.iAvailable; i++)
          {
                    if ( Term.pClientData[i].inUse )
                              (*Term.Delete)(pECB, i);
          }

          Term.iNext                              = 0;
          Term.iAvailable               = 0;
          Term.iMasterSyncId  = 1;

          if ( Term.pClientData )
                    free(Term.pClientData);
          Term.pClientData    = NULL;
          Term.bInit                              =
FALSE;

          (*Term.Init)();
          if ( !(*Term.Allocate)() )
          {
                    ErrorMessage(pECB,
ERR_MAX_CONNECT_PARAM, ERR_TYPE_WEBDLL, NULL, iTermId,
iSyncId);
                    return;
          }
          for(i=Term.iNext; i<Term.iAvailable; i++)
                    Term.pClientData[i].inUse = 0;
          Term.pClientData[0].inUse = 1;

          LeaveCriticalSection(&CriticalSection);

          WriteZString(pECB, MakeWelcomeForm() );

          return;
}

/* FUNCTION: void MenuCmd(EXTENSION_CONTROL_BLOCK
*pECB, int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:          This function causes an exit to the
main menu
 *
 * ARGUMENTS:       int
                              iFormId
          unused
 *                                    int

          iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                    int

          iSyncId             sync id of calling
browser
 *
          EXTENSION_CONTROL_BLOCK       *pECB
          structure pointer to passed in internet
 *

                              service information.
 * RETURNS:                  None
 *
 * COMMENTS:        None
 *
 */

void MenuCmd(EXTENSION_CONTROL_BLOCK *pECB, int
iFormId, int iTermId, int iSyncId)
{
```

```
          WriteZString(pECB, MakeMainMenuForm(iTermId,
iSyncId) );

          return;
}

/* FUNCTION: void
NumberOfConnectionsCmd(EXTENSION_CONTROL_BLOCK *pECB,
int iFormId, int iTermId, int iSyncId)
 *
 * PURPOSE:         This function returns to the
browser the total number of active terminal ids
 *
 * ARGUMENTS:       int
                              iFormId
          unused
 *                                    int

          iTermId             id of calling browser,
i.e. TERMID= from http command line
 *                                    int

          iSyncId             sync id of calling
browser
 *
          EXTENSION_CONTROL_BLOCK       *pECB
          structure pointer to passed in internet
 *

                              service information.
 * RETURNS:                  None
 *
 * COMMENTS:        None
 */

/* FUNCTION: void WriteZString(EXTENSION_CONTROL_BLOCK
*pECB, char *szStr)
 *
 * PURPOSE:         This function is the low level
output function. It writes a string of text back to the
 *                            client browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                    char
                    *szStr
          string to display in the client browser.
 *
 * RETURNS:                  None
 *
 * COMMENTS:        This function assumes that the
string to written to the client browser has
 *                                    been formatted
in an HTML manner.
 */

static void WriteZString(EXTENSION_CONTROL_BLOCK *pECB,
char *szStr)
{
          FILE      *fp;
          int       lpbSize;
          int       iSize;
          char      szHeader[128];
          char      szHeader1[128];

          lpbSize = strlen(szStr)+1;

          if ( bLog )
          {
                    SYSTEMTIME          systemTime;

                    fp = fopen(szTpccLogPath, "ab");

                    GetLocalTime(&systemTime);

                    fprintf(fp, "* HTML PAGE *
%2.2d/%2.2d/%2.2d %2.2d:%2.2d:%2.2d\r\n\r\n%s\r\n\r\n",
                              systemTime.wYear,
systemTime.wMonth, systemTime.wDay,
                              systemTime.wHour,
systemTime.wMinute, systemTime.wSecond,
                              szStr);

                    fclose(fp);
          }

          iSize = sprintf(szHeader, "200 Ok");
          sprintf(szHeader1, "Connection: keep-
alive\r\nContent-type: text/html\r\nContent-length:
%d\r\n\r\n", lpbSize);


#ifdef PURE_PERFORMIX
          (*pECB->ServerSupportFunction)(pECB->ConnID,
HSE_REQ_DONE_WITH_SESSION, NULL, 0, 0);
#else
          (*pECB->ServerSupportFunction)(pECB->ConnID,
HSE_REQ_SEND_RESPONSE_HEADER, szHeader, &iSize,
(LPDWORD)szHeader1);
#endif

          (*pECB->WriteClient)(pECB->ConnID, szStr,
&lpbSize, 0);

          return;
}

/* FUNCTION: void h_printf(EXTENSION_CONTROL_BLOCK
*pECB, char *format, ...)
 *
 * PURPOSE:         This function forms a high level
printf for an HTML browser
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                    char
                    *format
          printf style format string
 *                                    ...

                    other arguments as required by
printf style format string.
 *
 * RETURNS:                  None
 *
 * COMMENTS:        This function is mainly used for
developmental support.
 */

static void h_printf(EXTENSION_CONTROL_BLOCK *pECB,
char *format, ...)
{
          int lpbSize;
          char szBuff[512];
          char szTmp[512];

          va_list marker;
          va_start( marker, format );
          vsprintf(szTmp, format, marker);
          va_end( marker );

          lpbSize = wsprintf(szBuff, "<html>%s</html>",
szTmp) + 1;

          (*pECB->WriteClient)(pECB->ConnID, szBuff,
&lpbSize, 0);


          return;
}

void LogTuxError( int TpErrno, char *ErrMessage )
{
          FILE *fp;
          SYSTEMTIME          systemTime;

          GetLocalTime(&systemTime);

          fp = fopen(szErrorLogPath, "ab");
          fprintf(fp, "\r\nError: %2.2d/%2.2d/%2.2d
%2.2d:%2.2d:%2.2d\r\n",
                    systemTime.wYear,
systemTime.wMonth, systemTime.wDay,
                    systemTime.wHour,
systemTime.wMinute, systemTime.wSecond);
          fprintf(fp, "Thread %d: TPCCWEB(%d): %s: %s",
                    GetCurrentThreadId(), TpErrno,
tpstrerror(TpErrno), ErrMessage);
          fclose(fp);
}

/* FUNCTION: void ErrorMessage(EXTENSION_CONTROL_BLOCK
*pECB, int iError, int iErrorType, char *szMsg)
 *
 * PURPOSE:         This function displays an error
message in the client browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                    int
                              iError
          id of error message
 *                                    int
                              iErrorType
          error type, ERR_TYPE_SQL, ERR_TYPE_DBLIB, or
ERR_TYPE_WEBDLL
```

```
 *                                    int
                            iTermId
          terminal id from browser
 *                                    int
                            iSyncid
          sync id from browser
 *                                    char *
                    szMsg
          optional error message string used with
ERR_TYPE_SQL and
 *
                    ERR_TYPE_DBLIB
 *
 * RETURNS:                  None
 *
 * COMMENTS:        If the error type is
ERR_TYPE_WEBDLL the szmsg parameter may be NULL because
it
 *                                    is ignored. If
the error type is ERR_TYPE_SQL or ERR_TYPE_DBLIB then
the szMsg
 *                                    parameter
contains the text of the error message, so the szMsg
parameter cannot
 *                                    be NULL.
 *
 */


void ErrorMessage(EXTENSION_CONTROL_BLOCK *pECB, int
iError, int iErrorType, char *szMsg, int iTermId, int
iSyncId)
{
          int i;
          SYSTEMTIME          systemTime;

          static SERRORMSG errorMsgs[] =
          {
                    {         ERR_SUCCESS,

          "Success, no error."

                                                  },
                    {         ERR_COMMAND_UNDEFINED,

          "Command undefined."

                                                  },
                    {         ERR_NOT_IMPLEMENTED_YET,
                                        "Not
Implemented Yet."

                                        },
                    {         ERR_CANNOT_INIT_TERMINAL,
                                        "Cannot
initialize client connection."

          },
                    {         ERR_OUT_OF_MEMORY,

          "insufficient memory."

                                                  },
                    {
          ERR_NEW_ORDER_NOT_PROCESSED,
          "Cannot process new Order form."

                                        },
                    {
          ERR_PAYMENT_NOT_PROCESSED,
                    "Cannot process payment form."

                                                  },
                    {         ERR_NO_SERVER_SPECIFIED,
                                        "No Server name
specified."

                    },
                    {
          ERR_ORDER_STATUS_NOT_PROCESSED,
                    "Cannot process order status form."

                                        },
                    {         ERR_W_ID_INVALID,

          "Invalid Warehouse ID."

                                                  },
                    {
          ERR_CAN_NOT_SET_MAX_CONNECTIONS,
          "Insufficient memory to allocate #
connections."
          },
                    {         ERR_NOSUCH_CUSTOMER,
                                                  "No
such customer."

                                        },
                    {         ERR_D_ID_INVALID,

          "Invalid District ID Must be 1 to 10."

                    },
                    {         ERR_MAX_CONNECT_PARAM,
                                                  "Max
client connections exceeded, run install to increase."
                    },
                    {
          ERR_INVALID_SYNC_CONNECTION,
          "Invalid Terminal Sync ID."

                                        },
                    {         ERR_INVALID_TERMID,

          "Invalid Terminal ID."

                                                  },
                    {
          ERR_PAYMENT_INVALID_CUSTOMER,
          "Payment Form, No such Customer."

                    },
                    {         ERR_SQL_OPEN_CONNECTION,

          "SQLOpenConnection API Failed."

                                        },
                    {
          ERR_STOCKLEVEL_MISSING_THRESHOLD_KEY,
          "Stock Level missing Threshold key \"TT*\"."

          },
                    {
          ERR_STOCKLEVEL_THRESHOLD_INVALID,
          "Stock Level Threshold invalid data type
range = 1 - 99."                    },
                    {
          ERR_STOCKLEVEL_THRESHOLD_RANGE,
                    "Stock Level Threshold out of
range, range must be 1 - 99."                       },
                    {
          ERR_STOCKLEVEL_NOT_PROCESSED,
          "Stock Level not processed."

                    },
                    {
          ERR_NEWORDER_FORM_MISSING_DID,
                    "New Order missing District key
\"DID*\"."
                    },
                    {
          ERR_NEWORDER_DISTRICT_INVALID,
                    "New Order District ID Invalid
range 1 - 10."
                    },
                    {
          ERR_NEWORDER_DISTRICT_RANGE,
          "New Order District ID out of Range. Range =
1 - 10."                    },
                    {
          ERR_NEWORDER_CUSTOMER_KEY,
                    "New Order missing Customer key
\"CID*\"."
                    },
                    {
          ERR_NEWORDER_CUSTOMER_INVALID,
                    "New Order customer id invalid data
type, range = 1 to 3000."               },
                    {
          ERR_NEWORDER_CUSTOMER_RANGE,
          "New Order customer id out of range, range =
1 to 3000."                    },
                    {
          ERR_NEWORDER_MISSING_IID_KEY,
          "New Order missing Item Id key \"IID*\"."

                    },
                    {
          ERR_NEWORDER_ITEM_BLANK_LINES,
                    "New Order blank order lines all
orders must be continuous."             },
                    {
          ERR_NEWORDER_ITEMID_INVALID,
```

```
          "New Order Item Id is wrong data type, must
be numeric."                              },
                    {
          ERR_NEWORDER_MISSING_SUPPW_KEY,
                    "New Order missing Supp_W key
\"SP##*\"."
                              },
                    {
          ERR_NEWORDER_SUPPW_INVALID,
                    "New Order Supp_W invalid data type
must be numeric."                                 },
                    {
          ERR_NEWORDER_MISSING_QTY_KEY,
          "New Order Missing Qty key \"Qty##*\"."

                    },
                    {         ERR_NEWORDER_QTY_INVALID,
                                        "New Order Qty
invalid must be numeric range 1 - 99."
                    },
                    {         ERR_NEWORDER_SUPPW_RANGE,
                                        "New Order
Supp_W value out of range range = 1 - Max Warehouses."
          },
                    {
          ERR_NEWORDER_ITEMID_RANGE,
                    "New Order Item Id is out of range.
Range = 1 to 999999."                             },
                    {         ERR_NEWORDER_QTY_RANGE,
                                        "New
Order Qty is out of range. Range = 1 to 99."
                                        },
                    {
          ERR_PAYMENT_DISTRICT_INVALID,
          "Payment District ID is invalid must be 1 -
10."                                              },
                    {
          ERR_NEWORDER_SUPPW_WITHOUT_ITEMID,
          "New Order Supp_W field entered without a
corrisponding Item_Id."       },
                    {
          ERR_NEWORDER_QTY_WITHOUT_ITEMID,
          "New Order Qty entered without a
corrisponding Item_Id."                           },
                    {
          ERR_NEWORDER_NOITEMS_ENTERED,
          "New Order Blank Items between items, items
must be continuous."          },
                    {
          ERR_PAYMENT_MISSING_DID_KEY,
          "Payment missing District Key \"DID*\"."

                    },
                    {
          ERR_PAYMENT_DISTRICT_RANGE,
                    "Payment District Out of range,
range = 1 - 10."
          },
                    {
          ERR_PAYMENT_MISSING_CID_KEY,
          "Payment missing Customer Key \"CID*\"."

                    },
                    {
          ERR_PAYMENT_CUSTOMER_INVALID,
          "Payment Customer data type invalid, must be
numeric."                                 },
                    {         ERR_PAYMENT_MISSING_CLT,
                                        "Payment
missing Customer Last Name Key \"CLT*\"."
                                        },
                    {
          ERR_PAYMENT_LAST_NAME_TO_LONG,
                    "Payment Customer last name longer
than 16 characters."
          },
                    {
          ERR_PAYMENT_CUSTOMER_RANGE,
                    "Payment Customer ID out of range,
must be 1 to 3000."                               },
                    {         ERR_PAYMENT_CID_AND_CLT,
                                        "Payment
Customer ID and Last Name entered must be one or
other."   },
                    {
          ERR_PAYMENT_MISSING_CDI_KEY,
          "Payment missing Customer district key
\"CDI*\"."
          },
                    {         ERR_PAYMENT_CDI_INVALID,
                                        "Payment
Customer district invalid must be numeric."
                                        },
                    {         ERR_PAYMENT_CDI_RANGE,

          "Payment Customer district out of range must
be 1 - 10."                               },
                    {
          ERR_PAYMENT_MISSING_CWI_KEY,
          "Payment missing Customer Warehouse key
\"CWI*\"."
          },
                    {         ERR_PAYMENT_CWI_INVALID,
                                        "Payment
Customer Warehouse invalid must be numeric."
                                        },
                    {         ERR_PAYMENT_CWI_RANGE,

          "Payment Customer Warehouse out of range, 1
to Max Warehouses."           },
                    {
          ERR_PAYMENT_MISSING_HAM_KEY,
          "Payment missing Amount key \"HAM*\"."

                    },
                    {         ERR_PAYMENT_HAM_INVALID,
                                        "Payment Amount
invalid data type must be numeric."
                                        },
                    {         ERR_PAYMENT_HAM_RANGE,

          "Payment Amount out of range, 0 - 9999.99."

                    },
                    {
          ERR_ORDERSTATUS_MISSING_DID_KEY,
          "Order Status missing District key \"DID*\"."

          },
                    {
          ERR_ORDERSTATUS_DID_INVALID,
          "Order Status District invalid, value must be
numeric 1 - 10."              },
                    {
          ERR_ORDERSTATUS_DID_RANGE,
                    "Order Status District out of range
must be 1 - 10."                                  },
                    {
          ERR_ORDERSTATUS_MISSING_CID_KEY,
          "Order Status missing Customer key \"CID*\"."

          },
                    {
          ERR_ORDERSTATUS_MISSING_CLT_KEY,
          "Order Status missing Customer Last Name key
\"CLT*\"."                                        },
                    {
          ERR_ORDERSTATUS_CLT_RANGE,
                    "Order Status Customer last name
longer than 16 characters."             },
                    {
          ERR_ORDERSTATUS_CID_INVALID,
          "Order Status Customer ID invalid, range must
be numeric 1 - 3000."         },
                    {
          ERR_ORDERSTATUS_CID_RANGE,
                    "Order Status Customer ID out of
range must be 1 - 3000."                          },
                    {
          ERR_ORDERSTATUS_CID_AND_CLT,
          "Order Status Customer ID and LastName
entered must be only one."    },
                    {
          ERR_DELIVERY_MISSING_OCD_KEY,
          "Delivery missing Carrier ID key \"OCD*\"."

                    },
                    {
          ERR_DELIVERY_CARRIER_INVALID,
          "Delivery Carrier ID invalid must be numeric
1 - 10."                                  },
                    {
          ERR_DELIVERY_CARRIER_ID_RANGE,
                    "Delivery Carrier ID out of range
must be 1 - 10."
          },
                    {
          ERR_PAYMENT_MISSING_CLT_KEY,
          "Payment missing Customer Last Name key
\"CLT*\"."
          },
                    {         0,

                              ""
```

```
                                                  }
        };

        static char szNoMsg[] = "";
        char                *szForm;

        GetLocalTime(&systemTime);

        if ( !szMsg  )
                szMsg = szNoMsg;

        if ( iTermId > 0 && IsValidTermId(iTermId) )
                szForm =
Term.pClientData[iTermId].szBuffer; //if termid valid
use common terminal static buffer.
        else
                szForm =
Term.pClientData[0].szBuffer; //else term id invalid so
use common terminal static buffer.
        switch(iErrorType)
        {
                case ERR_TYPE_WEBDLL:
                        for(i=0;
errorMsgs[i].szMsg[0]; i++)
                        {
                                if ( iError ==
errorMsgs[i].iError )

        break;
                        }
                        if (
!errorMsgs[i].szMsg[0] )
                                i = 1;
                        strcpy(szForm,
        "<HTML><HEAD><TITLE>Welcome To TPC-
C</TITLE></HEAD><BODY><FORM ACTION=\"tpcc.dll\"
METHOD=\"GET\">");

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">",
iErrorType);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"%d\">", iError);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);

        wsprintf(szForm+strlen(szForm), "Error:
TPCCWEB(%d): %s", iError, errorMsgs[i].szMsg);
                        strcat(szForm,
"</FORM><BODY></HTML>");
                        WriteZString(pECB,
szForm);
                        break;
                case ERR_TYPE_SQL:
                        strcpy(szForm,
        "<HTML><HEAD><TITLE>Welcome To TPC-
C</TITLE></HEAD><BODY><FORM ACTION=\"tpcc.dll\"
METHOD=\"GET\">");

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">",
iErrorType);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"%d\">", iError);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);

        wsprintf(szForm+strlen(szForm), "Error:
SQLSVR(%d): %s", iError, szMsg);
                        strcat(szForm,
"</FORM><BODY></HTML>");
                        WriteZString(pECB,
szForm);
                        break;
                case ERR_TYPE_DBLIB:
                        strcpy(szForm,
        "<HTML><HEAD><TITLE>Welcome To TPC-
C</TITLE></HEAD><BODY><FORM ACTION=\"tpcc.dll\"
METHOD=\"GET\">");

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">",
iErrorType);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"%d\">", iError);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);

        wsprintf(szForm+strlen(szForm), "Error:
DBLIB(%d): %s", iError, szMsg);
                        strcat(szForm,
"</FORM><BODY></HTML>");
                        WriteZString(pECB,
szForm);
                        break;
                case ERR_TYPE_ODBC:
                        strcpy(szForm,
        "<HTML><HEAD><TITLE>Welcome To TPC-
C</TITLE></HEAD><BODY><FORM ACTION=\"tpcc.dll\"
METHOD=\"GET\">");

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">",
iErrorType);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"%d\">", iError);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);

        wsprintf(szForm+strlen(szForm), "Error:
ODBC");
                        strcat(szForm,
"</FORM><BODY></HTML>");
                        WriteZString(pECB,
szForm);
                        break;
                case ERR_TYPE_SOCKET:
                        strcpy(szForm,
        "<HTML><HEAD><TITLE>Welcome To TPC-
C</TITLE></HEAD><BODY><FORM ACTION=\"tpcc.dll\"
METHOD=\"GET\">");

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">",
iErrorType);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"%d\">", iError);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);

        wsprintf(szForm+strlen(szForm), "Error:
SOCKET");
                        strcat(szForm,
"</FORM><BODY></HTML>");
                        WriteZString(pECB,
szForm);
                        break;
                case ERR_TYPE_DEADLOCK:
                        strcpy(szForm,
        "<HTML><HEAD><TITLE>Welcome To TPC-
C</TITLE></HEAD><BODY><FORM ACTION=\"tpcc.dll\"
METHOD=\"GET\">");

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">",
iErrorType);

        wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"%d\">", iError);
```

```
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);

          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);

          wsprintf(szForm+strlen(szForm), "Error:
Deadlock");
                              strcat(szForm,
"</FORM><BODY></HTML>");
                              WriteZString(pECB,
szForm);
                              break;
          }
          return;
}

/* FUNCTION: BOOL GetKeyValue(char *pQueryString, char
*pKey, char *pValue, int iMax)
 *
 * PURPOSE:         This function parses a http
formatted string for specific key values.
 *
 * ARGUMENTS:       char
          *pQueryString       http string from client
browser
 *                                    char
                    *pKey                         key
value to look for
 *                                    char
                    *pValue
          character array into which to place key's
value
 *                                    int
                              iMax
          maximum length of key value array.
 *
 * RETURNS:                   BOOL      FALSE     key
value not found
 *
          TRUE      key valud found
 *
 *
 * COMMENTS:        http keys are formatted either
KEY=value& or KEY=value\0. This DLL formats
 *                                    TPC-C input
fields in such a manner that the keys can be extracted
in the
 *                                    above manner.
 */

static BOOL GetKeyValue(char *pQueryString, char *pKey,
char *pValue, int iMax)
{
          char *ptr;


          if ( !(ptr=strstr(pQueryString, pKey)) )
                    return FALSE;
          if ( !(ptr=strchr(ptr, '=')) )
                    return FALSE;
          ptr++;
          iMax--;
          while( *ptr && *ptr != '&' && iMax)
          {
                    *pValue++ = *ptr++;
                    iMax--;
          }
          *pValue = 0;
          return TRUE;
}

/* FUNCTION: void TermInit(void)
 *
 * PURPOSE:         This function initializes the
client ternimal structure it is called when the
TPCC.DLL
 *                              is first loaded by the
inet service.
 *
 * ARGUMENTS:       none
 *
 * RETURNS:                   None
 *
 * COMMENTS:        None
 *
 */

static void TermInit(void)
{
          if ( Term.bInit )
                    return;
          Term.iNext                              = 0;
          Term.iMasterSyncId  = 1;

          Term.iAvailable               = 0;
          Term.pClientData    = NULL;
          Term.bInit                              =
TRUE;

          return;
}

/* FUNCTION: void TermRestore(void)
 *
 * PURPOSE:         This function frees allocated
resources associated with the terminal structure.
 *
 * ARGUMENTS:       none
 *
 * RETURNS:                   None
 *
 * COMMENTS:        This function is called only with
the inet service unloads the TPCC.DLL
 *
 */

static void TermRestore(void)
{
          Term.iNext                              = 0;
          Term.iAvailable               = 0;
          Term.iMasterSyncId  = 0;
          if ( Term.pClientData )
                    free(Term.pClientData);
          Term.pClientData    = NULL;
          Term.bInit                              =
FALSE;


          return;
}

/* FUNCTION: int TermAllocate(void)
 *
 * PURPOSE:         This function allocates more
terminal array entries in the Term structure.
 *
 * ARGUMENTS:   None
 *
 * RETURNS:                   int       TRUE or 1 if
sucessfull
 *                                    int       FALSE
or 0 if terminal id cannot be allocated.
 *
 * COMMENTS:        None
 *
 */

static int TermAllocate(void)
{
          Term.iAvailable += 32;
          if ( !Term.pClientData )
                    Term.pClientData =
(PCLIENTDATA)malloc(Term.iAvailable *
sizeof(CLIENTDATA));
          else
                    Term.pClientData =
(PCLIENTDATA)realloc(Term.pClientData, Term.iAvailable
* sizeof(CLIENTDATA));
          return ( Term.pClientData ) ? 1 : 0;
}

/* FUNCTION: int TermAdd(EXTENSION_CONTROL_BLOCK *pECB,
char *pQueryString)
 *
 * PURPOSE:         This function assigns a terminal id
which is used to identify a client browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
                              passed in structure
pointer from inetsrv.
 *                                  char
                                    *pQueryString
          http query string passed to this DLL.
 *
 * RETURNS:                   int
          assigned terminal id
 *                                                  -1
          cannot assign id error occured.
 *
 *
 * COMMENTS:        if the terminal id cannot be
assigned it is because of insufficient memory or the
```

```
 *                                      SQL connection
cannot be allocated.
 *
 */

static int TermAdd(EXTENSION_CONTROL_BLOCK *pECB, char
*pQueryString)
{
          char      szTmp[32];
          int                 i, iCurrent,
iTotalConnections, iTickCount;

          EnterCriticalSection(&CriticalSection);

          for(i=0, iTotalConnections = 0;
i<Term.iAvailable; i++)
          {
                    if ( Term.pClientData[i].inUse )
                              iTotalConnections++;
          }

          if ( iTotalConnections >= iMaxConnections )
          {
                    for(iCurrent = 1, i=1, iTickCount =
0x7FFFFFFF; i<iMaxConnections; i++)
                    {
                              if ( iTickCount >
Term.pClientData[i].iTickCount )
                              {
                                        iTickCount =
Term.pClientData[i].iTickCount;
                                        iCurrent = i;
                              }
                    }
          }
          else
          {
                    for(i=0; i<Term.iAvailable; i++)
                    {
                              if (
!Term.pClientData[i].inUse )
                                        break;
                    }
                    iCurrent = i;
          }

          if ( i == Term.iAvailable )
          {
                    Term.iNext = Term.iAvailable;
                    if ( !(*Term.Allocate)() )
                              goto TermAddErr1;
                    for(i=Term.iNext;
i<Term.iAvailable; i++)
                              Term.pClientData[i].inUse
= 0;
                    iCurrent = Term.iNext;
          }

          Term.pClientData[iCurrent].inUse = 1;

          if ( !GetKeyValue(pQueryString, "w_id",
szTmp, sizeof(szTmp)) )
                    goto TermAddErr1;

          Term.pClientData[iCurrent].w_id =
(short)atoi(szTmp);

          if ( !GetKeyValue(pQueryString, "d_id",
szTmp, sizeof(szTmp)) )
                    goto TermAddErr1;

          Term.pClientData[iCurrent].d_id =
atoi(szTmp);

          Term.pClientData[iCurrent].iTickCount =
GetTickCount();
          Term.pClientData[iCurrent].iSyncId =
Term.iMasterSyncId++;

          if ( Init(pECB, iCurrent,
Term.pClientData[iCurrent].iSyncId, szServer, szUser,
szPassword, szDatabase) )
          {
                    (*Term.Delete)(pECB, iCurrent);
                    goto TermAddErr1;
          }

          LeaveCriticalSection(&CriticalSection);
          return iCurrent;

TermAddErr1:
          LeaveCriticalSection(&CriticalSection);
          return -1;          //terminal unsuccessfully
added
}

/* FUNCTION: void TermDelete(EXTENSION_CONTROL_BLOCK
*pECB, int id)
 *
 * PURPOSE:          This function makes a terminal
entry in the Term array available for reuse.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                      int
                                                  id
                    Terminal id of client exiting
 *
 * RETURNS:                   None
 *
 * COMMENTS:         None
 *
 */

static void TermDelete(EXTENSION_CONTROL_BLOCK *pECB,
int id)
{
          if ( id >= 0 && id < Term.iAvailable )
          {
                    Close(pECB, id, -1);
                    Term.pClientData[id].inUse = 0;

#ifndef LOCAL_ALLOC
                    tpfree((char
*)Term.pClientData[id].NewOrderDataPtr);
                    tpfree((char
*)Term.pClientData[id].PaymentDataPtr);
                    tpfree((char
*)Term.pClientData[id].OrderStatusDataPtr);
                    tpfree((char
*)Term.pClientData[id].DeliveryDataPtr);
                    tpfree((char
*)Term.pClientData[id].StockLevelDataPtr);
#endif  // Not LOCAL_ALLOC

          }

          return;
}

/* FUNCTION: BOOL Init(EXTENSION_CONTROL_BLOCK *pECB,
int iTermId, int iSyncId, char *szServer, char *szUser,
char *szPassword, char *szDatabase)
 *
 * PURPOSE:          This function initializes the sql
connection for use.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK       *pECB
                    passed in structure pointer from
inetsrv.
 *                                      int

          iTermId             id of browser client that
this connection is for.
 *                                      int

          iSyncId             sync id for this client
session
 *                                      char
                                        *szServer sql
server name
 *                                      char
                                        *szUser
          user name
 *                                      char
                                        *szPassword
          user password
 *                                      char
                                        *szDatabase
          database to use
 *
 * RETURNS:                   BOOL      FALSE     if
successfull
 *
          TRUE      if an error occurs and connection
cannot be established.
 *
 * COMMENTS:         None
 *
 */

BOOL Init(EXTENSION_CONTROL_BLOCK *pECB, int iTermId,
int iSyncId, char *szServer, char *szUser, char
*szPassword, char *szDatabase)
{
```

```
          char      szApp[32];
#ifndef LOCAL_ALLOC
          char      buf[64];
          int               TpRc;
#endif // Not LOCAL_ALLOC

          sprintf(szApp, "TPCC:%ld", (int)iTermId);

          Term.pClientData[iTermId].dbproc = NULL;

#ifndef LOCAL_ALLOC // Globally allocate tuxedo
structures

                    if ( dLog )
                    {
                              FILE *fp;
                              fp = fopen(szTpccLogPath,
"ab");
                              fprintf(fp,
"Sizeof(TUX_DATA) %d \r\n", sizeof(TUX_DATA));
                              fprintf(fp,
"Sizeof(NEW_ORDER_DATA) %d \r\n",
sizeof(NEW_ORDER_DATA));
                              fprintf(fp,
"Sizeof(PAYMENT_DATA) %d \r\n", sizeof(PAYMENT_DATA));
                              fprintf(fp,
"Sizeof(ORDER_STATUS_DATA) %d \r\n",
sizeof(ORDER_STATUS_DATA));
                              fprintf(fp,
"Sizeof(DELIVERY_DATA) %d \r\n",
sizeof(DELIVERY_DATA));
                              fprintf(fp,
"Sizeof(STOCK_LEVEL_DATA) %d \r\n",
sizeof(STOCK_LEVEL_DATA));
                              fclose(fp);
                    }

//        Add initialization of Tuxedo Structures
// NEW ORDER
                    if
((Term.pClientData[iTermId].NewOrderDataPtr =
(NEW_ORDER_DATA *)tpalloc("CARRAY", NULL,
sizeof(NEW_ORDER_DATA))) == NULL)
                    {
                              TpRc = tperrno;
                              sprintf(buf, "Tuxedo
tpalloc failed:");
                              LogTuxError(TpRc, buf);
                              ErrorMessage(pECB,
ERR_CANNOT_INIT_TERMINAL, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                              return TRUE;
                    }

                    if ( dLog )
                    {
                              FILE *fp;
                              fp = fopen(szTpccLogPath,
"ab");
                              fprintf(fp, "Thread %d
iTermId %d *  NewOrderDataPtr: %x \r\n",
GetCurrentThreadId(), iTermId,
&Term.pClientData[iTermId].NewOrderDataPtr);
                              fclose(fp);
                    }

//PAYMENT
                    if
((Term.pClientData[iTermId].PaymentDataPtr =
(PAYMENT_DATA *)tpalloc("CARRAY", NULL,
sizeof(PAYMENT_DATA))) == NULL)
                    {
                              TpRc = tperrno;
                              sprintf(buf, "Tuxedo
tpalloc failed:");
                              LogTuxError(TpRc, buf);
                              ErrorMessage(pECB,
ERR_CANNOT_INIT_TERMINAL, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                              return TRUE;
                    }

                    if ( dLog )
                    {
                              FILE *fp;

                              fp = fopen(szTpccLogPath,
"ab");

                              fprintf(fp, "Thread %d
iTermId %d *  PaymentDataPtr: %x \r\n",
          GetCurrentThreadId(), iTermId,
&Term.pClientData[iTermId].PaymentDataPtr);
                              fclose(fp);
                    }

//ORDER STATUS
                    if
((Term.pClientData[iTermId].OrderStatusDataPtr =
(ORDER_STATUS_DATA *)tpalloc("CARRAY", NULL,
sizeof(ORDER_STATUS_DATA))) == NULL)
                    {
                              TpRc = tperrno;
                              sprintf(buf, "Tuxedo
tpalloc failed:");
                              LogTuxError(TpRc, buf);
                              ErrorMessage(pECB,
ERR_CANNOT_INIT_TERMINAL, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                              return TRUE;
                    }

                    if ( dLog )
                    {
                              FILE *fp;

                              fp = fopen(szTpccLogPath,
"ab");

                              fprintf(fp, "Thread %d
iTermId %d *  OrderStatusDataPtr: %x \r\n",

          GetCurrentThreadId(), iTermId,
&Term.pClientData[iTermId].OrderStatusDataPtr);
                              fclose(fp);
                    }

//DELIVERY
                    if
((Term.pClientData[iTermId].DeliveryDataPtr =
(DELIVERY_DATA *)tpalloc("CARRAY", NULL,
sizeof(DELIVERY_DATA))) == NULL)
                    {
                              TpRc = tperrno;
                              sprintf(buf, "Tuxedo
tpalloc failed:");
                              LogTuxError(TpRc, buf);
                              ErrorMessage(pECB,
ERR_CANNOT_INIT_TERMINAL, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                              return TRUE;
                    }

                    if ( dLog )
                    {
                              FILE *fp;

                              fp = fopen(szTpccLogPath,
"ab");

                              fprintf(fp, "Thread %d
iTermId %d *  DeliveryDataPtr: %x \r\n",

          GetCurrentThreadId(), iTermId,
&Term.pClientData[iTermId].DeliveryDataPtr);
                              fclose(fp);
                    }

//STOCK LEVEL
                    if
((Term.pClientData[iTermId].StockLevelDataPtr =
(STOCK_LEVEL_DATA *)tpalloc("CARRAY", NULL,
sizeof(STOCK_LEVEL_DATA))) == NULL)
                    {
                              TpRc = tperrno;
                              sprintf(buf, "Tuxedo
tpalloc failed:");
                              LogTuxError(TpRc, buf);
                              ErrorMessage(pECB,
ERR_CANNOT_INIT_TERMINAL, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                              return TRUE;
                    }

                    if ( dLog )
                    {
                              FILE *fp;

                              fp = fopen(szTpccLogPath,
"ab");

                              fprintf(fp, "Thread %d
iTermId %d *  StockLevelDataPtr: %x \r\n",
```

```
          GetCurrentThreadId(), iTermId,
&Term.pClientData[iTermId].StockLevelDataPtr);
                              fclose(fp);
                    }
#endif // LOCAL_ALLOC

          return FALSE;
}

/* FUNCTION: BOOL Close(EXTENSION_CONTROL_BLOCK
          *pECB, int iTermId, int iSyncId)
 *
 * PURPOSE:          This function closes the sql
connection for use.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                        int

          iTermId   id of browser client that this
connection is for.
 *                                        int

          iSyncId   sync id of client browser
 *
 * RETURNS:                    BOOL      FALSE     if
successfull
 *
          TRUE      if an error occurs and connection
cannot be terminated.
 *
 * COMMENTS:         None
 *
 */

static BOOL Close(EXTENSION_CONTROL_BLOCK *pECB, int
iTermId, int iSyncId)
{
          PECBINFO  pEcbInfo;

          if (Term.pClientData[iTermId].dbproc != NULL)
          {
                    if ( (pEcbInfo =
(PECBINFO)dbgetuserdata(Term.pClientData[iTermId].dbpro
c)) )
                    {
                              pEcbInfo->iTermId = -1;
                              pEcbInfo->iSyncId = -1;
                              free(pEcbInfo);
          //free up user info
                    }
//                  return SQLCloseConnection(pECB,
Term.pClientData[iTermId].dbproc);
                    return 1;
          }

          UNUSEDPARAM(iSyncId);
}


/* FUNCTION: void FormatString(char *szDest, char
*szPic, char *szSrc)
 *
 * PURPOSE:          This function formats a character
string for inclusion in the
 *                              HTML formatted page being
constructed.
 *
 * ARGUMENTS:        char                *szDest
          Destination buffer where formatted string is
to be placed
 *                                        char
          *szPic    picture string which describes how
character value is to be
 *
                                        formatted.
 *                                      char
          *szSrc    character string value.
 *
 * RETURNS:                    None
 *
 * COMMENTS:         This functions is used to format
TPC-C phone and zip value strings.
 *
 */

static void FormatString(char *szDest, char *szPic,
char *szSrc)
{
          while( *szPic )
          {
                    if ( *szPic == 'X' )
                    {
                              if ( *szSrc )
                                        *szDest++ =
*szSrc++;
                              else
                                        *szDest++ = '
';
                    }
                    else
                              *szDest++ = *szPic;
                    szPic++;
          }
          *szDest = 0;

          return;
}

/* FUNCTION: char *MakeStockLevelForm(int iTermId, int
iSyncId, BOOL bInput)
 *
 * PURPOSE:          This function constructs the Stock
Level HTML page.
 *
 * ARGUMENTS:        int
          iTermId   client browser terminal id
 *                                      int
                    iSyncId   client browser
sync id
 *                                      BOOL
                    bInput    TRUE if form is being
constructed for input else FALSE
 *
 * RETURNS:                    char *
          A pointer to buffer inside client structure
where HTML form is built.
 *
 * COMMENTS:         The internal client buffer is
created when the terminal id is assigned and should not
 *                                      be freed except
when the client terminal id is no longer needed.
 */

static char *MakeStockLevelForm(int iTermId, int
iSyncId, BOOL bInput)
{
          char      *szForm;

          szForm = (char
*)Term.pClientData[iTermId].szBuffer;

          Term.pClientData[iTermId].StockLevelData.w_id
                                        =
(short)Term.pClientData[iTermId].w_id;
          Term.pClientData[iTermId].StockLevelData.d_id
                                        =
(short)Term.pClientData[iTermId].d_id;
          Term.pClientData[iTermId].StockLevelData.num_
deadlocks = 0;

          strcpy(szForm, "<HTML><HEAD><TITLE>TPC-C
Stock Level</TITLE></HEAD>");
          strcat(szForm, "<FORM ACTION=\"tpcc.dll\"
METHOD=\"GET\">");
          if ( bInput )
                    strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">");
          strcat(szForm, "<INPUT TYPE=\"hidden\"
NAME=\"STATUSID\" VALUE=\"0\">");
          strcat(szForm, "<INPUT TYPE=\"hidden\"
NAME=\"ERROR\" VALUE=\"0\">");
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">",
STOCK_LEVEL_FORM);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);
          strcat(szForm, "<PRE>
Stock-Level<BR>");
          wsprintf(szForm+strlen(szForm), "Warehouse:
%4.4d   District: %2.2d<BR><BR>",
Term.pClientData[iTermId].StockLevelData.w_id,
Term.pClientData[iTermId].StockLevelData.d_id);
          if ( bInput )
          {
                    strcat(szForm,      "Stock Level
Threshold: <INPUT NAME=\"TT*\" SIZE=2><BR><BR>"

          "low stock:    <BR><HR>"
```

```
          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Menu\">" );
          }
          else
          {
                    wsprintf(szForm+strlen(szForm),
"Stock Level Threshold: %2.2d<BR><BR>",
Term.pClientData[iTermId].StockLevelData.thresh_hold);

                    wsprintf(szForm+strlen(szForm),
"low stock: %3.3d</PRE><BR><HR>",
Term.pClientData[iTermId].StockLevelData.low_stock);
                    strcat(szForm,      "<INPUT
TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">" );
          }

          strcat(szForm, "</FORM></HTML>");

          return szForm;
}

/* FUNCTION: char *MakeMainMenuForm(int iTermId, int
iSyncId)
 *
 * PURPOSE:         This function
 *
 * ARGUMENTS:       int
          iTermId   client browser terminal id
 *                                    int
                              iSyncId   client browser
sync id
 *
 * RETURNS:                 char *
          A pointer to buffer inside client structure
where HTML form is built.
 *
 * COMMENTS:        The internal client buffer is
created when the terminal id is assigned and should not
 *                                    be freed except
when the client terminal id is no longer needed.
 */

static char *MakeMainMenuForm(int iTermId, int iSyncId)
{
          char      *szForm;

          szForm = (char
*)Term.pClientData[iTermId].szBuffer;

          strcpy(szForm,      "<HTML><HEAD><TITLE>TPC-C
Main Menu</TITLE></HEAD><BODY>"

          "Select Desired Transaction.<BR><HR>"

          "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">");
          strcat(szForm, "<INPUT TYPE=\"hidden\"
NAME=\"STATUSID\" VALUE=\"0\">");
          strcat(szForm, "<INPUT TYPE=\"hidden\"
NAME=\"ERROR\" VALUE=\"0\">");
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">",
MAIN_MENU_FORM);
          strcat(szForm,      "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..NewOrder..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">"

          "</FORM>"

          "</HTML>" );

          return szForm;
}

/* FUNCTION: char *MakeWelcomeForm(void)
 *
 * PURPOSE:         This function
 *
 * ARGUMENTS:   None
 *
 * RETURNS:                 char *
          A pointer to the static HTML welcome form.
 *
 * COMMENTS:        The welcome form is static.
 */

static char *MakeWelcomeForm(void)
{
          return szWelcomeForm;
}

/* FUNCTION: char *MakeNewOrderForm(int iTermId, int
SyncId, BOOL Rollback, BOOL bInput, BOOL bValid)
 *
 * PURPOSE:         This function
 *
 * ARGUMENTS:       int
          iTermId   client browser terminal id
 *                                    int
                              iSyncId   client browser
sync id
 *                                    BOOL
                    bInput    TRUE if form is being
constructed for input else FALSE
 *                                    BOOL
                    bValid    TRUE if NewOrderData
valid, ELSE FALSE effects output only
 *
 * RETURNS:                 char *
          A pointer to buffer inside client structure
where HTML form is built.
 *
 * COMMENTS:        The internal client buffer is
created when the terminal id is assigned and should not
 *                                    be freed except
when the client terminal id is no longer needed.
 */

static char *MakeNewOrderForm(int iTermId, int iSyncId,
BOOL Rollback, BOOL bInput, BOOL bValid)
{
          char      *szForm;
          char      szName[146];
          char      szCredit[14];
          int                 i;

          szForm = (char
*)Term.pClientData[iTermId].szBuffer;

          Term.pClientData[iTermId].NewOrderData.w_id =
Term.pClientData[iTermId].w_id;

          strcpy(szForm,      "<HTML>"

          "<HEAD><TITLE>TPC-C New
Order</TITLE></HEAD><BODY>"

          "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
);

          if ( bInput )
          {
                    strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">");
                    strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">");
          }
```

```
          else
          {
                    if ( bValid )
                              strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">");
                    else

          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">",
ERR_BAD_ITEM_ID);
          }

          if (Rollback == FALSE)
                    strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">");
          else
                    strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"1\">");

          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">",
NEW_ORDER_FORM);
          wsprintf(szForm+strlen(szForm),
          "<INPUT TYPE=\"hidden\" NAME=\"TERMID\"
VALUE=\"%d\">", iTermId);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);
          strcat(szForm, "<PRE>
New Order<BR>");

          if ( bInput )
          {
                    wsprintf(szForm+strlen(szForm),
"Warehouse: %4.4d   District: <INPUT NAME=\"DID*\"
SIZE=1>                         Date:<BR>",
Term.pClientData[iTermId].NewOrderData.w_id);
                    strcat(szForm,      "Customer:
<INPUT NAME=\"CID*\" SIZE=4>   Name:
Credit:      %Disc:<BR>"

          "Order Number:           Number of Lines:
W_tax:         D_tax:<BR><BR>"

          " Supp_W  Item_Id  Item Name
Qty  Stock  B/G  Price    Amount<BR>"

          " <INPUT NAME=\"SP00*\" SIZE=4>  <INPUT
NAME=\"IID00*\" SIZE=6>
<INPUT NAME=\"Qty00*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP01*\" SIZE=4>  <INPUT
NAME=\"IID01*\" SIZE=6>
<INPUT NAME=\"Qty01*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP02*\" SIZE=4>  <INPUT
NAME=\"IID02*\" SIZE=6>
<INPUT NAME=\"Qty02*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP03*\" SIZE=4>  <INPUT
NAME=\"IID03*\" SIZE=6>
<INPUT NAME=\"Qty03*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP04*\" SIZE=4>  <INPUT
NAME=\"IID04*\" SIZE=6>
<INPUT NAME=\"Qty04*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP05*\" SIZE=4>  <INPUT
NAME=\"IID05*\" SIZE=6>
<INPUT NAME=\"Qty05*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP06*\" SIZE=4>  <INPUT
NAME=\"IID06*\" SIZE=6>
<INPUT NAME=\"Qty06*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP07*\" SIZE=4>  <INPUT
NAME=\"IID07*\" SIZE=6>
<INPUT NAME=\"Qty07*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP08*\" SIZE=4>  <INPUT
NAME=\"IID08*\" SIZE=6>
<INPUT NAME=\"Qty08*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP09*\" SIZE=4>  <INPUT
NAME=\"IID09*\" SIZE=6>
<INPUT NAME=\"Qty09*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP10*\" SIZE=4>  <INPUT
NAME=\"IID10*\" SIZE=6>
<INPUT NAME=\"Qty10*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP11*\" SIZE=4>  <INPUT
NAME=\"IID11*\" SIZE=6>
<INPUT NAME=\"Qty11*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP12*\" SIZE=4>  <INPUT
NAME=\"IID12*\" SIZE=6>
<INPUT NAME=\"Qty12*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP13*\" SIZE=4>  <INPUT
NAME=\"IID13*\" SIZE=6>
<INPUT NAME=\"Qty13*\" SIZE=1><BR>"

          " <INPUT NAME=\"SP14*\" SIZE=4>  <INPUT
NAME=\"IID14*\" SIZE=6>
<INPUT NAME=\"Qty14*\" SIZE=1><BR>"

          "Execution Status:
Total:<BR><HR>"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Menu\">"

          "</FORM>"

          "</HTML>" );

          }
          else
          {
                    if ( bValid )
                    {

          wsprintf(szForm+strlen(szForm), "Warehouse:
%4.4d   District: %2.2d                         Date:
%2.2d-%2.2d-%4.4d %2.2d:%2.2d:%2.2d <BR>",

          Term.pClientData[iTermId].NewOrderData.w_id,

          Term.pClientData[iTermId].NewOrderData.d_id,

          Term.pClientData[iTermId].NewOrderData.o_entr
y_d.day,

          Term.pClientData[iTermId].NewOrderData.o_entr
y_d.month,

          Term.pClientData[iTermId].NewOrderData.o_entr
y_d.year,

          Term.pClientData[iTermId].NewOrderData.o_entr
y_d.hour,

          Term.pClientData[iTermId].NewOrderData.o_entr
y_d.minute,

          Term.pClientData[iTermId].NewOrderData.o_entr
y_d.second);
                    }
                    else
                    {

          wsprintf(szForm+strlen(szForm), "Warehouse:
%4.4d   District: %2.2d
Date:<BR>",

          Term.pClientData[iTermId].NewOrderData.w_id,

          Term.pClientData[iTermId].NewOrderData.d_id);
                    }

                    FormatHTMLString(szName,
Term.pClientData[iTermId].NewOrderData.c_last, 16),
                    FormatHTMLString(szCredit,
Term.pClientData[iTermId].NewOrderData.c_credit, 2);

                    wsprintf(szForm+strlen(szForm),
"Customer:  %4.4d   Name: %s   Credit: %s   ",

          Term.pClientData[iTermId].NewOrderData.c_id,
szName, szCredit);


                    if ( bValid )
                    {

          sprintf(szForm+strlen(szForm), "%%Disc: %5.2f
<BR>",
Term.pClientData[iTermId].NewOrderData.c_discount *
100.0);

          sprintf(szForm+strlen(szForm), "Order Number:
```

```
%8.8d  Number of Lines: %2.2d        W_tax: %5.2f
D_tax: %5.2f  <BR><BR>",

          Term.pClientData[iTermId].NewOrderData.o_id,

          Term.pClientData[iTermId].NewOrderData.o_ol_c
nt,

          Term.pClientData[iTermId].NewOrderData.w_tax
* 100.0,

          Term.pClientData[iTermId].NewOrderData.d_tax
* 100.0);

                              strcat(szForm, " Supp_W
Item_Id  Item Name                 Qty  Stock  B/G
Price    Amount<BR>");
                              for(i=0;
i<Term.pClientData[iTermId].NewOrderData.o_ol_cnt; i++)
                              {

          FormatHTMLString(szName,
Term.pClientData[iTermId].NewOrderData.Ol[i].ol_i_name,
24);


          sprintf(szForm+strlen(szForm), "  %4.4d
%6.6d   %s  %2.2d    %3.3d    %1.1s   $%6.2f  $%7.2f
<BR>",

          Term.pClientData[iTermId].NewOrderData.Ol[i].
ol_supply_w_id,

          Term.pClientData[iTermId].NewOrderData.Ol[i].
ol_i_id,

          szName,

          Term.pClientData[iTermId].NewOrderData.Ol[i].
ol_quantity,

          Term.pClientData[iTermId].NewOrderData.Ol[i].
ol_stock,

          Term.pClientData[iTermId].NewOrderData.Ol[i].
ol_brand_generic,

          Term.pClientData[iTermId].NewOrderData.Ol[i].
ol_i_price,

          Term.pClientData[iTermId].NewOrderData.Ol[i].
ol_amount );
                              }
                    }
                    else
                    {
                              strcat(szForm,
"%Disc:<BR>");

          sprintf(szForm+strlen(szForm), "Order Number:
%8.8d  Number of Lines:              W_tax:
D_tax:<BR><BR>",

          Term.pClientData[iTermId].NewOrderData.o_id);

                              strcat(szForm, " Supp_W
Item_Id  Item Name                 Qty  Stock  B/G
Price    Amount<BR>");

                              i = 0;
                    }
                    for(; i<15; i++)
                              strcat(szForm, "<BR>");

                    if ( bValid )
                    {

          sprintf(szForm+strlen(szForm), "Execution
Status: %24.24s                 Total:  $%8.2f  ",

          Term.pClientData[iTermId].NewOrderData.execut
ion_status,

          Term.pClientData[iTermId].NewOrderData.total_
amount);
                    }
                    else
                    {

          sprintf(szForm+strlen(szForm), "Execution
Status: %24.24s                 Total:",

          Term.pClientData[iTermId].NewOrderData.execut
ion_status);
                    }

                    strcat(szForm,
          "</PRE><HR><BR>"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..NewOrder..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">" );
                    strcat(szForm, "</FORM></HTML>");
          }

          return szForm;

}

/* FUNCTION: char *MakePaymentForm(int iTermId, int
iSyncId, BOOL bInput)
 *
 * PURPOSE:          This function
 *
 * ARGUMENTS:        int
          iTermId   client browser terminal id
 *                                   int
                              iSyncId   client browser
sync id
 *                                   BOOL
                    bInput    TRUE if form is being
constructed for input else FALSE
 *
 * RETURNS:                  char *
          A pointer to buffer inside client structure
where HTML form is built.
 *
 * COMMENTS:         The internal client buffer is
created when the terminal id is assigned and should not
 *                                   be freed except
when the client terminal id is no longer needed.
 */

static char *MakePaymentForm(int iTermId, int iSyncId,
BOOL bInput)
{
          char      *szForm;
          char      *ptr;
          char      szTmp[64];
          char      szW_Zip[26];
          char      szD_Zip[26];
          char      szC_Zip[26];
          char      szC_Phone[26];
          char      szTmpStr1[122];
          char      szTmpStr2[122];
          char      szTmpStr3[122];
          char      szTmpStr4[122];
          int                 i;
          int                 l;
          char      *szZipPic = "XXXXX-XXXX";

          szForm = (char
*)Term.pClientData[iTermId].szBuffer;

          Term.pClientData[iTermId].PaymentData.w_id =
Term.pClientData[iTermId].w_id;

          strcpy(szForm,      "<HTML><HEAD><TITLE>TPC-C
Payment</TITLE></HEAD><BODY>"

          "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">");
          if ( bInput )
                    strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">");

          strcat(szForm, "<INPUT TYPE=\"hidden\"
NAME=\"STATUSID\" VALUE=\"0\">");
          strcat(szForm, "<INPUT TYPE=\"hidden\"
NAME=\"ERROR\" VALUE=\"0\">");
```

```
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">",
PAYMENT_FORM);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);

          strcat(szForm,      "<PRE>
Payment<BR>");

          if ( bInput )
                    strcat(szForm,      "Date:<BR><BR>"
);
          else
          {
                    wsprintf(szForm+strlen(szForm),
"Date: %2.2d-%2.2d-%4.4d %2.2d:%2.2d:%2.2d <BR><BR>",

          Term.pClientData[iTermId].PaymentData.h_date.
day,

          Term.pClientData[iTermId].PaymentData.h_date.
month,

          Term.pClientData[iTermId].PaymentData.h_date.
year,

          Term.pClientData[iTermId].PaymentData.h_date.
hour,

          Term.pClientData[iTermId].PaymentData.h_date.
minute,

          Term.pClientData[iTermId].PaymentData.h_date.
second);
          }
          wsprintf(szForm+strlen(szForm), "Warehouse:
%4.4d", Term.pClientData[iTermId].PaymentData.w_id);

          if ( bInput )
          {
                    strcat(szForm,      "
District: <INPUT NAME=\"DID*\"
SIZE=1><BR><BR><BR><BR><BR>"

          "Customer: <INPUT NAME=\"CID*\" SIZE=4>"

          "Cust-Warehouse: <INPUT NAME=\"CWI*\" SIZE=4>
"

          "Cust-District: <INPUT NAME=\"CDI*\"
SIZE=1><BR>"

          "Name:                         <INPUT
NAME=\"CLT*\" SIZE=16>                    Since:<BR>"

          "
Credit:<BR>"

          "
Disc:<BR>"

          "
Phone:<BR><BR>"

          "Amount Paid:          $<INPUT NAME=\"HAM*\"
SIZE=7>      New Cust Balance:<BR>"

          "Credit Limit:<BR><BR>Cust-Data:
<BR><BR><BR><BR></PRE><HR>"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\"><INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Menu\">"

          "</BODY></FORM></HTML>" );
          }
          else
          {
                    sprintf(szForm+strlen(szForm),
          "                           District:
%2.2d<BR>",

          Term.pClientData[iTermId].PaymentData.d_id);

                    FormatHTMLString(szTmpStr1,
Term.pClientData[iTermId].PaymentData.w_street_1, 20);
                    FormatHTMLString(szTmpStr2,
Term.pClientData[iTermId].PaymentData.d_street_1, 20);

                    sprintf(szForm+strlen(szForm),"%s
%s<BR>", szTmpStr1, szTmpStr2);

                    FormatHTMLString(szTmpStr1,
Term.pClientData[iTermId].PaymentData.w_street_2, 20);
                    FormatHTMLString(szTmpStr2,
Term.pClientData[iTermId].PaymentData.d_street_2, 20);

                    sprintf(szForm+strlen(szForm),"%s
%s<BR>", szTmpStr1, szTmpStr2);

                    FormatString(szW_Zip, szZipPic,
Term.pClientData[iTermId].PaymentData.w_zip);
                    FormatString(szD_Zip, szZipPic,
Term.pClientData[iTermId].PaymentData.d_zip);

                    FormatHTMLString(szTmpStr1,
Term.pClientData[iTermId].PaymentData.w_city, 20);
                    FormatHTMLString(szTmpStr2,
Term.pClientData[iTermId].PaymentData.w_state, 2);
                    FormatHTMLString(szTmpStr3,
Term.pClientData[iTermId].PaymentData.d_city, 20);
                    FormatHTMLString(szTmpStr4,
Term.pClientData[iTermId].PaymentData.d_state, 2);

                    wsprintf(szForm+strlen(szForm), "%s
%s %10.10s       %s %s %10.10s<BR><BR>",
                              szTmpStr1, szTmpStr2,
szW_Zip, szTmpStr3, szTmpStr4, szD_Zip );

                    wsprintf(szForm+strlen(szForm),
"Customer: %4.4d  Cust-Warehouse: %4.4d  Cust-District:
%2.2d<BR>",

          Term.pClientData[iTermId].PaymentData.c_id,

          Term.pClientData[iTermId].PaymentData.c_w_id,

          Term.pClientData[iTermId].PaymentData.c_d_id)
;

                    FormatHTMLString(szTmpStr1,
Term.pClientData[iTermId].PaymentData.c_first, 16);
                    FormatHTMLString(szTmpStr2,
Term.pClientData[iTermId].PaymentData.c_middle, 2);
                    FormatHTMLString(szTmpStr3,
Term.pClientData[iTermId].PaymentData.c_last, 16);

                    wsprintf(szForm+strlen(szForm),
"Name:   %s %s %s     Since:  %2.2d-%2.2d-%4.4d<BR>",
                              szTmpStr1, szTmpStr2,
szTmpStr3,

          Term.pClientData[iTermId].PaymentData.c_since
.day,

          Term.pClientData[iTermId].PaymentData.c_since
.month,

          Term.pClientData[iTermId].PaymentData.c_since
.year);

                    FormatHTMLString(szTmpStr1,
Term.pClientData[iTermId].PaymentData.c_street_1, 20);
                    FormatHTMLString(szTmpStr2,
Term.pClientData[iTermId].PaymentData.c_credit, 2);

          wsprintf(szForm+strlen(szForm), "        %s
Credit: %s<BR>", szTmpStr1, szTmpStr2);

                    FormatHTMLString(szTmpStr1,
Term.pClientData[iTermId].PaymentData.d_street_2, 20);
          sprintf(szForm+strlen(szForm), "        %s
%%Disc:  %5.2f<BR>",
                              szTmpStr1,
Term.pClientData[iTermId].PaymentData.c_discount *
100.0);

                    FormatString(szC_Zip, szZipPic,
Term.pClientData[iTermId].PaymentData.c_zip);
                    FormatString(szC_Phone, "XXXXXX-
XXX-XXX-XXXX",
Term.pClientData[iTermId].PaymentData.c_phone);

                    FormatHTMLString(szTmpStr1,
Term.pClientData[iTermId].PaymentData.c_city, 20);
                    FormatHTMLString(szTmpStr2,
Term.pClientData[iTermId].PaymentData.c_state, 2);

                    wsprintf(szForm+strlen(szForm), "
%s %s %10.10s       Phone:  %-19.19s<BR><BR>",
                              szTmpStr1, szTmpStr2,
szC_Zip, szC_Phone );
```

```
                    sprintf(szForm+strlen(szForm),
"Amount Paid:          $%7.2f      New Cust Balance:
$%14.2f<BR>",

          Term.pClientData[iTermId].PaymentData.h_amoun
t,

          Term.pClientData[iTermId].PaymentData.c_balan
ce);

                    sprintf(szForm+strlen(szForm),
"Credit Limit:   $%13.2f<BR><BR>",

          Term.pClientData[iTermId].PaymentData.c_credi
t_lim);

                    ptr =
Term.pClientData[iTermId].PaymentData.c_credit;
                    if ( *ptr == 'B' && *(ptr+1) == 'C'
)
                    {
                              ptr =
Term.pClientData[iTermId].PaymentData.c_data;
                              l = strlen( ptr ) / 50;
                              for(i=0; i<4; i++, ptr +=
50)
                              {
                                        if ( i <= l )

          UtilStrCpy(szTmp, ptr, 50);
                                        else

          szTmp[0] = 0;
                                        if ( !i )
                                        {

          FormatHTMLString(szTmpStr1, szTmp, 50);

          wsprintf(szForm+strlen(szForm), "Cust-Data:
%s<BR>", szTmpStr1);
                                        }
                                        else
                                        {

          FormatHTMLString(szTmpStr1, szTmp, 50);

          wsprintf(szForm+strlen(szForm), "
%s<BR>", szTmpStr1);
                                        }
                              }
                    }
                    else
                              strcat(szForm, "Cust-
Data: <BR><BR><BR><BR>");

                    strcat(szForm,
          "</PRE><HR><BR>"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..NewOrder..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">"

          "</BODY></FORM></HTML>");
          }

          return szForm;
}

/* FUNCTION: char *MakeOrderStatusForm(int iTermId, int
iSyncId, BOOL bInput)
 *
 * PURPOSE:         This function
 *
 * ARGUMENTS:       int
          iTermId   client browser terminal id
 *                                  int
                    iSyncId   client browser
sync id
 *                                  BOOL
                    bInput    TRUE if form is being
constructed for input else FALSE
 *
 * RETURNS:                 char *
          A pointer to buffer inside client structure
where HTML form is built.
 *
 * COMMENTS:        The internal client buffer is
created when the terminal id is assigned and should not
 *                                  be freed except
when the client terminal id is no longer needed.
 */

static char *MakeOrderStatusForm(int iTermId, int
iSyncId, BOOL bInput)
{
          char      *szForm;
          char      c_first[98];
          char      c_middle[14];
          char      c_last[98];
          int                 i;

          szForm = (char
*)Term.pClientData[iTermId].szBuffer;

          Term.pClientData[iTermId].OrderStatusData.w_i
d = Term.pClientData[iTermId].w_id;

          strcpy(szForm,      "<HTML><HEAD><TITLE>TPC-C
Order-Status</TITLE></HEAD><BODY>"

          "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">");

          if ( bInput )
                    strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">");

          strcat(szForm, "<INPUT TYPE=\"hidden\"
NAME=\"STATUSID\" VALUE=\"0\">");
          strcat(szForm, "<INPUT TYPE=\"hidden\"
NAME=\"ERROR\" VALUE=\"0\">");
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">",
ORDER_STATUS_FORM);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);

          strcat(szForm,      "<PRE>
Order-Status<BR>" );
          wsprintf(szForm+strlen(szForm), "Warehouse:
%4.4d   ",
Term.pClientData[iTermId].OrderStatusData.w_id);

          if ( bInput )
          {
                    strcat(szForm,      "District:
<INPUT NAME=\"DID*\" SIZE=1><BR>"

          "Customer: <INPUT NAME=\"CID*\" SIZE=4>
Name:                    <INPUT NAME=\"CLT*\"
SIZE=23><BR>"

          "Cust-Balance:<BR><BR>"

          "Order-Number:            Entry-Date:
Carrier-Number:<BR>"

          "Supply-W     Item-Id    Qty     Amount
Delivery-Date<BR></PRE>"

          "<HR><INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\"><INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Menu\">"

          "</BODY></FORM></HTML>" );
          }
          else
          {
                    wsprintf(szForm+strlen(szForm),
"District: %2.2d<BR>",
Term.pClientData[iTermId].OrderStatusData.d_id);

                    FormatHTMLString(c_first,
Term.pClientData[iTermId].OrderStatusData.c_first, 16);
                    FormatHTMLString(c_middle,
Term.pClientData[iTermId].OrderStatusData.c_middle, 2);
                    FormatHTMLString(c_last,
Term.pClientData[iTermId].OrderStatusData.c_last, 16);
```

```
                    wsprintf(szForm+strlen(szForm),
"Customer: %4.4d   Name: %s %s %s<BR>",

          Term.pClientData[iTermId].OrderStatusData.c_i
d, c_first, c_middle, c_last);

                    sprintf(szForm+strlen(szForm),
"Cust-Balance: $%9.2f<BR><BR>",

          Term.pClientData[iTermId].OrderStatusData.c_b
alance);

                    wsprintf(szForm+strlen(szForm),
"Order-Number: %8.8d   Entry-Date: %2.2d-%2.2d-%4.4d
%2.2d:%2.2d:%2.2d   Carrier-Number: %2.2d<BR>",

          Term.pClientData[iTermId].OrderStatusData.o_i
d,

          Term.pClientData[iTermId].OrderStatusData.o_e
ntry_d.day,

          Term.pClientData[iTermId].OrderStatusData.o_e
ntry_d.month,

          Term.pClientData[iTermId].OrderStatusData.o_e
ntry_d.year,

          Term.pClientData[iTermId].OrderStatusData.o_e
ntry_d.hour,

          Term.pClientData[iTermId].OrderStatusData.o_e
ntry_d.minute,

          Term.pClientData[iTermId].OrderStatusData.o_e
ntry_d.second,

          Term.pClientData[iTermId].OrderStatusData.o_c
arrier_id);
                    strcat(szForm+strlen(szForm),
"Supply-W     Item-Id    Qty     Amount      Delivery-
Date<BR>");

                    for(i=0;
i<Term.pClientData[iTermId].OrderStatusData.o_ol_cnt;
i++)
                    {

          sprintf(szForm+strlen(szForm), "  %4.4d
%6.6d     %2.2d     $%8.2f      %2.2d-%2.2d-%4.4d<BR>",

          Term.pClientData[iTermId].OrderStatusData.Ol0
rderStatusData[i].ol_supply_w_id,

          Term.pClientData[iTermId].OrderStatusData.Ol0
rderStatusData[i].ol_i_id,

          Term.pClientData[iTermId].OrderStatusData.Ol0
rderStatusData[i].ol_quantity,

          Term.pClientData[iTermId].OrderStatusData.Ol0
rderStatusData[i].ol_amount,

          Term.pClientData[iTermId].OrderStatusData.Ol0
rderStatusData[i].ol_delivery_d.day,

          Term.pClientData[iTermId].OrderStatusData.Ol0
rderStatusData[i].ol_delivery_d.month,

          Term.pClientData[iTermId].OrderStatusData.Ol0
rderStatusData[i].ol_delivery_d.year);
                    }

                    strcat(szForm,
          "<BR></PRE><HR><INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..NewOrder..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">"

          "</BODY></FORM></HTML>" );
          }

          return szForm;
}

/* FUNCTION: char *MakeDeliveryForm(int iTermId, int
iSyncId, BOOL bInput, BOOL bSuccess)
 *
 * PURPOSE:         This function
 *
 * ARGUMENTS:       int
          iTermId   client browser terminal id
 *                                    int
                              iSyncId   client browser
sync id
 *                                    BOOL
                    bInput    TRUE if form is being
constructed for input else FALSE
 *                                    BOOL
                    bSuccess TRUE if Delivery succeeded
else FALSE
 *
 * RETURNS:                   char *
          A pointer to buffer inside client structure
where HTML form is built.
 *
 * COMMENTS:        The internal client buffer is
created when the terminal id is assigned and should not
 *                                    be freed except
when the client terminal id is no longer needed.
 */

static char *MakeDeliveryForm(int iTermId, int iSyncId,
BOOL bInput, BOOL bSuccess)
{
          char      *szForm;

          szForm = (char
*)Term.pClientData[iTermId].szBuffer;

          Term.pClientData[iTermId].DeliveryData.w_id =
Term.pClientData[iTermId].w_id;

          strcpy( szForm,     "<HTML><HEAD><TITLE>TPC-C
Delivery</TITLE></HEAD><BODY>"

          "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">");

          if ( bInput )
          {
                    strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">");
                    strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">");
          }
          else
          {
                    if ( !bSuccess )
                    {

          sprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">",
ERR_TYPE_DELIVERY_POST);
                              strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"2\">");
                    }
                    else
                    {
                              strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">");
                              strcat(szForm, "<INPUT
TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">");
                    }
          }

          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">",
DELIVERY_FORM);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">",
iTermId);
          wsprintf(szForm+strlen(szForm), "<INPUT
TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
iSyncId);

          strcat(szForm,      "<PRE>
Delivery<BR>" );

          wsprintf(szForm+strlen(szForm), "Warehouse:
%4.4d<BR><BR>",
Term.pClientData[iTermId].DeliveryData.w_id);
```

```
          if ( bInput )
                    strcat( szForm, "Carrier Number:
<INPUT NAME=\"OCD*\" SIZE=1><BR><BR>");
          else
          {
                    wsprintf(szForm+strlen(szForm),
"Carrier Number: %2.2d<BR><BR>",

          Term.pClientData[iTermId].DeliveryData.o_carr
ier_id);
          }
          if ( bInput )
          {
                    strcat( szForm, "Execution
Status:<BR></PRE>"

          "<HR><INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Menu\">" );
          }
          else
          {
                    wsprintf(szForm+strlen(szForm),
"Execution Status: %25.25s<BR></PRE>",

          Term.pClientData[iTermId].DeliveryData.execut
ion_status);

                    strcat(szForm,      "<HR><INPUT
TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"

          "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">" );
          }

          strcat( szForm,     "</BODY></FORM></HTML>"
);

          return szForm;
}

/* FUNCTION: void UtilStrCpy(char * pDest, char * pSrc,
int n)
 *
 * PURPOSE:          This function copies n characters
from string pSrc to pDst and places a
 *                            null character at the end
of the destination string.
 *
 * ARGUMENTS:        char
          *pDest    destination string pointer
 *                                    char
                    *pSrc     source string pointer
 *                                  int
                              n
          number of characters to copy
 *
 * RETURNS:                   None
 *
 * COMMENTS:         Unlike strncpy this function
ensures that the result string is
 *                                  always null
terminated.
 *
 */

static void UtilStrCpy(char * pDest, char * pSrc, int
n)
{
    strncpy(pDest, pSrc, n);
    pDest[n] = '\0';

          return;
}

/* FUNCTION: void
ProcessNewOrderForm(EXTENSION_CONTROL_BLOCK
          *pECB, int iTermId, int iSyncId)
 *
 * PURPOSE:          This function gets and validates
the input data from the new order form
 *                            filling in the required
input variables. it then calls the SQLNewOrder
 *                           transaction, constructs
the output form and writes it back to client
 *                           browser.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                    int

          iTermId   client browser terminal id
 *                                    int

          iSyncId client browser sync id
 *
 * RETURNS:                   None
 *
 * COMMENTS:         None
 *
 */

 // Allocate the tpalloc structure for each transaction
//        This saves on some memory at the expense of
some CPU cycles.
static void ProcessNewOrderForm(EXTENSION_CONTROL_BLOCK
          *pECB, int iTermId, int iSyncId)
{
           int      TpRc, iRc, iError;
           long     ilen, *olen;
           char     buf[128];
#ifdef LOCAL_ALLOC
           NEW_ORDER_DATA
          *NewOrderDataPtr;   //New Order Tuxedo Buffer
#endif
           if((iRc = ThrTpInit()) <0)
           {
                      // This is bad
                     sprintf(buf, "ProcessNewOrder
Failed ThrTpInit: iRc = %d", iRc);
                     LogTuxError(0, buf);
                     ErrorMessage(pECB,
ERR_NEW_ORDER_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                     return;
           }

          memset(&Term.pClientData[iTermId].NewOrderDat
a, 0, sizeof(NEW_ORDER_DATA));

          Term.pClientData[iTermId].NewOrderData.w_id =
Term.pClientData[iTermId].w_id;

          if ( (iError=GetNewOrderData(pECB-
>lpszQueryString,
&Term.pClientData[iTermId].NewOrderData)) !=
ERR_SUCCESS )
          {
                    ErrorMessage(pECB, iError,
ERR_TYPE_WEBDLL, NULL, iTermId, iSyncId);
                    return;
          }

#ifdef LOCAL_ALLOC
          if ((NewOrderDataPtr = (NEW_ORDER_DATA
*)tpalloc("CARRAY", NULL, sizeof(NEW_ORDER_DATA))) ==
NULL)
          {
                     TpRc = tperrno;
                     sprintf(buf, "ProcessNewOrder
Tpcalloc Failed");
                     LogTuxError(TpRc, buf);
                     ErrorMessage(pECB,
ERR_NEW_ORDER_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                     tpfree((char *)NewOrderDataPtr);
                     return;
          }

           *NewOrderDataPtr =
Term.pClientData[iTermId].NewOrderData;

     ilen = sizeof(NEW_ORDER_DATA);
     olen = &ilen;

           if ( dLog )
           {
                     FILE *fp;

                     fp = fopen(szTpccLogPath, "ab");
```

```
                    fprintf(fp, "* ProcessNewOrderL
Thread %d iTermId %d NewOrderDataPtr: %x size %d \r\n",
                                GetCurrentThreadId(),
iTermId, &NewOrderDataPtr, sizeof(*NewOrderDataPtr));
                    fclose(fp);
          }
#ifdef MULTI_TRANS
          if ((iRc = tpcall(qname[iTermId%10], (char
*)NewOrderDataPtr, ilen,
                    (char **)&NewOrderDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
#else
     if ((iRc = tpcall("NEWORDER", (char
*)NewOrderDataPtr, ilen,
                    (char **)&NewOrderDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
#endif
          {
         TpRc = tperrno;
                     sprintf(buf, "Neworder tpcall
failed");
                     LogTuxError(TpRc, buf);
                     ErrorMessage(pECB,
ERR_NEW_ORDER_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
         tpfree((char *)NewOrderDataPtr);
                     return;
     }

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "** ProcessNewOrderL
Thread %d iTermId %d NewOrderDataPtr: %x size %d\r\n",
                                GetCurrentThreadId(),
iTermId, &NewOrderDataPtr, sizeof(*NewOrderDataPtr));
                    fclose(fp);
           }

           Term.pClientData[iTermId].NewOrderData =
*NewOrderDataPtr;

     iRc = NewOrderDataPtr->retval;
           iError = NewOrderDataPtr->error;

           tpfree((char *)NewOrderDataPtr);

          if ( iRc < 0 )
          {
                    if (iError == ERR_TYPE_DEADLOCK)
                              ErrorMessage(pECB,
ERR_NEW_ORDER_NOT_PROCESSED, ERR_TYPE_DEADLOCK, NULL,
iTermId, iSyncId);
                    else
                              ErrorMessage(pECB,
ERR_NEW_ORDER_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
          }
          else
                    if ( iError == ERR_BAD_ITEM_ID)
                              WriteZString(pECB,
MakeNewOrderForm(iTermId, iSyncId, TRUE, FALSE,
(BOOL)iRc) );
                    else
                              WriteZString(pECB,
MakeNewOrderForm(iTermId, iSyncId, FALSE, FALSE,
(BOOL)iRc) );

          return;

#else // Not LOCAL_ALLOC

          *Term.pClientData[iTermId].NewOrderDataPtr =
Term.pClientData[iTermId].NewOrderData;

     ilen = sizeof(NEW_ORDER_DATA);
     olen = &ilen;

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "* ProcessNewOrder
Thread %d iTermId %d NewOrderDataPtr: %x size %d \r\n",
                              GetCurrentThreadId(),
iTermId, &Term.pClientData[iTermId].NewOrderDataPtr,

          sizeof(*Term.pClientData[iTermId].NewOrderDat
aPtr));
                    fclose(fp);
           }

          }
#ifdef MULTI_TRANS
          if ((iRc = tpcall(qname[iTermId%10], (char
*)Term.pClientData[iTermId].NewOrderDataPtr, ilen,
                    (char
**)&Term.pClientData[iTermId].NewOrderDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
#else
     if ((iRc = tpcall("NEWORDER", (char
*)Term.pClientData[iTermId].NewOrderDataPtr, ilen,
                    (char
**)&Term.pClientData[iTermId].NewOrderDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
#endif
     //if ((iRc = tpcall("NEWORDER", (char
*)Term.pClientData[iTermId].TuxDataPtr, ilen,
          //         (char
**)&Term.pClientData[iTermId].TuxDataPtr, (long *)olen,
TPSIGRSTRT)) == -1)
          {
         TpRc = tperrno;
                    sprintf(buf, "Neworder tpcall
failed");
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_NEW_ORDER_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
         tpfree((char *)NewOrderDataPtr);
                    return;
     }

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "** ProcessNewOrder
Thread %d iTermId %d NewOrderDataPtr: %x size %d\r\n",
                              GetCurrentThreadId(),
iTermId, &Term.pClientData[iTermId].NewOrderDataPtr,

          sizeof(*Term.pClientData[iTermId].NewOrderDat
aPtr));
                    fclose(fp);
           }

     Term.pClientData[iTermId].NewOrderData =
*Term.pClientData[iTermId].NewOrderDataPtr;

     iRc = Term.pClientData[iTermId].NewOrderDataPtr-
>retval;

           iError =
Term.pClientData[iTermId].NewOrderDataPtr->error;
#endif
          if ( iRc < 0 )
          {
                    if (iError == ERR_TYPE_DEADLOCK)
                              ErrorMessage(pECB,
ERR_NEW_ORDER_NOT_PROCESSED, ERR_TYPE_DEADLOCK, NULL,
iTermId, iSyncId);
                    else
                              ErrorMessage(pECB,
ERR_NEW_ORDER_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
          }
          else
                    if ( iError == ERR_BAD_ITEM_ID)
                              WriteZString(pECB,
MakeNewOrderForm(iTermId, iSyncId, TRUE, FALSE,
(BOOL)iRc) );
                    else
                              WriteZString(pECB,
MakeNewOrderForm(iTermId, iSyncId, FALSE, FALSE,
(BOOL)iRc) );

          return;
}

/* FUNCTION: void
ProcessPaymentForm(EXTENSION_CONTROL_BLOCK *pECB, int
iTermId, int iSyncId)
 *
 * PURPOSE:          This function gets and validates
the input data from the payment form
 *                              filling in the required
input variables. It then calls the SQLPayment
 *                              transaction, constructs
the output form and writes it back to client
 *                              browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
```

```
 *                                      int

         iTermId   client browser terminal id
 *                                      int

         iSyncId client browser sync id
 *
 * RETURNS:                    None
 *
 * COMMENTS:        None
 *
 */

 // Allocate the tpalloc structure for each transaction
//        This saves on some memory at the expense of
some CPU cycles.
static void ProcessPaymentForm(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId)
{
           int               TpRc, iRc, iError;
           long     ilen, *olen;
           char     buf[128];

#ifdef LOCAL_ALLOC
           PAYMENT_DATA
          *PaymentDataPtr;    //Payment Tuxedo Buffer
#endif
     if((iRc = ThrTpInit()) <0)
           {
                     // This is bad
                    sprintf(buf, "ProcessPayment
Failed ThrTpInit: iRc = %d", iRc);
                    LogTuxError(0, buf);
                    ErrorMessage(pECB,
ERR_PAYMENT_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    return;
           }


memset(&Term.pClientData[iTermId].PaymentData, 0,
sizeof(PAYMENT_DATA));

     Term.pClientData[iTermId].PaymentData.w_id =
Term.pClientData[iTermId].w_id;

     if ( (iError=GetPaymentData(pECB->lpszQueryString,
&Term.pClientData[iTermId].PaymentData)) != ERR_SUCCESS
)
     {
          ErrorMessage(pECB, iError, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
          return;
     }

#ifdef LOCAL_ALLOC
           if ((PaymentDataPtr = (PAYMENT_DATA
*)tpalloc("CARRAY", NULL, sizeof(PAYMENT_DATA))) ==
NULL)
          {
                    TpRc = tperrno;
                    sprintf(buf, "ProcessPayment
Tpcalloc Failed");
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_PAYMENT_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    tpfree((char *)PaymentDataPtr);
                    return;
          }

     *PaymentDataPtr =
Term.pClientData[iTermId].PaymentData;

           ilen = sizeof(PAYMENT_DATA);
     olen = &ilen;

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "* ProcessPayment
Thread %d iTermId %d PaymentDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &PaymentDataPtr);
                    fclose(fp);
           }

#ifdef MULTI_TRANS
           if (( iRc = tpcall(qname[iTermId%10], (char
*)PaymentDataPtr, ilen,
                    (char **)&PaymentDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
#else
          if (( iRc = tpcall("PAYMENT", (char
*)PaymentDataPtr, ilen,
                    (char **)&PaymentDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
#endif
           {
          TpRc = tperrno;
                    sprintf(buf, "ProcessPayment
tpcall failed,qname=%s",qname[iTermId%10]);
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_PAYMENT_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
          tpfree((char *)PaymentDataPtr);
                    return;
     }

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "** ProcessPayment
Thread %d iTermId %d PaymentDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &PaymentDataPtr);
                    fclose(fp);
           }

     Term.pClientData[iTermId].PaymentData =
*PaymentDataPtr;

     iRc = PaymentDataPtr->retval;
           iError = PaymentDataPtr->error;

           tpfree((char *)PaymentDataPtr);

     if ( iRc < 0 )
           {
                    if (iError == ERR_TYPE_DEADLOCK )
                              ErrorMessage(pECB,
ERR_PAYMENT_NOT_PROCESSED, ERR_TYPE_DEADLOCK, NULL,
iTermId, iSyncId);
                    else if (iError ==
ERR_NOSUCH_CUSTOMER)
                              ErrorMessage(pECB,
ERR_PAYMENT_INVALID_CUSTOMER, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    else
                              ErrorMessage(pECB,
ERR_PAYMENT_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
     }
     else
          WriteZString(pECB, MakePaymentForm(iTermId,
iSyncId, FALSE) );

     //return;


#else // Not LOCAL_ALLOC
           *Term.pClientData[iTermId].PaymentDataPtr =
Term.pClientData[iTermId].PaymentData;

           ilen = sizeof(PAYMENT_DATA);
     olen = &ilen;

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "* ProcessPayment
Thread %d iTermId %d PaymentDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &Term.pClientData[iTermId].PaymentDataPtr);
                    fclose(fp);
           }
#ifdef MULTI_TRANS
     if (( iRc = tpcall(qname[iTermId%10], (char
*)Term.pClientData[iTermId].PaymentDataPtr, ilen,
                    (char
**)&Term.pClientData[iTermId].PaymentDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
#else
           if (( iRc = tpcall("PAYMENT", (char
*)Term.pClientData[iTermId].PaymentDataPtr, ilen,
                    (char
**)&Term.pClientData[iTermId].PaymentDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
```

```
#endif
          {
         TpRc = tperrno;
                    sprintf(buf, "ProcessPayment
tpcall failed");
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_PAYMENT_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
         //tpfree((char *)PaymentDataPtr);
                    return;
     }

          if ( dLog )
          {
                   FILE *fp;

                   fp = fopen(szTpccLogPath, "ab");
                   fprintf(fp, "** ProcessPayment
Thread %d iTermId %d PaymentDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &Term.pClientData[iTermId].PaymentDataPtr);
                   fclose(fp);
          }

     Term.pClientData[iTermId].PaymentData =
*Term.pClientData[iTermId].PaymentDataPtr;

     iRc = Term.pClientData[iTermId].PaymentDataPtr-
>retval;
          iError =
Term.pClientData[iTermId].PaymentDataPtr->error;
     if ( iRc < 0 )
          {
                    if (iError == ERR_TYPE_DEADLOCK )
                              ErrorMessage(pECB,
ERR_PAYMENT_NOT_PROCESSED, ERR_TYPE_DEADLOCK, NULL,
iTermId, iSyncId);
                    else if (iError ==
ERR_NOSUCH_CUSTOMER)
                              ErrorMessage(pECB,
ERR_PAYMENT_INVALID_CUSTOMER, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    else
                              ErrorMessage(pECB,
ERR_PAYMENT_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
     }
     else
          WriteZString(pECB, MakePaymentForm(iTermId,
iSyncId, FALSE) );
#endif

     return;
}

/* FUNCTION: void
ProcessOrderStatusForm(EXTENSION_CONTROL_BLOCK *pECB,
int iTermId, int iSyncId)
 *
 * PURPOSE:         This function gets and validates
the input data from the Order Status
 *                           form filling in the
required input variables. It then calls the
 *                           SQLOrderStatus
transaction, constructs the output form and writes it
 *                           back to client browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                    int

          iTermId   client browser terminal id
 *                                    int

          iSyncId client browser sync id
 *
 * RETURNS:                  None
 *
 * COMMENTS:        None
 *
 */

 // Allocate the tpalloc structure for each transaction
//        This saves on some memory at the expense of
some CPU cycles.
static void
ProcessOrderStatusForm(EXTENSION_CONTROL_BLOCK *pECB,
int iTermId, int iSyncId)
{
          int               TpRc, iRc, iError;
          long      ilen, *olen;
          char      buf[128];

#ifdef LOCAL_ALLOC
          ORDER_STATUS_DATA   *OrderStatusDataPtr;
          //Order Status Tuxedo Buffer
#endif
     if((iRc = ThrTpInit()) <0)
          {
                    // This is bad
                    sprintf(buf, "ProcessOrderStatus
Failed ThrTpInit: iRc = %d", iRc);
                    LogTuxError(0, buf);
                    ErrorMessage(pECB,
ERR_ORDER_STATUS_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    return;
          }

     memset(&Term.pClientData[iTermId].OrderStatusData,
0, sizeof(ORDER_STATUS_DATA));

     Term.pClientData[iTermId].OrderStatusData.w_id =
Term.pClientData[iTermId].w_id;

     if ( (iError=GetOrderStatusData(pECB-
>lpszQueryString,
&Term.pClientData[iTermId].OrderStatusData)) !=
ERR_SUCCESS )
     {
          ErrorMessage(pECB, iError, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
          return;
     }

#ifdef LOCAL_ALLOC

          if ((OrderStatusDataPtr = (ORDER_STATUS_DATA
*)tpalloc("CARRAY", NULL, sizeof(ORDER_STATUS_DATA)))
== NULL)
          {
                    TpRc = tperrno;
                    sprintf(buf, "ProcessOrderStatus
Tpcalloc Failed");
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_ORDER_STATUS_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    tpfree((char *)OrderStatusDataPtr);
                    return;
          }

     *OrderStatusDataPtr =
Term.pClientData[iTermId].OrderStatusData;

           ilen = sizeof(ORDER_STATUS_DATA);
     olen = &ilen;

          if ( dLog )
          {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "* ProcessOrderStatus
Thread %d iTermId %d OrderStatusDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &OrderStatusDataPtr);
                    fclose(fp);
          }

#ifdef MULTI_TRANS
     if (( iRc = tpcall(qname[iTermId%10], (char
*)OrderStatusDataPtr, ilen,
                    (char **)&OrderStatusDataPtr,
(long *)olen, TPSIGRSTRT)) == -1)
#else
          if (( iRc = tpcall("ORDERSTATUS", (char
*)OrderStatusDataPtr, ilen,
                    (char **)&OrderStatusDataPtr,
(long *)olen, TPSIGRSTRT)) == -1)
#endif
          {
         TpRc = tperrno;
                    sprintf(buf, "ProcessOrderStatus
tpcall failed");
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_ORDER_STATUS_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
         tpfree((char *)OrderStatusDataPtr);
                    return;
     }

          if ( dLog )
          {
```

```
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "** ProcessOrderStatus
Thread %d iTermId %d OrderStatusDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &OrderStatusDataPtr);
                    fclose(fp);
          }

     Term.pClientData[iTermId].OrderStatusData =
*OrderStatusDataPtr;

     iRc = OrderStatusDataPtr->retval;
          // iError = OrderStatusDataPtr->error;

          tpfree((char *)OrderStatusDataPtr);

     if ( iRc < 0 )
          {
                    if ( iError == ERR_TYPE_DEADLOCK )
                              ErrorMessage(pECB,
ERR_ORDER_STATUS_NOT_PROCESSED, ERR_TYPE_DEADLOCK,
NULL, iTermId, iSyncId);
                    else if (iError ==
ERR_NOSUCH_CUSTOMER)
                              ErrorMessage(pECB,
ERR_NOSUCH_CUSTOMER, ERR_TYPE_WEBDLL, NULL, iTermId,
iSyncId);
                    else
                              ErrorMessage(pECB,
ERR_ORDER_STATUS_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
     }
     else
          WriteZString(pECB,
MakeOrderStatusForm(iTermId, iSyncId, FALSE) );

     //return;

#else // Not LOCAL_ALLOC

     *Term.pClientData[iTermId].OrderStatusDataPtr =
Term.pClientData[iTermId].OrderStatusData;

          ilen = sizeof(ORDER_STATUS_DATA);
     olen = &ilen;

          if ( dLog )
          {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "* ProcessOrderStatus
Thread %d iTermId %d OrderStatusDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId,
&Term.pClientData[iTermId].OrderStatusDataPtr);
                    fclose(fp);
          }

#ifdef MULTI_TRANS
          if (( iRc = tpcall(qname[iTermId%10], (char
*)Term.pClientData[iTermId].OrderStatusDataPtr, ilen,
                    (char
**)&Term.pClientData[iTermId].OrderStatusDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
#else
     if (( iRc = tpcall("ORDERSTATUS", (char
*)Term.pClientData[iTermId].OrderStatusDataPtr, ilen,
                    (char
**)&Term.pClientData[iTermId].OrderStatusDataPtr, (long
*)olen, TPSIGRSTRT)) == -1)
#endif
          {
          TpRc = tperrno;
                    sprintf(buf, "ProcessOrderStatus
tpcall failed");
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_ORDER_STATUS_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    return;
     }

          if ( dLog )
          {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "** ProcessOrderStatus
Thread %d iTermId %d OrderStatusDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId,
&Term.pClientData[iTermId].OrderStatusDataPtr);
                    fclose(fp);
          }

     Term.pClientData[iTermId].OrderStatusData =
*Term.pClientData[iTermId].OrderStatusDataPtr;

     iRc =
Term.pClientData[iTermId].OrderStatusDataPtr->retval;
          iError =
Term.pClientData[iTermId].OrderStatusDataPtr->error;
     if ( iRc < 0 )
          {
                    if ( iError == ERR_TYPE_DEADLOCK )
                              ErrorMessage(pECB,
ERR_ORDER_STATUS_NOT_PROCESSED, ERR_TYPE_DEADLOCK,
NULL, iTermId, iSyncId);
                    else if (iError ==
ERR_NOSUCH_CUSTOMER)
                              ErrorMessage(pECB,
ERR_NOSUCH_CUSTOMER, ERR_TYPE_WEBDLL, NULL, iTermId,
iSyncId);
                    else
                              ErrorMessage(pECB,
ERR_ORDER_STATUS_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
     }
     else
          WriteZString(pECB,
MakeOrderStatusForm(iTermId, iSyncId, FALSE) );

#endif
     return;

}

/* FUNCTION: void
ProcessDeliveryForm(EXTENSION_CONTROL_BLOCK *pECB, int
iTermId, int iSyncId)
 *
 * PURPOSE:         This function gets and validates
the input data from the delivery form
 *                           filling in the required
input variables. It then calls the PostDeliveryInfo
 *                           Api, The client is then
informed that the transaction has been posted.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                  int

          iTermId   client browser terminal id
 *                                  int

          iSyncId   clinet browser sync id
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

 // Allocate the tpalloc structure for each transaction
//        This saves on some memory at the expense of
some CPU cycles.
static void ProcessDeliveryForm(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId)
{
          int               TpRc, iRc;
          char      szTmp[26];
          BOOL      bSuccess;
          long      ilen, *olen;
          char      buf[128];

#ifdef LOCAL_ALLOC
          DELIVERY_DATA
          *DeliveryDataPtr;   //Delivery Tuxedo Buffer
#endif

     if((iRc = ThrTpInit()) <0)
          {
                    // This is bad
                    sprintf(buf, "ProcessDelivery
Failed ThrTpInit: iRc = %d", iRc);
                    LogTuxError(0, buf);
                    ErrorMessage(pECB,
ERR_DELIVERY_MISSING_OCD_KEY, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    return;
          }
```

```
memset(&Term.pClientData[iTermId].DeliveryData, 0,
sizeof(DELIVERY_DATA));

          if ( !GetKeyValue(pECB->lpszQueryString,
"OCD*", szTmp, sizeof(szTmp)) )
          {
                    ErrorMessage(pECB,
ERR_DELIVERY_MISSING_OCD_KEY, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    return;
          }

          if ( !IsNumeric(szTmp) )
          {
                    ErrorMessage(pECB,
ERR_DELIVERY_CARRIER_INVALID, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    return;
          }

          Term.pClientData[iTermId].DeliveryData.o_carr
ier_id    = atoi(szTmp);

          if (
Term.pClientData[iTermId].DeliveryData.o_carrier_id >
10 ||
Term.pClientData[iTermId].DeliveryData.o_carrier_id < 1
)
          {
                    ErrorMessage(pECB,
ERR_DELIVERY_CARRIER_ID_RANGE, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    return;
          }

     Term.pClientData[iTermId].DeliveryData.w_id =
Term.pClientData[iTermId].w_id;


GetLocalTime(&Term.pClientData[iTermId].DeliveryData.qu
eue_time);

#ifdef LOCAL_ALLOC
          if ((DeliveryDataPtr = (DELIVERY_DATA
*)tpalloc("CARRAY", NULL, sizeof(DELIVERY_DATA))) ==
NULL)
          {
                    TpRc = tperrno;
                    sprintf(buf, "ProcessDelivery
Tpcalloc Failed");
                    LogTuxError(TpRc, buf);

          strcpy(Term.pClientData[iTermId].DeliveryData
.execution_status, "Delivery Post Failed");
                    bSuccess = FALSE;
                    WriteZString(pECB,
MakeDeliveryForm(iTermId, iSyncId, FALSE, bSuccess) );
                    tpfree((char *)DeliveryDataPtr);
                    return;
          }

     *DeliveryDataPtr =
Term.pClientData[iTermId].DeliveryData;

     ilen = sizeof(DELIVERY_DATA);
     olen = &ilen;

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "* ProcessDelivery
Thread %d iTermId %d DeliveryDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &DeliveryDataPtr);
                    fclose(fp);
           }

     if (( iRc = tpacall("DELIVERY", (char
*)DeliveryDataPtr, ilen, TPNOREPLY)) == -1)
           {
                    TpRc = tperrno;
                    sprintf(buf, "ProcessDelivery
Tpcalloc Failed");
                    LogTuxError(TpRc, buf);

          strcpy(Term.pClientData[iTermId].DeliveryData
.execution_status, "Delivery Post Failed");
                    bSuccess = FALSE;
                    WriteZString(pECB,
MakeDeliveryForm(iTermId, iSyncId, FALSE, bSuccess) );
                    tpfree((char *)DeliveryDataPtr);
                    return;
     }

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "** ProcessDelivery
Thread %d iTermId %d DeliveryDataPtr: %x TpRc = %d
\r\n",
                              GetCurrentThreadId(),
iTermId, &DeliveryDataPtr, TpRc);
                    fclose(fp);
           }

          tpfree((char *)DeliveryDataPtr);


#else // Not LOCAL_ALLOC

     *Term.pClientData[iTermId].DeliveryDataPtr =
Term.pClientData[iTermId].DeliveryData;

     ilen = sizeof(DELIVERY_DATA);
     olen = &ilen;

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "* ProcessDelivery
Thread %d iTermId %d TuxDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &Term.pClientData[iTermId].TuxDataPtr);
                    fclose(fp);
           }

     if (( iRc = tpacall("DELIVERY", (char
*)Term.pClientData[iTermId].DeliveryDataPtr, ilen,
TPNOREPLY)) == -1)
           {
                    TpRc = tperrno;
                    sprintf(buf, "ProcessDelivery
Tpcalloc Failed");
                    LogTuxError(TpRc, buf);

          strcpy(Term.pClientData[iTermId].DeliveryData
.execution_status, "Delivery Post Failed");
                    bSuccess = FALSE;
                    WriteZString(pECB,
MakeDeliveryForm(iTermId, iSyncId, FALSE, bSuccess) );
                    return;
     }

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "** ProcessDelivery
Thread %d iTermId %d DeliveryDataPtr: %x TpRc = %d
\r\n",
                              GetCurrentThreadId(),
iTermId, &Term.pClientData[iTermId].DeliveryDataPtr,
TpRc);
                    fclose(fp);
           }
#endif
          strcpy(Term.pClientData[iTermId].DeliveryData
.execution_status, "Delivery has been queued.");
          bSuccess = TRUE;

          WriteZString(pECB, MakeDeliveryForm(iTermId,
iSyncId, FALSE, bSuccess) );
     return;

}


/* FUNCTION: void
ProcessStockLevelForm(EXTENSION_CONTROL_BLOCK *pECB,
int iTermId, int iSyncId)
 *
 * PURPOSE:          This function gets and validates
the input data from the Stock Level
 *                           form filling in the
required input variables. It then calls the
 *                           SQLStockLevel
transaction, constructs the output form and writes it
```

```
 *                           back to client browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
         passed in structure pointer from inetsrv.
 *                                    int

         iTermId   client browser terminal id
 *                                    int

         iSyncId   client browser sync id
 *
 * RETURNS:                  None
 *
 * COMMENTS:        None
 *
 */

 // Allocate the tpalloc structure for each transaction
//        This saves on some memory at the expense of
some CPU cycles.
static void
ProcessStockLevelForm(EXTENSION_CONTROL_BLOCK *pECB,
int iTermId, int iSyncId)
{
          int                TpRc, iRc, iError;
          char      szTmp[26];
          long      ilen, *olen;
          char      buf[128];
#ifdef LOCAL_ALLOC
          STOCK_LEVEL_DATA    *StockLevelDataPtr;
          //Stock Level Tuxedo Buffer
#endif
           if((iRc = ThrTpInit()) <0)
           {
                     // This is bad
                    sprintf(buf, "ProcessStockLevel
Failed ThrTpInit: iRc = %d", iRc);
                    LogTuxError(0, buf);
                    ErrorMessage(pECB,
ERR_STOCKLEVEL_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    return;
           }

     memset(&Term.pClientData[iTermId].StockLevelData,
0, sizeof(STOCK_LEVEL_DATA));

     Term.pClientData[iTermId].StockLevelData.w_id =
Term.pClientData[iTermId].w_id;
     Term.pClientData[iTermId].StockLevelData.d_id =
Term.pClientData[iTermId].d_id;


     if ( !GetKeyValue(pECB->lpszQueryString, "TT*",
szTmp, sizeof(szTmp)) )
     {
          ErrorMessage(pECB,
ERR_STOCKLEVEL_MISSING_THRESHOLD_KEY, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
          return;
     }

     if ( !IsNumeric(szTmp) )
     {
          ErrorMessage(pECB,
ERR_STOCKLEVEL_THRESHOLD_INVALID, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
          return;
     }


Term.pClientData[iTermId].StockLevelData.thresh_hold =
atoi(szTmp);

     if (
Term.pClientData[iTermId].StockLevelData.thresh_hold >=
100 ||
Term.pClientData[iTermId].StockLevelData.thresh_hold <
0 )
     {
          ErrorMessage(pECB,
ERR_STOCKLEVEL_THRESHOLD_RANGE, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
          return;
     }

#ifdef LOCAL_ALLOC
          if ((StockLevelDataPtr = (STOCK_LEVEL_DATA
*)tpalloc("CARRAY", NULL, sizeof(STOCK_LEVEL_DATA))) ==
NULL)
          {
                    TpRc = tperrno;
                    sprintf(buf, "ProcessStockLevel
Tpcalloc Failed");
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_STOCKLEVEL_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    tpfree((char *)StockLevelDataPtr);
                    return;
          }

    *StockLevelDataPtr =
Term.pClientData[iTermId].StockLevelData;

     ilen = sizeof(STOCK_LEVEL_DATA);
     olen = &ilen;

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "* ProcessStockLevel
Thread %d iTermId %d StockLevelDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &StockLevelDataPtr);
                    fclose(fp);
           }

#ifdef MULTI_TRANS
     if (( iRc = tpcall(qname[iTermId%10], (char
*)StockLevelDataPtr, ilen,
                    (char **)&StockLevelDataPtr, (long
*) olen, TPSIGRSTRT)) == -1)
#else
          if (( iRc = tpcall("STOCKLEVEL", (char
*)StockLevelDataPtr, ilen,
                    (char **)&StockLevelDataPtr, (long
*) olen, TPSIGRSTRT)) == -1)
#endif
          {
         TpRc = tperrno;
                    sprintf(buf, "ProcessStockLevel
tpcall failed");
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_STOCKLEVEL_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
                    tpfree((char *)StockLevelDataPtr);
                    return;
     }

           if ( dLog )
           {
                    FILE *fp;

                    fp = fopen(szTpccLogPath, "ab");
                    fprintf(fp, "** ProcessStockLevel
Thread %d iTermId %d StockLevelDataPtr: %x \r\n",
                              GetCurrentThreadId(),
iTermId, &StockLevelDataPtr);
                    fclose(fp);
           }

     Term.pClientData[iTermId].StockLevelData =
*StockLevelDataPtr;

     iRc = StockLevelDataPtr->retval;
           iError = StockLevelDataPtr->error;

           tpfree((char *)StockLevelDataPtr);

           if ( iRc == 0 )
           {
                    if ( iError == ERR_TYPE_DEADLOCK )
                              ErrorMessage(pECB,
ERR_STOCKLEVEL_NOT_PROCESSED, ERR_TYPE_DEADLOCK, NULL,
iTermId, iSyncId);
                    else
                              ErrorMessage(pECB,
ERR_STOCKLEVEL_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
     }
     else
          WriteZString(pECB,
MakeStockLevelForm(iTermId, iSyncId, FALSE) );


#else // Not LOCAL_ALLOC

     *Term.pClientData[iTermId].StockLevelDataPtr=
Term.pClientData[iTermId].StockLevelData;

     ilen = sizeof(STOCK_LEVEL_DATA);
```

```
     olen = &ilen;

          if ( dLog )
          {
                   FILE *fp;

                   fp = fopen(szTpccLogPath, "ab");
                   fprintf(fp, "* ProcessStockLevel
Thread %d iTermId %d StockLevelDataPtr: %x \r\n",
                                GetCurrentThreadId(),
iTermId, &Term.pClientData[iTermId].StockLevelDataPtr);
                   fclose(fp);
          }
#ifdef MULTI_TRANS
     if (( iRc = tpcall(qname[iTermId%10], (char
*)Term.pClientData[iTermId].StockLevelDataPtr, ilen,
                    (char
**)&Term.pClientData[iTermId].StockLevelDataPtr, (long
*) olen, TPSIGRSTRT)) == -1)
#else
          if (( iRc = tpcall("STOCKLEVEL", (char
*)Term.pClientData[iTermId].StockLevelDataPtr, ilen,
                    (char
**)&Term.pClientData[iTermId].StockLevelDataPtr, (long
*) olen, TPSIGRSTRT)) == -1)
#endif
          {
         TpRc = tperrno;
                    sprintf(buf, "ProcessStockLevel
tpcall failed");
                    LogTuxError(TpRc, buf);
                    ErrorMessage(pECB,
ERR_STOCKLEVEL_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
         //tpfree((char *)StockLevelDataPtr);
                    return;
     }

          if ( dLog )
          {
                   FILE *fp;

                   fp = fopen(szTpccLogPath, "ab");
                   fprintf(fp, "** ProcessStockLevel
Thread %d iTermId %d StockLevelDataPtr: %x \r\n",
                                GetCurrentThreadId(),
iTermId, &Term.pClientData[iTermId].StockLevelDataPtr);
                   fclose(fp);
          }

     Term.pClientData[iTermId].StockLevelData =
*Term.pClientData[iTermId].StockLevelDataPtr;

     iRc = Term.pClientData[iTermId].StockLevelDataPtr-
>retval;
          iError =
Term.pClientData[iTermId].StockLevelDataPtr->error;
          if ( iRc == 0 )
          {
                    if ( iError == ERR_TYPE_DEADLOCK )
                              ErrorMessage(pECB,
ERR_STOCKLEVEL_NOT_PROCESSED, ERR_TYPE_DEADLOCK, NULL,
iTermId, iSyncId);
                    else
                              ErrorMessage(pECB,
ERR_STOCKLEVEL_NOT_PROCESSED, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
     }
     else
          WriteZString(pECB,
MakeStockLevelForm(iTermId, iSyncId, FALSE) );

#endif
          return;

}


/* FUNCTION: int GetNewOrderData(LPSTR lpszQueryString,
NEW_ORDER_DATA *pNewOrderData)
 *
 * PURPOSE:         This function extracts and
validates the new order form data from an http command
string.
 *
 * ARGUMENTS:       LPSTR
          lpszQueryString              client browser
http command string
 *                                     NEW_ORDER_DATA
          *pNewOrderData               pointer to new
order data structure
 *
 * RETURNS:                  int

                   error code indicating reason for
failure
 *                                    ERR_SUCCESS

                   new order input data successfully
parsed
 *
 *
 * COMMENTS:        None
 *
 */

static int GetNewOrderData(LPSTR lpszQueryString,
NEW_ORDER_DATA *pNewOrderData)
{
          char      szTmp[26];
          char      szKey[26];
          int                 i;
          short     items;
          BOOL      bCheck;

          if ( !GetKeyValue(lpszQueryString, "DID*",
szTmp, sizeof(szTmp)) )
                    return
ERR_NEWORDER_FORM_MISSING_DID;

          if ( !IsNumeric(szTmp) )
                    return
ERR_NEWORDER_DISTRICT_INVALID;

          pNewOrderData->d_id = atoi(szTmp);

          if ( !GetKeyValue(lpszQueryString, "CID*",
szTmp, sizeof(szTmp)) )
                    return ERR_NEWORDER_CUSTOMER_KEY;

          if ( !IsNumeric(szTmp) )
                    return
ERR_NEWORDER_CUSTOMER_INVALID;
          pNewOrderData->c_id = atoi(szTmp);

          bCheck = FALSE;
          for(i=0, items=0; i<15; i++)
          {
                    wsprintf(szKey, "IID%2.2d*", i);
                    if ( !GetKeyValue(lpszQueryString,
szKey, szTmp, sizeof(szTmp)) )
                              return
ERR_NEWORDER_MISSING_IID_KEY;
                    if ( szTmp[0] )
                    {
                              //if blank lines between
item ids
                              if ( bCheck )
                                        return
ERR_NEWORDER_ITEM_BLANK_LINES;
                              if ( !IsNumeric(szTmp) )
                                        return
ERR_NEWORDER_ITEMID_INVALID;
                              pNewOrderData-
>Ol[i].ol_i_id = atoi(szTmp);

                              wsprintf(szKey,
"SP%2.2d*", i);
                              if (
!GetKeyValue(lpszQueryString, szKey, szTmp,
sizeof(szTmp)) )
                                        return
ERR_NEWORDER_MISSING_SUPPW_KEY;
                              // ETW Fix for warehouse
out of range
                              if ( !IsNumeric(szTmp) )
                                        return
ERR_NEWORDER_SUPPW_INVALID;
                              if ( (short)atoi(szTmp) >
iMaxWareHouses )
                                        return
ERR_NEWORDER_SUPPW_RANGE;
                              pNewOrderData-
>Ol[i].ol_supply_w_id = (short)atoi(szTmp);

                              wsprintf(szKey,
"Qty%2.2d*", i);
                              if (
!GetKeyValue(lpszQueryString, szKey, szTmp,
sizeof(szTmp)) )
                                        return
ERR_NEWORDER_MISSING_QTY_KEY;

                              if ( !IsNumeric(szTmp) )
```

```
                                        return
ERR_NEWORDER_QTY_INVALID;

                                pNewOrderData-
>Ol[i].ol_quantity = atoi(szTmp);
                                items++;

                                if ( pNewOrderData-
>Ol[i].ol_i_id >= 1000000 || pNewOrderData-
>Ol[i].ol_i_id < 1 )
                                        return
ERR_NEWORDER_ITEMID_RANGE;
                                if ( pNewOrderData-
>Ol[i].ol_quantity >= 100 || pNewOrderData-
>Ol[i].ol_quantity < 1 )
                                        return
ERR_NEWORDER_QTY_RANGE;
                        }
                        else
                        {
                                wsprintf(szKey,
"SP%2.2d*", i);
                                if (
!GetKeyValue(lpszQueryString, szKey, szTmp,
sizeof(szTmp)) )
                                        return
ERR_NEWORDER_MISSING_QTY_KEY;

                                if ( szTmp[0] )
                                        return
ERR_NEWORDER_SUPPW_WITHOUT_ITEMID;

                                wsprintf(szKey,
"Qty%2.2d*", i);
                                if (
!GetKeyValue(lpszQueryString, szKey, szTmp,
sizeof(szTmp)) )
                                        return
ERR_NEWORDER_MISSING_QTY_KEY;

                                if ( szTmp[0] )
                                        return
ERR_NEWORDER_QTY_WITHOUT_ITEMID;

                                bCheck = TRUE;
                        }
                }
                if ( items == 0 )
                        return
ERR_NEWORDER_NOITEMS_ENTERED;

                pNewOrderData->o_ol_cnt = items;

                return ERR_SUCCESS;
}

/* FUNCTION: int GetPaymentData(LPSTR lpszQueryString,
PAYMENT_DATA *pPaymentData)
 *
 * PURPOSE:         This function extracts and
validates the payment form data from an http command
string.
 *
 * ARGUMENTS:       LPSTR
                lpszQueryString                 client browser
http command string
 *                                              PAYMENT_DATA
                *pPaymentData                   pointer to
payment data structure
 *
 * RETURNS:                                     int

                        error code indicating reason for
failure
 *
                ERR_SUCCESS
                                all input data
successfully parsed
 *
 * COMMENTS:        None
 *
 */

static int GetPaymentData(LPSTR lpszQueryString,
PAYMENT_DATA *pPaymentData)
{
                char      szTmp[26];
                char      *ptr;

                if ( !GetKeyValue(lpszQueryString, "DID*",
szTmp, sizeof(szTmp)) )
                        return ERR_PAYMENT_MISSING_DID_KEY;
                if ( !IsNumeric(szTmp) )
                        return
ERR_PAYMENT_DISTRICT_INVALID;
                pPaymentData->d_id = atoi(szTmp);

                if ( !GetKeyValue(lpszQueryString, "CID*",
szTmp, sizeof(szTmp)) )
                        return ERR_PAYMENT_MISSING_CID_KEY;

                if ( szTmp[0] && !IsNumeric(szTmp) )
                        return
ERR_PAYMENT_CUSTOMER_INVALID;


                pPaymentData->c_id = atoi(szTmp);

                if ( szTmp[0] == 0 )
                {
                        if ( !GetKeyValue(lpszQueryString,
"CLT*", szTmp, sizeof(szTmp)) )
                                return
ERR_PAYMENT_MISSING_CLT;
                        _strupr( szTmp );

                        strcpy(pPaymentData->c_last,
szTmp);
                        if ( strlen(pPaymentData->c_last) >
16 )
                                return
ERR_PAYMENT_LAST_NAME_TO_LONG;
                }
                else
                {
                        if ( !GetKeyValue(lpszQueryString,
"CLT*", szTmp, sizeof(szTmp)) )
                                return
ERR_PAYMENT_MISSING_CLT_KEY;
                        if ( szTmp[0] )
                                return
ERR_PAYMENT_CID_AND_CLT;
                }

                if ( !GetKeyValue(lpszQueryString, "CDI*",
szTmp, sizeof(szTmp)) )
                        return ERR_PAYMENT_MISSING_CDI_KEY;
                if ( !IsNumeric(szTmp) )
                        return ERR_PAYMENT_CDI_INVALID;
                pPaymentData->c_d_id = atoi(szTmp);

                if ( !GetKeyValue(lpszQueryString, "CWI*",
szTmp, sizeof(szTmp)) )
                        return ERR_PAYMENT_MISSING_CWI_KEY;

                if ( !IsNumeric(szTmp) )
                        return ERR_PAYMENT_CWI_INVALID;

                pPaymentData->c_w_id = atoi(szTmp);


                if ( !GetKeyValue(lpszQueryString, "HAM*",
szTmp, sizeof(szTmp)) )
                        return ERR_PAYMENT_MISSING_HAM_KEY;

                ptr = szTmp;

                while( *ptr )
                {
                        if ( *ptr == '.' )
                        {
                                ptr++;
                                if ( !*ptr )
                                        break;
                                if ( *ptr < '0' || *ptr >
'9' )
                                        return
ERR_PAYMENT_HAM_INVALID;
                                ptr++;
                                if ( !*ptr )
                                        break;
                                if ( *ptr < '0' || *ptr >
'9' )
                                        return
ERR_PAYMENT_HAM_INVALID;
                                if ( !*ptr )
                                        return
ERR_PAYMENT_HAM_INVALID;
                        }
                        else if ( *ptr < '0' || *ptr > '9'
)
                                        return
ERR_PAYMENT_HAM_INVALID;
                        ptr++;
                }
```

```
          pPaymentData->h_amount = atof(szTmp);
          if ( pPaymentData->h_amount >= 10000.00 ||
pPaymentData->h_amount < 0 )
                    return ERR_PAYMENT_HAM_RANGE;

          return ERR_SUCCESS;
}

/* FUNCTION: int GetOrderStatusData(LPSTR
lpszQueryString, ORDER_STATUS_DATA *pOrderStatusData)
 *
 * PURPOSE:         This function extracts and
validates the payment form data from an http command
string.
 *
 * ARGUMENTS:       LPSTR
          lpszQueryString               client browser
http command string
 *
          ORDER_STATUS_DATA   *pOrderStatusData
          pointer to order status data structure
 *
 * RETURNS:                   int

                    error code indicating reason for
failure
 *                                    ERR_SUCCESS

                    successfully parsed all required
input data
 *
 * COMMENTS:        None
 *
 */
static int GetOrderStatusData(LPSTR lpszQueryString,
ORDER_STATUS_DATA *pOrderStatusData)
{
          char      szTmp[26];

          if ( !GetKeyValue(lpszQueryString, "DID*",
szTmp, sizeof(szTmp)) )
                    return
ERR_ORDERSTATUS_MISSING_DID_KEY;
          if ( !IsNumeric(szTmp) )
                    return ERR_ORDERSTATUS_DID_INVALID;
          pOrderStatusData->d_id = atoi(szTmp);

          if ( !GetKeyValue(lpszQueryString, "CID*",
szTmp, sizeof(szTmp)) )
                    return
ERR_ORDERSTATUS_MISSING_CID_KEY;

          if ( szTmp[0] == 0 )
          {
                    pOrderStatusData->c_id = 0;
                    if ( !GetKeyValue(lpszQueryString,
"CLT*", szTmp, sizeof(szTmp)) )
                              return
ERR_ORDERSTATUS_MISSING_CLT_KEY;
                    _strupr( szTmp );
                    strcpy(pOrderStatusData->c_last,
szTmp);
                    if ( strlen(pOrderStatusData-
>c_last) > 16 )
                              return
ERR_ORDERSTATUS_CLT_RANGE;
          }
          else
          {
                    if ( !IsNumeric(szTmp) )
                              return
ERR_ORDERSTATUS_CID_INVALID;
                    pOrderStatusData->c_id =
atoi(szTmp);
                    if ( !GetKeyValue(lpszQueryString,
"CLT*", szTmp, sizeof(szTmp)) )
                              return
ERR_ORDERSTATUS_MISSING_CLT_KEY;
                    if ( szTmp[0] )
                              return
ERR_ORDERSTATUS_CID_AND_CLT;
          }

          return ERR_SUCCESS;
}

/* FUNCTION: BOOL ReadRegistrySettings(void)
 *
 * PURPOSE:         This function reads the NT registry
for startup parameters. There parameters are
 *                          under the TPCC key.
 *
 * ARGUMENTS:   None
 *
 * RETURNS:                   None
 *
 * COMMENTS:        This function also sets up required
operation variables to their default value
 *                                    so if registry
is not setup the default values will be used.
 *
 */

static BOOL ReadRegistrySettings(void)
{
          HKEY      hKey;
          DWORD     size;
          DWORD     type;
          char      szTmp[256];

          bLog                          = FALSE;
          dLog                          = FALSE;
          iMaxWareHouses      = 500;
          iThreads            = 5;
          iDelayMs            = 100;
          iDeadlockRetry      = (short)3;
          strcpy(szTpccLogPath, "tpcclog.");
          strcpy(szErrorLogPath, "tpccerr.");


          if ( RegOpenKeyEx(HKEY_LOCAL_MACHINE,
"SOFTWARE\\Microsoft\\TPCC", 0, KEY_READ, &hKey) !=
ERROR_SUCCESS )
                    return TRUE;
          size = sizeof(szTmp);

          if ( RegQueryValueEx(hKey, "PATH", 0, &type,
szTmp, &size) == ERROR_SUCCESS )
          {
                    strcpy(szTpccLogPath, szTmp);
                    strcat(szTpccLogPath, "tpcclog.");
                    strcpy(szErrorLogPath, szTmp);
                    strcat(szErrorLogPath, "tpccerr.");
          }

          size = sizeof(szTmp);
          if ( RegQueryValueEx(hKey, "LOG", 0, &type,
szTmp, &size) == ERROR_SUCCESS )
          {
                    if ( !stricmp(szTmp, "ON") )
                              bLog = TRUE;
          }

          size = sizeof(szTmp);
          if ( RegQueryValueEx(hKey, "DEBUG", 0, &type,
szTmp, &size) == ERROR_SUCCESS )
          {
                    if ( !stricmp(szTmp, "ON") )
                              dLog = TRUE;
          }

          size = sizeof(szTmp);
          if ( RegQueryValueEx(hKey,
"MaximumWarehouses", 0, &type, szTmp, &size) ==
ERROR_SUCCESS )
          {
                    iMaxWareHouses = atoi(szTmp);
                    if ( iMaxWareHouses == 0 )
                              iMaxWareHouses = 500;
          }

                    size = sizeof(szTmp);
          if ( RegQueryValueEx(hKey, "BackoffDelay", 0,
&type, szTmp, &size) == ERROR_SUCCESS )
                    iDelayMs = atoi(szTmp);
                    if ( !iDelayMs )
                              iDelayMs = 100;

          size = sizeof(szTmp);
          if ( RegQueryValueEx(hKey, "DeadlockRetry",
0, &type, szTmp, &size) == ERROR_SUCCESS )
                    iDeadlockRetry =
(short)atoi(szTmp);
                    if ( !iDeadlockRetry )
                              iDeadlockRetry =
(short)3;

          size = sizeof(szTmp);
          if ( RegQueryValueEx(hKey, "MaxConnections",
0, &type, szTmp, &size) == ERROR_SUCCESS )
                    iMaxConnections =
(short)atoi(szTmp);
                    if ( !iMaxConnections )
                              iMaxConnections =
(short)25;
```

```
          RegCloseKey(hKey);

          return FALSE;

}



/* FUNCTION: BOOL IsNumeric(char *ptr)
 *
 * PURPOSE:         This function determines if a
string is numeric. It fails if any characters other
 *                          than numeric and null
terminator are present.
 *
 * ARGUMENTS:       char                        *ptr
          pointer to string to check.
 *
 * RETURNS:                 BOOL      FALSE     if
string is not all numeric
 *
          TRUE      if string contains only numeric
characters i.e. '0' - '9'
 *
 * COMMENTS:        None
 *
 */

static BOOL IsNumeric(char *ptr)
{
          if ( *ptr == 0 )
                    return FALSE;

          while( *ptr && isdigit(*ptr) )
                    ptr++;
          return ( !*ptr );
}


/* FUNCTION: void FormatHTMLString(char *szBuff, int
iLen, char *szStr)
 *
 * PURPOSE:         This function Handles translation
of HTML specific character field data
 *                          when an HTML output form
is generated.
 *
 * ARGUMENTS:       char      *szBuff   Returned string
information
 *                                    char
          *szStr    input string to be formatted.
 *                                    int
          iLen      Length of returned string
 *
 * RETURNS:                 none
 *
 * COMMENTS:        The length paramter is the absolute
length of the returned string in
 *                                    HTML
characters. For example the input string > would be
returned as
 *                                    &gt; which
would be counted as 1 character.If the number of input
 *                                    characters is
less than the iLen parameter spaces are appended to
 *                                    the end of the
string to ensure that at least iLen characters are
 *                                    returned in the
szBuff parameter.
 *
 */

static void FormatHTMLString(char *szBuff, char *szStr,
int iLen)
{
          while( iLen && *szStr )
          {
                    switch( *szStr )
                    {
                              case '>':
                                        *szBuff++ =
'&';
                                        *szBuff++ =
'g';
                                        *szBuff++ =
't';
                                        *szBuff++ =
';';
                                        szStr++;
                                        break;
                              case '<':
                                        *szBuff++ =
'&';
                                        *szBuff++ =
'l';
                                        *szBuff++ =
't';
                                        *szBuff++ =
';';
                                        szStr++;
                                        break;
                              case '&':
                                        *szBuff++ =
'&';
                                        *szBuff++ =
'a';
                                        *szBuff++ =
'm';
                                        *szBuff++ =
'p';
                                        *szBuff++ =
';';
                                        szStr++;
                                        break;
                              case '\"':
                                        *szBuff++ =
'&';
                                        *szBuff++ =
'q';
                                        *szBuff++ =
'u';
                                        *szBuff++ =
'o';
                                        *szBuff++ =
't';
                                        *szBuff++ =
';';
                                        szStr++;
                                        break;
                              default:
                                        *szBuff++ =
*szStr++;
                                        break;
                    }
                    iLen--;
          }
          while( iLen-- )
                    *szBuff++ = ' ';

          *szBuff = 0;

          return;
}


static int ThrTpInit()
{

     static int num_tpinits=0;
     static int x=1;
     static int once=0;
     static CRITICAL_SECTION     TpCriticalSection;
           int lasterr, iRc, TpRc;
           int retry = 0;
           BOOL Success = FALSE;

     if(!TlsGetValue(TLSIsTpInitedKey))
     {
                    if (!once)
                    {

          InitializeCriticalSection(&TpCriticalSection)
;
                              once=1;
                    }

                    if ( dLog )
                    {
                              FILE *fp;

                              fp = fopen(szTpccLogPath,
"ab");
                              fprintf(fp, "* In
ThrTpInit Thread %d * \r\n", GetCurrentThreadId());
                              fclose(fp);
                    }

                    while ( retry < TP_MAX_RETRIES )
                    {

          EnterCriticalSection(&TpCriticalSection);
                              if(tpinf == NULL)
                              {
                                        if ((tpinf = (
TPINIT *)tpalloc("TPINIT", NULL, sizeof(TPINIT))) ==
NULL)
```

```
                                {

        LeaveCriticalSection(&TpCriticalSection);
                                                TpRc
= tperrno;
                                                {

        FILE *fp;


        fp = fopen(szErrorLogPath, "ab");

        fprintf(fp, ">>>> ThrTpInit:%d : tpalloc of
tpinit failed: %d : %s\r\n",

                GetCurrentThreadId(), TpRc,
tpstrerror(TpRc));

        fclose(fp);
                                                }

        retry++;

        continue;
                                }
                            tpinf-
>flags|=TPMULTICONTEXTS;
                            }

                            if (retry == 0)

        itoa(++num_tpinits, tpinf->cltname, 10);

// Do the TPINIT
                            iRc = tpinit(tpinf);
                            TpRc = tperrno;

                            // check tpalloc() ?
                            if (iRc < 0)
                            {

        LeaveCriticalSection(&TpCriticalSection);
                                    retry++;
                                    lasterr =
GetLastError();
/*                                  TpRc = tperrno;
                                    {
                                            FILE
*fp;

                                                fp =
fopen(szErrorLogPath, "ab");

        fprintf(fp, ">>>> ThrTpInit:%d : tpinit
failed: %d %s :try # %d\r\n",

        GetCurrentThreadId(), iRc, tpstrerror(TpRc),
retry);

        fclose(fp);
                                    }
*/
                            }
                            else
                            {
                                    Success = TRUE;

        LeaveCriticalSection(&TpCriticalSection);
                                    break;
                            }

                            Sleep(50); // Relinquish
thread timeslice
                } // retry the tpinit if it failed
the first time

                if ( Success == FALSE )
                {
                        {
                                    char ebuf[128];


                                    sprintf(ebuf,
">>>> ThrTpInit %d : Cannot tpinit after %d tries iRc =
%d LastErr = %d \r\n",

        GetCurrentThreadId(), TP_MAX_RETRIES, iRc,
lasterr);

        LogTuxError(TpRc, ebuf);
                            }
                            return -1;
                }
                if ( Success == TRUE )
                {
                        if ( retry > 0 )
                        {
                                    char ebuf[128];


                                    sprintf(ebuf,
">>>> ThrTpInit %d : Cannot tpinit after %d tries iRc =
%d LastErr = %d \r\n",

        GetCurrentThreadId(), TP_MAX_RETRIES, iRc,
lasterr);

                                    sprintf(ebuf,
"* ThrTpInit Thread %d Success retry count %d with
LastErr = %d * \r\n",

        GetCurrentThreadId(), retry, lasterr);

        LogTuxError(TpRc, ebuf);
                            }

                            if ( (
iRc=TlsSetValue(TLSIsTpInitedKey,&x)) == 0)
                            {
                                    {
                                                FILE
*fp;

                                                fp =
fopen(szErrorLogPath, "ab");

        fprintf(fp, ">>>> ThrTpInit %d : TlsSetValue
Failed iRc: %d \r\n",

        GetCurrentThreadId(), iRc);

        fclose(fp);
                                    }
                            }
                    }
        }
        else
        {
                    if ( dLog )
                    {
                            FILE *fp;

                            fp = fopen(szTpccLogPath,
"ab");
                            fprintf(fp, "* ThrTpInit
Thread %d already tpinited * \r\n",
GetCurrentThreadId());
                            fclose(fp);
                    }
        }
    return 0;
}
```

Multi_Trans.c

```
/*      FILE:               TRANSACTIONS.C
 *
 *      Based on: Microsoft TPC-C Kit Ver. 3.00.000
 *
 *                                  Copyright
Microsoft, 1996
 *                                  Copyright
Performance Tuning Corporation, 1997
 *
 *      PURPOSE:  New Order Tuxedo Server.
 *      Author:             Philip Durr
 *
 *      philipdu@Microsoft.com
 *
 *  MODIFIED        Changed for modularity and to allow
for the Tuxedo TM
 *
 *  Author:                 Edward Whalen
 *                          Performance
Tuning Corporation
 *
 *      ewhalen@perftuning.com
 */

#include <windows.h>
#include <process.h>
#include <stdio.h>
#include <stdarg.h>
#include <malloc.h>
```

Appendix A-Application Code

```
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <sys\timeb.h>
#include <io.h>

#define DBNTWIN32
#include <sqlfront.h>
#include <sqldb.h>

#include "trans.h"
                    //tpckit transaction header
contains definations of structures specific to TPC-C
#include "httpext.h"
                    //ISAPI DLL information header

#include "tpcc.h"
                    //this dlls specific structure,
value e.t. header.

#include <tmenv.h>
#include <xa.h>
#include <atmi.h>

void NEWORDER();
void PAYMENT();
void ORDERSTATUS();
void STOCKLEVEL();

extern BOOL         bLog;
BOOL      bFlush;                       //Flush
delivery log info when written.
BOOL      verbose                       = FALSE;
BOOL      bError                        = FALSE;

extern int                    iThreads;
extern int                    iMaxWareHouses;
extern int                    iDelayMs;
short     iMaxConnections     = (short)1;
short     iDeadlockRetry      = (short)3;

DBPROCESS *pdbproc;

static char
          szServer[32];
          //SQL server name
static char
          szDatabase[32];
          //tpcc database name
static char
          szUser[32];
                    //user name
static char
          szPassword[32];
          //user password
int spId;


TERM Term;

static char         szTpccLogPath[256]; //path to html
log file if logging turned on in registry.
static char szErrorLogPath[256];        //path to error
log file.

static    CRITICAL_SECTION
          CriticalSection;
static    CRITICAL_SECTION
          ErrorLogCriticalSection;
static    EXTENSION_CONTROL_BLOCK       *gpECB = NULL;
static    int
                    bTpccExit;          //exit delivery
disconnect loop as dll exiting.

void TMLog();
extern BOOL SQLInit();
extern void UtilStrCpy();
extern void UtilStrCpy();
extern BOOL SQLOpenConnection();
extern BOOL SQLCloseConnection();
extern BOOL SQLDetectDeadlock();

/* FUNCTION: tpsvrinit ( int argc, char *argv[])
 *
 * PURPOSE:         Initialize the Server to Database
connection.
 *
 * RETURNS:                   int       0
          Success
 *                                                  -1
          Failure
 *
 *
 * COMMENTS:        None
 *
 */

int tpsvrinit ( int argc, char *argv[] )
{

          if ( GetParameters(argc, argv) )
          {
                    PrintParameters();
                    return -1;
          }

          if ( verbose )
                    TMLog("TPSVRINIT: Q: Server %s,
Database %s, User %s, Password %s, Flush %d.",
                              szServer, szDatabase,
szUser, szPassword, bFlush);


          if ( ! SQLInit() )
          {
                    TMLog( "Q: SQLInit Failed" );
                    return -1;
          }

          if ( SQLOpenConnection ( NULL, 0, 0,
&pdbproc, szServer, szDatabase, szUser, szPassword,
szDatabase, &spId))
          {
                    TMLog ( "Q: SQLOpenConnection
Failed" );
                    dbexit();
                    return -1;
          }
          return 0;
}

/* FUNCTION: tpsvrdone ( void )
 *
 * PURPOSE:         Initialize the Server to Database
connection.
 *
 * RETURNS:                   int       0
          Success
 *                                                  -1
          Failure
 *
 *
 * COMMENTS:        None
 *
 */

void tpsvrdone ( void )
{
          SQLCloseConnection( NULL, pdbproc);
          dbexit();
}

void T1 (TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}


void T2 (TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
```

```
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}

void T3 (TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}

void T4 (TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}

void T5 (TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}

void T6 (TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}

void T7 (TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}

void T8 (TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}

void T9 (TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}

void T10(TPSVCINFO *rqst)
{
          int size=rqst->len;
          switch(size)
          {
          case sizeof(NEW_ORDER_DATA):
                    NEWORDER(rqst);
                    break;
          case sizeof(PAYMENT_DATA):
                    PAYMENT(rqst);
                    break;
          case sizeof(ORDER_STATUS_DATA):
                    ORDERSTATUS(rqst);
                    break;
          case sizeof(STOCK_LEVEL_DATA):
                    STOCKLEVEL(rqst);
                    break;
          default:
                    exit;
          }
}
```

```
/* FUNCTION: NEWORDER ( TPSVCINFO *rqst )
 *
 * PURPOSE:         Process a New Order request.
 *
 * RETURNS:                    int       0
         Success
 *                                                -1
         Failure
 *
 *
 * COMMENTS:        None
 *
 */

void NEWORDER ( TPSVCINFO *rqst )
{
          PECBINFO pECBInfo = dbgetuserdata(pdbproc);
          int size = rqst->len;
          NEW_ORDER_DATA      *NewOrderDataPtr;


          NewOrderDataPtr = (NEW_ORDER_DATA *) rqst-
>data;

          if (verbose )
          {
                    TMLog(" NEWORDER: w_id %d ",
NewOrderDataPtr->w_id);
                    TMLog(" NEWORDER: d_id %d ",
NewOrderDataPtr->d_id);
                    TMLog(" NEWORDER: c_id %d ",
NewOrderDataPtr->c_id);
          }

          bError = FALSE;

          NewOrderDataPtr->retval = SQLNewOrder( NULL,
0, 0, pdbproc, NewOrderDataPtr, iDeadlockRetry);

          if (bError == TRUE)
                    NewOrderDataPtr->retval = -1;

          if (verbose )
                    TMLog(" NEWORDER: Return Value %d",
NewOrderDataPtr->retval);


          tpreturn( TPSUCCESS, 0, rqst->data, size, 0);
}

/* FUNCTION: int SQLNewOrder(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId, int iTermId, int
iSyncId, DBPROCESS *dbproc, NEW_ORDER_DATA  *pNewOrder,
short deadlock_retry)
 *
 * PURPOSE:         This function handles the new order
transaction.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
                             passed in structure
pointer from inetsrv.
 *                                    int

          iTermId                        terminal id of
browser
 *                                    int

          iSyncId                        sync id of
browser
 *                                    DBPROCESS
                             *dbproc
          connection db process id
 *                                    NEW_ORDER_DATA
                             *pNewOrder
          pointer to new order structure for
input/output data
 *                                    short
                                       deadlock_retry
          retry count if deadlocked
 *
 * RETURNS:                   int       TRUE
          transaction committed
 *                                                 FALSE
          item number not valid
 *                                                 -1
                    deadlock max retry reached
 *
 *
 * COMMENTS:        None
 *
 */

static int SQLNewOrder(EXTENSION_CONTROL_BLOCK
          *pECB, int iTermId, int iSyncId, DBPROCESS
*dbproc, NEW_ORDER_DATA  *pNewOrder, short
deadlock_retry)
{
          RETCODE                        rc;
          int                                  i;
          DBINT                          commit_flag;
          int
          tryit;
          char                           printbuf[25];
          char                           tmpbuf[30];
          DBDATETIME                     datetime;
          BYTE                           *pData;
          PECBINFO            pEcbInfo;

          if ( (pEcbInfo =
(PECBINFO)dbgetuserdata(dbproc)) )
          {
                    pEcbInfo->pECB = pECB;
                    pEcbInfo->bFailed = FALSE;
                    pEcbInfo->iTermId = iTermId;
                    pEcbInfo->iSyncId = iSyncId;
          }

          pNewOrder->num_deadlocks = 0;

          strcpy(tmpbuf, "tpcc_neworder");

          for (tryit=0; tryit < deadlock_retry;
tryit++)
          {
                    if (dbrpcinit(dbproc, tmpbuf, 0) ==
SUCCEED)
                    {
                                       dbrpcparam(dbproc, NULL,
0, SQLINT2, -1, -1, (BYTE *) &pNewOrder->w_id);
                                       dbrpcparam(dbproc, NULL,
0, SQLINT1, -1, -1, (BYTE *) &pNewOrder->d_id);
                                       dbrpcparam(dbproc, NULL,
0, SQLINT4, -1, -1, (BYTE *) &pNewOrder->c_id);
                                       dbrpcparam(dbproc, NULL,
0, SQLINT1, -1, -1, (BYTE *) &pNewOrder->o_ol_cnt);
//                                     dbrpcparam(dbproc, NULL,
0, SQLINT1, -1, -1, (BYTE *) &pNewOrder->o_all_local);

                                       pNewOrder->o_all_local =
1;
                                       for (i = 0; i <
pNewOrder->o_ol_cnt; i++)
                                       {
                                                 if ( pNewOrder-
>o_all_local && pNewOrder->Ol[i].ol_supply_w_id !=
pNewOrder->w_id )

          pNewOrder->o_all_local = 0;
                                       }
                                       dbrpcparam(dbproc, NULL,
0, SQLINT1, -1, -1, (BYTE *) &pNewOrder->o_all_local);

                                       for (i = 0; i <
pNewOrder->o_ol_cnt; i++)
                                       {

          dbrpcparam(dbproc, NULL, 0, SQLINT4, -1, -1,
(BYTE *) &pNewOrder->Ol[i].ol_i_id);

          dbrpcparam(dbproc, NULL, 0, SQLINT2, -1, -1,
(BYTE *) &pNewOrder->Ol[i].ol_supply_w_id);

          dbrpcparam(dbproc, NULL, 0, SQLINT2, -1, -1,
(BYTE *) &pNewOrder->Ol[i].ol_quantity);
                                       }

                                       if (dbrpcexec(dbproc) ==
SUCCEED)
                                       {
                                                 pNewOrder-
>total_amount=0;

                                                 // Get resutls
from order line
                                                 for (i = 0;
i<pNewOrder->o_ol_cnt; i++)
                                                 {
                                                           if
(((rc = dbresults(dbproc)) != NO_MORE_RESULTS) && (rc
!= FAIL))
                                                           {

          if (DBROWS(dbproc) && (dbnumcols(dbproc) ==
5))
```

```
          {

                    while (dbnextrow(dbproc) !=
NO_MORE_ROWS)

                    {

                              if(pData=dbdata(dbproc,
1))


          UtilStrCpy(pNewOrder->Ol[i].ol_i_name, pData,
dbdatlen(dbproc, 1));

                              if(pData=dbdata(dbproc,
2))

                                        pNewOrder-
>Ol[i].ol_stock = (*(DBSMALLINT *) pData);

                              if(pData=dbdata(dbproc,
3))


          UtilStrCpy(pNewOrder->Ol[i].ol_brand_generic,
pData, dbdatlen(dbproc, 3));

                              if(pData=dbdata(dbproc,
4))


          dbconvert(dbproc, SQLNUMERIC, pData,
dbdatlen(dbproc,4), SQLFLT8, (BYTE *)&pNewOrder-
>Ol[i].ol_i_price, 8);



                              if(pData=dbdata(dbproc,
5))



          dbconvert(dbproc, SQLNUMERIC, pData,
dbdatlen(dbproc,5), SQLFLT8, (BYTE *)&pNewOrder-
>Ol[i].ol_amount, 8);









                              pNewOrder->total_amount =
pNewOrder->total_amount + pNewOrder->Ol[i].ol_amount;

                    }

          }
                                                  }
                                        }
                                        while (((rc =
dbresults(dbproc)) != NO_MORE_RESULTS) && (rc != FAIL))
                                        {
                                                  if
(DBROWS(dbproc) && (dbnumcols(dbproc) == 8))
                                                  {

          while (((rc = dbnextrow(dbproc)) !=
NO_MORE_ROWS) && (rc != FAIL))

          {

                    if(pData=dbdata(dbproc, 1))




                              dbconvert(dbproc,
SQLNUMERIC, pData, dbdatlen(dbproc,1), SQLFLT8, (BYTE
*)&pNewOrder->w_tax, 8);









                    if(pData=dbdata(dbproc, 2))
                              dbconvert(dbproc,
SQLNUMERIC, pData, dbdatlen(dbproc,2), SQLFLT8, (BYTE
*)&pNewOrder->d_tax, 8);









                    if(pData=dbdata(dbproc, 3))

                              pNewOrder->o_id =
(*(DBINT *) pData);

                    if(pData=dbdata(dbproc, 4))

                              UtilStrCpy(pNewOrder-
>c_last, pData, dbdatlen(dbproc, 4));

                    if(pData=dbdata(dbproc, 5))

                              dbconvert(dbproc,
SQLNUMERIC, pData, dbdatlen(dbproc,5), SQLFLT8, (BYTE
*)&pNewOrder->c_discount, 8);









                    if(pData=dbdata(dbproc, 6))

                              UtilStrCpy(pNewOrder-
>c_credit, pData, dbdatlen(dbproc, 6));

                    if(pData=dbdata(dbproc, 7))

                    {

                              datetime = *((DBDATETIME
*) pData);

                              dbdatecrack(dbproc,
&pNewOrder->o_entry_d, &datetime);

                    }

                    if(pData=dbdata(dbproc,
8))commit_flag = (*(DBTINYINT *) pData);

          }
                                                  }
                              }
                              }
                    }
                    if (SQLDetectDeadlock(dbproc))
                    {
                              pNewOrder-
>num_deadlocks++;

          sprintf(printbuf,"deadlock: retry:
%d",pNewOrder->num_deadlocks);

          Sleep(DEADLOCKWAIT*tryit);
                    }
                    else
                    {
                              if (commit_flag == 1)
                              {
                                        pNewOrder-
>total_amount = pNewOrder->total_amount * ((1 +
pNewOrder->w_tax + pNewOrder->d_tax) * (1 - pNewOrder-
>c_discount));

          strcpy(pNewOrder-
>execution_status,"Transaction commited.");
                                        return TRUE;
                              }
                              else
                              {

          strcpy(pNewOrder->execution_status,"Item
number is not valid.");
                                        pNewOrder-
>error=ERR_BAD_ITEM_ID;
                                        return FALSE;
```

```
                        }
                }
        }

        // If we reached here, it means we quit after
MAX_RETRY deadlocks
        strcpy(pNewOrder->execution_status,"Hit
deadlock max.  ");
        pNewOrder->error=ERR_TYPE_DEADLOCK;
        if ( verbose )
                TMLog(" NEWORDER: SQLNewOrder Max
Deadlocks %d", tryit);

        return -1;          //        "deadlock max
retry reached!"
}


void PAYMENT ( TPSVCINFO *rqst )
{
        PECBINFO pECBInfo = dbgetuserdata(pdbproc);
        PAYMENT_DATA *PaymentDataPtr;

        int size = rqst->len;



        PaymentDataPtr = (PAYMENT_DATA *) rqst->data;

        if (verbose )
        {
                TMLog(" PAYMENT: w_id %d ",
PaymentDataPtr->w_id);
                TMLog(" PAYMENT: d_id %d ",
PaymentDataPtr->d_id);
        }

        bError = FALSE;

        PaymentDataPtr->retval = SQLPayment( NULL, 0,
0, pdbproc, PaymentDataPtr, iDeadlockRetry);

        if (bError == TRUE)
                PaymentDataPtr->retval = -1;

        if (verbose )
                TMLog(" PAYMENT: Return Value %d",
PaymentDataPtr->retval);


        tpreturn( TPSUCCESS, 0, rqst->data, size, 0);
}

/* FUNCTION: int SQLPayment(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId, DBPROCESS *dbproc,
PAYMENT_DATA *pPayment, short deadlock_retry)
 *
 * PURPOSE:         This function handles the payment
transaction.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
                            passed in structure
pointer from inetsrv.
 *                                  int

        iTermId                     terminal id of
browser
 *                                  int

        iSyncId                     sync id of
browser
 *                                  DBPROCESS
                            *dbproc
        connection db process id
 *                                  PAYMENT_DATA
                            *pPayment
        pointer to payment input/output data
structure
 *                                  short
                                    deadlock_retry
        deadlock retry count
 *
 * RETURNS:                 int       TRUE
        success
 *                                          -1
                            max deadlocked reached
 *
 * COMMENTS:        None
 *
 */
```

```
static int SQLPayment(EXTENSION_CONTROL_BLOCK *pECB,
int iTermId, int iSyncId, DBPROCESS *dbproc,
PAYMENT_DATA *pPayment, short deadlock_retry)
{
        RETCODE             rc;
        int                         tryit;
        char                printbuf[26];
        DBDATETIME          datetime;
        BYTE                *pData;
        PECBINFO  pEcbInfo;

        if ( (pEcbInfo =
(PECBINFO)dbgetuserdata(dbproc)) )
        {
                pEcbInfo->pECB = pECB;
                pEcbInfo->bFailed = FALSE;
                pEcbInfo->iTermId = iTermId;
                pEcbInfo->iSyncId = iSyncId;
        }

        pPayment->num_deadlocks = 0;

        for (tryit=0; tryit < deadlock_retry;
tryit++)
        {
                if (dbrpcinit(dbproc,
"tpcc_payment", 0) == SUCCEED)
                {
                        dbrpcparam(dbproc, NULL,
0, SQLINT2, -1, -1, (BYTE *) &pPayment->w_id);
                        dbrpcparam(dbproc, NULL,
0, SQLINT2, -1, -1, (BYTE *) &pPayment->c_w_id);
                        dbrpcparam(dbproc, NULL,
0, SQLFLT8, -1, -1, (BYTE *) &pPayment->h_amount);
                        dbrpcparam(dbproc, NULL,
0, SQLINT1, -1, -1, (BYTE *) &pPayment->d_id);
                        dbrpcparam(dbproc, NULL,
0, SQLINT1, -1, -1, (BYTE *) &pPayment->c_d_id);
                        dbrpcparam(dbproc, NULL,
0, SQLINT4, -1, -1, (BYTE *) &pPayment->c_id);
                        if (pPayment->c_id == 0)
                        {

        dbrpcparam(dbproc, NULL, 0, SQLCHAR, -1,
strlen(pPayment->c_last), pPayment->c_last);
                        }
                }
                if (dbrpcexec(dbproc) == SUCCEED)
                {
                        while (((rc =
dbresults(dbproc)) != NO_MORE_RESULTS) && (rc != FAIL))
                        {
                                if
(DBROWS(dbproc) && (dbnumcols(dbproc) == 27))
                                {
                                        while
(((rc = dbnextrow(dbproc)) != NO_MORE_ROWS) && (rc !=
FAIL))
                                        {

        if(pData=dbdata(dbproc, 1))

                pPayment->c_id = *((DBINT *)
pData);

        if(pData=dbdata(dbproc, 2))

                UtilStrCpy(pPayment->c_last, pData,
dbdatlen(dbproc, 2));

        if(pData=dbdata(dbproc, 3))

        {

                datetime = *((DBDATETIME *) pData);

                dbdatecrack(dbproc, &pPayment-
>h_date, &datetime);

        }

        if(pData=dbdata(dbproc, 4))

                UtilStrCpy(pPayment->w_street_1,
pData, dbdatlen(dbproc, 4));

        if(pData=dbdata(dbproc, 5))

                UtilStrCpy(pPayment->w_street_2,
pData, dbdatlen(dbproc, 5));

        if(pData=dbdata(dbproc, 6))
```

```
                    UtilStrCpy(pPayment->w_city, pData,
dbdatlen(dbproc, 6));

          if(pData=dbdata(dbproc, 7))

                    UtilStrCpy(pPayment->w_state,
pData, dbdatlen(dbproc, 7));

          if(pData=dbdata(dbproc, 8))

                    UtilStrCpy(pPayment->w_zip, pData,
dbdatlen(dbproc, 8));

          if(pData=dbdata(dbproc, 9))

                    UtilStrCpy(pPayment->d_street_1,
pData, dbdatlen(dbproc, 9));

          if(pData=dbdata(dbproc, 10))

                    UtilStrCpy(pPayment->d_street_2,
pData, dbdatlen(dbproc, 10));

          if(pData=dbdata(dbproc, 11))

                    UtilStrCpy(pPayment->d_city, pData,
dbdatlen(dbproc, 11));

          if(pData=dbdata(dbproc, 12))

                    UtilStrCpy(pPayment->d_state,
pData, dbdatlen(dbproc, 12));

          if(pData=dbdata(dbproc, 13))

                    UtilStrCpy(pPayment->d_zip, pData,
dbdatlen(dbproc, 13));

          if(pData=dbdata(dbproc, 14))

                    UtilStrCpy(pPayment->c_first,
pData, dbdatlen(dbproc, 14));

          if(pData=dbdata(dbproc, 15))

                    UtilStrCpy(pPayment->c_middle,
pData, dbdatlen(dbproc, 15));

          if(pData=dbdata(dbproc, 16))

                    UtilStrCpy(pPayment->c_street_1,
pData, dbdatlen(dbproc, 16));

          if(pData=dbdata(dbproc, 17))

                    UtilStrCpy(pPayment->c_street_2,
pData, dbdatlen(dbproc, 17));

          if(pData=dbdata(dbproc, 18))

                    UtilStrCpy(pPayment->c_city, pData,
dbdatlen(dbproc, 18));

          if(pData=dbdata(dbproc, 19))

                    UtilStrCpy(pPayment->c_state,
pData, dbdatlen(dbproc, 19));

          if(pData=dbdata(dbproc, 20))

                    UtilStrCpy(pPayment->c_zip, pData,
dbdatlen(dbproc, 20));

          if(pData=dbdata(dbproc, 21))

                    UtilStrCpy(pPayment->c_phone,
pData, dbdatlen(dbproc, 21));

          if(pData=dbdata(dbproc, 22))

          {

                    datetime = *((DBDATETIME *) pData);

                    dbdatecrack(dbproc, &pPayment-
>c_since, &datetime);

          }

          if(pData=dbdata(dbproc, 23))

                    UtilStrCpy(pPayment->c_credit,
pData, dbdatlen(dbproc, 23));

          if(pData=dbdata(dbproc, 24))

                    dbconvert(dbproc, SQLNUMERIC,
pData, dbdatlen(dbproc,24), SQLFLT8, (BYTE *)&pPayment-
>c_credit_lim, 8);

          if(pData=dbdata(dbproc, 25))

                    dbconvert(dbproc, SQLNUMERIC,
pData, dbdatlen(dbproc,25), SQLFLT8, (BYTE *)&pPayment-
>c_discount, 8);

          if(pData=dbdata(dbproc, 26))

                    dbconvert(dbproc, SQLNUMERIC,
pData, dbdatlen(dbproc,26), SQLFLT8, (BYTE *)&pPayment-
>c_balance, 8);

          if(pData=dbdata(dbproc, 27))

                    UtilStrCpy(pPayment->c_data, pData,
dbdatlen(dbproc, 27));
                                                  }
                                        }
                              }
                    }
                    if (SQLDetectDeadlock(dbproc))
                    {
                              pPayment-
>num_deadlocks++;

          sprintf(printbuf,"deadlock: retry:
%d",pPayment->num_deadlocks);

          Sleep(DEADLOCKWAIT*tryit);
                    }
                    else
                    {
                              if ( pPayment->c_id == 0
)
                              {

          strcpy(pPayment->execution_status,"Invalid
Customer id,name.");
                                        pPayment-
>error=ERR_NOSUCH_CUSTOMER;
                                        TMLog("
PAYMENT: No such customer ");
                                        return 0;
                              }
                              else

          strcpy(pPayment-
>execution_status,"Transaction commited.");
                              return TRUE;
                    }
          }
          // If we reached here, it means we quit after
MAX_RETRY deadlocks
          strcpy(pPayment->execution_status,"Hit
deadlock max.  ");
          pPayment->error=ERR_TYPE_DEADLOCK;
          return -1; //"deadlock max retry reached!"
}


/* FUNCTION: STOCKLEVEL ( TPSVCINFO *rqst )
 *
 * PURPOSE:          Process a Stock Level request.
 *
 * RETURNS:                    int       0
          Success
 *                                                   -1
          Failure
 *
 *
 * COMMENTS:         None
 *
 */

void STOCKLEVEL ( TPSVCINFO *rqst )
{
          PECBINFO pECBInfo = dbgetuserdata(pdbproc);
          STOCK_LEVEL_DATA *StockLevelDataPtr;

          int size = rqst->len;
```

```
          StockLevelDataPtr = (STOCK_LEVEL_DATA *)
rqst->data;

          if (verbose )
          {
                    TMLog(" STOCKLEVEL: w_id %d ",
StockLevelDataPtr->w_id);
                    TMLog(" STOCKLEVEL: d_id %d ",
StockLevelDataPtr->d_id);
                    TMLog(" STOCKLEVEL: c_id %d ",
StockLevelDataPtr->thresh_hold);
          }

          bError = FALSE;

          StockLevelDataPtr->retval = SQLStockLevel(
NULL, 0, 0, pdbproc, StockLevelDataPtr,
iDeadlockRetry);


          if (bError == TRUE)
                    StockLevelDataPtr->retval = -1;

          if ( verbose )
                    TMLog(" STOCKLEVEL: Return Value
%d", StockLevelDataPtr->retval);

          tpreturn( TPSUCCESS, 0, rqst->data, size, 0);
}

/* FUNCTION: SQLStockLevel(EXTENSION_CONTROL_BLOCK
          *pECB, int iTermId, int iSyncId, DBPROCESS
*dbproc, STOCK_LEVEL_DATA *pStockLevel, short
deadlock_retry)
 *
 * PURPOSE:          This function handles the stock
level transaction.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK       *pECB
                              passed in structure
pointer from inetsrv.
 *                                    int

          iTermId                   terminal id of
browser
 *                                    int

          iSyncId                   sync id of
browser
 *                                    DBPROCESS
                              *dbproc
          connection db process id
 *
          STOCK_LEVEL_DATA          *pStockLevel
          stock level input / output data structure
 *                                    short
                                    deadlock_retry
          retry count if deadlocked
 *
 * RETURNS:                  BOOL      FALSE
          if successfull
 *
          TRUE              if deadlocked
 *
 * COMMENTS:         None
 *
 */

static BOOL SQLStockLevel(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId, DBPROCESS *dbproc,
STOCK_LEVEL_DATA *pStockLevel, short deadlock_retry)
{
          int                       tryit;
          RETCODE             rc;
          char                printbuf[25];
          BYTE                *pData;
          PECBINFO  pEcbInfo;

          //update pECB and bFailed flag
          if ( (pEcbInfo =
(PECBINFO)dbgetuserdata(dbproc)) )
          {
                    pEcbInfo->pECB = pECB;
                    pEcbInfo->bFailed = FALSE;
                    pEcbInfo->iTermId = iTermId;
                    pEcbInfo->iSyncId = iSyncId;
          }

          pStockLevel->num_deadlocks = 0;

          for (tryit=0; tryit < deadlock_retry;
tryit++)
          {
                    if (dbrpcinit(dbproc,
"tpcc_stocklevel", 0) == SUCCEED)
                    {
                              dbrpcparam(dbproc, NULL,
0, SQLINT2, -1, -1, (BYTE *) &pStockLevel->w_id);
                              dbrpcparam(dbproc, NULL,
0, SQLINT1, -1, -1, (BYTE *) &pStockLevel->d_id);
                              dbrpcparam(dbproc, NULL,
0, SQLINT2, -1, -1, (BYTE *) &pStockLevel-
>thresh_hold);

                              if (dbrpcexec(dbproc) ==
SUCCEED)
                              {
                                        while (((rc =
dbresults(dbproc)) != NO_MORE_RESULTS) && (rc != FAIL))
                                        {
                                                  if
(DBROWS(dbproc))
                                                  {

          while (((rc = dbnextrow(dbproc)) !=
NO_MORE_ROWS) && (rc != FAIL))

          {

                    if(pData=dbdata(dbproc, 1))

                              pStockLevel->low_stock =
*((long *) pData);

          }
                                                  }
                                        }
                              }
                    }
                    if (SQLDetectDeadlock(dbproc))
                    {
                              pStockLevel-
>num_deadlocks++;

          sprintf(printbuf,"deadlock: retry:
%d",pStockLevel->num_deadlocks);
                              Sleep(10 * tryit);
                    }
                    else
                    {
                              strcpy(pStockLevel-
>execution_status, "Transaction commited.");
                              return TRUE;
                    }
          }

          // If we reached here, it means we quit after
MAX_RETRY deadlocks
          strcpy(pStockLevel->execution_status, "Hit
deadlock max. ");
          pStockLevel->error=ERR_TYPE_DEADLOCK;
          return -1;
}


/* FUNCTION: ORDERSTATUS ( TPSVCINFO *rqst )
 *
 * PURPOSE:          Process an Order Status request.
 *
 * RETURNS:                  int       0
          Success
 *                                             -1
          Failure
 *
 *
 * COMMENTS:         None
 *
 */

void ORDERSTATUS ( TPSVCINFO *rqst )
{
          PECBINFO pECBInfo = dbgetuserdata(pdbproc);
          ORDER_STATUS_DATA   *OrderStatusDataPtr;

          int size = rqst->len;


          OrderStatusDataPtr = (ORDER_STATUS_DATA *)
rqst->data;

          if (verbose )
          {
                    TMLog(" ORDERSTATUS: w_id %d ",
OrderStatusDataPtr->w_id);
                    TMLog(" ORDERSTATUS: d_id %d ",
OrderStatusDataPtr->d_id);
```

```
                    TMLog(" ORDERSTATUS: c_id %d ",
OrderStatusDataPtr->c_id);
          }

          bError = FALSE;

          OrderStatusDataPtr->retval = SQLOrderStatus(
NULL, 0, 0, pdbproc, OrderStatusDataPtr,
iDeadlockRetry);

          if (bError == TRUE)
                    OrderStatusDataPtr->retval = -1;

          if ( verbose )
                    TMLog(" ORDERSTATUS: Return Value
%d", OrderStatusDataPtr->retval);


          tpreturn( TPSUCCESS, 0, rqst->data, size, 0);
}

/* FUNCTION: int SQLOrderStatus(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId, DBPROCESS *dbproc,
ORDER_STATUS_DATA *pOrderStatus, short deadlock_retry)
 *
 * PURPOSE:         This function processes the Order
Status transaction.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
                              passed in structure
pointer from inetsrv.
 *                                    int

          iTermId                       terminal id of
browser
 *                                    int

          iSyncId                       sync id of
browser
 *                                    DBPROCESS
                              *dbproc
          connection db process id
 *
          ORDER_STATUS_DATA             *pOrderStatus
          pointer to Order Status data input/output
structure
 *                                    short
                                       deadlock_retry
          deadlock retry count
 *
 * RETURNS:                  int       -1
          max deadlock reached
 *                                                  0
                    No orders found for customer
 *                                                  1
                    Transaction successfull
 *
 * COMMENTS:        None
 *
 */

static int SQLOrderStatus(EXTENSION_CONTROL_BLOCK
*pECB, int iTermId, int iSyncId, DBPROCESS *dbproc,
ORDER_STATUS_DATA *pOrderStatus, short deadlock_retry)
{
          RETCODE             rc;
          int                           tryit;
          int                           i;
          char                printbuf[25];
          DBDATETIME          datetime;
          BYTE                *pData;
          PECBINFO  pEcbInfo;

          if ( (pEcbInfo =
(PECBINFO)dbgetuserdata(dbproc)) )
          {
                    pEcbInfo->pECB = pECB;
                    pEcbInfo->bFailed = FALSE;
                    pEcbInfo->iTermId = iTermId;
                    pEcbInfo->iSyncId = iSyncId;
          }

          pOrderStatus->num_deadlocks = 0;

          for (tryit=0; tryit < deadlock_retry;
tryit++)
          {
                    if (dbrpcinit(dbproc,
"tpcc_orderstatus", 0) == SUCCEED)
                    {
                              dbrpcparam(dbproc, NULL,
0, SQLINT2, -1, -1, (BYTE *) &pOrderStatus->w_id);
                              dbrpcparam(dbproc, NULL,
0, SQLINT1, -1, -1, (BYTE *) &pOrderStatus->d_id);
                              dbrpcparam(dbproc, NULL,
0, SQLINT4, -1, -1, (BYTE *) &pOrderStatus->c_id);
                              if (pOrderStatus->c_id ==
0)
                              {


          dbrpcparam(dbproc, NULL, 0, SQLCHAR, -1,
strlen(pOrderStatus->c_last), pOrderStatus->c_last);
                              }
                    }
                    if (dbrpcexec(dbproc) == SUCCEED)
                    {
                              while (((rc =
dbresults(dbproc)) != NO_MORE_RESULTS) && (rc != FAIL))

                              {
                                        if
(DBROWS(dbproc) && (dbnumcols(dbproc) == 5))
                                        {
                                                  i=0;
                                                  while
(((rc = dbnextrow(dbproc)) != NO_MORE_ROWS) && (rc !=
FAIL))
                                                  {

          if(pData=dbdata(dbproc, 1))

                    pOrderStatus-
>OlOrderStatusData[i].ol_supply_w_id = (*(DBSMALLINT *)
pData);

          if(pData=dbdata(dbproc, 2))

                    pOrderStatus-
>OlOrderStatusData[i].ol_i_id = (*(DBINT *) pData);

          if(pData=dbdata(dbproc, 3))

                    pOrderStatus-
>OlOrderStatusData[i].ol_quantity = (*(DBSMALLINT *)
pData);

          if(pData=dbdata(dbproc, 4))

                    dbconvert(dbproc, SQLNUMERIC,
pData, dbdatlen(dbproc,4), SQLFLT8, (BYTE
*)&pOrderStatus->OlOrderStatusData[i].ol_amount, 8);

          if(pData=dbdata(dbproc, 5))

          {

                    datetime = *((DBDATETIME *) pData);

                    dbdatecrack(dbproc, &pOrderStatus-
>OlOrderStatusData[i].ol_delivery_d, &datetime);

          }

          i++;
                                                  }

          pOrderStatus->o_ol_cnt = i;
                                        }

                                        else if
(DBROWS(dbproc) && (dbnumcols(dbproc) == 8))

                                        {
                                                  while
(((rc = dbnextrow(dbproc)) != NO_MORE_ROWS) && (rc !=
FAIL))
                                                  {

          if(pData=dbdata(dbproc, 1))

                    pOrderStatus->c_id = (*(DBINT *)
pData);

          if(pData=dbdata(dbproc, 2))

                    UtilStrCpy(pOrderStatus->c_last,
pData, dbdatlen(dbproc,2));

          if(pData=dbdata(dbproc, 3))

                    UtilStrCpy(pOrderStatus->c_first,
pData, dbdatlen(dbproc,3));

          if(pData=dbdata(dbproc, 4))
```

```
                    UtilStrCpy(pOrderStatus->c_middle,
pData, dbdatlen(dbproc, 4));

          if(pData=dbdata(dbproc, 5))

          {

                    datetime = *((DBDATETIME *) pData);

                    dbdatecrack(dbproc, &pOrderStatus-
>o_entry_d, &datetime);

          }

          if(pData=dbdata(dbproc, 6))

                    pOrderStatus->o_carrier_id =
(*(DBSMALLINT *) pData);

          if(pData=dbdata(dbproc, 7))

                    dbconvert(dbproc, SQLNUMERIC,
pData, dbdatlen(dbproc,7), SQLFLT8, (BYTE
*)&pOrderStatus->c_balance, 8);

          if(pData=dbdata(dbproc, 8))

                    pOrderStatus->o_id = (*(DBINT *)
pData);
                                                  }
                                        }
                                        if (i==0)

          return 0; //"No orders found for customer"
                              }
                    }
                    if (SQLDetectDeadlock(dbproc))
                    {
                              pOrderStatus-
>num_deadlocks++;

          sprintf(printbuf,"deadlock: retry:
%d",pOrderStatus->num_deadlocks);

          Sleep(DEADLOCKWAIT*tryit);
                    }
                    else
                    {
                              if (pOrderStatus->c_id ==
0 && pOrderStatus->c_last[0] == 0)
                              {

          strcpy(pOrderStatus-
>execution_status,"Invalid Customer id,name.");
                    //                  pOrderStatus-
>error=ERR_NOSUCH_CUSTOMER;
                                        TMLog("
ORDERSTATUS: No such customer ");
                              }
                              else

          strcpy(pOrderStatus-
>execution_status,"Transaction commited.");
                              return 1;
                    }
          }
          // If we reached here, it means we quit after
MAX_RETRY deadlocks
          strcpy(pOrderStatus->execution_status,"Hit
deadlock max.  ");
          //pOrderStatus->error=ERR_TYPE_DEADLOCK;
          return -1; //"deadlock max retry reached!"
}

/*
 *        Common Code for all Servers
 */

 /* FUNCTION: BOOL SQLInit()
 *
 * PURPOSE:         This function initializes SQL
Server for later use.
 *
 *
 * RETURNS:                  BOOL      FALSE     if
successfull
 *
          TRUE      if an error occurs and connection
cannot be established.
 *
 * COMMENTS:        None
 *
 */
BOOL SQLInit ()
{

          dbinit();

          if ( dbgetmaxprocs() < iMaxConnections )
          {
                    if ( dbsetmaxprocs(iMaxConnections)
== FAIL )
                    {
                              //set for fail error
message when HttpExtensionProc() is called because
                              //at this point we don't
have a pECB so no way to show error message.
                              iMaxConnections = -1;
                    }
          }

          // install error and message handlers
          dbmsghandle((DBMSGHANDLE_PROC)msg_handler);
          dberrhandle((DBERRHANDLE_PROC)err_handler);

          return TRUE;
}


/* FUNCTION: BOOL
SQLOpenConnection(EXTENSION_CONTROL_BLOCK
          *pECB, int iTermId, int iSyncId, DBPROCESS
**dbproc, char *server, char *database, char *user,
char *password, char *app, int *spid, long *pack_size)
 *
 * PURPOSE:         This function opens the sql
connection for use.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                  int
                              iTermId
          terminal id of browser
 *                                  int
                              iSyncId             sync
id of browser
 *                                  DBPROCESS
          **dbproc  pointer to returned DBPROCESS
 *                                  char
                    *server             SQL server name
 *                                  char
                    *database SQL server database
 *                                  char
                    *user               user name
 *                                  char
                    *password user password
 *                                  char
                    *app                pointer to
returned application array
 *                                  int
                              *spid
          pointer to returned spid
 *                                  long
                    *pack_size          pointer to
returned default pack size
 *
 * RETURNS:                  BOOL      FALSE     if
successfull
 *
          TRUE      if an error occurs
 *
 * COMMENTS:        None
 *
 */


#ifdef USE_ODBC
          static BOOL
SQLOpenConnection(EXTENSION_CONTROL_BLOCK *pECB, int
iTermId, int iSyncId, DBPROCESS **dbproc, char *server,
char *database, char *user, char *password, char *app,
int *spid, long *pack_size)
          {

                    RETCODE   rc;
                    char      buffer[30];

                    *dbproc = (DBPROCESS
*)malloc(sizeof(DBPROCESS));
                    if ( !*dbproc )
                              return TRUE;

                    //set pECB data into dbproc
                    (*dbproc)->bDeadlock = FALSE;
                    (*dbproc)->bFailed = FALSE;
```

```
                    (*dbproc)->pECB = pECB;
                    (*dbproc)->iTermId = iTermId;
                    (*dbproc)->iSyncId = iSyncId;


                    if ( SQLAllocConnect(henv,
&(*dbproc)->hdbc) == SQL_ERROR )
                              return TRUE;

                    if ( SQLSetConnectOption((*dbproc)-
>hdbc, SQL_PACKET_SIZE, pack_size) == SQL_ERROR )
                              return TRUE;

                    rc = SQLConnect((*dbproc)->hdbc,
server, SQL_NTS, user, SQL_NTS, password, SQL_NTS);
                    if (rc != SQL_SUCCESS && rc !=
SQL_SUCCESS_WITH_INFO)
                              return TRUE;
                    rc = SQLAllocStmt((*dbproc)->hdbc,
&(*dbproc)->hstmt);
                    if (rc == SQL_ERROR)
                              return TRUE;

                    sprintf(buffer,"use %s", Client-
>database);

                    rc = SQLExecDirect((*dbproc)-
>hstmt, buffer, SQL_NTS);
                    if (rc != SQL_SUCCESS && rc !=
SQL_SUCCESS_WITH_INFO)
                              return TRUE;

                    SQLFreeStmt((*dbproc)->hstmt,
SQL_CLOSE);
                    sprintf(buffer,"set nocount on");
                    rc = SQLExecDirect((*dbproc)-
>hstmt, buffer, SQL_NTS);
                    if (rc != SQL_SUCCESS && rc !=
SQL_SUCCESS_WITH_INFO)
                              return TRUE;
                    SQLFreeStmt((*dbproc)->hstmt,
SQL_CLOSE);

                    sprintf(buffer,"select @@spid");

                    rc = SQLExecDirect((*dbproc)-
>hstmt, buffer, SQL_NTS);
                    if (rc != SQL_SUCCESS && rc !=
SQL_SUCCESS_WITH_INFO)
                              return TRUE;

                    if ( SQLBindCol((*dbproc)->hstmt,
1, SQL_C_SSHORT, &(*dbproc)->spid, 0, NULL) ==
SQL_ERROR )
                              return TRUE;


                    if ( SQLFetch((*dbproc)->hstmt) ==
SQL_ERROR )
                              return TRUE;

                    SQLFreeStmt((*dbproc)->hstmt,
SQL_CLOSE);

                    return FALSE;
          }

#else

          static BOOL
SQLOpenConnection(EXTENSION_CONTROL_BLOCK *pECB, int
iTermId, int iSyncId, DBPROCESS **dbproc, char *server,
char *database, char *user, char *password, char *app,
int *spid)
          {
                    LOGINREC  *login;
                    PECBINFO  pEcbInfo;

                    //set local msg proc for login
record
                    //attach pECB record

                    //this is necessary as dblib
provides no way to pass user data in a login structure.
So until
                    //there is an allocated dbproc we
need to use a static which means that the login attempt
must
                    //be serialized.

                    gpECB = pECB;

                    login = dblogin();

                    if ( !*user )
                              DBSETLUSER(login, "sa");
                    else
                              DBSETLUSER(login, user);

                    DBSETLPWD(login,  password);
                    DBSETLHOST(login, app);

//                  Do not set the packet size.  Use
the size set up in SQL Server.
                    DBSETLPACKET(login, (unsigned
short)DEFCLPACKSIZE);

//                  This can potentially cut down on
data conversion
                    DBSETLVERSION(login, DBVER60);

                    if ((*dbproc = dbopen(login, server
)) == NULL)
                              return TRUE;

                    //set pECB data into dbproc
                    pEcbInfo =
(PECBINFO)malloc(sizeof(ECBINFO));
                    pEcbInfo->bDeadlock = FALSE;
                    pEcbInfo->pECB = pECB;
                    pEcbInfo->iTermId = iTermId;
                    pEcbInfo->iSyncId = iSyncId;
                    dbsetuserdata(*dbproc, pEcbInfo);

                    // Use the the right database
                    dbuse(*dbproc, database);

                    dbcmd(*dbproc, "select @@spid");

                    dbsqlexec(*dbproc);
                    while (dbresults(*dbproc) !=
NO_MORE_RESULTS)
                    {
                              dbbind(*dbproc, 1,
SMALLBIND, (DBINT) 0, (BYTE *) spid);
                              while (dbnextrow(*dbproc)
!= NO_MORE_ROWS)
                              ;
                    }
                    dbcmd(*dbproc, "set nocount on");

                    dbsqlexec(*dbproc);
                    while (dbresults(*dbproc) !=
NO_MORE_RESULTS)
                    {
                              while (dbnextrow(*dbproc)
!= NO_MORE_ROWS)
                              ;
                    }

                    //rollback transaction on abort
                    dbcmd(*dbproc, "set XACT_ABORT
ON");

                    dbsqlexec(*dbproc);
                    while (dbresults(*dbproc) !=
NO_MORE_RESULTS)
                    {
                              while (dbnextrow(*dbproc)
!= NO_MORE_ROWS)
                              ;
                    }

                    return FALSE;
          }

#endif

/* FUNCTION: BOOL
SQLCloseConnection(EXTENSION_CONTROL_BLOCK *pECB,
DBPROCESS *dbproc)
 *
 * PURPOSE:          This function closes the sql
connection.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK       *pECB
          passed in structure pointer from inetsrv.
 *                                   DBPROCESS
          *dbproc   pointer to DBPROCESS
 *
 * RETURNS:                   BOOL      FALSE     if
successfull
 *
          TRUE      if an error occurs
 *
 * COMMENTS:         None
 *
```

```
 */

#ifdef USE_ODBC
          static BOOL
SQLCloseConnection(EXTENSION_CONTROL_BLOCK *pECB,
DBPROCESS *dbproc)
          {
                    if ( dbproc )
                    {
                              SQLFreeStmt(dbproc-
>hstmt, SQL_DROP);
                              SQLDisconnect(dbproc-
>hdbc);
                              SQLFreeConnect(dbproc-
>hdbc);
                              free(dbproc);
                              dbproc = NULL;
                    }
                    return FALSE;
          }
#else
          static BOOL
SQLCloseConnection(EXTENSION_CONTROL_BLOCK *pECB,
DBPROCESS *dbproc)
          {
                    if (dbclose(dbproc) == FAIL)
                              return TRUE;
                    return FALSE;
          }
#endif

/* FUNCTION: BOOL SQLDetectDeadlock(DBPROCESS *dbproc)
 *
 * PURPOSE:         This function checks to see if a
sql server deadlock condition exists.
 *
 * ARGUMENTS:       DBPROCESS
          *dbproc                         connection db
process id to check
 *
 * RETURNS:                  BOOL      FALSE
          no deadlock detected
 *
          TRUE                deadlock condition exists
 *
 * COMMENTS:        None
 *
 */

BOOL SQLDetectDeadlock(DBPROCESS *dbproc)
{
          PECBINFO  pEcbInfo;

          if ( (pEcbInfo =
(PECBINFO)dbgetuserdata(dbproc)) )
          {
                    if ( pEcbInfo->bDeadlock )
                    {
                              pEcbInfo->bDeadlock =
FALSE;
                              return TRUE;
                    }
          }
          return FALSE;
}

// Lifted from HP FDR since they did such a nice job
void TMLog( char *format, ... )
{
          va_list args;
          char buf[4096];
          int len;
          va_start( args, format );
          _strtime( buf );
          strcat( buf," ");
          len = strlen( buf );
          (void)_vsnprintf( buf+ len, sizeof( buf) -
len - 1, format, args);
          buf[sizeof( buf )- 1]= '\0';
          va_end( args );
          userlog( buf );
}

/* FUNCTION: void UtilStrCpy(char *pDest, char *pSrc,
int n)
 *
 * PURPOSE:         This function copies n characters
from string pSrc to pDst and places a
 *                           null character at the end
of the destination string.
 *
 * ARGUMENTS:       char
          *pDest    destination string pointer
 *                                  char
                    *pSrc     source string pointer
 *                                  int
                              n
          number of characters to copy
 *
 * RETURNS:                  None
 *
 * COMMENTS:        Unlike strncpy this function
ensures that the result string is
 *                                  always null
terminated.
 *
 */

static void UtilStrCpy(char *pDest, char *pSrc, int n)
{
    strncpy(pDest, pSrc, n);
    pDest[n] = '\0';

          return;
}

/* FUNCTION: int err_handler(DBPROCESS *dbproc, int
severity, int dberr, int oserr, char *dberrstr, char
*oserrstr)
 *
 * PURPOSE:         This function handles DB-Library
errors
 *
 * ARGUMENTS:       DBPROCESS           *dbproc
                    DBPROCESS id pointer
 *                                  int
                              severity
          severity of error
 *                                  int
                              dberr
          error id
 *                                  int
                              oserr
          operating system specific error code
 *                                  char
                    *dberrstr           printable error
description of dberr
 *                                  char
                    *oserrstr           printable error
description of oserr
 *
 * RETURNS:                  int
                    INT_CONTINUE        continue if
error is SQLETIME else INT_CANCEL action
 *
 * COMMENTS:        None
 *
 */


int err_handler(DBPROCESS *dbproc, int severity, int
dberr, int oserr, char *dberrstr, char *oserrstr)
{
          PECBINFO
          pEcbInfo;
          EXTENSION_CONTROL_BLOCK       *pECB;
          FILE
          *fp;
          SYSTEMTIME
          systemTime;
          char
          szTmp[256];
          int
                    iTermId;
          int
                    iSyncId;

          pEcbInfo = NULL;

          if ((dbproc == NULL) || (DBDEAD(dbproc)))
          {
                    TMLog("DBPROC is invalid");
                    return INT_CANCEL;
          }

          if ( !(pEcbInfo =
(PECBINFO)dbgetuserdata(dbproc)) )
          {
                    pECB = gpECB;
                    iTermId = 0;
                    iSyncId = 0;
          }
          else
          {
                    pECB = pEcbInfo->pECB;
                    iTermId = pEcbInfo->iTermId;
```

```
                iSyncId = pEcbInfo->iSyncId;
        }

        if ( pEcbInfo && pEcbInfo->bFailed )
        {
                bError == FALSE;
                return INT_CANCEL;
        }

        if ( oserr != DBNOERR )
        {
                TMLog("DBLIB Error %s", oserrstr);
                if ( pEcbInfo )
                {
                        pEcbInfo->bFailed = TRUE;
                        bError = TRUE;
                }

                GetLocalTime(&systemTime);
                fp = fopen(szErrorLogPath, "ab");

                sprintf(szTmp, "ErrorHandler:
DBLIB(%d): %s", oserr, oserrstr);

                TMLog("%2.2d/%2.2d/%2.2d
%2.2d:%2.2d:%2.2d\r\n\r\n%s\r\n\r\n",
                        systemTime.wYear,
systemTime.wMonth, systemTime.wDay,
                        systemTime.wHour,
systemTime.wMinute, systemTime.wSecond,
                        szTmp);

                fclose(fp);
        }

        return INT_CANCEL;
}


/* FUNCTION: int msg_handler(DBPROCESS *dbproc, DBINT
msgno, int msgstate, int severity, char *msgtext)
 *
 * PURPOSE:         This function handles DB-Library
SQL Server error messages
 *
 * ARGUMENTS:       DBPROCESS           *dbproc
                    DBPROCESS id pointer
 *                                      DBINT
                    msgno
        message number
 *                                      int
                            msgstate
        message state
 *                                      int
                            severity
        message severity
 *                                      char
                    *msgtext            printable
message description
 *
 * RETURNS:                 int
                    INT_CONTINUE        continue if
error is SQLETIME else INT_CANCEL action
 *
                            INT_CANCEL
        cancel operation
 *
 * COMMENTS:        This function also sets the dead
lock dbproc variable if necessary.
 *
 */

int msg_handler(DBPROCESS *dbproc, DBINT msgno, int
msgstate, int severity, char *msgtext)
{
        PECBINFO
        pEcbInfo;
        EXTENSION_CONTROL_BLOCK       *pECB;
        //FILE
        *fp;
        SYSTEMTIME
        systemTime;
        char
        szTmp[256];
        int
                iTermId;
        int
                iSyncId;

        if ( !(pEcbInfo =
(PECBINFO)dbgetuserdata(dbproc)) )
        {
                pECB = gpECB;
                iTermId = 0;
                iSyncId = 0;
        }
        else
        {
                pECB = pEcbInfo->pECB;
                iTermId = pEcbInfo->iTermId;
                iSyncId = pEcbInfo->iSyncId;
        }


        if ( (msgno == 5701) || (msgno == 2528) ||
(msgno == 5703) || (msgno == 6006) )
                return INT_CONTINUE;

        // deadlock message
        if (msgno == 1205)
        {
                // set the deadlock indicator
                if ( pEcbInfo )
                        pEcbInfo->bDeadlock =
TRUE;
                else
                        TMLog("Error,
dbgetuserdata returned NULL.");
                return INT_CONTINUE;
        }
        if ( pEcbInfo && pEcbInfo->bFailed )
        {
                TMLog("SQL Error ");
                return INT_CANCEL;
        }

        if (msgno == 0)
                return INT_CONTINUE;
        else
        {
                TMLog("MsgHandler: SQL Error %s",
msgtext);

                if ( pEcbInfo )
                        pEcbInfo->bFailed = TRUE;

                bError = TRUE;

                sprintf(szTmp, "Error: SQLSVR(%d):
%s", msgno, msgtext);

                TMLog("%2.2d/%2.2d/%2.2d
%2.2d:%2.2d:%2.2d\r\n\r\n%s\r\n\r\n",
                        systemTime.wYear,
systemTime.wMonth, systemTime.wDay,
                        systemTime.wHour,
systemTime.wMinute, systemTime.wSecond,
                        szTmp);
        }
        return INT_CANCEL;
}


/* FUNCTION: BOOL GetParameters(int argc, char *argv[])
 *
 * PURPOSE:         This function parses the command
line passed in to the delivery executable, initializing
 *                          and filling in global
variable parameters.
 *
 * ARGUMENTS:       int                 argc
        number of command line arguments passed to
delivery
 *                                      char
        *argv[]   array of command line argument
pointers
 *
 * RETURNS:                 BOOL      FALSE
        parameter read successfull
 *
        TRUE      user has requested parameter
information screen be displayed.
 *
 * COMMENTS:        None
 *
 */

static BOOL GetParameters(int argc, char *argv[])
{
        int i;

        szServer[0]                 = 0;
        szPassword[0]       = 0;
        bFlush                      = FALSE;
        strcpy(szDatabase, "tpcc");
        strcpy(szUser, "sa");
```

```
          for(i=0; i<argc; i++)
          {
                    if ( argv[i][0] == '-' ||
argv[i][0] == '/' )
                    {
                              switch(argv[i][1])
                              {
                                        case 'S':
                                        case 's':

          strcpy(szServer, argv[i]+2);

          break;
                                        case 'V':
                                        case 'v':

          verbose = TRUE;

          break;
                                        case '?':

          return TRUE;
                              }
                    }
          }
          return FALSE;
}

/* FUNCTION: void PrintParameters(void)
 *
 * PURPOSE:         This function displays the
supported command line flags.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                  None
 *
 * COMMENTS:        None
 *
 */

static void PrintParameters(void)
{
          TMLog("Performance Tuning Corporation Tuxedo
Kit");
          TMLog("   www.perftuning.com   (281) 251-3495
");
          TMLog("NewOrder: -S Server [-v (verbose)]" );
          TMLog("NewOrder: Server %s", szServer);
}
```


Appendix B – Database Design

Build

Createdb.sql

```
 --  File:     CREATEDB.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates tpcc database and backup files


use master
go

--  remove any existing database and backup files

exec sp_dbremove tpcc, dropdev
exec sp_dropdevice 'tpccback1'
exec sp_dropdevice 'tpccback2'
exec sp_dropdevice 'tpccback3'
exec sp_dropdevice 'tpccback4'
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

--  create main database files

create database tpcc on
          (name="MSSQL70_tpcc_root",filename="C:\MSSQL7
0_tpcc_root.mdf",size=10MB, FILEGROWTH=0)
log on
          (name="MSSQL70_tpcc_log",filename="E:",size=4
5000MB, FILEGROWTH=0)

--  create filegroups

alter database tpcc add filegroup MSSQL70_cs_fg
alter database tpcc add filegroup MSSQL70_misc_fg

--  add files to filegroups

alter database tpcc add file
          (name="MSSQL70_cs1",filename="F:",size=17950M
B, FILEGROWTH=0),
          (name="MSSQL70_cs2",filename="G:",size=17950M
B, FILEGROWTH=0),
          (name="MSSQL70_cs3",filename="H:",size=17950M
B, FILEGROWTH=0),
          (name="MSSQL70_cs4",filename="I:",size=17950M
B, FILEGROWTH=0),
          (name="MSSQL70_cs5",filename="J:",size=17950M
B, FILEGROWTH=0),
          (name="MSSQL70_cs6",filename="K:",size=17950M
B, FILEGROWTH=0)
to filegroup MSSQL70_cs_fg

alter database tpcc add file
          (name="MSSQL70_misc1",filename="L:",size=9500
MB, FILEGROWTH=0),
          (name="MSSQL70_misc2",filename="M:",size=9500
MB, FILEGROWTH=0),
          (name="MSSQL70_misc3",filename="N:",size=9500
MB, FILEGROWTH=0),
          (name="MSSQL70_misc4",filename="O:",size=9500
MB, FILEGROWTH=0),
          (name="MSSQL70_misc5",filename="P:",size=9500
MB, FILEGROWTH=0),
          (name="MSSQL70_misc6",filename="Q:",size=9500
MB, FILEGROWTH=0)
to filegroup MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)
go

--  create backup devices

exec sp_addumpdevice
'disk','tpccback1','R:\tpccback1.dmp'
exec sp_addumpdevice
'disk','tpccback2','S:\tpccback2.dmp'
exec sp_addumpdevice
'disk','tpccback3','T:\tpccback3.dmp'
exec sp_addumpdevice
'disk','tpccback4','U:\tpccback4.dmp'
go
```

Tables.sql

```
 --  File:     TABLES.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates TPC-C tables


use tpcc
go

if exists ( select name from sysobjects where name =
'warehouse' )
          drop table warehouse
go
create table warehouse
(
          w_id
          smallint,
          w_name
          char(10),
          w_street_1
          char(20),
          w_street_2
          char(20),
          w_city
          char(20),
          w_state
          char(2),
          w_zip
          char(9),
          w_tax
          numeric(4,4),
          w_ytd
          numeric(12,2)
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name =
'district' )
          drop table district
go
create table district
(
          d_id
          tinyint,
          d_w_id
          smallint,
          d_name
          char(10),
          d_street_1
          char(20),
          d_street_2
          char(20),
          d_city
          char(20),
          d_state
          char(2),
          d_zip
          char(9),
          d_tax
          numeric(4,4),
          d_ytd
          numeric(12,2),
          d_next_o_id                                int
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name =
'customer' )
          drop table customer
go
create table customer
(
          c_id                                       int,
          c_d_id
          tinyint,
          c_w_id
          smallint,
```

```
        c_first
        char(16),
        c_middle                    char(2),
        c_last
        char(16),
        c_street_1
        char(20),
        c_street_2
        char(20),
        c_city
        char(20),
        c_state
        char(2),
        c_zip
        char(9),
        c_phone
        char(16),
        c_since
        datetime,
        c_credit                    char(2),
        c_credit_lim                numeric(12,2),
        c_discount
        numeric(4,4),
        c_balance                   numeric(12,2),
        c_ytd_payment               numeric(12,2),
        c_payment_cnt               smallint,
        c_delivery_cnt              smallint,
        c_data
        char(500)
) on MSSQL70_cs_fg
go

if exists ( select name from sysobjects where name =
'history' )
        drop table history
go
create table history
(
        h_c_id                                  int,
        h_c_d_id                    tinyint,
        h_c_w_id                    smallint,
        h_d_id
        tinyint,
        h_w_id
        smallint,
        h_date
        datetime,
        h_amount                    numeric(6,2),
        h_data
        char(24)
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name =
'new_order' )
        drop table new_order
go
create table new_order
(
        no_o_id                                 int,
        no_d_id
        tinyint,
        no_w_id
        smallint
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name =
'orders' )
        drop table orders
go
create table orders
(
        o_id                                    int,
        o_d_id
        tinyint,
        o_w_id
        smallint,
        o_c_id                                  int,
        o_entry_d                   datetime,
        o_carrier_id                tinyint,
        o_ol_cnt                    tinyint,
        o_all_local
        tinyint
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name =
'order_line' )
        drop table order_line
go
create table order_line
(
        ol_o_id                                 int,
        ol_d_id
        tinyint,
        ol_w_id
        smallint,
        ol_number                   tinyint,
        ol_i_id                                 int,
        ol_supply_w_id              smallint,
        ol_delivery_d               datetime,
        ol_quantity
        smallint,
        ol_amount                   numeric(6,2),
        ol_dist_info                char(24)
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name =
'item' )
        drop table item
go
create table item
(
        i_id                                    int,
        i_im_id                                 int,
        i_name
        char(24),
        i_price
        numeric(5,2),
        i_data
        char(50)
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name =
'stock' )
        drop table stock
go
create table stock
(
        s_i_id                                  int,
        s_w_id
        smallint,
        s_quantity
        smallint,
        s_dist_01                   char(24),
        s_dist_02                   char(24),
        s_dist_03                   char(24),
        s_dist_04                   char(24),
        s_dist_05                   char(24),
        s_dist_06                   char(24),
        s_dist_07                   char(24),
        s_dist_08                   char(24),
        s_dist_09                   char(24),
        s_dist_10                   char(24),
        s_ytd                                   int,
        s_order_cnt
        smallint,
        s_remote_cnt                smallint,
        s_data
        char(50)
) on MSSQL70_cs_fg
go
```

Idxcuscl.sql

```
--  File:     IDXCUSCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates clustered index on customer table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where name =
'customer_c1' )
        drop index customer.customer_c1

create unique clustered index customer_c1 on
customer(c_w_id, c_d_id, c_id)
        on MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
```



```
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

ldxcusnc.sql

```
--  File:     IDXCUSNC.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates non-clustered index on customer
table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name =
'customer_nc1' )
          drop index customer.customer_nc1

create unique nonclustered index customer_nc1 on
customer(c_w_id, c_d_id, c_last, c_first, c_id)
          on MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

ldxdiscl.sql

```
--  File:     IDXDISCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates clustered index on district table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where name =
'district_c1' )
          drop index district.district_c1

create unique clustered index  district_c1 on
district(d_w_id, d_id)
          with fillfactor=100 on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

ldxitmcl.sql

```
--  File:     IDXITMCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates clustered index on item table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name =
'item_c1' )
          drop index item.item_c1

create unique clustered index item_c1 on item(i_id)
          on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

ldxnodcl.sql

```
--  File:     IDXNODCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates clustered index on new_order
table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name =
'new_order_c1' )
          drop index new_order.new_order_c1

create unique clustered index new_order_c1 on
new_order(no_w_id, no_d_id, no_o_id)
          on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

ldxodlcl.sql

```
--  File:     IDXODLCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates clustered index on order_line
table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name =
'order_line_c1' )
          drop index order_line.order_line_c1

create unique clustered index order_line_c1 on
order_line(ol_w_id, ol_d_id, ol_o_id, ol_number)
          on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

Appendix B-Database Design

Idxordcl.sql

```
--  File:     IDXORDCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates clustered index on orders table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name =
'orders_c1' )
          drop index orders.orders_c1

create unique clustered index orders_c1 on
orders(o_w_id, o_d_id, o_id)
          on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

Idxordnc.sql

```
--  File:     IDXORDNC.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates non-clustered index on orders
table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name =
'orders_nc1' )
          drop index orders.orders_nc1

create index orders_nc1 on orders(o_w_id, o_d_id,
o_c_id, o_id)
          on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

Idxstkcl.sql

```
--  File:     IDXSTKCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates clustered index on stock table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name =
'stock_c1' )
          drop index stock.stock_c1

create unique clustered index stock_c1 on stock(s_i_id,
s_w_id)
          on MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

Idxwarcl.sql

```
--  File:     IDXWARCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates clustered index on warehouse
table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where name =
'warehouse_c1' )
          drop index warehouse.warehouse_c1

create unique clustered index warehouse_c1 on
warehouse(w_id)
          with fillfactor=100 on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second,
@startdate, @enddate)

go
```

Dbopt1.sql

```
--  File:     DBOPT1.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Sets database options for data load


use master
go

exec sp_dboption tpcc,'select into/bulkcopy',true
exec sp_dboption tpcc,'trunc. log on chkpt.',true
go

use tpcc
go

checkpoint
go
```

Dbopt2.sql

```
--  File:     DBOPT2.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.01
--            Copyright Microsoft, 1998
--  Purpose:  Resets database options after data load


use master
go

sp_dboption tpcc,'select ',false
go

sp_dboption tpcc,'trunc. ',false
go

use tpcc
go
```

```
checkpoint
go

sp_configure allow,1
go

reconfigure with override
go

/*                                          */
/* Set option values for user-defined indexes */
/*                                          */

sp_indexoption      'customer',
         'AllowPageLocks',   FALSE
go
sp_indexoption      'district',
         'AllowPageLocks',   FALSE
go
sp_indexoption      'warehouse',
         'AllowPageLocks',   FALSE
go
sp_indexoption      'stock',  'AllowPageLocks',   FALSE
go
sp_indexoption      'order_line',
         'AllowPageLocks',   TRUE
go
sp_indexoption      'order_line',
         'AllowRowLocks',    FALSE
go
sp_indexoption      'orders', 'AllowPageLocks',   TRUE
go
sp_indexoption      'orders', 'AllowRowLocks',    FALSE
go
sp_indexoption      'new_order',
         'AllowRowLocks',    FALSE
go
sp_indexoption      'item',
         'AllowRowLocks',    FALSE
go
sp_indexoption      'item',
         'AllowPageLocks',   FALSE
go

Print ' '
Print '******************'
Print 'Pre-specified Locking Hierarchy:'
Print '   Lockflag = 0 ==> No pre-specified hierarchy'
Print '   Lockflag = 1 ==> Lock at Page-level then
Table-level'
Print '   Lockflag = 2 ==> Lock at Row-level then
Table-level'
Print '   Lockflag = 3 ==> Lock at Table-level'
Print ' '

select    name,lockflags
from      sysindexes
where     object_id("warehouse")        = id or
          object_id("district")         = id or
          object_id("customer")         = id or
          object_id("stock")  = id or
          object_id("orders") = id or
          object_id("order_line")       = id or
          object_id("history")          = id or
          object_id("new_order")        = id or
          object_id("item")   = id
order     by lockflags asc
go

sp_configure allow,0
go

reconfigure with override
go

exec sp_dboption tpcc,        'auto update statistics',
         FALSE
exec sp_dboption tpcc,        'auto create statistics',
         FALSE
go

exec sp_tableoption "district",         "pintable",true
exec sp_tableoption "warehouse",        "pintable",true
exec sp_tableoption "new_order",        "pintable",true
exec sp_tableoption "item",             "pintable",true
go
```

Stored Procedures

Neword.sql

```
--  File:     NEWORD.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.01
--            Copyright Microsoft, 1998
--  Purpose:  Creates new order transaction stored
procedure
--
--

use tpcc
go

if exists ( select name from sysobjects where name =
"tpcc_neworder" )
         drop procedure tpcc_neworder
go

create proc tpcc_neworder
                                                    @w_id
smallint,
                                                    @d_id
tinyint,
                                                    @c_id
int,

         @o_ol_cnt       tinyint,

         @o_all_local    tinyint,

         @i_id1  int = 0, @s_w_id1  smallint = 0,
@ol_qty1  smallint = 0,

         @i_id2  int = 0, @s_w_id2  smallint = 0,
@ol_qty2  smallint = 0,

         @i_id3  int = 0, @s_w_id3  smallint = 0,
@ol_qty3  smallint = 0,

         @i_id4  int = 0, @s_w_id4  smallint = 0,
@ol_qty4  smallint = 0,

         @i_id5  int = 0, @s_w_id5  smallint = 0,
@ol_qty5  smallint = 0,

         @i_id6  int = 0, @s_w_id6  smallint = 0,
@ol_qty6  smallint = 0,

         @i_id7  int = 0, @s_w_id7  smallint = 0,
@ol_qty7  smallint = 0,

         @i_id8  int = 0, @s_w_id8  smallint = 0,
@ol_qty8  smallint = 0,

         @i_id9  int = 0, @s_w_id9  smallint = 0,
@ol_qty9  smallint = 0,

         @i_id10 int = 0, @s_w_id10 smallint = 0,
@ol_qty10 smallint = 0,

         @i_id11 int = 0, @s_w_id11 smallint = 0,
@ol_qty11 smallint = 0,

         @i_id12 int = 0, @s_w_id12 smallint = 0,
@ol_qty12 smallint = 0,

         @i_id13 int = 0, @s_w_id13 smallint = 0,
@ol_qty13 smallint = 0,

         @i_id14 int = 0, @s_w_id14 smallint = 0,
@ol_qty14 smallint = 0,

         @i_id15 int = 0, @s_w_id15 smallint = 0,
@ol_qty15 smallint = 0


as
declare   @w_tax          numeric(4,4),
        @d_tax          numeric(4,4),
        @c_last         char(16),
        @c_credit       char(2),
        @c_discount     numeric(4,4),
        @i_price        numeric(5,2),
        @i_name         char(24),
        @i_data         char(50),
        @o_entry_d      datetime,
        @remote_flag    int,
```

```
        @s_quantity     smallint,
        @s_data         char(50),
        @s_dist         char(24),
          @li_no          int,
          @o_id               int,
          @commit_flag    tinyint,
        @li_id          int,
        @li_s_w_id      smallint,
        @li_qty         smallint,
          @ol_number          int,
          @c_id_local         int

begin

begin transaction n

--  get district tax and next availible order id and
update
--  plus initialize local variables

          update    district
          set       @d_tax       = d_tax,
                    @o_id        = d_next_o_id,
                    d_next_o_id  = d_next_o_id + 1,
                    @o_entry_d   = getdate(),
                    @li_no       = 0,
                    @commit_flag = 1
          where     d_w_id       = @w_id and
                    d_id         = @d_id

--  process orderlines

          while (@li_no < @o_ol_cnt)
          begin

                    select @li_no = @li_no + 1


--  set i_id, s_w_id, and qty for this lineitem

                    select    @li_id = case @li_no
                                       when 1 then
@i_id1
                                       when 2 then
@i_id2
                                       when 3 then
@i_id3
                                       when 4 then
@i_id4
                                       when 5 then
@i_id5
                                       when 6 then
@i_id6
                                       when 7 then
@i_id7
                                       when 8 then
@i_id8
                                       when 9 then
@i_id9
                                       when 10 then
@i_id10
                                       when 11 then
@i_id11
                                       when 12 then
@i_id12
                                       when 13 then
@i_id13
                                       when 14 then
@i_id14
                                       when 15 then
@i_id15
                                       end,

                              @li_s_w_id = case @li_no
                                                 when 1
then @s_w_id1
                                                 when 2
then @s_w_id2
                                                 when 3
then @s_w_id3
                                                 when 4
then @s_w_id4
                                                 when 5
then @s_w_id5
                                                 when 6
then @s_w_id6
                                                 when 7
then @s_w_id7
                                                 when 8
then @s_w_id8
                                                 when 9
then @s_w_id9
                                                 when 10
then @s_w_id10
                                                 when 11
then @s_w_id11
                                                 when 12
then @s_w_id12
                                                 when 13
then @s_w_id13
                                                 when 14
then @s_w_id14
                                                 when 15
then @s_w_id15
                                                 end,

                                        @li_qty = case  @li_no
                                                 when 1 then
@ol_qty1
                                                 when 2 then
@ol_qty2
                                                 when 3 then
@ol_qty3
                                                 when 4 then
@ol_qty4
                                                 when 5 then
@ol_qty5
                                                 when 6 then
@ol_qty6
                                                 when 7 then
@ol_qty7
                                                 when 8 then
@ol_qty8
                                                 when 9 then
@ol_qty9
                                                 when 10 then
@ol_qty10
                                                 when 11 then
@ol_qty11
                                                 when 12 then
@ol_qty12
                                                 when 13 then
@ol_qty13
                                                 when 14 then
@ol_qty14
                                                 when 15 then
@ol_qty15
                                                 end

--  get item data (no one updates item)

                    select    @i_price = i_price,
                              @i_name  = i_name,
                              @i_data  = i_data
                    from      item (tablock
repeatableread)
                    where     i_id = @li_id

--  update stock values

                    update    stock
                    set       s_ytd               =
s_ytd + @li_qty,
                              @s_quantity         =
s_quantity   = s_quantity - @li_qty +

                         case when (s_quantity - @li_qty < 10)
then 91 else 0 end,
                              s_order_cnt         =
s_order_cnt + 1,
                              s_remote_cnt        =
s_remote_cnt + case when (@li_s_w_id = @w_id) then 0
else 1 end,
                              @s_data             =
s_data,
                              @s_dist             =
case @d_id

when 1  then s_dist_01

when 2  then s_dist_02

when 3  then s_dist_03

when 4  then s_dist_04

when 5  then s_dist_05

when 6  then s_dist_06

when 7  then s_dist_07

when 8  then s_dist_08

when 9  then s_dist_09
```

```
when 10 then s_dist_10
                                                  end
                    where     s_i_id              =
@li_id and
                              s_w_id              =
@li_s_w_id

--  if there actually is a stock (and item) with these
ids, go to work

                    if (@@rowcount > 0)
                    begin

--  insert order_line data (using data from item and
stock)

                              insert into order_line
values(@o_id,

                @d_id,

                @w_id,

                @li_no,

                @li_id,

                @li_s_w_id,

                "dec 31, 1899",

                @li_qty,

                @i_price * @li_qty,

                @s_dist)

--  send line-item data to client

                              select    @i_name,
                                        @s_quantity,
                                        b_g = case when
( (patindex("%ORIGINAL%",@i_data) > 0) and

            (patindex("%ORIGINAL%",@s_data) > 0) )
                                                  then "B"
else "G" end,
                                        @i_price,
                                        @i_price *
@li_qty

                    end
                    else
                    begin

--  no item (or stock) found - triggers rollback
condition

                              select "",0,"",0,0
                              select @commit_flag = 0

                    end
          end

--  get customer last name, discount, and credit rating

          select    @c_last     = c_last,
                    @c_discount = c_discount,
                    @c_credit   = c_credit,
                    @c_id_local = c_id
          from      customer (repeatableread)
          where     c_id                = @c_id and
                    c_w_id              = @w_id and
                    c_d_id              = @d_id

--  insert fresh row into orders table

          insert into orders values (   @o_id,

          @d_id,

          @w_id,

          @c_id_local,

          @o_entry_d,
                                                  0,

          @o_ol_cnt,

          @o_all_local)

--  insert corresponding row into new-order table

          insert into new_order values (
          @o_id,

          @d_id,

          @w_id)

--  select warehouse tax

          select    @w_tax    = w_tax
          from      warehouse (repeatableread)
          where     w_id      = @w_id

          if (@commit_flag = 1)

                    commit transaction n
          else

--  all that work for nuthin!!!

                    rollback transaction n

--  return order data to client

          select    @w_tax,
                    @d_tax,
                    @o_id,
                    @c_last,
                    @c_discount,
                    @c_credit,
                    @o_entry_d,
                    @commit_flag

end

go
```

Payment.sql

```
 --  File:     PAYMENT.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates payment transaction stored
procedure


use tpcc
go

if exists (select name from sysobjects where name =
"tpcc_payment" )
                                                  drop procedure tpcc_payment
go

create proc tpcc_payment @w_id           smallint,




as
declare  @w_street_1     char(20),
         @w_street_2     char(20),
         @w_city         char(20),
         @w_state        char(2),
         @w_zip          char(9),
         @w_name         char(10),
         @d_street_1     char(20),
         @d_street_2     char(20),
         @d_city         char(20),
         @d_state        char(2),
         @d_zip          char(9),
         @d_name         char(10),
         @c_first        char(16),
         @c_middle       char(2),
         @c_street_1     char(20),
         @c_street_2     char(20),
         @c_city         char(20),
         @c_state        char(2),
         @c_zip          char(9),
         @c_phone        char(16),
         @c_since        datetime,
         @c_credit       char(2),
         @c_credit_lim   numeric(12,2),
         @c_balance      numeric(12,2),
```

```
        @c_discount     numeric(4,4),                                                                                   se
        @data           char(500),                                                                                  end
        @c_data         char(500),
        @datetime       datetime,                           --  get district data and update year-to-date
        @w_ytd          numeric(12,2),
        @d_ytd          numeric(12,2),                                                              update district
        @cnt            smallint,                                                             set d_ytd       = d_ytd + @h_amou
        @val            smallint,
        @screen_data                                char(200),
                                                                                                @d_id_local           tinyint,
                                                                                                @w_id_local           smallint,
                                                                                                @c_id_local           int

select @screen_data = ""
                                                                                                    where d_w_id = @w_id and
begin tran p                                                                                              d_id   = @d_id

--  get payment date                                    --  get warehouse data and update year-to-date

                                            select @datetime = getdate()                                        update warehouse
                                                                                                          set w_ytd       = w_ytd + @h_amou
                                                   if (@c_id = 0)
                                                       begin                                                    @w_street_2 = w_street_2,
                                                                                                                  @w_city     = w_city,
--  get customer id and info using last name                                                                      @w_state    = w_state,
                                                                                                                  @w_zip      = w_zip,
                                                                                                 select @cnt = count(*)       @w_name     = w_name,
                                                                                          from customer (repeatableread)
                                                                                              where c_last = @c_lastwhere w_id = @w_id
                                                                                                    c_w_id = @c_w_id and
                                                   --  create history record                          c_d_id = @c_d_id

                                                                                            select @val = insert into history values  (@c_id_l
                                                                                                       set rowcount @val

                                                                                                  select @c_id = c_id
                                                                                          from customer (repeatableread)
                                                                                              where c_last = @c_last and
                                                                                                    c_w_id = @c_w_id and
                                                                                                        c_d_id = @c_d_id
                                                                                                order by c_last, c_first
                                                   commit tran p
                                                                                                       set rowcount 0
                                                     end   --  return data to client

--  get customer info and update balances               select  @c_id,

                                                update customer set
                                                                    @c_balance     = c_balance = c_balance - @h_amount,
                                                                                     c_payment_cnt  = c_payment_cnt + 1,
                                                                             c_ytd_payment  = c_ytd_payment + @h_amount,
                                                                                              @c_first       = c_first,
                                                                                             @c_middle      = c_middle,
                                                  @c_last        = c_last,
                                                @c_street_1    = c_street_1,
                                                                                           @c_street_2    = c_street_2,
                                                                                              @c_city        = c_city,
                                                                                             @c_state       = c_state,
                                                                                               @c_zip         = c_zip,
                                                                                             @c_phone       = c_phone,
                                                                                            @c_credit      = c_credit,
                                                                                          @c_credit_lim  = c_credit_lim,
                                                                                            @c_discount    = c_discount,
                                                                                               @c_since       = c_since,
                                                                                                @data          = c_data,
                                                                                                   @c_id_local    = c_id
                                                where c_id   = @c_id and
                                                      c_w_id = @c_w_id and
                                                        c_d_id = @c_d_id

--  if customer has bad credit get some more info

                                                if (@c_credit = "BC")
                                                       begin  go

--                                                       compute new info
```

Delivery.sql

```
                                                                                                  select @c_data = convert(char(5),@c_id) +

--  File:     DELIVERY.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates delivery transaction stored
                                                    procedure

--  update customer info

                                                    use tpcc
                                                    go                                          update customer set
                                                                                                   c_data = @c_data
                                                                                                 where c_id   = @c_id and
                                                    if exists (select name from sysobjects where name =
                                                    "tpcc_delivery" )                                                   c_w_id =
                                                             drop procedure tpcc_delivery c_d_id = @c_d_id
```

```
go

create proc tpcc_delivery     @w_id           smallint,



         @o_carrier_id   smallint
as


declare @d_id tinyint,
        @o_id int,
        @c_id int,
        @total numeric(12,2),
        @oid1 int,
        @oid2 int,
        @oid3 int,
        @oid4 int,
        @oid5 int,
        @oid6 int,
        @oid7 int,
        @oid8 int,
        @oid9 int,
        @oid10 int

select @d_id = 0

begin tran d

    while (@d_id < 10)
    begin

        select @d_id  = @d_id + 1,
               @total = 0,
               @o_id  = 0

                    select top 1 @o_id = no_o_id
                    from new_order (serializable
updlock)
        where no_w_id = @w_id and
                        no_d_id = @d_id
                    order by no_o_id asc

        if (@@rowcount <> 0)
        begin

--  claim the order for this district

            delete new_order
            where no_w_id = @w_id and
                             no_d_id = @d_id and
                                   no_o_id = @o_id

--  set carrier_id on this order (and get customer id)

            update orders
                        set o_carrier_id =
@o_carrier_id,
                @c_id       = o_c_id
            where o_w_id = @w_id and
                             o_d_id = @d_id and
                                   o_id   = @o_id

--  set date in all lineitems for this order (and sum
amounts)

            update order_line
                        set ol_delivery_d = getdate(),
                    @total        = @total +
ol_amount
            where ol_w_id = @w_id and
                  ol_d_id = @d_id and
                             ol_o_id = @o_id

--  accummulate lineitem amounts for this order into
customer

                        update customer
                             set c_balance      =
c_balance + @total,
                                 c_delivery_cnt =
c_delivery_cnt + 1
            where c_w_id = @w_id and
                             c_d_id = @d_id and
                                   c_id   = @c_id


        end

        select @oid1 = case @d_id when  1  then @o_id
else @oid1 end,
               @oid2 = case @d_id when  2  then @o_id
else @oid2 end,
               @oid3 = case @d_id when  3  then @o_id
else @oid3 end,
               @oid4 = case @d_id when  4  then @o_id
else @oid4 end,
               @oid5 = case @d_id when  5  then @o_id
else @oid5 end,
               @oid6 = case @d_id when  6  then @o_id
else @oid6 end,
               @oid7 = case @d_id when  7  then @o_id
else @oid7 end,
               @oid8 = case @d_id when  8  then @o_id
else @oid8 end,
               @oid9 = case @d_id when  9  then @o_id
else @oid9 end,
               @oid10 = case @d_id when 10 then @o_id
else @oid10 end

    end

commit tran d

-- return delivery data to client

select @oid1,
       @oid2,
       @oid3,
       @oid4,
       @oid5,
       @oid6,
       @oid7,
       @oid8,
       @oid9,
       @oid10

go
```

Ordstat.sql

```
--  File:     ORDSTAT.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates order status transaction stored
procedure


use tpcc
go

if exists ( select name from sysobjects where name =
"tpcc_orderstatus" )
         drop procedure   tpcc_orderstatus
go

create proc tpcc_orderstatus @w_id
          smallint,

                     @d_id              tinyint,



                     @c_id              int,

                     @c_last  char(16) = ""


as

declare @c_balance            numeric(12,2),
        @c_first              char(16),
        @c_middle             char(2),
        @o_id                          int,
        @o_entry_d            datetime,
        @o_carrier_id         smallint,
        @cnt                           smallint

begin tran o

          if (@c_id = 0)
                    begin

--  get customer id and info using last name

                    select @cnt = (count(*)+1)/2
                    from customer (repeatableread)
                    where c_last = @c_last and
                      c_w_id = @w_id and
                      c_d_id = @d_id

                    set rowcount @cnt
```

```
                    select @c_id = c_id,
                                @c_balance =
c_balance,
                                @c_first   = c_first,
                                @c_last    = c_last,
                                @c_middle  = c_middle
                    from customer (repeatableread)
                    where c_last = @c_last and
                      c_w_id = @w_id and
                      c_d_id = @d_id
                    order by c_w_id, c_d_id, c_last,
c_first

                    set rowcount 0
                    end

          else
                    begin

--  get customer info if by id

                    select @c_balance = c_balance,
                           @c_first   = c_first,
                           @c_middle  = c_middle,
                           @c_last    = c_last
                    from customer (repeatableread)
                    where c_id   = @c_id and
                          c_d_id = @d_id and
                          c_w_id = @w_id


                    select @cnt = @@rowcount

                    end

--  if no such customer

          if (@cnt = 0)
          begin
                    raiserror("Customer not
found",18,1)
                    goto custnotfound
          end

--  get order info

          select @o_id = o_id,
                      @o_entry_d    = o_entry_d,
             @o_carrier_id = o_carrier_id

          from orders (serializable)
          where o_c_id = @c_id and
          o_d_id = @d_id and
          o_w_id = @w_id
          order by o_id asc

--  select order lines for the current order

          select ol_supply_w_id,
             ol_i_id,
                ol_quantity,
                ol_amount,
                ol_delivery_d
          from order_line (repeatableread)
          where ol_o_id = @o_id and
                ol_d_id = @d_id and
                ol_w_id = @w_id

custnotfound:

commit tran o

--  return data to client

select @c_id,
            @c_last,
            @c_first,
            @c_middle,

            @o_entry_d,
            @o_carrier_id,
            @c_balance,
            @o_id

go
```

Stocklev.sql

```
 --  File:     STOCKLEV.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Creates stock level transaction stored
procedure


use tpcc
go

if exists (select name from sysobjects where name =
"tpcc_stocklevel" )
                                          drop procedure tpcc_stocklevel
go

create proc tpcc_stocklevel                          @w_id


as
                                               declare @o_id_low int,

    select @o_id_low  = (d_next_o_id - 20),
                                              @o_id_high = (d_next_o_id -
    from district
    where d_w_id = @w_id and


    select count(distinct(s_i_id))
                                              from  stock, order_line
    where ol_w_id    =  @w_id and
                                                 ol_d_id    =  @d_id and
                                          ol_o_id between @o_id_low and @o_i
          s_w_id     =  ol_w_id and
                                               s_i_id     =  ol_i_id and
                                               s_quantity <  @threshhold

go
```

Tpccldr.c

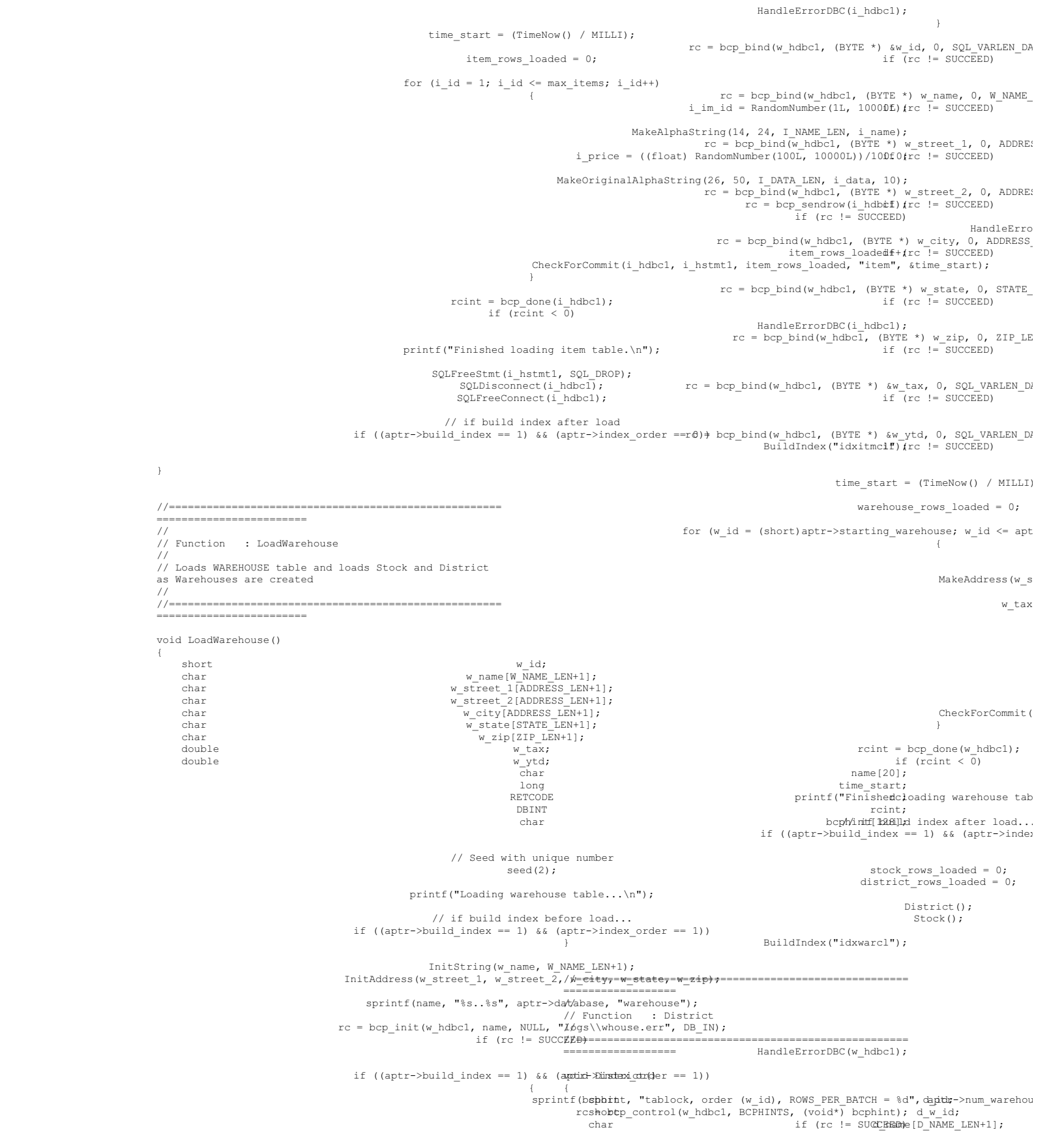
```
//                                                File:
//
//
//
//                                               Purpose:


// Includes
#include "tpcc.h"
#include "search.h"

// Defines
#define MAXITEMS                100000
#define MAXITEMS_SCALE_DOWN
#define CUSTOMERS_PER_DISTRICT  3000
#define CUSTOMERS_SCALE_DOWN                          30
#define DISTRICT_PER_WAREHOUSE  10
#define ORDERS_PER_DISTRICT     3000
#define ORDERS_SCALE_DOWN       30
#define MAX_CUSTOMER_THREADS    2
#define MAX_ORDER_THREADS       3
#define MAX_MAIN_THREADS        4

// Functions declarations

void HandleErrorDBC (SQLHDBC  hdbc1);

long NURand();
void LoadItem();
void LoadWarehouse();

void Stock();
void District();

void LoadCustomer();
void CustomerBufInit();
void CustomerBufLoad();
void LoadCustomerTable();
void LoadHistoryTable();

void LoadOrders();
void OrdersBufInit();
void OrdersBufLoad();
void LoadOrdersTable();
void LoadNewOrderTable();
void LoadOrderLineTable();
void GetPermutation();
void CheckForCommit();
void OpenConnections();
void BuildIndex();
```

Appendix B-Database Design

```
void FormatDate ();

// Shared memory structures

typedef struct
{
    long                ol;
    long                ol_i_id;
    short               ol_supply_w_id;
    short               ol_quantity;
    double              ol_amount;
    char                ol_dist_info[DIST_INFO_LEN+1];
} ORDER_LINE_STRUCT;

typedef struct
{
    long                o_id;
    short               o_d_id;
    short               o_w_id;
    long                o_c_id;
    short               o_carrier_id;
    short               o_ol_cnt;
    short               o_all_local;
} ORDERS_STRUCT;

typedef struct
{
    long
    short
    short
    char
    char
    char
    char
    char
    char
    char
    char
    double
    double
//  fix to avoid ODBC float to numeric conversion
problem.
//  double
    double
    char
} CUSTOMER_STRUCT;

typedef struct
{
    long
} CUSTOMER_SORT_STRUCT;

typedef struct
{
    long                time_start;
} LOADER_TIME_STRUCT;

// Global variables

char                szLastError[300];

HENV                henv;

HDBC                i_hdbc1;
HDBC                w_hdbc1;
HDBC                c_hdbc1;
HDBC                c_hdbc2;
HDBC                o_hdbc1;
HDBC                o_hdbc2;
HDBC                o_hdbc3;

HSTMT               i_hstmt1;
HSTMT               w_hstmt1;
HSTMT               c_hstmt1, c_hstmt2;
HSTMT               o_hstmt1, o_hstmt2, o_hstmt3;

ORDERS_STRUCT   orders_buf[ORDERS_PER_DISTRICT];
CUSTOMER_STRUCT customer_buf[CUSTOMERS_PER_DISTRICT];
long            orders_rows_loaded;
long            new_order_rows_loaded;
long            order_line_rows_loaded;
long            history_rows_loaded;
long            customer_rows_loaded;
long            stock_rows_loaded;
long            district_rows_loaded;
long            item_rows_loaded;
long            warehouse_rows_loaded;
long            main_time_start;
long            main_time_end;
long
long
long
long
long

TPCCLDR_ARGS    *aptr, args;

//=================================================
==================
//
// Function name: main
//
// Description:
//=================================================
==================

int main(int  argc, char **argv)
{
                                        DWORD           dwThreadID[MAX_MAIN_TH
                                          HANDLE          hThread[MAX_MAIN_THR
                                                  FILE            *fLoader;
                                                char            buffer[255];
                                                        int

                                                  for (i=0; i<MAX_MAIN_THREADS; i+

printf("\n*********************************************
******");
                                        printf("\n*
                                        printf("\n*  Microsoft SQL Server
                                        printf("\n*
                                        printf("\n*  TPC-C BENCHMARK KIT:  Database loa
                                    printf("\n*  Version %s
                                        printf("\n*
                                   printf("\n*****************************************

    short       // process command line arguments
    short
    char        aptr = &args;
    double      GetArgsLoader(argc, argv, aptr);

                                                        printf("Build interface is ODBC.\r

                                                    if (aptr->build_index == 0)
                                                                                    pri
                                                            else

                                                    if (aptr->index_order == 0)
                                                                printf("Cluster
                                                            else
                                                              printf("Cluster
                                                   // set database scale values
                                                    if (aptr->scale_down == 1)
                                                                {

                                                                                    c

                                                                }
                                                            else
                                                              {

                                                                                  cus

                                                                }

                                                // open connections to SQL Serve
                                                       OpenConnections();

                                                 // open file for loader results
    fLoader = fopen(aptr->loader_res_file, "w");
```

```
    if (fLoader == NULL)
        {
            printf("Error, loader result file open failed.");
            exit(-1);
        }

    // start loading data

    sprintf(buffer,"TPC-C load started for %ld warehouses.\n",aptr->num_warehouses);

    printf("%s",buffer);
    fprintf(fLoader,"%s",buffer);

    main_time_start = (TimeNow() / MILLI);

    // start parallel load threads

    if (aptr->tables_all || aptr->table_item)
        {
            fprintf(fLoader, "\nStarting loader threads for: item\n");

            hThread[0] = CreateThread(NULL,

            if (hThread[0] == NULL)
                {
                    printf("Er
                    exit(-1);
                }
        }

    if (aptr->tables_all || aptr->table_warehouse)
        {
            fprintf(fLoader, "Starting loader threads for: warehouse\n");

            hThread[1] = CreateThread(NULL,

            if (hThread[1] == NULL)
                {
                    printf("Er
                    exit(-1);
                }
        }

    if (aptr->tables_all || aptr->table_customer)
        {
            fprintf(fLoader, "Starting loader threads for: customer\n");

            hThread[2] = CreateThread(NULL,

            if (hThread[2] == NULL)
                {
                    printf("Er
                    exit(-1);
                }
        }

    if (aptr->tables_all || aptr->table_orders)
        {
            fprintf(fLoader, "Starting loader threads for: orders\n");

            hThread[3] = CreateThread(NULL,

            if (hThread[3] == NULL)
                {
                    printf("Er
                    exit(-1);
                }
        }

    // Wait for threads to finish...
    for (i=0; i<MAX_MAIN_THREADS; i++)
        {
        }

    main_time_end = (TimeNow() / MILL

    sprintf(buffer,"\nTPC-C load completed successfully in %ld minutes.\n",

    printf("%s",buffer);
    fprintf(fLoader, "%s", buffer);

    fclose(fLoader);

    SQLFreeEnv(henv);

    exit(0);

    return 0;
}
```

```
//=======================================================
==================
//
// Function name: LoadItem
//
//=======================================================
==================

void LoadItem()
{
    long
    long
    char
    double
    char
    char
    long
    RETCODE
    DBINT
    char

    // Seed with unique number
    seed(1);

    printf("Loading item table...\n"

    // if build index before load
    if ((aptr->build_index == 1) && (aptr->index

    InitString(i_name, I_NAME_LEN+1)
    InitString(i_data, I_DATA_LEN+1)

    sprintf(name, "%s..%s", aptr->database,

    rc = bcp_init(i_hdbc1, name, NULL, "logs\\ite
    if (rc != SUCCEED)
        {
            printf("Er
            exit(-1);
        }

    if ((aptr->build_index == 1) && (aptr->index
        {
            sprintf(bcpint
            rc = bcp
        }

    rc = bcp_bind(i_hdbc1, (BYTE *) &i_id, 0, SQL_VARLEN_DA
    if (rc != SUCCEED)

    rc = bcp_bind(i_hdbc1, (BYTE *) &i_im_id, 0, SQL_VARLEN_I
    if (rc != SUCCEED)

    rc = bcp_bind(i_hdbc1, (BYTE *) i_name, 0, I_NAME_
    if (rc != SUCCEED)

    rc = bcp_bind(i_hdbc1, (BYTE *) &i_price, 0, SQL_VARLEN_I
    if (rc != SUCCEED)

    rc = bcp_bind(i_hdbc1, (BYTE *) i_data, 0, I_DATA_
    if (rc != SUCCEED)
```

```
char          d_street_1[ADDRESS_LEN+1];
char          d_street_2[ADDRESS_LEN+1];                                for (d_id
char          d_city[ADDRESS_LEN+1];
char          d_state[STATE_LEN+1];
char          d_zip[ZIP_LEN+1];
double        d_tax;
double        d_ytd;
              char                         name[20];
long          d_next_o_id;
              long                         time_start;
              int                                              w_id;
              RETCODE                      rc;
              DBINT                        rcint;
              char                         bcphint[128];

    // Seed with unique number
    seed(4);     }

    printf("Loading district table...\n");          rcint = bcp_done(w_hdbc1);
                                                    if (rcint < 0)

    // build index before load
    if ((aptr->build_index == 1) && (aptr->index_order == 1))
                                     BuildIndex("district");  printf("Finished loading district tabl

    InitString(d_name, D_NAME_LEN+1);               // if build index after load...
    InitAddress(d_street_1, d_street_2, d_city, d_state, d_zip);   if ((aptr->build_index == 1) && (aptr->index
    sprintf(name, "%s..%s", aptr->database, "district");

    rc = bcp_init(w_hdbc1, name, NULL, "logs\\district.err", DB_IN);   return;
    if (rc != SUCCEED)       }
                                     HandleErrorDBC(w_hdbc1);

    if ((aptr->build_index == 1) && (aptr->index_order == 1))   //==================================================
    {                          ==================
    sprintf(bcphint, "tablock, order (d_w_id, d_id), ROWS_PER_BATCH = %u", (aptr->nu
         //rc = bcp_control(w_hdbc1, BCPHINTS, (void*) bcphint);     // Function : Stock
         //                        if (rc != SUCCEED)
         //====================================================        HandleErro
    }    ==================

    rc = bcp_bind(w_hdbc1, (BYTE *) &d_id, 0, SQL_VARLEN_DATA, NULL, 0, SQLINT2, 1);   void LoadStock()
    if (rc != SUCCEED)       {
         long                HandleErrorDBC(w_hdbc1);   s_i_id;
         short                                           s_w_id;
    rc = bcp_bind(w_hdbc1, (BYTE *) &d_w_id, 0, SQL_VARLEN_DATA, NULL, 0, SQLINT2, 2);   short   s_quantity;
    if (rc != SUCCEED)  char                          s_dist_01[S_DIST_LEN+1];
         char                HandleErrorDBC(w_hdbc1);   s_dist_02[S_DIST_LEN+1];
         char                                          s_dist_03[S_DIST_LEN+1];
    rc = bcp_bind(w_hdbc1, (BYTE *) d_name, 0, D_NAME_LEN, NULL, 0, 0, 3);   char   s_dist_04[S_DIST_LEN+1];
    if (rc != SUCCEED)  char                          s_dist_05[S_DIST_LEN+1];
         char                HandleErrorDBC(w_hdbc1);   s_dist_06[S_DIST_LEN+1];
         char                                          s_dist_07[S_DIST_LEN+1];
    rc = bcp_bind(w_hdbc1, (BYTE *) d_street_1, 0, ADDRESS_LEN, NULL, 0, 0, 4);   char   s_dist_08[S_DIST_LEN+1];
    if (rc != SUCCEED)  char                          s_dist_09[S_DIST_LEN+1];
         char                HandleErrorDBC(w_hdbc1);   s_dist_10[S_DIST_LEN+1];
         long                                          s_ytd;
    rc = bcp_bind(w_hdbc1, (BYTE *) d_street_2, 0, ADDRESS_LEN, NULL, 0, 0, 5);   short   s_order_cnt;
    if (rc != SUCCEED)  short                         s_remote_cnt;
         char                HandleErrorDBC(w_hdbc1);   s_data[S_DATA_LEN+1];
         short                                         len;
    rc = bcp_bind(w_hdbc1, (BYTE *) d_city, 0, ADDRESS_LEN, NULL, 0, 0, 6);   char
    if (rc != SUCCEED)                                long
                             HandleErrorDBC(w_hdbc1);  RETCODE
                                                        DBINT
    rc = bcp_bind(w_hdbc1, (BYTE *) d_state, 0, STATE_LEN, NULL, 0, 0, 7);   char
    if (rc != SUCCEED)
                             HandleErrorDBC(w_hdbc1);   // Seed with unique number
                                                        seed(3);
    rc = bcp_bind(w_hdbc1, (BYTE *) d_zip, 0, ZIP_LEN, NULL, 0, 0, 8);
    if (rc != SUCCEED)                                  // if build index before load..
                             HandleErrorDBC(w_hdbc1);   if ((aptr->build_index == 1) && (aptr->index

    rc = bcp_bind(w_hdbc1, (BYTE *) &d_tax, 0, SQL_VARLEN_DATA, NULL, 0, SQLFLT8, 9);
    if (rc != SUCCEED)                   sprintf(name, "%s..%s", aptr->database,
                             HandleErrorDBC(w_hdbc1);
                             rc = bcp_init(w_hdbc1, name, NULL, "logs\\sto
    rc = bcp_bind(w_hdbc1, (BYTE *) &d_ytd, 0, SQL_VARLEN_DATA, NULL, 0, SQLFLT8, 10);   if (rc != SUCCEED)
    if (rc != SUCCEED)
                             HandleErrorDBC(w_hdbc1);
                             if ((aptr->build_index == 1) && (aptr->index
    rc = bcp_bind(w_hdbc1, (BYTE *) &d_next_o_id, 0, SQL_VARLEN_DATA, NULL, 0, SQLINT4, 11);   {
    if (rc != SUCCEED)                                  sprintf(bcphint
                             HandleErrorDBC(w_hdbc1);            rc = bcp

    d_ytd  = 30000.0;
                                                        }
    d_next_o_id = orders_per_district+1;
                             rc = bcp_bind(w_hdbc1, (BYTE *) &s_i_id, 0, SQL_VARLEN_D
    time_start = (TimeNow() / MILLI);                  if (rc != SUCCEED)

    for (w_id = aptr->starting_warehouse; w_id <= aptr->num_warehouses; w_id++)
    {                        bcp_bind(w_hdbc1, (BYTE *) &s_w_id, 0, SQL_VARLEN_DAT
                                     d_w_id = w_id;  if (rc != SUCCEED)
```

```
                                                                HandleErrorDBC(w_hdbc1);

 rc = bcp_bind(w_hdbc1, (BYTE *) &s_quantity, 0, SQL_VARLEN_DATA, NULL, 0, SQLINT2, 3);
                                if (rc != SUCCEED)                                                  }
                                                                HandleErrorDBC(w_hdbc1);
                                                                        rcint = bcp_done(w_hdbc1);
       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_01, 0, S_DIST_LEN, NULL, 0, 0, 4);                    if (rcint < 0)
                                if (rc != SUCCEED)
                                                                HandleErrorDBC(w_hdbc1);
                                                                        printf("Finished loading stock table
       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_02, 0, S_DIST_LEN, NULL, 0, 0, 5);
                                if (rc != SUCCEED)                                      SQLFreeStmt(w_hstmt1, SQL_DROP)
                                                                HandleErrorDBC(w_hdbc1, SQLDisconnect(w_hdbc1);
                                                                          SQLFreeConnect(w_hdbc1);
       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_03, 0, S_DIST_LEN, NULL, 0, 0, 6);
                                if (rc != SUCCEED)                                       // if build index after load...
                                                                HandleErrorDBC(w_hdbc1, if ((aptr->build_index == 1) && (aptr->index

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_04, 0, S_DIST_LEN, NULL, 0, 0, 7);
                                if (rc != SUCCEED)                                                      return;
                                    }                           HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_05, 0, S_DIST_LEN, NULL, 0, 0, 8);
                                if (rc != SUCCEED)  //============================================================
                                                    ==================  HandleErrorDBC(w_hdbc1);
                                                    //
       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_06, //0, S_DIST_LEN, NULL, 0, 0, 9);
                                if (rc != SUCCEED)  //
                                                    //================================HandleErrorDBC(w_hdbc1);
                                                    ==================
      rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_07, 0, S_DIST_LEN, NULL, 0, 0, 10);
                                if (rc != SUCCEED) LoadCustomer()
                                                    {           HandleErrorDBC(w_hdbc1);
                                                                  LOADER_TIME_STRUCT      customer_time
       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_08, 0, S_DIST_LEN, NULL, 0, 0, 11);      LOADER_TIME_STRUCT      history_time_
                                if (rc != SUCCEED)                                                  short
                                        short               HandleErrorDBC(w_hdbc1);  d_id;
                                                                                      DWORD
       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_09, 0, S_DIST_LEN, NULL, 0, 0, 12);     HANDLE
                                if (rc != SUCCEED)                                    char
                                                    HandleErrorDBC(w_hdbc1); RETCODE
                                                                                      DBINT
       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_10, 0, S_DIST_LEN, NULL, 0, 0, 13);     char
                                if (rc != SUCCEED)                                    char
                                                    HandleErrorDBC(w_hdbc1);// SQLRETURN
                                                                                    // SQLSMALLINT
   rc = bcp_bind(w_hdbc1, (BYTE *) &s_ytd, 0, SQL_VARLEN_DATA, NULL, 0, SQLINT4, 14);    // SQLCHAR
                                if (rc != SUCCEED)                                     // SQLINTEGER
                                                    HandleErrorDBC(w_hdbc1);
                                                                          // Seed with unique number
rc = bcp_bind(w_hdbc1, (BYTE *) &s_order_cnt, 0, SQL_VARLEN_DATA, NULL, 0, SQLINT2, 15);      seed(5);
                                if (rc != SUCCEED)
                                                    HandlepreError("DBC(w_hdbc1); tomer and history tab

rc = bcp_bind(w_hdbc1, (BYTE *) &s_remote_cnt, 0, SQL_VARLEN_DATA, NULL, 0, SQLINT2, 16);     // if build index before load..
                                if (rc != SUCCEED)                                  if ((aptr->build_index == 1) && (aptr->index
                                                    HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) s_data, 0, S_DATA_LEN, NULL, 0, 0, 17);                 // Initialize bulk copy
                                if (rc != SUCCEED)                          sprintf(name, "%s..%s", aptr->database, "
                                                    HandleErrorDBC(w_hdbc1);
                                                  rc = bcp_init(c_hdbc1, name, NULL, "logs\\custo
                    s_ytd = s_order_cnt = s_remote_cnt = 0;                         if (rc != SUCCEED)

                       time_start = (TimeNow() / MILLI);

                                                    if ((aptr->build_index == 1) && (aptr->index
                     printf("...Loading stock table\n");                                 {
                                                                                          sprintf(bcphint
                  for (s_i_id=1; s_i_id <= max_items; s_i_id++)                                  rc = bcp
                                    {

                                  for (s_w_id = (short)aptr->starting_warehouse; s_w_id <= aptr->num_warehouses; s
                                                                                          {
                                                                  sprintf(name, "%s..%s", aptr->database, '
                                                                                            s_quantity
                                                                rc = bcp_init(c_hdbc2, name, NULL, "logs\\Make
                                                                                 if (rc != SUCCEED)= Make
                                                                                            len = Make
                                                                                            len = Make
                                                                                  sprintf(bcphint, "tablock,Make
                                                                rc = bcp_control(c_hdbc2, BCPHINTS, (void Make
                                                                                 if (rc != SUCCEED)= Make
                                                                                            len = Make
                                                                                            len = Make
                                                                                customer_rows_loaded len = 0;Make
                                                                                history_rows_loaded     = 0;
                                                                                            len = Make
                                                                                CustomerBufInit();
                                                                                            rc = bcp_s
                                                                customer_time_start.time_start = (TimeNow
                                                                history_time_start.time_start = (TimeNow(

                                                  for (w_id = (short)aptr->starting_warehouse; w_id <= aptr
```

```
{
      for (d_id = 1; d_id <= DISTRICT_PER_WAREHOUSE; d_id++)
          return;
      }

      //=================================================
      ==================
      //
      // Function   : CustomerBufInit
      //
      //=================================================
      ==================

      void CustomerBufInit()
      {
      }

      //=================================================
      ==================
      //
      // Function   : CustomerBufLoad
      //
      // Fills shared buffer for HISTORY table
      //=================================================
      ==================

      void CustomerBufLoad(int d_id, int w_id)
      {
            CUSTOMER_SORT_STRUCT    c[CUSTOMERS_PER_DISTRICT];
            for (i=0;i<customers_per_district;
            {
sprintf(cmd, "isql -S%s -U%s -P%s -d%s -e -Q\"update customer set c_first = 'C_LOAD = %d' where c_id = 1 and c_w_id = 1 and c_d_id = 1\" > l
                  system(cmd);
            }
      }
```

```
SQLFreeStmt(c_hstmt2, SQL_DROP)
SQLDisconnect(c_hdbc2);
SQLFreeConnect(c_hdbc2);
CustomerBu
// Start p
// Start c
printf("..
hThread[0]
if (hThrea
{
int     i;
for (i=0;i<customers_per_district;
{
// Start H
printf("..
hThread[1]
if (hThrea
{
}
WaitForSin
// WaitForSi
if (CloseH
{
}
if (CloseH
{
}
}
}
}
// flush the bulk connection
rcint = bcp_done(c_hdbc1);
if (rcint < 0)
{
HandleErrorDBC(c_hdbc1);
rcint = bcp_done(c_hdbc2);
if (rcint < 0)
{
HandleErrorDBC(c_hdbc2);
}
printf("Finished loading customer table.\n");
// if build index after load...
if ((aptr->build_index == 1) && (aptr->index_order == 0))
BuildIndex("idxcuscl");   long
// build non-clustered index
if (aptr->build_index == 1)
BuildIndex("idxcusnc");
// Output the NURAND used for the loader into C_FIRST for C_ID = 1,
// C_W_ID = 1, and C_D_ID = 1
MakeAl
SQLFreeStmt(c_hstmt1, SQL_DROP);
SQLDisconnect(c_hdbc1);
SQLFreeConnect(c_hdbc1);
printf("...Loading customer buffer for: d_id =
```

```
                                        for (i=0;i<customers_per_district;i++)
                                                {
                                                    rc = bcp_bind(c_hdbc1, (BYTE *) &c_d_id, 0,
                                                SQL_VARLEN_DATA, NULL, 0, SQLINT2, 2);
                                                                customer_buf[i].c_d_id = d_id;
                                                                customer_buf[i].c_w_id = w_id;
                                                                customer_buf[i].h_amount = 10.0;

                                                            rc = bcp_bind(c_hdbc1, (BYTE *) &c_w_id, 0,
                                                                                                    if (rc != SUCCEED)
                                                                customer_buf[i].c_payment_cnt  = 1;
                                                                customer_buf[i].c_delivery_cnt = 0;
                                                    rc = bcp_bind(c_hdbc1, (BYTE *) c_first, 0,
                                                FIRST_NAME_LEN, NULL, 0, 0);
                                                                                                    if (rc != SUCCEED)
                                                                 customer_buf[i].c_id = c[i].c_id;

                                                    rc = bcp_bind(c_hdbc1, (BYTE *) c_middle, 0,
                                                MIDDLE_NAME_LEN, NULL, 0, 0);
                                                                                                    if (rc != SUCCEED)
                                                                customer_buf[i].c_middle[0] = 'O';
                                                                customer_buf[i].c_middle[1] = 'E';
                                                    rc = bcp_bind(c_hdbc1, (BYTE *) c_last, 0,
                                                LAST_NAME_LEN, NULL, 0, 0);
                                                                                                    if (rc != SUCCEED)


                                                    rc = bcp_bind(c_hdbc1, (BYTE *) c_street_1, 0,
                                                ADDRESS_LEN, NULL, 0, 0, 7);
                                            MakeNumberString(16, 16, PHONE_LEN, customer_buf[i].c_phone);
                                                                                                    if (rc != SUCCEED)

                                                                 if (RandomNumber(1L, 100L) > 10)
                                                    rc = bcp_bind(c_hdbc1, (BYTE *) c_street_2, 0,
                                                ADDRESS_LEN, NULL,0,0, 8);                          else
                                                                                                    if (rc != SUCCEED)
                                                                customer_buf[i].c_credit[1] = 'C';

                                                    rc = bcp_bind(c_hdbc1, (BYTE *) &c_credit_lim, 0,
                                                SQL_VARLEN_DATA, NULL, 0, SQLFLT8, 9);              customer_buf[i].c_credit_lim =
                                            MakeAlphaString(0, 1, STATE_LEN, customer_buf[i].c_state);
                                                                                                    if (rc != SUCCEED)
                                                                // Generate HISTORY data
                                            MakeAlphaString(12, 24, H_DATA_LEN, customer_buf[i].h_data);
                                                    rc = bcp_bind(c_hdbc1, (BYTE *) c_phone, 0,
                                                PHONE_LEN, NULL, 0, 0, 12);
                                                                                                    if (rc != SUCCEED)

                                                    rc = bcp_bind(c_hdbc1, (BYTE *) c_credit, 0,
                                                CREDIT_LEN, NULL, 0, 0, 14);
                                                                                                    if (rc != SUCCEED)

                                                    rc = bcp_bind(c_hdbc1, (BYTE *) &c_credit_lim, 0,
                                                SQL_VARLEN_DATA, NULL, 0, SQLFLT8, 15);
                                                                                                    if (rc != SUCCEED)

                                                    rc = bcp_bind(c_hdbc1, (BYTE *) &c_discount, 0,
                                                SQL_VARLEN_DATA, NULL, 0, SQLFLT8, 16);
                                                                                                    if (rc != SUCCEED)


                                                                // fix to avoid ODBC float to numeric conve
                                                    // rc = bcp_bind(c_hdbc1, (BYTE *) &c_balance, 0,
                                                SQL_VARLEN_DATA, NULL, 0, SQLFLT8, 17);
                                        //  fix to avoid ODBC float to numeric conversion problem.          //  if (rc != SUCCEED)
                                    //  double                                                                  //    c_balance;
                                       char       rc = bcp_bind(c_hdbc1, (BYTE *) c_balance, 0, 5,            c_balance[
                                                NULL, 0, SQLCHARACTER, 17);
    double      c_ytd_payment;                                                                              if (rc != SUCCEED)
    short       c_payment_cnt;
    short       c_delivery_cnt;
    char        c_data[C_DATA_LEN+1];
                                         char       rc = bcp_bind(c_hdbc1, (BYTE *) &c_ytd_payment, 0,          c_since[C_
                                       RETCODE  SQL_VARLEN_DATA, NULL, 0, SQLFLT8, 18);                     rc;
                                                                                                    if (rc != SUCCEED)
    rc = bcp_bind(c_hdbc1, (BYTE *) &c_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT4, 1);
                                    if (rc != SUCCEED)
}

//=====================================================
==================
//
// Function   : LoadCustomerTable
//
//=====================================================
==================

void LoadCustomerTable(LOADER_TIME_STRUCT
*customer_time_start)
{
                                        int         i;
    long        c_id;
    short       c_d_id;
    short       c_w_id;
    char        c_first[FIRST_NAME_LEN+1];
    char        c_middle[MIDDLE_NAME_LEN+1];
    char        c_last[LAST_NAME_LEN+1];
    char        c_street_1[ADDRESS_LEN+1];
    char        c_street_2[ADDRESS_LEN+1];
    char        c_city[ADDRESS_LEN+1];
    char        c_state[STATE_LEN+1];
    char        c_zip[ZIP_LEN+1];
    char        c_phone[PHONE_LEN+1];
    char        c_credit[CREDIT_LEN+1];
    double      c_credit_lim;
    double      c_discount;
```

```
    rc = bcp_bind(c_hdbc1, (BYTE *) &c_payment_cnt, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 19);
if (rc != SUCCEED)

    rc = bcp_bind(c_hdbc1, (BYTE *) &c_delivery_cnt,0,
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 20);
if (rc != SUCCEED)

    rc = bcp_bind(c_hdbc1, (BYTE *) c_data, 0, 500,
NULL, 0, 0, 21);
if (rc != SUCCEED)

    for (i = 0; i < customers_per_district; i++)
    {

    }
}

//=====================================================
==================
//
// Function   : LoadHistoryTable
//
//=====================================================
==================

void LoadHistoryTable(LOADER_TIME_STRUCT
*history_time_start)
{
    long        c_id;
    short       c_d_id;
    short       c_w_id;
    char        h_data[H_DATA_LEN+1];

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT4, 1);
if (rc != SUCCEED)

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_d_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 2);
if (rc != SUCCEED)

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_w_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 3);
if (rc != SUCCEED)
                                        rc = bcp_bind(c_hdbc2, (BYTE *) &c_d_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 4);
                                        HandleErrorDBC(c_hdbc1);
                                        rc = bcp_bind(c_hdbc2, (BYTE *) &c_w_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 5);
                                        HandleErrorDBC(c_hdbc1);
                                    rc = bcp_bind(c_hdbc2, (BYTE *) &h_date, 0, H_DATE_LEN,
                                        HandleErrorDBC(c_hdbc1);
                                        rc = bcp_bind(c_hdbc2, (BYTE *) &h_amount, 0,
                                    SQL_VARLEN_DATA, NULL, 0, SQLFLT8, 7);
                                        rc = bcp_bind(c_hdbc2, (BYTE *) h_data, 0,
                                    H_DATA_LEN, NULL, 0, 8);
                                    for (i = 0; i < customers_per_district; i++)
                                    {
                                    //  fix to avoid ODBC float to numeric conversion problem.
                                    CheckForCommit(c_hdbc1, c_hstmt1, customer_rows_loaded, "customer", &customer_ti
                                        void LoadOrders()
                                        {
                                        short
                                        // seed with unique number
                                        printf("Loading orders...\n");
                                        HandleErrorDBC(c_hdbc2);
                                        HandleErrorDBC(c_hdbc2);
                                        HandleErrorDBC(c_hdbc2);
```

```
                                                    c_id = customer_buf[i].c_id;
                                                c_d_id = customer_buf[i].c_d_id;
                                                c_w_id = customer_buf[i].c_w_id;
                                    strcpy(h_data, customer_buf[i].c_first);
                                            strcpy(c_middle, customer_buf[i].c_middle);
                                                strcpy(c_last, customer_buf[i].c_last);
                                        strcpy(c_street_1, customer_buf[i].c_street_1);
                                        strcpy(c_street_2, customer_buf[i].c_street_2);
                                            strcpy(c_city, customer_buf[i].c_city);
                                        strcpy(c_state, customer_buf[i].c_state);
                                            strcpy(c_zip, customer_buf[i].c_zip);
                                        strcpy(c_phone, customer_buf[i].c_phone);
                                        strcpy(c_credit, customer_buf[i].c_credit);

                                                FormatDate(&c_since);

                                        c_credit_lim = customer_buf[i].c_credit_lim;
                                            c_discount = customer_buf[i].c_discount;

                                            //  c_balance = customer_buf[i].c_balance;
                                        strcpy(c_balance, customer_buf[i].c_balance);

                                        c_ytd_payment = customer_buf[i].c_ytd_payment;
                                    }   c_payment_cnt = customer_buf[i].c_payment_cnt;
                                    c_delivery_cnt = customer_buf[i].c_delivery_cnt;
                                    }
                                            strcpy(c_data, customer_buf[i].c_data);

                                    //==============================Send data to server
                                    =======---------------      rc = bcp_sendrow(c_hdbc1);
                                    //                              if (rc != SUCCEED)
                                    // Function   : LoadOrders                                  HandleErro
                                    //
                                    //==============================customer_rows_loaded++;
                                                                                                    CheckForCommit(
                                        LOADER_TIME_STRUCT      orders_time_
                                        LOADER_TIME_STRUCT      new_order_time
                                        LOADER_TIME_STRUCT      order_line_tim
                                                                short
                                        short                   d_id;
                                                                DWORD
                                                                HANDLE
                                                                char
                                                                RETCODE
                                                                char

                                                    // seed with unique number
                                                        seed(6);
                                        int         i;
                                                    printf("Loading orders...\n");

                                                    // if build index before load..
                                        double      h_amount;
                                                    if ((aptr->build_index == 1) && (aptr->index
                                                    {
                                        char                                                h_date[H_D
                                        RETCODE                                             rc;

                                                    }

                                                    // initialize bulk copy
                                                    sprintf(bcphint, "hints=\"ORDER(%s)\"", aptr->database,

                                                rc = bcp_init(o_hdbc1, name, NULL, "logs\\orde
                                                    if (rc != SUCCEED)

                                                    if ((aptr->build_index == 1) && (aptr->index
                                                    {
                                                    sprintf(bcphint
                                                        rc = bcp
```

```
                                                                                                                                HandleErro
                                                    }

                                sprintf(name, "%s..%s", aptr->database, "new_order");

                             rc = bcp_init(o_hdbc2, name, NULL, "logs\\neword.err", DB_IN);
                                                if (rc != SUCCEED)
                                                                                        HandleErrorDBC(o_hdbc2);

                               if ((aptr->build_index == 1) && (aptr->index_order == 1))
                                                    {
                                                     sprintf(bcphint, "tablock, order (no_w_id, no_d_id, no_o_id), ROWS_PER_BATCH = %
                                                            rc = bcp_control(o_hdbc2, BCPHINTS, (void*) bcphint);
                                                                                        if (rc != SUCCEED)
                                                                                                                                HandleErro
                                                    }

                                sprintf(name, "%s..%s", aptr->database, "order_line");

                             rc = bcp_init(o_hdbc3, name, NULL, "logs\\ordline.err", DB_IN);
                                                if (rc != SUCCEED)
                                                                                        HandleErrorDBC(o_hdbc3);

    if ((aptr->build_index == 1) && (aptr->index_order
== 1))
                                                    {
                                                     sprintf(bcphint, "tablock, order (ol_w_id, ol_d_id, ol_o_id, ol_number), ROWS_PE
                                                            rc = bcp_control(o_hdbc3, BCPHINTS, (void*) bcphint);
                                                                                        if (rc != SUCCEED)
                                                                                                                    }     HandleErro
                                                    }
                                                                                                 printf("Finished loading orders.\r
                                            orders_rows_loaded      = 0;
                                            new_order_rows_loaded   = 0;
                                            order_line_rows_loaded return;
                                                                }
                                                  OrdersBufInit();

                                  orders_time_start.time_start = //(TimeNow() - MILLI);==================================
                                 new_order_time_start.time_start ==(TimeNow() - MILLI);
                                order_line_time_start.time_start = // (TimeNow() / MILLI);
                                                                // Function   : OrdersBufInit
                    for (w_id = (short)aptr->starting_warehouse; w_id <= aptr->num_warehouses; w_id++)
                                                    {     // Clears shared buffer for ORDERS, NEWORDER, and
                                                          ORDERLINE
                                                                for (d_id = 1; d_id <= DISTRICT_PER_WAREHOUSE; d_id++)
                                                                //                                        {
                                                                //=====================================================
                                                                ==================                                OrdersBufL

                                                                void OrdersBufInit()                              // start p
                                                                {
                                                                                                        int     i;// start O
                                                                                                           int
                                                                                                                  printf("..
                                                                                                 for (i=0;i<orders_per_district;i-
                                                                                                                  {     hThread[0]

                                                                                                                                if (hThrea
                                                                                                                                {


                                                                                                                                }

                                                                                                                                // start N

                                                                                                                                printf("..

                                                                                                                                hThread[1]
                                                                                                                  }

                                                                }


                                                                //=====================================================
                                                                ==================                                if (hThrea
                                                                //                                                {
                                                                // Function   : OrdersBufLoad
                                                                //
                                                                // Fills shared buffer for ORDERS, NEWORDER, and                  }
                                                                ORDERLINE
                                                                //                                                // start O
                                                                //=====================================================
                                                                ==================                                printf("..

                                                                void OrdersBufLoad(int d_id, int w_id)            hThread[2]
                                                                {

                                                                                                      int     cust[ORDERS_PER_DIST+1]
```

```
                                        long    o_id;
                                        short   ol;
                                                rc = bcp_bind(o_hdbc1, (BYTE *) &o_w_id, 0,
                printf("...Loading Order Buffer for W_ID=%d\n", w_id);
                                                SQL_VARLEN_DATA, NULL, 0, SQLINT2, 3);
                                                                                        if (rc != SUCCEED)

                   GetPermutation(cust, ORDERS_PER_DIST);
                                                rc = bcp_bind(o_hdbc1, (BYTE *) &o_c_id, 0,
           for (o_id=0;o_id<orders_per_district;o_id++)
                                                SQL_VARLEN_DATA, NULL, 0, SQLINT4, 4);
                                        {                                               if (rc != SUCCEED)

                                                        // Generate ORDER and NEW-ORDER data
                                                    rc = bcp_bind(o_hdbc1, (BYTE *) &o_entry_d, 0, O_ENTRY_D_LE
                                                            orders_buf[o_id].o_d_id = d_id;
                                                            orders_buf[o_id].o_w_id = w_id;
                                                            orders_buf[o_id].o_id = o_id+1;
                                                rc = bcp_bind(o_hdbc1, (BYTE *) &o_carrier_id, 0,
                                            orders_buf[o_id].o_c_id = cust[o_id];
                                                SQL_VARLEN_DATA, NULL, 0, SQLINT2, RandomNumber(5L, 15L);
                                                                                        if (rc != SUCCEED)
                                                                if (o_id < first_new_order)
                                                                                        {
                                                rc = bcp_bind(o_hdbc1, (BYTE *) &o_ol_cnt, 0,                orders_buf
                                                SQL_VARLEN_DATA, NULL, 0, SQLINT2, 7);                       orders_buf
                                                                                        if (rc != SUCCEED)
                                                                                        else
                                                                                        {
                                                rc = bcp_bind(o_hdbc1, (BYTE *) &o_all_local, 0,             orders_buf
                                                SQL_VARLEN_DATA, NULL, 0, SQLINT2, 8);                       orders_buf
                                                                                        if (rc != SUCCEED)

                                                     for (ol=0; ol<orders_buf[o_id].o_ol_cnt; ol++)
                                                for (i = 0; i < orders_per_district; i++)        {
                                                {
                                                                                                             orders_buf
                                                                                                             orders_buf
                                                                                                             orders_buf
                                                                                                             orders_buf
                                                                                                             MakeAlphaS

                                                                                                             // Generat
                                                                                                             if (o_id <
                                                                                                             {









                                                                                                             }
                                                                                                             else
                                                                                                             {
                                                                                                          CheckForCommit(
                                                }

                                                                                        // rcint = bcp_batch(o_hdbc1);
                                                                                              // if (rcint < 0)
                                                                                                    //
                                                                                                             }
                                                                                    if ((o_w_id == aptr->num_warehouses) && (o_
                                        }                                                                    {
}


//=====================================================
==================
//
// Function   : LoadOrdersTable
//
//=====================================================
==================                                                                                        if ((aptr->b

void LoadOrdersTable(LOADER_TIME_STRUCT
*orders_time_start)
{
                                                  int         i;                                          }
    long        o_id;                                 short                                       o_d_id;
                                                      short   }                                   o_w_id;
    long        o_c_id;
    short       o_carrier_id;
    short       o_ol_cnt;                                 //=====================================================
    short       o_all_local;                              ==================
                                                       char   //                                                  o_entry_d[
                                                     RETCODE  // Function   : LoadNewOrderTable                   rc;
                                                      DBINT   //                                                  rcint;
                                                              //=====================================================
                                                  // bind ORDER data==============
    rc = bcp_bind(o_hdbc1, (BYTE *) &o_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT4, 1);                        void LoadNewOrderTable(LOADER_TIME_STRUCT
                                                  if (rc != SUCCEED)
                                                              {                                HandleErrorDBC(o_hdbc1);
                                                                  int         i;
    rc = bcp_bind(o_hdbc1, (BYTE *) &o_d_id, 0,                   long        o_id;
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 2);                            short       o_d_id;
                                                  if (rc != SUCCEED)short       o_w_id;
```

```
                                                    RETCODE      rc = bcp_bind(o_hdbc3, (BYTE *) &o_w_id, 0,                 rc;
                                                     DBINT   SQL_VARLEN_DATA, NULL, 0, SQLINT2, 3);                      rcint;
                                                                                                                  if (rc != SUCCEED)
                                                // Bind NEW-ORDER data

    rc = bcp_bind(o_hdbc2, (BYTE *) &o_id, 0,                       rc = bcp_bind(o_hdbc3, (BYTE *) &ol, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT4, 1);                          SQL_VARLEN_DATA, NULL, 0, SQLINT4, 4);
                                                 if (rc != SUCCEED)                                                  if (rc != SUCCEED)
                                                                                                                        HandleErrorDBC(o_hdbc2);

    rc = bcp_bind(o_hdbc2, (BYTE *) &o_d_id, 0,                     rc = bcp_bind(o_hdbc3, (BYTE *) &ol_i_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 2);                          SQL_VARLEN_DATA, NULL, 0, SQLINT4, 5);
                                                 if (rc != SUCCEED)                                                  if (rc != SUCCEED)
                                                                                                                        HandleErrorDBC(o_hdbc2);

    rc = bcp_bind(o_hdbc2, (BYTE *) &o_w_id, 0,                     rc = bcp_bind(o_hdbc3, (BYTE *) &ol_supply_w_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 3);                          SQL_VARLEN_DATA, NULL, 0, SQLINT2, 6);
                                                 if (rc != SUCCEED)                                                  if (rc != SUCCEED)
                                                                                                                        HandleErrorDBC(o_hdbc2);

                                    for (i = first_new_order; i < last_new_order; i++)  rc = bcp_bind(o_hdbc3, (BYTE *) &ol_delivery_d, 0, OL_DELIVERY_
                                                                                    {                                   if (rc != SUCCEED)
                                                                                                          o_id    = orders_buf[i].o_id;
                                                                                                        o_d_id  = orders_buf[i].o_d_id;
                                                                                        rc = bcp_bind(o_hdbc3, (BYTE *) &orders_buf[i].o_w_id, 0,
                                                                                    SQL_VARLEN_DATA, NULL, 0, SQLINT2, 8);
                                                                                                          rc = bcp_sendrow(o_hdbc2);  if (rc != SUCCEED)
                                                                                                                  if (rc != SUCCEED)
                                                                                                                                              HandleErro
                                                                                        rc = bcp_bind(o_hdbc3, (BYTE *) &ol_amount, 0,
                                                                                    SQL_VARLEN_DATA, NULL, 0, SQLFLT8, 0);  new_order_rows_loaded++;
                                                                                 CheckForCommit(o_hdbc2, o_hstmt2, new_order_rows_loaded, if (rc != SUCCEED)  new_order
                                                                                    }

                                                // rcint = bcp_batch(o_hdbc2);  rc = bcp_bind(o_hdbc3, (BYTE *) ol_dist_info, 0,
                                                       // if (rcint < 0)    DIST_INFO_LEN, NULL, 0, 0, 10);
                                                                                     //                                 if (rc != SUCCEED)

                                    if ((o_w_id == aptr->num_warehouses) && (o_d_id == 10))
                                                                                    {                                                   for (i = 0; i < orders_per_district;
                                                                                                {                           rcint = bcp_done(o_hdbc2);
                                                                                                                              if (rcint < 0)
                                                                                                                                              HandleErro

                                                                                                                  SQLFreeStmt(o_hstmt2, SQL_DROP);
                                                                                                                           SQLDisconnect(o_hdbc2);
                                                                                                                          SQLFreeConnect(o_hdbc2);

                                                                                                                 // if build index after load...
                                                                                    if ((aptr->build_index == 1) && (aptr->index_order == 0))
                                                                                                                                              BuildIndex

                                                                                    }

}


//=====================================================
==================
//
// Function   : LoadOrderLineTable
//
//=====================================================
==================                                                                                                  }

void LoadOrderLineTable(LOADER_TIME_STRUCT                                                                  // rcint = bcp_batch(o_hdbc3);
*order_line_time_start)                                                                                           // if (rcint < 0)
{                                                                                                                      //
                                                    int         i,j;
    long        o_id;
                                                     short                                  if ((o_w_id == aptr->num_warehouses) && (o_
                                                     short                                                  o_w_id;     {
    long        ol;
                                                      long                                                 ol_i_id;
    short       ol_supply_w_id;
    short       ol_quantity;
    double      ol_amount;
    char        ol_dist_info[DIST_INFO_LEN+1];
                                                      char                                                                  ol_deliver
                                                    RETCODE                                                                 rc;
                                                     DBINT                                                                  rcint;
                                                                                                                          if ((aptr->b
                                                // bind ORDER-LINE data
    rc = bcp_bind(o_hdbc3, (BYTE *) &o_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT4, 1);                                                                                                  }
                                                 if (rc != SUCCEED)
                                                                }                                   HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &o_d_id, 0,
SQL_VARLEN_DATA, NULL, 0, SQLINT2, 2);                          //=====================================================
                                                 if (rc != SUCCEED)==============
                                                                //                                  HandleErrorDBC(o_hdbc3);
                                                                // Function   : GetPermutation
```

```
//
//=====================================================
=================

void GetPermutation(int perm[], int n)
{
                                    int i, r, t;

                                    for (i=1;i<=n;i++)
                                                perm[i] = i;

                                    for (i=1;i<=n;i++)
                                          {
                                                r = RandomNumber(i,n);
                                                t = perm[i];
                                                perm[i] = perm[r];
                                                perm[r] = t;
                                          }
}

//=====================================================
=================
//
// Function   : CheckForCommit
//
//=====================================================
=================

void CheckForCommit(HDBC hdbc,

{
    long                                    time_end, time_diff;
                                    // DBINT                                          rcint;

    if ( !(rows_loaded % aptr->batch) )
    {
                                          // rcint = bcp_batch(hdbc);
                                                // if (rcint < 0)
                                                //          HandleErro
                                          time_end = (TimeNow() / MILLI);
                                          time_diff = time_end - *time_start;

                              printf("-> Loaded %ld rows into %s in %ld sec - Total = %d (%.2f rps)\n",
                                                                              aptr->batc
                                                                              table_name
                                                                              rows_loade

                                          *time_start = time_end;
    }

    return;
}

//=====================================================
=================
//
// Function   : OpenConnections
//
//=====================================================
=================

void OpenConnections()
{

                                    RETCODE          rc;

                                          char
                                          char
                                    SQLSMALLINT

                  SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &henv);

            SQLSetEnvAttr(henv, SQL_ATTR_ODBC_VERSION, (void*)SQL_OV_ODBC3, 0 );

                        SQLAllocHandle(SQL_HANDLE_DBC, henv , &i_hdbc1);
                        SQLAllocHandle(SQL_HANDLE_DBC, henv , &w_hdbc1);
                        SQLAllocHandle(SQL_HANDLE_DBC, henv , &c_hdbc1);
                        SQLAllocHandle(SQL_HANDLE_DBC, henv , &c_hdbc2);
                        SQLAllocHandle(SQL_HANDLE_DBC, henv , &o_hdbc1);
                        SQLAllocHandle(SQL_HANDLE_DBC, henv , &o_hdbc2);
                        SQLAllocHandle(SQL_HANDLE_DBC, henv , &o_hdbc3);
```

```
SQLSetConnectAttr(i_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_
SQLSetConnectAttr(w_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_
SQLSetConnectAttr(c_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_
SQLSetConnectAttr(c_hdbc2, SQL_COPT_SS_BCP, (void *)SQL_
SQLSetConnectAttr(o_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_
SQLSetConnectAttr(o_hdbc2, SQL_COPT_SS_BCP, (void *)SQL_
SQLSetConnectAttr(o_hdbc3, SQL_COPT_SS_BCP, (void *)SQL_

                        // Open connections to SQL Serve

                              // Connection 1

sprintf( szDriverString , "DRIVER={SQL Server};SERVER=%s;U

      rc = SQLSetConnectOption (i_hdbc1, SQL_PACKET_SIZE
                              if (rc != SUCCEED)

                        rc = SQLDriverConnect ( i_hdbc1

                              if (rc != SUCCEED)

                              // Connection 2

sprintf( szDriverString , "DRIVER={SQL Server};SERVER=%s;U

      rc = SQLSetConnectOption (w_hdbc1, SQL_PACKET_SIZE
                              if (rc != SUCCEED)

                        rc = SQLDriverConnect ( w_hdbc1

                              if (rc != SUCCEED)

                              // Connection 3

sprintf( szDriverString , "DRIVER={SQL Server};SERVER=%s;U

      rc = SQLSetConnectOption (c_hdbc1, SQL_PACKET_SIZE
                              if (rc != SUCCEED)

                        rc = SQLDriverConnect ( c_hdbc1

                              if (rc != SUCCEED)

                                    cbDriverSt
                              // Connection 4
sprintf( szDriverString , "DRIVER={SQL Server};SERVER=%s;U

      rc = SQLSetConnectOption (c_hdbc2, SQL_PACKET_SIZE
                              if (rc != SUCCEED)

                        rc = SQLDriverConnect ( c_hdbc2
```



```
                                                                                printf("Starting index creation:  %s\n",in

                                                                              sprintf(cmd, "isql -S%s -U%s -P%s -e -i%s\\%s.sq



                                        if (rc != SUCCEED)
                                                                                    HandleErrorDBC(c_hdbc2);

                                         // Connection 5                                                        system(cmd);

                sprintf( szDriverString , "DRIVER={SQL Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                                                                                                    printf("Finished index creation:  %s\n",in
                                                    }


                                                    void HandleErrorDBC (SQLHDBC  hdbc1)
                                                    {
                        rc = SQLSetConnectOption (o_hdbc1, SQL_PACKET_SIZE, aptr->pack_size);
                                        if (rc != SUCCEED)                                                  SQLCHAR
                                                                                    HandleErrorDBC(o_hdbc1);  SQLINTEGER  NativeError;
                                                                                                            SQLSMALLINT i, MsgLen;
                                    rc = SQLDriverConnect ( o_hdbc1,                                          SQLRETURN   rc2;
                                                                                                                 char
                                                                                                                 char
                                                                                                                 FILE

                                                                                                               i = 1;
                                                                                  while (( rc2 = SQLGetDiagRec(SQL_HANDLE_DBC , hdbc1, i,

                                        if (rc != SUCCEED)                                                        {
                                                                                    HandleErrorDBC(o_hdbc1);

                                         // Connection 6

                sprintf( szDriverString , "DRIVER={SQL Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,

                                                                                                                printf( "[%s



                        rc = SQLSetConnectOption (o_hdbc2, SQL_PACKET_SIZE, aptr->pack_size);
                                        if (rc != SUCCEED)
                                                                                    HandleErrorDBC(o_hdbc2);

                                    rc = SQLDriverConnect ( o_hdbc2,




                                                                                                               }
                                                    }

                                        if (rc != SUCCEED)
                                                    void FormatDate ( char* szTimeChar, SQLHDBC hdbc1)  HandleErrorDBC(o_hdbc2);
                                                    {
                                         // Connection 7
                                                                                                  struct tm when;
                sprintf( szDriverString , "DRIVER={SQL Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,    time_t now;

                                                                                                   time( &now );
                                                                                             when  = *localtime( &now );

                                                                                                  mktime( &when );
                        rc = SQLSetConnectOption (o_hdbc3, SQL_PACKET_SIZE, aptr->pack_size);
                                        if (rc != SUCCEED)                                     // odbc datetime format
                                                                                  strftime( szTimeChar, 30 , "%Y-%m-%d %H:%M:
                                                                                    HandleErrorDBC(o_hdbc2);

                                    rc = SQLDriverConnect ( o_hdbc3,                                        return;

                                                    }
```

Tpcc.h

```
                                                     //                                                  File:
                                        if (rc != SUCCEED)
                                                     //
                                                     //                                 HandleErrorDBC(o_hdbc3);
                                                     //                                                Purpose:
}

//=====================================================          // Build number of TPC Benchmark Kit
==================                                             #define TPCKIT_VER                                "4.00"
//
// Function name: BuildIndex                                   // General headers
//                                                             #include <windows.h>
//=====================================================          #include <winbase.h>
==================                                             #include <stdlib.h>
                                                               #include <stdio.h>
void BuildIndex(char                          *index_script)   #include <process.h>
{                                                              #include <stddef.h>
                                                    char       #include <stdarg.h>                        cmd[256];
```

```
#include <string.h>
#include <time.h>
#include <sys\timeb.h>
#include <sys\types.h>

// ODBC headers
#include <sql.h>
#include <sqlext.h>
#include <odbcss.h>

// General constants
#define MILLI
#define FALSE
#define TRUE
#define UNDEF
#define MINPRINTASCII
#define MAXPRINTASCII

// Default environment constants
#define SERVER
#define DATABASE
#define USER
#define PASSWORD

// Default loader arguments
#define BATCH
#define DEFLDPACKSIZE
#define ORDERS_PER_DIST
#define LOADER_RES_FILE
#define LOADER_NURAND_C
#define DEF_STARTING_WAREHOUSE
#define BUILD_INDEX
#define INDEX_ORDER
#define SCALE_DOWN
#define INDEX_SCRIPT_PATH                    "scripts"

typedef struct
{
    char
    char
    char
    char
                                                BOOL
                                                BOOL
                                                BOOL
                                                BOOL
                                                BOOL
    long
    long
    long
                                                long
                                                char
                                                char
                                                long
                                                long
                                                long
                                                long
                                                long
                                                char
} TPCCLDR_ARGS;

// String length constants
#define SERVER_NAME_LEN                 20
#define DATABASE_NAME_LEN               20
#define USER_NAME_LEN                   20
#define PASSWORD_LEN                    20
#define TABLE_NAME_LEN                  20
#define I_DATA_LEN                      50
#define I_NAME_LEN                      24
#define BRAND_LEN                        1
#define LAST_NAME_LEN                   16
#define W_NAME_LEN                      10
#define ADDRESS_LEN                     20
#define STATE_LEN                        2
#define ZIP_LEN                          9
#define S_DIST_LEN                      24
#define S_DATA_LEN                      50
#define D_NAME_LEN                      10
#define FIRST_NAME_LEN                  16
#define MIDDLE_NAME_LEN                  2
#define PHONE_LEN                       16
#define CREDIT_LEN                       2
#define C_DATA_LEN                     500
#define H_DATA_LEN                      24
#define DIST_INFO_LEN                   24
#define MAX_OL_NEW_ORDER_ITEMS          15
#define MAX_OL_ORDER_STATUS_ITEMS       15
#define STATUS_LEN                      25
      #define OL_DIST_INFO_LEN                                          24
        #define C_SINCE_LEN
         #define H_DATE_LEN
     #define OL_DELIVERY_D_LEN                                          23
#define O_ENTRY_D_LEN                   23

// Functions in random.c
void                                                    seed();
long                                                   irand();
double                                                 drand();
void                                                 WUCreate();
short                                                 WURand();
long                                    RandomNumber(long lower, long upp

// Functions in getargs.c;
void                                                1000ArgsLoader();
void                                               GetArgsLoaderUsage();
                                                                  1

// Functions in time.c
long                                                    TimeNow();   32
                                                                    126

// Functions in strings.c
void                                                  MakeAddress();
void                                                   LastName();""
int                                                  MakeAlphaString();
int                                                MakeOriginalAlphaString();
int                                                   MakeNumberString();
int                                                 MakeZipNumberString();
void                                                     InitString();
void                                                    InitAddress();
void                                                     PaddString();2768
                                                                 3000
                                                "logs\\load.out"
                                                                    123
                                                                      1
```

Time.c

```
//        File:               TIME.C
//                                                Microsoft TPC-C
Kit Ver. 4.00
//                                                Copyright
Microsoft, 1996, 1997, 1998
//        Purpose:  Source file for time functions


// Includes
#include "tpcc.h"

// Globals
static long start_sec;

//=====================================================
==================
//
// Function name: TimeNow
//
//=====================================================
==================

long TimeNow()
{
   long            time_now;
   struct _timeb el_time;

#ifdef DEBUG
    printf("[%ld]DBG: Entering TimeNow()\n", (int)
GetCurrentThreadId());
#endif

   _ftime(&el_time);

   time_now = ((el_time.time - start_sec) * 1000) +
el_time.millitm;

   return time_now;
}
```

Strings.c

```
//        File:               STRINGS.C
//                                                Microsoft TPC-C
Kit Ver. 4.00
//                                                Copyright
Microsoft, 1996, 1997, 1998
//        Purpose:  Source file for database loader
                  string functions

// Includes
#include "tpcc.h"
#include <string.h>
#include <ctype.h>
```

```
//=====================================================
=================
//
// Function name: MakeAddress
//
//=====================================================
=================

void MakeAddress(char *street_1,
                                        char
*street_2,
                                        char *city,
                                        char *state,
                                        char *zip)
{

#ifdef DEBUG
    printf("[%ld]DBG: Entering MakeAddress()\n", (int)
GetCurrentThreadId());
#endif

    MakeAlphaString (10, 20, ADDRESS_LEN, street_1);
    MakeAlphaString (10, 20, ADDRESS_LEN, street_2);
    MakeAlphaString (10, 20, ADDRESS_LEN, city);
    MakeAlphaString ( 2,  2, STATE_LEN, state);
    MakeZipNumberString( 9,  9, ZIP_LEN, zip);

#ifdef DEBUG
    printf("[%ld]DBG: MakeAddress: street_1: %s,
street_2: %s, city: %s, state: %s, zip: %s\n",
                                (int)
GetCurrentThreadId(), street_1, street_2, city, state,
zip);
#endif

    return;

}


//=====================================================
=================
//
// Function name: LastName
//
//=====================================================
=================

void LastName(int  num,
                            char *name)
{
    static char *n[] =
    {
                    "BAR" , "OUGHT", "ABLE" , "PRI"  ,
"PRES",
                    "ESE" , "ANTI" , "CALLY", "ATION",
"EING"
    };

#ifdef DEBUG
    printf("[%ld]DBG: Entering LastName()\n", (int)
GetCurrentThreadId());
#endif

    if ((num >= 0) && (num < 1000))
    {
                    strcpy(name, n[(num/100)%10]);
                    strcat(name, n[(num/10)%10]);
                    strcat(name, n[(num/1)%10]);

                    if (strlen(name) < LAST_NAME_LEN)
                    {
                              PaddString(LAST_NAME_LEN,
name);
                    }

    }
    else
    {
                    printf("\nError in LastName()...
num <%ld> out of range (0,999)\n", num);
                    exit(-1);
    }


#ifdef DEBUG
    printf("[%ld]DBG: LastName: num = [%d] ==>
[%d][%d][%d]\n",
                                (int)
GetCurrentThreadId(), num, num/100, (num/10)%10,
num%10);
          printf("[%ld]DBG: LastName: String = %s\n",
(int) GetCurrentThreadId(), name);
#endif

    return;

}


//=====================================================
=================
//
// Function name: MakeAlphaString
//
//=====================================================
=================

//philipdu 08/13/96 Changed MakeAlphaString to use A-Z,
a-z, and 0-9 in
//accordance with spec see below:
//The spec says:
//4.3.2.2 The notation random a-string [x .. y]
//(respectively, n-string [x .. y]) represents a string
of random alphanumeric
//(respectively, numeric) characters of a random length
of minimum x, maximum y,
//and mean (y+x)/2. Alphanumerics are A..Z, a..z, and
0..9.  The only other
//requirement is that the character set used "must be
able to represent a minimum
//of 128 different characters".  We are using 8-bit
chars, so this is a non issue.
//It is completely unreasonable to stuff non-printing
chars into the text fields.
//-CLevine 08/13/96

int MakeAlphaString( int  x, int  y, int  z, char *str)
{
          int                len;
          int                i;
          static    char chArray[] =
"0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqr
stuvwxyz";
          static    int       chArrayMax = 61;

#ifdef DEBUG
    printf("[%ld]DBG: Entering MakeAlphaString()\n",
(int) GetCurrentThreadId());
#endif

          len= RandomNumber(x, y);

          for (i=0; i<len; i++)
                    str[i] = chArray[RandomNumber(0,
chArrayMax)];
          if ( len < z )
                    memset(str+len, ' ', z - len);
          str[len] = 0;

    return len;
}


//=====================================================
=================
//
// Function name: MakeOriginalAlphaString
//
//=====================================================
=================

int MakeOriginalAlphaString(int  x,

              int  y,

                    int  z,

              char *str,

          int  percent)
{
    int             len;
    int             val;
    int             start;

#ifdef DEBUG
    printf("[%ld]DBG: Entering
MakeOriginalAlphaString()\n", (int)
GetCurrentThreadId());
#endif

    // verify prercentage is valid
    if ((percent < 0) || (percent > 100))
```

```
    {
                    printf("MakeOrigianlAlphaString:
Invalid percentage: %d\n", percent);
                    exit(-1);
    }

    // verify string is at least 8 chars in length
    if ((x + y) <= 8)
    {
                    printf("MakeOriginalAlphaString:
string length must be >= 8\n");
                    exit(-1);
    }

    // Make Alpha String
    len = MakeAlphaString(x,y, z, str);

    val = RandomNumber(1,100);
    if (val <= percent)
    {
                    start = RandomNumber(0, len - 8);
                    strncpy(str + start, "ORIGINAL",
8);
    }


#ifdef DEBUG
    printf("[%ld]DBG: MakeOriginalAlphaString: : %s\n",
                                (int)
GetCurrentThreadId(), str);
#endif

    return strlen(str);
}


//=====================================================
==================
//
// Function name: MakeNumberString
//
//=====================================================
==================
int MakeNumberString(int  x, int  y, int  z, char *str)
{
          char tmp[16];

          //MakeNumberString is always called
MakeZipNumberString(16, 16, 16, string)

          memset(str, '0', 16);
          itoa(RandomNumber(0, 99999999), tmp, 10);
          memcpy(str, tmp, strlen(tmp));

          itoa(RandomNumber(0, 99999999), tmp, 10);
          memcpy(str+8, tmp, strlen(tmp));

          str[16] = 0;

    return 16;
}


//=====================================================
==================
//
// Function name: MakeZipNumberString
//
//=====================================================
==================
int MakeZipNumberString(int  x, int  y, int  z, char
*str)
{
          char tmp[16];

          //MakeZipNumberString is always called
MakeZipNumberString(9, 9, 9, string)

          strcpy(str, "000011111");

          itoa(RandomNumber(0, 9999), tmp, 10);
          memcpy(str, tmp, strlen(tmp));

    return 9;
}


//=====================================================
==================
//
// Function name: InitString
//
//=====================================================
==================
void InitString(char *str, int len)
{
#ifdef DEBUG
    printf("[%ld]DBG: Entering InitString()\n", (int)
GetCurrentThreadId());
#endif

          memset(str, ' ', len);
    str[len] = 0;
}


//=====================================================
==================
// Function name: InitAddress
//
// Description:
//
//=====================================================
==================

void InitAddress(char *street_1, char *street_2, char
*city, char *state, char *zip)
{
          memset(street_1, ' ', ADDRESS_LEN+1);
          memset(street_2, ' ', ADDRESS_LEN+1);
          memset(city, ' ', ADDRESS_LEN+1);

    street_1[ADDRESS_LEN+1] = 0;
    street_2[ADDRESS_LEN+1] = 0;
    city[ADDRESS_LEN+1]     = 0;

          memset(state, ' ', STATE_LEN+1);
    state[STATE_LEN+1] = 0;

          memset(zip, ' ', ZIP_LEN+1);
    zip[ZIP_LEN+1] = 0;
}


//=====================================================
==================
//
// Function name: PaddString
//
//=====================================================
==================

void PaddString(int max, char *name)
{
          int                 len;

          len = strlen(name);
          if ( len < max )
                    memset(name+len, ' ', max - len);
          name[max] = 0;

          return;
}
```

Random.c

```
 //       File:               RANDOM.C
//                                        Microsoft TPC-C
Kit Ver. 4.00
//                                        Copyright
Microsoft, 1996, 1997, 1998
//        Purpose:  Random number generation routines
for database loader


// Includes
#include "tpcc.h"
#include "math.h"

// Defines
#define A            16807
#define M       2147483647
#define Q           127773      /* M div A */
#define R             2836      /* M mod A */
#define Thread      __declspec(thread)

// Globals
long      Thread Seed = 0;       /* thread local seed
*/

/*****************************************************
************************
```



```
*
*
* random -
*
*       Implements a GOOD pseudo random number
generator.  This generator     *
*       will/should? run the complete period before
repeating.               *
*
*
* Copied from:
*
*       Random Numbers Generators: Good Ones Are Hard
to Find.                *
*       Communications of the ACM - October 1988 Volume
31 Number 10          *
*
*
* Machine Dependencies:
*
*       long must be 2 ^ 31 - 1 or greater.
*
*
*
*********************************************************
**********************/

/*********************************************************
************************
* seed - load the Seed value used in irand and drand.
Should be used before  *
*       first call to irand or drand.
*
*********************************************************
**********************/

void seed(long val)
{

#ifdef DEBUG
    printf("[%ld]DBG: Entering seed()...\n", (int)
GetCurrentThreadId());
          printf("Old Seed %ld New Seed %ld\n",Seed,
val);
#endif

    if ( val < 0 )
              val = abs(val);

    Seed = val;

}


/*********************************************************
**********************
*
*
* irand - returns a 32 bit integer pseudo random number
with a period of     *
*       1 to 2 ^ 32 - 1.
*
*
*
* parameters:
*
*       none.
*
*
*
* returns:
*
*       32 bit integer - defined as long ( see above ).
*
*
*
* side effects:
*
*       seed get recomputed.
*
*********************************************************
**********************/

long irand()
{
    register long    s;      /* copy of seed */
    register long    test;   /* test flag */
    register long    hi;     /* tmp value for speed */
    register long    lo;     /* tmp value for speed */

#ifdef DEBUG
    printf("[%ld]DBG: Entering irand()...\n", (int)
GetCurrentThreadId());
#endif

    s = Seed;
    hi = s / Q;
    lo = s % Q;

    test = A * lo - R * hi;
    if ( test > 0 )
                  Seed = test;
    else
                  Seed = test + M;

    return( Seed );
}


/*********************************************************
**********************
*
*
* drand - returns a double pseudo random number between
0.0 and 1.0.         *
*       See irand.
*
*********************************************************
**********************/
double drand()
{

#ifdef DEBUG
    printf("[%ld]DBG: Entering drand()...\n", (int)
GetCurrentThreadId());
#endif

    return( (double)irand() / 2147483647.0);
}


//=====================================================
==================
// Function   : RandomNumber
//
// Description:
//=====================================================
==================
long RandomNumber(long lower, long upper)
{
    long rand_num;

#ifdef DEBUG
    printf("[%ld]DBG: Entering RandomNumber()...\n",
(int) GetCurrentThreadId());
#endif

          if ( upper == lower )         /* pgd 08-13-96
perf enhancement */
                  return lower;

          upper++;

    if ( upper <= lower )
                  rand_num = upper;
    else
                  rand_num = lower + irand() % (upper
- lower); /* pgd 08-13-96 perf enhancement */


#ifdef DEBUG
    printf("[%ld]DBG: RandomNumber between %ld & %ld
==> %ld\n",
                                        (int)
GetCurrentThreadId(), lower, upper, rand_num);
#endif


    return rand_num;
}


#if 0

//Orginal code pgd 08/13/96

long RandomNumber(long lower,
                                              long upper)
{
    long rand_num;

#ifdef DEBUG
    printf("[%ld]DBG: Entering RandomNumber()...\n",
(int) GetCurrentThreadId());
#endif
```

```
        upper++;

    if ((upper <= lower))
                    rand_num = upper;
    else
                    rand_num = lower + irand() %
((upper > lower) ? upper - lower : upper);


#ifdef DEBUG
    printf("[%ld]DBG: RandomNumber between %ld & %ld
==> %ld\n",
                                        (int)
GetCurrentThreadId(), lower, upper, rand_num);
#endif


    return rand_num;
}
#endif

//=====================================================
==================
// Function   : NURand
//
// Description:
//=====================================================
==================
long NURand(int iConst,
              long x,
              long y,
              long C)
{
    long rand_num;

#ifdef DEBUG
    printf("[%ld]DBG: Entering NURand()...\n", (int)
GetCurrentThreadId());
#endif

    rand_num = (((RandomNumber(0,iConst) |
RandomNumber(x,y)) + C) % (y-x+1))+x;

#ifdef DEBUG
    printf("[%ld]DBG: NURand: num = %d\n", (int)
GetCurrentThreadId(), rand_num);
#endif

    return rand_num;
}
```

Getargs.c

```
//        File:               GETARGS.C
//                                      Microsoft TPC-C
Kit Ver. 4.00
//                                      Copyright
Microsoft, 1996, 1997, 1998
//        Purpose:  Source file for command line
processing


// Includes
#include "tpcc.h"

//=====================================================
==================
//
// Function name: GetArgsLoader
//
//=====================================================
==================

void GetArgsLoader(int argc, char **argv, TPCCLDR_ARGS
*pargs)
{
    int             i;
    char  *ptr;

#ifdef DEBUG
    printf("[%ld]DBG: Entering GetArgsLoader()\n",
(int) GetCurrentThreadId());
#endif

    /* init args struct with some useful values */
    pargs->server                       =  SERVER;
    pargs->user                         =  USER;
    pargs->password                     =  PASSWORD;
    pargs->database                     =  DATABASE;
    pargs->batch                        =  BATCH;
    pargs->num_warehouses               =  UNDEF;
          pargs->tables_all                       =
TRUE;
          pargs->table_item                       =
FALSE;
          pargs->table_warehouse                  =
FALSE;
          pargs->table_customer                   =
FALSE;
          pargs->table_orders                     =
FALSE;
          pargs->loader_res_file                  =
LOADER_RES_FILE;
          pargs->pack_size                        =
DEFLDPACKSIZE;
          pargs->starting_warehouse     =
DEF_STARTING_WAREHOUSE;
          pargs->build_index                      =
BUILD_INDEX;
          pargs->index_order                      =
INDEX_ORDER;
          pargs->index_script_path      =
INDEX_SCRIPT_PATH;
          pargs->scale_down                       =
SCALE_DOWN;

    /* check for zero command line args */
    if ( argc == 1 )
                    GetArgsLoaderUsage();

    for (i = 1; i < argc; ++i)
    {
                    if (argv[i][0] != '-' && argv[i][0]
!= '/')
                    {
                    printf("\nUnrecognized command");
                    GetArgsLoaderUsage();
                    exit(1);
                    }

                    ptr = argv[i];

                    switch (ptr[1])
                    {
                    case 'h':     /* Fall throught */
                    case 'H':

          GetArgsLoaderUsage();
                                        break;

                    case 'D':
                                        pargs->database
= ptr+2;
                                        break;

                    case 'P':
                                        pargs->password
= ptr+2;
                                        break;

                    case 'S':
                                        pargs->server =
ptr+2;
                                        break;

                    case 'U':
                                        pargs->user =
ptr+2;
                                        break;

                    case 'b':
                                        pargs->batch =
atol(ptr+2);
                                        break;

                    case 'W':
                                        pargs-
>num_warehouses = atol(ptr+2);
                                        break;

                    case 's':
                                        pargs-
>starting_warehouse = atol(ptr+2);
                                        break;

                    case 't':
                                        {

          pargs->tables_all = FALSE;
                                                  if
(strcmp(ptr+2,"item") == 0)
```

```
          pargs->table_item = TRUE;
                                                  else
if (strcmp(ptr+2,"warehouse") == 0)

          pargs->table_warehouse = TRUE;
                                                  else
if (strcmp(ptr+2,"customer") == 0)

          pargs->table_customer = TRUE;
                                                  else
if (strcmp(ptr+2,"orders") == 0)

          pargs->table_orders = TRUE;
                                                  else
                                                  {


          printf("\nUnrecognized command");

          GetArgsLoaderUsage();

          exit(1);
                                                  }


          break;
                                        }
                                case 'f':
                                        pargs-
>loader_res_file = ptr+2;
                                        break;

                                case 'p':
                                        pargs-
>pack_size = atol(ptr+2);
                                        break;

                                case 'i':
                                        pargs-
>build_index = atol(ptr+2);
                                        break;

                                case 'o':
                                        pargs-
>index_order = atol(ptr+2);
                                        break;

                                case 'c':
                                        pargs-
>scale_down = atol(ptr+2);
                                        break;

                        case 'd':
                                        pargs-
>index_script_path = ptr+2;
                                        break;

                        default:
                                GetArgsLoaderUsage();
                                exit(-1);
                                break;
                        }
    }

    /* check for required args */
    if (pargs->num_warehouses == UNDEF )
    {
                        printf("Number of Warehouses is
required\n");
                        exit(-2);
    }

    return;
}


//=====================================================
==================
//
// Function name: GetArgsLoaderUsage
//
//=====================================================
==================

void GetArgsLoaderUsage()
{

#ifdef DEBUG
    printf("[%ld]DBG: Entering GetArgsLoaderUsage()\n",
(int) GetCurrentThreadId());
#endif

    printf("TPCCLDR:\n\n");
            printf("Parameter
Default\n");
            printf("-------------------------------------
-------------------------\n");
    printf("-W Number of Warehouses to Load
Required \n");
    printf("-S Server
%s\n", SERVER);
    printf("-U Username
%s\n", USER);
    printf("-P Password
%s\n", PASSWORD);
    printf("-D Database
%s\n", DATABASE);
            printf("-b Batch Size
%ld\n", (long) BATCH);
            printf("-p TDS packet size
%ld\n", (long) DEFLDPACKSIZE);
            printf("-f Loader Results Output Filename
%s\n", LOADER_RES_FILE);
            printf("-s Starting Warehouse
%ld\n", (long) DEF_STARTING_WAREHOUSE);
            printf("-i Build Option (data = 0, data and
index = 1)        %ld\n", (long) BUILD_INDEX);
            printf("-o Cluster Index Build Order (before
= 1, after = 0)  %ld\n", (long) INDEX_ORDER);
            printf("-c Build Scaled Database (normal = 0,
tiny = 1)       %ld\n", (long) SCALE_DOWN);
            printf("-d Index Script Path
%s\n", INDEX_SCRIPT_PATH);
            printf("-t Table to Load
all tables \n");
    printf("   [item|warehouse|customer|orders]\n");
    printf("   Notes: \n");
    printf("     - the '-t' parameter may be included
multiple times to \n");
    printf("       specify multiple tables to be loaded
\n");
    printf("     - 'item' loads ITEM table \n");
    printf("     - 'warehouse' loads WAREHOUSE,
DISTRICT, and STOCK tables \n");
    printf("     - 'customer' loads CUSTOMER and
HISTORY tables \n");
    printf("     - 'orders' load NEW-ORDER, ORDERS,
ORDER-LINE tables \n");

            printf("\nNote:  Command line switches are
case sensitive.\n");

    exit(0);
}
```

Appendix C – Tunable Parameters

Microsoft Windows NT Server version 4.0 Tunable Parameters

```
Key Name:          SYSTEM\CurrentControlSet\Services\NDIS
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            DisplayName
  Type:            REG_SZ
  Data:            Microsoft NDIS System Driver

Value 1
  Name:            ErrorControl
  Type:            REG_DWORD
  Data:            0x1

Value 2
  Name:            Group
  Type:            REG_SZ
  Data:            NDIS

Value 3
  Name:            Start
  Type:            REG_DWORD
  Data:            0x1

Value 4
  Name:            Type
  Type:            REG_DWORD
  Data:            0x1


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\Enum
Class Name:        <NO CLASS>
Last Write Time:   5/13/99 - 7:52 PM
Value 0
  Name:            0
  Type:            REG_SZ
  Data:            Root\LEGACY_NDIS\0000

Value 1
  Name:            Count
  Type:            REG_DWORD
  Data:            0x1

Value 2
  Name:            NextInstance
  Type:            REG_DWORD
  Data:            0x1


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\MediaTypes
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM

Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM

Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM

Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\3C592
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x20596d50

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xf0ffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            3C592


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\3C597
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
```



```
  Type:            REG_DWORD
  Data:            0x70596d50

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xf0ffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            3C597


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\BONSAI
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x62110e

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            BONSAI


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\C320TNT
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x32530e

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            C320TNT


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\DE425
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x5042a310

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xf0ffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            DE425


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\DEC300
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x230a310

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            DEC300


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\DEC422
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
```

```
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x2042a310

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xf0ffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            DEC422


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\DURANGO
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x260110e

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            DURANGO


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\ELNK3EISA
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x90506d50

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xf0ffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            ELNK3EISA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\ES3210
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x12949

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            ES3210


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\F70XX
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x6690e

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            F70XX


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\FL32
```

```
Class Name:         <NO CLASS>
Last Write Time:    3/4/99 - 1:26 AM
Value 0
  Name:             Id
  Type:             REG_DWORD
  Data:             0x1010d425

Value 1
  Name:             Mask
  Type:             REG_DWORD
  Data:             0xffffffff

Value 2
  Name:             token
  Type:             REG_SZ
  Data:             FL32


Key Name:           SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\FLNK
Class Name:         <NO CLASS>
Last Write Time:    3/4/99 - 1:26 AM
Value 0
  Name:             Id
  Type:             REG_DWORD
  Data:             0x776d50

Value 1
  Name:             Mask
  Type:             REG_DWORD
  Data:             0xffffff

Value 2
  Name:             token
  Type:             REG_SZ
  Data:             FLNK


Key Name:           SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\J2577A
Class Name:         <NO CLASS>
Last Write Time:    3/4/99 - 1:26 AM
Value 0
  Name:             Id
  Type:             REG_DWORD
  Data:             0x4019f022

Value 1
  Name:             Mask
  Type:             REG_DWORD
  Data:             0xf0ffffff

Value 2
  Name:             token
  Type:             REG_SZ
  Data:             J2577A


Key Name:           SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\MAPLE
Class Name:         <NO CLASS>
Last Write Time:    10/10/96 - 4:09 PM
Value 0
  Name:             Id
  Type:             REG_DWORD
  Data:             0x160110e

Value 1
  Name:             Mask
  Type:             REG_DWORD
  Data:             0xfffffff

Value 2
  Name:             token
  Type:             REG_SZ
  Data:             MAPLE


Key Name:           SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\NE3200
Class Name:         <NO CLASS>
Last Write Time:    10/10/96 - 4:09 PM
Value 0
  Name:             Id
  Type:             REG_DWORD
  Data:             0x7cc3a

Value 1
  Name:             Mask
  Type:             REG_DWORD
  Data:             0xffffff

Value 2
  Name:             token
  Type:             REG_SZ
  Data:             NE3200
```

```
Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\NETFLEX3
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x20f1110e

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xf0ffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            NETFLEX3


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\NETFLEX3.1
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x40f1110e

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xf0ffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            NETFLEX3


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\NETFLX
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x61110e

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            NETFLX


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\NF3500
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x84633a

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            NF3500


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\NPEISA.1
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            id
  Type:            REG_DWORD
  Data:            0x2093a

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
```

```
  Data:            NPEISA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\NPEISA.2
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            id
  Type:            REG_DWORD
  Data:            0x3093a

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            NPEISA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\P1990
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x604f42

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            P1990


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\RODAN
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x63110e

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            RODAN


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\SKETHNT
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x2644d

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xfffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            SKETHNT


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\SKFENT
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x1644d

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xfffff

Value 2
```

```
  Name:            token
  Type:            REG_SZ
  Data:            SKFENT


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\SMC8232
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x80a34d

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            SMC8232


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\TLNK3E
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x9c616d50

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xf0ffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            TLNK3E


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EISA\TLNK3EISA
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x90616d50

Value 1
  Name:            Mask
  Type:            REG_DWORD
  Data:            0xf0ffffff

Value 2
  Name:            token
  Type:            REG_SZ
  Data:            TLNK3EISA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM

Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\AT1700
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x6413

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            AT1700


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\EE16MC
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x628b

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            EE16MC
```

```
Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\ElINK527
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x41

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            ElINK527


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\ELNK3MCA.1
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x627c

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            ELNK3MCA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\ELNK3MCA.2
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x627d

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            ELNK3MCA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\ELNK3MCA.3
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x61db

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            ELNK3MCA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\ELNK3MCA.4
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x62f6

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            ELNK3MCA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\ELNK3MCA.5
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x62f7

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            ELNK3MCA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\ELNKMC
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x6042
```

```
Value 1
  Name:            token
  Type:            REG_SZ
  Data:            ELNKMC


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\F30XX
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x70

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            F30XX


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\HPMCA
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x63ca

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            HPMCA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\IBMENIIN
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xffe0

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            IBMENIIN


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\IBMTOKA
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xe000

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            IBMTOKA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\IBMTOKMC
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xe001

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            IBMTOKMC


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\IRMAtrac.1
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x5c1c

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            IRMAtrac


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\IRMAtrac.2
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
```



```
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x5c1d

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            IRMAtrac


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\NCRTOK
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x100

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            NCRTOK


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\NPMCA
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x69

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            NPMCA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\OCTK16.1
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xa84

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            OCTK16


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\OCTK16.2
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xa85

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            OCTK16


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\OCTK16.3
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xa86

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            OCTK16


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\QUADENET.1
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x8f6d

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            QUADENET
```

```
Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\QUADENET.2
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x8f6a

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            QUADENET


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\SKFMNT.1
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x83

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            SKFMNT


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\SKFMNT.2
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xab

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            SKFMNT


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\STREAMER.1
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x8fa0

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            STREAMER


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\STREAMER.2
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x8fa2

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            STREAMER


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\STREAMER.3
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x8fa8

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            STREAMER


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\STREAMER.4
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x8faa
```



```
Value 1
  Name:            token
  Type:            REG_SZ
  Data:            STREAMER


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\TC$4046E
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x51

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            TC$4046E


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\UBPS
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x7012

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            UBPS


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\WAVELAN_MCA
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x6a14

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            WAVELAN_MCA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\WD8003EA
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x67c0

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            WD8003EA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\WD8003WA
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x67c2

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            WD8003WA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\WD8013EPA
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x61c8

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            WD8013EPA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\WD8013WPA
Class Name:        <NO CLASS>
```

```
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x61c9

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            WD8013WPA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM

Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\3C590
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x590010b7

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            3C590


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\3C595
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x595010b7

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            3C595


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\3C905
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x905010b7

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            3C905


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\ALANE0
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x59009004

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            ALANE0


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\AMDPCI
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x20001022

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            AMDPCI


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\DC21040
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x21011
```

```
Value 1
  Name:            token
  Type:            REG_SZ
  Data:            DC21040


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\DC21041
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x141011

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            DC21041


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\DC21140
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x91011

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            DC21140


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\DC21142
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x191011

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            DC21142


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\DEFPA
Class Name:        <NO CLASS>
Last Write Time:   10/10/96 - 4:09 PM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xf1011

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            DEFPA


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\E100BPCI
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x12298086

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            E100BPCI


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\E10PCI
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x12268086

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            E10PCI


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\LEC
Class Name:        <NO CLASS>
```

```
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x100110b6

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            LEC


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NCPF
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x111bc

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            NCPF


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.1
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xf1300e11

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            NETFLEX3


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.2
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xae320e11

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            NETFLEX3


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.3
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xae340e11

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            NETFLEX3


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.4
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xae350e11

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            NETFLEX3


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.5
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xae430e11

Value 1
  Name:            token
  Type:            REG_SZ
```

```
  Data:            NETFLEX3


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.6
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xae400e11

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            NETFLEX3


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.7
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0xf1500e11

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            NETFLEX3


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\O100PCI
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x11108d

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            O100PCI


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\OCE4XMP
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x13108d

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            OCE4XMP


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\OCTK16
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x1108d

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            OCTK16


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\RNSFDDI
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x22001112

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            RNSFDDI


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\RTL8029
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
```

```
  Data:            0x802910ec

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            RTL8029


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\SKFPNT
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x40001148

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            SKFPNT


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\SKTOKNT_PCI
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x42001148

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            SKTOKNT_PCI


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\STREAMER
Class Name:        <NO CLASS>
Last Write Time:   3/4/99 - 1:26 AM
Value 0
  Name:            Id
  Type:            REG_DWORD
  Data:            0x181014

Value 1
  Name:            token
  Type:            REG_SZ
  Data:            STREAMER


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\Parameters
Class Name:        <NO CLASS>
Last Write Time:   3/31/99 - 1:25 PM
Value 0
  Name:            ProcessorAffinityMask
  Type:            REG_DWORD
  Data:            0
```

Microsoft SQL Server version 7.0 Startup Parameters

Microsoft SQL Server was started with the following command line options

sqlservr -c -x -t3502 –g42
where

-c	Start SQL Server independently of the Microsoft Windows NT Service Control Manager.
-x	Disable the keeping of CPU time and cache-hit ration statistics.
-t3502	Prints a message to the log at the beginning and end of each checkpoint.
-g42	Reserver 42 MB for non-buffer pool allocations

Microsoft SQL Server 7.0 Configuration Parameters

```
1> 2> 3> 4> 5> 6> 7> 8> 9> 10> 11>
--  File:     VERSION.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Returns SQL Server version string
```



```
print " "
select convert(char(30), getdate(),9)
print " "

 ------------------------------
 May 13 1999  5:01:32:107PM

(1 row affected)

1> 2> 3>
select @@version



 ------------------------------------------------------------------------------
          ----------------------------------------------------------------------
          ----------------------------------------------------------------------
          ----------------------------------
 Microsoft SQL Server  7.00 - 7.00.623 (Intel X86)
          Nov 13 1998 02:37:14
          Cop
          yright (c) 1988-1998 Microsoft Corporation
          Enterprise Edition on Windo
          ws NT 4.0 (Build 1381: Service Pack 4)



(1 row affected)
1> 2>
1> 2> 3> 4> 5> 6> 7> 8> 9> 10>
--  File:     CONFIG.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.00
--            Copyright Microsoft, 1996
--  Purpose:  Collects SQL Server configuration parameters


print " "
select convert(char(30), getdate(),9)
print " "

 ------------------------------
 May 13 1999  5:01:33:140PM

(1 row affected)

1> 2> 3> DBCC execution completed. If DBCC printed error messages, contact your system administrator.
Configuration option changed. Run the RECONFIGURE statement to install.

sp_configure "show advanced",1
1> 2> reconfigure with override
1> 2> sp_configure
 name                                                                 minimum     maximum     config_value run_value
 -------------------------------------------------------------------- ----------- ----------- ------------ -----------
 affinity mask                                                              0  2147483647           15          15
 allow updates                                                              0           1            0           0
 cost threshold for parallelism                                             0       32767            5           5
 cursor threshold                                                          -1  2147483647           -1          -1
 default language                                                           0        9999            0           0
 default sortorder id                                                       0         255           50          50
 extended memory size (MB)                                                  0  2147483647            0           0
 fill factor (%)                                                            0         100            0           0
 index create memory (KB)                                                 704     1600000            0           0
 language in cache                                                          3         100            3           3
 language neutral full-text                                                 0           1            0           0
 lightweight pooling                                                        0           1            1           1
 locks                                                                   5000  2147483647            0           0
 max async IO                                                               1         255          255         255
 max degree of parallelism                                                  0          32            1           1
 max server memory (MB)                                                     4  2147483647         2950        2950
 max text repl size (B)                                                     0  2147483647        65536       65536
 max worker threads                                                        10        1024          202         202
 media retention                                                            0         365            0           0
 min memory per query (KB)                                                512  2147483647          512         512
 min server memory (MB)                                                     0  2147483647         2950        2950
 nested triggers                                                            0           1            1           1
 network packet size (B)                                                  512       65535         4096        4096
 open objects                                                               0  2147483647            0           0
 priority boost                                                             0           1            1           1
 query governor cost limit                                                  0  2147483647            0           0
 query wait (s)                                                            -1  2147483647           -1          -1
 recovery interval (min)                                                    0       32767        32767       32767
 remote access                                                              0           1            1           1
 remote login timeout (s)                                                   0  2147483647            5           5
 remote proc trans                                                          0           1            0           0
 remote query timeout (s)                                                   0  2147483647            0           0
 resource timeout (s)                                                       5  2147483647           10          10
 scan for startup procs                                                     0           1            0           0
 set working set size                                                       0           1            1           1
 show advanced options                                                      0           1            1           1
 spin counter                                                               1  2147483647        10000       10000
 time slice (ms)                                                           50        1000          100         100
```

```
 two digit year cutoff                                          1753        9999         2049         2049
 Unicode comparison style                                          0  2147483647            0            0
 Unicode locale id                                                 0  2147483647        33280        33280
 user connections                                                  0       32767          200          200
 user options                                                      0        4095            0            0

1>
```

Disk Array Configuration Parameters

```
*******************************************************
***************
     *        MYLEX Disk Array Controller -
Configuration Utility         *
     *                            Version 4.78-15
*

*******************************************************
***************

     CONFIGURATION INFORMATION OF :
     ==============================

     3 Channel - 15 Target  DAC1164P   #1    Firmware
version  5.07-0-2

     Auto Rebuild Management         : Enabled
     Storage Works Fault Management  : Disabled
     Rebuild/Add Capacity Rate       : 50
     Stripe Size                     : 64K
     Cache Segment Size              : 8K

     SCSI Transfer Parameters
     ------------------------

     Data Transfer Rate for channel 0: 40 Mhz
     Data Bus Width for channel 0    : 16 Bit
     Command Tags for channel 0      : Enabled

     Data Transfer Rate for channel 1: 40 Mhz
     Data Bus Width for channel 1    : 16 Bit
     Command Tags for channel 1      : Enabled

     Data Transfer Rate for channel 2: 40 Mhz
     Data Bus Width for channel 2    : 16 Bit
     Command Tags for channel 2      : Enabled


     Startup Parameters
     ------------------
     Spin Up Option                  : Automatic
     Number of devices per spin up   : 2
     Length of delay                 : 6 seconds
     Sequence delay                  : 6 seconds

     PHYSICAL PACK INFORMATION :
     ===========================
     Number of Packs = 5
     Pack  0 : [0:0]

     Pack  1 : [1:0]  [2:0]

     Pack  2 : [1:1]  [2:1]

     Pack  3 : [1:2]  [2:2]

     Pack  4 : [1:3]  [2:3]


     SYSTEM DRIVE INFORMATION :
     ==========================
     Number of System Drives = 2

     Sys Drv# Phy. Size   Raid Level Eff. Size  Write
Policy State
     ======== =========   ========== =========
============ =====
      0         8683 MB     7        8683 MB    Write
Thru   Online
      1        138928 MB     1       69464 MB    Write
Thru   Online

     Device Information
     ------------------
     Chnl/Targ Vendor   Model            Version  Size
State
     --------- ------   -----            -------  ----
-----
        0-0    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-0    SEAGATE  ST118202LC       0005    17366
MB  Online
        1-1    SEAGATE  ST118202LC       0005    17366
MB  Online
        1-2    SEAGATE  ST118202LC       0005    17366
MB  Online
        1-3    SEAGATE  ST118202LC       0005    17366
MB  Online
        2-0    SEAGATE  ST118202LC       0005    17366
MB  Online
        2-1    SEAGATE  ST118202LC       0005    17366
MB  Online
        2-2    SEAGATE  ST118202LC       0005    17366
MB  Online
        2-3    SEAGATE  ST118202LC       0005    17366
MB  Online



*******************************************************
***************




*******************************************************
***************
     *        MYLEX Disk Array Controller -
Configuration Utility         *
     *                            Version 4.78-15
*

*******************************************************
***************

     CONFIGURATION INFORMATION OF :
     ==============================

     3 Channel - 15 Target  DAC1164P   #2    Firmware
version  5.07-0-2

     Auto Rebuild Management         : Enabled
     Storage Works Fault Management  : Disabled
     Rebuild/Add Capacity Rate       : 50
     Stripe Size                     : 64K
     Cache Segment Size              : 8K

     SCSI Transfer Parameters
     ------------------------

     Data Transfer Rate for channel 0: 40 Mhz
     Data Bus Width for channel 0    : 16 Bit
     Command Tags for channel 0      : Enabled

     Data Transfer Rate for channel 1: 40 Mhz
     Data Bus Width for channel 1    : 16 Bit
     Command Tags for channel 1      : Enabled

     Data Transfer Rate for channel 2: 40 Mhz
     Data Bus Width for channel 2    : 16 Bit
     Command Tags for channel 2      : Enabled


     Startup Parameters
     ------------------
     Spin Up Option                  : Automatic
     Number of devices per spin up   : 2
     Length of delay                 : 6 seconds
     Sequence delay                  : 6 seconds

     PHYSICAL PACK INFORMATION :
     ===========================
     Number of Packs = 4
     Pack  0 : [0:0]  [1:0]  [2:0]  [0:1]  [1:1]  [2:1]
[0:2]  [1:2]
```

```
     Pack  1 : [2:2]  [0:3]  [1:3]  [2:3]  [0:4]  [1:4]
[2:4]  [0:5]

     Pack  2 : [1:5]  [2:5]  [0:8]  [1:8]  [2:8]  [0:9]
[1:9]  [2:9]

     Pack  3 : [0:10]  [1:10]  [2:10]  [0:11]  [1:11]
[2:11]  [0:12]  [1:12]


     SYSTEM DRIVE INFORMATION :
     ==========================
     Number of System Drives = 1

     Sys Drv# Phy. Size   Raid Level Eff. Size  Write
Policy State
     ======== =========   ========== =========
============ =====
      0        277856 MB     0       277856 MB    Write
Thru   Online

     Device Information
     ------------------
     Chnl/Targ Vendor   Model            Version  Size
State
     --------- ------   -----            -------  ----
-----
        0-0    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-1    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-2    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-3    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-4    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-5    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-8    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-9    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-10   SEAGATE  ST39102LC        0005     8683
MB  Online
        0-11   SEAGATE  ST39102LC        0005     8683
MB  Online
        0-12   SEAGATE  ST39102LC        0005     8683
MB  Online
        1-0    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-1    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-2    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-3    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-4    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-5    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-8    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-9    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-10   SEAGATE  ST39102LC        0005     8683
MB  Online
        1-11   SEAGATE  ST39102LC        0005     8683
MB  Online
        1-12   SEAGATE  ST39102LC        0005     8683
MB  Online
        2-0    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-1    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-2    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-3    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-4    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-5    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-8    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-9    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-10   SEAGATE  ST39102LC        0005     8683
MB  Online
        2-11   SEAGATE  ST39102LC        0005     8683
MB  Online
```

```
*******************************************************
***************


*******************************************************
***************
     *        MYLEX Disk Array Controller -
Configuration Utility        *
     *                           Version 4.78-15
*
*******************************************************
***************

     CONFIGURATION INFORMATION OF :
     ==============================

     3 Channel - 15 Target  DAC1164P   #3    Firmware
version  5.07-0-2

     Auto Rebuild Management         : Enabled
     Storage Works Fault Management  : Disabled
     Rebuild/Add Capacity Rate       : 50
     Stripe Size                     : 64K
     Cache Segment Size              : 8K

     SCSI Transfer Parameters
     ------------------------

     Data Transfer Rate for channel 0: 40 Mhz
     Data Bus Width for channel 0    : 16 Bit
     Command Tags for channel 0      : Enabled

     Data Transfer Rate for channel 1: 40 Mhz
     Data Bus Width for channel 1    : 16 Bit
     Command Tags for channel 1      : Enabled

     Data Transfer Rate for channel 2: 40 Mhz
     Data Bus Width for channel 2    : 16 Bit
     Command Tags for channel 2      : Enabled


     Startup Parameters
     ------------------
     Spin Up Option                  : Automatic
     Number of devices per spin up   : 2
     Length of delay                 : 6 seconds
     Sequence delay                  : 6 seconds

     PHYSICAL PACK INFORMATION :
     ===========================
     Number of Packs = 4
     Pack  0 : [0:0]  [1:0]  [2:0]  [0:1]  [1:1]  [2:1]
[0:2]  [1:2]

     Pack  1 : [2:2]  [0:3]  [1:3]  [2:3]  [0:4]  [1:4]
[2:4]  [0:5]

     Pack  2 : [1:5]  [2:5]  [0:8]  [1:8]  [2:8]  [0:9]
[1:9]  [2:9]

     Pack  3 : [0:10]  [1:10]  [2:10]  [0:11]  [1:11]
[2:11]  [0:12]  [1:12]


     SYSTEM DRIVE INFORMATION :
     ==========================
     Number of System Drives = 1

     Sys Drv# Phy. Size   Raid Level Eff. Size  Write
Policy State
     ======== =========   ========== =========
============ =====
      0        277856 MB     0       277856 MB    Write
Thru   Online

     Device Information
     ------------------
     Chnl/Targ Vendor   Model            Version  Size
State
     --------- ------   -----            -------  ----
-----
        0-0    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-1    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-2    SEAGATE  ST39102LC        0005     8683
MB  Online
```

```
       0-3    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-4    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-5    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-8    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-9    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-10   SEAGATE  ST39102LC        0005     8683
MB  Online
       0-11   SEAGATE  ST39102LC        0005     8683
MB  Online
       0-12   SEAGATE  ST39102LC        0005     8683
MB  Online
       1-0    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-1    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-2    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-3    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-4    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-5    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-8    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-9    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-10   SEAGATE  ST39102LC        0005     8683
MB  Online
       1-11   SEAGATE  ST39102LC        0005     8683
MB  Online
       1-12   SEAGATE  ST39102LC        0005     8683
MB  Online
       2-0    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-1    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-2    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-3    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-4    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-5    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-8    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-9    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-10   SEAGATE  ST39102LC        0005     8683
MB  Online
       2-11   SEAGATE  ST39102LC        0005     8683
MB  Online



**********************************************************
***************



**********************************************************
***************
     *        MYLEX Disk Array Controller -
Configuration Utility         *
     *                           Version 4.78-15
*

**********************************************************
***************

     CONFIGURATION INFORMATION OF :
     ==============================

     3 Channel - 15 Target  DAC1164P   #4    Firmware
version  5.07-0-2

     Auto Rebuild Management         : Enabled
     Storage Works Fault Management  : Disabled
     Rebuild/Add Capacity Rate       : 50
     Stripe Size                     : 64K
     Cache Segment Size              : 8K

     SCSI Transfer Parameters
     ------------------------

     Data Transfer Rate for channel 0: 40 Mhz
     Data Bus Width for channel 0    : 16 Bit
     Command Tags for channel 0      : Enabled

     Data Transfer Rate for channel 1: 40 Mhz
     Data Bus Width for channel 1    : 16 Bit
     Command Tags for channel 1      : Enabled

     Data Transfer Rate for channel 2: 40 Mhz
     Data Bus Width for channel 2    : 16 Bit
     Command Tags for channel 2      : Enabled


     Startup Parameters
     ------------------
     Spin Up Option                  : Automatic
     Number of devices per spin up   : 2
     Length of delay                 : 6 seconds
     Sequence delay                  : 6 seconds

     PHYSICAL PACK INFORMATION :
     ===========================
     Number of Packs = 4
     Pack  0 : [0:0]  [1:0]  [2:0]  [0:1]  [1:1]  [2:1]
[0:2]  [1:2]

     Pack  1 : [2:2]  [0:3]  [1:3]  [2:3]  [0:4]  [1:4]
[2:4]  [0:5]

     Pack  2 : [1:5]  [2:5]  [0:8]  [1:8]  [2:8]  [0:9]
[1:9]  [2:9]

     Pack  3 : [0:10]  [1:10]  [2:10]  [0:11]  [1:11]
[2:11]  [0:12]  [1:12]


     SYSTEM DRIVE INFORMATION :
     ==========================
     Number of System Drives = 1

     Sys Drv# Phy. Size   Raid Level Eff. Size  Write
Policy State
     ======== =========   ========== =========
============ =====
      0        277856 MB     0       277856 MB    Write
Thru   Online

     Device Information
     ------------------
     Chnl/Targ Vendor   Model            Version  Size
State
     --------- ------   -----            -------  ----
-----
       0-0    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-1    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-2    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-3    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-4    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-5    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-8    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-9    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-10   SEAGATE  ST39102LC        0005     8683
MB  Online
       0-11   SEAGATE  ST39102LC        0005     8683
MB  Online
       0-12   SEAGATE  ST39102LC        0005     8683
MB  Online
       1-0    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-1    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-2    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-3    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-4    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-5    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-8    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-9    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-10   SEAGATE  ST39102LC        0005     8683
MB  Online
```

```
        1-11   SEAGATE  ST39102LC        0005     8683
MB  Online
        1-12   SEAGATE  ST39102LC        0005     8683
MB  Online
        2-0    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-1    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-2    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-3    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-4    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-5    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-8    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-9    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-10   SEAGATE  ST39102LC        0005     8683
MB  Online
        2-11   SEAGATE  ST39102LC        0005     8683
MB  Online



*******************************************************
**************



*******************************************************
**************
     *        MYLEX Disk Array Controller -
Configuration Utility         *
     *                           Version 4.78-15
*

*******************************************************
**************

     CONFIGURATION INFORMATION OF :
     ==============================

     3 Channel - 15 Target  DAC1164P   #5    Firmware
version  5.07-0-2

     Auto Rebuild Management         : Enabled
     Storage Works Fault Management  : Disabled
     Rebuild/Add Capacity Rate       : 50
     Stripe Size                     : 64K
     Cache Segment Size              : 8K

     SCSI Transfer Parameters
     ------------------------

     Data Transfer Rate for channel 0: 40 Mhz
     Data Bus Width for channel 0    : 16 Bit
     Command Tags for channel 0      : Enabled

     Data Transfer Rate for channel 1: 40 Mhz
     Data Bus Width for channel 1    : 16 Bit
     Command Tags for channel 1      : Enabled

     Data Transfer Rate for channel 2: 40 Mhz
     Data Bus Width for channel 2    : 16 Bit
     Command Tags for channel 2      : Enabled


     Startup Parameters
     ------------------
     Spin Up Option                  : Automatic
     Number of devices per spin up   : 2
     Length of delay                 : 6 seconds
     Sequence delay                  : 6 seconds

     PHYSICAL PACK INFORMATION :
     ===========================
     Number of Packs = 4
     Pack  0 : [0:0]  [1:0]  [2:0]  [0:1]  [1:1]  [2:1]
[0:2]  [1:2]

     Pack  1 : [2:2]  [0:3]  [1:3]  [2:3]  [0:4]  [1:4]
[2:4]  [0:5]

     Pack  2 : [1:5]  [2:5]  [0:8]  [1:8]  [2:8]  [0:9]
[1:9]  [2:9]

     Pack  3 : [0:10]  [1:10]  [2:10]  [0:11]  [1:11]
[2:11]  [0:12]  [1:12]


     SYSTEM DRIVE INFORMATION :
     ==========================
     Number of System Drives = 1

     Sys Drv# Phy. Size   Raid Level Eff. Size  Write
Policy State
     ======== =========   ========== =========
============ =====
      0        277856 MB     0       277856 MB    Write
Thru   Online

     Device Information
     ------------------
     Chnl/Targ Vendor   Model            Version  Size
State
     --------- ------   -----            -------  ----
-----
        0-0    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-1    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-2    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-3    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-4    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-5    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-8    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-9    SEAGATE  ST39102LC        0005     8683
MB  Online
        0-10   SEAGATE  ST39102LC        0005     8683
MB  Online
        0-11   SEAGATE  ST39102LC        0005     8683
MB  Online
        0-12   SEAGATE  ST39102LC        0005     8683
MB  Online
        1-0    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-1    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-2    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-3    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-4    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-5    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-8    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-9    SEAGATE  ST39102LC        0005     8683
MB  Online
        1-10   SEAGATE  ST39102LC        0005     8683
MB  Online
        1-11   SEAGATE  ST39102LC        0005     8683
MB  Online
        1-12   SEAGATE  ST39102LC        0005     8683
MB  Online
        2-0    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-1    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-2    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-3    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-4    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-5    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-8    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-9    SEAGATE  ST39102LC        0005     8683
MB  Online
        2-10   SEAGATE  ST39102LC        0005     8683
MB  Online
        2-11   SEAGATE  ST39102LC        0005     8683
MB  Online


*******************************************************
**************
```

```
*******************************************************
**************
     *        MYLEX Disk Array Controller -
Configuration Utility         *
     *                           Version 4.78-15
*

*******************************************************
**************

     CONFIGURATION INFORMATION OF :
     ==============================

     3 Channel - 15 Target  DAC1164P   #6    Firmware
version  5.07-0-2

     Auto Rebuild Management         : Enabled
     Storage Works Fault Management  : Disabled
     Rebuild/Add Capacity Rate       : 50
     Stripe Size                     : 64K
     Cache Segment Size              : 8K

     SCSI Transfer Parameters
     ------------------------

     Data Transfer Rate for channel 0: 40 Mhz
     Data Bus Width for channel 0    : 16 Bit
     Command Tags for channel 0      : Enabled

     Data Transfer Rate for channel 1: 40 Mhz
     Data Bus Width for channel 1    : 16 Bit
     Command Tags for channel 1      : Enabled

     Data Transfer Rate for channel 2: 40 Mhz
     Data Bus Width for channel 2    : 16 Bit
     Command Tags for channel 2      : Enabled


     Startup Parameters
     ------------------
     Spin Up Option                  : Automatic
     Number of devices per spin up   : 2
     Length of delay                 : 6 seconds
     Sequence delay                  : 6 seconds

     PHYSICAL PACK INFORMATION :
     ===========================
     Number of Packs = 4
     Pack  0 : [0:0]  [1:0]  [2:0]  [0:1]  [1:1]  [2:1]
[0:2]  [1:2]

     Pack  1 : [2:2]  [0:3]  [1:3]  [2:3]  [0:4]  [1:4]
[2:4]  [0:5]

     Pack  2 : [1:5]  [2:5]  [0:8]  [1:8]  [2:8]  [0:9]
[1:9]  [2:9]

     Pack  3 : [0:10]  [1:10]  [2:10]  [0:11]  [1:11]
[2:11]  [0:12]  [1:12]


     SYSTEM DRIVE INFORMATION :
     ===========================
     Number of System Drives = 1

     Sys Drv# Phy. Size   Raid Level Eff. Size  Write
Policy State
     ======== =========   ========== =========
============ =====
      0        277856 MB     0       277856 MB    Write
Thru   Online

     Device Information
     ------------------
     Chnl/Targ Vendor   Model            Version  Size
State
     --------- ------   -----            -------  ----
-----
       0-0    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-1    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-2    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-3    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-4    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-5    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-8    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-9    SEAGATE  ST39102LC        0005     8683
MB  Online
       0-10   SEAGATE  ST39102LC        0005     8683
MB  Online
       0-11   SEAGATE  ST39102LC        0005     8683
MB  Online
       0-12   SEAGATE  ST39102LC        0005     8683
MB  Online
       1-0    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-1    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-2    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-3    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-4    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-5    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-8    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-9    SEAGATE  ST39102LC        0005     8683
MB  Online
       1-10   SEAGATE  ST39102LC        0005     8683
MB  Online
       1-11   SEAGATE  ST39102LC        0005     8683
MB  Online
       1-12   SEAGATE  ST39102LC        0005     8683
MB  Online
       2-0    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-1    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-2    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-3    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-4    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-5    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-8    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-9    SEAGATE  ST39102LC        0005     8683
MB  Online
       2-10   SEAGATE  ST39102LC        0005     8683
MB  Online
       2-11   SEAGATE  ST39102LC        0005     8683
MB  Online


*******************************************************
**************




*******************************************************
**************
     *        MYLEX Disk Array Controller -
Configuration Utility         *
     *                           Version 4.78-15
*

*******************************************************
**************

     CONFIGURATION INFORMATION OF :
     ==============================

     3 Channel - 15 Target  DAC1164P   #7    Firmware
version  5.07-0-2

     Auto Rebuild Management         : Enabled
     Storage Works Fault Management  : Disabled
     Rebuild/Add Capacity Rate       : 50
     Stripe Size                     : 64K
     Cache Segment Size              : 8K

     SCSI Transfer Parameters
     ------------------------

     Data Transfer Rate for channel 0: 40 Mhz
     Data Bus Width for channel 0    : 16 Bit
     Command Tags for channel 0      : Enabled

     Data Transfer Rate for channel 1: 40 Mhz
```

```
     Data Bus Width for channel 1    : 16 Bit
     Command Tags for channel 1      : Enabled

     Data Transfer Rate for channel 2: 40 Mhz
     Data Bus Width for channel 2    : 16 Bit
     Command Tags for channel 2      : Enabled


     Startup Parameters
     ------------------
     Spin Up Option                  : Automatic
     Number of devices per spin up   : 2
     Length of delay                 : 6 seconds
     Sequence delay                  : 6 seconds

     PHYSICAL PACK INFORMATION :
     ===========================
     Number of Packs = 4
     Pack  0 : [0:0]  [1:0]  [2:0]  [0:1]  [1:1]  [2:1]
[0:2]  [1:2]

     Pack  1 : [2:2]  [0:3]  [1:3]  [2:3]  [0:4]  [1:4]
[2:4]  [0:5]

     Pack  2 : [1:5]  [2:5]  [0:8]  [1:8]  [2:8]  [0:9]
[1:9]  [2:9]

     Pack  3 : [0:10]  [1:10]  [2:10]  [0:11]  [1:11]
[2:11]  [0:12]  [1:12]


     SYSTEM DRIVE INFORMATION :
     ==========================
     Number of System Drives = 1

     Sys Drv# Phy. Size   Raid Level Eff. Size  Write
Policy State
     ======== =========   ========== =========
============ =====
      0        277856 MB     0       277856 MB    Write
Thru   Online

     Device Information
     ------------------
     Chnl/Targ Vendor   Model            Version  Size
State
     --------- ------   -----            -------  ----
-----
         0-0    SEAGATE  ST39102LC        0005     8683
MB  Online
         0-1    SEAGATE  ST39102LC        0005     8683
MB  Online
         0-2    SEAGATE  ST39102LC        0005     8683
MB  Online
         0-3    SEAGATE  ST39102LC        0005     8683
MB  Online
         0-4    SEAGATE  ST39102LC        0005     8683
MB  Online
         0-5    SEAGATE  ST39102LC        0005     8683
MB  Online
         0-8    SEAGATE  ST39102LC        0005     8683
MB  Online
         0-9    SEAGATE  ST39102LC        0005     8683
MB  Online
         0-10   SEAGATE  ST39102LC        0005     8683
MB  Online
         0-11   SEAGATE  ST39102LC        0005     8683
MB  Online
         0-12   SEAGATE  ST39102LC        0005     8683
MB  Online
         1-0    SEAGATE  ST39102LC        0005     8683
MB  Online
         1-1    SEAGATE  ST39102LC        0005     8683
MB  Online
         1-2    SEAGATE  ST39102LC        0005     8683
MB  Online
         1-3    SEAGATE  ST39102LC        0005     8683
MB  Online
         1-4    SEAGATE  ST39102LC        0005     8683
MB  Online
         1-5    SEAGATE  ST39102LC        0005     8683
MB  Online
         1-8    SEAGATE  ST39102LC        0005     8683
MB  Online
         1-9    SEAGATE  ST39102LC        0005     8683
MB  Online
         1-10   SEAGATE  ST39102LC        0005     8683
MB  Online
         1-11   SEAGATE  ST39102LC        0005     8683
MB  Online
         1-12   SEAGATE  ST39102LC        0005     8683
MB  Online
         2-0    SEAGATE  ST39102LC        0005     8683
MB  Online
         2-1    SEAGATE  ST39102LC        0005     8683
MB  Online
         2-2    SEAGATE  ST39102LC        0005     8683
MB  Online
         2-3    SEAGATE  ST39102LC        0005     8683
MB  Online
         2-4    SEAGATE  ST39102LC        0005     8683
MB  Online
         2-5    SEAGATE  ST39102LC        0005     8683
MB  Online
         2-8    SEAGATE  ST39102LC        0005     8683
MB  Online
         2-9    SEAGATE  ST39102LC        0005     8683
MB  Online
         2-10   SEAGATE  ST39102LC        0005     8683
MB  Online
         2-11   SEAGATE  ST39102LC        0005     8683
MB  Online



**********************************************************
**************
```

Server Hardware Configuration

```
Microsoft Diagnostics Report For \\TPCC-X5
------------------------------------------------------------
---------------

OS Version Report
------------------------------------------------------------
---------------
Microsoft (R) Windows NT (TM) Server
Version 4.0 (Build 1381: Service Pack 4) x86
Multiprocessor Free
Registered Owner: Acer Altos 21000, Acer, Inc.
Product Number: 70238-111-1111111-25498
------------------------------------------------------------
---------------

System Report
------------------------------------------------------------
---------------
System: AT/AT COMPATIBLE
Hardware Abstraction Layer: MPS 1.4 - APIC platform
BIOS Date: 01/01/97
BIOS Version: <unavailable>

Processor list:
   0:  x86 Family 6 Model 7 Stepping 2 GenuineIntel
~500 Mhz
   1:  x86 Family 6 Model 7 Stepping 2 GenuineIntel
~500 Mhz
   2:  x86 Family 6 Model 7 Stepping 2 GenuineIntel
~500 Mhz
   3:  x86 Family 6 Model 7 Stepping 2 GenuineIntel
~500 Mhz
------------------------------------------------------------
---------------

Video Display Report
------------------------------------------------------------
---------------
BIOS Date: <unavailable>
Adapter:
   Setting: 800 x 600 x 65536
            60 Hz
   Type: atirage compatible display adapter
   String: 1002-4756-3A-1002-4756
   Memory: 2 MB
   Chip Type: ATI 3D RAGE IIC PCI (GT-B3U1)
   DAC Type: ATI Internal DAC
Driver:
   Vendor: ATI Technologies Inc.
```



```
   File(s): atirage.sys, atirage.dll
   Version: 5.2.040, 4.0.0


Drives Report
--------------------------------------------------------
---------------
C:\  (Local - NTFS)  Total: 4,192,933 KB, Free:
1,522,238 KB
   Serial Number: 2C2D - B889
   Bytes per cluster: 512
   Sectors per cluster: 1
   Filename length: 255
D:\  (Local - NTFS)  Total: 4,690,976 KB, Free: 448,600
KB
   Serial Number: 2018 - 9C4B
   Bytes per cluster: 512
   Sectors per cluster: 8
   Filename length: 255
R:\  (Local - NTFS)  Total: 61,440,592 KB, Free:
23,694,072 KB
   Serial Number: 54F8 - EC2A
   Bytes per cluster: 512
   Sectors per cluster: 8
   Filename length: 255
S:\  (Local - NTFS)  Total: 61,440,592 KB, Free:
23,658,104 KB
   Serial Number: 1009 - BDFF
   Bytes per cluster: 512
   Sectors per cluster: 8
   Filename length: 255
T:\  (Local - NTFS)  Total: 61,440,592 KB, Free:
25,594,736 KB
   Serial Number: 4C18 - 9B24
   Bytes per cluster: 512
   Sectors per cluster: 8
   Filename length: 255
U:\  (Local - NTFS)  Total: 61,440,592 KB, Free:
25,656,340 KB
   Serial Number: F82A - 998E
   Bytes per cluster: 512
   Sectors per cluster: 8
   Filename length: 255
Z:\  (CDROM - CDFS) Benchmark Total: 355,538 KB, Free:
0 KB
   Serial Number: 185A - 9B7D
   Bytes per cluster: 2048
   Sectors per cluster: 1
   Filename length: 110

Memory Report
--------------------------------------------------------
---------------
Handles: 2,169
Threads: 105
Processes: 16

Physical Memory (K)
   Total: 3,407,280
   Available: 191,612
   File Cache: 16,916

Kernel Memory (K)
   Total: 14,868
   Paged: 9,580
   Nonpaged: 5,288

Commit Charge (K)
   Total: 3,075,944
   Limit: 8,514,564
   Peak: 3,079,652

Pagefile Space (K)
   Total: 5,241,856
   Total in use: 3,184
   Peak: 3,192

   C:\pagefile.sys
      Total: 1,048,576
      Total in use: 1,560
      Peak: 1,560

   D:\pagefile.sys
      Total: 4,193,280
      Total in use: 1,624
      Peak: 1,632



Services Report
--------------------------------------------------------
---------------
Alerter                                          Stopped
(Manual)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      LanmanWorkstation
Computer Browser                                 Stopped
(Manual)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      LanmanWorkstation
      LanmanServer
      LmHosts
ClipBook Server                                  Stopped
(Manual)
   C:\WINNT\system32\clipsrv.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      NetDDE
DAC960Monitor                                    Stopped
(Manual)
   \winnt\system32\gamserv\dacmon.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process, Interactive
   Service Dependencies:
      EventLog
DHCP Client (TDI)                                Stopped
(Disabled)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      Tcpip
      Afd
      NetBT
EventLog (Event log)                             Running
(Automatic)
   C:\WINNT\system32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
GAM Server Services                              Stopped
(Manual)
   C:\WINNT\SYSTEM32\GAMSERV\gamscm.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
Server                                           Running
(Automatic)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Group Dependencies:
      TDI
Workstation (NetworkProvider)                    Running
(Automatic)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Group Dependencies:
      TDI
License Logging Service                          Stopped
(Manual)
   C:\WINNT\System32\llssrv.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
TCP/IP NetBIOS Helper                            Stopped
(Manual)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Group Dependencies:
      NetworkProvider
Messenger                                        Stopped
(Manual)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
```

```
      LanmanWorkstation
      NetBios
MSDTC (MS Transactions)                    Stopped
(Manual)
   C:\WINNT\System32\msdtc.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      RPCSS
      NTLMSSP
MSSQLServer                                Stopped
(Manual)
   C:\MSSQL7\binn\sqlservr.exe
   Service Account Name: .\Administrator
   Error Severity: Normal
   Service Flags: Own Process
Network DDE (NetDDEGroup)                  Stopped
(Manual)
   C:\WINNT\system32\netdde.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      NetDDEDSDM
Network DDE DSDM                           Stopped
(Manual)
   C:\WINNT\system32\netdde.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
Net Logon (RemoteValidation)               Stopped
(Manual)
   C:\WINNT\System32\lsass.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      LanmanWorkstation
      LmHosts
NT LM Security Support Provider            Running
(Manual)
   C:\WINNT\System32\SERVICES.EXE
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
Plug and Play (PlugPlay)                   Stopped
(Manual)
   C:\WINNT\system32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
Protected Storage                          Stopped
(Manual)
   c:\winnt\system32\pstores.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process, Interactive
   Service Dependencies:
      RpcSs
Directory Replicator                       Stopped
(Manual)
   C:\WINNT\System32\lmrepl.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      LanmanWorkstation
      LanmanServer
Remote Procedure Call (RPC) Locator        Stopped
(Manual)
   C:\WINNT\System32\LOCATOR.EXE
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      LanmanWorkstation
      Rdr
Remote Procedure Call (RPC) Service        Running
(Automatic)
   C:\WINNT\system32\RpcSs.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
Schedule                                   Stopped
(Manual)
   C:\WINNT\System32\AtSvc.Exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
SNMP                                       Stopped
(Manual)
   C:\WINNT\System32\snmp.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      Tcpip
      EventLog
SNMP Trap Service                          Stopped
(Manual)
   C:\WINNT\System32\snmptrap.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      Tcpip
      EventLog
Spooler (SpoolerGroup)                     Stopped
(Manual)
   C:\WINNT\system32\spoolss.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process, Interactive
SQLServerAgent                             Stopped
(Manual)
   C:\MSSQL7\binn\sqlagent.exe
   Service Account Name: .\Administrator
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      MSSQLServer
Telephony Service                          Stopped
(Manual)
   C:\WINNT\system32\tapisrv.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
UPS                                        Stopped
(Manual)
   C:\WINNT\System32\ups.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process


Drivers Report
------------------------------------------------------------
---------------
Abiosdsk (Primary disk)                    Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
AFD Networking Support Environment (TDI)   Running
(Automatic)
   C:\WINNT\System32\drivers\afd.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Aha154x (SCSI miniport)                    Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Aha174x (SCSI miniport)                    Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
aic78xx (SCSI miniport)                    Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Always (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
ami0nt (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
amsint (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Arrow (SCSI miniport)                      Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
atapi (SCSI miniport)                      Running
(Boot)
   C:\WINNT\System32\DRIVERS\atapi.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Atdisk (Primary disk)                      Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
```

```
ati (Video)                                    Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
atirage (Video)                                Running
(System)
   System32\DRIVERS\atirage.sys
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Beep (Base)                                    Running
(System)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
BusLogic (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Busmouse (Pointer Port)                        Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Cdaudio (Filter)                               Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Cdfs (File system)                             Running
(Disabled)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
   Group Dependencies:
      SCSI CDROM Class
Cdrom (SCSI CDROM Class)                       Running
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Changer (Filter)                               Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
cirrus (Video)                                 Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Cpqarray (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
cpqfws2e (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
dac960nt (SCSI miniport)                       Running
(Boot)
   C:\WINNT\system32\drivers\dac960nt.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
dce376nt (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Delldsa (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Dell_DGX (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Disk (SCSI Class)                              Running
(Boot)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Diskperf (Filter)                              Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
DptScsi (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
dtc329x (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Intel(R) PRO NDIS Driver (NDIS)                Running
(Automatic)
   C:\WINNT\System32\drivers\E100BNT.SYS
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
et4000 (Video)                                 Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Fastfat (Boot file system)                     Running
(Disabled)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Fd16_700 (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Fd7000ex (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Fd8xx (SCSI miniport)                          Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
flashpnt (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Floppy (Primary disk)                          Running
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Ftdisk (Filter)                                Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
gamdrv (SCSI Class)                            Running
(Boot)
   C:\WINNT\System32\drivers\gamdrv.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
i8042 Keyboard and PS/2 Mouse Port Driver (Keyboard
Port) Running   (System)
   System32\DRIVERS\i8042prt.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Inport (Pointer Port)                          Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Jazzg300 (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Jazzg364 (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Jzvxl484 (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Keyboard Class Driver (Keyboard Class)         Running
(System)
   System32\DRIVERS\kbdclass.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
KSecDD (Base)                                  Running
(System)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
macdisk (Filter)                               Running
(Boot)
   C:\WINNT\System32\drivers\macdisk.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
mga (Video)                                    Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
mga_mil (Video)                                Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
mitsumi (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
mkecr5xx (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Modem (Extended base)                          Stopped
(Manual)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
```

Appendix C – Tunable Parameters

```
Mouse Class Driver (Pointer Class)            Running
(System)
   System32\DRIVERS\mouclass.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Msfs (File system)                            Running
(System)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Mup (Network)                                 Running
(Manual)
   C:\WINNT\System32\drivers\mup.sys
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Ncr53c9x (SCSI miniport)                      Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
ncr77c22 (Video)                              Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Ncrc700 (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Ncrc710 (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Microsoft NDIS System Driver (NDIS)           Running
(System)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
NetBIOS Interface (NetBIOSGroup)              Stopped
(Manual)
   C:\WINNT\System32\drivers\netbios.sys
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
   Group Dependencies:
      TDI
WINS Client(TCP/IP) (PNP_TDI)                 Running
(Automatic)
   C:\WINNT\System32\drivers\netbt.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
   Service Dependencies:
      Tcpip
NetDetect                                     Stopped
(Manual)
   C:\WINNT\system32\drivers\netdtect.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Npfs (File system)                            Running
(System)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Ntfs (File system)                            Running
(Disabled)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Null (Base)                                   Running
(System)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Oliscsi (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Parallel (Extended base)                      Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Service Dependencies:
      Parport
   Group Dependencies:
      Parallel arbitrator
Parport (Parallel arbitrator)                 Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
ParVdm (Extended base)                        Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Service Dependencies:
      Parport
   Group Dependencies:
      Parallel arbitrator
PCIDump (PCI Configuration)                   Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Pcmcia (System Bus Extender)                  Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
PnP ISA Enabler Driver (Base)                 Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
psidisp (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Ql10wnt (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
qv (Video)                                    Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Rdr (Network)                                 Running
(Manual)
   C:\WINNT\System32\drivers\rdr.sys
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
s3 (Video)                                    Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Scsiprnt (Extended base)                      Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Scsiscan (SCSI Class)                         Running
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Serial (Extended base)                        Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Sermouse (Pointer Port)                       Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Sfloppy (Primary disk)                        Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Simbad (Filter)                               Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
slcd32 (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Sparrow (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Spock (SCSI miniport)                         Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Srv (Network)                                 Running
(Manual)
   C:\WINNT\System32\drivers\srv.sys
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
symc810 (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
T128 (SCSI miniport)                          Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
T13B (SCSI miniport)                          Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
TCP/IP Service (PNP_TDI)                      Running
(Automatic)
   C:\WINNT\System32\drivers\tcpip.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
```

```
tga (Video)                                     Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
tmv1 (SCSI miniport)                            Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Ultra124 (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Ultra14f (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Ultra24f (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
update (Base)                                   Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
v7vram (Video)                                  Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
VgaSave (Video Save)                            Running
(System)
   C:\WINNT\System32\drivers\vga.sys
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
VgaStart (Video Init)                           Stopped
(System)
   C:\WINNT\System32\drivers\vga.sys
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Wd33c93 (SCSI miniport)                         Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
wd90c24a (Video)                                Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
wdvga (Video)                                   Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
weitekp9 (Video)                                Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Xga (Video)                                     Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process



IRQ and Port Report
-------------------------------------------------------
---------------
Devices                         Vector Level  Affinity
-------------------------------------------------------
---------------
MPS 1.4 - APIC platform              8     8 0x0000000f
MPS 1.4 - APIC platform              0     0 0x0000000f
MPS 1.4 - APIC platform              1     1 0x0000000f
MPS 1.4 - APIC platform              2     2 0x0000000f
MPS 1.4 - APIC platform              3     3 0x0000000f
MPS 1.4 - APIC platform              4     4 0x0000000f
MPS 1.4 - APIC platform              5     5 0x0000000f
MPS 1.4 - APIC platform              6     6 0x0000000f
MPS 1.4 - APIC platform              7     7 0x0000000f
MPS 1.4 - APIC platform              8     8 0x0000000f
MPS 1.4 - APIC platform              9     9 0x0000000f
MPS 1.4 - APIC platform             10    10 0x0000000f
MPS 1.4 - APIC platform             11    11 0x0000000f
MPS 1.4 - APIC platform             12    12 0x0000000f
MPS 1.4 - APIC platform             13    13 0x0000000f
MPS 1.4 - APIC platform             14    14 0x0000000f
MPS 1.4 - APIC platform             15    15 0x0000000f
MPS 1.4 - APIC platform             16    16 0x0000000f
MPS 1.4 - APIC platform             17    17 0x0000000f
MPS 1.4 - APIC platform             18    18 0x0000000f
MPS 1.4 - APIC platform             19    19 0x0000000f
MPS 1.4 - APIC platform             20    20 0x0000000f
MPS 1.4 - APIC platform             21    21 0x0000000f
MPS 1.4 - APIC platform             22    22 0x0000000f
MPS 1.4 - APIC platform             23    23 0x0000000f
MPS 1.4 - APIC platform             24    24 0x0000000f
MPS 1.4 - APIC platform             25    25 0x0000000f
MPS 1.4 - APIC platform             26    26 0x0000000f
MPS 1.4 - APIC platform             27    27 0x0000000f
MPS 1.4 - APIC platform             28    28 0x0000000f
MPS 1.4 - APIC platform             29    29 0x0000000f
MPS 1.4 - APIC platform             30    30 0x0000000f
MPS 1.4 - APIC platform             31    31 0x0000000f
MPS 1.4 - APIC platform             32    32 0x0000000f
MPS 1.4 - APIC platform             33    33 0x0000000f
MPS 1.4 - APIC platform             34    34 0x0000000f
MPS 1.4 - APIC platform             35    35 0x0000000f
MPS 1.4 - APIC platform             36    36 0x0000000f
MPS 1.4 - APIC platform             37    37 0x0000000f
MPS 1.4 - APIC platform             38    38 0x0000000f
MPS 1.4 - APIC platform             39    39 0x0000000f
MPS 1.4 - APIC platform             40    40 0x0000000f
MPS 1.4 - APIC platform             41    41 0x0000000f
MPS 1.4 - APIC platform             42    42 0x0000000f
MPS 1.4 - APIC platform             43    43 0x0000000f
MPS 1.4 - APIC platform             44    44 0x0000000f
MPS 1.4 - APIC platform             45    45 0x0000000f
MPS 1.4 - APIC platform             46    46 0x0000000f
MPS 1.4 - APIC platform             47    47 0x0000000f
MPS 1.4 - APIC platform             61    61 0x0000000f
MPS 1.4 - APIC platform             65    65 0x0000000f
MPS 1.4 - APIC platform             80    80 0x0000000f
MPS 1.4 - APIC platform            193   193 0x0000000f
MPS 1.4 - APIC platform            225   225 0x0000000f
MPS 1.4 - APIC platform            253   253 0x0000000f
MPS 1.4 - APIC platform            254   254 0x0000000f
MPS 1.4 - APIC platform            255   255 0x0000000f
i8042prt                             1     1 0xffffffff
i8042prt                            12    12 0xffffffff
E100B                               44    44 0x00000000
Floppy                               6     6 0x00000000
atapi                                0    14 0x00000000
dac960nt                            32    32 0x00000000
dac960nt                            32    32 0x00000000
dac960nt                            32    32 0x00000000
dac960nt                            32    32 0x00000000
dac960nt                            32    32 0x00000000
dac960nt                            32    32 0x00000000
dac960nt                            32    32 0x00000000
-------------------------------------------------------
---------------
Devices                       Physical Address  Length
-------------------------------------------------------
---------------
MPS 1.4 - APIC platform          0x00000000
0x0000000010
MPS 1.4 - APIC platform          0x00000020
0x0000000002
MPS 1.4 - APIC platform          0x00000040
0x0000000004
MPS 1.4 - APIC platform          0x00000048
0x0000000004
MPS 1.4 - APIC platform          0x00000061
0x0000000001
MPS 1.4 - APIC platform          0x00000070
0x0000000002
MPS 1.4 - APIC platform          0x00000080
0x0000000010
MPS 1.4 - APIC platform          0x00000092
0x0000000001
MPS 1.4 - APIC platform          0x000000a0
0x0000000002
MPS 1.4 - APIC platform          0x000000c0
0x0000000010
MPS 1.4 - APIC platform          0x000000f0
0x0000000010
i8042prt                         0x00000060
0x0000000001
i8042prt                         0x00000064
0x0000000001
E100B                            0x00003000
0x000000001e
Floppy                           0x000003f0
0x0000000006
Floppy                           0x000003f7
0x0000000001
atapi                            0x000001f0
0x0000000008
atapi                            0x000003f6
0x0000000001
dac960nt                         0x00004000
0x0000000080
dac960nt                         0x00005000
0x0000000080
dac960nt                         0x00006000
0x0000000080
dac960nt                         0x0000a000
0x0000000080
```

```
dac960nt                        0x0000b000
0x0000000080
dac960nt                        0x0000c000
0x0000000080
dac960nt                        0x0000d000
0x0000000080
atirage                         0x00007000
0x0000000100
atirage                         0x000003b0
0x000000000c
atirage                         0x000003c0
0x0000000020
VgaSave                         0x000003b0
0x000000000c
VgaSave                         0x000003c0
0x0000000020
VgaSave                         0x000001ce
0x0000000002



DMA and Memory Report
-------------------------------------------------------
---------------
Devices                         Channel    Port
-------------------------------------------------------
---------------
Floppy                             2     0
-------------------------------------------------------
---------------
Devices                         Physical Address
Length
-------------------------------------------------------
---------------
MPS 1.4 - APIC platform          0xfec00000  0x00000400
MPS 1.4 - APIC platform          0xfee00000  0x00000400
E100B                            0xd6100000  0x0000001e
dac960nt                         0xd0c00000  0x00000080
dac960nt                         0xd8000000  0x02000000
dac960nt                         0xd2100000  0x00000080
dac960nt                         0xdc000000  0x02000000
dac960nt                         0xd4100000  0x00000080
dac960nt                         0xe0000000  0x02000000
dac960nt                         0xe4700000  0x00000080
dac960nt                         0xee000000  0x02000000
dac960nt                         0xe6100000  0x00000080
dac960nt                         0xf2000000  0x02000000
dac960nt                         0xe8100000  0x00000080
dac960nt                         0xf6000000  0x02000000
dac960nt                         0xea100000  0x00000080
dac960nt                         0xfa000000  0x02000000
atirage                          0xe3000000  0x01000000
atirage                          0xe2100000  0x00001000
atirage                          0x000a0000  0x00020000
atirage                          0x000c0000  0x00008000
VgaSave                          0x000a0000  0x00020000



Environment Report
-------------------------------------------------------
---------------



System Environment Variables
   ComSpec=C:\WINNT\system32\cmd.exe
   NUMBER_OF_PROCESSORS=4
   OS=Windows_NT
   Os2LibPath=C:\WINNT\system32\os2\dll;
   Path=C:\WINNT\system32;C:\WINNT;C:\MSSQL7\BINN
   PROCESSOR_ARCHITECTURE=x86
   PROCESSOR_IDENTIFIER=x86 Family 6 Model 7 Stepping
2, GenuineIntel
   PROCESSOR_LEVEL=6
   PROCESSOR_REVISION=0702
   windir=C:\WINNT



Environment Variables for Current User
   include=c:\program
files\devstudio\vc\include;c:\program
files\devstudio\vc\atl\include;c:\program
files\devstudio\vc\mfc\include;c:\program
files\devstudio\vc\include;c:\program
files\devstudio\vc\atl\include;c:\program
files\devstudio\vc\mfc\include;%include%
   lib=c:\program files\devstudio\vc\lib;c:\program
files\devstudio\vc\mfc\lib;c:\program
files\devstudio\vc\lib;c:\program
files\devstudio\vc\mfc\lib;%lib%
   MSDevDir=C:\Program Files\DevStudio\SharedIDE
   path=c:\program
files\devstudio\sharedide\bin\ide;c:\program
files\devstudio\sharedide\bin;c:\program
files\devstudio\vc\bin
   TEMP=C:\TEMP
   TMP=C:\TEMP



Network Report
-------------------------------------------------------
---------------
Your Access Level: Admin & Local
Workgroup or Domain: WORKGROUP
Network Version: 4.0
LanRoot: WORKGROUP
Logged On Users: 1
Current User (1): Administrator
  Logon Domain: TPCC-X5
  Logon Server: TPCC-X5

Transport: NetBT_E100B1, 00-00-E2-1A-CC-EE, VC's: 0,
Wan: Wan

Character Wait: 3,600
Collection Time: 250
Maximum Collection Count: 16
Keep Connection: 600
Maximum Commands: 5
Session Time Out: 45
Character Buffer Size: 512
Maximum Threads: 17
Lock Quota: 6,144
Lock Increment: 10
Maximum Locks: 500
Pipe Increment: 10
Maximum Pipes: 500
Cache Time Out: 40
Dormant File Limit: 45
Read Ahead Throughput: 4,294,967,295
Mailslot Buffers: 3
Server Announce Buffers: 20
Illegal Datagrams: 5
Datagram Reset Frequency: 60
Log Election Packets: False
Use Opportunistic Locking: True
Use Unlock Behind: True
Use Close Behind: True
Buffer Pipes: True
Use Lock, Read, Unlock: True
Use NT Caching: True
Use Raw Read: True
Use Raw Write: True
Use Write Raw Data: True
Use Encryption: True
Buffer Deny Write Files: True
Buffer Read Only Files: True
Force Core Creation: True
512 Byte Max Transfer: False
Bytes Received: 35,345
SMB's Received: 162
Paged Read Bytes Requested: 0
Non Paged Read Bytes Requested: 0
Cache Read Bytes Requested: 0
Network Read Bytes Requested: 0
Bytes Transmitted: 30,982
SMB's Transmitted: 160
Paged Read Bytes Requested: 0
Non Paged Read Bytes Requested: 153,941
Cache Read Bytes Requested: 0
Network Read Bytes Requested: 149,978
Initally Failed Operations: 0
Failed Completion Operations: 0
Read Operations: 0
Random Read Operations: 0
Read SMB's: 0
Large Read SMB's: 0
Small Read SMB's: 0
Write Operations: 4
Random Write Operations: 0
Write SMB's: 4
Large Write SMB's: 3
Small Write SMB's: 0
Raw Reads Denied: 0
Raw Writes Denied: 0
Network Errors: 0
Sessions: 5
Failed Sessions: 0
Reconnects: 0
Core Connects: 0
LM 2.0 Connects: 0
LM 2.x Connects: 0
Windows NT Connects: 3
```

```
Server Disconnects: 0
Hung Sessions: 0
Use Count: 1
Failed Use Count: 0
Current Commands: 0
Server File Opens: 0
Server Device Opens: 0
Server Jobs Queued: 0
Server Session Opens: 0
Server Sessions Timed Out: 0
Server Sessions Errored Out: 0
Server Password Errors: 0
Server Permission Errors: 0
Server System Errors: 0
Server Bytes Sent: 0
Server Bytes Received: 0
Server Average Response Time: 0
Server Request Buffers Needed: 0
Server Big Buffers Needed: 0
```

Client Hardware Configuration

```
Microsoft Diagnostics Report For \\CLIENT1
------------------------------------------------------------
---------------

OS Version Report
------------------------------------------------------------
---------------
Microsoft (R) Windows NT (TM) Server
Version 4.0 (Build 1381: Service Pack 4) x86
Multiprocessor Free
Registered Owner: csbu, acer
Product Number: 21296-OEM-0123456-01235
------------------------------------------------------------
---------------

System Report
------------------------------------------------------------
---------------
System: AT/AT COMPATIBLE
Hardware Abstraction Layer: MPS 1.4 - APIC platform
BIOS Date: 03/12/99
BIOS Version: <unavailable>

Processor list:
   0:  x86 Family 6 Model 5 Stepping 2 GenuineIntel
~451 Mhz
   1:  x86 Family 6 Model 5 Stepping 2 GenuineIntel
~451 Mhz
------------------------------------------------------------
---------------

Video Display Report
------------------------------------------------------------
---------------
BIOS Date: <unavailable>
Adapter:
   Setting: 1024 x 768 x 256
            70 Hz
   Type: ati compatible display adapter
   String: ATI Graphics Accelerator
   Memory: 2 MB
   Chip Type: Mach 64 VT
   DAC Type: DAC built into ASIC
Driver:
   Vendor: ATI Technologies, Inc.
   File(s): ati.sys, ati.dll, 8514a.dll
   Version: 3.0.62, 4.0.0


Drives Report
------------------------------------------------------------
---------------
C:\  (Local - NTFS)  Total: 4,192,933 KB, Free:
3,127,794 KB
   Serial Number: 2CED - 65DE
   Bytes per cluster: 512
   Sectors per cluster: 1
   Filename length: 255

Memory Report
------------------------------------------------------------
---------------
Handles: 7,795
Threads: 130
Processes: 18

Physical Memory (K)
   Total: 523,696
   Available: 456,576
   File Cache: 16,872

Kernel Memory (K)
   Total: 21,184
   Paged: 11,008
   Nonpaged: 10,176

Commit Charge (K)
   Total: 40,228
   Limit: 1,019,812
   Peak: 41,032

Pagefile Space (K)
   Total: 524,288
   Total in use: 1,148
   Peak: 1,148

   C:\pagefile.sys
      Total: 524,288
      Total in use: 1,148
      Peak: 1,148


Services Report
------------------------------------------------------------
---------------
Alerter                                         Stopped
(Manual)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      LanmanWorkstation
Computer Browser                                Stopped
(Manual)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      LanmanWorkstation
      LanmanServer
      LmHosts
ClipBook Server                                 Stopped
(Manual)
   C:\WINNT\system32\clipsrv.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      NetDDE
DHCP Client (TDI)                               Stopped
(Disabled)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      Tcpip
      Afd
      NetBT
EventLog (Event log)                            Running
(Automatic)
   C:\WINNT\system32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
Server                                          Running
(Automatic)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Group Dependencies:
      TDI
Workstation (NetworkProvider)                   Running
(Automatic)
   C:\WINNT\System32\services.exe
```

```
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Group Dependencies:
      TDI
License Logging Service                    Stopped
(Manual)
   C:\WINNT\System32\llssrv.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
TCP/IP NetBIOS Helper                      Stopped
(Manual)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Group Dependencies:
      NetworkProvider
Messenger                                  Stopped
(Automatic)
   C:\WINNT\System32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      LanmanWorkstation
MSDTC (MS Transactions)                    Running
(Automatic)
   C:\WINNT\System32\msdtc.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process, Interactive
   Service Dependencies:
      RPCSS
Network DDE (NetDDEGroup)                  Stopped
(Manual)
   C:\WINNT\system32\netdde.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      NetDDEDSDM
Network DDE DSDM                           Stopped
(Manual)
   C:\WINNT\system32\netdde.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
Net Logon (RemoteValidation)               Stopped
(Manual)
   C:\WINNT\System32\lsass.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
   Service Dependencies:
      LanmanWorkstation
      LmHosts
NT LM Security Support Provider            Running
(Manual)
   C:\WINNT\System32\SERVICES.EXE
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
Plug and Play (PlugPlay)                   Stopped
(Manual)
   C:\WINNT\system32\services.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Shared Process
Protected Storage                          Running
(Automatic)
   c:\winnt\system32\pstores.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process, Interactive
   Service Dependencies:
      RpcSs
Directory Replicator                       Stopped
(Manual)
   C:\WINNT\System32\lmrepl.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      LanmanWorkstation
      LanmanServer
Remote Procedure Call (RPC) Locator        Stopped
(Manual)
   C:\WINNT\System32\LOCATOR.EXE
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
   Service Dependencies:
      LanmanWorkstation
      Rdr
Remote Procedure Call (RPC) Service        Running
(Automatic)
   C:\WINNT\system32\RpcSs.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
Schedule                                   Stopped
(Manual)
   C:\WINNT\System32\AtSvc.Exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
Spooler (SpoolerGroup)                     Stopped
(Manual)
   C:\WINNT\system32\spoolss.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process, Interactive
Telephony Service                          Stopped
(Manual)
   C:\WINNT\system32\tapisrv.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
TUXEDO IPC Helper                          Running
(Automatic)
   C:\TUXEDO\bin\tuxipc.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
TListen (Port: 3050)                       Stopped
(Manual)
   C:\TUXEDO\bin\slisten.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
UPS                                        Stopped
(Manual)
   C:\WINNT\System32\ups.exe
   Service Account Name: LocalSystem
   Error Severity: Normal
   Service Flags: Own Process
World Wide Web Publishing Service          Running
(Automatic)
   C:\WINNT\System32\inetsrv\inetinfo.exe
   Service Account Name: LocalSystem
   Error Severity: Ignore
   Service Flags: Shared Process
   Service Dependencies:
      RPCSS
      NTLMSSP


Drivers Report
--------------------------------------------------------
---------------
Abiosdsk (Primary disk)                    Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
AFD Networking Support Environment (TDI)   Running
(Automatic)
   C:\WINNT\System32\drivers\afd.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Aha154x (SCSI miniport)                    Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Aha174x (SCSI miniport)                    Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
aic78xx (SCSI miniport)                    Running
(Boot)
   C:\WINNT\system32\drivers\aic78xx.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Always (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
ami0nt (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
amsint (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
```

Appendix C – Tunable Parameters

```
   Service Flags: Kernel Driver, Shared Process
Arrow (SCSI miniport)                          Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
atapi (SCSI miniport)                          Running
(Boot)
   C:\WINNT\System32\DRIVERS\atapi.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Atdisk (Primary disk)                          Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
ati (Video)                                    Running
(System)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Beep (Base)                                    Running
(System)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
BusLogic (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Busmouse (Pointer Port)                        Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Cdaudio (Filter)                               Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Cdfs (File system)                             Running
(Disabled)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
   Group Dependencies:
      SCSI CDROM Class
Cdrom (SCSI CDROM Class)                       Running
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Changer (Filter)                               Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
cirrus (Video)                                 Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Cpqarray (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
cpqfws2e (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
dac960nt (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
dce376nt (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Delldsa (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Dell_DGX (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Disk (SCSI Class)                              Running
(Boot)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Diskperf (Filter)                              Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
DptScsi (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
dtc329x (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Intel(R) PRO NDIS Driver (NDIS)                Running
(Automatic)
   C:\WINNT\System32\drivers\E100BNT.SYS
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
et4000 (Video)                                 Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Fastfat (Boot file system)                     Running
(Disabled)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Fd16_700 (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Fd7000ex (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Fd8xx (SCSI miniport)                          Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
flashpnt (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Floppy (Primary disk)                          Running
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Ftdisk (Filter)                                Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
i8042 Keyboard and PS/2 Mouse Port Driver (Keyboard
Port) Running   (System)
   System32\DRIVERS\i8042prt.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Inport (Pointer Port)                          Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Jazzg300 (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Jazzg364 (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Jzvxl484 (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Keyboard Class Driver (Keyboard Class)         Running
(System)
   System32\DRIVERS\kbdclass.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
KSecDD (Base)                                  Running
(System)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
mga (Video)                                    Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
mga_mil (Video)                                Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
mitsumi (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
mkecr5xx (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Modem (Extended base)                          Stopped
(Manual)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Mouse Class Driver (Pointer Class)             Running
(System)
```

```
   System32\DRIVERS\mouclass.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Msfs (File system)                          Running
(System)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Mup (Network)                               Running
(Manual)
   C:\WINNT\System32\drivers\mup.sys
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Ncr53c9x (SCSI miniport)                    Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
ncr77c22 (Video)                            Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Ncrc700 (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Ncrc710 (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Microsoft NDIS System Driver (NDIS)         Running
(System)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
WINS Client(TCP/IP) (PNP_TDI)               Running
(Automatic)
   C:\WINNT\System32\drivers\netbt.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
   Service Dependencies:
      Tcpip
NetDetect                                   Stopped
(Manual)
   C:\WINNT\system32\drivers\netdtect.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Npfs (File system)                          Running
(System)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Ntfs (File system)                          Running
(Disabled)
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
Null (Base)                                 Running
(System)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Oliscsi (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Parallel (Extended base)                    Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Service Dependencies:
      Parport
   Group Dependencies:
      Parallel arbitrator
Parport (Parallel arbitrator)               Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
ParVdm (Extended base)                      Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Service Dependencies:
      Parport
   Group Dependencies:
      Parallel arbitrator
PCIDump (PCI Configuration)                 Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Pcmcia (System Bus Extender)                Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
PnP ISA Enabler Driver (Base)               Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
psidisp (Video)                             Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Ql10wnt (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
qv (Video)                                  Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Rdr (Network)                               Running
(Manual)
   C:\WINNT\System32\drivers\rdr.sys
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
s3 (Video)                                  Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Scsiprnt (Extended base)                    Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Scsiscan (SCSI Class)                       Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Serial (Extended base)                      Stopped
(Automatic)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Sermouse (Pointer Port)                     Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Sfloppy (Primary disk)                      Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
   Group Dependencies:
      SCSI miniport
Simbad (Filter)                             Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
slcd32 (SCSI miniport)                      Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Sparrow (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Spock (SCSI miniport)                       Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Srv (Network)                               Running
(Manual)
   C:\WINNT\System32\drivers\srv.sys
   Error Severity: Normal
   Service Flags: File System Driver, Shared Process
symc810 (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
T128 (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
T13B (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
TCP/IP Service (PNP_TDI)                    Running
(Automatic)
   C:\WINNT\System32\drivers\tcpip.sys
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
tga (Video)                                 Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
tmv1 (SCSI miniport)                        Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
```

Appendix C – Tunable Parameters

```
Ultra124 (SCSI miniport)                    Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Ultra14f (SCSI miniport)                    Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
Ultra24f (SCSI miniport)                    Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
update (Base)                               Stopped
(System)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
v7vram (Video)                              Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
VgaSave (Video Save)                        Running
(System)
   C:\WINNT\System32\drivers\vga.sys
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
VgaStart (Video Init)                       Stopped
(System)
   C:\WINNT\System32\drivers\vga.sys
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Wd33c93 (SCSI miniport)                     Stopped
(Disabled)
   Error Severity: Normal
   Service Flags: Kernel Driver, Shared Process
wd90c24a (Video)                            Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
wdvga (Video)                               Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
weitekp9 (Video)                            Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
Xga (Video)                                 Stopped
(Disabled)
   Error Severity: Ignore
   Service Flags: Kernel Driver, Shared Process
```

```
IRQ and Port Report
-------------------------------------------------------
---------------
Devices                        Vector Level  Affinity
-------------------------------------------------------
---------------
MPS 1.4 - APIC platform             8     8 0x00000003
MPS 1.4 - APIC platform             0     0 0x00000003
MPS 1.4 - APIC platform             1     1 0x00000003
MPS 1.4 - APIC platform             2     2 0x00000003
MPS 1.4 - APIC platform             3     3 0x00000003
MPS 1.4 - APIC platform             4     4 0x00000003
MPS 1.4 - APIC platform             5     5 0x00000003
MPS 1.4 - APIC platform             6     6 0x00000003
MPS 1.4 - APIC platform             7     7 0x00000003
MPS 1.4 - APIC platform             8     8 0x00000003
MPS 1.4 - APIC platform             9     9 0x00000003
MPS 1.4 - APIC platform            10    10 0x00000003
MPS 1.4 - APIC platform            11    11 0x00000003
MPS 1.4 - APIC platform            12    12 0x00000003
MPS 1.4 - APIC platform            13    13 0x00000003
MPS 1.4 - APIC platform            14    14 0x00000003
MPS 1.4 - APIC platform            15    15 0x00000003
MPS 1.4 - APIC platform            16    16 0x00000003
MPS 1.4 - APIC platform            17    17 0x00000003
MPS 1.4 - APIC platform            18    18 0x00000003
MPS 1.4 - APIC platform            19    19 0x00000003
MPS 1.4 - APIC platform            20    20 0x00000003
MPS 1.4 - APIC platform            21    21 0x00000003
MPS 1.4 - APIC platform            22    22 0x00000003
MPS 1.4 - APIC platform            23    23 0x00000003
MPS 1.4 - APIC platform            24    24 0x00000003
MPS 1.4 - APIC platform            25    25 0x00000003
MPS 1.4 - APIC platform            26    26 0x00000003
MPS 1.4 - APIC platform            27    27 0x00000003
MPS 1.4 - APIC platform            28    28 0x00000003
MPS 1.4 - APIC platform            29    29 0x00000003
MPS 1.4 - APIC platform            30    30 0x00000003
MPS 1.4 - APIC platform            31    31 0x00000003
MPS 1.4 - APIC platform            32    32 0x00000003
MPS 1.4 - APIC platform            33    33 0x00000003
MPS 1.4 - APIC platform            34    34 0x00000003
MPS 1.4 - APIC platform            35    35 0x00000003
MPS 1.4 - APIC platform            36    36 0x00000003
MPS 1.4 - APIC platform            37    37 0x00000003
MPS 1.4 - APIC platform            38    38 0x00000003
MPS 1.4 - APIC platform            39    39 0x00000003
MPS 1.4 - APIC platform            40    40 0x00000003
MPS 1.4 - APIC platform            41    41 0x00000003
MPS 1.4 - APIC platform            42    42 0x00000003
MPS 1.4 - APIC platform            43    43 0x00000003
MPS 1.4 - APIC platform            44    44 0x00000003
MPS 1.4 - APIC platform            45    45 0x00000003
MPS 1.4 - APIC platform            46    46 0x00000003
MPS 1.4 - APIC platform            47    47 0x00000003
MPS 1.4 - APIC platform            61    61 0x00000003
MPS 1.4 - APIC platform            65    65 0x00000003
MPS 1.4 - APIC platform            80    80 0x00000003
MPS 1.4 - APIC platform           193   193 0x00000003
MPS 1.4 - APIC platform           225   225 0x00000003
MPS 1.4 - APIC platform           253   253 0x00000003
MPS 1.4 - APIC platform           254   254 0x00000003
MPS 1.4 - APIC platform           255   255 0x00000003
i8042prt                            1     1 0xffffffff
i8042prt                           12    12 0xffffffff
E100B                              16    16 0x00000000
E100B                              16    16 0x00000000
E100B                              20    20 0x00000000
E100B                              16    16 0x00000000
E100B                              20    20 0x00000000
E100B                              16    16 0x00000000
E100B                              20    20 0x00000000
E100B                              20    20 0x00000000
Floppy                              6     6 0x00000000
aic78xx                            24    24 0x00000000
atapi                               0    14 0x00000000
-------------------------------------------------------
---------------
Devices                      Physical Address  Length
-------------------------------------------------------
---------------
MPS 1.4 - APIC platform         0x00000000
0x0000000010
MPS 1.4 - APIC platform         0x00000020
0x0000000002
MPS 1.4 - APIC platform         0x00000040
0x0000000004
MPS 1.4 - APIC platform         0x00000048
0x0000000004
MPS 1.4 - APIC platform         0x00000061
0x0000000001
MPS 1.4 - APIC platform         0x00000070
0x0000000002
MPS 1.4 - APIC platform         0x00000080
0x0000000010
MPS 1.4 - APIC platform         0x00000092
0x0000000001
MPS 1.4 - APIC platform         0x000000a0
0x0000000002
MPS 1.4 - APIC platform         0x000000c0
0x0000000010
MPS 1.4 - APIC platform         0x000000f0
0x0000000010
i8042prt                        0x00000060
0x0000000001
i8042prt                        0x00000064
0x0000000001
E100B                           0x00009000
0x000000001e
E100B                           0x0000b000
0x000000001e
E100B                           0x0000b040
0x000000001e
E100B                           0x0000c000
0x000000001e
E100B                           0x0000c040
0x000000001e
E100B                           0x0000d000
0x000000001e
E100B                           0x0000d040
0x000000001e
E100B                           0x00009040
0x000000001e
Floppy                          0x000003f0
0x0000000006
Floppy                          0x000003f7
0x0000000001
aic78xx                         0x0000a000
0x0000000100
atapi                           0x000001f0
0x0000000008
atapi                           0x000003f6
0x0000000001
```

Appendix C – Tunable Parameters

```
ati                             0x000003b0
0x000000000c
ati                             0x000003c0
0x0000000020
ati                             0x000003c4
0x0000000002
ati                             0x000003c5
0x0000000001
ati                             0x000003ce
0x0000000002
ati                             0x000003cf
0x0000000001
ati                             0x000001ce
0x0000000002
ati                             0x000001cf
0x0000000001
ati                             0x0000e400
0x0000000004
ati                             0x0000e404
0x0000000004
ati                             0x0000e408
0x0000000004
ati                             0x0000e40c
0x0000000004
ati                             0x0000e410
0x0000000004
ati                             0x0000e414
0x0000000004
ati                             0x0000e418
0x0000000004
ati                             0x0000e41c
0x0000000004
ati                             0x0000e440
0x0000000004
ati                             0x0000e444
0x0000000004
ati                             0x0000e448
0x0000000004
ati                             0x0000e460
0x0000000004
ati                             0x0000e464
0x0000000004
ati                             0x0000e468
0x0000000004
ati                             0x0000e46c
0x0000000004
ati                             0x0000e470
0x0000000004
ati                             0x0000e480
0x0000000004
ati                             0x0000e484
0x0000000004
ati                             0x0000e490
0x0000000004
ati                             0x0000e4a0
0x0000000004
ati                             0x0000e4b0
0x0000000004
ati                             0x0000e4b4
0x0000000004
ati                             0x0000e4b8
0x0000000004
ati                             0x0000e4c0
0x0000000004
ati                             0x0000e4c4
0x0000000004
ati                             0x0000e4d0
0x0000000004
ati                             0x0000e4dc
0x0000000004
ati                             0x0000e4e0
0x0000000004
ati                             0x0000e4e4
0x0000000004
ati                             0x0000e4e8
0x0000000004
VgaSave                         0x000003b0
0x000000000c
VgaSave                         0x000003c0
0x0000000020
VgaSave                         0x000001ce
0x0000000002



DMA and Memory Report
-------------------------------------------------------
---------------
Devices                         Channel    Port
-------------------------------------------------------
---------------
Floppy                             2     0
```

```
-------------------------------------------------------
---------------
Devices                         Physical Address
Length
-------------------------------------------------------
---------------
MPS 1.4 - APIC platform         0xfec00000  0x00000400
MPS 1.4 - APIC platform         0xfee00000  0x00000400
aic78xx                         0x21600000  0x00001000
ati                             0x000a0000  0x00028000
ati                             0x23000000  0x00800000
VgaSave                         0x000a0000  0x00020000



Environment Report
-------------------------------------------------------
---------------



System Environment Variables
   APPDIR=C:\InetPub\wwwroot
   ComSpec=C:\WINNT\system32\cmd.exe
   NUMBER_OF_PROCESSORS=2
   OS=Windows_NT
   Os2LibPath=C:\WINNT\system32\os2\dll;

Path=C:\WINNT\system32;C:\WINNT;C:\MSSQL7\BINN;C:\TUXED
O\bin
   PROCESSOR_ARCHITECTURE=x86
   PROCESSOR_IDENTIFIER=x86 Family 6 Model 5 Stepping
2, GenuineIntel
   PROCESSOR_LEVEL=6
   PROCESSOR_REVISION=0502
   TMCONTEXTS=1
   TUXCONFIG=C:\InetPub\wwwroot\tuxconfig
   TUXDIR=C:\TUXEDO
   windir=C:\WINNT



Environment Variables for Current User
   TEMP=C:\TEMP
   TMP=C:\TEMP



Network Report
-------------------------------------------------------
---------------
Your Access Level: Admin & Local
Workgroup or Domain: WORKGROUP
Network Version: 4.0
LanRoot: WORKGROUP
Logged On Users: 1
Current User (1): Administrator
  Logon Domain: CLIENT1
  Logon Server: CLIENT1

Transport: NetBT_E100B2, 00-90-27-4C-ED-7B, VC's: 0,
Wan: Wan
Transport: NetBT_E100B3, 00-90-27-4C-ED-7C, VC's: 0,
Wan: Wan
Transport: NetBT_E100B4, 00-90-27-4C-ED-7D, VC's: 0,
Wan: Wan
Transport: NetBT_E100B5, 00-90-27-4C-EB-48, VC's: 0,
Wan: Wan
Transport: NetBT_E100B6, 00-90-27-4C-EB-49, VC's: 0,
Wan: Wan
Transport: NetBT_E100B7, 00-90-27-4C-EC-76, VC's: 0,
Wan: Wan
Transport: NetBT_E100B8, 00-90-27-4C-EC-77, VC's: 0,
Wan: Wan
Transport: NetBT_E100B1, 00-90-27-4C-ED-7A, VC's: 0,
Wan: Wan

Character Wait: 3,600
Collection Time: 250
Maximum Collection Count: 16
Keep Connection: 600
Maximum Commands: 5
Session Time Out: 45
Character Buffer Size: 512
Maximum Threads: 17
Lock Quota: 6,144
Lock Increment: 10
Maximum Locks: 500
Pipe Increment: 10
Maximum Pipes: 500
Cache Time Out: 40
Dormant File Limit: 45
Read Ahead Throughput: 4,294,967,295
```

```
Mailslot Buffers: 3
Server Announce Buffers: 20
Illegal Datagrams: 5
Datagram Reset Frequency: 60
Log Election Packets: False
Use Opportunistic Locking: True
Use Unlock Behind: True
Use Close Behind: True
Buffer Pipes: True
Use Lock, Read, Unlock: True
Use NT Caching: True
Use Raw Read: True
Use Raw Write: True
Use Write Raw Data: True
Use Encryption: True
Buffer Deny Write Files: True
Buffer Read Only Files: True
Force Core Creation: True
512 Byte Max Transfer: False
Bytes Received: 111,445
SMB's Received: 306
Paged Read Bytes Requested: 73,728
Non Paged Read Bytes Requested: 8,458
Cache Read Bytes Requested: 8,458
Network Read Bytes Requested: 61,622
Bytes Transmitted: 30,235
SMB's Transmitted: 306
Paged Read Bytes Requested: 0
Non Paged Read Bytes Requested: 0
Cache Read Bytes Requested: 0
Network Read Bytes Requested: 0
Initally Failed Operations: 0
Failed Completion Operations: 0
Read Operations: 34
Random Read Operations: 0
Read SMB's: 18
Large Read SMB's: 0
Small Read SMB's: 0
Write Operations: 0
Random Write Operations: 0
Write SMB's: 0
Large Write SMB's: 0
Small Write SMB's: 0
Raw Reads Denied: 0
Raw Writes Denied: 0
Network Errors: 0
Sessions: 6
Failed Sessions: 0
Reconnects: 0
Core Connects: 0
LM 2.0 Connects: 0
LM 2.x Connects: 0
Windows NT Connects: 4
Server Disconnects: 0
Hung Sessions: 0
Use Count: 13
Failed Use Count: 0
Current Commands: 0
Server File Opens: 53
Server Device Opens: 0
Server Jobs Queued: 0
Server Session Opens: 3
Server Sessions Timed Out: 1
Server Sessions Errored Out: 1
Server Password Errors: 0
Server Permission Errors: 0
Server System Errors: 0
Server Bytes Sent: 3,141,878
Server Bytes Received: 2,553,584
Server Average Response Time: 0
Server Request Buffers Needed: 0
Server Big Buffers Needed: 0
```

Client NT Registry Parameters

InetInfo Parameters

```
Key Name:
SYSTEM\CurrentControlSet\Services\InetInfo
Class Name:        <NO CLASS>
Last Write Time:   5/12/99 - 3:48 PM

Key Name:
SYSTEM\CurrentControlSet\Services\InetInfo\Parameters
Class Name:        <NO CLASS>
Last Write Time:   5/13/99 - 3:37 PM
Value 0
  Name:            BandwidthLevel
  Type:            REG_DWORD
  Data:            0xffffffff

Value 1
  Name:            ListenBackLog
  Type:            REG_DWORD
  Data:            0x800

Value 2
  Name:            PoolThreadLimit
  Type:            REG_DWORD
  Data:            0x190

Value 3
  Name:            ThreadTimeout
  Type:            REG_DWORD
  Data:            0x1c20


Key Name:
SYSTEM\CurrentControlSet\Services\InetInfo\Parameters\F
ilter
Class Name:        <NO CLASS>
Last Write Time:   5/12/99 - 3:48 PM
Value 0
  Name:            FilterType
  Type:            REG_DWORD
  Data:            0

Value 1
  Name:            NumDenySites
  Type:            REG_DWORD
  Data:            0

Value 2
  Name:            NumGrantSites
  Type:            REG_DWORD
  Data:            0


Key Name:
SYSTEM\CurrentControlSet\Services\InetInfo\Parameters\M
imeMap
Class Name:        <NO CLASS>
Last Write Time:   5/12/99 - 3:48 PM
Value 0
  Name:            application/envoy,evy,,5
  Type:            REG_SZ
  Data:

Value 1
  Name:            application/mac-binhex40,hqx,,4
  Type:            REG_SZ
  Data:

Value 2
  Name:            application/msword,doc,,5
  Type:            REG_SZ
  Data:

Value 3
  Name:            application/msword,dot,,5
  Type:            REG_SZ
  Data:

Value 4
  Name:            application/octet-stream,*,,5
  Type:            REG_SZ
  Data:

Value 5
  Name:            application/octet-stream,bin,,5
  Type:            REG_SZ
  Data:

Value 6
  Name:            application/octet-stream,exe,,5
  Type:            REG_SZ
  Data:

Value 7
  Name:            application/oda,oda,,5
  Type:            REG_SZ
  Data:
```

```
Value 8
  Name:           application/pdf,pdf,,5
  Type:           REG_SZ
  Data:

Value 9
  Name:           application/postscript,ai,,5
  Type:           REG_SZ
  Data:

Value 10
  Name:           application/postscript,eps,,5
  Type:           REG_SZ
  Data:

Value 11
  Name:           application/postscript,ps,,5
  Type:           REG_SZ
  Data:

Value 12
  Name:           application/rtf,rtf,,5
  Type:           REG_SZ
  Data:

Value 13
  Name:           application/winhlp,hlp,,5
  Type:           REG_SZ
  Data:

Value 14
  Name:           application/x-bcpio,bcpio,,5
  Type:           REG_SZ
  Data:

Value 15
  Name:           application/x-cpio,cpio,,5
  Type:           REG_SZ
  Data:

Value 16
  Name:           application/x-csh,csh,,5
  Type:           REG_SZ
  Data:

Value 17
  Name:           application/x-director,dcr,,5
  Type:           REG_SZ
  Data:

Value 18
  Name:           application/x-director,dir,,5
  Type:           REG_SZ
  Data:

Value 19
  Name:           application/x-director,dxr,,5
  Type:           REG_SZ
  Data:

Value 20
  Name:           application/x-dvi,dvi,,5
  Type:           REG_SZ
  Data:

Value 21
  Name:           application/x-gtar,gtar,,9
  Type:           REG_SZ
  Data:

Value 22
  Name:           application/x-hdf,hdf,,5
  Type:           REG_SZ
  Data:

Value 23
  Name:           application/x-latex,latex,,5
  Type:           REG_SZ
  Data:

Value 24
  Name:           application/x-msaccess,mdb,,5
  Type:           REG_SZ
  Data:

Value 25
  Name:           application/x-mscardfile,crd,,5
  Type:           REG_SZ
  Data:

Value 26
  Name:           application/x-msclip,clp,,5
  Type:           REG_SZ
  Data:

Value 27
  Name:           application/x-msexcel,xla,,5
  Type:           REG_SZ
  Data:

Value 28
  Name:           application/x-msexcel,xlc,,5
  Type:           REG_SZ
  Data:

Value 29
  Name:           application/x-msexcel,xlm,,5
  Type:           REG_SZ
  Data:

Value 30
  Name:           application/x-msexcel,xls,,5
  Type:           REG_SZ
  Data:

Value 31
  Name:           application/x-msexcel,xlt,,5
  Type:           REG_SZ
  Data:

Value 32
  Name:           application/x-msexcel,xlw,,5
  Type:           REG_SZ
  Data:

Value 33
  Name:           application/x-msmediaview,m13,,5
  Type:           REG_SZ
  Data:

Value 34
  Name:           application/x-msmediaview,m14,,5
  Type:           REG_SZ
  Data:

Value 35
  Name:           application/x-msmetafile,wmf,,5
  Type:           REG_SZ
  Data:

Value 36
  Name:           application/x-msmoney,mny,,5
  Type:           REG_SZ
  Data:

Value 37
  Name:           application/x-mspowerpoint,ppt,,5
  Type:           REG_SZ
  Data:

Value 38
  Name:           application/x-msproject,mpp,,5
  Type:           REG_SZ
  Data:

Value 39
  Name:           application/x-mspublisher,pub,,5
  Type:           REG_SZ
  Data:

Value 40
  Name:           application/x-msterminal,trm,,5
  Type:           REG_SZ
  Data:

Value 41
  Name:           application/x-msworks,wks,,5
  Type:           REG_SZ
  Data:

Value 42
  Name:           application/x-mswrite,wri,,5
  Type:           REG_SZ
  Data:

Value 43
  Name:           application/x-netcdf,cdf,,5
  Type:           REG_SZ
  Data:

Value 44
  Name:           application/x-netcdf,nc,,5
  Type:           REG_SZ
  Data:

Value 45
```

```
  Name:           application/x-perfmon,pma,,5
  Type:           REG_SZ
  Data:

Value 46
  Name:           application/x-perfmon,pmc,,5
  Type:           REG_SZ
  Data:

Value 47
  Name:           application/x-perfmon,pml,,5
  Type:           REG_SZ
  Data:

Value 48
  Name:           application/x-perfmon,pmr,,5
  Type:           REG_SZ
  Data:

Value 49
  Name:           application/x-perfmon,pmw,,5
  Type:           REG_SZ
  Data:

Value 50
  Name:           application/x-sh,sh,,5
  Type:           REG_SZ
  Data:

Value 51
  Name:           application/x-shar,shar,,5
  Type:           REG_SZ
  Data:

Value 52
  Name:           application/x-sv4cpio,sv4cpio,,5
  Type:           REG_SZ
  Data:

Value 53
  Name:           application/x-sv4crc,sv4crc,,5
  Type:           REG_SZ
  Data:

Value 54
  Name:           application/x-tar,tar,,5
  Type:           REG_SZ
  Data:

Value 55
  Name:           application/x-tcl,tcl,,5
  Type:           REG_SZ
  Data:

Value 56
  Name:           application/x-tex,tex,,5
  Type:           REG_SZ
  Data:

Value 57
  Name:           application/x-texinfo,texi,,5
  Type:           REG_SZ
  Data:

Value 58
  Name:           application/x-texinfo,texinfo,,5
  Type:           REG_SZ
  Data:

Value 59
  Name:           application/x-troff,roff,,5
  Type:           REG_SZ
  Data:

Value 60
  Name:           application/x-troff,t,,5
  Type:           REG_SZ
  Data:

Value 61
  Name:           application/x-troff,tr,,5
  Type:           REG_SZ
  Data:

Value 62
  Name:           application/x-troff-man,man,,5
  Type:           REG_SZ
  Data:

Value 63
  Name:           application/x-troff-me,me,,5
  Type:           REG_SZ
  Data:

Value 64
  Name:           application/x-troff-ms,ms,,5
  Type:           REG_SZ
  Data:

Value 65
  Name:           application/x-ustar,ustar,,5
  Type:           REG_SZ
  Data:

Value 66
  Name:           application/x-wais-source,src,,7
  Type:           REG_SZ
  Data:

Value 67
  Name:           application/zip,zip,,9
  Type:           REG_SZ
  Data:

Value 68
  Name:           audio/basic,au,,<
  Type:           REG_SZ
  Data:

Value 69
  Name:           audio/basic,snd,,<
  Type:           REG_SZ
  Data:

Value 70
  Name:           audio/x-aiff,aif,,<
  Type:           REG_SZ
  Data:

Value 71
  Name:           audio/x-aiff,aifc,,<
  Type:           REG_SZ
  Data:

Value 72
  Name:           audio/x-aiff,aiff,,<
  Type:           REG_SZ
  Data:

Value 73
  Name:           audio/x-pn-realaudio,ram,,<
  Type:           REG_SZ
  Data:

Value 74
  Name:           audio/x-wav,wav,,<
  Type:           REG_SZ
  Data:

Value 75
  Name:           image/bmp,bmp,,:
  Type:           REG_SZ
  Data:

Value 76
  Name:           image/cis-cod,cod,,5
  Type:           REG_SZ
  Data:

Value 77
  Name:           image/gif,gif,,g
  Type:           REG_SZ
  Data:

Value 78
  Name:           image/ief,ief,,:
  Type:           REG_SZ
  Data:

Value 79
  Name:           image/jpeg,jpe,,:
  Type:           REG_SZ
  Data:

Value 80
  Name:           image/jpeg,jpeg,,:
  Type:           REG_SZ
  Data:

Value 81
  Name:           image/jpeg,jpg,,:
  Type:           REG_SZ
  Data:

Value 82
  Name:           image/tiff,tif,,:
```

```
  Type:           REG_SZ
  Data:

Value 83
  Name:           image/tiff,tiff,,:
  Type:           REG_SZ
  Data:

Value 84
  Name:           image/x-cmu-raster,ras,,:
  Type:           REG_SZ
  Data:

Value 85
  Name:           image/x-cmx,cmx,,5
  Type:           REG_SZ
  Data:

Value 86
  Name:           image/x-portable-anymap,pnm,,:
  Type:           REG_SZ
  Data:

Value 87
  Name:           image/x-portable-bitmap,pbm,,:
  Type:           REG_SZ
  Data:

Value 88
  Name:           image/x-portable-graymap,pgm,,:
  Type:           REG_SZ
  Data:

Value 89
  Name:           image/x-portable-pixmap,ppm,,:
  Type:           REG_SZ
  Data:

Value 90
  Name:           image/x-rgb,rgb,,:
  Type:           REG_SZ
  Data:

Value 91
  Name:           image/x-xbitmap,xbm,,:
  Type:           REG_SZ
  Data:

Value 92
  Name:           image/x-xpixmap,xpm,,:
  Type:           REG_SZ
  Data:

Value 93
  Name:           image/x-xwindowdump,xwd,,:
  Type:           REG_SZ
  Data:

Value 94
  Name:           text/html,htm,,h
  Type:           REG_SZ
  Data:

Value 95
  Name:           text/html,html,,h
  Type:           REG_SZ
  Data:

Value 96
  Name:           text/html,stm,,h
  Type:           REG_SZ
  Data:

Value 97
  Name:           text/plain,bas,,0
  Type:           REG_SZ
  Data:

Value 98
  Name:           text/plain,c,,0
  Type:           REG_SZ
  Data:

Value 99
  Name:           text/plain,h,,0
  Type:           REG_SZ
  Data:

Value 100
  Name:           text/plain,txt,,0
  Type:           REG_SZ
  Data:

Value 101
  Name:           text/richtext,rtx,,0
  Type:           REG_SZ
  Data:

Value 102
  Name:           text/tab-separated-values,tsv,,0
  Type:           REG_SZ
  Data:

Value 103
  Name:           text/x-setext,etx,,0
  Type:           REG_SZ
  Data:

Value 104
  Name:           video/mpeg,mpe,,;
  Type:           REG_SZ
  Data:

Value 105
  Name:           video/mpeg,mpeg,,;
  Type:           REG_SZ
  Data:

Value 106
  Name:           video/mpeg,mpg,,;
  Type:           REG_SZ
  Data:

Value 107
  Name:           video/quicktime,mov,,;
  Type:           REG_SZ
  Data:

Value 108
  Name:           video/quicktime,qt,,;
  Type:           REG_SZ
  Data:

Value 109
  Name:           video/x-msvideo,avi,,<
  Type:           REG_SZ
  Data:

Value 110
  Name:           video/x-sgi-movie,movie,,<
  Type:           REG_SZ
  Data:

Value 111
  Name:           x-world/x-vrml,flr,,5
  Type:           REG_SZ
  Data:

Value 112
  Name:           x-world/x-vrml,wrl,,5
  Type:           REG_SZ
  Data:

Value 113
  Name:           x-world/x-vrml,wrz,,5
  Type:           REG_SZ
  Data:

Value 114
  Name:           x-world/x-vrml,xaf,,5
  Type:           REG_SZ
  Data:

Value 115
  Name:           x-world/x-vrml,xof,,5
  Type:           REG_SZ
  Data:


Key Name:
SYSTEM\CurrentControlSet\Services\InetInfo\Performance
Class Name:       <NO CLASS>
Last Write Time:  5/12/99 - 3:48 PM
Value 0
  Name:           Close
  Type:           REG_SZ
  Data:           CloseINFOPerformanceData

Value 1
  Name:           Collect
  Type:           REG_SZ
  Data:           CollectINFOPerformanceData

Value 2
  Name:           First Counter
  Type:           REG_DWORD
```

```
  Data:           0x7f0

Value 3
  Name:           First Help
  Type:           REG_DWORD
  Data:           0x7f1

Value 4
  Name:           Last Counter
  Type:           REG_DWORD
  Data:           0x80e

Value 5
  Name:           Last Help
  Type:           REG_DWORD
  Data:           0x80f

Value 6
  Name:           Library
  Type:           REG_SZ
  Data:           infoctrs.DLL

Value 7
  Name:           Open
  Type:           REG_SZ
  Data:           OpenINFOPerformanceData
```

TCPIP Parameters

```
Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
Class Name:         GenericClass
Last Write Time:    5/12/99 - 4:09 PM
Value 0
  Name:           DataBasePath
  Type:           REG_EXPAND_SZ
  Data:           %SystemRoot%\System32\drivers\etc

Value 1
  Name:           Domain
  Type:           REG_SZ
  Data:

Value 2
  Name:           EnableSecurityFilters
  Type:           REG_DWORD
  Data:           0

Value 3
  Name:           ForwardBroadcasts
  Type:           REG_DWORD
  Data:           0

Value 4
  Name:           Hostname
  Type:           REG_SZ
  Data:           client1

Value 5
  Name:           IPEnableRouter
  Type:           REG_DWORD
  Data:           0

Value 6
  Name:           MaxUserPort
  Type:           REG_DWORD
  Data:           0x4e20

Value 7
  Name:           NameServer
  Type:           REG_SZ
  Data:

Value 8
  Name:           SearchList
  Type:           REG_SZ
  Data:
```

```
Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Pers
istentRoutes
Class Name:         GenericClass
Last Write Time:    4/13/99 - 11:09 PM

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Wins
ock
Class Name:         GenericClass
Last Write Time:    4/13/99 - 11:09 PM
Value 0
  Name:           HelperDllName
  Type:           REG_EXPAND_SZ
  Data:           %SystemRoot%\System32\wshtcpip.dll

Value 1
  Name:           Mapping
  Type:           REG_BINARY
  Data:
00000000   0b 00 00 00 03 00 00 00 - 02 00 00 00 01 00
00 00  ................
00000010   06 00 00 00 02 00 00 00 - 01 00 00 00 00 00
00 00  ................
00000020   02 00 00 00 00 00 00 00 - 06 00 00 00 00 00
00 00  ................
00000030   00 00 00 00 06 00 00 00 - 00 00 00 00 01 00
00 00  ................
00000040   06 00 00 00 02 00 00 00 - 02 00 00 00 11 00
00 00  ................
00000050   02 00 00 00 02 00 00 00 - 00 00 00 00 02 00
00 00  ................
00000060   00 00 00 00 11 00 00 00 - 00 00 00 00 00 00
00 00  ................
00000070   11 00 00 00 00 00 00 00 - 02 00 00 00 11 00
00 00  ................
00000080   02 00 00 00 03 00 00 00 - 00 00 00 00
............

Value 2
  Name:           MaxSockAddrLength
  Type:           REG_DWORD
  Data:           0x10

Value 3
  Name:           MinSockAddrLength
  Type:           REG_DWORD
  Data:           0x10
```

TPCC Parameters

```
Key Name:           SOFTWARE\Microsoft\TPCC
Class Name:         <NO CLASS>
Last Write Time:    5/10/99 - 6:17 PM
Value 0
  Name:           LOG
  Type:           REG_SZ
  Data:           OFF

Value 1
  Name:           MaxConnections
  Type:           REG_SZ
  Data:           8000

Value 2
  Name:           MaximumWarehouses
  Type:           REG_SZ
  Data:           1900

Value 3
  Name:           PATH
  Type:           REG_SZ
  Data:           C:\Inetpub\wwwroot\
```

Tuxedo Parameters

```
Key Name:           SOFTWARE\BEA Systems\TUXEDO
Class Name:         <NO CLASS>
Last Write Time:    4/13/99 - 5:10 PM

Key Name:           SOFTWARE\BEA Systems\TUXEDO\6.4
Class Name:         <NO CLASS>
Last Write Time:    4/13/99 - 5:11 PM

Value 0
  Name:           Company_Name
  Type:           REG_SZ
  Data:           acer

Value 1
  Name:           Install_Date
```



```
  Type:            REG_SZ
  Data:            4-13-1999

Value 2
  Name:            License_Token
  Type:            REG_DWORD
  Data:            0

Value 3
  Name:            Major_Version
  Type:            REG_DWORD
  Data:            0x6

Value 4
  Name:            Minor_Version
  Type:            REG_DWORD
  Data:            0x4

Value 5
  Name:            Serial_Number
  Type:            REG_DWORD
  Data:            0

Value 6
  Name:            User_Name
  Type:            REG_SZ
  Data:            csbu

Value 7
  Name:            Volume_Number
  Type:            REG_DWORD
  Data:            0x1


Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\All
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\All\libfml.lib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\All\libfml32.lib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\All\libgp.lib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Client
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Client\libbuft.l
ib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Client\libtux.li
b
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Client\libtux2.l
ib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Server
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Server\libbuft.l
ib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Server\libtux.li
b
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Server\libtux2.l
ib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Workstation
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Workstation\libb
uft.lib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Workstation\libn
wi.lib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Workstation\libn
ws.lib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Developer\Libraries\Workstation\libw
sc.lib
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Environment
Class Name:        <NO CLASS>
Last Write Time:   5/11/99 - 6:31 PM
Value 0
  Name:            NLSPATH
  Type:            REG_SZ
  Data:            C:\TUXEDO\locale\C

Value 1
  Name:            TUXDIR
  Type:            REG_SZ
  Data:            C:\TUXEDO

Value 2
  Name:            TUXIPC_MSG_BYTES
  Type:            REG_DWORD
  Data:            0x100000

Value 3
  Name:            TUXIPC_MSG_HDRS
  Type:            REG_DWORD
  Data:            0x1fc0

Value 4
  Name:            TUXIPC_MSG_QUEUE_BYTES
  Type:            REG_DWORD
  Data:            0x100000

Value 5
  Name:            TUXIPC_MSG_QUEUES
  Type:            REG_DWORD
  Data:            0x400

Value 6
  Name:            TUXIPC_MSG_SEG_BYTES
  Type:            REG_DWORD
  Data:            0x40

Value 7
  Name:            TUXIPC_MSG_SEGS
  Type:            REG_DWORD
  Data:            0x7fff

Value 8
  Name:            TUXIPC_PROC
```

```
  Type:            REG_DWORD
  Data:            0x400

Value 9
  Name:            TUXIPC_SEM
  Type:            REG_DWORD
  Data:            0x1000

Value 10
  Name:            TUXIPC_SEM_IDS
  Type:            REG_DWORD
  Data:            0x1000

Value 11
  Name:            TUXIPC_SEM_UNDO
  Type:            REG_DWORD
  Data:            0x1000

Value 12
  Name:            TUXIPC_SHM_PROCS
  Type:            REG_DWORD
  Data:            0x400

Value 13
  Name:            TUXIPC_SHM_SEGS
  Type:            REG_DWORD
  Data:            0x32

Value 14
  Name:            ULOGDIR
  Type:            REG_SZ
  Data:            C:\TUXEDO

Value 15
  Name:            ULOGOUT
  Type:            REG_DWORD
  Data:            0x2

Value 16
  Name:            ULOGPFX
  Type:            REG_SZ
  Data:            C:\ULOG


Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Environment\Services
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\Environment\Services\3050
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM

Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\IPCResources
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:35 PM
Value 0
  Name:            CurrentResource
  Type:            REG_SZ
  Data:            tpcc


Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\IPCResources\tpcc
Class Name:        <NO CLASS>
Last Write Time:   5/11/99 - 6:32 PM
Value 0
  Name:            TUXIPC_MSG_BYTES
  Type:            REG_DWORD
  Data:            0x100000

Value 1
  Name:            TUXIPC_MSG_HDRS
  Type:            REG_DWORD
  Data:            0x1fc0

Value 2
  Name:            TUXIPC_MSG_QUEUE_BYTES
  Type:            REG_DWORD
  Data:            0x100000

Value 3
  Name:            TUXIPC_MSG_QUEUES
  Type:            REG_DWORD
  Data:            0x400

Value 4
  Name:            TUXIPC_MSG_SEG_BYTES
  Type:            REG_DWORD
  Data:            0x40

Value 5
  Name:            TUXIPC_MSG_SEGS
  Type:            REG_DWORD
  Data:            0x7fff

Value 6
  Name:            TUXIPC_PROC
  Type:            REG_DWORD
  Data:            0x400

Value 7
  Name:            TUXIPC_SEM
  Type:            REG_DWORD
  Data:            0x1000

Value 8
  Name:            TUXIPC_SEM_IDS
  Type:            REG_DWORD
  Data:            0x1000

Value 9
  Name:            TUXIPC_SEM_UNDO
  Type:            REG_DWORD
  Data:            0x1000

Value 10
  Name:            TUXIPC_SHM_PROCS
  Type:            REG_DWORD
  Data:            0x400

Value 11
  Name:            TUXIPC_SHM_SEGS
  Type:            REG_DWORD
  Data:            0x32


Key Name:          SOFTWARE\BEA
Systems\TUXEDO\6.4\SECURITY
Class Name:        <NO CLASS>
Last Write Time:   4/13/99 - 5:12 PM
```

W3SVC Parameters

```
Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC
Class Name:        <NO CLASS>
Last Write Time:   5/12/99 - 3:48 PM
Value 0
  Name:            DependOnGroup
  Type:            REG_MULTI_SZ
  Data:

Value 1
  Name:            DependOnService
  Type:            REG_MULTI_SZ
  Data:            RPCSS
                   NTLMSSP


Value 2
  Name:            DisplayName
  Type:            REG_SZ
  Data:            World Wide Web Publishing Service

Value 3
  Name:            ErrorControl
  Type:            REG_DWORD
  Data:            0

Value 4
  Name:            ImagePath
  Type:            REG_EXPAND_SZ
  Data:
C:\WINNT\System32\inetsrv\inetinfo.exe

Value 5
  Name:            ObjectName
  Type:            REG_SZ
  Data:            LocalSystem

Value 6
  Name:            Start
  Type:            REG_DWORD
  Data:            0x2

Value 7
  Name:            Type
  Type:            REG_DWORD
```



```
  Data:           0x20


Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\Enum
Class Name:       <NO CLASS>
Last Write Time:  5/13/99 - 7:54 PM
Value 0
  Name:           0
  Type:           REG_SZ
  Data:           Root\LEGACY_W3SVC\0000

Value 1
  Name:           Count
  Type:           REG_DWORD
  Data:           0x1

Value 2
  Name:           NextInstance
  Type:           REG_DWORD
  Data:           0x1


Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\HTMLA
Class Name:       <NO CLASS>
Last Write Time:  5/12/99 - 3:48 PM

Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\Parameters
Class Name:       <NO CLASS>
Last Write Time:  5/13/99 - 1:52 PM
Value 0
  Name:           AcceptExOutstanding
  Type:           REG_DWORD
  Data:           0x800

Value 1
  Name:           AccessDeniedMessage
  Type:           REG_SZ
  Data:           Error: Access is Denied.

Value 2
  Name:           AdminEmail
  Type:           REG_SZ
  Data:           Admin@corp.com

Value 3
  Name:           AdminName
  Type:           REG_SZ
  Data:           Administrator

Value 4
  Name:           AnonymousUserName
  Type:           REG_SZ
  Data:           IUSR_CLIENT1

Value 5
  Name:           Authorization
  Type:           REG_DWORD
  Data:           0x5

Value 6
  Name:           CacheExtensions
  Type:           REG_DWORD
  Data:           0x1

Value 7
  Name:           CheckForWAISDB
  Type:           REG_DWORD
  Data:           0

Value 8
  Name:           ConnectionTimeOut
  Type:           REG_DWORD
  Data:           0x384

Value 9
  Name:           DebugFlags
  Type:           REG_DWORD
  Data:           0x8

Value 10
  Name:           Default Load File
  Type:           REG_SZ
  Data:           Default.htm

Value 11
  Name:           Dir Browse Control
  Type:           REG_DWORD
  Data:           0x4000001e

Value 12
  Name:           Filter DLLs
  Type:           REG_SZ
  Data:
C:\WINNT\System32\inetsrv\sspifilt.dll

Value 13
  Name:           GlobalExpire
  Type:           REG_DWORD
  Data:           0xffffffff

Value 14
  Name:           InstallPath
  Type:           REG_SZ
  Data:           C:\WINNT\System32\inetsrv

Value 15
  Name:           LogFileDirectory
  Type:           REG_EXPAND_SZ
  Data:           %SystemRoot%\System32\LogFiles

Value 16
  Name:           LogFileFormat
  Type:           REG_DWORD
  Data:           0

Value 17
  Name:           LogFilePeriod
  Type:           REG_DWORD
  Data:           0x1

Value 18
  Name:           LogFileTruncateSize
  Type:           REG_DWORD
  Data:           0x1388000

Value 19
  Name:           LogSqlDataSource
  Type:           REG_SZ
  Data:           HTTPLOG

Value 20
  Name:           LogSqlPassword
  Type:           REG_SZ
  Data:           sqllog

Value 21
  Name:           LogSqlTableName
  Type:           REG_SZ
  Data:           Internetlog

Value 22
  Name:           LogSqlUserName
  Type:           REG_SZ
  Data:           InternetAdmin

Value 23
  Name:           LogType
  Type:           REG_DWORD
  Data:           0

Value 24
  Name:           MajorVersion
  Type:           REG_DWORD
  Data:           0x2

Value 25
  Name:           MaxConnections
  Type:           REG_DWORD
  Data:           0x186a0

Value 26
  Name:           MinorVersion
  Type:           REG_DWORD
  Data:           0

Value 27
  Name:           NTAuthenticationProviders
  Type:           REG_SZ
  Data:           NTLM

Value 28
  Name:           ScriptTimeout
  Type:           REG_DWORD
  Data:           0x384

Value 29
  Name:           SecurePort
  Type:           REG_DWORD
  Data:           0x1bb

Value 30
  Name:           ServerComment
  Type:           REG_SZ
```

```
  Data:

Value 31
  Name:            ServerSideIncludesEnabled
  Type:            REG_DWORD
  Data:            0x1

Value 32
  Name:            ServerSideIncludesExtension
  Type:            REG_SZ
  Data:            .stm


Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Scri
pt Map
Class Name:        <NO CLASS>
Last Write Time:   5/12/99 - 3:48 PM
Value 0
  Name:            .idc
  Type:            REG_SZ
  Data:
C:\WINNT\System32\inetsrv\httpodbc.dll


Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Virt
ual Roots
Class Name:        <NO CLASS>
Last Write Time:   5/12/99 - 3:55 PM
Value 0
  Name:            /,
  Type:            REG_SZ
  Data:            C:\InetPub\wwwroot,,5

Value 1
  Name:            /iisadmin,
  Type:            REG_SZ
  Data:
C:\WINNT\System32\inetsrv\iisadmin,,1

Value 2
  Name:            /Scripts,
  Type:            REG_SZ
  Data:            C:\InetPub\scripts,,4


Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\Performance
Class Name:        <NO CLASS>
Last Write Time:   5/12/99 - 3:48 PM
Value 0
  Name:            Close
  Type:            REG_SZ
  Data:            CloseW3PerformanceData

Value 1
  Name:            Collect
  Type:            REG_SZ
  Data:            CollectW3PerformanceData

Value 2
  Name:            First Counter
  Type:            REG_DWORD
  Data:            0x810

Value 3
  Name:            First Help
  Type:            REG_DWORD
  Data:            0x811

Value 4
  Name:            Last Counter
  Type:            REG_DWORD
  Data:            0x848

Value 5
  Name:            Last Help
  Type:            REG_DWORD
  Data:            0x849

Value 6
  Name:            Library
  Type:            REG_SZ
  Data:            w3ctrs.DLL

Value 7
  Name:            Open
  Type:            REG_SZ
  Data:            OpenW3PerformanceData


Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\Security
Class Name:        <NO CLASS>
Last Write Time:   5/12/99 - 3:48 PM
Value 0
  Name:            Security
  Type:            REG_BINARY
  Data:
00000000   01 00 14 80 c0 00 00 00 - cc 00 00 00 14 00
00 00  ................
00000010   34 00 00 00 02 00 20 00 - 01 00 00 00 02 80
18 00  4..... ........
00000020   ff 01 0f 00 01 01 00 00 - 00 00 00 01 00 00
00 00  ................
00000030   20 02 00 00 02 00 8c 00 - 05 00 00 00 00 00
18 00   ...............
00000040   8d 01 02 00 01 01 00 00 - 00 00 00 01 00 00
00 00  ................
00000050   6d 00 63 00 00 00 1c 00 - fd 01 02 00 01 02
00 00  m.c.............
00000060   00 00 00 05 20 00 00 00 - 23 02 00 00 69 00
61 00  .... ...#...i.a.
00000070   00 00 1c 00 ff 01 0f 00 - 01 02 00 00 00 00
00 05  ................
00000080   20 00 00 00 20 02 00 00 - 69 00 61 00 00 00
1c 00   ... ...i.a.....
00000090   ff 01 0f 00 01 02 00 00 - 00 00 00 05 20 00
00 00  ............ ...
000000a0   25 02 00 00 69 00 61 00 - 00 00 18 00 fd 01
02 00  %...i.a.........
000000b0   01 01 00 00 00 00 00 05 - 12 00 00 00 25 02
00 00  ............%...
000000c0   01 01 00 00 00 00 00 05 - 12 00 00 00 01 01
00 00  ................
000000d0   00 00 00 05 12 00 00 00 -
........


Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\W3SAMP
Class Name:        <NO CLASS>
Last Write Time:   5/12/99 - 3:48 PM
```

Tuxedo Servers Configuration File

```
*RESOURCES
IPCKEY 133133

MAXACCESSERS 600
MAXSERVERS 90
MAXSERVICES 700
MODEL SHM
MASTER CLIENT1
LDBAL Y
SCANUNIT 15
BLOCKTIME 60
BBLQUERY 60

*MACHINES
DEFAULT:

"CLIENT1" LMID= CLIENT1
          TUXDIR="C:\Tuxedo"
          APPDIR="C:\InetPub\wwwroot"
          TUXCONFIG="C:\InetPub\wwwroot\tuxconfig"
          ULOGPFX="C:\InetPub\wwwroot\ULOG"
          TYPE="WinNT"
          UID= 0
          GID= 0

*GROUPS
T1
          LMID=CLIENT1 GRPNO=1 OPENINFO=NONE

T2
          LMID=CLIENT1 GRPNO=2 OPENINFO=NONE

T3
          LMID=CLIENT1 GRPNO=3 OPENINFO=NONE

T4
          LMID=CLIENT1 GRPNO=4 OPENINFO=NONE
```

Appendix C – Tunable Parameters

```
T5
          LMID=CLIENT1 GRPNO=5 OPENINFO=NONE

T6
          LMID=CLIENT1 GRPNO=6 OPENINFO=NONE

T7
          LMID=CLIENT1 GRPNO=7 OPENINFO=NONE

T8
          LMID=CLIENT1 GRPNO=8 OPENINFO=NONE

T9
          LMID=CLIENT1 GRPNO=9 OPENINFO=NONE

T10
          LMID=CLIENT1 GRPNO=10 OPENINFO=NONE

GROUPDEL
          LMID=CLIENT1 GRPNO=11 OPENINFO=NONE



*SERVERS
DEFAULT:

MULTI_TRANS SRVGRP=T1
          SRVID=100
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q1q REPLYQ=Y

MULTI_TRANS SRVGRP=T2
          SRVID=200
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q2q REPLYQ=Y

MULTI_TRANS SRVGRP=T3
          SRVID=300
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q3q REPLYQ=Y

MULTI_TRANS SRVGRP=T4
          SRVID=400
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q4q REPLYQ=Y

MULTI_TRANS SRVGRP=T5
          SRVID=500
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q5q REPLYQ=Y

MULTI_TRANS SRVGRP=T6
          SRVID=600
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q6q REPLYQ=Y

MULTI_TRANS SRVGRP=T7
          SRVID=700
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5  -DTPCC"
          RQADDR=q7q REPLYQ=Y

MULTI_TRANS SRVGRP=T8
          SRVID=800
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q8q REPLYQ=Y

MULTI_TRANS SRVGRP=T9
          SRVID=900
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q9q REPLYQ=Y

MULTI_TRANS SRVGRP=T10
          SRVID=1000
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q10q REPLYQ=Y

delivery SRVGRP=GROUPDEL
          SRVID=1100
          MIN=5 MAX=7
          CLOPT="-A -- -STPCC-X5 -F -DTPCC"
          RQADDR=delq REPLYQ=N

*SERVICES



*RESOURCES
IPCKEY 133133

MAXACCESSERS 600
MAXSERVERS 90
MAXSERVICES 700
MODEL SHM
MASTER CLIENT2
LDBAL Y
SCANUNIT 15
BLOCKTIME 60
BBLQUERY 60

*MACHINES
DEFAULT:

"CLIENT2" LMID= CLIENT2
          TUXDIR="C:\Tuxedo"
          APPDIR="C:\InetPub\wwwroot"
          TUXCONFIG="C:\InetPub\wwwroot\tuxconfig"
          ULOGPFX="C:\InetPub\wwwroot\ULOG"
          TYPE="WinNT"
          UID= 0
          GID= 0

*GROUPS
T1
          LMID=CLIENT2 GRPNO=1 OPENINFO=NONE

T2
          LMID=CLIENT2 GRPNO=2 OPENINFO=NONE

T3
          LMID=CLIENT2 GRPNO=3 OPENINFO=NONE

T4
          LMID=CLIENT2 GRPNO=4 OPENINFO=NONE

T5
          LMID=CLIENT2 GRPNO=5 OPENINFO=NONE

T6
          LMID=CLIENT2 GRPNO=6 OPENINFO=NONE

T7
          LMID=CLIENT2 GRPNO=7 OPENINFO=NONE

T8
          LMID=CLIENT2 GRPNO=8 OPENINFO=NONE

T9
          LMID=CLIENT2 GRPNO=9 OPENINFO=NONE

T10
          LMID=CLIENT2 GRPNO=10 OPENINFO=NONE

GROUPDEL
          LMID=CLIENT2 GRPNO=11 OPENINFO=NONE



*SERVERS
DEFAULT:

MULTI_TRANS SRVGRP=T1
          SRVID=100
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q1q REPLYQ=Y

MULTI_TRANS SRVGRP=T2
          SRVID=200
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q2q REPLYQ=Y

MULTI_TRANS SRVGRP=T3
          SRVID=300
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q3q REPLYQ=Y

MULTI_TRANS SRVGRP=T4
          SRVID=400
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q4q REPLYQ=Y
```

Appendix C – Tunable Parameters

```
MULTI_TRANS SRVGRP=T5
          SRVID=500
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q5q REPLYQ=Y

MULTI_TRANS SRVGRP=T6
          SRVID=600
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q6q REPLYQ=Y

MULTI_TRANS SRVGRP=T7
          SRVID=700
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5  -DTPCC"
          RQADDR=q7q REPLYQ=Y

MULTI_TRANS SRVGRP=T8
          SRVID=800
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q8q REPLYQ=Y

MULTI_TRANS SRVGRP=T9
          SRVID=900
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q9q REPLYQ=Y

MULTI_TRANS SRVGRP=T10
          SRVID=1000
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q10q REPLYQ=Y

delivery SRVGRP=GROUPDEL
          SRVID=1100
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -F -DTPCC"
          RQADDR=delq REPLYQ=N


*SERVICES




*RESOURCES
IPCKEY 133133

MAXACCESSERS 600
MAXSERVERS 90
MAXSERVICES 700
MODEL SHM
MASTER CLIENT2
LDBAL Y
SCANUNIT 15
BLOCKTIME 60
BBLQUERY 60

*MACHINES
DEFAULT:

"CLIENT2" LMID= CLIENT2
          TUXDIR="C:\Tuxedo"
          APPDIR="C:\InetPub\wwwroot"
          TUXCONFIG="C:\InetPub\wwwroot\tuxconfig"
          ULOGPFX="C:\InetPub\wwwroot\ULOG"
          TYPE="WinNT"
          UID= 0
          GID= 0

*GROUPS
T1
          LMID=CLIENT2 GRPNO=1 OPENINFO=NONE

T2
          LMID=CLIENT2 GRPNO=2 OPENINFO=NONE

T3
          LMID=CLIENT2 GRPNO=3 OPENINFO=NONE

T4
          LMID=CLIENT2 GRPNO=4 OPENINFO=NONE

T5
          LMID=CLIENT2 GRPNO=5 OPENINFO=NONE

T6
          LMID=CLIENT2 GRPNO=6 OPENINFO=NONE

T7
          LMID=CLIENT2 GRPNO=7 OPENINFO=NONE

T8
          LMID=CLIENT2 GRPNO=8 OPENINFO=NONE

T9
          LMID=CLIENT2 GRPNO=9 OPENINFO=NONE

T10
          LMID=CLIENT2 GRPNO=10 OPENINFO=NONE

GROUPDEL
          LMID=CLIENT2 GRPNO=11 OPENINFO=NONE



*SERVERS
DEFAULT:

MULTI_TRANS SRVGRP=T1
          SRVID=100
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q1q REPLYQ=Y

MULTI_TRANS SRVGRP=T2
          SRVID=200
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q2q REPLYQ=Y

MULTI_TRANS SRVGRP=T3
          SRVID=300
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q3q REPLYQ=Y

MULTI_TRANS SRVGRP=T4
          SRVID=400
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q4q REPLYQ=Y

MULTI_TRANS SRVGRP=T5
          SRVID=500
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q5q REPLYQ=Y

MULTI_TRANS SRVGRP=T6
          SRVID=600
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q6q REPLYQ=Y

MULTI_TRANS SRVGRP=T7
          SRVID=700
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5  -DTPCC"
          RQADDR=q7q REPLYQ=Y

MULTI_TRANS SRVGRP=T8
          SRVID=800
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q8q REPLYQ=Y

MULTI_TRANS SRVGRP=T9
          SRVID=900
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q9q REPLYQ=Y

MULTI_TRANS SRVGRP=T10
          SRVID=1000
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -DTPCC"
          RQADDR=q10q REPLYQ=Y

delivery SRVGRP=GROUPDEL
          SRVID=1100
          MIN=6 MAX=7
          CLOPT="-A -- -STPCC-X5 -F -DTPCC"
          RQADDR=delq REPLYQ=N


*SERVICES
```

Appendix C – Tunable Parameters

RTE Parameters

```
Profile:  1880-3c-2
File Path:          C:\benchcrf\1880-3c-2.pro
Version:  1.0.1

          Number of Engines: 20

                    Name: DRIVER1A
                    Description: client1a
                    Directory: c:\logs\driver1a.log
                    Machine: driver1
                    Parameter Set: 1900-3c-1
                    Index: 0
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER1341960218
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 0

                    Name: DRIVER1B
                    Description: client1b
                    Directory: c:\logs\driver1b.log
                    Machine: driver1
                    Parameter Set: 1900-3c-1
                    Index: 100000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER2342008984
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 1

                    Name: DRIVER1C
                    Description: client1c
                    Directory: c:\logs\driver1c.log
                    Machine: driver1
                    Parameter Set: 1900-3c-1
                    Index: 200000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER3342082593
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 0

                    Name: DRIVER1D
                    Description: client1d
                    Directory: c:\logs\driver1d.log
                    Machine: driver1
                    Parameter Set: 1900-3c-1
                    Index: 300000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER4342123500
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 1

                    Name: DRIVER2A
                    Description: client1e
                    Directory: c:\logs\driver2a.log
                    Machine: driver2
                    Parameter Set: 1900-3c-1
                    Index: 400000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER5342162531
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 0

                    Name: DRIVER2B
                    Description: client1f
                    Directory: c:\logs\driver2b.log
                    Machine: driver2
                    Parameter Set: 1900-3c-1
                    Index: 500000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER6342208734
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 1

                    Name: DRIVER2C
                    Description: client2a
                    Directory: c:\logs\driver2c.log
                    Machine: driver2
                    Parameter Set: 1900-3c-1
                    Index: 600000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER7342246265
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 0

                    Name: DRIVER2D
                    Description: client2b
                    Directory: c:\logs\driver2d.log
                    Machine: driver2
                    Parameter Set: 1900-3c-1
                    Index: 700000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER8342288484
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 1

                    Name: DRIVER3A
                    Description: client2c
                    Directory: c:\logs\driver3a.log
                    Machine: driver3
                    Parameter Set: 1900-3c-1
                    Index: 800000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER9342326312
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 0

                    Name: DRIVER3B
                    Description: client2d
                    Directory: c:\logs\driver3b.log
                    Machine: driver3
                    Parameter Set: 1900-3c-1
                    Index: 900000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER10342374500
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 1

                    Name: DRIVER3C
                    Description: client2e
                    Directory: c:\logs\driver3c.log
                    Machine: driver3
                    Parameter Set: 1900-3c-1
                    Index: 1000000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER11342419609
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 0

                    Name: DRIVER3D
                    Description: client2f
                    Directory: c:\logs\driver3d.log
                    Machine: driver3
                    Parameter Set: 1900-3c-1
                    Index: 1100000000
                    Seed: 86922
                    Configured Users: 940
                    Pipe Name: DRIVER12342474765
                    Connect Rate: 200
                    Start Rate: 100
                    CLIENT_NURAND: 208
                    CPU: 1

                    Name: DRIVER4A
                    Description: client2g
                    Directory: c:\logs\driver4a.log
                    Machine: driver4
                    Parameter Set: 1900-3c-1
                    Index: 1200000000
```


Appendix C – Tunable Parameters

```
Seed: 86922
Configured Users: 940
Pipe Name: DRIVER13342521109
Connect Rate: 200
Start Rate: 100
CLIENT_NURAND: 208
CPU: 0

Name: DRIVER4B
Description: client3a
Directory: c:\logs\driver4b.log
Machine: driver4
Parameter Set: 1900-3c-1
Index: 1300000000
Seed: 86922
Configured Users: 940
Pipe Name: DRIVER14342568937
Connect Rate: 200
Start Rate: 100
CLIENT_NURAND: 208
CPU: 1

Name: DRIVER4C
Description: client3b
Directory: c:\logs\driver4c.log
Machine: driver4
Parameter Set: 1900-3c-1
Index: 1400000000
Seed: 86922
Configured Users: 940
Pipe Name: DRIVER15342607140
Connect Rate: 200
Start Rate: 100
CLIENT_NURAND: 208
CPU: 0

Name: DRIVER4D
Description: client3c
Directory: c:\logs\driver4d.log
Machine: driver4
Parameter Set: 1900-3c-1
Index: 1500000000
Seed: 86922
Configured Users: 940
Pipe Name: DRIVER16342649656
Connect Rate: 200
Start Rate: 100
CLIENT_NURAND: 208
CPU: 1

Name: DRIVER5A
Description: client3d
Directory: c:\logs\driver5a.log
Machine: driver5
Parameter Set: 1900-3c-1
Index: 1600000000
Seed: 86922
Configured Users: 940
Pipe Name: DRIVER17342690421
Connect Rate: 200
Start Rate: 100
CLIENT_NURAND: 208
CPU: 0

Name: DRIVER5B
Description: client3e
Directory: c:\logs\driver5b.log
Machine: driver5
Parameter Set: 1900-3c-1
Index: 1700000000
Seed: 86922
Configured Users: 940
Pipe Name: DRIVER18342736140
Connect Rate: 200
Start Rate: 100
CLIENT_NURAND: 208
CPU: 1

Name: DRIVER5C
Description: client3f
Directory: c:\logs\driver5c.log
Machine: driver5
Parameter Set: 1900-3c-1
Index: 1800000000
Seed: 86922
Configured Users: 940
Pipe Name: DRIVER19342798781
Connect Rate: 200
Start Rate: 100
CLIENT_NURAND: 208
CPU: 0

Name: DRIVER5D
Description: client3g
Directory: c:\logs\driver5d.log
Machine: driver5
Parameter Set: 1900-3c-1
Index: 1900000000
Seed: 86922
Configured Users: 940
Pipe Name: DRIVER20342839968
Connect Rate: 200
Start Rate: 100
CLIENT_NURAND: 208
CPU: 1

Number of User groups: 20

Driver Engine: DRIVER1A
IIS Server: client1a
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1 - 94
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER3B
IIS Server: client2d
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 847 - 940
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER3A
IIS Server: client2c
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 941 - 1034
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER3D
IIS Server: client2f
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1035 - 1128
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER3C
IIS Server: client2e
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1129 - 1222
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER4A
IIS Server: client2g
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1223 - 1316
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER4C
IIS Server: client3b
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1317 - 1410
```

Appendix C – Tunable Parameters

```
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER4B
IIS Server: client3a
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1411 - 1504
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER4D
IIS Server: client3c
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1505 - 1598
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER5B
IIS Server: client3e
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1599 - 1692
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER5A
IIS Server: client3d
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1693 - 1786
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER1B
IIS Server: client1b
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 95 - 188
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER5C
IIS Server: client3f
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 1787 - 1880
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER5D
IIS Server: client3g
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 189 - 282
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER1D
IIS Server: client1d
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 283 - 376
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER1C
IIS Server: client1c
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 377 - 470
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER2A
IIS Server: client1e
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 471 - 564
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER2C
IIS Server: client2a
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 565 - 658
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER2B
IIS Server: client1f
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 659 - 752
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Driver Engine: DRIVER2D
IIS Server: client2b
SQL Server: tpcc-x5
User: sa
Protocol: Html
w_id Range: 753 - 846
w_id Max Warehouse: 1880
Scale: Normal
User Count: 940
District id: 1
Scale Down: No

Number of Parameter Sets: 2



~Default
Default Parameter Set
                Txn     Think   Key     RT      RT      Menu
                Weight  Time    Time    Delay   Fence   Delay
New Order         10.00         12.05    18.01    0.10    5.00    0.10
Payment           10.00         12.05     3.01    0.10    5.00    0.10
Delivery           1.00          5.05     2.01    0.10    5.00    0.10
Stock Level        1.00          5.05     2.01    0.10   20.00    0.10
```

Order Status	1.00		10.05	2.01	0.10	5.00	0.10

1900-3c-1
modified 5-4-5

	Txn Weight	Think Time	Key Time	RT Delay	RT Fence	Menu Delay	
New Order	44.88		12.05	18.01	0.10	5.00	0.10
Payment	43.03		12.05	3.01	0.10	5.00	0.10
Delivery	4.04		5.05	2.01	0.10	5.00	0.10
Stock Level	4.02		5.05	2.01	0.10	20.00	0.10
Order Status	4.03		10.05	2.01	0.10	5.00	0.10

Appendix D – Disk Storage

TPC-C 180 Day Space Requirements						
Warehouses	**1900**				**TpmC**	**23,235.57**
Table	**Rows**	**Data KB**	**Index KB**	**Extra 5% KB**	**8hr Space**	**Total Space KB**
Warehouse	1900	208	40	12		260
District	19000	2112	40	108		2260
Customer	57000000	41454552	2661672	2,205,811		46322035
History	57000000	3166680	16		356,510	3166696
NewOrder	17100000	270360	680			271040
Orders	57000000	1747128	964912		2,149,529	2712040
OrderLine	570001928	35625128	88664		3,530,739	35713792
Item	100000	9528	64	480		10072
Stock	190000000	60800000	136064	3,046,803		63982867
Total		143,075,696	3,852,152	5,253,214	6,036,779	152,181,062

	MB	
Dynamic Space	39,589	Sum of Data for Order, Orderline and History
Static Space	109,026	Sum of Data+Index+5%-Dynamic Space
Free Space	na	Total Allocated Spac - (Dynamic + Static Space)
Daily Growth	7,727	(Dynamic Space/(W*62.5))*tpmc
Daily Spread	-	(Free Space -1.5*Daily Growth) Zero Assumed

180 Day Space MB	1,499,908		18 GB Drive	17.300	GB	
180 Day Space GB	**1,464.75**	**GB**	9 GB Drive	8.474	GB	
Log Size	48,000	MB				
KB Per New Order	5.3498	KB				
8 hr log MB	58,124	MB				
8 hr log GB	**56.7621**	**GB**				

Space Usage	**GB Needed**	**Disks Measured**	**GB Priced**			
180 Day Space DB	1,464.75	0	0.00	18GB		
		192	1626.94	9GB		
Total DB		**192**	**1626.94**	**GB**		
8-hr log + mirror	113.5242	8	138.40	GB		
OS, Swap	6	1	8.475	GB		
Total Storage	**1,584.28 GB**		**1,773.81 GB**			

Log Space OK
Total Space OK

Appendix E – Price Quotations

Microsoft

May 10, 1999

Duncan Liu
Acer Inc.
21F, 88, Sec. 1, Hsin Tai Wu Road,
Hsichih, Taipei Hsien 221,
Taiwan
Phone:886-2-8691-1530
Fax: 886-2-8691-2379
E-mail: duncanliu@acer.com.

Dear Mr. Liu:

Here is the information you requested regarding U.S. pricing of several Microsoft products that were used in a recent TPC-C benchmark:

Microsoft SQL Server 7.0, Enterprise Edition (one server plus unlimited CALs)	$28,999
Microsoft Windows NT Server 4.0, Enterprise Edition (one server plus 25 CALs)	$3,999
Windows NT Server 4.0 (one server plus 5 CALs)	$809
Visual C++ Professional 6.0 (single copy)	$1299
5-year maintenance for above software @ $2095/yr	$10,475

This quote is valid for the next 90 days.

If I can be of any further assistance, please contact me at 425-936-5301 or tomkr@microsoft.com.

Yours truly,

Thomas Kreyche
Product Manager
SQL Server Marketing

MYLEX
MOVING DATA SMARTER

fax transmission

To:	From:
Mr. Duncan Liu	Soogil Stephen Cho, VP of Asia Pacific Sales
Cc:	**Date:**
Mr. Frank Hsu – Mylex Taiwan	05/12/99
Company:	**Total pages including cover:**
Acer Inc. Taipei, Taiwan	one
Fax Number	**Phone & Fax Number**
886-2-8691-2379	(510) 608-2266 (Direct) / (510) 745-7521 (Fax)

RE: Your Inquiry - Quotation

Duncan,

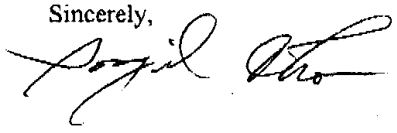
Please see the price quote for the DAC1164P series product that you have requested.

Part Number	Description	Unit Price (US$)
DAC1164P-3E-32-MY	3 External and 2 internal SCSI LVD with 32MB memory with Integrated battery back up.	$ 1,895

NOTES: 1. Above price is firm for 90 days and based on FOB, ex-factory, Fremont, CA.
2. 3 years normal warranty. US$ 50 per year for the estimated maintenance cost for additional 2 more years.
3. Normal delivery lead time is 30 days ARO.

Should you have any questions or require any additional information, please advise me immediately.

Sincerely,



Appendix E – Price Quotations

May 6, 1999

Duncan Liu
Acer Inc.
21F, 88, Sec. 1, Hsin Tai Wu Road,
Hsichih, Taipei Hsien 221,
Taiwan
Phone: 886-2-8691-1530
Fax: 886-2-8691-2379
E-mail: duncanliu@acer.com.tw

Dear Mr. Liu;

Per your request I am enclosing the pricing information regarding TUXEDO 6.x that you requested. This pricing applies to Tuxedo 6.1, 6.2, 6.3, 6.4 and 6.5. Please note that Tuxedo 6.5 is our most recent version of Tuxedo but that all 6.x releases are generally available. Core functionality services pricing is appropriate for your activities. As per the table below, server systems are classified in one of 5 tiers based on CPU type and capacity. The AcerAltos 21000 with 4 Pentium III Xeon (400MHz/2MB) and 4GB of memory is a tier 2 system. This quote is valid for 90 days from the date of issue of this letter.

Tuxedo Core Functionality Services (CFS) Program Product Pricing and Description

TUX-CFS provides a basic level of middleware support for distributed computing, and is best used by organizations with substantial resources and knowledge for advanced distributed computing implementations.

TUX-CFS prices are server only and are based on the overall performance characteristics of the server and uses the same five tier computer classification as TUXEDO 6.x. Prices range from $3,000 for Tier 1 to $250,000 for Tier 5. Under this pricing option EVERY system running TUX-CFS at the user site must have a TUXEDO license installed and pay the appropriate per server license fees.

BEA Tux/CFS Unlimited User License Fees Per Server

Unlimited User License fees per server	Number of Users	Dollar Amount	Maintenance (5 x 8) per year	Maintenance (7 x 24) per year
Tier 1 -- PC Servers with 1 or 2 CPUs, entry level RISC Uni-processor workstations and servers (Class 1and Class 2)	Unlimited	$3,000.00	$480.00	$690.00
Tier 2 -- PC Servers with 3 or 4 CPUs, Midrange RISC Uni-processor servers and workstations (class 3)	Unlimited	$12,000.00	$1,920.00	$2,760.00

Tier 3 -- Midrange Multiprocessors, up to 8 CPUs per system capacity (Class 4 and 5)	Unlimited	$30,000.00	$4,800.00	$6,900.00
Tier 4 -- Large (more than 8, less than 32 CPUs) and Mainframe Systems (Class 6)	Unlimited	$100,000.00	$16,000.00	$23,000.00
Tier 5 -- Massively Parallel Systems, > 32 processors	Unlimited	$250,000.00	$40,000.00	$57,500.00

Intel based server tier classifications:

Platform	Operating System	Tier 1	Tier 1	Tier 2	Tier 3	Tier 3
Intel Pentium/ Pentium Pro PCs	Interactive R3.2 ESIX SVR 4.0 SCO UNIX 3.2.2 and 3.2.4 SCO ODT 2.x,3.x Solaris x86 2.X UnixWare, Windows NT 3.5/4.0	All 386/486 PCs are Class 1	ALL Pentium and Pentium Pro PCs with 1 or 2 CPUs capacity are Tier 1	ALL Pentium and Pentium Pro PCs with 3 or 4 CPUs capacity are Tier 2		ALL Pentium and Pentium Pro PCs with 5,6,7, or 8 CPUs are Tier 3

Very Truly Yours,

Lewis D. Brentano,
Director, Market Planning