Compaq Computer Corporation

TPC Benchmark™ C
Full Disclosure Report
for
ProLiant 2500 6/200 (2 Proc)
using
Microsoft SQL Server v.6.5
and
Microsoft Windows NT 4.0

First Edition
May 1997

COMPAQ

First Edition – May 1997

Compaq Computer Corporation (Compaq) believes that the information in this document is accurate as of the publication date. The information in this document is subject to change without notice. Compaq assumes no responsibility for any errors that may appear in this document. The pricing information in this document is believed to accurately reflect the current prices as of the publication date. However, Compaq 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 of 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. Compaq 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.

Printed in U.S.A., 1997
Compaq, ProLiant 2500, ProLiant, and ProLiant 800 are registered trademarks of Compaq Computer Corporation.

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

Pentium®Pro is a registered trademark of Intel.

Empower is a trademark of Performix, Inc.

TPC Benchmark is a trademark of the Transaction Processing Performance Council.

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

Table of Contents

Preface

The TPC Benchmark C was developed by the Transaction Processing Performance Council (TPC). The TPC was founded to define transaction processing benchmarks and to disseminate objective, verifiable performance data to the industry. This full disclosure report is based on the TPC Benchmark C Standard Specifications Version 3.2, released August 27, 1996.

TPC Benchmark C Overview

The TPC describes this benchmark in Clause 0.1 of the specifications as follows:

TPC Benchmark C is an On Line Transaction Processing (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 of 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.

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 other environments 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 benchmark when critical capacity planning and/or product evaluation decisions are contemplated.

Abstract

Overview

This report documents the methodology and results of the TPC Benchmark C test conducted on the Compaq ProLiant 2500. The operating system used for the benchmark was Microsoft Windows NT 4.0. The DBMS used was Microsoft SQL Server 6.5 (Service Pack 3) for Window NT.

TPC Benchmark C Metrics

The standard TPC Benchmark C metrics, tpmC (transactions per minute), price per tpmC (five year capital cost per measured tpmC), and the availability date are reported as:

4864.97 tpmC
$62.49 per tpmC

All software components are currently available. The Pentium Pro 200 512KB cache processor boards will be available in May 5 of 1997.

Standard and Executive Summary Statements

The following pages contain executive summary of results for this benchmark.

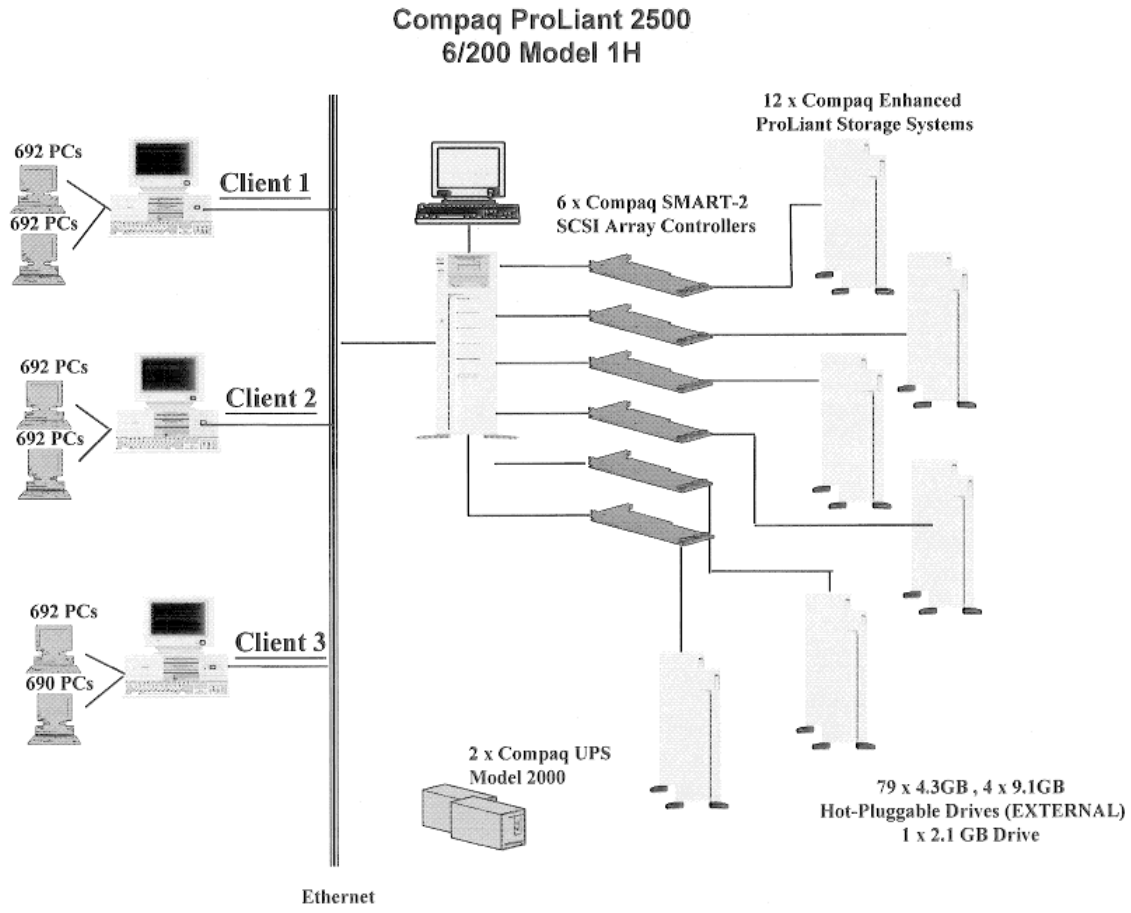
Auditor

The benchmark configuration, environment and methodology were audited by Lorna Livingtree of Performance Metrics, Inc. to verify compliance with the relevant TPC specifications.

Compaq Computer Corporation	ProLiant 2500 6/200 (2 Proc) C/S with 3 ProLiant 800	TPC-C Rev. 3.3 Report Date: May 1, 1997

Total System Cost	TPC-C Throughput	Price/Performance	Availability Date
$304,046	**4864.97**	**$62.49**	**05-May-1997**

Processors	Database Manager	Operating System	Other Software	Number of Users
2 Pentium®Pro 200 MHz 512K Cache	Microsoft SQL Server 6.5.252 (SP3)	Microsoft Windows NT 4.0 (SP1)	Microsoft Internet Connector License, Microsoft Visual C++	**4150**



System Components	Server		Clients	
	Quantity	Description	Quantity	Description
Processor	2	200MHz Pentium®Pro 512K Level 2 Cache	3	180MHz Pentium®Pro 256K Level 2 Cache
Memory	1	1024MB	3	96MB
Disk Controllers	6	SMART2/P Array Controller	3	Integrated SCSI Disk Controller
Disk Drives	79 4 1	4.3 GB SCSI-2 Drives 9.1 GB SCSI-2 Drives 2.1 GB SCSI-2 Drive	3	4.3 GB SCSI-2 Drive
Total Storage		378.2 GB		12.9 GB
Tape Drives	1	4/16 TurboDAT		

Compaq Computer Corporation	ProLiant 2500 - 6/200 Model 1H Client/Server					TPC-C REV 3.3 Report Date:	2-May-97
Description	**Part Number**	**Third Party Brand**	**Pricing**	**Unit Price**	**Qty**	**Extended Price**	**5 yr. Maint. Price**
Server Hardware							
ProLiant 2500 6/200 - 512K Model 1H PentiumPro/200MHz 512K Processor Board Netelligent 10/100 TX Ethernet Controller Integrated Fast-SCSI-2 Controller CD-ROM SmartStart Compaq System Configuration Utility	307550-001			6,655	1	6,655	2,329
ProLiant 2500 6/200-512K Option Kit	300906-001			2,905	1	2,905	1,017
256 MB DIMM Kit	271910-001			5,238	4	20,952	7,333
SMART-2/P SCSI Array Controller	194753-001			2,138	6	12,828	4,490
Enhanced ProLiant Storage System	189600-001			863	12	10,356	3,625
2.1 GB Pluggable SCSI-2 Drive	199876-001			744	1	744	260
4.3 GB Pluggable SCSI-2 Drive	146742-006			1,227	79	96,933	33,927
9.1 GB Pluggable SCSI-2 Drive	199882-001			2,127	4	8,508	2,978
Compaq V50 Color Monitor	264150-001			373	1	373	131
Compaq Uninteruptible Power Supply T2000	242688-005			903	2	1,806	632
4\16GB TurboDAT Drive	142181-001			1,084	1	1,084	379
					Subtotal	**163,144**	**57,100**
Server Software							
Microsoft Windows NT Server v. 4.0		Microsoft	1	809	1	809	0*
* All maint. is covered by the maint. Cost of Microsoft SQL Server							
Microsoft SQL Server 6.5 plus User License (w/ 5 cal)		Microsoft	1	1,399	1	1,399	10,475
Microsoft Internet Connector License		Microsoft	1	2,999	1	2,999	0*
					Subtotal	**5,207**	**10,475**
Client Hardware							
ProLiant 800 6/180 - Model 4300	273750-003			3,347	3	10,041	3,514
64-MB Memory Kit (1x64-MB, 60 ns, ECC)	225483-001			625	6	3,750	1,313
NetFlex 3 PCI Ethernet Controller	169810-001			162	6	972	340
Compaq V50 Color Monitor	264150-001			373	3	1,119	392
					Subtotal	**15,882**	**5,559**
Client Software							
Microsoft Windows NT Server v. 4.0		Microsoft	1	809	3	2,427	0*
Microsoft SQL Workstation		Microsoft	1	499	1	499	0*
Microsoft Visual C++ v. 4.0		Microsoft	1	499	1	499	0*
					Subtotal	**3,425**	**0**
User Connectivity							
NetLux 8-Port 100TX TrueFAST Ethernet Hub		NetLux	1	658	3	1,974	5 yr warranty
CentreCOM 24-Port Ethernet Hub	AT3024TR	CentreCOM	1	215	192	41,280	5 yr warranty
					Subtotal	**43,254**	**0**
					Total	**$230,912**	**$73,134**

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

Five-Year Cost of Ownership: $304,046

tpmC Rating: 4864.97

$ / tpmC: $62.49

Note: The benchmark results and test methodology were audited by Lorna Livingtree of Performance Metrics, Inc.

Numerical Quantities Summary

MQTH, Computed Maximum Qualified Throughput 4864.97 tpmC

% throughput difference, reported & reproducibility runs 0.01%

Response Times (in seconds)	Average	90%	Maximum
New-Order	1.53	2.19	5.19
Payment	1.39	2.05	5.23
Order-Status	1.68	2.35	5.08
Delivery (interactive portion)	0.13	0.15	0.84
Delivery (deferred portion)	3.46	7.16	32.50
Stock-Level	3.51	4.62	7.77
Menu	0.13	0.15	3.14

Transaction Mix, in percent of total transaction

New-Order	43.19%
Payment	44.61%
Order-Status	4.06%
Delivery	4.09%
Stock-Level	4.05%

Emulation Delay (in seconds)	Resp.Time	Menu
New-Order	0.1	0.1
Payment	0.1	0.1
Order-Status	0.1	0.1
Delivery (interactive)	0.1	0.1
Stock-Level	0.1	0.1

Keying/Think Times (in seconds)	Min.	Average	Max.
New-Order	18.00/0.00	18.01/12.29	18.26/124.70
Payment	3.00/0.00	3.01/12.23	3.09/122.31
Order-Status	2.00/0.00	2.01/10.15	2.05/93.00
Delivery (interactive)	2.00/0.00	2.01/5.14	2.05/50.90
Stock-Level	2.00/0.00	2.01/5.21	2.05/45.50

Test Duration

Ramp-up time	15 minutes
Measurement interval	30 minutes
Transactions (all types) completed during measurement interval	650,507
Ramp down time	10 minutes

Checkpointing

Number of checkpoints	1
Checkpoint interval	30 minutes

General Items

Application Code and Definition Statements

The application program (as defined in clause 2.1.7) must be disclosed. This includes, but is not limited to, the code implementing the five transactions and the terminal input output functions.

Appendix A contains all source code implemented in this benchmark.

Test Sponsor

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

This benchmark was sponsored by Compaq Computer Corporation. The benchmark was developed and engineered by Compaq Computer Corporation and Microsoft Corporation. Testing took place at Compaq benchmarking laboratories in Houston, Texas.

Parameter Settings

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

- *Database options*
- *Recover/commit options*
- *Consistency locking options*
- *Operating system and application configuration parameters*

This requirement can be satisfied by providing a full list of all parameters.

Appendix C contains the tunable parameters to for the database, the operating system, and the transaction monitor.

Configuration Items

Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences.
The configuration diagrams for both the tested and priced systems are included on the following pages.

Figure 1. Benchmarked Configuration

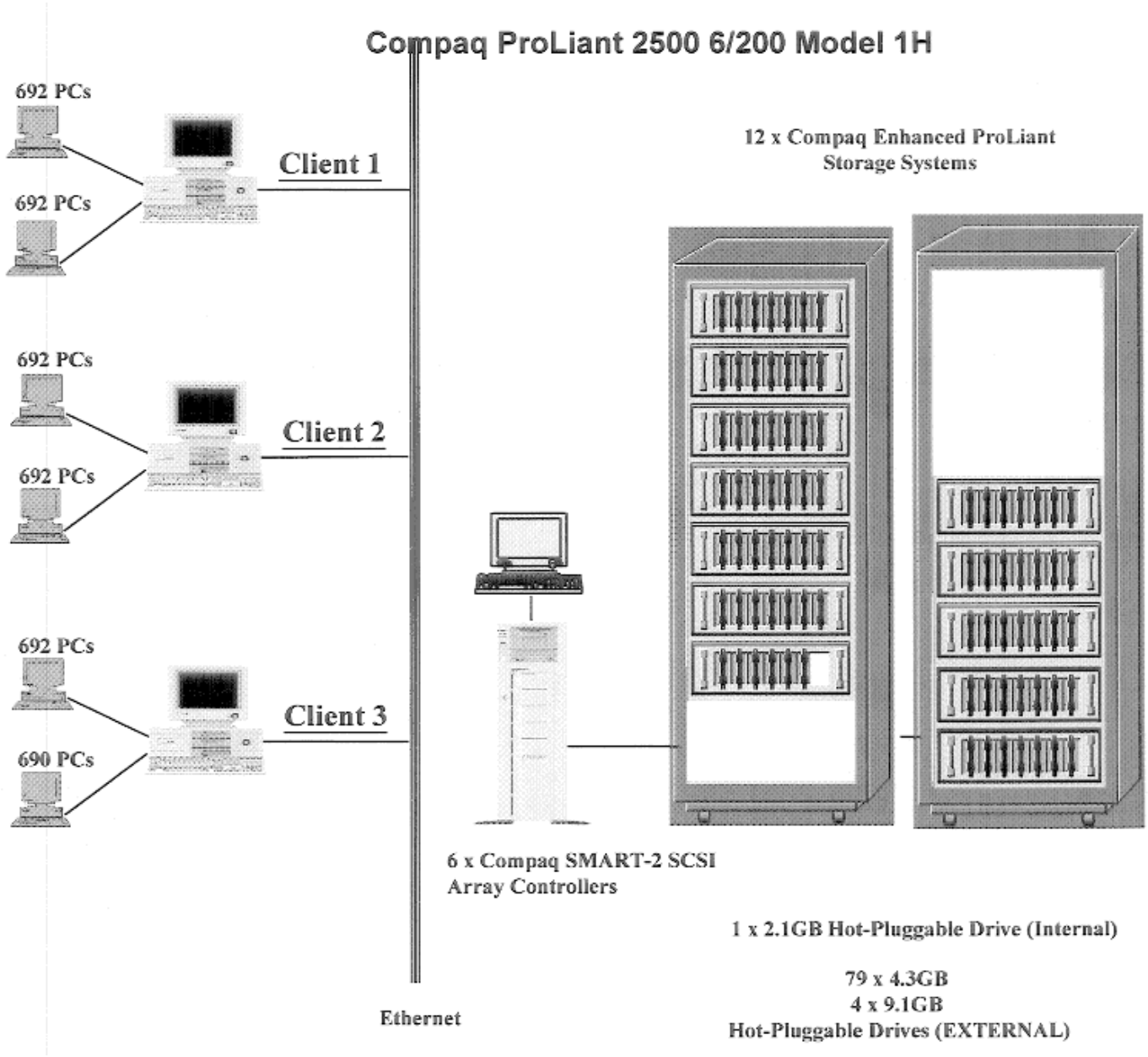
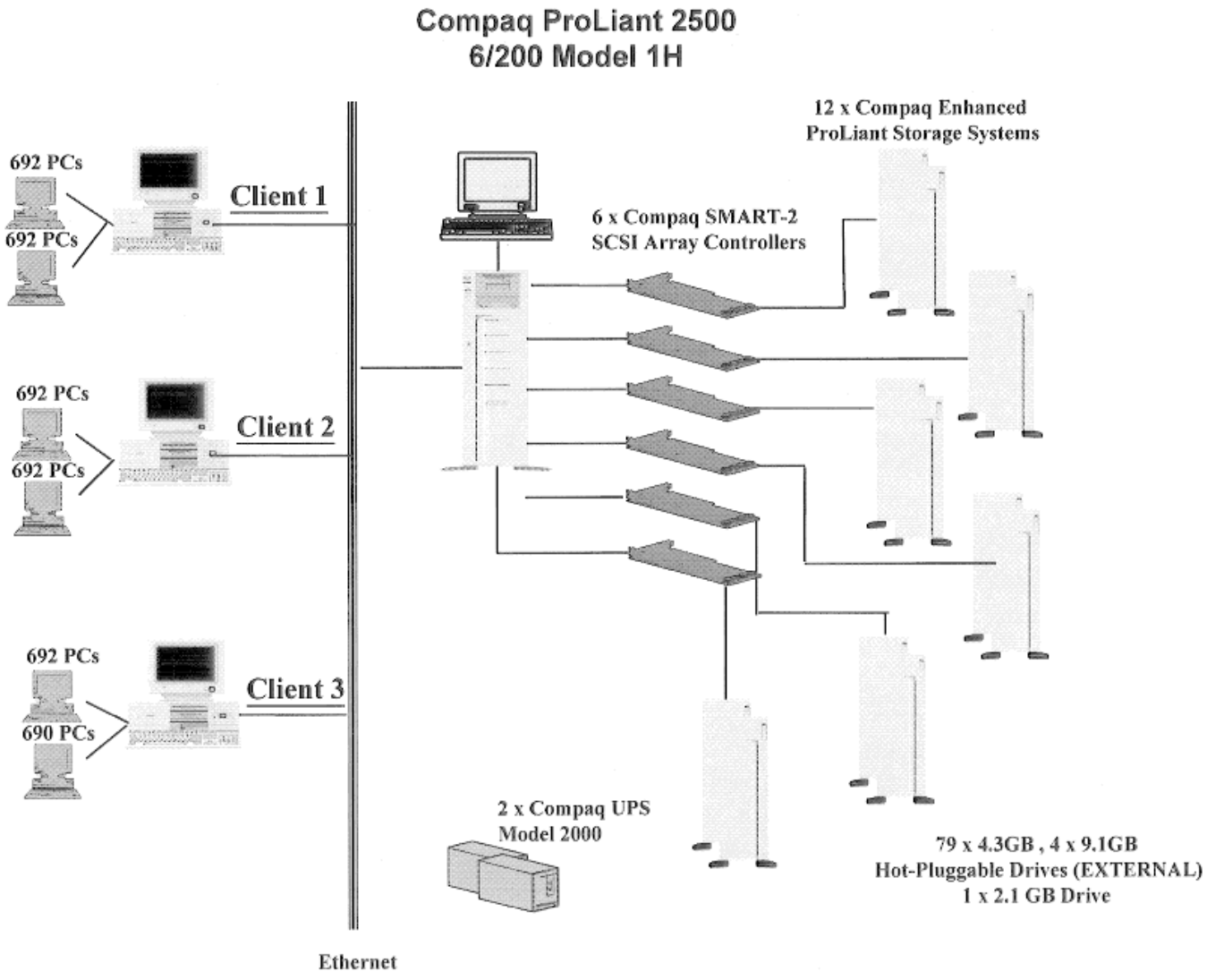


Figure 2. Priced Configuration



Clause 1 Related Items

Table Definitions

Listing must be provided for all table definition statements and all other statements used to set up the database.

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

Physical Organization of Database

The physical organization of tables and indices within the database must be disclosed.

The tested configuration consisted of 79 drives at 4.3GB each, 4 drives at 9.1GB each, and 1 drive at 2.1GB for the operating system files.

Benchmarked Configuration:

Following is an outline of the system logical drives, including the hardware (drives, controllers) used to create them, and the database devices which they hold.

Integrated SCSI-2 Fast/Wide Controller

EISA UTILITIES PARTITION	Total Capacity = 39 MB	
Compaq System Configuration Utilities		100%
LOGICAL DRIVE C:	Total Capacity = 1961 MB	
Microsoft Windows NT v. 4.0		100%
Swap Space (Windows NT paging file)		
523MB initial, 523 MB max.		100%

SMART-2/P Controller, Slot 1, Array A - Fourteen 4.3 GB drives

LOGICAL DRIVE E:	RAID 0	Total Capacity = 12588 MB
Bigdev1 device.		
LOGICAL DRIVE Z:	RAID 5	Total Capacity = 44113 MB
Database backup		

SMART-2/P Controller, Slot 2, Array A – Fourteen 4.3 GB drives

LOGICAL DRIVE F:	RAID 0	Total Capacity = 12588 MB
Bigdev2 device.		
LOGICAL DRIVE Z:	RAID 5	Total Capacity = 44113 MB
Database backup		

SMART-2/P Controller, Slot 3, Array A – Fourteen 4.3 GB drives

LOGICAL DRIVE G:	RAID 0	Total Capacity = 12588 MB
Bigdev3 device.		
LOGICAL DRIVE Z:	RAID 5	Total Capacity = 44113 MB
Database backup		

SMART-2/P Controller, Slot 4, Array A – Fourteen 4.3 GB drives

LOGICAL DRIVE H:	RAID 0	Total Capacity = 12588 MB
Bigdev4 device.		
LOGICAL DRIVE Z:	RAID 5	Total Capacity = 44113 MB
Database backup		

SMART-2/P Controller, Slot 5, Array A – Seven 4.3 GB drives

LOGICAL DRIVE I:	RAID 0	Total Capacity = 6295 MB
Bigdev5 device.		
UNUSED		Total Capacity = 22361 MB

SMART-2/P Controller, Slot 5, Array B – Seven 4.3 GB drives
LOGICAL DRIVE J: RAID 0 Total Capacity = 10485 MB
OL1 device.
UNUSED Total Capacity = 18171 MB

SMART-2/P Controller, Slot 6, Array A – Four 9.1 GB drives
LOGICAL DRIVE K: RAID 1 Total Capacity = 18194 MB
Transaction Log device.

SMART-2/P Controller, Slot 6, Array B – Nine 4.3 GB drives
LOGICAL DRIVE L: RAID 0 Total Capacity = 2097 MB
Misc1 device.
UNUSED Total Capacity = 34746 MB

Priced Configuration:

The priced configuration and the tested configuration differ only in that the benchmarked configuration was run using Rack Mountable ProLiant Storage Systems, whereas the priced configuration was run using all tower (non-rack) storage. Both systems were tower Proliant 2500 units, the tested unit being attached to its rack storage by means of external SCSI cables.

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 restrictions 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 minimum key value for these new rows.

All insert and delete functions were fully operational during the entire benchmark.

Partitioning

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

Partitioning was not used on any table in this benchmark.

Replication, Duplication or Additions

Replication of tables, if used, must be disclosed. Additional and/or duplicated attributes in any table must be disclosed along with a statement on the impact on performance.

No replications, duplications or additional attributes were used in this benchmark.

Clause 2 Related Items

Random Number Generation

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

The driver used the Empower RTE from Performix, Inc. to generate random numbers for the input data. The random number generator received a unique seed for each user.

The Empower RTE computes random integers using an implementation of a linear congruential sequence as described in "The Art of Computer Programming, Volume 2/ Seminumerical Algorithms" by Donald E. Knuth, copyright 1981 by Addison-Wesley Publishing Company.

The seeds for each user were captured and verified by the auditor to be unique. In addition, the contents of the database were systematically searched, and randomly sampled by the auditor for patterns that would indicate the random number generator had effected any kind of a discernible pattern; none were found.

Input/Output Screen Layout

The actual layout of the terminal input/output screens must be disclosed.

All screen layouts followed the specifications exactly.

Priced Terminal Feature 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).

The terminal attributes were verified by the auditor manually exercising each specification on a representative Compaq Proliant 800 with Microsoft Windows NT 4.0 and the Microsoft Internet Explorer v3.0.

Presentation Manager or Intelligent Terminal

Any usage of presentation managers or intelligent terminals must be explained.

Application code running on the client machines implemented the TPC-C user interface. No presentation manager software or intelligent terminal features were used. The source code for the forms applications is listed in Appendix A.

Transaction Statistics

Table 2.1 lists the numerical quantities that Clauses 8.1.3.5 to 8.1.3.11 require.

Table 2. 1 Transaction Statistics

Statistic		Value
New Order	Home warehouse order lines	89.60%
	Remote warehouse order lines	10.34%
	Rolled back transactions	0.92%
	Average items per order	10.03%
Payment	Home warehouse payments	84.83%
	Remote warehouse payments	15.17%
	Accessed by last name	60.43%

Statistic		Value
Order Status	Accessed by last name	60.21%
Transaction Mix	New Order	44.60%
	Payment	43.20%
	Order status	4.06%
	Delivery	4.09%
	Stock level	4.05%

Queueing Mechanism

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

The client application processes submitted delivery transactions to named pipe delivery server software running on the client machines. There was a single delivery server running on each client machine. These delivery servers were responsible for processing deliveries queued to the named pipe and submitting them to the database server.

The source code is listed in Appendix A.

Clause 3 Related Items

Transaction System Properties (ACID)

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.

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 seven 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.

In addition, the phantom tests and the stock level tests were executed and verified.

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.. The standard driving mechanism was used to generate the transaction load of 100 users for the Loss of Data. The fully scaled database under full load would also have passed the following test.

Loss of Data

To demonstrate recovery from a permanent failure of durable medium containing TPC-C tables, the following steps were executed. Loss of data was demonstrated on a 10 warehouse system with identical software and similar (although scaled down) hardware as the SUT.

1. The database was backed up to extra disks.
2. The total number of New Orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving the beginning count.
3. The RTE was started with 100 users.
4. The Loss of Log test was performed (see below).
5. The test was allowed to run for a further 10 minutes.
6. One of the data disks was removed from the drive cabinet.
7. When SQL Server recorded errors about not being able to access the database, the database server and the RTEs were shut down.
8. A new data disk was inserted into the drive cabinet, powered back up and system restarted.
9. SQL Server was started and a dump of the transaction log was taken.
10. The database was dropped and recreated as an empty database.
11. The database was restored from the database backup, followed by the restore of the transaction log.
12. Consistency condition #3 was executed and verified.
13. Step 2 was repeated and the difference between the first and second counts was noted.
14. An RTE report was generated for the entire run time giving the number of NEW-ORDERS successfully returned to the RTE.
15. The counts in step 12 and 13 were compared and the results verified that all committed transactions had been successfully recovered.
16. Samples were taken from the RTE files and used to query the database to demonstrate successful transactions had corresponding rows in the ORDER table.

Loss of Log

Loss of Log was also demonstrated on the 10 warehouse database, and was performed during the same run as the Loss of Data test. 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 SQL Server transaction log data, the following steps were executed:

1. The total number of New Orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving the beginning count.
2. The RTE was started with 100 users.
3. The test was allowed to run for a minimum of 10 minutes.
4. One log disk was removed from the drive cabinet.
5. Since the disk was mirrored, processing was not interrupted.
6. The test was allowed to run for a minimum of 10 minutes again, and then the Loss of Data test was performed.

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 415 warehouses under a full load of 4150 users. The following steps were executed:

1. The total number of New Orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving the beginning count.
2. The RTE was started with 4150 users.
3. The test was allowed to run for over the minimum of 10 minutes.
4. System crash and loss of memory were induced by pulling out the power cord from the back of the computer. No battery backup or Uninterruptible Power Supply (UPS) were used to preserve the contents of memory.
5. The RTE was shutdown.
6. Power was restored and the system restarted.
7. SQL Server was restarted and performed an automatic recovery.
8. Consistency condition #3 was executed and verified.
9. The total number of New Orders was determined and the difference between the first and second counts was noted.
10. An RTE report was generated for the entire run time giving the number of New Orders successfully returned to the RTE.
11. The two New Order counts were compared and the results verified that all committed transactions had been successfully recovered.
12. Samples were taken from the RTE files and used to query the database to demonstrate successful transactions had corresponding rows in the ORDER table.

Clause 4 Related Items

Initial Cardinality of Tables

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

Table 4.1 Number of Rows for Server

Table	Cardinality as benchmarked
Warehouse	415
District	4150
Customer	12,450,000
History	12,450,000
Orders	12,450,000
New Order	3,735,000
Order Line	124,503,957
Stock	41,500,000
Item	100,000
Deleted Warehouses	0

Constant Values

The following constant values were used during the database build and benchmark test for the NURand function.

Table 4.2 C Constants for NURand

Constant C	Value
C_LAST (Build)	123
C_LAST (run)	15
C_ID	877
OL_ID	1217

Database Layout

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

The benchmarked configuration used six (6) SMART-2/P Array controllers, each with two (2) SCSI channels. Each controller is capable of accessing up to fourteen (14) disk drives, seven (7) disk drives per each channel, and supports RAID 0, RAID 1 and RAID 5 per each logical volume configured. The data tables were stored on seven (7) logical volumes, which spanned a combined total of seventy nine (79) 4.3GB drives across twelve (12) Rack Mountable ProLiant Storage Cabinets; each volume was configured with RAID 0 and the Array Accelerator was enabled. The first internal 2.1GB drive stored the operating system. Four (4) 9.1GB drives were configured as a RAID 1 volume, and stored the transaction log. The transaction log volume had the Array Accelerator enabled. All RAID volumes used hardware RAID.

The benchmarked configuration also used four (4) of the SMART-2/P Array controllers to provide space for backups (dumps) of the database. These were configured with RAID 5, and combined into one space using Windows NT striping.

Section 1.2 of this report details the distribution of database devices across all disks. The code that creates the database sections across those devices, the segments which map to the devices, and the tables which reside on them is included in Appendix B.

Type of Database

A statement must be provided that describes:

1. *The data model implemented by 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 transaction. 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 v6.50.242 is a relational DBMS.

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

Database Mapping

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

The database was not replicated.

180 Day Space

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.

To calculate the space required to sustain the database log for 8 hours of growth at steady state, the following steps were followed:

1. The free space on the log file was queried using *DBCC checktable(syslogs)*.
2. Transactions were run against the database with a full load of users.
3. The free space was again queried using *DBCC checktable(syslogs)*.
4. The space used was calculated as the difference between the first and second query.
5. The number of NEW-ORDERS was verified from an RTE report covering the entire run.
6. The space used was divided by the number of NEW-ORDERS giving a space used per NEW-ORDER transaction.
7. 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 24.7 GB (including mirror) to sustain the log for 8 hours. Space available on the transaction log volume was 34.7 GB (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 Related Items

Throughput

Measured tpmC must be reported

Measured tpmC 4864.97 tpmC
Price per tpmC $62.49 per 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.

Table 5.2: Response Times

Type	Average	Maximum	90th %
New-Order	1.53	5.19	2.19
Payment	1.39	5.23	2.05
Order-Status	1.68	5.08	2.35
Interactive Delivery	0.13	0.84	0.15
Deferred Delivery	3.47	32.50	7.16
Stock-Level	3.51	7.77	4.62
Menu	0.13	3.14	0.15

Keying and Think Times

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

Table 5.3: Keying Times

Type	Minimum	Average	Maximum
New-Order	18.00	18.01	18.41
Payment	3.01	3.01	3.33
Order-Status	2.01	2.01	2.18
Interactive Delivery	2.01	2.01	2.31
Stock-Level	2.00	2.01	2.19

Table 5.4: Think Times

Type	Minimum	Average	Maximum
New-Order	0.00	12.29	124.70
Payment	0.00	12.23	122.31
Order-Status	0.00	10.15	93.00
Interactive Delivery	0.00	5.14	50.90
Stock-Level	0.00	5.21	45.50

Response Time Frequency Distribution Curves and Other Graphs

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

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

Think Time frequency distribution curves (see Clause 5.6.3) must be reported for each transaction type.

Keying Time frequency distribution curves (see Clause 5.6.4) must be reported for each transaction type.

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

Figure 5.1: New Order Response Time Distribution

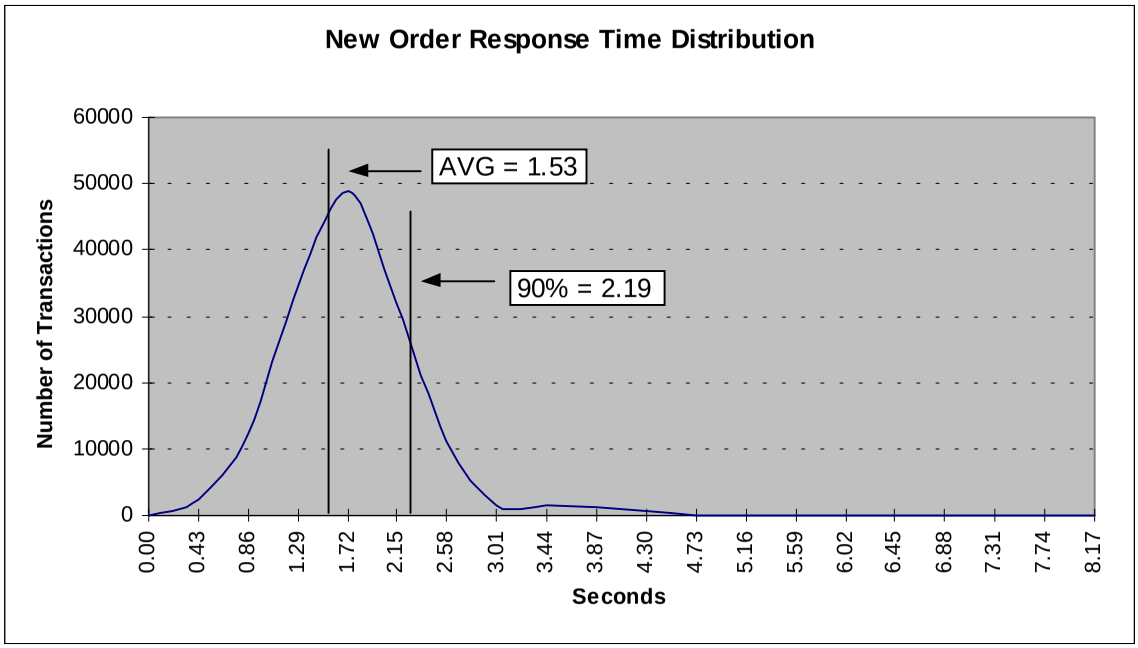


Figure 5.2: Payment Response Time Distribution

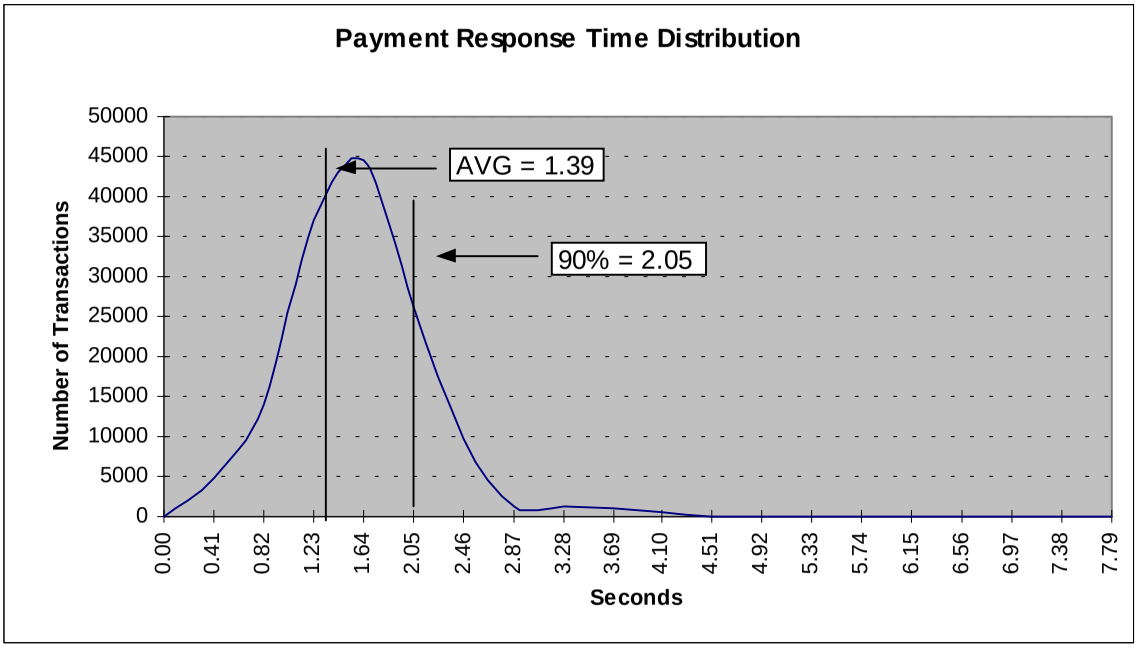


Figure 5.3: Order Status Response Time Distribution

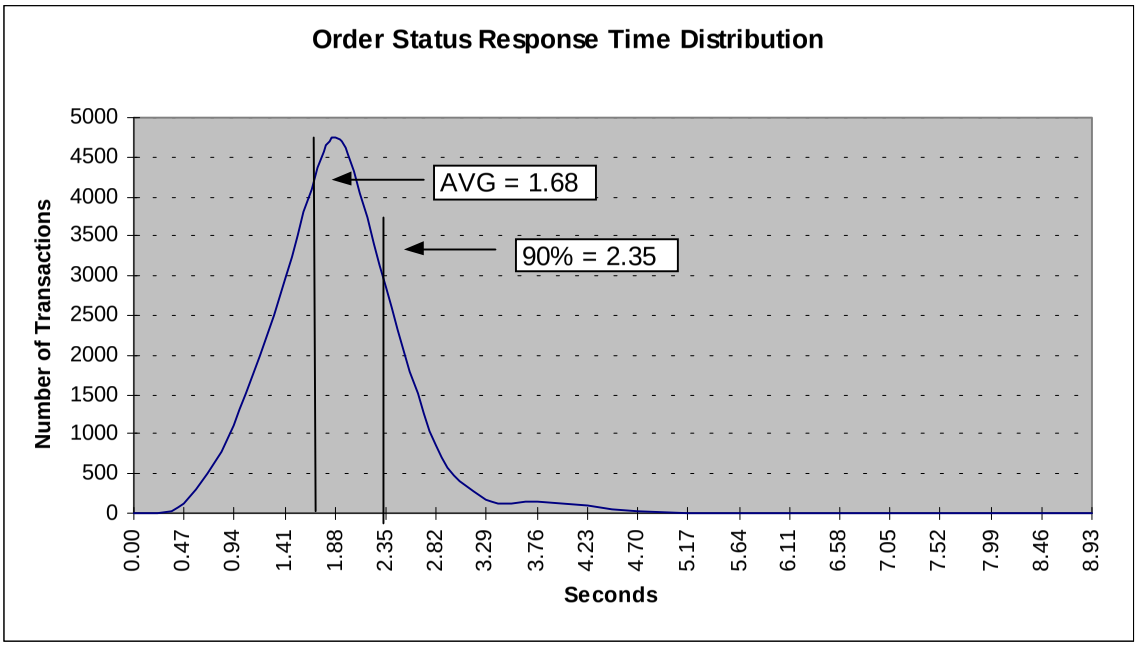


Figure 5.4: Delivery Response Time Distribution

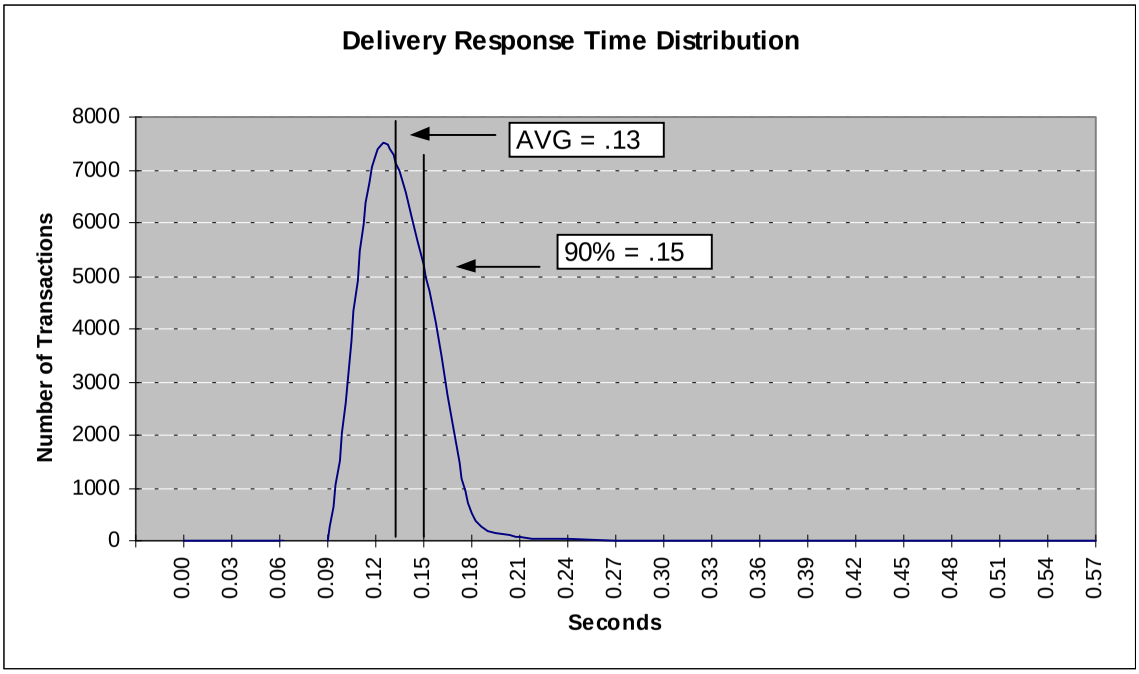


Figure 5.5: Stock Level Response Time Distribution

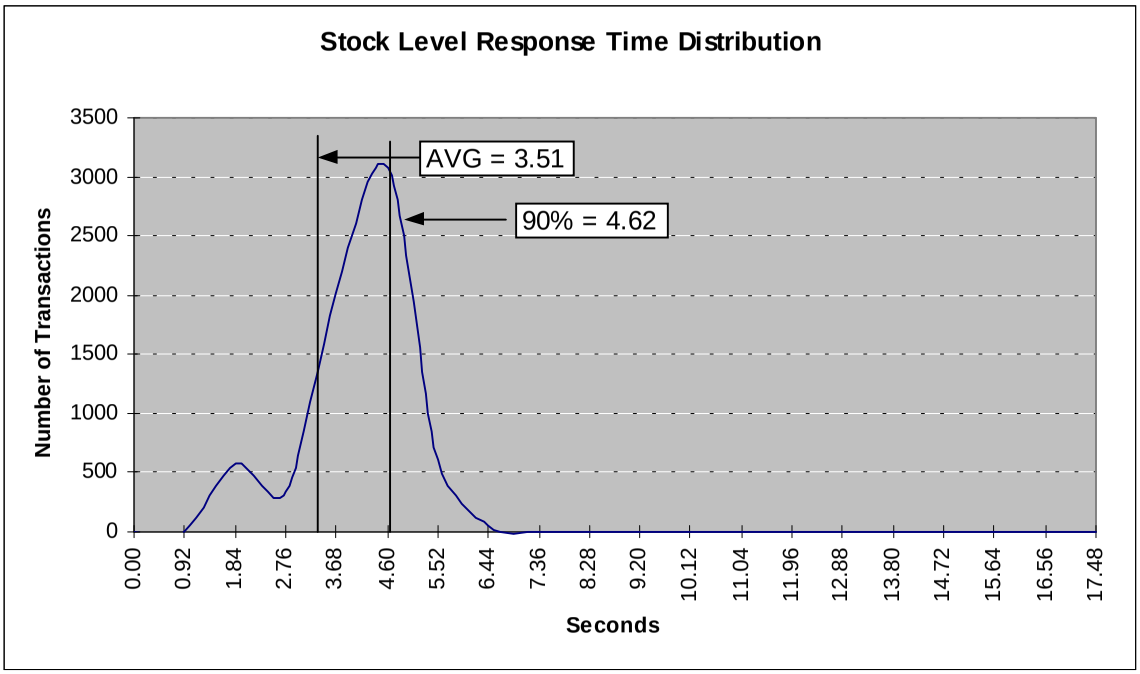


Figure 5.6: New Order Think Time Distribution

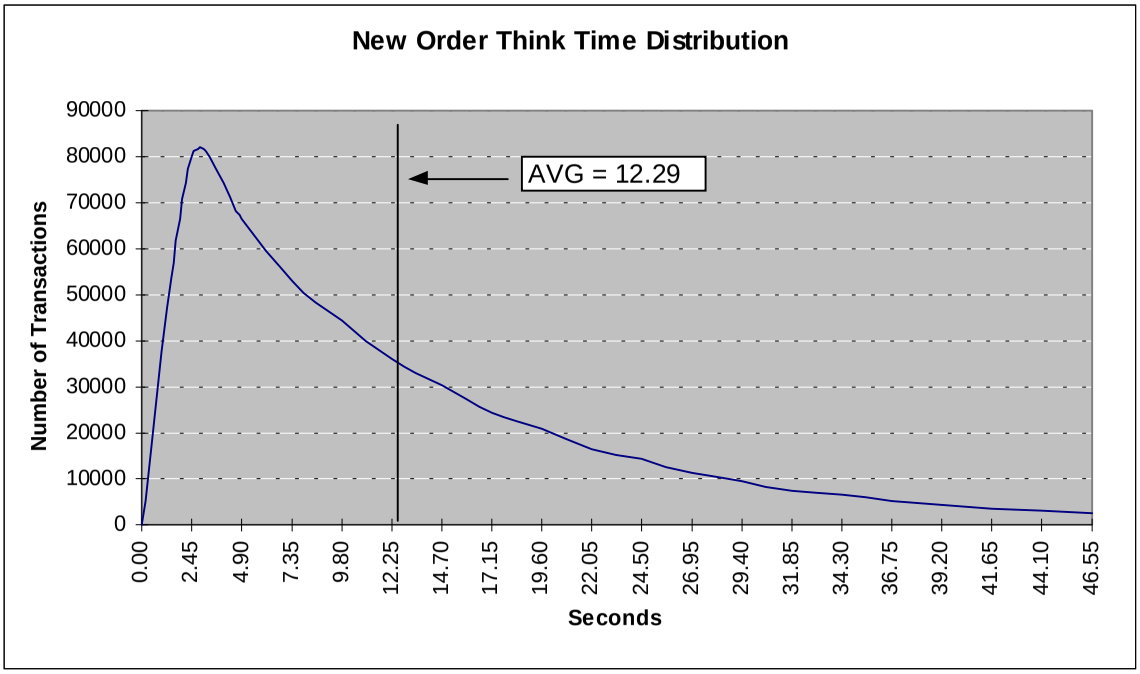


Figure 5.7: Response Time versus Throughput

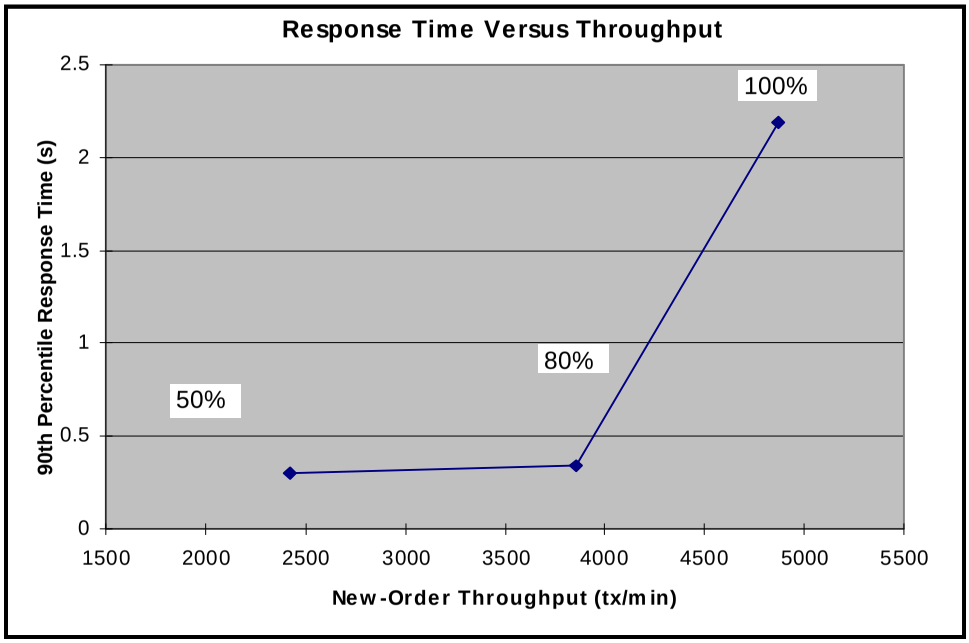
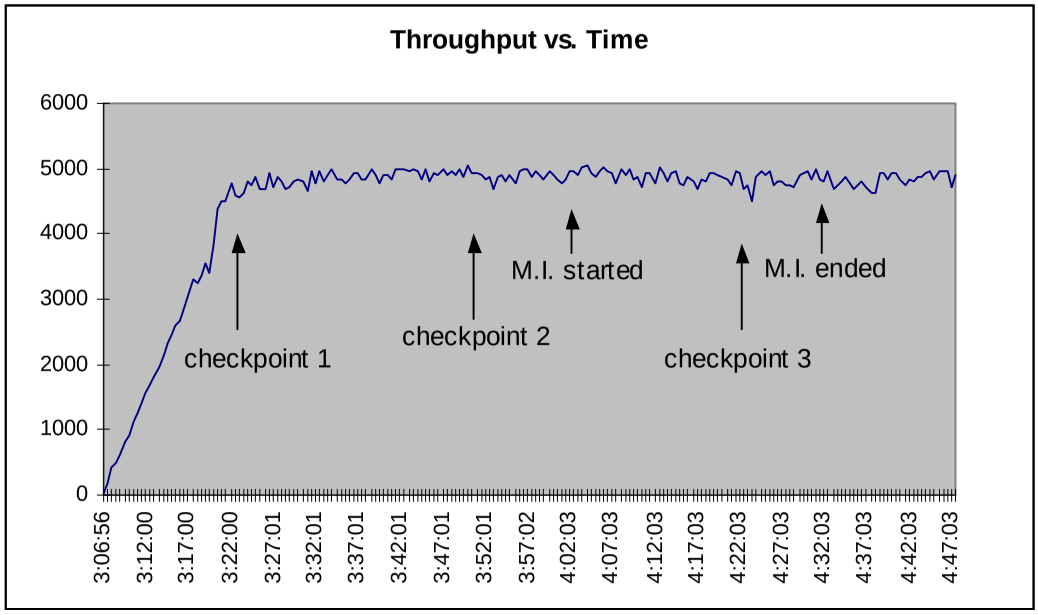


Figure 5.8: Throughput versus Time



Steady State Determination

The method used to determine that the SUT had reached a steady state prior to commencing the measurement interval must be disclosed.

Steady state was determined using real time monitor utilities from both the operating system and the RTE. Steady state was further confirmed by the throughput data collected during the run and graphed in Figure 5.17.

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.

The RTE generated the required input data to choose a transaction from the menu. This data was timestamped and captured in RTE log files before being transmitted. There was one log file for each user. The input screen for the requested transaction was returned and it was also captured and timestamped in the RTE log files. The difference between these two timestamps was the menu response time.

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 and captured in RTE log files. The return of the screen with the required response data was timestamped and captured in the RTE log files. The difference between these two timestamps was the response time for that transaction.

The RTE then waited the required think time interval before repeating the process starting at selecting a 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 5.17.

Reproducibility

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

We allowed the database to ramp up and to reach a steady state. The steady state was sustained for a 30-minute (measurement) interval, and was followed by a ramp-down. The repeatable interval result was within 0.003% of the reported interval result.

Measurement Period Duration

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

The reported measured interval was exactly 30 minutes long.

Regulation of Transaction Mix

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.

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

Transaction Statistics

The percentage of the total mix for each transaction type must be disclosed. The percentage of New-Order transactions rolled back as a result of invalid item number must be disclosed. The average number of order-lines entered per New-Order transaction must be disclosed. The percentage of remote order lines per New-Order transaction must be disclosed. The percentage of remote Payment transactions must be disclosed. The percentage of customer selections by customer last name in the Payment and Order-Status transactions must be disclosed. The percentage of Delivery transactions skipped due to there being fewer than necessary orders in the New-Order table must be disclosed.

Table 5.5: Transaction Statistics

Statistic		Value
New Order	Home warehouse order lines	89.60%
	Remote warehouse order lines	10.40%
	Rolled back transactions	0.92%
	Average items per order	10.03%
Payment	Home warehouse payments	84.83%
	Remote warehouse payments	15.17%
	Accessed by last name	60.43%
Order Status	Accessed by last name	61.21%
Transaction Mix	New Order	44.60%
	Payment	43.19%
	Order status	4.06%
	Delivery	4.05%
	Stock level	4.05%

Checkpoint Count and Location

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.

The initial checkpoint was started 15 minutes after the start of the ramp-up. The second checkpoint was started 30 minutes after the 1st checkpoint, and the third checkpoint was started 30 minutes after the 2nd. The checkpoint in the measurement interval lasted 77 seconds. The measurement interval contains the third checkpoint, begins 7.5 minutes after the second checkpoint (37.5 after the first), and is clear of the guard zones.

Clause 6 Related Items

RTE Descriptions

If the RTE is commercially available, then its inputs must be specified. Otherwise, a description must be supplied of what inputs (e.g., scripts) to the RTE had been used.

The RTE used was Empower by Performix, Inc. and the code used with Empower is included in Appendix A.

Emulated Components

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

The driver system consisted of six (6) Compaq ProLiant 4000 servers. The driver machines were attached to three (3) Compaq ProLiant 800 client machines through six (6) Ethernet LAN segments. Since this configuration is exactly the same devices and connectivity of the priced system, no components were being emulated. Therefore, the test described in Clause 6.6.3.4 was not required.

Functional 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 hardware and software functionality being performed on the Driver System and its interface to the SUT must be disclosed.

The driver system performed the data generation and input functions of the priced display device. It also captured the input and output data and timestamps for post-processing of the reported metrics. No other functionality was included on the driver system.

Section 1.4 of this report contains detailed diagrams of both the benchmark configuration and the priced configuration.

Networks

The network configuration of both the tested services and 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.

The bandwidth of the networks used in the tested/priced configuration must be disclosed.

In the tested configuration, six (6) LAN segments were used to connect six (6) driver (RTE) machines to three (3) client machines. 692 network connections were generated by each of the RTE machines[1] on six (6) LAN segments. The three (3) client machines were then connected to the server (SUT) via another LAN segment. All of the LAN segments between the six (6) RTE machines and the three (3) client machines were Ethernet LANs with a bandwidth of 10 megabits per second. The LAN segment between the three (3) client machines and the server (SUT) was an Ethernet LAN with a bandwidth of 100 megabits per second. The LAN segment connecting the client machines and the server machine supported 4150 concurrent users.

The priced configuration has 4150 physical workstations connected to the client machines via six (6) LAN segments, 692 workstations per six (6) segments. From the analysis of network utilization, we believe that for the priced configuration would sufficiently support 4150 physical workstations at the reported transaction rate and have network capacity still available.

Operator Intervention

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

---

[1] The sixth RTE generated 690 users.

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

Clause 7 Related Items

System Pricing

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 data. 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 and effective date(s) of price(s) must also be reported.

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.

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.

Availability, Throughput, and Price Performance

The committed delivery date for general availability (availability date) of products used in the price calculation must be reported. When the priced system included 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.

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

❑ **Maximum Qualified Throughput**	**4864.97 tpmC**
❑ **Price per tpmC**	**$62.49 per tpmC**
❑ **Hardware Available**	**May 05, 1997**
❑ **Software Available**	**Currently available**

All software components are currently available. The 512KB cache Pentium Pro 200 MHz processor boards will be available May 5, 1997.

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

This system is being priced for the United States of America.

Usage Pricing

For any usage pricing, the sponsor must disclose:

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

The component pricing based on usage is shown below:

- Four (4) Microsoft Windows NT 4.0 licenses
- One (1) Microsoft SQL Server v.6.50.242 (Includes five (5) User Licenses.
- One (1) Microsoft Internet Connector License
- One (1) Microsoft Visual C++ v. 4
- Compaq Servers include 3 years of support.

Clause 9 Related Items

Auditor's Report

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.

This implementation of the TPC Benchmark C was audited by Lorna Livingtree of Performance Metrics, Inc.

Performance Metrics, Inc.
2229 Benita Dr., Suite 101
Rancho Cordova, CA 95670
(phone) (916) 635-2822
(fax) (916) 858-0109
e-mail: lorna@perfmetrics.com

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

Compaq Computer Corporation
Database Performance Engineering
P.O. Box 692000
Houston, TX 77269-2000

PERFORMANCE METRICS INC.
TPC Certified Auditors

May 2, 1997

Mr. Syed Shabbir
Systems Engineer, Database Performance Engineering
Compaq Computer Corporation
20555 SH249
Houston, TX 77070

I have verified on site and remotely the TPC Benchmark™ C client/server for the following configuration:

Platform:	Compaq ProLiant 2500 6/200 Model-1H
Database Manager:	Microsoft SQL Server 6.5
Operating System:	Microsoft Windows NT Server version 4.0

Server: ProLiant 2500 6/200				
CPU's	Memory	Disks	90% Response	tpmC
2 PentiumPro @ 200 MHz	Main: 1 GB Cache: 512 KB	1 @ 2.1GB 79 @ 4.3 GB 4 @ 9.1GB	2.19 sec	4,864.97
3 Clients: Proliant 800 6/180 -Model 4300				
PentiumPro @ 180 Mhz	Main: 96 MB Cache: 256 KB	1 @ 4.3 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.

2229 Benita Dr. Suite 101, Rancho Cordova, CA 95670
(916) 635-2822 fax: (916) 858-0109 email: Lorna@PerfMetrics.com

- The database was properly scaled with 415 warehouses.
- The system loss durability test was 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.
- 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 performance was demonstrated to be repeatable.
- The system pricing was checked for major components and maintenance.
- Third party quotes were verified for compliance.

Sincerely,

Lorna Livingtree

Lorna Livingtree
Auditor

2229 Benita Dr. Suite 101, Rancho Cordova, CA 95670
(916) 635-2822 fax: (916) 858-0109 email: Lorna@PerfMetrics.com

Appendix A: Source Code

The client source code is listed below.

MIX

```
mon1, ./norate null mon1.l ./no1.log
mon2@d11, ./norate null mon2.l ./no2.log
mon3@d12, ./norate null mon3.l ./no3.log
mon4@d13, ./norate null mon4.l ./no4.log
mon5@d14, ./norate null mon5.l ./no5.log
mon6@d15, ./norate null mon6.l ./no6.log
user0001, ./tpcc -l 3 log/u0001.l client32b augustus_ip 1 1 415
user0002, ./tpcc -l 3 log/u0002.l client32b augustus_ip 1 2 415
user0003, ./tpcc -l 3 log/u0003.l client32b augustus_ip 1 3 415
user0004, ./tpcc -l 1 log/u0004.l client32b augustus_ip 1 4 415
user0005, ./tpcc -l 1 log/u0005.l client32b augustus_ip 1 5 415
          .
          .
          .
user0693@d11, ./tpcc -l 3 log/u0693.l client30a augustus_ip 70 3 415
user0694@d11, ./tpcc -l 3 log/u0694.l client30a augustus_ip 70 4 415
user0695@d11, ./tpcc -l 3 log/u0695.l client30a augustus_ip 70 5 415
user0696@d11, ./tpcc -l 1 log/u0696.l client30a augustus_ip 70 6 415
user0697@d11, ./tpcc -l 1 log/u0697.l client30a augustus_ip 70 7 415
          .
          .
          .
user2077@d13, ./tpcc -l 3 log/u2077.l client31a augustus_ip 208 7 415
user2078@d13, ./tpcc -l 3 log/u2078.l client31a augustus_ip 208 8 415
user2079@d13, ./tpcc -l 3 log/u2079.l client31a augustus_ip 208 9 415
user2080@d13, ./tpcc -l 1 log/u2080.l client31a augustus_ip 208 10 415
user2081@d13, ./tpcc -l 1 log/u2081.l client31a augustus_ip 209 1 415
          .
          .
          .
user2769@d14, ./tpcc -l 3 log/u2769.l client31b augustus_ip 277 9 415
user2770@d14, ./tpcc -l 3 log/u2770.l client31b augustus_ip 277 10 415
user2771@d14, ./tpcc -l 3 log/u2771.l client31b augustus_ip 278 1 415
user2772@d14, ./tpcc -l 1 log/u2772.l client31b augustus_ip 278 2 415
user2773@d14, ./tpcc -l 1 log/u2773.l client31b augustus_ip 278 3 415
          .
          .
          .
user4146@d15, ./tpcc -l 1 log/u4146.l client32a augustus_ip 415 6 415
user4147@d15, ./tpcc -l 1 log/u4147.l client32a augustus_ip 415 7 415
user4148@d15, ./tpcc -l 1 log/u4148.l client32a augustus_ip 415 8 415
user4149@d15, ./tpcc -l 1 log/u4149.l client32a augustus_ip 415 9 415
user4150@d15, ./tpcc -l 1 log/u4150.l client32a augustus_ip 415 10 415
```

DELIRPT.C

```
/*        FILE:               DELIRPT.C
 *                                    Microsoft TPC-C Kit Ver. 3.00.000
 *
 *                                    Copyright Microsoft, 1996
 *
 *        PURPOSE:  Delivery report processing application
 *        Author:             Philip Durr
 *                                    philipdu@Microsoft.com
 */

#include <windows.h>
#include <stdio.h>
#include <stdlib.h>

#define LOGFILE_READ_EOF      0
                              //check log file flag return current state
#define LOGFILE_CLEAR_EOF     1
                              //clear end of log file flag
#define LOGFILE_SET_EOF               2
                                      //set flag end of log file reached

#define INTERVAL                      .01
                                      //90th percentile calculation bucket interval

#define ERR_SUCCESS                                                   1000
                              //success no error
#define ERR_READING_LOGFILE                                   1001
                    //io errors occured reading delivery log file
#define ERR_INSUFFICIENT_MEMORY                               1002
                    //insuficient memory to process 90th percentile report
#define ERR_CANNOT_OPEN_RESULTS_FILE    1005
          //Cannot open delivery results file delilog.

typedef struct _RPTLINE
{
          SYSTEMTIME          start;
                                              //delilog report line start time
          SYSTEMTIME          end;
                                              //delilog report line end time
          int                           response;
                                              //delilog report line time delivery
took in milliseconds
          int                           w_id;
                                                        //delilog report line
warehouse id for delivery
          int                           o_carrier_id;
                                              //delilog report line carier id for
delivery
          int                           items[10];
                                                        //delilog report line
delivery line items
} RPTLINE, *PRPTLINE;

//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;

int                             versionMS = 3;
                        //delirpt version
int                             versionMM = 0;
int                             versionLS = 2;
int                             iReport;
                        //delirpt report to process
int                             iStartTime;
                                //begin times to accept for report
int                             iEndTime;
                        //end times to accept for report
FILE                    *fpLog;
                        //log file stream

//Local function prototypes
void            main(int argc, char *argv[]);
static int      Init(void);
static void     Restore(void);
static int      DoReport(void);
int                     AverageResponse(void);
int                     SkippedDelivery(void);
int                     Percentile90th(void);
BOOL            CheckTimes(PRPTLINE pRptLine);
static int      OpenLogFile(void);
static void     CloseLogFile(void);
static void     ResetLogFile(void);
static BOOL     LogEOF(int iOperation);
static BOOL     ReadReportLine(char *szBuffer, PRPTLINE pRptLine);
static BOOL     ParseReportLine(char *szLine, PRPTLINE pRptLine);
static BOOL     ParseDate(char *szDate, LPSYSTEMTIME pTime);
static BOOL     ParseTime(char *szTime, LPSYSTEMTIME pTime);
static void     ErrorMessage(int iError);
static BOOL     GetParameters(int argc, char *argv[]);
static void     PrintParameters(void);
static void     PrintHeader(void);
static void     cls(void);
static BOOL     IsNumeric(char *ptr);


/* FUNCTION: int main(int argc, char *argv[])
 *
 * PURPOSE:         This function is the beginning execution point for the delivery
executable.
 *
 * ARGUMENTS:       int             argc      number of command line arguments
passed to delivery
 *                                  char      *argv[]   array of command line
argument pointers
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

void main(int argc, char *argv[])
{
        int     iError;

        PrintHeader();

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

        if ( (iError=Init()) != ERR_SUCCESS )
        {
                ErrorMessage(iError);
                Restore();
                return;
        }

        if ( (iError = DoReport()) != ERR_SUCCESS )
                ErrorMessage(iError);

        Restore();

        return;
}

/* FUNCTION: static int Init(void)
 *
 * PURPOSE:         This function initializes the delirtp application.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static int Init(void)
{
        int iError;

        if ( (iError = OpenLogFile()) )
                return iError;
        return TRUE;
}

/* FUNCTION: static void Restore(void)
 *
 * PURPOSE:         This function cleans up the delirpt application before
termination.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static void Restore(void)
{
        CloseLogFile();
        return;
}
```

```
/* FUNCTION: static int DoReport(void)
 *
 * PURPOSE:                    This function dispatches the requested report.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                    ERR_SUCCESS if successfull or error code if an error
occurs.
 *
 * COMMENTS:        None
 *
 */

static int DoReport(void)
{
          int iRc;

          switch(iReport)
          {
                    case 1:
                              iRc = AverageResponse();
                              break;
                    case 2:
                              iRc = Percentile90th();
                              break;
                    case 3:
                              iRc = SkippedDelivery();
                              break;
                    case 4:
                              if ( (iRc = AverageResponse()) != ERR_SUCCESS )
                                        break;
                              if ( (iRc = Percentile90th()) != ERR_SUCCESS )
                                        break;
                              if ( (iRc = SkippedDelivery()) != ERR_SUCCESS )
                                        break;
                              break;
          }
          return iRc;
}

/* FUNCTION: int AverageResponse(void)
 *
 * PURPOSE:                    This function processes the AverageResponse report.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                    ERR_SUCCESS if successfull or error code if an error
occurs.
 *
 * COMMENTS:        None
 *
 */

int AverageResponse(void)
{
          RPTLINE reportLine;
          int                 iTotalResponse;
          int                 iLines;
          double    fAverage;
          char      szDelivery[128];

          ResetLogFile();

          iTotalResponse = 0;
          iLines = 0;
          printf("\n\n******** Average Response Time Report *******\n");
          while ( !LogEOF(LOGFILE_READ_EOF) )
          {
                    if ( ReadReportLine(szDelivery, &reportLine) )
                              return ERR_READING_LOGFILE;
                    if ( szDelivery[0] == '*' )
                              continue;
                    if ( !LogEOF(LOGFILE_READ_EOF) )
                    {
                              if ( CheckTimes(&reportLine) )
                                        continue;
                              iLines++;
                              iTotalResponse += reportLine.response;

                              if ( iLines % 10 == 0 )
                                        printf("Reading Report Line:\t%d\r", iLines);
                    }
          }
          printf("                                                \r");
          if ( iLines == 0 )
          {
                    printf("No deliveries found.\n");
          }
          else
          {
                    fAverage = ((double)iTotalResponse /
(double)iLines)/(double)1000;
                    printf("Total Deliveries:      %10.0f\n", (float)iLines);
                    printf("Total Response Times:  %10.3f\n",
((float)iTotalResponse/(float)1000));
                    printf("Average Response Time: %10.3f\n", fAverage);
          }

          return ERR_SUCCESS;
}

/* FUNCTION: int Percentile90th(void)
 *
 * PURPOSE:                    This function processes the 90th percentile report.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                    ERR_SUCCESS if successfull or error code if an error
occurs.
 *
 * COMMENTS:        This function requires enough space to allocate needed
 *                             buckets which will be 2 * max response time
in
 *                             deci-seconds.
 *
 */

int Percentile90th(void)
{
          RPTLINE reportLine;
          int                 iBucketSize;
          int                 i;
          int                 iResponseSeconds;
```

```
int                 iMaxSeconds;
int                 iTotalBuckets;
double    iTotal;
double    i90thPercent;
short     *psBuckets;
char      szDelivery[128];

printf("\n\n******** 90th Percentile *******\n");
printf("Calculating Max Response Seconds...\n");

ResetLogFile();

iMaxSeconds = -1;
while ( !LogEOF(LOGFILE_READ_EOF) )
{
          if ( ReadReportLine(szDelivery, &reportLine) )
                    return ERR_READING_LOGFILE;
          if ( szDelivery[0] == '*' )
                    continue;
          if ( !LogEOF(LOGFILE_READ_EOF) )
          {
                    if ( iMaxSeconds < reportLine.response )
                              iMaxSeconds = reportLine.response;
          }
}

printf("Max Response Time: %10.3f\n", (float)iMaxSeconds/(float)1000);

iTotalBuckets = iMaxSeconds + 1;

printf("Allocating Buckets...\n");

iBucketSize = iTotalBuckets * sizeof(short);

if ( !(psBuckets = (short *)malloc(iBucketSize)) )
          return ERR_INSUFFICIENT_MEMORY;

ZeroMemory(psBuckets, iBucketSize);

iTotal = 0;

ResetLogFile();
printf("Calculating Distribution...\n");

while ( !LogEOF(LOGFILE_READ_EOF) )
{
          if ( ReadReportLine(szDelivery, &reportLine) )
                    return ERR_READING_LOGFILE;
          if ( szDelivery[0] == '*' )
                    continue;
          if ( !LogEOF(LOGFILE_READ_EOF) )
          {
                    if ( CheckTimes(&reportLine) )
                              continue;
                    psBuckets[reportLine.response]++;
                    iTotal++;
          }
}

i90thPercent = iTotal * .9;

          for(i=0, iTotal = 0.0; iTotal < i90thPercent; iTotal +=
(double)psBuckets[i] )
                    i++;

          printf("90th Percentile = %d.%d\n", i/1000, (i % 1000));

          free(psBuckets);

          return ERR_SUCCESS;
}

/* FUNCTION: int SkippedDelivery(void)
 *
 * PURPOSE:                  This function processes the Skipped Deliveries report.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                  ERR_SUCCESS if successfull or error code if an error
occurs.
 *
 * COMMENTS:        None
 *
 */

int SkippedDelivery(void)
{
          RPTLINE reportLine;
          char      szDelivery[128];
          int                 i;
          int                 items[10];

          ResetLogFile();

          printf("\n\n******** Skipped Delivery Report *******\n");
          memset(items, 0, sizeof(items));
          printf("Reading Delivery Log File...");

          while ( !LogEOF(LOGFILE_READ_EOF) )
          {
                    if ( ReadReportLine(szDelivery, &reportLine) )
                              return ERR_READING_LOGFILE;
                    if ( szDelivery[0] == '*' )
                              continue;
                    if ( !LogEOF(LOGFILE_READ_EOF) )
                    {
                              if ( CheckTimes(&reportLine) )
                                        continue;
                              for(i=0; i<10; i++)
                              {
                                        if ( !reportLine.items[i] )
                                                  items[i]++;
                              }
                    }
          }
          printf("\n");
          printf("Skipped delivery table.\n");
          printf("  1    2    3    4    5    6    7    8    9   10 \n");
          printf("---- ---- ---- ---- ---- ---- ---- ---- ---- ----\n");
          for(i=0; i<10; i++)
                    printf("%4.4d ", items[i]);
          printf("\n");
```

```
        return ERR_SUCCESS;
}

/* FUNCTION: BOOL CheckTimes(PRPTLINE pRptLine)
 *
 * PURPOSE:         This function checks to see of the delilog record falls withing
the
 *                          begin and end time from the command line.
 *
 * ARGUMENTS:       PRPTLINE  pRptLine  delilog processed report line.
 *
 * RETURNS:                  BOOL      FALSE     if report line is not within the
 *
        requested start and end times.
 *                                                          TRUE      if the report
line is within the
 *
        requested start and end times.
 *
 * COMMENTS:        If startTime and endTime are both 0 then the user requested
 *                                  the default behavior which is all records in
delilog are
 *                                  valid.
 */

BOOL CheckTimes(PRPTLINE pRptLine)
{
        int       iRptEndTime;
        int       iRptStartTime;

        iRptStartTime = (pRptLine->start.wHour * 3600000) + (pRptLine-
>start.wMinute * 60000) + (pRptLine->start.wSecond * 1000) + pRptLine-
>start.wMilliseconds;
        iRptEndTime = (pRptLine->end.wHour * 3600000) + (pRptLine->end.wMinute *
60000) + (pRptLine->end.wSecond * 1000) + pRptLine->end.wMilliseconds;

        if ( iStartTime == 0 && iEndTime == 0 )
                return FALSE;

        if ( iStartTime <= iRptStartTime && iEndTime >= iRptEndTime )
                return FALSE;

        return TRUE;
}

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

static int OpenLogFile(void)
{
        fpLog = fopen("delilog.", "rb");

        if ( !fpLog )
                return ERR_CANNOT_OPEN_RESULTS_FILE;

        return ERR_SUCCESS;
}

/* FUNCTION: int CloseLogFile(void)
 *
 * PURPOSE:         This function closes the delivery log file.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                  None
 *
 * COMMENTS:        None
 *
 */

static void CloseLogFile(void)
{
        if ( fpLog )
                fclose(fpLog);

        return;
}


/* FUNCTION: static void ResetLogFile(void)
 *
 * PURPOSE:         This function prepares the delilog. file for reading
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                  None
 *
 * COMMENTS:        None
 *
 */

static void ResetLogFile(void)
{
        fseek(fpLog, 0L, SEEK_SET);
        LogEOF(LOGFILE_CLEAR_EOF);

        return;
}

/* FUNCTION: static BOOL LogEOF(int iOperation)
 *
 * PURPOSE:         This function tracks and reports the end of file condition
 *                          on the delilog file.
 *
 * ARGUMENTS:       int iOperation      requested operation this can be:
 *
        LOGFILE_READ_EOF    check log file flag return current state
 *
        LOGFILE_CLEAR_EOF   clear end of log file flag
 *
        LOGFILE_SET_EOF               set flag end of log file reached
```

```
 *
 *
 * RETURNS:                    None
 *
 * COMMENTS:        None
 *
 */

static BOOL LogEOF(int iOperation)
{
          static BOOL bEOF;

          switch(iOperation)
          {
                    case LOGFILE_READ_EOF:
                              return bEOF;
                              break;
                    case LOGFILE_CLEAR_EOF:
                              bEOF = FALSE;
                              break;
                    case LOGFILE_SET_EOF:
                              bEOF = TRUE;
                              break;
          }
          return FALSE;
}

/* FUNCTION: static BOOL ReadReportLine(char *szBuffer, PRPTLINE pRptLine)
 *
 * PURPOSE:         This function reads a text line from the delilog file.
 *                            on the delilog file.
 *
 * ARGUMENTS:       char                *szBuffer buffer to placed read delilog file
line into.
 *                            PRPTLINE  pRptLine  returned structure
containing parsed delilog
 *
                    report line.
 *
 * RETURNS:                   FALSE     if successfull or TRUE if an error occurs.
 *
 * COMMENTS:        None
 *
 */

static BOOL ReadReportLine(char *szBuffer, PRPTLINE pRptLine)
{
          int i = 0;
          int ch;
          int iEof;

          while( i < 128 )
          {
                    ch = fgetc(fpLog);
                    if ( iEof = feof(fpLog) )
                              break;
                    if ( ch == '\r' )
                    {
                              if ( i )
                                        break;
                              continue;
                    }
                    if ( ch == '\n' )
                              continue;
                    szBuffer[i++] = ch;

          }

          //delivery item format is to long cannot be a valid delivery item
          if ( i >= 128 )
                    return TRUE;

          szBuffer[i] = 0;
          if ( iEof )
          {
                    LogEOF(LOGFILE_SET_EOF);
                    if ( i == 0 )
                              return FALSE;
          }
          if ( szBuffer[0] == '*' )
          {
                    //error line ignore
                    return FALSE;
          }
          return ParseReportLine(szBuffer, pRptLine);
}

/* FUNCTION: static BOOL ParseReportLine(char *szLine, PRPTLINE pRptLine)
 *
 * PURPOSE:         This function reads a text line from the delilog file.
 *                            on the delilog file.
 *
 * ARGUMENTS:       char                *szLine             buffer containing the
delilog file line to be parsed.
 *                            PRPTLINE  pRptLine  returned structure
containing parsed delilog
 *
                    report line values.
 *
 * RETURNS:                   FALSE     if successfull or TRUE if an error occurs.
 *
 * COMMENTS:        None
 *
 */

static BOOL ParseReportLine(char *szLine, PRPTLINE pRptLine)
{
          int i;

          if ( ParseDate(szLine, &pRptLine->start) )
                    return TRUE;

          pRptLine->end.wYear = pRptLine->start.wYear;
          pRptLine->end.wMonth = pRptLine->start.wMonth;
          pRptLine->end.wDay = pRptLine->start.wDay;

          if ( !(szLine = strchr(szLine, ',')) )
                    return TRUE;
          szLine++;

          if ( ParseTime(szLine, &pRptLine->start) )
                    return TRUE;

          if ( !(szLine = strchr(szLine, ',')) )
```

```
                return TRUE;
        szLine++;

        if ( ParseTime(szLine, &pRptLine->end) )
                return TRUE;

        if ( !(szLine = strchr(szLine, ',')) )
                return TRUE;
        szLine++;

        if ( !IsNumeric(szLine) )
                return TRUE;
        pRptLine->response = atoi(szLine);

        if ( !(szLine = strchr(szLine, ',')) )
                return TRUE;
        szLine++;

        if ( !IsNumeric(szLine) )
                return TRUE;
        pRptLine->w_id = atoi(szLine);

        if ( !(szLine = strchr(szLine, ',')) )
                return TRUE;
        szLine++;

        if ( !IsNumeric(szLine) )
                return TRUE;
        pRptLine->o_carrier_id = atoi(szLine);

        if ( !(szLine = strchr(szLine, ',')) )
                return TRUE;
        szLine++;

        for(i=0; i<10; i++)
        {
                if ( !IsNumeric(szLine) )
                        return TRUE;
                pRptLine->items[i] = atoi(szLine);

                if ( i<9 && !(szLine = strchr(szLine, ',')) )
                        return TRUE;
                szLine++;
        }

        return FALSE;
}

/* FUNCTION: static BOOL ParseDate(char *szDate, LPSYSTEMTIME pTime)
 *
 * PURPOSE:         This function validates and extracts a date string in the format
 *                          yy/mm/dd into an SYSTEMTIME structure.
 *
 * ARGUMENTS:       char                            *szDate             buffer
containing the date to be parsed.
 *                                  LPSYSTEMTIME        pTime
        system time structure where date will be placed.
 *
 * RETURNS:                 FALSE if successfull or TRUE if an error occurs.
 *
 * COMMENTS:        None
 *
 */

static BOOL ParseDate(char *szDate, LPSYSTEMTIME pTime)
{
          if ( !isdigit(*szDate) || !isdigit(*(szDate+1)) || *(szDate+2) != '/' ||
                    !isdigit(*(szDate+3)) || !isdigit(*(szDate+4)) || *(szDate+5) !=
'/' ||
                    !isdigit(*(szDate+6)) || !isdigit(*(szDate+7)) )
                    return TRUE;

          pTime->wYear = atoi(szDate);

          pTime->wMonth = atoi(szDate+3);

          pTime->wDay = atoi(szDate+6);

          if ( pTime->wMonth > 12 || pTime->wMonth < 0 || pTime->wDay > 31 || pTime-
>wDay < 0 )
                    return TRUE;

          return FALSE;
}

/* FUNCTION: static BOOL ParseTime(char *szTime, LPSYSTEMTIME pTime)
 *
 * PURPOSE:         This function validates and extracts a time string in the format
 *                          hh:mm:ss:mmm into an SYSTEMTIME structure.
 *
 * ARGUMENTS:       char                            *szTime             buffer
containing the time to be parsed.
 *                                  LPSYSTEMTIME        pTime
          system time structure where date will be placed.
 *
 * RETURNS:                 FALSE if successfull or TRUE if an error occurs.
 *
 * COMMENTS:        None
 *
 */

static BOOL ParseTime(char *szTime, LPSYSTEMTIME pTime)
{
          if ( !isdigit(*szTime) || !isdigit(*(szTime+1)) || *(szTime+2) != ':' ||
                    !isdigit(*(szTime+3)) || !isdigit(*(szTime+4)) || *(szTime+5) !=
':' ||
                     !isdigit(*(szTime+6)) || !isdigit(*(szTime+7)) || *(szTime+8) !=
':' ||
                     !isdigit(*(szTime+9)) || !isdigit(*(szTime+10)) ||
!isdigit(*(szTime+11)) )
                     return TRUE;

          pTime->wHour = atoi(szTime);
          pTime->wMinute = atoi(szTime+3);
          pTime->wSecond = atoi(szTime+6);
          pTime->wMilliseconds  = atoi(szTime+9);

          if ( pTime->wHour > 23 || pTime->wHour < 0 ||
                    pTime->wMinute > 59 || pTime->wMinute < 0 ||
                    pTime->wSecond > 59 || pTime->wSecond < 0 ||
                    pTime->wMilliseconds < 0 )
                    return TRUE;

          if ( pTime->wMilliseconds > 999 )
```

```
        {
                pTime->wSecond += (pTime->wMilliseconds/1000);
                pTime->wMilliseconds = pTime->wMilliseconds  % 1000;
        }

        return FALSE;
}

/* FUNCTION: void ErrorMessage(int iError)
 *
 * PURPOSE:         This function displays an error message in the delivery
executable's console window.
 *
 * ARGUMENTS:       int                 iError    error id to be displayed
 *
 * RETURNS:                    None
 *
 * COMMENTS:        None
 *
 */

static void ErrorMessage(int iError)
{
        int i;

        static SERRORMSG errorMsgs[] =
        {
                {         ERR_SUCCESS,
                          "Success, no error."
                                                                          },
                {         ERR_CANNOT_OPEN_RESULTS_FILE,
        "Cannot open delivery results file delilog."
                },
                {         ERR_READING_LOGFILE,
                "Reading delivery log file, Delivery item format incorrect."
        },
                {         ERR_INSUFFICIENT_MEMORY,
        "insufficient memory to process 90th percentile report."          },
                {         0,
                                        ""

                                        }
        };

        for(i=0; errorMsgs[i].szMsg[0]; i++)
        {
                if ( iError == errorMsgs[i].iError )
                {
                        printf("\nError(%d): %s", iError, errorMsgs[i].szMsg);
                        return;
                }
        }
        printf("Error(%d): %s", errorMsgs[0].szMsg);
        return;
}

/* 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;
        SYSTEMTIME          startTime;
        SYSTEMTIME          endTime;

        iStartTime = 0;
        iEndTime = 0;
        iReport = 4;

        for(i=0; i<argc; i++)
        {
                if ( argv[i][0] == '-' || argv[i][0] == '/' )
                {
                        switch(argv[i][1])
                        {
                                case 'S':
                                case 's':
                                        if ( ParseTime(argv[i]+2,
&startTime) )
                                                return TRUE;
                                        iStartTime = (startTime.wHour *
3600000) + (startTime.wMinute * 60000) + (startTime.wSecond * 1000) +
startTime.wMilliseconds;
                                        break;
                                case 'E':
                                case 'e':
                                        if ( ParseTime(argv[i]+2, &endTime)
)
                                                return TRUE;
                                        iEndTime = (endTime.wHour *
3600000) + (endTime.wMinute * 60000) + (endTime.wSecond * 1000) +
endTime.wMilliseconds;
                                        break;
                                case 'R':
                                case 'r':
                                        iReport = atoi(argv[i]+2);
                                        if ( iReport > 4 || iReport < 1 )
                                                iReport = 4;
                                        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)
{
         PrintHeader();
         printf("DELIRPT:\n\n");
         printf("Parameter
Default\n");
         printf("-------------------------------------------------------------------
----\n");
         printf("-S Start Time HH:MM:SS:MMM                                        All
\n");
         printf("-E End Time HH:MM:SS:MMM                                          All
\n");
         printf("-R 1)Average Response, 2)90th 3) Skipped 4) All                   All
\n");
         printf("-? This help screen\n\n");
         printf("Note:  Command line switches are NOT case sensitive.\n");

         return;
}

/* FUNCTION: void PrintHeader(void)
 *
 * PURPOSE:         This function displays the delivery report applications banner
information.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static void PrintHeader(void)
{
         cls();

         printf("************************************************\n");
         printf("*                                              *\n");
         printf("*  Microsoft SQL Server 6.5                    *\n");
         printf("*                                              *\n");
         printf("*  HTML TPC-C BENCHMARK KIT: Delivery Report   *\n");
         printf("*  Version %d.%2.2d.%3.3d                          *\n",
versionMS, versionMM, versionLS);
         printf("*                                              *\n");
         printf("************************************************\n\n");

         return;
}

/* FUNCTION: void cls(void)
 *
 * PURPOSE:         This function clears the console window
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static void cls(void)
{
         HANDLE    hConsole;
         COORD     coordScreen = { 0, 0 };                         //here's where
we'll home the cursor
         DWORD     cCharsWritten;
         CONSOLE_SCREEN_BUFFER_INFO    csbi;             //to get buffer info
         DWORD                                                     dwConSize;
         //number of character cells in the current buffer

         hConsole = GetStdHandle(STD_OUTPUT_HANDLE);

         //get the number of character cells in the current buffer

         GetConsoleScreenBufferInfo( hConsole, &csbi );
         dwConSize = csbi.dwSize.X * csbi.dwSize.Y;

         //fill the entire screen with blanks
         FillConsoleOutputCharacter( hConsole, (TCHAR) ' ', dwConSize, coordScreen,
&cCharsWritten );
         GetConsoleScreenBufferInfo( hConsole, &csbi );

         //now set the buffer's attributes accordingly
         FillConsoleOutputAttribute( hConsole, csbi.wAttributes,dwConSize,
coordScreen, &cCharsWritten );

         //put the cursor at (0, 0)
         SetConsoleCursorPosition( hConsole, coordScreen );

         return;
}

/* 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:        A comma is counted as a valid delimiter.
 *
 */

static BOOL IsNumeric(char *ptr)
{
```

```
        if ( *ptr == 0 )
                return FALSE;

        while( *ptr && isdigit(*ptr) )
                ptr++;
        if ( !*ptr || *ptr == ',' )
                return TRUE;
        else
                return FALSE;
}
```

DELISRV.C

```
/*      FILE:           DELISRV.C
 *                              Microsoft TPC-C Kit Ver. 3.00.000
 *                              Audited 08/23/96, By Francois Raab
 *
 *                              Copyright Microsoft, 1996
 *
 *      PURPOSE:  Delivery TPC-C transaction executable
 *      Author:         Philip Durr
 *                              philipdu@Microsoft.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>
#include <conio.h>
#include <ctype.h>

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

#include "delisrv.h"

char                                    szServer[32];
        //SQL server name
char                                    szDatabase[32];
        //tpcc database name
char                                    szUser[32];
                //user name
char                                    szPassword[32];
        //user password
int                                             iNumThreads
        = 4;            //number of threads to create
int                                             iDelayMs        =
1000;           //delay between delivery queue checks
int                                             iDeadlockRetry  = 3;
        //number of read check retries.
int                                             iQSlotts        =
3000;           //delivery transaction queues
```

```
FILE                                            *fpLog;
                                //pointer to log file
CRITICAL_SECTION        WriteLogCriticalSection;        //critical section for
delivery write log
CRITICAL_SECTION        DeliveryCriticalSection;        //critical section for
delivery transactions cache
static  LPTSTR                          lpszPipeName = TEXT("\\\\.\\pipe\\DELISRV");
        //delivery pipe name

HANDLE                                          hPipe           =
INVALID_HANDLE_VALUE;   //delivery pipe handle
HANDLE                                          hComPort  = INVALID_HANDLE_VALUE;
        //delivery pipe completion port handle.

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

LPDELIVERY_PACKET       pDeliveryCache;

int                                                     versionMS = 3;
                                //delivery executable version number.
int                                                     versionMM = 0;
                                //formatted as MS.MM.LS, 1.00.005
int                                                     versionLS = 2;

/* FUNCTION: int main(int argc, char *argv[])
 *
 * PURPOSE:         This function is the beginning execution point for the delivery
executable.
 *
 * ARGUMENTS:       int             argc    number of command line arguments
passed to delivery
 *                                      char    *argv[]  array of command line
argument pointers
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

void main(int argc, char *argv[])
{
        int     iError;

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

        if ( (iError=Init()) )
        {
                ErrorMessage(iError);
                Restore();
                return;
        }
```

```
        if ( (iError = RunDelivery()) != ERR_SUCCESS )
                ErrorMessage(iError);

        Restore();

        return;
}

/* FUNCTION: void cls(void)
 *
 * PURPOSE:         This function clears the console window
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static void cls(void)
{
        HANDLE    hConsole;
        COORD     coordScreen = { 0, 0 };                       //here's where
we'll home the cursor
        DWORD     cCharsWritten;
        CONSOLE_SCREEN_BUFFER_INFO    csbi;           //to get buffer info
        DWORD                                                   dwConSize;
        //number of character cells in the current buffer

        hConsole = GetStdHandle(STD_OUTPUT_HANDLE);

        //get the number of character cells in the current buffer

        GetConsoleScreenBufferInfo( hConsole, &csbi );
        dwConSize = csbi.dwSize.X * csbi.dwSize.Y;

        //fill the entire screen with blanks
        FillConsoleOutputCharacter( hConsole, (TCHAR) ' ', dwConSize, coordScreen,
&cCharsWritten );
        GetConsoleScreenBufferInfo( hConsole, &csbi );

        //now set the buffer's attributes accordingly
        FillConsoleOutputAttribute( hConsole, csbi.wAttributes,dwConSize,
coordScreen, &cCharsWritten );

        //put the cursor at (0, 0)
        SetConsoleCursorPosition( hConsole, coordScreen );

        return;
}

/* FUNCTION: int RunDelivery(void)
 *
 * PURPOSE:         This function executes the main delivery executable loop.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 int       ERR_CANNOT_OPEN_PIPE
        cannot open named pipe
 *                                              ERR_CANNOT_CREATE_THREAD
        cannot create required threads
 *                                              ERR_SUCCESS
                        successfull no error
 *
 *
 * COMMENTS:        None
 *
 */

static int RunDelivery(void)
{
        SECURITY_ATTRIBUTES sa;
        int                                                     i;

        cls();

        PrintHeader();

        printf("\n<Starting Delivery Service with %d Threads.>\n", iNumThreads);
        printf("\nPress <Ctrl>C to exit.\n");

        bDone = FALSE;
        _beginthread( CheckKey, 0, NULL );

        printf("\nWaiting for delivery pipe: ");

        while( !bDone )
        {
                AnimateWait1();
                if ( WaitNamedPipe(lpszPipeName, NMPWAIT_USE_DEFAULT_WAIT) )
                {
                        sa.nLength                                              =
sizeof(sa);
                        sa.lpSecurityDescriptor       = NULL;
                        sa.bInheritHandle             = TRUE;

                        hPipe = CreateFile(lpszPipeName, GENERIC_READ |
GENERIC_WRITE, FILE_SHARE_READ | FILE_SHARE_WRITE, NULL, OPEN_EXISTING,
                                        FILE_FLAG_OVERLAPPED, NULL);
                        if ( hPipe  == INVALID_HANDLE_VALUE )
                                return ERR_CANNOT_OPEN_PIPE;
                        hComPort = CreateIoCompletionPort(hPipe, NULL, 0, 256);
                        break;
                }
                Sleep(100);
        }

        if ( !bDone )
        {
                if ( _beginthread( DeliveryHandler, 0, NULL ) == -1 )
                        return ERR_CANNOT_CREATE_THREAD;

                for(i=0; i<iNumThreads; i++)
                {
                        if ( _beginthread( DeliveryThread, 0, NULL ) == -1 )
                                return ERR_CANNOT_CREATE_THREAD;
                }

                printf(" \nRunning : ");

                while( !bDone )
                        AnimateWait();
        }
```

```
        return ERR_SUCCESS;
}

/* FUNCTION: void AnimateWait1(void)
 *
 * PURPOSE:         This function provides a visual indicator that the delivery
executable is waiting for
 *                          the delivery pipe to appear.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static void AnimateWait1(void)
{
        const static char szStr[] = "+-|*";
        static char *ptr = (char *)szStr;

        printf("%c\x8", *ptr);
        ptr = (*(ptr+1)) ? ptr + 1 : (char *)szStr;
        Sleep(100);

        return;
}

/* FUNCTION: void AnimateWait(void)
 *
 * PURPOSE:         This function provides a visual indicator that the delivery
executable is waiting for
 *                          and processing transactions.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static void AnimateWait(void)
{
        const static char szStr[] = "/-\\|/-\\|";
        static char *ptr = (char *)szStr;

        printf("%c\x8", *ptr);
        ptr = (*(ptr+1)) ? ptr + 1 : (char *)szStr;
        Sleep(100);

        return;
}


/* FUNCTION: int Init(void)
 *
 * PURPOSE:         This function prepares the delivery executable for processing.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 int       iError              Error code if
unsuccessfull
 *                                                      ERR_SUCCESS         No error
successfull code
 *
 *
 * COMMENTS:        None
 *
 */

static int Init(void)
{
        int       iError;

        InitializeCriticalSection(&WriteLogCriticalSection);
        InitializeCriticalSection(&DeliveryCriticalSection);

        fpLog     = NULL;

        if ( !(pDeliveryCache = malloc(sizeof(DELIVERY_PACKET) * iQSlotts)) )
                return ERR_INSUFFICIENT_MEMORY;

        memset(pDeliveryCache, 0, sizeof(DELIVERY_PACKET) * iQSlotts);

        if ( (iError = ReadRegistrySettings()) )
                return iError;

        if ( (iError=OpenLogFile()) )
                return iError;

        //initialize db library for use
        dbinit();

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

        return ERR_SUCCESS;
}

/* FUNCTION: void Restore(void)
 *
 * PURPOSE:         This function cleans up allocated objects to allow for
termination of the
 *                          delivery executable.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static void Restore(void)
{
        int       iret, l, d;

        DeleteCriticalSection(&WriteLogCriticalSection);
        DeleteCriticalSection(&DeliveryCriticalSection);
```

```
        l = 1;
        iret = WriteFile(hPipe, &l, 1, &d, NULL);

        if ( hPipe  != INVALID_HANDLE_VALUE )
                iret = CloseHandle(hPipe);

        if ( fpLog )
                fclose(fpLog);

        fpLog = NULL;

        dbexit();

        return;
}

/* FUNCTION: void ErrorMessage(int iError)
 *
 * PURPOSE:         This function displays an error message in the delivery
executable's console window.
 *
 * ARGUMENTS:       int                 iError    error id to be displayed
 *
 * RETURNS:                    None
 *
 * COMMENTS:        None
 *
 */

static void ErrorMessage(int iError)
{
        int i;

        static SERRORMSG errorMsgs[] =
        {
                {       ERR_SUCCESS,
                        "Success, no error."
                                },
                {       ERR_CANNOT_CREATE_THREAD,
        "Cannot create thread."
                },
                {       ERR_DBGETDATA_FAILED,
                "Get data failed."
                        },
                {       ERR_REGISTRY_NOT_SETUP,
                "Registry not setup for tpcc."
                },
                {       ERR_CANNOT_ACCESS_DELIVERY_FN,
        "Cannot access ReadDelivery cache."                             },
                {       ERR_CANNOT_ACCESS_REGISTRY,
        "Cannot access registry key TPCC."                              },
                {       ERR_CANNOT_CREATE_RESULTS_FILE,
        "Cannot create results file."                                   },
                {       ERR_CANNOT_OPEN_PIPE,
                "Cannot open delivery pipe."
        },
                {       ERR_READ_PIPE,
                        "Reading Delivery Pipe."
                        },
                {       ERR_INSUFFICIENT_MEMORY,
        "Insufficient memory."
                },
                {       0,
                                        ""

        }
        };

        for(i=0; errorMsgs[i].szMsg[0]; i++)
        {
                if ( iError == errorMsgs[i].iError )
                {
                        printf("\nError(%d): %s", iError, errorMsgs[i].szMsg);
                        if ( fpLog )
                        {

        EnterCriticalSection(&WriteLogCriticalSection);
                                        fprintf(fpLog, "*Error(%d): %s\r\n", iError,
errorMsgs[i].szMsg);
                                        if ( bFlush )
                                                fflush(fpLog);

        LeaveCriticalSection(&WriteLogCriticalSection);
                                }
                                return;
                        }
                }

                printf("Error(%d): Unknown Error.");
                EnterCriticalSection(&WriteLogCriticalSection);
                fprintf(fpLog, "*Error(%d): Unknown Error.\r\n", iError);
                if ( bFlush )
                        fflush(fpLog);
                LeaveCriticalSection(&WriteLogCriticalSection);

                return;
}

/* 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 'D':
                                case 'd':
                                        strcpy(szDatabase, argv[i]+2);
                                        break;
                                case 'U':
                                case 'u':
                                        strcpy(szUser, argv[i]+2);
                                        break;
                                case 'P':
                                case 'p':
                                        strcpy(szPassword, argv[i]+2);
                                        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)
{
        PrintHeader();
        printf("DELISRV:\n\n");
        printf("Parameter
Default\n");
        printf("-------------------------------------------------------------------
----\n");
        printf("-S Server
\n");
        printf("-D Database
tpcc   \n");
        printf("-U Username                                                    sa
\n");
        printf("-P Password
\n");
        printf("-F Flush output to delilog file when written.                  OFF
\n");
        printf("-? This help screen\n\n");
        printf("Note:  Command line switches are NOT case sensitive.\n");

        return;
}

/* FUNCTION: void PrintHeader(void)
 *
 * PURPOSE:         This function displays the delivery executable's banner
information.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static void PrintHeader(void)
{
        printf("************************************************\n");
        printf("*                                              *\n");
        printf("*  Microsoft SQL Server 6.5                    *\n");
        printf("*                                              *\n");
        printf("*  HTML TPC-C BENCHMARK KIT: Delivery Server   *\n");
        printf("*  Version %d.%2.2d.%3.3d                           *\n",
versionMS, versionMM, versionLS);
        printf("*                                              *\n");
        printf("************************************************\n\n");

        return;
}

/* FUNCTION: int ReadRegistrySettings(void)
 *
 * PURPOSE:         This function reads the system registry filling in required key
parameters.
 *
 * ARGUMENTS:       None
 *
 * RETURNS:                 int       ERR_REGISTRY_NOT_SETUP
        registry not setup tpcc.exe needs to be run
 *
                                        to setup registry.
 *                                             ERR_SUCCESS
                                Reqistry read Successfull, no error
 *
 *
 * COMMENTS:        None
 */

static int ReadRegistrySettings(void)
{
```

```
        HKEY      hKey;
        DWORD     size;
        DWORD     type;
        char      szTmp[256];

        if ( RegOpenKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\\Microsoft\\TPCC", 0,
KEY_READ, &hKey) != ERROR_SUCCESS )
                  return ERR_REGISTRY_NOT_SETUP;

        size = sizeof(szTmp);

        iNumThreads = 4;
        if ( RegQueryValueEx(hKey, "NumberOfDeliveryThreads", 0, &type, szTmp,
&size) == ERROR_SUCCESS )
                  iNumThreads = atoi(szTmp);
                  if ( !iNumThreads )
                            iNumThreads = 4;

        iDelayMs = 1000;
        if ( RegQueryValueEx(hKey, "BackoffDelay", 0, &type, szTmp, &size) ==
ERROR_SUCCESS )
                  iDelayMs = atoi(szTmp);
                  if ( !iDelayMs )
                            iDelayMs = 1000;

        iDeadlockRetry = 3;
        if ( RegQueryValueEx(hKey, "DeadlockRetry", 0, &type, szTmp, &size) ==
ERROR_SUCCESS )
                  iDeadlockRetry = atoi(szTmp);
                  if ( !iDeadlockRetry )
                            iDeadlockRetry = 3;

        iQSlotts = 3000;
        size = sizeof(szTmp);
        if ( RegQueryValueEx(hKey, "QueueSlotts", 0, &type, szTmp, &size) ==
ERROR_SUCCESS )
                  iQSlotts = atoi(szTmp);
                  if ( !iQSlotts )
                            iQSlotts = 3000;

        RegCloseKey(hKey);

        return ERR_SUCCESS;
}

/* FUNCTION: void CheckKey(void *ptr)
 *
 * PURPOSE:         This function checks for a key press on the delivery executable's
console. If the
 *                            key press is a Ctrl C then the execution termination
flag variable bDone is set to
 *                            TRUE which will start the termination of the delivery
executable.
 *
 * ARGUMENTS:       void      *ptr      dummy argument passed in though thread
manager, unused NULL.
 *
 * RETURNS:                   None
 *
 * COMMENTS:        None
 *
 */

static void CheckKey(void *ptr)
{
        while( _getch() != CTRL_C)
                  ;
        bDone = TRUE;

        return;
}

/* FUNCTION: void DeliveryHandler( void *ptr )
 *
 * PURPOSE:         This function is executed in it's own thread what it does is to
check for delivery
 *                            postings in the delivery named pipe. If any are present
then it pulls them off and
 *                            places them in the next available delivery queue array
element.
 *
 * ARGUMENTS:       void      *ptr      dummy argument passed in though thread
manager, unused NULL.
 *
 * RETURNS:                   None
 *
 * COMMENTS:        None
 *
 */

static void DeliveryHandler( void *ptr )
{
        int       i;
        int       size;
        int       iError;

        while( !bDone )
        {
                  for(i=0; i<iQSlotts; i++)
                  {
                            if ( !pDeliveryCache[i].bInUse )
                                      break;
                  }
                  if ( i < iQSlotts )
                  {
                            EnterCriticalSection(&DeliveryCriticalSection);
                            pDeliveryCache[i].bInUse = TRUE;
                            LeaveCriticalSection(&DeliveryCriticalSection);
                  }
                  else
                  {
                            EnterCriticalSection(&DeliveryCriticalSection);
                            if ( !(pDeliveryCache =
(LPDELIVERY_PACKET)realloc(pDeliveryCache, sizeof(DELIVERY_PACKET) * (iQSlotts+512)))
)
                            {
                                      ErrorMessage(ERR_INSUFFICIENT_MEMORY);

        LeaveCriticalSection(&DeliveryCriticalSection);
                                      return;
                            }
                            for(i=iQSlotts; i<iQSlotts+512; i++)
                                      pDeliveryCache[i].bInUse = FALSE;
                            i = iQSlotts;
```

```
                        pDeliveryCache[i].bInUse = TRUE;
                        LeaveCriticalSection(&DeliveryCriticalSection);
                }

                pDeliveryCache[i].ov.Offset                     = i;
                pDeliveryCache[i].ov.Internal           = 0;
                pDeliveryCache[i].ov.InternalHigh       = 0;
                pDeliveryCache[i].ov.OffsetHigh                 = 1;
                pDeliveryCache[i].ov.hEvent                     = NULL;

                while( !bDone )
                {
                        if ( ReadFile(hPipe, &pDeliveryCache[i].trans,
sizeof(DELIVERY_TRANSACTION), &size, &pDeliveryCache[i].ov) )
                                break;
                        if ( bDone )
                                break;
                        iError = GetLastError();
                        if ( iError == ERROR_IO_PENDING )
                        {
                                while( pDeliveryCache[i].ov.OffsetHigh )
                                        Sleep(10);
                                break;
                        }
                        else
                        {
                                ErrorMessage(ERR_READ_PIPE);
                                return;
                        }
                }
                Sleep(1);
        }

        return;
}

/* FUNCTION: void DeliveryThread( void *ptr )
 *
 * PURPOSE:         This function is executed inside the delivery threads. The queue
array
 *                  is continuously check and if any array elements are in
use then the
 *                  array entry is read, cleared and this function
processes it.
 *
 * ARGUMENTS:       void      *ptr      dummy argument passed in though thread
manager, unused NULL.
 *
 * RETURNS:                   None
 *
 * COMMENTS:        The registry key HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\TPCC
 *                            value NumberOfDeliveryThreads controls how
many of these
 *                            functions are running. The
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\TPCC
 *                            value BackoffDelay controls the amount of
time this function waits
 *                            between checks of the delivery queue.
 *
 */

static void DeliveryThread( void *ptr )
{
        int                                     size;
        int                                     key;
        LPOVERLAPPED        pov;
        DELIVERY            delivery;
        int                                     iError;

        if ( SQLOpenConnection(&delivery.dbproc, szServer, szDatabase, szUser,
szPassword, &delivery.spid) )
                return;   //error posting tbd

        //while delisrv running i.e. user has not requested termination
        while( !bDone )
        {
                if ( GetQueuedCompletionStatus(hComPort, &size, &key, &pov,
(DWORD)-1) )
                {
                        pov->OffsetHigh = 0;          //clear to notify
delivery handler ok to read another entry.
                        //some delivery to do so process it
                        memcpy(&delivery.queue, &pDeliveryCache[pov-
>Offset].trans.queue, sizeof(SYSTEMTIME));
                        delivery.w_id                           =
pDeliveryCache[pov->Offset].trans.w_id;
                        delivery.o_carrier_id         = pDeliveryCache[pov-
>Offset].trans.o_carrier_id;

                        if ( (iError=SQLDelivery(&delivery)) )
                        {
                                ErrorMessage(iError);
                                printf("Running : ");
                                continue;
                        }

                        //update log
                        WriteLog(&delivery);

                        EnterCriticalSection(&DeliveryCriticalSection);
                        pDeliveryCache[pov->Offset].bInUse = FALSE;
                        LeaveCriticalSection(&DeliveryCriticalSection);
                }
        }

        return;
}

/* FUNCTION: static 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
 *
 */

static int err_handler(DBPROCESS *dbproc, int severity, int dberr, int oserr, char
*dberrstr, char *oserrstr)
{
        if (oserr != DBNOERR)
                printf("(%d) %s", oserr, oserrstr);

        if ((dbproc == NULL) || (DBDEAD(dbproc)))
                ExitThread((unsigned long)-1);

        return INT_CONTINUE;
}

/* FUNCTION: static 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.
 *
 */

static int msg_handler(DBPROCESS *dbproc, DBINT msgno, int msgstate, int severity,
char *msgtext)
{
    if ( (msgno == 5701) || (msgno == 2528) || (msgno == 5703) || (msgno == 6006) )
                return INT_CONTINUE;

    // deadlock message
    if (msgno == 1205)
    {
                // set the deadlock indicator
                if (dbgetuserdata(dbproc) != NULL)
                *((BOOL *) dbgetuserdata(dbproc)) = TRUE;
                else
                printf("\nError, dbgetuserdata returned NULL.\n");
                return INT_CONTINUE;

    }

        if (msgno == 0)
                return INT_CONTINUE;
        else
                printf("SQL Server Message (%ld) : %s\n", msgno, msgtext);
        return INT_CANCEL;
}

/* FUNCTION: BOOL SQLOpenConnection(DBPROCESS **dbproc, char *server, char *database,
char *user, char *password, int *spid)
 *
 * PURPOSE:         This function opens the sql connection for use.
 *
 * ARGUMENTS:       DBPROCESS           **dbproc  pointer to returned DBPROCESS
 *                                      char                        *server
        SQL server name
 *                                      char                        *database SQL
server database
 *                                      char                        *user
        user name
 *                                      char                        *password user
password
 *                                      int                                 *spid
                pointer to returned spid
 *
 * RETURNS:                  BOOL      FALSE     if successfull
 *                                                 TRUE      if an error
occurs
 *
 * COMMENTS:        None
 *
 */

static BOOL SQLOpenConnection(DBPROCESS **dbproc, char *server, char *database, char
*user, char *password, int *spid)
{
        LOGINREC   *login;

        login = dblogin();
        DBSETLUSER(login, user);
        DBSETLPWD(login,  password);

        DBSETLPACKET(login, (USHORT)DEFCLPACKSIZE);

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

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

        dbsetuserdata(*dbproc, malloc(sizeof(BOOL)));
        *((BOOL *)dbgetuserdata(*dbproc)) = FALSE;

        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);

        return FALSE;
}


//queue time, end time, elapsed time, w_id, o_carrier_id, o_id1, ... o_id10
/* FUNCTION: void WriteLog(LPDELIVERY pDelivery)
 *
 * PURPOSE:         This function writes the delivery results to the delivery log
file.
 *
 * ARGUMENTS:       LPDELIVERY          pDelivery Pointer to delivery information.
 *
 * RETURNS:                 None
 *
 * COMMENTS:        None
 *
 */

static void WriteLog(LPDELIVERY pDelivery)
{
        int elapsed;

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

        EnterCriticalSection(&WriteLogCriticalSection);

        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,%d\r\n",
                pDelivery->trans_end.wYear - 1900, pDelivery->trans_end.wMonth,
pDelivery->trans_end.wDay,
                pDelivery->queue.wHour, pDelivery->queue.wMinute, pDelivery-
>queue.wSecond, pDelivery->queue.wMilliseconds,
                pDelivery->trans_end.wHour, pDelivery->trans_end.wMinute,
pDelivery->trans_end.wSecond, pDelivery->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);

        LeaveCriticalSection(&WriteLogCriticalSection);

        return;
}

/* 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(DELIVERY *pDelivery)
 *
 * 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 successfull, no error
 *
 *
 * COMMENTS:        None
 *
 */

static int SQLDelivery(DELIVERY *pDelivery)
{
```

```
        RETCODE   rc;
        int               i;
        int               deadlock_count;
        BYTE      *pData;

        deadlock_count = 0;


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

                        if (dbrpcexec(pDelivery->dbproc) == SUCCEED)
                        {
                                while (((rc = dbresults(pDelivery->dbproc))
!= NO_MORE_RESULTS) && (rc != FAIL))
                                {
                                        while (((rc = dbnextrow(pDelivery-
>dbproc)) != NO_MORE_ROWS) && (rc != FAIL))
                                        {
                                                for (i=0;i<10;i++)
                                                {

        if(pData=dbdata(pDelivery->dbproc, i+1))

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

        pDelivery->o_id[i] = 0;
                                                }
                                        }
                                }
                        }
                if ( !SQLDetectDeadlock(pDelivery->dbproc) )
                        break;
                deadlock_count++;
                Sleep(10 * deadlock_count);
        }
        GetLocalTime(&pDelivery->trans_end);

        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;
}
```

DELISRV.H

```
/*      FILE:           DELISRV.H
 *                              Microsoft TPC-C Kit Ver. 3.00.000
 *                              Audited 08/23/96, By Francois Raab
 *
 *                              Copyright Microsoft, 1996
 *
 *      PURPOSE:  Header file for delivery service executable
 *      Author:         Philip Durr
 *                              philipdu@Microsoft.com
 */

#define AVAILABLE               0
        //queue array element available
#define WRITE_LOCKED            1
        //queue array element is being written to
#define READ_LOCKED             2
        //queue array element is begin read
#define   INUSE                 4
                //queue array element has information stored in it

#define   CTRL_C                3
                //<Ctrl> C, exit key code

#define DEFCLPACKSIZE                                           4096
        //default DB Library SQL Connection pack size

#define   ERR_SUCCESS                                                   0
                //Success, no error.
#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.
#define   ERR_CANNOT_OPEN_PIPE                          1006    //Cannot open
delivery pipe.
#define   ERR_READ_PIPE                                 1007
        //Error reading pipe
#define   ERR_INSUFFICIENT_MEMORY                       1008    //insufficient
memory

typedef struct _DELIVERY_TRANSACTION
{
        SYSTEMTIME          queue;                      //time delivery
transaction queued
        short               w_id;                       //delivery warehouse
        short               o_carrier_id;       //carrier id
} DELIVERY_TRANSACTION;

typedef DELIVERY_TRANSACTION *LPDELIVERY_TRANSACTION;       //pointer to delivery
transaction queue

typedef struct _DELIVERY_PACKET
{
        BOOL                                                bInUse;
        //entry current in use
        OVERLAPPED                                          ov;
                //pipe io overlapped structure
        DELIVERY_TRANSACTION          trans;                        //delivery
transaction information
} DELIVERY_PACKET, *LPDELIVERY_PACKET;

typedef struct _SERRORMSG
{
        int                 iError;                     //error mesage id
        char    szMsg[80];                      //error message
} SERRORMSG;

//delivery transaction structure
typedef struct DELIVERY
{
        short               w_id;                       //warehouse id
        short               o_carrier_id;       //carrier id
        int                             spid;                       //db library
spid
        long                o_id[10];           //returned delivery transaction ids
        DBPROCESS *dbproc;          //db library DBPROCESS pointer
        SYSTEMTIME          queue;                      //delivery transaction
queue time
        SYSTEMTIME          trans_end;                  //delivery transaction
finished time
} DELIVERY;

typedef DELIVERY *LPDELIVERY; //pointer to delivery structure

//function prototypes
void                        main(int argc, char *argv[]);
static    void      cls(void);
static    int               RunDelivery(void);
static    void      QuitStatus(void);
static    void      AnimateWait1(void);
static    void      AnimateWait(void);
static    int               Init(void);
static    void      Restore(void);
```

```
static    void      ErrorMessage(int iError);
static    BOOL      GetParameters(int argc, char *argv[]);
static    void      PrintParameters(void);
static    void      PrintHeader(void);
static    int               ReadRegistrySettings(void);
static    void      CheckKey(void *ptr);
static    void      DeliveryHandler( void *ptr );
static    void      DeliveryThread( void *ptr );
static    int               err_handler(DBPROCESS *dbproc, int severity, int dberr,
int oserr, char *dberrstr, char *oserrstr);
static    int               msg_handler(DBPROCESS *dbproc, DBINT msgno, int
msgstate, int severity, char *msgtext);
static    BOOL      SQLOpenConnection(DBPROCESS **dbproc, char *server, char
*database, char *user, char *password, int *spid);
static    void      WriteLog(LPDELIVERY pDelivery);
static    void      CalculateElapsedTime(int *pElapsed, LPSYSTEMTIME lpBegin,
LPSYSTEMTIME lpEnd);
static    int               SQLDelivery(DELIVERY *pDelivery);
static    BOOL      SQLDetectDeadlock(DBPROCESS *dbproc);
static    BOOL      ReadDeliveryInfo(short *w_id, short *o_carrier_id);
static    BOOL      PostDeliveryInfo(short w_id, short o_carrier_id);
static    int               OpenLogFile(void);
```

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_
```

INSTALL.C

```
/*        FILE:               INSTALL.C
 *                                    Microsoft TPC-C Kit Ver. 3.00.000
 *                                    Audited 08/23/96, By Francois Raab
 *
 *                                    Copyright Microsoft, 1996
 *
 *        PURPOSE:  Automated installation application for TPC-C Web Kit
 *        Author:             Philip Durr
 *                                    philipdu@Microsoft.com
 */

#include <windows.h>
#include <direct.h>
#include  <io.h>
#include <stdlib.h>
#include <stdio.h>
#include <commctrl.h>
#include "install.h"

HICON               hIcon;
HINSTANCE hInst;

DWORD                           versionExeMS;
DWORD                           versionExeLS;
DWORD                           versionDllMS;
DWORD                           versionDllLS;

static    BOOL      bLog;
static    int                   iThreads;
static    int                   iMaxWareHouse;
static    int                   iDelayMs;
static    int                   iDeadlockRetry;
static    int                   iMaxConnections;
static    int                   iPoolThreadsLimit;
static    int                   iThreadTimeout;
static    int                   iListenBackLog;
static    int                   iAcceptExOutstanding;
static    int                   iQSlotts;

static    int                   iMaxPhysicalMemory;           //max physical memory in
MB

BOOL      CALLBACK  UpdatedDlgProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM
lParam);
BOOL      CALLBACK  MainDlgProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);
BOOL      CALLBACK  CopyDlgProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);
static    void                  ReadRegistrySettings(void);
static    void                  WriteRegistrySettings(char *szDllPath);
static    int                             CopyFiles(HWND hDlg, char *szDllPath);
static    BOOL                  GetInstallPath(char *szDllPath);
static    void                  GetVersionInfo(char *szDLLPath, char *szExePath);
static    BOOL                  StartWWWebService(void);
static    BOOL                  StopWWWebService(void);
static    void                  UpdateDialog(HWND hDlg);

int WINAPI WinMain( HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine,
int nCmdShow )
{
          int iRc;

          hInst = hInstance;

          InitCommonControls();

          hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDI_ICON1));

          iRc = DialogBox(hInstance, MAKEINTRESOURCE(IDD_DIALOG1),
GetDesktopWindow(), MainDlgProc);
          if ( iRc )
                                  DialogBoxParam(hInstance, MAKEINTRESOURCE(IDD_DIALOG2),
GetDesktopWindow(), UpdatedDlgProc, (LPARAM)iRc);
          DestroyIcon(hIcon);

          return 0;
}

BOOL CALLBACK UpdatedDlgProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
          switch(uMsg)
          {
```

```
                case WM_INITDIALOG:
                        if ( lParam == 1 )
                                SetDlgItemText(hwnd, IDC_RESULTS, "HTML TPCC
Installation Successfull");
                        else
                                SetDlgItemText(hwnd, IDC_RESULTS, "HTML TPCC
Registry Updated");
                        return TRUE;
                case WM_COMMAND:
                        if ( wParam == IDOK )
                                EndDialog(hwnd, TRUE);
                        break;
                default:
                        break;
        }
        return FALSE;
}

BOOL CALLBACK MainDlgProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
        PAINTSTRUCT             ps;
        MEMORYSTATUS    memoryStatus;
        int                             d;
        int                             rc;
        HWND                    hDlg;
        char                    szTmp[256];
        static char             szDllPath[256];
        static char             szExePath[256];

        switch(uMsg)
        {
                case WM_INITDIALOG:
                        GlobalMemoryStatus(&memoryStatus);
                        iMaxPhysicalMemory  = (memoryStatus.dwTotalPhys/
1048576);

                        if ( GetInstallPath(szDllPath) )
                        {
                                MessageBox(hwnd, "Error internet service
inetsrv is not installed.", NULL, MB_ICONSTOP | MB_OK);
                                EndDialog(hwnd, FALSE);
                                return TRUE;
                        }

                        bLog                                            =
FALSE;
                        iThreads                                = 4;
                        iMaxWareHouse                   = 500;
                        iDelayMs                                = 500;
                        iDeadlockRetry                  = 3;
                        iMaxConnections                 = 25;
                        iPoolThreadsLimit               = iMaxPhysicalMemory * 2;
                        iThreadTimeout                  = 86400;
                        iListenBackLog                  = 15;
                        iAcceptExOutstanding    = 40;

                        ReadRegistrySettings();

                        GetModuleFileName(hInst, szExePath, sizeof(szExePath));
                        GetVersionInfo(szDllPath, szExePath);
                        if ( bLog )
                                CheckDlgButton(hwnd, BN_LOG, 1);
                        wsprintf(szTmp, "Version %d.00.%3.3d", versionExeMS,
versionExeLS);
                        SetDlgItemText(hwnd, IDC_VERSION, szTmp);

                        SetDlgItemText(hwnd, IDC_PATH, szDllPath);
                        SetDlgItemInt(hwnd, ED_MAXWARE, iMaxWareHouse, FALSE);
                        SetDlgItemInt(hwnd, ED_THREADS, iThreads, FALSE);
                        SetDlgItemInt(hwnd, ED_MAXCONNECTION, iMaxConnections,
FALSE);
                        SetDlgItemInt(hwnd, ED_IIS_MAX_THEAD_POOL_LIMIT,
iPoolThreadsLimit, FALSE);
                        SetDlgItemInt(hwnd, ED_IIS_THREAD_TIMEOUT,
iThreadTimeout, FALSE);
                        SetDlgItemInt(hwnd, ED_IIS_LISTEN_BACKLOG,
iListenBackLog, FALSE);
                        SetDlgItemInt(hwnd, ED_WEB_SERVICE_BACKLOG_QUEUE_SIZE,
iAcceptExOutstanding, FALSE);

                        return TRUE;
                case WM_PAINT:
                        if ( IsIconic(hwnd) )
                        {
                                BeginPaint(hwnd, &ps);
                                DrawIcon(ps.hdc, 0, 0, hIcon);
                                EndPaint(hwnd, &ps);
                                return TRUE;
                        }
                        break;
                case WM_COMMAND:
                        if ( wParam == IDOK )
                        {
                                if ( IsDlgButtonChecked(hwnd, BN_LOG) )
                                        bLog = TRUE;
                                else
                                        bLog = FALSE;
                                iThreads  = GetDlgItemInt(hwnd, ED_THREADS,
&d, FALSE);
                                iMaxWareHouse = GetDlgItemInt(hwnd,
ED_MAXWARE, &d, FALSE);
                                iMaxConnections = GetDlgItemInt(hwnd,
ED_MAXCONNECTION, &d, FALSE);


                                iPoolThreadsLimit = GetDlgItemInt(hwnd,
ED_IIS_MAX_THEAD_POOL_LIMIT, &d, FALSE);
                                iThreadTimeout = GetDlgItemInt(hwnd,
ED_IIS_THREAD_TIMEOUT,          &d, FALSE);
                                iListenBackLog = GetDlgItemInt(hwnd,
ED_IIS_LISTEN_BACKLOG, &d, FALSE);
                                iAcceptExOutstanding = GetDlgItemInt(hwnd,
ED_WEB_SERVICE_BACKLOG_QUEUE_SIZE, &d, FALSE);

                                ShowWindow(hwnd, SW_HIDE);
                                hDlg = CreateDialog(hInst,
MAKEINTRESOURCE(IDD_DIALOG3), hwnd, CopyDlgProc);
                                ShowWindow(hDlg, SW_SHOWNA);
                                UpdateDialog(hDlg);
                                rc = CopyFiles(hDlg, szDllPath);
                                if ( !rc )
                                {
                                        ShowWindow(hwnd, SW_SHOWNA);
```

```
                                                DestroyWindow(hDlg);
                                                MessageBox(hwnd, "Error(s) occured
when creating tpcc.dll", NULL, MB_ICONSTOP | MB_OK);
                                                EndDialog(hwnd, 0);
                                                return TRUE;
                                        }
                                        SetDlgItemText(hDlg, IDC_STATUS, "Updating
Registry.");
                                        SendDlgItemMessage(hDlg, IDC_PROGRESS1,
PBM_STEPIT, 0, 0);
                                        UpdateDialog(hDlg);

                                        WriteRegistrySettings(szDllPath);

                                        Sleep(100);

                                        ShowWindow(hwnd, SW_SHOWNA);
                                        DestroyWindow(hDlg);

                                        EndDialog(hwnd, rc);
                                        return TRUE;
                                }
                                if ( wParam == IDCANCEL )
                                {
                                        EndDialog(hwnd, FALSE);
                                        return TRUE;
                                }
                                break;
                        default:
                                break;
                }
                return FALSE;
}

static void ReadRegistrySettings(void)
{
                HKEY      hKey;
                DWORD     size;
                DWORD     type;
                char      szTmp[256];

                if ( RegOpenKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\\Microsoft\\TPCC", 0,
KEY_READ, &hKey) == ERROR_SUCCESS )
                {
                        size = sizeof(szTmp);

                        bLog = FALSE;
                        if ( RegQueryValueEx(hKey, "LOG", 0, &type, szTmp, &size) ==
ERROR_SUCCESS )
                                if ( !stricmp(szTmp, "ON") )
                                        bLog = TRUE;
                        iThreads = 4;
                        size = sizeof(szTmp);
                        if ( RegQueryValueEx(hKey, "NumberOfDeliveryThreads", 0, &type,
szTmp, &size) == ERROR_SUCCESS )
                                iThreads = atoi(szTmp);
                        if ( iThreads == 0 )
                                iThreads = 4;

                        iMaxWareHouse = 500;
                        size = sizeof(szTmp);
                        if ( RegQueryValueEx(hKey, "MaximumWarehouses", 0, &type, szTmp,
&size) == ERROR_SUCCESS )
                                iMaxWareHouse = atoi(szTmp);
                        if ( iMaxWareHouse == 0 )
                                iMaxWareHouse = 500;

                        iDelayMs = 500;
                        size = sizeof(szTmp);
                        if ( RegQueryValueEx(hKey, "BackoffDelay", 0, &type, szTmp,
&size) == ERROR_SUCCESS )
                                iDelayMs = atoi(szTmp);
                        if ( iDelayMs == 0 )
                                iDelayMs = 500;

                        iDeadlockRetry = 3;
                        size = sizeof(szTmp);
                        if ( RegQueryValueEx(hKey, "DeadlockRetry", 0, &type, szTmp,
&size) == ERROR_SUCCESS )
                                iDeadlockRetry = atoi(szTmp);
                                if ( !iDeadlockRetry )
                                        iDeadlockRetry = 3;

                        iMaxConnections = 25;
                        size = sizeof(szTmp);
                        if ( RegQueryValueEx(hKey, "MaxConnections", 0, &type, szTmp,
&size) == ERROR_SUCCESS )
                                iMaxConnections = atoi(szTmp);
                                if ( !iMaxConnections )
                                        iMaxConnections = 25;

                        iQSlotts = 3000;
                        size = sizeof(szTmp);
                        if ( RegQueryValueEx(hKey, "QueueSlotts", 0, &type, szTmp, &size)
== ERROR_SUCCESS )
                                iQSlotts = atoi(szTmp);
                        if ( iQSlotts == 0 )
                                iQSlotts = 3000;

                        RegCloseKey(hKey);

                        if ( RegOpenKeyEx(HKEY_LOCAL_MACHINE,
"SYSTEM\\CurrentControlSet\\Services\\Inetinfo\\Parameters", 0, KEY_READ, &hKey) ==
ERROR_SUCCESS )
                        {
                                iPoolThreadsLimit =  iMaxPhysicalMemory * 2;
                                size = sizeof(iPoolThreadsLimit);
                                if ( RegQueryValueEx(hKey, "PoolThreadsLimit", 0,
&type, (char *)&iPoolThreadsLimit, &size) == ERROR_SUCCESS )
                                if ( !iPoolThreadsLimit )
                                        iPoolThreadsLimit =  iMaxPhysicalMemory * 2;

                                iThreadTimeout = 86400;
                                size = sizeof(iThreadTimeout);
                                if ( RegQueryValueEx(hKey, "ThreadTimeout", 0, &type,
(char *)&iThreadTimeout, &size) == ERROR_SUCCESS )
                                        if ( !iThreadTimeout )
                                                iThreadTimeout = 86400;

                                iListenBackLog = 15;
                                size = sizeof(iListenBackLog);
                                if ( RegQueryValueEx(hKey, "ListenBackLog", 0, &type,
(char *)&iListenBackLog, &size) == ERROR_SUCCESS )
```

```
                                        if ( !iListenBackLog )
                                                iListenBackLog = 15;
                }

                RegCloseKey(hKey);

                if ( RegOpenKeyEx(HKEY_LOCAL_MACHINE,
"SYSTEM\\CurrentControlSet\\Services\\W3SVC\\Parameters", 0, KEY_READ, &hKey) ==
ERROR_SUCCESS )
                {
                        iAcceptExOutstanding = 40;
                        size = sizeof(iAcceptExOutstanding);
                        if ( RegQueryValueEx(hKey, "AcceptExOutstanding", 0,
&type, (char *)&iAcceptExOutstanding, &size) == ERROR_SUCCESS )
                                        if ( !iAcceptExOutstanding )
                                                iAcceptExOutstanding = 40;
                }

                RegCloseKey(hKey);
        }

        return;
}

static void WriteRegistrySettings(char *szDllPath)
{
        HKEY    hKey;
        DWORD   dwDisposition;
        char    szTmp[256];
        char    *ptr;
        int             iRc;

        if ( RegCreateKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\\Microsoft\\TPCC", 0,
NULL, REG_OPTION_NON_VOLATILE, KEY_ALL_ACCESS, NULL, &hKey, &dwDisposition) ==
ERROR_SUCCESS )
        {
                strcpy(szTmp, szDllPath);
                ptr = strstr(szTmp, "tpcc");
                if ( ptr )
                        *ptr = 0;

                RegSetValueEx(hKey, "PATH", 0, REG_SZ, szTmp, strlen(szTmp));

                if ( bLog )
                        RegSetValueEx(hKey, "LOG", 0, REG_SZ, "ON", 2);
                else
                        RegSetValueEx(hKey, "LOG", 0, REG_SZ, "OFF", 3);

                itoa(iThreads, szTmp, 10);
                RegSetValueEx(hKey, "NumberOfDeliveryThreads", 0, REG_SZ, szTmp,
strlen(szTmp));

                itoa(iMaxWareHouse, szTmp, 10);
                RegSetValueEx(hKey, "MaximumWarehouses", 0, REG_SZ, szTmp,
strlen(szTmp));

                itoa(iDelayMs, szTmp, 10);
                RegSetValueEx(hKey, "BackoffDelay", 0, REG_SZ, szTmp,
strlen(szTmp));

                itoa(iDeadlockRetry, szTmp, 10);
                RegSetValueEx(hKey, "DeadlockRetry", 0, REG_SZ, szTmp,
strlen(szTmp));

                itoa(iMaxConnections, szTmp, 10);
                RegSetValueEx(hKey, "MaxConnections", 0, REG_SZ, szTmp,
strlen(szTmp));

                itoa(iQSlotts, szTmp, 10);
                RegSetValueEx(hKey, "QueueSlotts", 0, REG_SZ, szTmp,
strlen(szTmp));

                RegFlushKey(hKey);

                RegCloseKey(hKey);
        }

        if ( (iRc=RegCreateKeyEx(HKEY_LOCAL_MACHINE,
"SYSTEM\\CurrentControlSet\\Services\\Inetinfo\\Parameters", 0, NULL,
REG_OPTION_NON_VOLATILE, KEY_ALL_ACCESS, NULL, &hKey, &dwDisposition)) ==
ERROR_SUCCESS )
        {
                RegSetValueEx(hKey, "PoolThreadsLimit", 0, REG_DWORD, (char
*)&iPoolThreadsLimit, sizeof(iPoolThreadsLimit));
                RegSetValueEx(hKey, "ThreadTimeout", 0, REG_DWORD, (char
*)&iThreadTimeout, sizeof(iThreadTimeout));
                RegSetValueEx(hKey, "ListenBackLog", 0, REG_DWORD, (char
*)&iListenBackLog, sizeof(iListenBackLog));

                RegFlushKey(hKey);
                RegCloseKey(hKey);
        }

        if ( (iRc=RegCreateKeyEx(HKEY_LOCAL_MACHINE,
"SYSTEM\\CurrentControlSet\\Services\\W3SVC\\Parameters", 0, NULL,
REG_OPTION_NON_VOLATILE, KEY_ALL_ACCESS, NULL, &hKey, &dwDisposition)) ==
ERROR_SUCCESS )
        {
                RegSetValueEx(hKey, "AcceptExOutstanding", 0, REG_DWORD, (char
*)&iAcceptExOutstanding, sizeof(iAcceptExOutstanding));

                RegFlushKey(hKey);
                RegCloseKey(hKey);
        }

        return;
}

BOOL CALLBACK CopyDlgProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
        if ( uMsg == WM_INITDIALOG )
        {
                SendDlgItemMessage(hwnd, IDC_PROGRESS1, PBM_SETRANGE, 0,
MAKELPARAM(0, 8));
                SendDlgItemMessage(hwnd, IDC_PROGRESS1, PBM_SETSTEP, (WPARAM)1,
0);
                return TRUE;
        }
        return FALSE;
}

static int CopyFiles(HWND hDlg, char *szDllPath)
{
```

```
        HGLOBAL                 hDLL;
        HGLOBAL                 hExe;
        HRSRC                   hResInfo;
        BYTE                    *pSrc;
        HANDLE                  hFile;
        DWORD                   dwSize;
        DWORD                   d;
        char                    szTmp[256];
        char                    *ptr;
        BOOL                    bSvcRunning;

        SetDlgItemText(hDlg, IDC_STATUS, "Stopping Web Service.");
        SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_STEPIT, 0, 0);
        UpdateDialog(hDlg);

        bSvcRunning = !StopWWWebService();
        SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_STEPIT, 0, 0);
        UpdateDialog(hDlg);

        hResInfo = FindResource(hInst, MAKEINTRESOURCE(IDR_TPCCDLL1), "TPCCDLL");
        SetDlgItemText(hDlg, IDC_STATUS, "Copying Files...");
        SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_STEPIT, 0, 0);
        UpdateDialog(hDlg);

        dwSize = SizeofResource(hInst, hResInfo);
        hDLL = LoadResource(hInst, hResInfo );
        pSrc = (BYTE *)LockResource(hDLL);
        remove(szDllPath);

        if ( !(hFile = CreateFile(szDllPath, GENERIC_WRITE, 0, NULL, CREATE_ALWAYS,
FILE_ATTRIBUTE_NORMAL, NULL)) )
                return 0;

        if ( !WriteFile(hFile, pSrc, dwSize, &d, NULL) )
                return 0;

        CloseHandle(hFile);

        UnlockResource(hDLL);
        FreeResource(hDLL);

        SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_STEPIT, 0, 0);
        UpdateDialog(hDlg);

        hResInfo = FindResource(hInst, MAKEINTRESOURCE(IDR_DELIVERY1), "DELIVERY");

        dwSize = SizeofResource(hInst, hResInfo);
        hExe = LoadResource(hInst, hResInfo );
        pSrc = (BYTE *)LockResource(hExe);

        strcpy(szTmp, szDllPath);
        ptr = strstr(szTmp, "tpcc");
        if ( ptr )
                *ptr = 0;
        strcat(szTmp, "delisrv.exe");

        remove(szTmp);

        if ( !(hFile = CreateFile(szTmp, GENERIC_WRITE, 0, NULL, CREATE_ALWAYS,
FILE_ATTRIBUTE_NORMAL, NULL)) )
                return 0;

        if ( !WriteFile(hFile, pSrc, dwSize, &d, NULL) )
                return 0;

        CloseHandle(hFile);

        UnlockResource(hExe);
        FreeResource(hExe);

        SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_STEPIT, 0, 0);
        UpdateDialog(hDlg);

        //if we stopped service restart it.
        if ( !bSvcRunning )
        {
                SetDlgItemText(hDlg, IDC_STATUS, "Starting Web Service.");
                SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_STEPIT, 0, 0);
                UpdateDialog(hDlg);
                StartWWWebService();
        }

        SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_STEPIT, 0, 0);
        UpdateDialog(hDlg);

        return 1;
}

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

        szDllPath[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(szDllPath, szTmp);
                        bRc = FALSE;
                }
                RegCloseKey(hKey);
        }
        if ( (ptr = strchr(szDllPath, ',')) )
                *ptr = 0;

        len = strlen(szDllPath);
        if ( szDllPath[len-1] != '\\' )
        {
                szDllPath[len] = '\\';
                szDllPath[len+1] = 0;
```

```
        }
        strcat(szDllPath, "tpcc.dll");

        return bRc;
}

static void GetVersionInfo(char *szDLLPath, char *szExePath)
{
        DWORD                                   d;
        DWORD                                   dwSize;
        DWORD                                   dwBytes;
        char                                    *ptr;
        VS_FIXEDFILEINFO    *vs;

        versionDllMS = 0;
        versionDllLS = 0;
        if ( _access(szDLLPath, 00) == 0 )
        {
                dwSize = GetFileVersionInfoSize(szDLLPath, &d);
                if ( dwSize )
                {
                        ptr = (char *)malloc(dwSize);
                        GetFileVersionInfo(szDLLPath, 0, dwSize, ptr);
                        VerQueryValue(ptr, "\\",&vs, &dwBytes);
                        versionDllMS = vs->dwProductVersionMS;
                        versionDllLS = vs->dwProductVersionLS;
                        free(ptr);
                }
        }

        versionExeMS = 0x7FFF;
        versionExeLS = 0x7FFF;
        dwSize = GetFileVersionInfoSize(szExePath, &d);
        if ( dwSize )
        {
                ptr = (char *)malloc(dwSize);
                GetFileVersionInfo(szExePath, 0, dwSize, ptr);
                VerQueryValue(ptr, "\\",&vs, &dwBytes);

                versionExeMS = vs->dwProductVersionMS;
                versionExeLS = vs->dwProductVersionLS;
                free(ptr);
        }
        return;
}

static BOOL StartWWWebService(void)
{
        SC_HANDLE           schSCManager;
        SC_HANDLE           schService;
        SERVICE_STATUS      ssStatus;
        DWORD                           dwOldCheckPoint;

        schSCManager = OpenSCManager(NULL, NULL, SC_MANAGER_ALL_ACCESS);
        schService = OpenService(schSCManager, TEXT("W3SVC"), SERVICE_ALL_ACCESS);
        if (schService == NULL)
                return FALSE;

        if (! StartService(schService, 0, NULL) )
                goto StartWWWebErr;
        //start Service pending, Check the status until the service is running.
        if (! QueryServiceStatus(schService, &ssStatus) )
                goto StartWWWebErr;
        while( ssStatus.dwCurrentState != SERVICE_RUNNING)
        {

                dwOldCheckPoint = ssStatus.dwCheckPoint;
        //Save the current checkpoint.
                Sleep(ssStatus.dwWaitHint);
                        //Wait for the specified interval.
                if ( !QueryServiceStatus(schService, &ssStatus) ) //Check the
status again.
                        break;
                if (dwOldCheckPoint >= ssStatus.dwCheckPoint)
        //Break if the checkpoint has not been incremented.
                        break;
        }

        if (ssStatus.dwCurrentState == SERVICE_RUNNING)
                goto StartWWWebErr;

        CloseServiceHandle(schService);
        return TRUE;

StartWWWebErr:
        CloseServiceHandle(schService);
        return FALSE;
}

static BOOL StopWWWebService(void)
{
        SC_HANDLE           schSCManager;
        SC_HANDLE           schService;
        SERVICE_STATUS      ssStatus;
        DWORD                           dwOldCheckPoint;

        schSCManager = OpenSCManager(NULL, NULL, SC_MANAGER_ALL_ACCESS);
        schService = OpenService(schSCManager, TEXT("W3SVC"), SERVICE_ALL_ACCESS);
        if (schService == NULL)
                return FALSE;

        if (! QueryServiceStatus(schService, &ssStatus) )
                goto StopWWWebErr;

        if ( !ControlService(schService, SERVICE_CONTROL_STOP, &ssStatus) )
                goto StopWWWebErr;
        //start Service pending, Check the status until the service is running.
        if (! QueryServiceStatus(schService, &ssStatus) )
                goto StopWWWebErr;
        while( ssStatus.dwCurrentState == SERVICE_RUNNING)
        {

                dwOldCheckPoint = ssStatus.dwCheckPoint;
        //Save the current checkpoint.
                Sleep(ssStatus.dwWaitHint);
                        //Wait for the specified interval.
                if ( !QueryServiceStatus(schService, &ssStatus) ) //Check the
status again.
                        break;
                if (dwOldCheckPoint >= ssStatus.dwCheckPoint)
        //Break if the checkpoint has not been incremented.
                        break;
        }
```

```
        if (ssStatus.dwCurrentState == SERVICE_RUNNING)
                goto StopWWWebErr;

        CloseServiceHandle(schService);
        return TRUE;

StopWWWebErr:
        CloseServiceHandle(schService);
        return FALSE;
}

static void UpdateDialog(HWND hDlg)
{
        MSG msg;

        UpdateWindow(hDlg);
        while( PeekMessage(&msg, hDlg, 0, 0, PM_REMOVE) )
        {
                TranslateMessage(&msg);
                DispatchMessage(&msg);
        }
        Sleep(250);
        return;
}
```

INSTALL.H

```
//{{NO_DEPENDENCIES}}
// Microsoft Developer Studio generated include file.
// Used by install.rc
//
#define IDD_DIALOG1                             101
#define IDI_ICON1                               102
#define IDR_TPCCDLL1                            103
#define IDD_DIALOG2                             104
#define IDI_ICON2                               105
#define IDR_DELIVERY1                           109
#define IDD_DIALOG3                             110
#define BN_LOG                                  1001
#define ED_KEEP                                 1002
#define ED_THREADS                              1003
#define ED_THREADS2                             1004
#define ED_MAXWARE                              1006
#define IDC_PATH                                1007
#define IDC_VERSION                             1009
#define IDC_RESULTS                             1010
#define IDC_PROGRESS1                           1011
#define IDC_STATUS                              1012
#define IDC_BUTTON1                             1013
#define ED_MAXCONNECTION                        1014
#define ED_IIS_MAX_THEAD_POOL_LIMIT             1015
#define ED_WEB_SERVICE_BACKLOG_QUEUE_SIZE       1017
#define ED_IIS_THREAD_TIMEOUT                   1018
#define ED_IIS_LISTEN_BACKLOG                   1019

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE                111
#define _APS_NEXT_COMMAND_VALUE                 40001
#define _APS_NEXT_CONTROL_VALUE                 1015
#define _APS_NEXT_SYMED_VALUE                   101
#endif
#endif
```

INSTALL.RC

```
//Microsoft Developer Studio generated resource script.
//
#include "install.h"

#define APSTUDIO_READONLY_SYMBOLS
////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "afxres.h"

////////////////////////////////////////////////////////////////////////////
#undef APSTUDIO_READONLY_SYMBOLS

////////////////////////////////////////////////////////////////////////////
// English (U.S.) resources

#if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_ENU)
#ifdef _WIN32
LANGUAGE LANG_ENGLISH, SUBLANG_ENGLISH_US
#pragma code_page(1252)
#endif //_WIN32

////////////////////////////////////////////////////////////////////////////
//
// Dialog
//

IDD_DIALOG1 DIALOGEX 0, 0, 234, 174
STYLE DS_MODALFRAME | DS_CENTER | WS_MINIMIZEBOX | WS_POPUP | WS_CAPTION |
    WS_SYSMENU
CAPTION "TPCC Web Client Installation Utility"
FONT 8, "MS Sans Serif"
BEGIN
    EDITTEXT        ED_MAXWARE,199,37,21,12,ES_NUMBER,WS_EX_RTLREADING
    CONTROL         "",BN_LOG,"Button",BS_AUTOCHECKBOX | BS_LEFTTEXT |
                    BS_LEFT | BS_VCENTER | WS_TABSTOP,205,51,15,13,
                    WS_EX_STATICEDGE
    EDITTEXT        ED_THREADS,205,66,15,12,ES_NUMBER,WS_EX_RTLREADING
    EDITTEXT        ED_MAXCONNECTION,186,80,34,12,ES_NUMBER,WS_EX_RTLREADING
    EDITTEXT        ED_IIS_MAX_THEAD_POOL_LIMIT,186,94,34,12,ES_NUMBER,
                    WS_EX_RTLREADING
    EDITTEXT        ED_WEB_SERVICE_BACKLOG_QUEUE_SIZE,186,108,34,12,
                    ES_NUMBER,WS_EX_RTLREADING
    EDITTEXT        ED_IIS_THREAD_TIMEOUT,186,122,34,12,ES_NUMBER,
                    WS_EX_RTLREADING
    EDITTEXT        ED_IIS_LISTEN_BACKLOG,186,136,34,12,ES_NUMBER,
                    WS_EX_RTLREADING
    DEFPUSHBUTTON   "OK",IDOK,59,153,50,14
    PUSHBUTTON      "Cancel",IDCANCEL,125,153,50,14
```

```
    EDITTEXT        IDC_PATH,42,22,178,13,ES_AUTOHSCROLL | ES_READONLY
    LTEXT           "Max Number of Warehouses:",IDC_STATIC,42,37,115,12,
                    SS_SUNKEN
    LTEXT           "Write HTML To Log file:",IDC_STATIC,42,51,115,12,
                    SS_SUNKEN
    LTEXT           "Number of Delivery Threads:",IDC_STATIC,42,66,115,12,
                    SS_SUNKEN
    LTEXT           "Max Number of Connections:",IDC_STATIC,41,80,115,12,
                    SS_SUNKEN
    CTEXT           "Version 1.00.001",IDC_VERSION,42,6,178,14,SS_SUNKEN |
                    WS_BORDER,WS_EX_CLIENTEDGE
    ICON            IDI_ICON1,IDC_STATIC,9,6,21,20,0,WS_EX_CLIENTEDGE
    LTEXT           "IIS Max Thread Pool Limit:",IDC_STATIC,41,94,115,12,
                    SS_SUNKEN
    LTEXT           "Web Service Backlog Queue Size:",IDC_STATIC,41,108,115,
                    12,SS_SUNKEN
    LTEXT           "IIS Thread Timeout:",IDC_STATIC,41,122,115,12,SS_SUNKEN
    LTEXT           "IIS Listen Backlog:",IDC_STATIC,41,136,115,12,SS_SUNKEN
END

IDD_DIALOG2 DIALOGEX 0, 0, 117, 62
STYLE DS_SETFOREGROUND | DS_3DLOOK | DS_CENTER | WS_POPUP | WS_BORDER
EXSTYLE WS_EX_STATICEDGE
FONT 12, "MS Sans Serif", 0, 0, 0x1
BEGIN
    DEFPUSHBUTTON   "OK",IDOK,33,45,50,9
    CTEXT           "HTML TPCC Installation Successfull",IDC_RESULTS,7,22,
                    102,18,0,WS_EX_CLIENTEDGE
    ICON            IDI_ICON2,IDC_STATIC,50,7,18,20,SS_REALSIZEIMAGE,
                    WS_EX_TRANSPARENT
END

IDD_DIALOG3 DIALOG DISCARDABLE  0, 0, 91, 40
STYLE DS_SYSMODAL | DS_MODALFRAME | DS_3DLOOK | DS_CENTER | WS_CAPTION
CAPTION "Installing TPCC Web Service"
FONT 12, "Arial Black"
BEGIN
    CONTROL         "Progress1",IDC_PROGRESS1,"msctls_progress32",WS_BORDER,
                    7,20,77,13
    CTEXT           "Static",IDC_STATUS,7,7,77,12,SS_SUNKEN
END


/////////////////////////////////////////////////////////////////////////////
//
// DESIGNINFO
//

#ifdef APSTUDIO_INVOKED
GUIDELINES DESIGNINFO DISCARDABLE
BEGIN
    IDD_DIALOG1, DIALOG
    BEGIN
        LEFTMARGIN, 9
        RIGHTMARGIN, 220
        TOPMARGIN, 6
        BOTTOMMARGIN, 167
    END

    IDD_DIALOG2, DIALOG
    BEGIN
        LEFTMARGIN, 7
        RIGHTMARGIN, 109
        TOPMARGIN, 7
        BOTTOMMARGIN, 54
    END

    IDD_DIALOG3, DIALOG
    BEGIN
        LEFTMARGIN, 7
        RIGHTMARGIN, 84
        TOPMARGIN, 7
        BOTTOMMARGIN, 33
    END
END
#endif    // APSTUDIO_INVOKED


#ifdef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////////////////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE DISCARDABLE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE DISCARDABLE
BEGIN
    "#include ""afxres.h""\r\n"
    "\0"
END

3 TEXTINCLUDE DISCARDABLE
BEGIN
    "\r\n"
    "\0"
END

#endif    // APSTUDIO_INVOKED


/////////////////////////////////////////////////////////////////////////////
//
// Icon
//

// Icon with lowest ID value placed first to ensure application icon
// remains consistent on all systems.
IDI_ICON1               ICON    DISCARDABLE     "icon1.ico"
IDI_ICON2               ICON    DISCARDABLE     "icon2.ico"

/////////////////////////////////////////////////////////////////////////////
//
// TPCCDLL
//

IDR_TPCCDLL1            TPCCDLL DISCARDABLE     "tpcc.dll"

#ifndef _MAC
/////////////////////////////////////////////////////////////////////////////
//
```

```
// Version
//

VS_VERSION_INFO VERSIONINFO
 FILEVERSION 0,3,0,2
 PRODUCTVERSION 0,3,0,2
 FILEFLAGSMASK 0x3fL
#ifdef _DEBUG
 FILEFLAGS 0x1L
#else
 FILEFLAGS 0x0L
#endif
 FILEOS 0x40004L
 FILETYPE 0x1L
 FILESUBTYPE 0x0L
BEGIN
    BLOCK "StringFileInfo"
    BEGIN
        BLOCK "040904b0"
        BEGIN
            VALUE "CompanyName", "Microsoft\0"
            VALUE "FileDescription", "install\0"
            VALUE "FileVersion", "0, 3, 0, 2\0"
            VALUE "InternalName", "install\0"
            VALUE "LegalCopyright", "Copyright © 1996\0"
            VALUE "OriginalFilename", "install.exe\0"
            VALUE "ProductName", "Microsoft install\0"
            VALUE "ProductVersion", "0, 3, 0, 2\0"
        END
    END
    BLOCK "VarFileInfo"
    BEGIN
        VALUE "Translation", 0x409, 1200
    END
END

#endif    // !_MAC


/////////////////////////////////////////////////////////////////////////////
//
// DELIVERY
//

IDR_DELIVERY1           DELIVERY DISCARDABLE    "delisrv.exe"
#endif    // English (U.S.) resources
/////////////////////////////////////////////////////////////////////////////



#ifndef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 3 resource.
//


/////////////////////////////////////////////////////////////////////////////
#endif    // not APSTUDIO_INVOKED
```

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
```

Makefile

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

!IF "$(SQL_LOC)" == ""
SQL_LOC=C:\MSSQL\DBLIB
!MESSAGE No SQL_LOC specified.  Defaulting to C:\MSSQL\DBLIB
!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
OUTDIR          = .\Bin
DBLIB           = $(SQL_LOC)
DBLIBINC        = $(DBLIB)\INCLUDE
DBLIBDIR        = $(DBLIB)\LIB

!IF "$(CFG)" != "Debug"
LDEBUG       =
CDEBUG       =
LDEBUG_RG    =
```

```
CDEBUG_RG    =
DEBUG        =
FLAGS        = /D "WIN32" /D "_WINDOWS"
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 "_DEBUG" /D "WIN32" /D "_WINDOWS"
OPT          = /Od
!ENDIF

LINK32_LIBS     = user32.lib msacm32.lib advapi32.lib $(DBLIBDIR)\ntwdblib.lib
LINK32_OBJS     = "$(OBJDIR)\tpcc.obj" "$(OBJDIR)\tpcc.res"
LINK32_DEF      = "$(SRCDIR)\tpcc.def"
LINK32_FLAGS    = /nologo /subsystem:windows /dll /incremental:no $(LDEBUG)
/def:"$(LINK32_DEF)" /out:"$(OBJDIR)\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)\install.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) $(FLAGS)
/Fo$(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

$(OBJDIR)\tpcc.dll:     $(LINK32_OBJS) $(LINK32_DEF)
          link.exe   $(LINK32_FLAGS) $(LINK32_OBJS) $(LINK32_LIBS)

$(OBJDIR)\delisrv.exe:  $(SRCDIR)\delisrv.c $(SRCDIR)\delisrv.h
          cl.exe /nologo /MT /W3 $(CDEBUG) $(OPT) /I $(DBLIBINC) $(FLAGS)
/Fo$(OBJDIR)\delisrv.obj $(SRCDIR)\delisrv.c /link /out:$(OBJDIR)\delisrv.exe
$(DBLIBDIR)\ntwdblib.lib msacm32.lib advapi32.lib

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

$(OBJDIR)\install.obj:  $(SRCDIR)\install.c $(OBJDIR)\tpcc.dll $(OBJDIR)\delisrv.exe
$(OBJDIR)\install.res
          cl -W3 $(CDEBUG_RG) /Fo$(OBJDIR)\install.obj /c $(SRCDIR)\install.c

$(OUTDIR)\install.exe:     $(OBJDIR)\install.obj $(OBJDIR)\install.res
          link.exe @<<
  $(LINK32_FLAGS_RG) $(LINK32_OBJS_RG) $(LINK32_LIBS_RG)
<<
```

TPCC.C

```
/*        FILE:               TPCC.C
 *                                    Microsoft TPC-C Kit Ver. 3.00.000
 *                                    Audited 08/23/96    By Francois Raab
 *
 *                                    Copyright Microsoft, 1996
 *
 *        PURPOSE:  Main module for TPCC.DLL which is an ISAPI service dll.
 *        Author:             Philip Durr
 *                                    philipdu@Microsoft.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 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.

char      szServer[32]        = { 0 };            //global variables used with this
DLL
char      szUser[32]                  = { 0 };
char      szPassword[32]      = { 0 };
char      szDatabase[32]      = "tpcc";
BOOL      bLog                        = FALSE;
int               iThreads            = 5;
int               iMaxWareHouses      = 500;
int               iDelayMs            = 100;
short     iDeadlockRetry      = (short)3;
short     iMaxConnections = (short)25;

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


//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=\"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    LPTSTR                                          lpszPipeName        =
TEXT("\\\\.\\pipe\\DELISRV");
static    HANDLE                                          hDeliveryWrite      =
INVALID_HANDLE_VALUE;
static    HANDLE                                          hPipe
          = INVALID_HANDLE_VALUE;

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(&ErrorLogCriticalSection);

                              (*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;

                              // create a security descriptor that allows anyone to
access the pipe...
                              pSD = (PSECURITY_DESCRIPTOR)malloc(
SECURITY_DESCRIPTOR_MIN_LENGTH );
                              if ( pSD == NULL )
```

```
                        {
                                MessageBox(NULL, "Error
malloc(SECURITY_DESCRIPTOR_MIN_LENGTH)", "Init", MB_OK | MB_ICONSTOP);
                                return FALSE;
                        }
                        if ( !InitializeSecurityDescriptor(pSD,
SECURITY_DESCRIPTOR_REVISION) )
                        {
                                MessageBox(NULL, "Error
InitializeSecurityDescriptor()", "Init", MB_OK | MB_ICONSTOP);
                                return FALSE;
                        }
                        // add a NULL disc. ACL to the security descriptor.
                        if ( !SetSecurityDescriptorDacl(pSD, TRUE, (PACL) NULL,
FALSE) )
                        {
                                MessageBox(NULL, "Error
SetSecurityDescriptorDacl().", "Init", MB_OK | MB_ICONSTOP);
                                return FALSE;
                        }

                        sa.nLength                                      =
sizeof(sa);
                        sa.lpSecurityDescriptor       = pSD;
                        sa.bInheritHandle             = TRUE;

                        // open delivery named pipe...
                        hPipe = CreateNamedPipe(lpszPipeName,
FILE_FLAG_OVERLAPPED | PIPE_ACCESS_DUPLEX,
                                PIPE_TYPE_BYTE | PIPE_READMODE_BYTE |
PIPE_NOWAIT,
                                1, 65535, 65535, 250, &sa);

                        if ( hPipe == INVALID_HANDLE_VALUE )
                        {
                                MessageBox(NULL, "Error CreateNamedPipe().",
"Init", MB_OK | MB_ICONSTOP);
                                free(pSD);
                                return FALSE;
                        }

                        bTpccExit = FALSE;

                        if ( _beginthread( DeliveryDisconnect, 0, NULL ) == -1
)
                        {
                                MessageBox(NULL, "Error _beginthread()",
"Init", MB_OK | MB_ICONSTOP);
                                return FALSE;
                        }

                        #ifdef USE_ODBC
                                if ( SQLAllocEnv(&henv) == SQL_ERROR )
                                {
                                        MessageBox(NULL, "Error
SQLAllocEnv()", "Init", MB_OK | MB_ICONSTOP);
                                        return FALSE;
                                }
                        #else
                                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);
                        #endif
                        break;
                case DLL_THREAD_ATTACH:
                        break;
                case DLL_THREAD_DETACH:
                        break;
                case DLL_PROCESS_DETACH:
                        if ( pSD )
                                free( pSD );

                        bTpccExit = TRUE;

                        if ( hPipe )
                                DisconnectNamedPipe(hPipe);

                        if (hPipe != INVALID_HANDLE_VALUE )
                                CloseHandle(hPipe);

                        (*Term.Restore)();

                        #ifdef USE_ODBC
                                SQLFreeEnv(henv);
                        Else
                                dbexit();
                        #endif

                        DeleteCriticalSection(&CriticalSection);
                        DeleteCriticalSection(&ErrorLogCriticalSection);

                        break;
        }
        return TRUE;
}

/* FUNCTION: void DeliveryDisconnect(void *ptr)
 *
 * PURPOSE:         This function handles disconnecting the server side of the
delivery pipe when the
 *                          delivery handler application shuts down.
 *
 * ARGUMENTS:       void      *ptr      void pointer normally NULL passed from thread
handler.
 *
 * RETURNS:                   None
 *
 * COMMENTS:        This function runs as thread which allows the client pipe to
disconnect by
 *                          sending a byte back though the pipe to the
server i.e. this DLL.
```

```
 */

static void DeliveryDisconnect(void *ptr)
{
          int                                                    l, d;
          SECURITY_ATTRIBUTES           sa;
          PSECURITY_DESCRIPTOR          pSD;

          // create a security descriptor that allows anyone to access the pipe...
          pSD = (PSECURITY_DESCRIPTOR)malloc( SECURITY_DESCRIPTOR_MIN_LENGTH );
          InitializeSecurityDescriptor(pSD, SECURITY_DESCRIPTOR_REVISION);
          SetSecurityDescriptorDacl(pSD, TRUE, (PACL) NULL, FALSE);
          sa.nLength                                    = sizeof(sa);
          sa.lpSecurityDescriptor       = pSD;
          sa.bInheritHandle             = TRUE;

          while( !bTpccExit )
          {
                    if ( hPipe && ReadFile(hPipe, &l, 1, &d, NULL) )
                    {
                              DisconnectNamedPipe(hPipe);
                              CloseHandle(hPipe);
                              // open delivery named pipe...
                              hPipe = CreateNamedPipe(lpszPipeName,
FILE_FLAG_OVERLAPPED | PIPE_ACCESS_DUPLEX,
                                        PIPE_TYPE_BYTE | PIPE_READMODE_BYTE |
PIPE_NOWAIT,
                                        1, 65535, 65535, 250, &sa);
                    }
                    Sleep( 2000 );      //check for delivery application exit once
every 2 seconds.
          }

          free(pSD);

          return;
}

/* 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, 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) );

         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;

          if ( !GetKeyValue(pQueryString, "Server", szServer, sizeof(szServer)) )
          {
                    ErrorMessage(pECB, ERR_NO_SERVER_SPECIFIED, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
                    return;
          }

          if ( !GetKeyValue(pQueryString, "User", szUser, sizeof(szUser)) )
                    strcpy(szUser, "sa");

          if ( !GetKeyValue(pQueryString, "Psw", szPassword, sizeof(szPassword)) )
                    strcpy(szPassword, "");

          if ( !GetKeyValue(pQueryString, "Db", szDatabase, sizeof(szDatabase)) )
                    strcpy(szDatabase, "tpcc");

          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 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);

          (*pECB->ServerSupportFunction)(pECB->ConnID, HSE_REQ_DONE_WITH_SESSION,
NULL, 0, 0);

/*        (*pECB->ServerSupportFunction)(pECB->ConnID, HSE_REQ_SEND_RESPONSE_HEADER,
szHeader, &iSize, (LPDWORD)szHeader1); */

          (*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;
}

/* 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;

         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;

        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\">", 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\">", 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\">", 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;
        }
        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;
}
```

```
#ifndef USE_ODBC
        /* 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)))
                {
                        ErrorMessage(gpECB, -1, ERR_TYPE_DBLIB, "DBPROC is
invalid.", iTermId, iSyncId);
                        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 )
                        return INT_CANCEL;

                if ( oserr != DBNOERR )
                {
                        ErrorMessage(pECB, oserr, ERR_TYPE_DBLIB, oserrstr,
iTermId, iSyncId);

                        if ( pEcbInfo )
                                pEcbInfo->bFailed = TRUE;

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

                        EnterCriticalSection(&ErrorLogCriticalSection);

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

                        fprintf(fp, "%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);
                        LeaveCriticalSection(&ErrorLogCriticalSection);

                        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
                                ErrorMessage(pECB, -1, ERR_TYPE_SQL, "Error,
dbgetuserdata returned NULL.", iTermId, iSyncId);
                        return INT_CONTINUE;
                }
                if ( pEcbInfo && pEcbInfo->bFailed )
                        return INT_CANCEL;

                if (msgno == 0)
                        return INT_CONTINUE;
                else
                {
                        ErrorMessage(pECB, msgno, ERR_TYPE_SQL, msgtext,
iTermId, iSyncId);


                        if ( pEcbInfo )
                                pEcbInfo->bFailed = TRUE;

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

                        EnterCriticalSection(&ErrorLogCriticalSection);
                        sprintf(szTmp, "Error: SQLSVR(%d): %s", msgno,
msgtext);
                                fprintf(fp, "%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);
                                LeaveCriticalSection(&ErrorLogCriticalSection);

                                fclose(fp);
                        }

                        return INT_CANCEL;
                }
#endif

/* 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;
        }

        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];

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

          Term.pClientData[iTermId].dbproc = NULL;

          if ( SQLOpenConnection(pECB, iTermId, iSyncId,
&Term.pClientData[iTermId].dbproc, szServer, szDatabase, szUser, szPassword, szApp,
&Term.pClientData[iTermId].spid) )
          {
                   ErrorMessage(pECB, ERR_SQL_OPEN_CONNECTION, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
                   return TRUE;
          }
          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].dbproc)) )
                   {
                            pEcbInfo->iTermId = -1;
                            pEcbInfo->iSyncId = -1;
                            free(pEcbInfo);     //free up user info
                   }
                   return SQLCloseConnection(pECB,
Term.pClientData[iTermId].dbproc);
          }

          UNUSEDPARAM(iSyncId);
}

/* 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);   */
                DBSETLUSER(login, user);
                DBSETLPWD(login,  password);
                DBSETLHOST(login, app);

                DBSETLPACKET(login, (unsigned short)DEFCLPACKSIZE);

                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: 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 FALSE;
                }
        }

        // If we reached here, it means we quit after MAX_RETRY deadlocks
        strcpy(pStockLevel->execution_status, "Hit deadlock max. ");
        return TRUE;
}

/* 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);

                        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))

          pNewOrder->Ol[i].ol_i_price = (*(DBFLT8 *) pData);


          if(pData=dbdata(dbproc, 5))


          pNewOrder->Ol[i].ol_amount = (*(DBFLT8 *) pData);



          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))


          pNewOrder->w_tax = (*(DBFLT8 *) pData);




          if(pData=dbdata(dbproc, 2))

          pNewOrder->d_tax = (*(DBFLT8 *) pData);




          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))

          pNewOrder->c_discount = (*(DBFLT8 *) pData);




          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.");
                                return FALSE;
                        }
                }
        }

        // If we reached here, it means we quit after MAX_RETRY deadlocks
        strcpy(pNewOrder->execution_status,"Hit deadlock max.  ");

        return -1;          //        "deadlock max retry reached!"
}

/* 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];
        BOOL                by_name;
        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;

        if (pPayment->c_id == 0)
                by_name = TRUE;
        else
                by_name = FALSE;

        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))
                                                        pPayment-
>c_credit_lim = (*(DBFLT8 *) pData);
                                                if(pData=dbdata(dbproc,
25))
                                                        pPayment-
>c_discount = (*(DBFLT8 *) pData);
                                                if(pData=dbdata(dbproc,
26))
                                                        pPayment-
>c_balance = (*(DBFLT8 *) pData);
                                                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.");
                                        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.  ");
                return -1; //"deadlock max retry reached!"
}


/* 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];
        BOOL                by_name;
        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;

        if (pOrderStatus->c_id == 0)
                by_name = TRUE;
        else
                by_name = FALSE;

        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))
                                                        pOrderStatus-
>OlOrderStatusData[i].ol_amount = (*(DBFLT8 *) pData);
                                                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))
                                                                pOrderStatus-
>c_balance = (*(DBFLT8 *) pData);
                                                        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.");
                                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.  ");
                return -1; //"deadlock max retry reached!"
}

/* 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;
}

/* 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\">");
        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\">");
          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, 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 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\">");
          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>"
```

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

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

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

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

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

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

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

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

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

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

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

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

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

Total:<BR><HR>"

NAME=\"CMD\" VALUE=\"Process\">"

NAME=\"CMD\" VALUE=\"Menu\">"



        }
        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_entry_d.day,

        Term.pClientData[iTermId].newOrderData.o_entry_d.month,
```

```
" <INPUT NAME=\"SP02*\"
      <INPUT

" <INPUT NAME=\"SP03*\"
      <INPUT

" <INPUT NAME=\"SP04*\"
      <INPUT

" <INPUT NAME=\"SP05*\"
      <INPUT

" <INPUT NAME=\"SP06*\"
      <INPUT

" <INPUT NAME=\"SP07*\"
      <INPUT

" <INPUT NAME=\"SP08*\"
      <INPUT

" <INPUT NAME=\"SP09*\"
      <INPUT

" <INPUT NAME=\"SP10*\"
      <INPUT

" <INPUT NAME=\"SP11*\"
      <INPUT

" <INPUT NAME=\"SP12*\"
      <INPUT

" <INPUT NAME=\"SP13*\"
      <INPUT

" <INPUT NAME=\"SP14*\"
      <INPUT

"Execution Status:

"<INPUT TYPE=\"submit\"

"<INPUT TYPE=\"submit\"

"</FORM>"
"</HTML>" );
```

```
        Term.pClientData[iTermId].newOrderData.o_entry_d.year,

        Term.pClientData[iTermId].newOrderData.o_entry_d.hour,

        Term.pClientData[iTermId].newOrderData.o_entry_d.minute,

        Term.pClientData[iTermId].newOrderData.o_entry_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);
                        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_cnt,

        Term.pClientData[iTermId].newOrderData.w_tax*100,

        Term.pClientData[iTermId].newOrderData.d_tax*100);

                        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.execution_status,

		Term.pClientData[iTermId].newOrderData.total_amount);
			}
			else
			{
				sprintf(szForm+strlen(szForm), "Execution Status:
%24.24s		Total:",

		Term.pClientData[iTermId].newOrderData.execution_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\">");
		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);

                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_amount,
                              Term.pClientData[iTermId].paymentData.c_balance);

                    sprintf(szForm+strlen(szForm), "Credit Limit:   $%13.2f<BR><BR>",
                              Term.pClientData[iTermId].paymentData.c_credit_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_id =
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\">");
          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_id,
c_first, c_middle, c_last);

                    sprintf(szForm+strlen(szForm), "Cust-Balance: $%9.2f<BR><BR>",
                              Term.pClientData[iTermId].orderStatusData.c_balance);

                    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_id,

          Term.pClientData[iTermId].orderStatusData.o_entry_d.day,

          Term.pClientData[iTermId].orderStatusData.o_entry_d.month,

          Term.pClientData[iTermId].orderStatusData.o_entry_d.year,

          Term.pClientData[iTermId].orderStatusData.o_entry_d.hour,

          Term.pClientData[iTermId].orderStatusData.o_entry_d.minute,

          Term.pClientData[iTermId].orderStatusData.o_entry_d.second,

          Term.pClientData[iTermId].orderStatusData.o_carrier_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.OlOrderStatusData[i].ol_supply_w_
id,

          Term.pClientData[iTermId].orderStatusData.OlOrderStatusData[i].ol_i_id,

          Term.pClientData[iTermId].orderStatusData.OlOrderStatusData[i].ol_quantity,

          Term.pClientData[iTermId].orderStatusData.OlOrderStatusData[i].ol_amount,

          Term.pClientData[iTermId].orderStatusData.OlOrderStatusData[i].ol_delivery_
d.day,

          Term.pClientData[iTermId].orderStatusData.OlOrderStatusData[i].ol_delivery_
d.month,

          Term.pClientData[iTermId].orderStatusData.OlOrderStatusData[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)
 *
 * 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 *MakeDeliveryForm(int iTermId, int iSyncId, BOOL bInput)
{
          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\">");
        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_carrier_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.execution_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
 *
 */

static void ProcessNewOrderForm(EXTENSION_CONTROL_BLOCK     *pECB, int iTermId, int
iSyncId)
{
        int                         iRc;
        int                         iError;
        PECBINFO  pEcbInfo;
```

```
        memset(&Term.pClientData[iTermId].newOrderData, 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;
        }

        iRc = SQLNewOrder(pECB, iTermId, iSyncId, Term.pClientData[iTermId].dbproc,
&Term.pClientData[iTermId].newOrderData, iDeadlockRetry);

        if ( (pEcbInfo = (PECBINFO)dbgetuserdata(Term.pClientData[iTermId].dbproc))
)
        {
                if ( pEcbInfo->bFailed )
                        return;
        }

        if ( iRc < 0 )
                ErrorMessage(pECB, ERR_NEW_ORDER_NOT_PROCESSED, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
        else
                WriteZString(pECB, MakeNewOrderForm(iTermId, iSyncId, 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
 *
 */

static void ProcessPaymentForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId, int
iSyncId)
{
        int                         iRc;
        int                         iError;
        PECBINFO  pEcbInfo;

        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;
        }

        iRc = SQLPayment(pECB, iTermId, iSyncId, Term.pClientData[iTermId].dbproc,
&Term.pClientData[iTermId].paymentData, iDeadlockRetry);

        if ( (pEcbInfo = (PECBINFO)dbgetuserdata(Term.pClientData[iTermId].dbproc))
)
        {
                if ( pEcbInfo->bFailed )
                        return;
        }

        if ( iRc == 0 )
                ErrorMessage(pECB, ERR_PAYMENT_INVALID_CUSTOMER, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
        else if ( iRc < 0 )
                ErrorMessage(pECB, ERR_PAYMENT_NOT_PROCESSED, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
        else
                WriteZString(pECB, MakePaymentForm(iTermId, iSyncId, FALSE) );

        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
 *
 */
```

```
static void ProcessOrderStatusForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId, int
iSyncId)
{
        int                             iRc;
        int                             iError;
        PECBINFO  pEcbInfo;

        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;
        }

        iRc = SQLOrderStatus(pECB, iTermId, iSyncId,
Term.pClientData[iTermId].dbproc, &Term.pClientData[iTermId].orderStatusData,
iDeadlockRetry);
        if ( (pEcbInfo = (PECBINFO)dbgetuserdata(Term.pClientData[iTermId].dbproc))
)
        {
                if ( pEcbInfo->bFailed )
                        return;
        }

        if ( iRc == 0 )
                ErrorMessage(pECB, ERR_NOSUCH_CUSTOMER, ERR_TYPE_WEBDLL, NULL,
iTermId, iSyncId);
        else if ( iRc < 0 )
                ErrorMessage(pECB, ERR_ORDER_STATUS_NOT_PROCESSED,
ERR_TYPE_WEBDLL, NULL, iTermId, iSyncId);
        else
                WriteZString(pECB, MakeOrderStatusForm(iTermId, iSyncId, FALSE)
);

        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
 *
 */

static void ProcessDeliveryForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId, int
iSyncId)
{
        char      szTmp[26];

        memset(&Term.pClientData[iTermId].deliveryData, 0, sizeof(DELIVERY_DATA));

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

        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_carrier_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;
        }

        //post delivery info
        if ( PostDeliveryInfo(Term.pClientData[iTermId].deliveryData.w_id,
Term.pClientData[iTermId].deliveryData.o_carrier_id) )
                strcpy(Term.pClientData[iTermId].deliveryData.execution_status,
"Delivery Post Failed");
        else
                strcpy(Term.pClientData[iTermId].deliveryData.execution_status,
"Delivery has been queued.");

        WriteZString(pECB, MakeDeliveryForm(iTermId, iSyncId, FALSE) );

        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
 *
 */

static void ProcessStockLevelForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId, int
iSyncId)
{
          char                szTmp[26];
          BOOL                bRc;
          PECBINFO  pEcbInfo;

          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;
          }

          bRc = SQLStockLevel(pECB, iTermId, iSyncId,
Term.pClientData[iTermId].dbproc, &Term.pClientData[iTermId].stockLevelData,
iDeadlockRetry);

          if ( (pEcbInfo = (PECBINFO)dbgetuserdata(Term.pClientData[iTermId].dbproc))
)
          {
                    if ( pEcbInfo->bFailed )
                              return;
          }

          if ( bRc )
                    ErrorMessage(pECB, ERR_STOCKLEVEL_NOT_PROCESSED, ERR_TYPE_WEBDLL,
NULL, iTermId, iSyncId);
          else
                    WriteZString(pECB, MakeStockLevelForm(iTermId, iSyncId, FALSE) );

          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);
          pNewOrderData->o_all_local=1;

          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;
                        if ( !IsNumeric(szTmp) )
                                return ERR_NEWORDER_SUPPW_INVALID;

                        if ((pNewOrderData->Ol[i].ol_supply_w_id =
(short)atoi(szTmp)) != pNewOrderData->w_id)
                                pNewOrderData->o_all_local=0;

                        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;
          iMaxWareHouses      = 500;
          iThreads            = 5;
          iDelayMs            = 100;
          iDeadlockRetry      = (short)3;
          strcpy(szTpccLogPath, "tpcclog.");


          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, "MaximumWarehouses", 0, &type, szTmp, &size) ==
ERROR_SUCCESS )
          {
                    iMaxWareHouses = atoi(szTmp);
                    if ( iMaxWareHouses == 0 )
                              iMaxWareHouses = 500;
          }

          size = sizeof(szTmp);
          if ( RegQueryValueEx(hKey, "NumberOfDeliveryThreads", 0, &type, szTmp,
&size) == ERROR_SUCCESS )
                    iThreads = atoi(szTmp);
          if ( !iThreads )
                    iThreads = 5;

          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 PostDeliveryInfo(short w_id, short o_carrier_id)
 *
 * PURPOSE:         This function writes the delivery information to the delivery
pipe. The information is
 *                                  sent as a long.
 *
 * ARGUMENTS:       short                           w_id
          warehouse id
 *                                        short                           o_carrier_id
          carrier id
 *
 * RETURNS:                   BOOL      FALSE     delivery information posted
successfully
 *                                                            TRUE      error cannot
post delivery info
 *
 * COMMENTS:        The pipe is initially created with 16K buffer size this should
allow for
 *                                        up to 4096 deliveries to be queued before an
overflow condition would
 *                                        occur. The only reason that an overflow would
occur is if the delivery
 *                                        application stopped listening while
deliveries were being posted.
 *
 */

static BOOL PostDeliveryInfo(short w_id, short o_carrier_id)
{
          DELIVERY_TRANSACTION          deliveryTransaction;
          int                                                       d;
          int                                                       i;

          GetLocalTime(&deliveryTransaction.queue);

          deliveryTransaction.w_id                          =         w_id;
```

```
        deliveryTransaction.o_carrier_id        =       o_carrier_id;

        for(i=0; i<4; i++)
        {
                if ( WriteFile(hPipe, &deliveryTransaction,
sizeof(deliveryTransaction), &d, NULL) )
                        return FALSE;

                if ( GetLastError() != ERROR_PIPE_BUSY )
        //ERROR_PIPE_LISTENING
                        return TRUE;
        }

        return TRUE;
}

/* 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 );
}

#ifdef USE_ODBC
        /* FUNCTION: static int ODBCError(DBPROCESS *dbproc)
         *
         * PURPOSE:         This function Handles the processing of errors from
ODBC APIs
         *
         * ARGUMENTS:       PDBPROCESS
         *
         * RETURNS:                 BOOL      FALSE     if string is not all
numeric
         *                                                      TRUE      if
string contains only numeric characters i.e. '0' - '9'
         *
         * COMMENTS:        None
         *
         */

        static int ODBCError(DBPROCESS *dbproc)
        {
                RETCODE     rc;
                SDWORD      lNativeError;
                BOOL                bError;
                char        szState[6];
                char        szMsg[SQL_MAX_MESSAGE_LENGTH];
                char                timebuf[128];
                char                datebuf[128];

                pdbproc->deadlock_detected = TRUE;
                bError = FALSE;

                while( SQLError(dbproc->henv,
                                                        dbproc->hdbc,
                                                        dbproc->hstmt,
                                                        szState,
                                                        &lNativeError,
                                                        szMsg,
                                                        sizeof(szMsg),
                                                        NULL) !=
SQL_NO_DATA_FOUND )
                {
                        if (lNativeError == 1205)
                                dbproc->deadlock_detected = TRUE;
                        else
                        {
                                _strtime(timebuf);
                                _strdate(datebuf);

                                h_printf(dbproc->pECB, "%s %s : ODBC Error:
State=%s, Error=%ld, %s\n", datebuf, timebuf, szState, lNativeError, szMsg);
                                bError = TRUE;
                        }
                }
                if ( bError )
                        return -1;
                return dbproc->deadlock_detected;
        }
#endif

/* 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;
}
```

TPCC.DEF

```
LIBRARY TPCC.DLL

EXPORTS

        GetExtensionVersion @1
        HttpExtensionProc   @2
```

TPCC.H

```
/*      FILE:               TPCC.H
 *                                  Microsoft TPC-C Kit Ver. 3.00.000
 *                                  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
 */

//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   ERR_TYPE_WEBDLL
          1
#define   ERR_TYPE_SQL
          2
#define   ERR_TYPE_DBLIB
          3

#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*".

//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)

//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;

//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 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

          DBPROCESS *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
} 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

#ifdef USE_ODBC
          typedef struct _DBPROCESS
          {
                    HDBC                                                    hdbc;
                    HSTMT                                                   hstmt;
                    int                                                             spid;
                    int
          iTermId;
                    int
          iSyncId;
                    BOOL                                                    bDeadlock;
                    EXTENSION_CONTROL_BLOCK       *pECB;
          } DBPROCESS, *PDBPROCESS;
#else
          //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;
#endif

//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);
static void WriteZString(EXTENSION_CONTROL_BLOCK *pECB, char *szStr);
static void h_printf(EXTENSION_CONTROL_BLOCK *pECB, char *format, ...);
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);
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 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);
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 int ODBCError(DBPROCESS *dbproc);
static void FormatHTMLString(char *szBuff, char *szStr, int iLen);
```

TPCC.RC

```
//Microsoft Developer Studio generated resource script.
//
#include "resource.h"

#define APSTUDIO_READONLY_SYMBOLS
////////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "afxres.h"

////////////////////////////////////////////////////////////////////////////////
#undef APSTUDIO_READONLY_SYMBOLS

////////////////////////////////////////////////////////////////////////////////
// English (U.S.) resources

#if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_ENU)
#ifdef _WIN32
LANGUAGE LANG_ENGLISH, SUBLANG_ENGLISH_US
#pragma code_page(1252)
```

```
#endif //_WIN32

#ifndef _MAC
/////////////////////////////////////////////////////////////////////////////
//
// Version
//

VS_VERSION_INFO VERSIONINFO
 FILEVERSION 0,3,0,2
 PRODUCTVERSION 0,3,0,2
 FILEFLAGSMASK 0x3fL
#ifdef _DEBUG
 FILEFLAGS 0x1L
#else
 FILEFLAGS 0x0L
#endif
 FILEOS 0x40004L
 FILETYPE 0x2L
 FILESUBTYPE 0x0L
BEGIN
    BLOCK "StringFileInfo"
    BEGIN
        BLOCK "040904b0"
        BEGIN
            VALUE "Comments", "TPC-C HTML DLL Server\0"
            VALUE "CompanyName", "Microsoft\0"
            VALUE "FileDescription", "tpcc\0"
            VALUE "FileVersion", "0, 3, 0, 2\0"
            VALUE "InternalName", "tpcc\0"
            VALUE "LegalCopyright", "Copyright © 1996\0"
            VALUE "OriginalFilename", "tpcc.dll\0"
            VALUE "ProductName", "Microsoft tpcc\0"
            VALUE "ProductVersion", "0, 3, 0, 2\0"
        END
    END
    BLOCK "VarFileInfo"
    BEGIN
        VALUE "Translation", 0x409, 1200
    END
END

#endif    // !_MAC


#ifdef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////////////////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE DISCARDABLE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE DISCARDABLE
BEGIN
    "#include ""afxres.h""\r\n"
    "\0"
END

3 TEXTINCLUDE DISCARDABLE
BEGIN
    "\r\n"
    "\0"
END

#endif    // APSTUDIO_INVOKED

#endif    // English (U.S.) resources
/////////////////////////////////////////////////////////////////////////////


#ifndef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 3 resource.
//


/////////////////////////////////////////////////////////////////////////////
#endif    // not APSTUDIO_INVOKED
```

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;
                  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;
                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;
                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;
                DEL_ITEM                            DelItems[10];
                char                        execution_status[STATUS_LEN];
        } DELIVERY_DATA;

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

#endif
```

Appendix B: Database Design

The TPC-C database was created with the following Transact-SQL scripts:

DELIVERY.SQL

```
/*  File:       DELIVERY.SQL
*/
/*              Microsoft TPC-C Kit Ver. 3.00.000
*/
/*              Audited 08/23/96, By Francois Raab
*/
/*
*/
/*              Copyright Microsoft, 1996
*/
/*
*/
/*  Purpose:    Delivery transaction for Microsoft
TPC-C Benchmark Kit     */
/*  Author:     Damien Lindauer
*/
/*              damienl@Microsoft.com
*/


use tpcc
go

/* delivery transaction */

if exists (select name from sysobjects where name =
"tpcc_delivery" )
         drop procedure tpcc_delivery
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 @o_id = min(no_o_id)
        from new_order holdlock
        where no_w_id = @w_id and
                          no_d_id = @d_id

        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

select @oid1,
       @oid2,
       @oid3,
       @oid4,
       @oid5,
       @oid6,
       @oid7,
       @oid8,
       @oid9,
       @oid10

go
```

IDXCUSCL.SQL

```
/*  TPC-C Benchmark Kit                          */
/*                                  */
/*  IDXCUSCL.SQL                                 */
/*                                  */
```

```
/*  Creates clustered index on customer (seg)                   */


use tpcc
go


if exists ( select name from sysindexes where name = 'customer_c1' )
                drop index customer.customer_c1
go

select getdate()
go
create unique clustered index customer_c1 on customer(c_w_id, c_d_id, c_id)
                with sorted_data on big_seg
go
select getdate()
go
```

IDXCUSNC.SQL

```
/*  TPC-C Benchmark Kit                                         */
/*                                                  */
/*  IDXCUSNC.SQL                                    */
/*                                                  */
/*  Creates non-clustered index on customer (seg)               */


use tpcc
go


if exists ( select name from sysindexes where  name = 'customer_nc1' )
                drop index customer.customer_nc1
go

select getdate()
go
create unique nonclustered index customer_nc1 on customer(c_w_id, c_d_id,
c_last, c_first, c_id)
                on big_seg
go
select getdate()
go
```

IDXDISCL.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  IDXDISCL.SQL
*/
/*
*/
/*  Creates clustered index on district (seg)
*/


use tpcc
go


if exists ( select name from sysindexes where name =
'district_c1' )
          drop index district.district_c1
go

select getdate()
go
create unique clustered index  district_c1 on
district(d_w_id, d_id)
          with fillfactor=1 on misc_seg
go
select getdate()
go
```

IDXITMCL.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  IDXITMCL.SQL
*/
/*
*/
/*  Creates clustered index on item (seg)
*/


use tpcc
go


if exists ( select name from sysindexes where  name =
'item_c1' )
          drop index item.item_c1
go

select getdate()
go
create unique clustered index item_c1 on item(i_id)
          with sorted_data on misc_seg
go
select getdate()
go
```

IDXNODCL.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  IDXNODCL.SQL
*/
/*
*/
/*  Creates clustered index on new-order (seg)
*/


use tpcc
go


if exists ( select name from sysindexes where  name =
'new_order_c1' )
          drop index new_order.new_order_c1
go

select getdate()
go
create unique clustered index new_order_c1 on
new_order(no_w_id, no_d_id, no_o_id)
          with sorted_data on misc_seg
go
select getdate()
go
```

IDXODLCL.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  IDXODLCL.SQL
*/
/*
*/
/*  Creates clustered index on order-line (seg)
*/


use tpcc
go


if exists ( select name from sysindexes where  name =
'order_line_c1' )
          drop index order_line.order_line_c1
go

select getdate()
go
```

```
create unique clustered index order_line_c1 on
order_line(ol_w_id, ol_d_id, ol_o_id, ol_number)
         with sorted_data on ol_seg
go
select getdate()
go
```

IDXORDCL.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  IDXORDCL.SQL
*/
/*
*/
/*  Creates clustered index on orders (seg)
*/


use tpcc
go


if exists ( select name from sysindexes where  name =
'orders_c1' )
         drop index orders.orders_c1
go

select getdate()
go
create unique clustered index orders_c1 on
orders(o_w_id, o_d_id, o_id)
         with sorted_data on misc_seg
go
select getdate()
go
```

IDXSTKCL.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  IDXSTKCL.SQL
*/
/*
*/
/*  Creates clustered index on stock (seg)
*/


use tpcc
go

if exists ( select name from sysindexes where  name =
'stock_c1' )
         drop index stock.stock_c1
go

select getdate()
go
create unique clustered index stock_c1 on
stock(s_i_id, s_w_id)
         with sorted_data on big_seg
go
select getdate()
go
```

IDXWARCL.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  IDXWARCL.SQL
*/
/*
*/
/*  Creates clustered index on warehouse (seg)
*/


use tpcc
go


if exists ( select name from sysindexes where name =
'warehouse_c1' )
         drop index warehouse.warehouse_c1
go

select getdate()
go
create unique clustered index warehouse_c1 on
warehouse(w_id)
         with fillfactor=1 on misc_seg
go
select getdate()
go
```

NEWORD.SQL

```
/*  File:       NEWORD.SQL
*/
/*              Microsoft TPC-C Kit Ver. 3.00.000
*/
/*              Audited 08/23/96, By Francois Raab
*/
/*
*/
/*              Copyright Microsoft, 1996
*/
/*
*/
/*  Purpose:    New-Order transaction for Microsoft
TPC-C Benchmark Kit    */
/*  Author:     Damien Lindauer
*/
/*              damienl@Microsoft.com
*/


use tpcc
go

/* new-order transaction stored procedure */

if exists ( select name from sysobjects where name =
"tpcc_neworder" )
         drop procedure tpcc_neworder
go

/*  Modified by rick vicik, 2/4/97  */
/*  Combined initialization of local variables into
district update statement  */
/*  Combined 3 huge case select statements into a
single one  */

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    int,
        @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

              /*  Set i_id, s_w_id, and qty for this
lineitem */

              select @li_no=@li_no+1, @li_id = case
@li_no
                      when 0 then @i_id1
                                  when 1 then @i_id2
                                  when 2 then @i_id3
                                  when 3 then @i_id4
                                  when 4 then @i_id5
                                  when 5 then @i_id6
                                  when 6 then @i_id7
                                  when 7 then @i_id8
                                  when 8 then @i_id9
                                  when 9 then @i_id10
                                  when 10 then
@i_id11                           when 11 then
@i_id12                           when 12 then
@i_id13                           when 13 then
@i_id14                           when 14 then
@i_id15                           end,


                                  @li_s_w_id = case
@li_no                            when 0 then
@s_w_id1                          when 1 then
@s_w_id2                          when 2 then
@s_w_id3                          when 3 then
@s_w_id4                          when 4 then
@s_w_id5                          when 5 then
@s_w_id6                          when 6 then
@s_w_id7                          when 7 then
@s_w_id8                          when 8 then
@s_w_id9                          when 9 then
@s_w_id10                         when 10 then
@s_w_id11                         when 11 then
@s_w_id12                         when 12 then
@s_w_id13                         when 13 then
@s_w_id14                         when 14 then
@s_w_id15                         end,

                                  @li_qty = case
@li_no                            when 0 then
@ol_qty1                          when 1 then
@ol_qty2                          when 2 then
@ol_qty3                          when 3 then
@ol_qty4                          when 4 then
@ol_qty5                          when 5 then
@ol_qty6                          when 6 then
@ol_qty7                          when 7 then
@ol_qty8                          when 8 then
@ol_qty9                          when 9 then
@ol_qty10                         when 10 then
@ol_qty11                         when 11 then
@ol_qty12                         when 12 then
@ol_qty13                         when 13 then
@ol_qty14                         when 14 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 holdlock)
              where i_id = @li_id

            /* if there actually is an item with
this id, go to work */

                    if (@@rowcount > 0)
                    begin
              update stock set s_ytd        = s_ytd +
@li_qty,
```

```
                            s_quantity   =
s_quantity - @li_qty +
                                case when
(s_quantity - @li_qty < 10) then 91 else 0 end,
                            @s_quantity  =
s_quantity,
                            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

                           /* insert order_line
data (using data from item and stock) */

            insert into order_line values(@o_id,
/* from district update */
                                        @d_id,
                /* input param          */
                                        @w_id,
/* input param          */
                                        @li_no,
/* orderline number     */
                                        @li_id,
/* lineitem id          */
                                        @li_s_w_id,
/* lineitem warehouse   */
                                        "dec 31,
1889",      /* constant             */
                                        @li_qty,
/* lineitem qty         */
                                        @i_price *
@li_qty, /* ol_amount            */
                                              @s_dist)
/* from stock           */


            /* 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 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 holdlock
    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 holdlock
          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
```

ORDSTAT.SQL

```
/*  File:       ORDSTAT.SQL
*/
/*              Microsoft TPC-C Kit Ver. 3.00.000
*/
/*              Audited 08/23/96, By Francois Raab
*/
/*
*/
/*              Copyright Microsoft, 1996
*/
/*
*/
/*  Purpose:    Order-Status transaction for
Microsoft TPC-C Benchmark Kit  */
/*  Author:     Damien Lindauer
*/
/*              damienl@Microsoft.com
*/


use tpcc
go

if exists ( select name from sysobjects where name =
"tpcc_orderstatus" )
          drop procedure   tpcc_orderstatus
```

```
go

/*  Modified by rick vicik, 2/4/97  */
/*  Eliminated @val local variable  */

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 holdlock
                    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 holdlock
                    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 holdlock
                    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 holdlock
      where o_w_id = @w_id and
          o_d_id = @d_id and
           o_c_id = @c_id

          /*  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 holdlock
          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
```

PAYMENT.SQL

```
/*  File:       PAYMENT.SQL
*/
/*              Microsoft TPC-C Kit Ver. 3.00.000
*/
/*              Audited 08/23/96, By Francois Raab
*/
/*
*/
/*              Copyright Microsoft, 1996
*/
/*
*/
/*  Purpose:    Payment transaction for Microsoft
TPC-C Benchmark Kit      */
/*  Author:     Damien Lindauer
*/
/*              damienl@Microsoft.com
*/


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,

           @c_w_id         smallint,

           @h_amount       numeric(6,2),

           @d_id           tinyint,

           @c_d_id         tinyint,

           @c_id           int,

           @c_last         char(16) = ""



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),
        @data1          char(250),
        @data2          char(250),
        @c_data_1       char(250),
        @c_data_2       char(250),
        @datetime       datetime,
        @w_ytd          numeric(12,2),
        @d_ytd          numeric(12,2),
        @cnt            smallint,
        @val            smallint,
        @screen_data          char(200),
                @d_id_local         tinyint,
                @w_id_local         smallint,
                @c_id_local         int

select @screen_data = ""

begin tran p

        /* get payment date */

        select @datetime = getdate()

        if (@c_id = 0)
        begin
                /* get customer id and info using
last name */

                select @cnt = count(*)
                from customer holdlock
                where c_last = @c_last and
                      c_w_id = @c_w_id and
                      c_d_id = @c_d_id


                select @val = (@cnt + 1) / 2
                set rowcount @val

                select @c_id = c_id
                from customer holdlock
                where c_last = @c_last and
                      c_w_id = @c_w_id and
                      c_d_id = @c_d_id
                order by c_w_id, c_d_id, c_last,
c_first
                set rowcount 0
        end

        /* get customer info and update balances */

        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,
                @data1         = c_data_1,
                @data2         = c_data_2,
                @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

                /* compute new info */

                select @c_data_2 =
substring(@data1,209,42) +

            substring(@data2, 1, 208)
                select @c_data_1 =
convert(char(5),@c_id) +

            convert(char(4),@c_d_id) +

convert(char(5),@c_w_id) +

convert(char(4),@d_id) +

convert(char(5),@w_id) +

convert(char(19),@h_amount) +

substring(@data1, 1, 208)

                /* update customer info */

                update customer set
                       c_data_1 = @c_data_1,
                              c_data_2 = @c_data_2
                where c_id   = @c_id and
                            c_w_id = @c_w_id and
                      c_d_id = @c_d_id

                select @screen_data = substring
(@c_data_1,1,200)
        end



        /* get district data and update year-to-
date */

        update district
                set d_ytd       = d_ytd +
@h_amount,
                        @d_street_1 =
d_street_1,
                        @d_street_2 =
d_street_2,
                        @d_city     = d_city,
                        @d_state    = d_state,
                        @d_zip      = d_zip,
                        @d_name     = d_name,
                        @d_id_local = d_id
                where d_w_id = @w_id and
                      d_id   = @d_id

        /* get warehouse data and update year-to-
date */

        update warehouse
        set w_ytd       = w_ytd + @h_amount,
                @w_street_1 = w_street_1,
            @w_street_2 = w_street_2,
            @w_city     = w_city,
            @w_state    = w_state,
            @w_zip      = w_zip,
            @w_name     = w_name,
                @w_id_local = w_id
        where w_id = @w_id

        /* create history record */

        insert into history values  (@c_id_local,

                    @c_d_id,

                        @c_w_id,

                        @d_id_local,

                        @w_id_local,

                        @datetime,

                        @h_amount,
```

```
                                @w_name + "    " +
@d_name)

commit tran p

/* return data to client */

select  @c_id,
                    @c_last,
                    @datetime,
                    @w_street_1,
                    @w_street_2,
                    @w_city,
                    @w_state,
                    @w_zip,
                    @d_street_1,
                    @d_street_2,
                    @d_city,
                    @d_state,
                    @d_zip,
                    @c_first,
                    @c_middle,
                    @c_street_1,
                    @c_street_2,
                    @c_city,
                    @c_state,
                    @c_zip,
                    @c_phone,
                    @c_since,
                    @c_credit,
                    @c_credit_lim,
                    @c_discount,
                    @c_balance,
                    @screen_data

go
```

PINTABLE.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  PINTABLE.SQL
*/
/*
*/
/*  This script file is used to 'pin' certain tables
in the data cache        */

use tpcc
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
```

SEGMENT.SQL

```
/*  TPC-C Benchmark Kit
*/
/*  SEGMENT.SQL
*/
/*  This script is used to create the database
segments  */

use tpcc
go

/*                                                */
/* Create segment for Customer and Stock tables */
/*                                                */

exec sp_addsegment big_seg, bigdev1
exec sp_extendsegment big_seg, bigdev2
exec sp_extendsegment big_seg, bigdev3
exec sp_extendsegment big_seg, bigdev4
exec sp_extendsegment big_seg, bigdev5

/*                                     */
/* Create segment for Orderline table */
/*                                     */

exec sp_addsegment ol_seg, ol1

/*
*/
/* Create segment for all other tables :
*/
/*
*/
/* Warehouse, District, Item, New-Order, History,
Orders */
/*
*/

exec sp_addsegment misc_seg, misc1
go
```

STOCKLEV.SQL

```
/*  File:       STOCKLEV.SQL
*/
/*              Microsoft TPC-C Kit Ver. 3.00.000
*/
/*              Audited 08/23/96, By Francois Raab
*/
/*
*/
/*              Copyright Microsoft, 1996
*/
/*
*/
/*  Purpose:    Stock-Level transaction for Microsoft
TPC-C Benchmark Kit  */
/*  Author:     Damien Lindauer
*/
/*              damienl@Microsoft.com
*/


use tpcc
go

/* stock-level transaction stored procedure */

if exists (select name from sysobjects where name =
"tpcc_stocklevel" )
          drop procedure tpcc_stocklevel
go

/* Modified by rick vicik, 2/4/97 */
/* Eliminate 1 local variable, use derived table to
eliminate duplicate item#'s */

create proc tpcc_stocklevel   @w_id
          smallint,

          @d_id               tinyint,

          @threshhold         smallint
as
declare @o_id int

select @o_id = d_next_o_id
from district
where d_w_id = @w_id and
            d_id   = @d_id

select count(*) from stock,
    (select distinct(ol_i_id) from order_line
     where ol_w_id    =  @w_id and
                ol_d_id    =  @d_id and
                ol_o_id between (@o_id-20) and
(@o_id-1)) OL

where s_w_id     =  @w_id and
      s_i_id     =  OL.ol_i_id and
      s_quantity <  @threshhold
go
```

TABLES.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  TABLES.SQL
*/
/*
*/
```

```
/*  Creates TPC-C tables (seg)
*/


use tpcc
go

checkpoint
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 misc_seg
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 misc_seg
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_1                         char(250),
          c_data_2                         char(250)
) on big_seg
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 misc_seg
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 misc_seg
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 misc_seg
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 ol_seg
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 misc_seg
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 big_seg
go
```

TPCCIRL.SQL

```
/*  TPC-C Benchmark Kit
*/
/*
*/
/*  TPCCIRL.SQL
*/
/*
*/
/*  This script file sets the insert row lock option
on selected tables       */


use tpcc
go

exec sp_tableoption "history","insert row lock",true
exec sp_tableoption "new_order","insert row
lock",true
exec sp_tableoption "orders","insert row lock",true
exec sp_tableoption "order_line","insert row
lock",true
go
```

CREATEDB.SQL

```
/*  TPC-C Benchmark Kit                             */
/*                                                  */
/*  CREATEDB.SQL                                    */
/*                                                  */
/*  This script is used to create the database      */


use master
go

if exists ( select name from sysdatabases where name
= "tpcc" )
        drop database tpcc
go

create database tpcc on

                bigdev1         =       1100,
                bigdev2         =       1100,
                bigdev3         =       1100,
                bigdev4         =       1100,
                bigdev5         =       550,

                bigdev1         =       1100,
                bigdev2         =       1100,
                bigdev3         =       1100,
                bigdev4         =       1100,
                bigdev5         =       550,

                bigdev1         =       1100,
                bigdev2         =       1100,
                bigdev3         =       1100,
                bigdev4         =       1100,
                bigdev5         =       550,

                bigdev1         =       1100,
                bigdev2         =       1100,
                bigdev3         =       1100,
                bigdev4         =       1100,
                bigdev5         =       550,

                bigdev1         =       880,
                bigdev2         =       880,
                bigdev3         =       880,
                bigdev4         =       880,
                bigdev5         =       880,

                ol1             =       9200,

                misc1           =       1400

                log on tpclog   =       12800
go
```

DISKINIT.SQL

```
/*  TPC-C Benchmark Kit                             */
/*                                                  */
/*  DISKINIT.SQL                                    */
/*                                                  */
/*  This script is used create devices              */


use master
go

/*  Log device  */

disk init name = "tpclog",
```

```
          physname = "k:",
          vdevno   = 14,
          size     = 6815744
go

/*  Database devices */

disk init name = "bigdev1",
          physname = "e:",
          vdevno   = 15,
          size     = 3072000
go

disk init name = "bigdev2",
          physname = "f:",
          vdevno   = 16,
          size     = 3072000
go

disk init name = "bigdev3",
          physname = "g:",
          vdevno   = 17,
          size     = 3072000
go

disk init name = "bigdev4",
          physname = "h:",
          vdevno   = 18,
          size     = 3072000
go

disk init name = "bigdev5",
          physname = "i:",
          vdevno   = 19,
          size     = 3072000
go

disk init name = "oll",
          physname = "j:",
          vdevno   = 20,
          size     = 4710400
go

disk init name = "misc1",
          physname = "l:",
          vdevno   = 21,
          size     = 716800
go
```

Appendix C: Tunable Parameters

Microsoft Windows NT v. 4.0 Configuration Parameters

There were no Windows NT Registry parameters that were changed from their default settings.

Microsoft SQL Server v. 6.50.242 Startup Parameters

c:\sql65\binn\sqlservr -c -x -t3502 -t1081 –t812

Where:

- -c Start SQL Server independently of the Windows NT Service Control Manager
- -x Disables the keeping of CPU time and cache-hit ratio statistics
- -t1081 Allows the index pages a "second" trip through the cache
- -t3502 Prints a message to the SQL Server log at the start and end of each checkpoint
- -t812 Disables checkpoint io buffer sorting

Microsoft Windows NT v. 4.0 Configuration Parameters

The following services were disabled in the Windows NT Control Panel/Services on both the Server and the Client systems:

- Computer Browser
- License Logging Service
- Messenger
- NT LM Security Support Provider
- Plug and Play
- Server
- Spooler
- TCP/IP Netbios Helper

SQL Server Stack Size

The default stack size for Microsoft SQL Server 6.5.242 was changed using the EDITBIN utility. The EDITBIN utility ships with Microsoft Visual C++ V4.0. The command used to change the stack size is:

editbin /S:65536 sqlservr.exe

This command is fully documented as an article in the Microsoft Knowledge Base on the Microsoft Web Site at www.microsoft.com/support.

DBCC GAMINIT

Prior to the execution of the benchmark, the following script was run to proactively populate the Global Allocation Map (GAM) rather than allowing it to be populated on an as-needed basis.

```
Use tpcc
go
dbcc gaminit
go
```

This command is fully documented as an article in the Microsoft Knowledge Base on the Microsoft Web Site at www.microsoft.com/support.

Internet Information Server Registry Parameters

```
Key Name:         SYSTEM\CurrentControlSet\Services\InetInfo
Class Name:       <NO CLASS>
Last Write Time:  1/9/97 - 12:39 PM

Key Name:
SYSTEM\CurrentControlSet\Services\InetInfo\Parameters
Class Name:       <NO CLASS>
Last Write Time:  4/28/97 - 8:18 PM
Value 0
  Name:           BandwidthLevel
  Type:           REG_DWORD
  Data:           0xffffffff

Value 1
  Name:           ListenBackLog
  Type:           REG_DWORD
  Data:           0x800

Value 2
  Name:           PoolThreadsLimit
```

```
  Type:         REG_DWORD
  Data:         0x1fe

Value 3
  Name:           ThreadTimeout
  Type:         REG_DWORD
  Data:         0x15180


Key Name:
SYSTEM\CurrentControlSet\Services\lnetInfo\Parameters\Filter
Class Name:       <NO CLASS>
Last Write Time:  1/9/97 - 12:39 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\lnetInfo\Parameters\MimeMap
Class Name:       <NO CLASS>
Last Write Time:  1/9/97 - 12:39 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\lnetInfo\Performance
Class Name:       <NO CLASS>
Last Write Time:  1/9/97 - 12:39 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:         0x738

Value 3
  Name:         First Help
  Type:         REG_DWORD
  Data:         0x739

Value 4
  Name:         Last Counter
  Type:         REG_DWORD
  Data:         0x756

Value 5
  Name:         Last Help
  Type:         REG_DWORD
  Data:         0x757

Value 6
  Name:         Library
  Type:         REG_SZ
  Data:         infoctrs.DLL

Value 7
  Name:         Open
  Type:         REG_SZ
  Data:         OpenINFOPerformanceData

 Key Name:        SOFTWARE\Microsoft\INetMgr
Class Name:       <NO CLASS>
Last Write Time:  1/9/97 - 12:39 PM
Value 0
  Name:         InstalledBy
  Type:         REG_SZ
  Data:         INetStp


Key Name:        SOFTWARE\Microsoft\INetMgr\Parameters
Class Name:       <NO CLASS>
Last Write Time:  4/28/97 - 8:36 PM
Value 0
  Name:         dx
  Type:         REG_DWORD
  Data:         0x1e0

Value 1
  Name:         dy
  Type:         REG_DWORD
  Data:         0x141

Value 2
  Name:         HelpLocation
  Type:         REG_SZ
  Data:         iisadmin\htmldocs\inetdocs.htm

Value 3
  Name:         MajorVersion
  Type:         REG_DWORD
  Data:         0x2

Value 4
  Name:         MinorVersion
  Type:         REG_DWORD
  Data:         0

Value 5
  Name:         Mode
  Type:         REG_DWORD
  Data:         0x1

Value 6
  Name:         View
  Type:         REG_DWORD
  Data:         0x800b

Value 7
  Name:         WaitTime
  Type:         REG_DWORD
  Data:         0x7530

Value 8
  Name:         x
  Type:         REG_DWORD
  Data:         0x42

Value 9
  Name:         y
  Type:         REG_DWORD
  Data:         0x42


Key Name:
SOFTWARE\Microsoft\INetMgr\Parameters\AddOnServices
Class Name:       <NO CLASS>
Last Write Time:  1/9/97 - 12:39 PM
Value 0
  Name:         FTP
  Type:         REG_SZ
  Data:         fscfg.dll

Value 1
  Name:         Gopher
  Type:         REG_SZ
  Data:         gscfg.dll

Value 2
  Name:         WWW
  Type:         REG_SZ
  Data:         w3scfg.dll


Key Name:
SOFTWARE\Microsoft\INetMgr\Parameters\AddOnTools
Class Name:       <NO CLASS>
```

Last Write Time: 1/9/97 - 12:39 PM
Value 0
Name: &Key Manager
Type: REG_SZ
Data: C:\WINNT40\System32\inetsrv\keyring.exe;Key Manager

Key Name: SOFTWARE\Microsoft\Inetsrv
Class Name: GenericClass
Last Write Time: 1/9/97 - 12:38 PM

Key Name: SOFTWARE\Microsoft\Inetsrv\CurrentVersion
Class Name: GenericClass
Last Write Time: 1/9/97 - 12:39 PM
Value 0
Name: Description
Type: REG_SZ
Data: Microsoft Internet Information Server 2.0

Value 1
Name: InstallDate
Type: REG_DWORD
Data: 0x32d59dd4

Value 2
Name: MajorVersion
Type: REG_DWORD
Data: 0x4

Value 3
Name: MinorVersion
Type: REG_DWORD
Data: 0

Value 4
Name: OperationsSupport
Type: REG_DWORD
Data: 0x86

Value 5
Name: ServiceName
Type: REG_SZ
Data: Microsoft Internet Information Server 2.0

Value 6
Name: SoftwareType
Type: REG_SZ
Data: service

Value 7
Name: Title
Type: REG_SZ
Data: Microsoft Internet Information Server 2.0

Key Name:
SOFTWARE\Microsoft\Inetsrv\CurrentVersion\NetRules
Class Name: GenericClass
Last Write Time: 1/9/97 - 12:39 PM
Value 0
Name: InfName
Type: REG_SZ
Data: oemnsvin.inf

Value 1
Name: InfOption
Type: REG_SZ
Data: Inetsrv

World Wide Web Service Registry Parameters

Key Name: SYSTEM\CurrentControlSet\Services\W3SVC
Class Name: <NO CLASS>
Last Write Time: 1/9/97 - 12:39 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:\WINNT40\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: 5/1/97 - 4:13 AM
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\Parameters
Class Name: <NO CLASS>
Last Write Time: 4/28/97 - 8:36 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_CLIENT30

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:\WINNT40\System32\inetsrv\sspifilt.dll

Value 13
Name: GlobalExpire
Type: REG_DWORD
Data: 0xffffffff

Value 14
Name: InstallPath
Type: REG_SZ
Data: C:\WINNT40\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\Deny IP List
Class Name: <NO CLASS>
Last Write Time: 4/28/97 - 8:18 PM

Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Grant IP List
Class Name: <NO CLASS>
Last Write Time: 4/28/97 - 8:18 PM

Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Script Map
Class Name: <NO CLASS>
Last Write Time: 1/9/97 - 12:39 PM
Value 0
Name: .idc
Type: REG_SZ
Data: C:\WINNT40\System32\inetsrv\httpodbc.dll

Key Name:
SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Virtual Roots
Class Name: <NO CLASS>
Last Write Time: 4/28/97 - 8:36 PM
Value 0
Name: /,
Type: REG_SZ
Data: C:\InetPub\wwwroot,,5

Value 1
Name: /iisadmin,
Type: REG_SZ
Data: C:\WINNT40\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: 1/9/97 - 12:39 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: 0x758

Value 3

```
  Name:          First Help
  Type:          REG_DWORD
  Data:          0x759

Value 4
  Name:          Last Counter
  Type:          REG_DWORD
  Data:          0x790

Value 5
  Name:          Last Help
  Type:          REG_DWORD
  Data:          0x791

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:  1/9/97 - 12:39 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   00 00 76 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
..v.............
00000060   00 00 00 05 20 00 00 00 - 23 02 00 00 e0 00 14 00  ....
...#.......
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 - e0 00 14 00 00 00 1c 00   ...
...........
00000090   ff 01 0f 00 01 02 00 00 - 00 00 00 05 20 00 00 00  ............
...
000000a0   25 02 00 00 e0 00 14 00 - 00 00 18 00 fd 01 02 00
%...............
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:  1/9/97 - 12:39 PM
```

TPCC Application Registry Parameters

```
Key Name:         SOFTWARE\Microsoft\TPCC
Class Name:       <NO CLASS>
Last Write Time:  4/28/97 - 8:15 PM
Value 0
  Name:          BackoffDelay
  Type:          REG_SZ
  Data:          500

Value 1
  Name:          DeadlockRetry
  Type:          REG_SZ
  Data:          3

Value 2
  Name:          LOG
  Type:          REG_SZ
  Data:          OFF

Value 3
  Name:          MaxConnections
  Type:          REG_SZ
  Data:          2000

Value 4
  Name:          MaximumWarehouses
  Type:          REG_SZ
  Data:          420

Value 5
  Name:          NumberOfDeliveryThreads
  Type:          REG_SZ
  Data:          5

Value 6
  Name:          PATH
  Type:          REG_SZ
  Data:          C:\InetPub\wwwroot\

Value 7
  Name:          QueueSlotts
  Type:          REG_SZ
  Data:          3000
```

CLIENT SYSTEM CONFIGURATION UTILITY

All 3 client machines had identical hardware configurations, as displayed below:

```
"
Date . . . . . .  5/1/97
"
Time . . . . . .  04:09:55
"

"
Product . . . . . .  Compaq ProLiant 800

Machine ID
  From System Board . . . . . .  CPQ0579

Processor . . . . . .  Pentium Pro(R) at 180 MHz
Secondary Cache . . . . . .  256K
CPU ID . . . . . .  0617

Processor(s) Mapped Out . . . .  None

Numeric Coprocessor . . . . . .  Integrated 387-Compatible

Expansion Bus . . . . . .  ISA, PCI

System Identification Number . .  D648BJW30047

CPU Mode . . . . . .  Real Mode

System ROM
   Revision . . . . . .  11/14/1996
     Family . . . . . .  P02
   Flashable . . . . . .  Yes
   Supports F10 partition . . .  Yes

Video Controller ROM
   Revision . . . . . .  1.6 (Cirrus)

Keyboard Controller ROM
   Revision . . . . . .  C.8 09/24/95
     Family . . . . . .  K

Option ROMs
  Address Range . . . . . .  C0000 - C7FFF
    Data Dump . . . . . .  (CL-GD5440 VGA BIOS Version 1.06e
Copyright 19...)

  Address Range . . . . . .  E8000 - EDFFF
    Data Dump . . . . . .  (11/14/96 (C)Copyright COMPAQ Computer
Corporation ...)
```

```
Standby Recovery Server
  Status . . . . . .  Disabled
  COM Port . . . . . .  COM1
  Server Configuration . . . . .  Recovery
  Timeout Value  . . . . . .  1 minutes

Memory Boards Identified:
  System Board
    DIMM Slot  1 (EDO) . . . . .    32 Megabytes
    DIMM Slot  2 (EDO) . . . . .    64 Megabytes
    DIMM Slot  3 . . . . . .     0 Megabytes
    DIMM Slot  4 . . . . . .     0 Megabytes
Total Compaq Memory  . . . . . .    96 Megabytes

Keyboard . . . . . .  Enhanced

LPT Ports  . . . . . .  LPT1 (Address 3BC)

COM Ports  . . . . . .  COM1 (Address 3F8)
                        COM2 (Address 2F8)

Diskette Drive A . . . . . .  1.44 Megabyte (3.5 inch)

Graphics Mode  . . . . . .  03 (80-Column Text)


Primary Monitor attached to  . .  Cirrus CL-GD5430 Graphics Controller
                                  with Video Graphics Color Monitor

Total Video Memory . . . . . .  1024 Kbytes

Base Memory
  System Total . . . . . .    640 Kbytes
  Amount Free  . . . . . .    597 Kbytes (612016 Bytes)

Extended Memory
  Amount Free  . . . . . .  91584 Kbytes

Expanded Memory
  LIM Driver Support . . . . . .  LIM driver not loaded

Operating System . . . . . .  MS-DOS version 7.00 Rev. A

Status of file: C:\IO.SYS
  Date and Time  . . . . . .  07/11/95  09:50:00
  Size . . . . . .  223148 Bytes

Status of file: C:\MSDOS.SYS
  Date and Time  . . . . . .  01/09/97  17:22:18
  Size . . . . . .  1641 Bytes

------------------------------------------------------------------------------
** Dump of C:\CONFIG.SYS
------------------------------------------------------------------------------
** End of file

Status of file: C:\COMMAND.COM
  Date and Time  . . . . . .  07/11/95  09:50:00
  Size . . . . . .  92870 Bytes

------------------------------------------------------------------------------
** Dump of C:\AUTOEXEC.BAT
------------------------------------------------------------------------------
** End of file

Environment variables
  PATH=
  PROMPT=$P$G
  COMSPEC=A:\COMMAND.COM
  CMDLINE=inspect /u
End of environment

------------------------------------------------------------------------------
** Dump of C:\DIRECT.OUT
------------------------------------------------------------------------------
[boot]
system.drv=system.drv
drivers=mmsystem.dll
user.exe=user.exe
gdi.exe=gdi.exe
sound.drv=mmsound.drv
dibeng.drv=dibeng.dll
comm.drv=comm.drv
shell=Explorer.exe
keyboard.drv=keyboard.drv
fonts.fon=vgasys.fon
fixedfon.fon=vgafix.fon
oemfonts.fon=vgaoem.fon
386Grabber=vgafull.3gr
display.drv=pnpdrvr.drv
mouse.drv=mouse.drv
*DisplayFallback=0
[keyboard]
subtype=
type=4
keyboard.dll=
oemansi.bin=
[boot.description]
keyboard.typ=Standard 101/102-Key or Microsoft Natural Keyboard
aspect=100,96,96
display.drv=Cirrus Logic 5429/30/34
mouse.drv=Standard mouse
system.drv=Standard PC
[386Enh]
ebios=*ebios
device=*vshare
device=*dynapage
device=*vcd
device=*vpd
device=*int13
keyboard=*vkd
display=*vdd,*vflatd
mouse=*vmouse, msmouse.vxd
woafont=dosapp.fon
device=*enable
[power.drv]
[drivers]
wavemapper=*.drv
[iccvid.drv]
[mciseq.drv]
[mci]
cdaudio=mcicda.drv
sequencer=mciseq.drv
waveaudio=mciwave.drv
avivideo=mciavi.drv
videodisc=mcipionr.drv
vcr=mcivisca.drv
[NonWindowsApp]
[vcache]
[display]
[drivers32]
vidc.CVID=iccvid.dll
VIDC.IV31=ir32_32.dll
VIDC.IV32=ir32_32.dll
vidc.MSVC=msvidc32.dll
VIDC.MRLE=msrle32.dll
[nwnp32]
[MSNP32]
[Password Lists]
CLIENT30=C:\WINDOWS\CLIENT30.PWL
[TTFontDimenCache]
0 12=5 12
0 13=6 12
0 14=7 14
0 15=7 15
0 16=8 16
0 18=10 18
0 20=10 20
0 22=12 22
** End of file

------------------------------------------------------------------------------
** Dump of C:\DIRECT.OUT
------------------------------------------------------------------------------
[windows]
load=
run=
NullPort=None
[Desktop]
Wallpaper=(None)
TileWallpaper=0
WallpaperStyle=0
[intl]
iCountry=1
ICurrDigits=2
iCurrency=0
iDate=0
iDigits=2
iLZero=1
iMeasure=1
iNegCurr=0
iTime=0
iTLZero=0
s1159=AM
s2359=PM
sCountry=United States
sCurrency=$
sDate=/
sDecimal=.
sLanguage=enu
sList=,
sLongDate=dddd, MMMM dd, yyyy
sShortDate=M/d/yy
sThousand=,
sTime=:
[fonts]
```

```
[FontSubstitutes]
Helv=MS Sans Serif
Tms Rmn=MS Serif
Times=Times New Roman
Helvetica=Arial
MS Shell Dlg=MS Sans Serif
[Compatibility]
_3DPC=0x00400000
_BNOTES=0x224000
_LNOTES=0x00100000
ACAD=0x8000
ACT!=0x400004
ACROBAT=0x04000000
AD=0x10000000
ADW30=0x10000000
ALARMMGR=0x0040000
ALDSETUP=0x00400000
AMIPRINT=0x04000000
AMIPRO=0x04000010
APORIA=0x0100
APPROACH=0x0004
BALER=0x08000000
BMAPP=0x0004
CASMONEY=0x00200000
CAVOIDE=0x00200000
CCMAIL=0x00200000
CCMCWFY=0x80
CHARISMA=0x2000
CONFIG=0x00400000
CORELDRW=0x48000
CORELPNT=0x08000000
COSTAR=0x0004
CP=0x0040
CROSSTIE=0x00000400
DARCH=0x80
DESIGNER=0x00002000
DIRECTOR=0x00800000
DPLANNER=0x00200000
DRAW=0x2000
DS40=0x8000
DTWIN20=0x00000400
EAP=0x0004
ED=0x00010000
EXCEL=0x1000
EXPASTRO=0x04000000
EXTYPWND=0x00200000
FAXVIEW=0x04000000
FAXWORKS=0x00000400
FH4=0x00E08000
FLW2=0x8000
FMPRO=0x00200000
FREEHAND=0x8000
FULLTEXT=0x20000000
GIFTMAKE=0x20000000
GUIDE=0x1000
HDW=0x04800000
HGW=0x8000
HGW2EXE=0x8000
HGW3EXE=0x8000
HJDRAW=0x00400000
IDAPICFG=0x00400000
IDRAW=0x04008000
ILLUSTRATOR=0x8000
IMPROV2=0x00000000
INFOCENT=0x04000000
INSIGHT=0x00000400
INSTAL1=0x00400000
INSTALL=0x00400000
INTERMIS=0x10000000
IS20INST=0x00000000
IVIHEALT=0x00400000
JEOPARDY=0x00200000
JW=0x00000000
KALOAD2=0x00400000
KEYCAD=0x8000
LE_ADMIN=0x00400000
LUI=0x20000000
MAILSPL=0x10000000
MAKER=0x00200000
MAPS1=0x04008022
MATH=0x00000001
MAVIS=0x00200000
MCOURIER=0x0800
MFWIN20=0x02000000
MILESV3=0x1000
MILESV40=0x4
MOZART=0x40000000
MSARTIST=0x00100000
MSBHUMAN=0x4
MSREMIND=0x10000000
MVIEWER2=0x40200000
MYINV=0x00200000
MYST=0x08000000
NAFTA1=0x4008022
NBAMW4V4=0x04000000
NETSET2=0x0100
NOTES=0x200000
NOTSHELL=0x0001
OPERATOR=0x02000000
OUTPOST=0x00000000
OWLAPP=0x00400000
PACKRAT=0x0800
PAINTER=0x00000000
PAWC8DC3=0x00400000
PAWIN=0x4
PEACHW=0x04800004
PIXIE=0x0040
PLANIT=0x0004
PLANNER=0x2000
PLUS=0x1000
PM4=0xA000
PM5APP=0x8000
PP4=0x00000000
PR2=0x2000
PRINTHLP=0x0004
QAPLUSW=0x0004
QLIIFAX=0x00400000
QUAKE=0x80
QW=0x08000000
RELAY=0x20000000
REM=0x8022
RR2CD=0x00200000
RX=0x00000400
RXL=0x00000400
SETUP=0x00000000
SIDEKICK=0x0004
SLEEPER=0x10000000
SPCB=0x04008000
SPORTJEP=0x00200000
SPWIN20=0x00400000
ST2=0x4008022
STRAUSS=0x40000000
STRAV=0x40000000
SCHUBERT=0x40000000
SSBWIN=0x00200000
SWCWIN=0x00800004
TCVWIN=0x00200000
TCW=0x00400000
TCWIN=0x0004
TERRAIN=0x00400000
TISETUP=0x00200000
TL6=0x08000000
TME=0x0100
TMSWIN=0x20000000
TMTWIN=0x00200000
TMTWINCD=0x00200000
TOUCHUP=0x00400000
TURBOTAX=0x00080000
VB=0x0200
VEWINFIL=0x00400000
VISIO=0x00000004
VISIOHM=0x00000004
VISION=0x0040
W4GL=0x4000
W4GLR=0x4000
WGW=0x00440000
WIN2WRS=0x1210
WINCIM=0x4
WINLINK=0x20000000
WINPHONE=0x0004
WINSIM=0x2000
WINTACH=0x00200000
WORDSCAN=0x02200000
WPWINFIL=0x00000006
WPWIN60=0x00000400
WPWIN61=0x02000400
WSETUP=0x00200000
XPRESS=0x00000008
ZETA01=0x00400000
ZIFFBOOK=0x00200000
[Compatibility32]
CLWORKS=0x00A00000
MCAD=0x00600000
PHOTOSHP=0x00208000
PODW=0x00200000
SPSSWIN=0x00200000
TYPSTRY2=0x00200000
V32VM20=0x02000000
VISIO=0x00000000
VISIOHM=0x00000000
WINPHONE=0x00000004
WRDART32=0x00400000
[mci extensions]
mid=Sequencer
rmi=Sequencer
wav=waveaudio
```

```
avi=AVIVideo
[MCICompatibility]
QTWVideo=0x0001
MCIXSND=0x0001
GDAnim=0x0001
[mciavi]
[ModuleCompatibility]
ACEROOBE=0x0004
AIRNFM=0x0002
ALDNCD=0x0002
AMRES=0x0002
ATM=0x0002
ARCHANGEL=0x0002
CSNOV=0x0002
DEFDEMO=0x0002
DIBWND=0x0002
DIB=0x0002
DS=0x0001
EMLIB=0x0002
EMSAVE=0x0002
FH4=0x0002
GEDIT=0x0002
GEORGE=0x0002
GVBSETUP=0x0002
HRWCD=0x0002
ISLFAXPR=0x0002
KIDDESK=0x0002
KIDSTYPE=0x0000
KNPS=0x0002
LIONKING=0x0002
MAUI_DRV=0x0002
MGXWMF=0x0002
MEMMAP=0x0002
MSARTIST=0x0002
MSCRWRTR=0x0002
MSCUISTF=0x0001
MVIEWER2=0x0002
MWAVSCAN=0x0002
MYINV=0x0002
OLESVR=0x0002
PDOXWIN=0x0002
PLANIT=0x0002
PP3=0x0002
PP4=0x0002
PPPP=0x0002
PXDSRV2=0x0002
REVIEWRT=0x0002
ROULETTE=0x0002
RRIRJ=0x0002
RR1=0x0002
RR2CD=0x0002
STL_DLG=0x0002
TECO=0x0001
TER=0x0002
TLW0LOC=0x0002
TMSWIN=0x0002
USA=0x0002
VOICE=0x0002
WFXVIEW=0x0004
WINFORM=0x0002
WPWIN61=0x0002
[Pscript.Drv]
ATMWorkaround=1
[Extensions]
txt=notepad.exe ^.txt
bmp=C:\Progra~1\Access~1\mspaint.exe ^.bmp
pcx=C:\Progra~1\Access~1\mspaint.exe ^.pcx
[Ports]
LPT1:=
LPT2:=
LPT3:=
COM1:=9600,n,8,1,x
COM2:=9600,n,8,1,x
COM3:=9600,n,8,1,x
COM4:=9600,n,8,1,x
FILE:=
[embedding]
AVIFile=Video Clip,Video Clip,C:\WINDOWS\mplayer.exe /avi,picture
Package=Package,Package,packager.exe,picture
PBrush=Paintbrush Picture,Paintbrush
Picture,C:\Progra~1\Access~1\MSPAINT.
EXE,picture
Paint.Picture=Bitmap Image,Bitmap
Image,C:\Progra~1\Access~1\MSPAINT.EXE,p
icture
mplayer=Media Clip,Media Clip,C:\WINDOWS\mplayer.exe,picture
midfile=MIDI Sequence,MIDI Sequence,C:\WINDOWS\mplayer.exe
/mid,picture
Wordpad.Document.1=WordPad Document,WordPad
Document,C:\Progra~1\Access~1\
WORDPAD.EXE,picture
[Devices]
[PrinterPorts]
[Sounds]
SystemDefault=,
** End of file

------------------------------------------------------------------------------
** Dump of C:\DIRECT.OUT
------------------------------------------------------------------------------
[Groups]
Group1=C:\WINDOWS\PROGRAMS.GRP
Group2=C:\WINDOWS\ACCESSOR.GRP
Group3=C:\WINDOWS\DESKTOP.GRP
Group4=C:\WINDOWS\SYSTEMTO.GRP
Group5=C:\WINDOWS\DOCUMENT.GRP
Group6=C:\WINDOWS\MULTIMED.GRP
Group7=C:\WINDOWS\MAIN.GRP
[Settings]
Order= 1 2 3 4 5 6 7
** End of file
Critical Error Log
==================
Correctable Memory Error Log
============================

System Configuration Utility . .  Version 2.33

Extended Non-volatile Memory
-------- ------------ ------
Slot . . . . . .  0
  Slot Type  . . . . . .  Embedded
  Board ID . . . . . .  CPQ0579
  CFG File Extension
  Revision Level . . . . . .  2.20
Type Entry(s)  . . . . . .  MSD,FPYCTL
  IRQ Entry(s):
    IRQ  6, Not Shared, Edge Triggered
  DMA Channel(s):
    Channel 2, Not Shared
    Timing: Type B
    Transfer Size: 8-bit (byte)
  Port Range(s):
    03F0h - 03F3h, Not Shared
    03F4h - 03F7h, Not Shared

Type Entry(s)  . . . . . .  MSD,UNIT0,FPYDRV;TYP=4
Type Entry(s)  . . . . . .  MSD,UNIT1,FPYDRV;TYP=0
Type Entry(s)  . . . . . .  MSD
Type Entry(s)  . . . . . .  MSD,DSKCTL;DIS1
Type Entry(s)  . . . . . .  MSD,UNIT0,DSKDRV
Type Entry(s)  . . . . . .  MSD,UNIT1,DSKDRV
Type Entry(s)  . . . . . .  MSD,DSKCTL;DIS2
Memory Entry(s):
                                Range              Size
                                -----              ----
  ROM: Other,    Cacheable       896K -        1M         128K

Type Entry(s)  . . . . . .  MEM;COMPAQ
Memory Entry(s):
                                Range              Size
                                -----              ----
  RAM: System,   Cacheable         0K -       640K         640K

Type Entry(s)  . . . . . .  MEM;COMPAQ
Memory Entry(s):
                                Range              Size
                                -----              ----
  RAM: System,   Cacheable         1M -        16M          15M

Type Entry(s)  . . . . . .  MEM;COMPAQ
Memory Entry(s):
                                Range              Size
                                -----              ----
  RAM: System,   Cacheable        16M -        64M          48M
  RAM: System,   Cacheable        64M -        96M          32M

Type Entry(s)  . . . . . .  MEM;COMPAQ
Type Entry(s)  . . . . . .  MEM;COMPAQ
Type Entry(s)  . . . . . .  COM,ASY;COM1;A
  IRQ Entry(s):
    IRQ  4, Not Shared, Edge Triggered
  Port Range(s):
    03F8h - 03FFh, Not Shared

Type Entry(s)  . . . . . .  COM,ASY;COM2;B
  IRQ Entry(s):
    IRQ  3, Not Shared, Edge Triggered
  Port Range(s):
    02F8h - 02FFh, Not Shared

Type Entry(s)  . . . . . .  PAR;LPT1
  IRQ Entry(s):
    IRQ  7, Not Shared, Edge Triggered
  DMA Channel(s):
    Channel 0, Not Shared
```

```
    Timing: ISA Compatible
    Transfer Size: 8-bit (byte)
  Port Range(s):
    03BCh - 03BEh, Not Shared

 Type Entry(s) . . . . . PTR,8042
  IRQ Entry(s):
   IRQ 12, Not Shared, Edge Triggered
 Type Entry(s) . . . . . OTH,CPQCSM
 Free Form Text . . . . . 0D 01 01 01 00 00 00 00 50 0C 00 00
 Type Entry(s) . . . . . OTH,A20
 Type Entry(s) . . . . . OTH,SOFTNMI
 Type Entry(s) . . . . . OTH,FLSFNMI
 Type Entry(s) . . . . . OTH,BUSNMI
 Type Entry(s) . . . . . OTH,DSKTDMA
 Type Entry(s) . . . . . OTH,REFRESH
 Type Entry(s) . . . . . OTH,PERR
 Type Entry(s) . . . . . OTH,SIMMSPD;AUTO
 Type Entry(s) . . . . . OTH,TABLE;DEFAULT6
 Type Entry(s) . . . . . OTH,CURREV
 Free Form Text . . . . . F0 52 18 06 53 59 FF 00 01 06 52 FF FF
FF
                    FF
 Type Entry(s) . . . . . OTH,PREREV
 Free Form Text . . . . . EE 52 18 06 53 59 FF 00 FF 06 52 FF FF
FF
                    FF
 Type Entry(s) . . . . . OTH,CPR,NMI
 Free Form Text . . . . . 01 00 0A 00 58 0C C1 C5
 Memory Entry(s):
                      Range            Size
                      -----            ----
   RAM: Virtual,  Non-Cacheable   2060M -  2109441K          1K

   IRQ Entry(s):
    IRQ 13, Not Shared, Edge Triggered
 Type Entry(s) . . . . . ISA;MAP
 Free Form Text . . . . . 81 82 83 84 85 26 27 E0 E0 E0 E0 E0 E0
E0
                    E0 00 00 00 00 00 00 00 00 00 00 00 00 00
                    00 00
 Type Entry(s) . . . . . ISA;PCIMAP
 Free Form Text . . . . . 01 00 90 02 00 98 03 01 38 04 01 40 05 01
                    48

Revisions Table
===============

Current Revisions
 Date . . . . . 4/23/97
 System Board Revision . . . . 01
 Assembly Version . . . . . 1
 Functional Revision Level . . A
 Riser Card Revision . . . . . Not Supported

Previous Revisions
 Date . . . . . 4/23/97
 System Board Revision . . . . Not Supported
 Riser Card Revision . . . . . Not Supported

Memory Allocation (including INSPECT)
 PSP   SIZE    NAME         TRAPPED INTERRUPTS
 ----  ------  ------------ ----------------------------------------
 08D0  007200  COMMAND.COM   2Fh  2Eh  24h  23h  22h
 0A9B  207216  INSPECT.EXE   FBh  EFh  3Fh  00h

System Configuration Memory
 00 - 0F :   56 00 09 00   04 00 04 01   05 97 26 82   50 80 00 00
 10 - 1F :   40 00 00 00   03 80 02 00   3C 00 00 00   00 00 00 00
 20 - 2F :   00 00 00 00   7F 20 04 40   00 82 00 00   00 10 02 76
 30 - 3F :   00 3C 19 80   00 00 XX XX   XX XX XX XX   XX XX XX
XX

BIOS Data Area
 40:0000 :   F8 03 F8 02   00 00 00 00   BC 03 00 00   00 00 06 02
 40:0010 :   27 44 00 80   02 00 00 00   00 00 1E 00   1E 00 00 00
 40:0020 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
 40:0030 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 01 01
 40:0040 :   1F 00 00 00   00 29 00 10   02 03 50 00   00 10 00 00
 40:0050 :   00 14 00 00   00 00 00 00   00 00 00 00   00 00 00 00
 40:0060 :   0E 0D 00 D4   03 29 30 C2   11 45 77 00   72 2A 04 00
 40:0070 :   00 00 00 12   00 01 00 00   14 14 14 14   01 01 01 01
 40:0080 :   1E 00 3E 00   18 10 00 60   F9 11 0B 01   00 00 00 05
 40:0090 :   17 00 00 00   29 00 10 00   00 00 00 00   00 00 00 00
 40:00A0 :   00 00 00 00   00 00 00 00   57 5B 00 C0   00 00 00 00
 40:00B0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
 40:00C0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
 40:00D0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
 40:00E0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
 40:00F0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00


Interrupt Vector Table (including INSPECT)
 00 - 03 :   0AAB:0555     0070:0465     0807:0016     0070:0465
 04 - 07 :   0070:0465     F000:FF54     F000:93CC     F000:9BD0
 08 - 0B :   0807:001F     0807:0028     F000:9BD0     F000:9BD0
 0C - 0F :   F000:9BD0     F000:9BD0     0807:009A     0070:0465
 10 - 13 :   C000:329C     F000:F84D     F000:F841     0070:03EE
 14 - 17 :   F000:E739     0247:0240     0070:042D     F000:EFD2
 18 - 1B :   F000:5167     08C8:002F     F000:FE6E     0070:045F
 1C - 1F :   F000:FF53     C000:1F24     0000:0522     C000:6747
 20 - 23 :   00C9:0FA8     00C9:0FB2     08D0:0314     08D0:016D
 24 - 27 :   08D0:0178     00C9:0FBC     00C9:0FC6     00C9:0FD0
 28 - 2B :   00C9:106F     0070:0466     00C9:106F     00C9:106F
 2C - 2F :   00C9:106F     00C9:106F     08D0:0162     08D1:01CC
 30 - 33 :   C90F:E4EA     F000:9B00     00C9:106F     00C9:106F
 34 - 37 :   00C9:106F     00C9:106F     00C9:106F     00C9:106F
 38 - 3B :   00C9:106F     00C9:106F     00C9:106F     00C9:106F
 3C - 3F :   00C9:106F     00C9:106F     00C9:106F     1B6A:04FD
 40 - 43 :   F000:EC59     0000:0000     F000:F065     C000:6347
 44 - 47 :   F000:9BD0     F000:9BD0     0000:0000     F000:9BD0
 48 - 4B :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
 4C - 4F :   F000:9BD0     F000:9BD0     F000:9BD0     0070:04FC
 50 - 53 :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
 54 - 57 :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
 58 - 5B :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
 5C - 5F :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
 60 - 63 :   0000:0000     0000:0000     0000:0000     0000:0000
 64 - 67 :   0000:0000     0000:0000     0000:0000     0000:0000
 68 - 6B :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
 6C - 6F :   F000:9BD0     C000:329C     F000:9BD0     F000:9BD0
 70 - 73 :   0807:0035     F000:9C1F     F000:9BD0     0807:00CA
 74 - 77 :   0807:00E2     F000:9C28     0807:00FA     F000:9BD0
 78 - 7B :   0000:0000     0000:0000     0000:0000     0000:0000
 7C - 7F :   0000:0000     0000:0000     0000:0000     0000:0000
 80 - 83 :   0000:0000     0000:0000     0000:0000     0000:0000
 84 - 87 :   0000:0000     0000:0000     0000:0000     0000:0000
 88 - 8B :   0000:0000     0000:0000     0000:0000     0000:0000
 8C - 8F :   0000:0000     0000:0000     0000:0000     0000:0000
 90 - 93 :   0000:0000     0000:0000     0000:0000     0000:0000
 94 - 97 :   0000:0000     0000:0000     0000:0000     0000:0000
 98 - 9B :   0000:0000     0000:0000     0000:0000     0000:0000
 9C - 9F :   0000:0000     0000:0000     0000:0000     0000:0000
 A0 - A3 :   0000:0000     0000:0000     0000:0000     0000:0000
 A4 - A7 :   0000:0000     0000:0000     0000:0000     0000:0000
 A8 - AB :   0000:0000     0000:0000     0000:0000     0000:0000
 AC - AF :   0000:0000     0000:0000     0000:0000     0000:0000
 B0 - B3 :   0000:0000     0000:0000     0000:0000     0000:0000
 B4 - B7 :   0000:0000     0000:0000     0000:0000     0000:0000
 B8 - BB :   0000:0000     0000:0000     0000:0000     0000:0000
 BC - BF :   0000:0000     0000:0000     0000:0000     0000:0000
 C0 - C3 :   0000:0300     0000:1200     0000:0000     0000:0000
 C4 - C7 :   0000:0000     0000:0000     0000:0000     0000:0000
 C8 - CB :   0000:0000     0000:0000     0000:0000     0000:0000
 CC - CF :   0000:0000     0000:0000     0000:0000     0000:0000
 D0 - D3 :   0000:0000     0000:0000     0000:0000     0000:0000
 D4 - D7 :   0000:0000     0000:0000     0000:0000     0000:0000
 D8 - DB :   0000:0000     0000:0000     0000:0000     0000:0000
 DC - DF :   0000:0000     0000:0000     0000:0000     0000:0000
 E0 - E3 :   0000:0000     0000:0000     0000:0000     72D4:EE00
 E4 - E7 :   F886:0040     F886:0006     EF6C:0006     F886:F886
 E8 - EB :   F886:0006     EF6C:0006     7086:F000     0000:0040
 EC - EF :   F000:26CE     0020:0203     0000:036B     2003:10DE
 F0 - F3 :   0008:7202     7C77:C000     03DA:0E06     568B:03CE
 F4 - F7 :   0008:6C77     03C4:0203     03C4:0000     0000:0ED0
 F8 - FB :   0000:03C4     FF10:020F     0000:0000     00F4:D9B8
 FC - FF :   0000:59B3     0000:0A6B     984D:0003     7246:F000

PCI Devices Information
 Signature . . . . . PCI
 Config Mechanism #1 . . . . . Supported
 Config Mechanism #2 . . . . . Not Supported
 Spec Cycle for Config #1 . . . Supported
 Spec Cycle for Config #2 . . . Not Supported
 BIOS Interface Version . . . . 2.10
 Last PCI Bus Number . . . . . 1
 Number of PCI Devices . . . . 6

 Bus Number . . . . . 0
 Device Number . . . . . 10
 Function Number . . . . . 00h
 Slot Number . . . . . 1
 Vendor ID . . . . . 0E11h
 Device ID . . . . . AE32h
 Revision ID . . . . . 10h
 Device Type . . . . . Other Network Controller
 Programming Interface . . . . 00h
 Expansion ROM Base Address . . FFFF0000h
 IRQ Line . . . . . 5
 IRQ Pin . . . . . INTA#
 IO Address Base . . . . . 6000h
 IO Address Length . . . . . 10h
 Memory Address Base . . . . . 40000000h
 Memory Address Length . . . . 10h

 Bus Number . . . . . 0
```

Device Number 11
Function Number 00h
Slot Number 2
Vendor ID 0E11h
Device ID AE32h
Revision ID 10h
Device Type . . . . . Other Network Controller
Programming Interface 00h
Expansion ROM Base Address . . FFFF0000h
IRQ Line 9
IRQ Pin . . . . . INTA#
IO Address Base 6010h
IO Address Length 10h
Memory Address Base 40000010h
Memory Address Length 10h

Bus Number 0
Device Number 16
Function Number 00h
Slot Number 0
Vendor ID 0E11h
Device ID AE35h
Revision ID 10h
Device Type . . . . . Other Network Controller
Programming Interface 00h
Expansion ROM Base Address . . FFFF0000h
IRQ Line 10
IRQ Pin . . . . . INTA#
IO Address Base 6020h
IO Address Length 10h
Memory Address Base 40000020h
Memory Address Length 10h

Bus Number 0
Device Number 20
Function Number 01h
Slot Number 0
Vendor ID 8086h
Device ID 7010h
Revision ID 00h
Device Type . . . . . IDE Controller
Programming Interface 80h
Expansion ROM Base Address . . 0h
IRQ Line 0
IRQ Pin . . . . . Not Used
IO Address Base 1000h
IO Address Length 10h

Bus Number 1
Device Number 4
Function Number 00h
Slot Number 0
Vendor ID 1000h
Device ID 000Fh
Revision ID 03h
Device Type . . . . . SCSI Bus Controller
Programming Interface 00h
Expansion ROM Base Address . . 0h
IRQ Line 11
IRQ Pin . . . . . INTA#
IO Address Base 7000h
IO Address Length 100h
Memory Address Base 40101000h
Memory Address Length 100h
Memory Address Base 40100000h
Memory Address Length 1000h

Bus Number 1
Device Number 9
Function Number 00h
Slot Number 5
Vendor ID 1013h
Device ID 00A0h
Revision ID 47h
Device Type . . . . . VGA Compatible Controller
Programming Interface 00h
Expansion ROM Base Address . . FF000000h
IRQ Line 9
IRQ Pin . . . . . INTA#
Memory Address Base 41000000h
Memory Address Length 1000000h

Compaq ProLiant 800 is a trademark of Compaq Computer Corporation.

SUT SYSTEM CONFIGURATION UTILITY

˝

Date 4/21/97
Time 14:59:46

Product . . . . . Compaq ProLiant 2500

Machine ID
From System Board CPQ0551

Processor ID 5F41

Processor . . . . . Pentium Pro(R) at 200 MHz
Secondary Cache 512K
CPU ID 0617

Processor 2 . . . . . Pentium Pro(R) at 200 MHz
Secondary Cache 512K
CPU ID 0617

Processor(s) Mapped Out . . . . None

Numeric Coprocessor . . . . . Integrated 387-Compatible

Expansion Bus . . . . . Extended ISA, PCI

System Identification Number . . D649HWA30213

CPU Mode . . . . . Real Mode

System ROM
Revision 12/30/1996
Family E24
Flashable . . . . . Yes
Supports F10 partition . . . Yes

Video Controller ROM
Revision . . . . . 1.6 (Cirrus)

Keyboard Controller ROM
Revision C.7 06/14/91
Family . . . . . K

Option ROMs
Address Range . . . . . C0000 - C7FFF
Data Dump . . . . . (CL-GD5440 VGA BIOS Version 1.06
Copyright 199...)

Address Range . . . . . C8000 - CBFFF
Data Dump . . . . . (04/11/96 SMART-2 Option ROM/BIOS
(C)Copyri...)

Address Range . . . . . E8000 - EDFFF
Data Dump . . . . . (11/14/96 (C)Copyright COMPAQ Computer
Corporation ...)

Bootblock ROM 07/04/96

Standby Recovery Server
Status . . . . . Disabled
COM Port COM1
Server Configuration . . . . . Recovery
Timeout Value . . . . . 1 minutes

Memory Boards Identified:
Processor Board
DIMM Slot 1 (EDO) . . . . . 256 Megabytes
DIMM Slot 2 (EDO) . . . . . 256 Megabytes
DIMM Slot 3 (EDO) . . . . . 256 Megabytes
DIMM Slot 4 (EDO) . . . . . 256 Megabytes
Total Compaq Memory . . . . . 1024 Megabytes

Keyboard . . . . . Enhanced

LPT Ports . . . . . Not Installed

COM Ports . . . . . Not Installed

Diskette Drive A . . . . . 1.44 Megabyte (3.5 inch)

Drive Controller 1, 32-Bit Compaq SMART-2/P Rev. B Array Controller
IDA Firmware Revision 1.62
Array Accelerator Memory . . . 4096 Kbytes
Reserved for writes. . . . . 4096 Kbytes
Accelerator Status . . . . . Enabled
Battery count 3
Batteries charged 3

```
  Batteries failed . . . . . 0
ProLiant . . . . . Bus 1 (internal/external), Rev. JM14
ProLiant . . . . . Bus 2 (external), Rev. JM14

Logical Drive  1 . . . . . 12588 Megabyte
Fault Tolerance  . . . . . None
OS Format  . . . . . Multi-Sector Distribution
Drive geometry (Cyl, Hds, Sec)  3013, 255, 32
Array Accelerator  . . . . . Enabled

  Hard Drive  1
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 0
  Serial Number  . . . . . 00267878
  Firmware Revision 1  . . . . 6213
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 208477
    Sectors read . . . . . *1768864450
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 110612385
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 465584
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 49
    Spin up time . . . . . 0
    Seek time track  . . . . . 75%
    Seek time third  . . . . . 75%
    Seek time full . . . . . 75%
    Reallocated sectors  . . . 142
    Recovers read failed . . . 1
    Bus faults . . . . . 76

  Hard Drive  2
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 1
  Serial Number  . . . . . 00444304
  Firmware Revision 1  . . . . 6213
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 218111
    Sectors read . . . . . *930636092
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 122566781
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 574099
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 35
    Spin up time . . . . . 0
    Seek time track  . . . . . 75%
    Seek time third  . . . . . 76%
    Seek time full . . . . . 75%
    Reallocated sectors  . . . 383
    Recovers read failed . . . 2
    Bus faults . . . . . 70

  Hard Drive  3
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 2
  Serial Number  . . . . . 00437956
  Firmware Revision 1  . . . . 6213
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 218118
    Sectors read . . . . . *931167511
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 122014606
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 580779
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 38
    Spin up time . . . . . 177
    Seek time track  . . . . . 78%
    Seek time third  . . . . . 76%
    Seek time full . . . . . 76%
    Reallocated sectors  . . . 372
    Recovers read failed . . . 3
    Bus faults . . . . . 102

  Hard Drive  4
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 3
  Serial Number  . . . . . 00444790
  Firmware Revision 1  . . . . 6213
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 172991
    Sectors read . . . . . *1683401348
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 88969975
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 203104
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 12
    Spin up time . . . . . 0
    Seek time track  . . . . . 0%
    Seek time third  . . . . . 0%
    Seek time full . . . . . 0%
    Reallocation sectors . . . 4294967295
    Reallocated sectors  . . . 53
    DRQ timeouts . . . . . 65535
    Recovers read failed . . . 2
    IRQ deglitch count . . . . 4294967295
    Bus faults . . . . . 64

  Hard Drive  5
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 4
  Serial Number  . . . . . 00243472
  Firmware Revision 1  . . . . 6213
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 201334
    Sectors read . . . . . *924121097
    Hard read errors . . . . . 0
    Read errors retry  . . . . 2
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 104953082
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 517054
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 33
    Spin up time . . . . . 0
    Seek time track  . . . . . 78%
    Seek time third  . . . . . 74%
    Seek time full . . . . . 75%
    Reallocated sectors  . . . 32
    Recovers read failed . . . 1
    Bus faults . . . . . 74

  Hard Drive  6
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 5
  Serial Number  . . . . . 02256925
  Firmware Revision 1  . . . . 6215
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 167816
    Sectors read . . . . . *926761173
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 96105299
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 84832
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 7
    Spin up time . . . . . 182
    Seek time track  . . . . . 75%
    Seek time third  . . . . . 75%
    Seek time full . . . . . 74%
    Reallocated sectors  . . . 74
    Recovers read failed . . . 1
    Bus faults . . . . . 60

  Hard Drive  7
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 6
  Serial Number  . . . . . 02262839
  Firmware Revision 1  . . . . 6215
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 167816
    Sectors read . . . . . *925078787
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 95602812
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 85712
    Seek errors  . . . . . 0
```

```
  Spin cycles . . . . . 7
  Spin up time . . . . . 186
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 134
  Recovers read failed . . . 1
  Bus faults . . . . . 58

Hard Drive  8
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 0
Serial Number . . . . . 00590376
Firmware Revision 1 . . . . 6213
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 178179
  Sectors read . . . . . *981792002
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 105630236
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 332212
  Seek errors . . . . . 0
  Spin cycles . . . . . 30
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 75%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 207
  Recovers read failed . . . 0
  Bus faults . . . . . 70

Hard Drive  9
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 1
Serial Number . . . . . 00576836
Firmware Revision 1 . . . . 6213
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 178118
  Sectors read . . . . . *980838198
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 104283902
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 326480
  Seek errors . . . . . 0
  Spin cycles . . . . . 29
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 75%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 97
  Recovers read failed . . . 0
  Bus faults . . . . . 68

Hard Drive 10
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 2
Serial Number . . . . . 00725149
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 172118
  Sectors read . . . . . *1041033171
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 93701279
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 336019
  Seek errors . . . . . 0
  Spin cycles . . . . . 20
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 75%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 356
  Recovers read failed . . . 0
  Bus faults . . . . . 68

Hard Drive 11
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 3
Serial Number . . . . . 00560929
Firmware Revision 1 . . . . 6213
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 189755
  Sectors read . . . . . *1044311069
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 94474683
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 330513
  Seek errors . . . . . 0
  Spin cycles . . . . . 24
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 76%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 548
  Recovers read failed . . . 0
  Bus faults . . . . . 68

Hard Drive 12
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 4
Serial Number . . . . . 01178888
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 190486
  Sectors read . . . . . *1345424530
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 367962299
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 278960
  Seek errors . . . . . 0
  Spin cycles . . . . . 182
  Spin up time . . . . . 181
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 147
  Recovers read failed . . . 0
  Bus faults . . . . . 130

Hard Drive 13
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 5
Serial Number . . . . . 00459185
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 168313
  Sectors read . . . . . *902406808
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 84942528
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 106128
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 0
  Seek time track . . . . . 0%
  Seek time third . . . . . 0%
  Seek time full . . . . . 0%
  Reallocation sectors . . . 4294967295
  Reallocated sectors . . . 842
  DRQ timeouts . . . . . 65535
  Recovers read failed . . . 0
  IRQ deglitch count . . . . 4294967295
  Bus faults . . . . . 64

Hard Drive 14
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 6
Serial Number . . . . . 00781287
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 168451
  Sectors read . . . . . *915759653
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 88225696
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 106480
  Seek errors . . . . . 0
  Spin cycles . . . . . 8
```

```
   Spin up time . . . . . 190
   Seek time track . . . . . 75%
   Seek time third . . . . . 74%
   Seek time full . . . . . 75%
   Reallocated sectors . . . 137
   Recovers read failed . . . 0
   Bus faults . . . . . 66

Logical Drive 2 . . . . . 44113 Megabyte
Fault Tolerance . . . . . Distributed Data Guarding
OS Format . . . . . Multi-Sector Distribution
Drive geometry (Cyl, Hds, Sec) 10559, 255, 32
Array Accelerator . . . . . Enabled

  Hard Drive 1
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 0
  Serial Number . . . . . 00267878
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 208478
    Sectors read . . . . . *1768864450
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 110612385
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 465760
    Seek errors . . . . . 0
    Spin cycles . . . . . 49
    Spin up time . . . . . 0
    Seek time track . . . . . 75%
    Seek time third . . . . . 75%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 142
    Recovers read failed . . . 1
    Bus faults . . . . . 76

  Hard Drive 2
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 1
  Serial Number . . . . . 00444304
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 218112
    Sectors read . . . . . *930636092
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 122566781
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 574275
    Seek errors . . . . . 0
    Spin cycles . . . . . 35
    Spin up time . . . . . 0
    Seek time track . . . . . 75%
    Seek time third . . . . . 76%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 383
    Recovers read failed . . . 2
    Bus faults . . . . . 70

  Hard Drive 3
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 2
  Serial Number . . . . . 00437956
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 218119
    Sectors read . . . . . *931167511
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 122014606
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 580955
    Seek errors . . . . . 0
    Spin cycles . . . . . 38
    Spin up time . . . . . 177
    Seek time track . . . . . 78%
    Seek time third . . . . . 76%
    Seek time full . . . . . 76%
    Reallocated sectors . . . 372
    Recovers read failed . . . 3
    Bus faults . . . . . 102

  Hard Drive 4
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 3
  Serial Number . . . . . 00444790
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 172992
    Sectors read . . . . . *1683401348
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 88969975
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 203280
    Seek errors . . . . . 0
    Spin cycles . . . . . 12
    Spin up time . . . . . 0
    Seek time track . . . . . 0%
    Seek time third . . . . . 0%
    Seek time full . . . . . 0%
    Reallocation sectors . . . 4294967295
    Reallocated sectors . . . 53
    DRQ timeouts . . . . . 65535
    Recovers read failed . . . 2
    IRQ deglitch count . . . . 4294967295
    Bus faults . . . . . 64

  Hard Drive 5
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 4
  Serial Number . . . . . 00243472
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 201335
    Sectors read . . . . . *924121097
    Hard read errors . . . . . 0
    Read errors retry . . . . 2
    ECC read errors . . . . . 0
    Sectors written . . . . . 104953082
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 517230
    Seek errors . . . . . 0
    Spin cycles . . . . . 33
    Spin up time . . . . . 0
    Seek time track . . . . . 78%
    Seek time third . . . . . 74%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 32
    Recovers read failed . . . 1
    Bus faults . . . . . 74

  Hard Drive 6
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 5
  Serial Number . . . . . 02256925
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 167817
    Sectors read . . . . . *926761173
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 96105299
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 85008
    Seek errors . . . . . 0
    Spin cycles . . . . . 7
    Spin up time . . . . . 182
    Seek time track . . . . . 75%
    Seek time third . . . . . 75%
    Seek time full . . . . . 74%
    Reallocated sectors . . . 74
    Recovers read failed . . . 1
    Bus faults . . . . . 60

  Hard Drive 7
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 6
  Serial Number . . . . . 02262839
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 167817
    Sectors read . . . . . *925078787
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 95602812
```

Hard write errors 0
Write errors retry 0
Seek count 85888
Seek errors 0
Spin cycles 7
Spin up time 186
Seek time track 75%
Seek time third 74%
Seek time full 75%
Reallocated sectors ... 134
Recovers read failed ... 1
Bus faults 58

Hard Drive 8
SCSI Bus ..... 2 (external)
SCSI ID 0
Serial Number 00590376
Firmware Revision 1 6213
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 178180
Sectors read *981792002
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 105630236
Hard write errors 0
Write errors retry 0
Seek count 332388
Seek errors 0
Spin cycles 30
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors ... 207
Recovers read failed ... 0
Bus faults 70

Hard Drive 9
SCSI Bus ..... 2 (external)
SCSI ID 1
Serial Number 00576836
Firmware Revision 1 6213
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 178119
Sectors read *980838198
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 104283902
Hard write errors 0
Write errors retry 0
Seek count 326656
Seek errors 0
Spin cycles 29
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors ... 97
Recovers read failed ... 0
Bus faults 68

Hard Drive 10
SCSI Bus ..... 2 (external)
SCSI ID 2
Serial Number 00725149
Firmware Revision 1 6215
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 172119
Sectors read *1041033171
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 93701279
Hard write errors 0
Write errors retry 0
Seek count 336195
Seek errors 0
Spin cycles 20
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors ... 356
Recovers read failed ... 0
Bus faults 68

Hard Drive 11
SCSI Bus ..... 2 (external)
SCSI ID 3
Serial Number 00560929
Firmware Revision 1 6213
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 189756
Sectors read *1044311069
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 94474683
Hard write errors 0
Write errors retry 0
Seek count 330689
Seek errors 0
Spin cycles 24
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 75%
Reallocated sectors ... 548
Recovers read failed ... 0
Bus faults 68

Hard Drive 12
SCSI Bus ..... 2 (external)
SCSI ID 4
Serial Number 01178888
Firmware Revision 1 6215
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 190487
Sectors read *1345424530
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 367962299
Hard write errors 0
Write errors retry 0
Seek count 279136
Seek errors 0
Spin cycles 182
Spin up time 181
Seek time track 75%
Seek time third 74%
Seek time full 75%
Reallocated sectors ... 147
Recovers read failed ... 0
Bus faults 130

Hard Drive 13
SCSI Bus ..... 2 (external)
SCSI ID 5
Serial Number 00459185
Firmware Revision 1 6215
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 168313
Sectors read *902406808
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 84942528
Hard write errors 0
Write errors retry 0
Seek count 106304
Seek errors 0
Spin cycles 6
Spin up time 0
Seek time track 0%
Seek time third 0%
Seek time full 0%
Reallocation sectors ... 4294967295
Reallocated sectors ... 842
DRQ timeouts 65535
Recovers read failed ... 0
IRQ deglitch count 4294967295
Bus faults 64

Hard Drive 14
SCSI Bus ..... 2 (external)
SCSI ID 6
Serial Number 00781287
Firmware Revision 1 6215
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 168451
Sectors read *915759653
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 88225696
Hard write errors 0

```
    Write errors retry . . . . 0
    Seek count . . . . . 106656
    Seek errors . . . . . 0
    Spin cycles . . . . . 8
    Spin up time . . . . . 190
    Seek time track . . . . . 75%
    Seek time third . . . . . 74%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 137
    Recovers read failed . . . 0
    Bus faults . . . . . 66

Drive Controller 2, 32-Bit Compaq SMART-2/P Rev. B Array Controller
  IDA Firmware Revision . . . . 1.62
  Array Accelerator Memory . . . 4096 Kbytes
    Reserved for writes. . . . . 4096 Kbytes
  Accelerator Status . . . . . Enabled
    Battery count . . . . . 3
    Batteries charged . . . . . 3
    Batteries failed . . . . . 0
  ProLiant . . . . . Bus 1 (internal/external), Rev. JM14
  ProLiant . . . . . Bus 2 (external), Rev. JM14

  Logical Drive 1 . . . . . 12588 Megabyte
  Fault Tolerance . . . . . None
  OS Format . . . . . Multi-Sector Distribution
  Drive geometry (Cyl, Hds, Sec) 3013, 255, 32
  Array Accelerator . . . . . Enabled

    Hard Drive 1
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 0
    Serial Number . . . . . 02230391
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 92413
      Sectors read . . . . . *1298464449
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 97360962
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 70925
      Seek errors . . . . . 0
      Spin cycles . . . . . 7
      Spin up time . . . . . 182
      Seek time track . . . . . 78%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 123
      Recovers read failed . . . 0
      Bus faults . . . . . 180

    Hard Drive 2
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 1
    Serial Number . . . . . 00647339
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 100892
      Sectors read . . . . . *1345512644
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 150408377
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 265584
      Seek errors . . . . . 0
      Spin cycles . . . . . 28
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 493
      Recovers read failed . . . 0
      Bus faults . . . . . 188

    Hard Drive 3
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 2
    Serial Number . . . . . 02065633
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 92452
      Sectors read . . . . . 3728171767
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 89188968
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 76912
      Seek errors . . . . . 0
      Spin cycles . . . . . 7
      Spin up time . . . . . 182
      Seek time track . . . . . 75%
      Seek time third . . . . . 74%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 562
      Recovers read failed . . . 0
      Bus faults . . . . . 180

    Hard Drive 4
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 3
    Serial Number . . . . . 00707490
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 101013
      Sectors read . . . . . 3777447840
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 143343798
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 273152
      Seek errors . . . . . 0
      Spin cycles . . . . . 29
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 76%
      Seek time full . . . . . 76%
      Reallocated sectors . . . 292
      Recovers read failed . . . 0
      Bus faults . . . . . 188

    Hard Drive 5
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 4
    Serial Number . . . . . 02230749
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 92414
      Sectors read . . . . . 3731828435
      Hard read errors . . . . . 0
      Read errors retry . . . . 4
      ECC read errors . . . . . 0
      Sectors written . . . . . 92685110
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 69520
      Seek errors . . . . . 0
      Spin cycles . . . . . 7
      Spin up time . . . . . 183
      Seek time track . . . . . 75%
      Seek time third . . . . . 74%
      Seek time full . . . . . 74%
      Reallocated sectors . . . 243
      Recovers read failed . . . 0
      Bus faults . . . . . 180

    Hard Drive 6
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 5
    Serial Number . . . . . 00652233
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 100952
      Sectors read . . . . . 3764026689
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 142578847
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 276496
      Seek errors . . . . . 0
      Spin cycles . . . . . 30
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 521
      Recovers read failed . . . 0
      Bus faults . . . . . 188

    Hard Drive 7
```

SCSI Bus . . . . . 1 (internal/external)
SCSI ID 6
Serial Number 00706117
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 101024
Sectors read 3764173085
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 142949282
Hard write errors 0
Write errors retry 0
Seek count 266992
Seek errors 0
Spin cycles 30
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 76%
Reallocated sectors . . . 124
Recovers read failed . . . 0
Bus faults 188

Hard Drive 8
SCSI Bus . . . . . 2 (external)
SCSI ID 0
Serial Number 02231593
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 92414
Sectors read 3728844089
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 89482065
Hard write errors 0
Write errors retry 0
Seek count 70928
Seek errors 0
Spin cycles 7
Spin up time 185
Seek time track 78%
Seek time third 74%
Seek time full 75%
Reallocated sectors . . . 111
Recovers read failed . . . 0
Bus faults 178

Hard Drive 9
SCSI Bus . . . . . 2 (external)
SCSI ID 1
Serial Number 02263494
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 92414
Sectors read 3728852571
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 89361616
Hard write errors 0
Write errors retry 0
Seek count 70224
Seek errors 0
Spin cycles 7
Spin up time 184
Seek time track 75%
Seek time third 74%
Seek time full 75%
Reallocated sectors . . . 146
Recovers read failed . . . 0
Bus faults 180

Hard Drive 10
SCSI Bus . . . . . 2 (external)
SCSI ID 2
Serial Number 00229901
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 100652
Sectors read 3771222671
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 145292124
Hard write errors 0
Write errors retry 0
Seek count 273856
Seek errors 0
Spin cycles 35
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors . . . 237
Recovers read failed . . . 0
Bus faults 196

Hard Drive 11
SCSI Bus . . . . . 2 (external)
SCSI ID 3
Serial Number 02210748
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 92415
Sectors read 3728765235
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 89290536
Hard write errors 0
Write errors retry 0
Seek count 72512
Seek errors 0
Spin cycles 7
Spin up time 180
Seek time track 75%
Seek time third 74%
Seek time full 75%
Reallocated sectors . . . 91
Recovers read failed . . . 0
Bus faults 180

Hard Drive 12
SCSI Bus . . . . . 2 (external)
SCSI ID 4
Serial Number 02264478
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 86859
Sectors read 3726863290
Hard read errors 0
Read errors retry 10
ECC read errors 0
Sectors written 85282338
Hard write errors 0
Write errors retry 0
Seek count 64240
Seek errors 0
Spin cycles 9
Spin up time 190
Seek time track 75%
Seek time third 74%
Seek time full 74%
Reallocated sectors . . . 246
Recovers read failed . . . 0
Bus faults 180

Hard Drive 13
SCSI Bus . . . . . 2 (external)
SCSI ID 5
Serial Number 00710324
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 101020
Sectors read *1333877864
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 151000371
Hard write errors 0
Write errors retry 0
Seek count 266464
Seek errors 0
Spin cycles 28
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors . . . 46
Recovers read failed . . . 0
Bus faults 188

Hard Drive 14
SCSI Bus . . . . . 2 (external)
SCSI ID 6
Serial Number 02246913
Firmware Revision 1 6215

```
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 92629
    Sectors read . . . . . *1296340357
    Hard read errors . . . . . 0
    Read errors retry  . . . . 1
    ECC read errors  . . . . . 1
    Sectors written  . . . . . 95614405
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 71280
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 6
    Spin up time . . . . . 180
    Seek time track  . . . . . 75%
    Seek time third  . . . . . 74%
    Seek time full . . . . . 74%
    Reallocated sectors  . . . 41
    Recovers read failed . . . 0
    Bus faults . . . . . 180

Logical Drive  2 . . . . . 44113 Megabyte
Fault Tolerance  . . . . . Distributed Data Guarding
OS Format  . . . . . Multi-Sector Distribution
Drive geometry (Cyl, Hds, Sec)  10559, 255, 32
Array Accelerator  . . . . . Enabled

  Hard Drive  1
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 0
  Serial Number  . . . . . 02230391
  Firmware Revision 1  . . . . 6215
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 92413
    Sectors read . . . . . *1298464449
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 97360962
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 71101
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 7
    Spin up time . . . . . 182
    Seek time track  . . . . . 78%
    Seek time third  . . . . . 75%
    Seek time full . . . . . 75%
    Reallocated sectors  . . . 123
    Recovers read failed . . . 0
    Bus faults . . . . . 180

  Hard Drive  2
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 1
  Serial Number  . . . . . 00647339
  Firmware Revision 1  . . . . 6215
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 100893
    Sectors read . . . . . *1345512644
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 150408377
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 265760
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 28
    Spin up time . . . . . 0
    Seek time track  . . . . . 78%
    Seek time third  . . . . . 75%
    Seek time full . . . . . 75%
    Reallocated sectors  . . . 493
    Recovers read failed . . . 0
    Bus faults . . . . . 188

  Hard Drive  3
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 2
  Serial Number  . . . . . 02065633
  Firmware Revision 1  . . . . 6215
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 92453
    Sectors read . . . . . 3728171767
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 89188968
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 77088
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 7
    Spin up time . . . . . 182
    Seek time track  . . . . . 75%
    Seek time third  . . . . . 74%
    Seek time full . . . . . 75%
    Reallocated sectors  . . . 562
    Recovers read failed . . . 0
    Bus faults . . . . . 180

  Hard Drive  4
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 3
  Serial Number  . . . . . 00707490
  Firmware Revision 1  . . . . 6215
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 101014
    Sectors read . . . . . 3777447840
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 143343798
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 273328
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 29
    Spin up time . . . . . 0
    Seek time track  . . . . . 78%
    Seek time third  . . . . . 76%
    Seek time full . . . . . 76%
    Reallocated sectors  . . . 292
    Recovers read failed . . . 0
    Bus faults . . . . . 188

  Hard Drive  5
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 4
  Serial Number  . . . . . 02230749
  Firmware Revision 1  . . . . 6215
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 92415
    Sectors read . . . . . 3731828435
    Hard read errors . . . . . 0
    Read errors retry  . . . . 4
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 92685110
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 69696
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 7
    Spin up time . . . . . 183
    Seek time track  . . . . . 75%
    Seek time third  . . . . . 74%
    Seek time full . . . . . 74%
    Reallocated sectors  . . . 243
    Recovers read failed . . . 0
    Bus faults . . . . . 180

  Hard Drive  6
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 5
  Serial Number  . . . . . 00652233
  Firmware Revision 1  . . . . 6215
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 100953
    Sectors read . . . . . 3764026689
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 142578847
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 276672
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 30
    Spin up time . . . . . 0
    Seek time track  . . . . . 78%
    Seek time third  . . . . . 75%
    Seek time full . . . . . 75%
    Reallocated sectors  . . . 521
    Recovers read failed . . . 0
    Bus faults . . . . . 188

  Hard Drive  7
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID  . . . . . 6
```

Serial Number 00706117
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 101025
Sectors read 3764173085
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 142949282
Hard write errors 0
Write errors retry 0
Seek count 267168
Seek errors 0
Spin cycles 30
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 76%
Reallocated sectors . . . 124
Recovers read failed . . . 0
Bus faults 188

Hard Drive 8
SCSI Bus . . . . . 2 (external)
SCSI ID 0
Serial Number 02231593
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 92415
Sectors read 3728844089
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 89482065
Hard write errors 0
Write errors retry 0
Seek count 71104
Seek errors 0
Spin cycles 7
Spin up time 185
Seek time track 78%
Seek time third 74%
Seek time full 75%
Reallocated sectors . . . 111
Recovers read failed . . . 0
Bus faults 178

Hard Drive 9
SCSI Bus . . . . . 2 (external)
SCSI ID 1
Serial Number 02263494
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 92415
Sectors read 3728852571
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 89361616
Hard write errors 0
Write errors retry 0
Seek count 70400
Seek errors 0
Spin cycles 7
Spin up time 184
Seek time track 75%
Seek time third 74%
Seek time full 75%
Reallocated sectors . . . 146
Recovers read failed . . . 0
Bus faults 180

Hard Drive 10
SCSI Bus . . . . . 2 (external)
SCSI ID 2
Serial Number 00229901
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 100653
Sectors read 3771222671
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 145292124
Hard write errors 0
Write errors retry 0
Seek count 274032
Seek errors 0
Spin cycles 35
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors . . . 237
Recovers read failed . . . 0
Bus faults 196

Hard Drive 11
SCSI Bus . . . . . 2 (external)
SCSI ID 3
Serial Number 02210748
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 92416
Sectors read 3728765235
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 89290536
Hard write errors 0
Write errors retry 0
Seek count 72688
Seek errors 0
Spin cycles 7
Spin up time 180
Seek time track 75%
Seek time third 74%
Seek time full 75%
Reallocated sectors . . . 91
Recovers read failed . . . 0
Bus faults 180

Hard Drive 12
SCSI Bus . . . . . 2 (external)
SCSI ID 4
Serial Number 02264478
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 86860
Sectors read 3726863290
Hard read errors 0
Read errors retry 10
ECC read errors 0
Sectors written 85282338
Hard write errors 0
Write errors retry 0
Seek count 64416
Seek errors 0
Spin cycles 9
Spin up time 190
Seek time track 75%
Seek time third 74%
Seek time full 74%
Reallocated sectors . . . 246
Recovers read failed . . . 0
Bus faults 180

Hard Drive 13
SCSI Bus . . . . . 2 (external)
SCSI ID 5
Serial Number 00710324
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 101021
Sectors read *1333877864
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 151000371
Hard write errors 0
Write errors retry 0
Seek count 266640
Seek errors 0
Spin cycles 28
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors . . . 46
Recovers read failed . . . 0
Bus faults 188

Hard Drive 14
SCSI Bus . . . . . 2 (external)
SCSI ID 6
Serial Number 02246913
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes

```
    Reference time . . . . . 92630
    Sectors read . . . . . *1296340357
    Hard read errors . . . . . 0
    Read errors retry . . . . 1
    ECC read errors . . . . . 1
    Sectors written . . . . . 95614405
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 71456
    Seek errors . . . . . 0
    Spin cycles . . . . . 6
    Spin up time . . . . . 180
    Seek time track . . . . . 75%
    Seek time third . . . . . 74%
    Seek time full . . . . . 74%
    Reallocated sectors . . . 41
    Recovers read failed . . . 0
    Bus faults . . . . . 180

Drive Controller 3, 32-Bit Compaq SMART-2/P Rev. B Array Controller
  IDA Firmware Revision . . . . 1.62
  Array Accelerator Memory . . . 4096 Kbytes
    Reserved for writes. . . . . 4096 Kbytes
  Accelerator Status . . . . . Enabled
    Battery count . . . . . 3
    Batteries charged . . . . . 3
    Batteries failed . . . . . 0
  ProLiant . . . . . Bus 1 (internal/external), Rev. JM14
  ProLiant . . . . . Bus 2 (external), Rev. JM14

  Logical Drive 1 . . . . . 12588 Megabyte
  Fault Tolerance . . . . . None
  OS Format . . . . . Multi-Sector Distribution
  Drive geometry (Cyl, Hds, Sec) 3013, 255, 32
  Array Accelerator . . . . . Enabled

    Hard Drive 1
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 0
    Serial Number . . . . . 00834843
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 97912
      Sectors read . . . . . 3925933515
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 93750316
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 216480
      Seek errors . . . . . 0
      Spin cycles . . . . . 46
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 88
      Recovers read failed . . . 0
      Bus faults . . . . . 182

    Hard Drive 2
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 1
    Serial Number . . . . . 00702794
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 102512
      Sectors read . . . . . 3716552758
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 114374496
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 269280
      Seek errors . . . . . 0
      Spin cycles . . . . . 32
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 74%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 59
      Recovers read failed . . . 0
      Bus faults . . . . . 188

    Hard Drive 3
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 2
    Serial Number . . . . . 01327924
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 93957
      Sectors read . . . . . 3693552815
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 87860129
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 130944
      Seek errors . . . . . 0
      Spin cycles . . . . . 12
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 83
      Recovers read failed . . . 0
      Bus faults . . . . . 180

    Hard Drive 4
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 3
    Serial Number . . . . . 01276385
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 93958
      Sectors read . . . . . 3694768374
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 88319628
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 133760
      Seek errors . . . . . 0
      Spin cycles . . . . . 12
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 76%
      Seek time full . . . . . 76%
      Reallocated sectors . . . 705
      Recovers read failed . . . 0
      Bus faults . . . . . 180

    Hard Drive 5
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 4
    Serial Number . . . . . 01322467
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 93957
      Sectors read . . . . . 3693638570
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 88036180
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 129712
      Seek errors . . . . . 0
      Spin cycles . . . . . 12
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 102
      Recovers read failed . . . 0
      Bus faults . . . . . 180

    Hard Drive 6
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 5
    Serial Number . . . . . 01290092
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 93957
      Sectors read . . . . . 3694421703
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 88249839
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 128656
      Seek errors . . . . . 0
      Spin cycles . . . . . 12
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
```

```
  Seek time third . . . . . 76%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 262
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive  7
SCSI Bus . . . . . 1 (internal/external)
SCSI ID . . . . . 6
Serial Number . . . . . 01308086
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 93958
  Sectors read . . . . . 3693204269
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 87764378
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 128306
  Seek errors . . . . . 1
  Spin cycles . . . . . 12
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 76%
  Seek time full . . . . . 73%
  Reallocated sectors . . . 197
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive  8
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 0
Serial Number . . . . . 02256414
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94104
  Sectors read . . . . . 3707009169
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 98564807
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 85184
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 185
  Seek time track . . . . . 78%
  Seek time third . . . . . 74%
  Seek time full . . . . . 74%
  Reallocated sectors . . . 258
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive  9
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 1
Serial Number . . . . . 02231164
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94200
  Sectors read . . . . . 3704582296
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 96585540
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 85184
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 177
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 58
  Recovers read failed . . . 0
  Bus faults . . . . . 182

Hard Drive 10
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 2
Serial Number . . . . . 02292040
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94069
  Sectors read . . . . . 3697610102
  Hard read errors . . . . . 0
  Read errors retry . . . . 8
  ECC read errors . . . . . 0
  Sectors written . . . . . 98081973
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 95920
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 183
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 74%
  Reallocated sectors . . . 265
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive 11
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 3
Serial Number . . . . . 02262416
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94167
  Sectors read . . . . . 3705749743
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 98063815
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 91696
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 184
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 178
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive 12
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 4
Serial Number . . . . . 02176696
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94106
  Sectors read . . . . . 3706656267
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 98260185
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 88352
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 177
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 162
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive 13
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 5
Serial Number . . . . . 02279843
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94068
  Sectors read . . . . . 3697038554
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 97692812
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 89232
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 184
  Seek time track . . . . . 75%
  Seek time third . . . . . 73%
  Seek time full . . . . . 74%
  Reallocated sectors . . . 108
  Recovers read failed . . . 0
```

```
  Bus faults . . . . . 182

 Hard Drive 14
 SCSI Bus . . . . . 2 (external)
 SCSI ID . . . . . 6
 Serial Number . . . . . 00644373
 Firmware Revision 1 . . . . 6213
 Model Number . . . . . COMPAQ ST15150W
 Initialized for Monitoring . Yes
  Reference time . . . . . 144964
  Sectors read . . . . . *4170834235
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 92780331
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 349024
  Seek errors . . . . . 0
  Spin cycles . . . . . 22
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 76%
  Seek time full . . . . . 76%
  Reallocated sectors . . . 134
  Recovers read failed . . . 0
  Bus faults . . . . . 12

Logical Drive 2 . . . . . 44113 Megabyte
Fault Tolerance . . . . . Distributed Data Guarding
OS Format . . . . . Multi-Sector Distribution
Drive geometry (Cyl, Hds, Sec) 10559, 255, 32
Array Accelerator . . . . . Enabled

  Hard Drive 1
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 0
  Serial Number . . . . . 00834843
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 97912
   Sectors read . . . . . 3925933515
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 93750316
   Hard write errors . . . . 0
   Write errors retry . . . . 0
   Seek count . . . . . 216656
   Seek errors . . . . . 0
   Spin cycles . . . . . 46
   Spin up time . . . . . 0
   Seek time track . . . . . 78%
   Seek time third . . . . . 75%
   Seek time full . . . . . 75%
   Reallocated sectors . . . 88
   Recovers read failed . . . 0
   Bus faults . . . . . 182

  Hard Drive 2
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 1
  Serial Number . . . . . 00702794
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 102512
   Sectors read . . . . . 3716552758
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 114374496
   Hard write errors . . . . 0
   Write errors retry . . . . 0
   Seek count . . . . . 269456
   Seek errors . . . . . 0
   Spin cycles . . . . . 32
   Spin up time . . . . . 0
   Seek time track . . . . . 78%
   Seek time third . . . . . 74%
   Seek time full . . . . . 75%
   Reallocated sectors . . . 59
   Recovers read failed . . . 0
   Bus faults . . . . . 188

  Hard Drive 3
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 2
  Serial Number . . . . . 01327924
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 93957
   Sectors read . . . . . 3693552815
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 87860129
   Hard write errors . . . . 0
   Write errors retry . . . . 0
   Seek count . . . . . 131120
   Seek errors . . . . . 0
   Spin cycles . . . . . 12
   Spin up time . . . . . 0
   Seek time track . . . . . 78%
   Seek time third . . . . . 75%
   Seek time full . . . . . 75%
   Reallocated sectors . . . 83
   Recovers read failed . . . 0
   Bus faults . . . . . 180

  Hard Drive 4
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 3
  Serial Number . . . . . 01276385
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 93958
   Sectors read . . . . . 3694768374
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 88319628
   Hard write errors . . . . 0
   Write errors retry . . . . 0
   Seek count . . . . . 133936
   Seek errors . . . . . 0
   Spin cycles . . . . . 12
   Spin up time . . . . . 0
   Seek time track . . . . . 78%
   Seek time third . . . . . 76%
   Seek time full . . . . . 76%
   Reallocated sectors . . . 705
   Recovers read failed . . . 0
   Bus faults . . . . . 180

 Hard Drive 5
 SCSI Bus . . . . . 1 (internal/external)
 SCSI ID . . . . . 4
 Serial Number . . . . . 01322467
 Firmware Revision 1 . . . . 6215
 Model Number . . . . . COMPAQ ST15150W
 Initialized for Monitoring . Yes
  Reference time . . . . . 93957
  Sectors read . . . . . 3693638570
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 88036180
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 129888
  Seek errors . . . . . 0
  Spin cycles . . . . . 12
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 75%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 102
  Recovers read failed . . . 0
  Bus faults . . . . . 180

 Hard Drive 6
 SCSI Bus . . . . . 1 (internal/external)
 SCSI ID . . . . . 5
 Serial Number . . . . . 01290092
 Firmware Revision 1 . . . . 6215
 Model Number . . . . . COMPAQ ST15150W
 Initialized for Monitoring . Yes
  Reference time . . . . . 93958
  Sectors read . . . . . 3694421703
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 88249839
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 128832
  Seek errors . . . . . 0
  Spin cycles . . . . . 12
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 76%
  Seek time full . . . . . 75%
```

```
  Reallocated sectors . . . 262
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive  7
SCSI Bus . . . . . 1 (internal/external)
SCSI ID . . . . . 6
Serial Number . . . . . 01308086
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 93959
  Sectors read . . . . . 3693204269
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 87764378
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 128482
  Seek errors . . . . . 1
  Spin cycles . . . . . 12
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 76%
  Seek time full . . . . . 73%
  Reallocated sectors . . . 197
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive  8
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 0
Serial Number . . . . . 02256414
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94105
  Sectors read . . . . . 3707009169
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 98564807
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 85360
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 185
  Seek time track . . . . . 78%
  Seek time third . . . . . 74%
  Seek time full . . . . . 74%
  Reallocated sectors . . . 258
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive  9
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 1
Serial Number . . . . . 02231164
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94201
  Sectors read . . . . . 3704582296
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 96585540
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 85360
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 177
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 58
  Recovers read failed . . . 0
  Bus faults . . . . . 182

Hard Drive 10
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 2
Serial Number . . . . . 02292040
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94070
  Sectors read . . . . . 3697610102
  Hard read errors . . . . . 0
  Read errors retry . . . . 8
  ECC read errors . . . . . 0
  Sectors written . . . . . 98081973
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 96096
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 183
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 74%
  Reallocated sectors . . . 265
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive 11
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 3
Serial Number . . . . . 02262416
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94168
  Sectors read . . . . . 3705749743
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 98063815
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 91872
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 184
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 178
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive 12
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 4
Serial Number . . . . . 02176696
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94107
  Sectors read . . . . . 3706656267
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 98260185
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 88528
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 177
  Seek time track . . . . . 75%
  Seek time third . . . . . 74%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 162
  Recovers read failed . . . 0
  Bus faults . . . . . 180

Hard Drive 13
SCSI Bus . . . . . 2 (external)
SCSI ID . . . . . 5
Serial Number . . . . . 02279843
Firmware Revision 1 . . . . 6215
Model Number . . . . . COMPAQ  ST15150W
Initialized for Monitoring . Yes
  Reference time . . . . . 94069
  Sectors read . . . . . 3697038554
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 97692812
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 89408
  Seek errors . . . . . 0
  Spin cycles . . . . . 6
  Spin up time . . . . . 184
  Seek time track . . . . . 75%
  Seek time third . . . . . 73%
  Seek time full . . . . . 74%
  Reallocated sectors . . . 108
  Recovers read failed . . . 0
  Bus faults . . . . . 182
```

```
  Hard Drive 14
  SCSI Bus . . . . . 2 (external)
  SCSI ID . . . . . 6
  Serial Number . . . . . 00644373
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 144965
    Sectors read . . . . . *4170834235
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 92780331
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 349200
    Seek errors . . . . . 0
    Spin cycles . . . . . 22
    Spin up time . . . . . 0
    Seek time track . . . . . 78%
    Seek time third . . . . . 76%
    Seek time full . . . . . 76%
    Reallocated sectors . . . 134
    Recovers read failed . . . 0
    Bus faults . . . . . 12

Drive Controller 4, 32-Bit Compaq SMART-2/P Rev. B Array Controller
  IDA Firmware Revision . . . . 1.62
  Array Accelerator Memory . . . 4096 Kbytes
    Reserved for writes. . . . . 4096 Kbytes
  Accelerator Status . . . . . Enabled
    Battery count . . . . . 3
    Batteries charged . . . . . 3
    Batteries failed . . . . . 0
  ProLiant . . . . . Bus 1 (internal/external), Rev. JM14
  ProLiant . . . . . Bus 2 (external), Rev. JM14

  Logical Drive 1 . . . . . 12588 Megabyte
  Fault Tolerance . . . . . None
  OS Format . . . . . Multi-Sector Distribution
  Drive geometry (Cyl, Hds, Sec) 3013, 255, 32
  Array Accelerator . . . . . Enabled

    Hard Drive 1
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 0
    Serial Number . . . . . 00993091
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 156875
      Sectors read . . . . . *1843831426
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 392494061
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 349008
      Seek errors . . . . . 0
      Spin cycles . . . . . 129
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 209
      Recovers read failed . . . 0
      Bus faults . . . . . 22

    Hard Drive 2
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 1
    Serial Number . . . . . 00700527
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 115024
      Sectors read . . . . . *387015968
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 125663286
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 345080
      Seek errors . . . . . 0
      Spin cycles . . . . . 22
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 76%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 162
      Recovers read failed . . . 0
      Bus faults . . . . . 16

    Hard Drive 3
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 2
    Serial Number . . . . . 00859089
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 108286
      Sectors read . . . . . *368636513
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 106177959
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 220880
      Seek errors . . . . . 0
      Spin cycles . . . . . 14
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 74%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 89
      Recovers read failed . . . 0
      Bus faults . . . . . 6

    Hard Drive 4
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 3
    Serial Number . . . . . 00642466
    Firmware Revision 1 . . . . 6213
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 113827
      Sectors read . . . . . *374831247
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 123796168
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 344229
      Seek errors . . . . . 0
      Spin cycles . . . . . 30
      Spin up time . . . . . 0
      Seek time track . . . . . 75%
      Seek time third . . . . . 74%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 93
      Recovers read failed . . . 0
      Bus faults . . . . . 12

    Hard Drive 5
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 4
    Serial Number . . . . . 00835103
    Firmware Revision 1 . . . . 6215
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 109121
      Sectors read . . . . . *390470408
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 112240306
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 259073
      Seek errors . . . . . 1
      Spin cycles . . . . . 13
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 76%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 46
      Recovers read failed . . . 0
      Bus faults . . . . . 10

    Hard Drive 6
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 5
    Serial Number . . . . . 00223094
    Firmware Revision 1 . . . . 6213
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 113491
      Sectors read . . . . . *368840409
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 109063370
```

Hard write errors 0
Write errors retry 0
Seek count 312048
Seek errors 0
Spin cycles 23
Spin up time 0
Seek time track 78%
Seek time third 74%
Seek time full 75%
Reallocated sectors ... 77
Recovers read failed ... 0
Bus faults 24

Hard Drive 7
SCSI Bus ..... 1 (internal/external)
SCSI ID 6
Serial Number 00917144
Firmware Revision 1 6215
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 150999
Sectors read *1917191224
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 379789870
Hard write errors 0
Write errors retry 0
Seek count 348832
Seek errors 0
Spin cycles 131
Spin up time 182
Seek time track 78%
Seek time third 76%
Seek time full 75%
Reallocated sectors ... 545
Recovers read failed ... 0
Bus faults 14

Hard Drive 8
SCSI Bus ..... 2 (external)
SCSI ID 0
Serial Number 01289644
Firmware Revision 1 6215
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 96596
Sectors read 4066439086
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 96122067
Hard write errors 0
Write errors retry 0
Seek count 138864
Seek errors 0
Spin cycles 10
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 76%
Reallocated sectors ... 313
Recovers read failed ... 2
Bus faults 4

Hard Drive 9
SCSI Bus ..... 2 (external)
SCSI ID 1
Serial Number 00590543
Firmware Revision 1 6213
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 106286
Sectors read 4146822305
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 114181100
Hard write errors 0
Write errors retry 0
Seek count 304103
Seek errors 0
Spin cycles 25
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 75%
Reallocated sectors ... 130
Recovers read failed ... 0
Bus faults 4

Hard Drive 10
SCSI Bus ..... 2 (external)
SCSI ID 2
Serial Number 01323024
Firmware Revision 1 6215
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 96596
Sectors read 4070077783
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 99229337
Hard write errors 0
Write errors retry 0
Seek count 154528
Seek errors 0
Spin cycles 10
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors ... 113
Recovers read failed ... 2
Bus faults 4

Hard Drive 11
SCSI Bus ..... 2 (external)
SCSI ID 3
Serial Number 01311571
Firmware Revision 1 6215
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 96596
Sectors read 4066067114
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 95217215
Hard write errors 0
Write errors retry 0
Seek count 146256
Seek errors 0
Spin cycles 10
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 76%
Reallocated sectors ... 89
Recovers read failed ... 2
Bus faults 4

Hard Drive 12
SCSI Bus ..... 2 (external)
SCSI ID 4
Serial Number 00578614
Firmware Revision 1 6213
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 110334
Sectors read *82872453
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 135676534
Hard write errors 0
Write errors retry 0
Seek count 287760
Seek errors 0
Spin cycles 127
Spin up time 0
Seek time track 78%
Seek time third 77%
Seek time full 76%
Reallocated sectors ... 728
Recovers read failed ... 1
Bus faults 16

Hard Drive 13
SCSI Bus ..... 2 (external)
SCSI ID 5
Serial Number 00563713
Firmware Revision 1 6213
Model Number ..... COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 110338
Sectors read *79999356
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 135244924
Hard write errors 0
Write errors retry 0
Seek count 304128
Seek errors 0

```
  Spin cycles . . . . . 131
  Spin up time . . . . . 0
  Seek time track . . . . . 75%
  Seek time third . . . . . 75%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 65
  Recovers read failed . . . 2
  Bus faults . . . . . 12

 Hard Drive 14
 SCSI Bus . . . . . 2 (external)
 SCSI ID . . . . . 6
 Serial Number . . . . . 00504453
 Firmware Revision 1 . . . . 6213
 Model Number . . . . . COMPAQ ST15150W
 Initialized for Monitoring . Yes
  Reference time . . . . . 126366
  Sectors read . . . . . 4076514636
  Hard read errors . . . . . 0
  Read errors retry . . . . 0
  ECC read errors . . . . . 0
  Sectors written . . . . . 126693621
  Hard write errors . . . . 0
  Write errors retry . . . . 0
  Seek count . . . . . 308836
  Seek errors . . . . . 0
  Spin cycles . . . . . 49
  Spin up time . . . . . 0
  Seek time track . . . . . 78%
  Seek time third . . . . . 76%
  Seek time full . . . . . 75%
  Reallocated sectors . . . 301
  Recovers read failed . . . 0
  Bus faults . . . . . 16

Logical Drive 2 . . . . . 44113 Megabyte
Fault Tolerance . . . . . Distributed Data Guarding
OS Format . . . . . Multi-Sector Distribution
Drive geometry (Cyl, Hds, Sec) 10559, 255, 32
Array Accelerator . . . . . Enabled

  Hard Drive 1
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 0
  Serial Number . . . . . 00993091
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 156875
   Sectors read . . . . . *1843831426
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 392494061
   Hard write errors . . . . 0
   Write errors retry . . . . 0
   Seek count . . . . . 349184
   Seek errors . . . . . 0
   Spin cycles . . . . . 129
   Spin up time . . . . . 0
   Seek time track . . . . . 78%
   Seek time third . . . . . 75%
   Seek time full . . . . . 75%
   Reallocated sectors . . . 209
   Recovers read failed . . . 0
   Bus faults . . . . . 22

  Hard Drive 2
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 1
  Serial Number . . . . . 00700527
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 115024
   Sectors read . . . . . *387015968
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 125663286
   Hard write errors . . . . 0
   Write errors retry . . . . 0
   Seek count . . . . . 345256
   Seek errors . . . . . 0
   Spin cycles . . . . . 22
   Spin up time . . . . . 0
   Seek time track . . . . . 78%
   Seek time third . . . . . 76%
   Seek time full . . . . . 75%
   Reallocated sectors . . . 162
   Recovers read failed . . . 0
   Bus faults . . . . . 16

  Hard Drive 3
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 2
  Serial Number . . . . . 00859089
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 108286
   Sectors read . . . . . *368636513
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 106177959
   Hard write errors . . . . 0
   Write errors retry . . . . 0
   Seek count . . . . . 221056
   Seek errors . . . . . 0
   Spin cycles . . . . . 14
   Spin up time . . . . . 0
   Seek time track . . . . . 78%
   Seek time third . . . . . 74%
   Seek time full . . . . . 75%
   Reallocated sectors . . . 89
   Recovers read failed . . . 0
   Bus faults . . . . . 6

  Hard Drive 4
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 3
  Serial Number . . . . . 00642466
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 113827
   Sectors read . . . . . *374831247
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 123796168
   Hard write errors . . . . 0
   Write errors retry . . . . 0
   Seek count . . . . . 344405
   Seek errors . . . . . 0
   Spin cycles . . . . . 30
   Spin up time . . . . . 0
   Seek time track . . . . . 75%
   Seek time third . . . . . 74%
   Seek time full . . . . . 75%
   Reallocated sectors . . . 93
   Recovers read failed . . . 0
   Bus faults . . . . . 12

  Hard Drive 5
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 4
  Serial Number . . . . . 00835103
  Firmware Revision 1 . . . . 6215
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 109121
   Sectors read . . . . . *390470408
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 112240306
   Hard write errors . . . . 0
   Write errors retry . . . . 0
   Seek count . . . . . 259249
   Seek errors . . . . . 1
   Spin cycles . . . . . 13
   Spin up time . . . . . 0
   Seek time track . . . . . 78%
   Seek time third . . . . . 76%
   Seek time full . . . . . 75%
   Reallocated sectors . . . 46
   Recovers read failed . . . 0
   Bus faults . . . . . 10

  Hard Drive 6
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 5
  Serial Number . . . . . 00223094
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
   Reference time . . . . . 113491
   Sectors read . . . . . *368840409
   Hard read errors . . . . . 0
   Read errors retry . . . . 0
   ECC read errors . . . . . 0
   Sectors written . . . . . 109063370
   Hard write errors . . . . 0
   Write errors retry . . . . 0
```

Seek count 312224
Seek errors 0
Spin cycles 23
Spin up time 0
Seek time track 78%
Seek time third 74%
Seek time full 75%
Reallocated sectors . . . 77
Recovers read failed . . . 0
Bus faults 24

Hard Drive 7
SCSI Bus . . . . . 1 (internal/external)
SCSI ID 6
Serial Number 00917144
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 150999
Sectors read *1917191224
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 379789870
Hard write errors 0
Write errors retry 0
Seek count 349008
Seek errors 0
Spin cycles 131
Spin up time 182
Seek time track 78%
Seek time third 76%
Seek time full 75%
Reallocated sectors . . . 545
Recovers read failed . . . 0
Bus faults 14

Hard Drive 8
SCSI Bus . . . . . 2 (external)
SCSI ID 0
Serial Number 01289644
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 96596
Sectors read 4066439086
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 96122067
Hard write errors 0
Write errors retry 0
Seek count 139040
Seek errors 0
Spin cycles 10
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 76%
Reallocated sectors . . . 313
Recovers read failed . . . 2
Bus faults 4

Hard Drive 9
SCSI Bus . . . . . 2 (external)
SCSI ID 1
Serial Number 00590543
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 106287
Sectors read 4146822305
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 114181100
Hard write errors 0
Write errors retry 0
Seek count 304279
Seek errors 0
Spin cycles 25
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 75%
Reallocated sectors . . . 130
Recovers read failed . . . 0
Bus faults 4

Hard Drive 10
SCSI Bus . . . . . 2 (external)
SCSI ID 2
Serial Number 01323024
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 96597
Sectors read 4070077783
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 99229337
Hard write errors 0
Write errors retry 0
Seek count 154704
Seek errors 0
Spin cycles 10
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors . . . 113
Recovers read failed . . . 2
Bus faults 4

Hard Drive 11
SCSI Bus . . . . . 2 (external)
SCSI ID 3
Serial Number 01311571
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 96597
Sectors read 4066067114
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 95217215
Hard write errors 0
Write errors retry 0
Seek count 146432
Seek errors 0
Spin cycles 10
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 76%
Reallocated sectors . . . 89
Recovers read failed . . . 2
Bus faults 4

Hard Drive 12
SCSI Bus . . . . . 2 (external)
SCSI ID 4
Serial Number 00578614
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 110335
Sectors read *82872453
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 135676534
Hard write errors 0
Write errors retry 0
Seek count 287936
Seek errors 0
Spin cycles 127
Spin up time 0
Seek time track 78%
Seek time third 77%
Seek time full 76%
Reallocated sectors . . . 728
Recovers read failed . . . 1
Bus faults 16

Hard Drive 13
SCSI Bus . . . . . 2 (external)
SCSI ID 5
Serial Number 00563713
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 110339
Sectors read *79999356
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 135244924
Hard write errors 0
Write errors retry 0
Seek count 304304
Seek errors 0
Spin cycles 131
Spin up time 0

```
    Seek time track . . . . . 75%
    Seek time third . . . . . 75%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 65
    Recovers read failed . . . 2
    Bus faults . . . . . 12

  Hard Drive 14
  SCSI Bus . . . . . 2 (external)
  SCSI ID . . . . . 6
  Serial Number . . . . . 00504453
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 126367
    Sectors read . . . . . 4076514636
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 126693621
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 309012
    Seek errors . . . . . 0
    Spin cycles . . . . . 49
    Spin up time . . . . . 0
    Seek time track . . . . . 78%
    Seek time third . . . . . 76%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 301
    Recovers read failed . . . 0
    Bus faults . . . . . 16

Drive Controller 5, 32-Bit Compaq SMART-2/P Rev. B Array Controller
  IDA Firmware Revision . . . . 1.62
  Array Accelerator Memory . . . 4096 Kbytes
    Reserved for writes. . . . . 4096 Kbytes
  Accelerator Status . . . . . Enabled
    Battery count . . . . . 3
    Batteries charged . . . . . 3
    Batteries failed . . . . . 0
  ProLiant . . . . . Bus 1 (internal/external), Rev. JM14
  ProLiant . . . . . Bus 2 (external), Rev. JM14

  Logical Drive 1 . . . . . 6295 Megabyte
  Fault Tolerance . . . . . None
  OS Format . . . . . Multi-Sector Distribution
  Drive geometry (Cyl, Hds, Sec) 1507, 255, 32
  Array Accelerator . . . . . Enabled

    Hard Drive 1
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 0
    Serial Number . . . . . 00551515
    Firmware Revision 1 . . . . 6213
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 131992
      Sectors read . . . . . 85832131
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 143974858
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 317152
      Seek errors . . . . . 0
      Spin cycles . . . . . 44
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 76%
      Seek time full . . . . . 76%
      Reallocated sectors . . . 39
      Recovers read failed . . . 0
      Bus faults . . . . . 20

    Hard Drive 2
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 1
    Serial Number . . . . . 00563147
    Firmware Revision 1 . . . . 6213
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 131992
      Sectors read . . . . . 85618144
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 143990977
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 319438
      Seek errors . . . . . 0
      Spin cycles . . . . . 44
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 199
      Recovers read failed . . . 0
      Bus faults . . . . . 18

    Hard Drive 3
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 2
    Serial Number . . . . . 00561164
    Firmware Revision 1 . . . . 6213
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 131992
      Sectors read . . . . . 85832791
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 144121685
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 316572
      Seek errors . . . . . 0
      Spin cycles . . . . . 44
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 76%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 54
      Recovers read failed . . . 0
      Bus faults . . . . . 18

    Hard Drive 4
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 3
    Serial Number . . . . . 00576560
    Firmware Revision 1 . . . . 6213
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 131992
      Sectors read . . . . . 96646075
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 160724183
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 318384
      Seek errors . . . . . 0
      Spin cycles . . . . . 44
      Spin up time . . . . . 0
      Seek time track . . . . . 75%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 65
      Recovers read failed . . . 0
      Bus faults . . . . . 16

    Hard Drive 5
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 4
    Serial Number . . . . . 00570251
    Firmware Revision 1 . . . . 6213
    Model Number . . . . . COMPAQ ST15150W
    Initialized for Monitoring . Yes
      Reference time . . . . . 131993
      Sectors read . . . . . 96769411
      Hard read errors . . . . . 0
      Read errors retry . . . . 0
      ECC read errors . . . . . 0
      Sectors written . . . . . 160815633
      Hard write errors . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 314160
      Seek errors . . . . . 0
      Spin cycles . . . . . 44
      Spin up time . . . . . 0
      Seek time track . . . . . 78%
      Seek time third . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors . . . 166
      Recovers read failed . . . 0
      Bus faults . . . . . 20

    Hard Drive 6
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID . . . . . 5
    Serial Number . . . . . 00467332
    Firmware Revision 1 . . . . 6213
    Model Number . . . . . COMPAQ ST15150W
```

```
Initialized for Monitoring .  Yes
  Reference time . . . . . .  131992
  Sectors read . . . . . .  96760431
  Hard read errors . . . . .  0
  Read errors retry . . . .  0
  ECC read errors . . . . .  0
  Sectors written . . . . .  160674314
  Hard write errors . . . .  0
  Write errors retry . . . .  0
  Seek count . . . . . .  312861
  Seek errors . . . . . .  0
  Spin cycles . . . . . .  44
  Spin up time . . . . . .  0
  Seek time track . . . . .  78%
  Seek time third . . . . .  75%
  Seek time full . . . . . .  75%
  Reallocated sectors . . .  132
  Recovers read failed . . .  0
  Bus faults . . . . . .  16

Hard Drive  7
SCSI Bus . . . . . .  1 (internal/external)
SCSI ID . . . . . .  6
Serial Number . . . . . .  01304458
Firmware Revision 1 . . . .  6215
Model Number . . . . . .  COMPAQ  ST15150W
Initialized for Monitoring .  Yes
  Reference time . . . . . .  94067
  Sectors read . . . . . .  63828094
  Hard read errors . . . . .  0
  Read errors retry . . . .  0
  ECC read errors . . . . .  0
  Sectors written . . . . .  80381137
  Hard write errors . . . .  0
  Write errors retry . . . .  0
  Seek count . . . . . .  139040
  Seek errors . . . . . .  0
  Spin cycles . . . . . .  12
  Spin up time . . . . . .  0
  Seek time track . . . . .  78%
  Seek time third . . . . .  75%
  Seek time full . . . . . .  75%
  Reallocated sectors . . .  467
  Recovers read failed . . .  1
  Bus faults . . . . . .  8

Logical Drive  2 . . . . . .  10485 Megabyte
Fault Tolerance . . . . . .  None
OS Format . . . . . .  Multi-Sector Distribution
Drive geometry (Cyl, Hds, Sec)  2510, 255, 32
Array Accelerator . . . . . .  Enabled

  Hard Drive  1
  SCSI Bus . . . . . .  2 (external)
  SCSI ID . . . . . .  0
  Serial Number . . . . . .  00566176
  Firmware Revision 1 . . . .  6213
  Model Number . . . . . .  COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . .  117630
    Sectors read . . . . . .  705974174
    Hard read errors . . . . .  0
    Read errors retry . . . .  0
    ECC read errors . . . . .  0
    Sectors written . . . . .  182252159
    Hard write errors . . . .  0
    Write errors retry . . . .  0
    Seek count . . . . . .  316976
    Seek errors . . . . . .  0
    Spin cycles . . . . . .  45
    Spin up time . . . . . .  0
    Seek time track . . . . .  75%
    Seek time third . . . . .  75%
    Seek time full . . . . . .  75%
    Reallocated sectors . . .  58
    Recovers read failed . . .  0
    Bus faults . . . . . .  14

  Hard Drive  2
  SCSI Bus . . . . . .  2 (external)
  SCSI ID . . . . . .  1
  Serial Number . . . . . .  00314010
  Firmware Revision 1 . . . .  6213
  Model Number . . . . . .  COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . .  121477
    Sectors read . . . . . .  714553002
    Hard read errors . . . . .  0
    Read errors retry . . . .  0
    ECC read errors . . . . .  0
    Sectors written . . . . .  183493166
    Hard write errors . . . .  0
    Write errors retry . . . .  0
    Seek count . . . . . .  310764
    Seek errors . . . . . .  0
    Spin cycles . . . . . .  41
    Spin up time . . . . . .  0
    Seek time track . . . . .  78%
    Seek time third . . . . .  75%
    Seek time full . . . . . .  75%
    Reallocated sectors . . .  397
    Recovers read failed . . .  0
    Bus faults . . . . . .  12

  Hard Drive  3
  SCSI Bus . . . . . .  2 (external)
  SCSI ID . . . . . .  2
  Serial Number . . . . . .  00641660
  Firmware Revision 1 . . . .  6213
  Model Number . . . . . .  COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . .  112783
    Sectors read . . . . . .  688277472
    Hard read errors . . . . .  0
    Read errors retry . . . .  0
    ECC read errors . . . . .  0
    Sectors written . . . . .  147288992
    Hard write errors . . . .  0
    Write errors retry . . . .  0
    Seek count . . . . . .  299376
    Seek errors . . . . . .  0
    Spin cycles . . . . . .  49
    Spin up time . . . . . .  0
    Seek time track . . . . .  78%
    Seek time third . . . . .  75%
    Seek time full . . . . . .  75%
    Reallocated sectors . . .  149
    Recovers read failed . . .  0
    Bus faults . . . . . .  30

  Hard Drive  4
  SCSI Bus . . . . . .  2 (external)
  SCSI ID . . . . . .  3
  Serial Number . . . . . .  00570167
  Firmware Revision 1 . . . .  6213
  Model Number . . . . . .  COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . .  117621
    Sectors read . . . . . .  698697055
    Hard read errors . . . . .  0
    Read errors retry . . . .  0
    ECC read errors . . . . .  0
    Sectors written . . . . .  191168691
    Hard write errors . . . .  0
    Write errors retry . . . .  0
    Seek count . . . . . .  316245
    Seek errors . . . . . .  0
    Spin cycles . . . . . .  44
    Spin up time . . . . . .  0
    Seek time track . . . . .  78%
    Seek time third . . . . .  76%
    Seek time full . . . . . .  75%
    Reallocated sectors . . .  24
    Recovers read failed . . .  0
    Bus faults . . . . . .  14

  Hard Drive  5
  SCSI Bus . . . . . .  2 (external)
  SCSI ID . . . . . .  4
  Serial Number . . . . . .  00536143
  Firmware Revision 1 . . . .  6213
  Model Number . . . . . .  COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . .  117631
    Sectors read . . . . . .  698391922
    Hard read errors . . . . .  0
    Read errors retry . . . .  0
    ECC read errors . . . . .  0
    Sectors written . . . . .  191103071
    Hard write errors . . . .  0
    Write errors retry . . . .  0
    Seek count . . . . . .  309232
    Seek errors . . . . . .  0
    Spin cycles . . . . . .  45
    Spin up time . . . . . .  0
    Seek time track . . . . .  75%
    Seek time third . . . . .  76%
    Seek time full . . . . . .  75%
    Reallocated sectors . . .  262
    Recovers read failed . . .  0
    Bus faults . . . . . .  18

  Hard Drive  6
  SCSI Bus . . . . . .  2 (external)
  SCSI ID . . . . . .  5
  Serial Number . . . . . .  00591543
```

```
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 117631
    Sectors read . . . . . 698192500
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 190838795
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 312003
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 45
    Spin up time . . . . . 0
    Seek time track  . . . . . 78%
    Seek time third  . . . . . 76%
    Seek time full . . . . . 72%
    Reallocated sectors  . . . 188
    Recovers read failed . . . 0
    Bus faults . . . . . 18

  Hard Drive  7
  SCSI Bus . . . . . 2 (external)
  SCSI ID  . . . . . 6
  Serial Number  . . . . . 00836485
  Firmware Revision 1  . . . . 6215
  Model Number . . . . . COMPAQ  ST15150W
  Initialized for Monitoring .  Yes
    Reference time . . . . . 106345
    Sectors read . . . . . 676173122
    Hard read errors . . . . . 0
    Read errors retry  . . . . 0
    ECC read errors  . . . . . 0
    Sectors written  . . . . . 157232499
    Hard write errors  . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 247808
    Seek errors  . . . . . 0
    Spin cycles  . . . . . 13
    Spin up time . . . . . 0
    Seek time track  . . . . . 78%
    Seek time third  . . . . . 76%
    Seek time full . . . . . 75%
    Reallocated sectors  . . . 50
    Recovers read failed . . . 0
    Bus faults . . . . . 8

Drive Controller 6, 32-Bit Compaq SMART-2/P Rev. B Array Controller
  IDA Firmware Revision  . . . . 1.62
  Array Accelerator Memory . . . 4096 Kbytes
    Reserved for writes. . . . . 4096 Kbytes
  Accelerator Status . . . . . Enabled
    Battery count  . . . . . 3
    Batteries charged  . . . . . 3
    Batteries failed . . . . . 0
  ProLiant . . . . . Bus 1 (internal/external), Rev. JM14
  ProLiant . . . . . Bus 2 (external), Rev. JM14

  Logical Drive  1 . . . . . 18194 Megabyte
  Fault Tolerance  . . . . . Mirroring
  OS Format  . . . . . Multi-Sector Distribution
  Drive geometry (Cyl, Hds, Sec)  4355, 255, 32
  Array Accelerator  . . . . . Enabled

    Hard Drive  1
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID  . . . . . 0
    Serial Number  . . . . . 6123104797
    Firmware Revision 1  . . . . C424
    Model Number . . . . . COMPAQ  3391SS
    Initialized for Monitoring .  Yes
      Reference time . . . . . 78039
      Sectors read . . . . . *1115668356
      Hard read errors . . . . . 0
      Read errors retry  . . . . 0
      ECC read errors  . . . . . 0
      Sectors written  . . . . . 747807435
      Hard write errors  . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 54736
      Seek errors  . . . . . 0
      Spin cycles  . . . . . 6
      Spin up time . . . . . 223
      Seek time track  . . . . . 63%
      Seek time third  . . . . . 70%
      Seek time full . . . . . 73%
      Reallocated sectors  . . . 275
      Recovers read failed . . . 0
      Bus faults . . . . . 6

    Hard Drive  2
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID  . . . . . 1
    Serial Number  . . . . . 6113100965
    Firmware Revision 1  . . . . C424
    Model Number . . . . . COMPAQ  3391WS
    Initialized for Monitoring .  No

    Hard Drive  3
    SCSI Bus . . . . . 2 (external)
    SCSI ID  . . . . . 0
    Serial Number  . . . . . 6123104333
    Firmware Revision 1  . . . . C424
    Model Number . . . . . COMPAQ  3391SS
    Initialized for Monitoring .  Yes
      Reference time . . . . . 78027
      Sectors read . . . . . *1126697912
      Hard read errors . . . . . 0
      Read errors retry  . . . . 0
      ECC read errors  . . . . . 0
      Sectors written  . . . . . 747116310
      Hard write errors  . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 54032
      Seek errors  . . . . . 0
      Spin cycles  . . . . . 7
      Spin up time . . . . . 227
      Seek time track  . . . . . 57%
      Seek time third  . . . . . 70%
      Seek time full . . . . . 74%
      Reallocated sectors  . . . 180
      Recovers read failed . . . 0
      Bus faults . . . . . 4

    Hard Drive  4
    SCSI Bus . . . . . 2 (external)
    SCSI ID  . . . . . 1
    Serial Number  . . . . . 6123104182
    Firmware Revision 1  . . . . C424
    Model Number . . . . . COMPAQ  3391SS
    Initialized for Monitoring .  Yes
      Reference time . . . . . 75449
      Sectors read . . . . . *3430097506
      Hard read errors . . . . . 0
      Read errors retry  . . . . 0
      ECC read errors  . . . . . 0
      Sectors written  . . . . . 689869037
      Hard write errors  . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 45056
      Seek errors  . . . . . 0
      Spin cycles  . . . . . 8
      Spin up time . . . . . 0
      Seek time track  . . . . . 57%
      Seek time third  . . . . . 71%
      Seek time full . . . . . 74%
      Reallocated sectors  . . . 2172
      Recovers read failed . . . 0
      Bus faults . . . . . 9

  Logical Drive  2 . . . . . 2097 Megabyte
  Fault Tolerance  . . . . . None
  OS Format  . . . . . Multi-Sector Distribution
  Drive geometry (Cyl, Hds, Sec)  527, 243, 32
  Array Accelerator  . . . . . Enabled

    Hard Drive  1
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID  . . . . . 2
    Serial Number  . . . . . 00241651
    Firmware Revision 1  . . . . 6213
    Model Number . . . . . COMPAQ  ST15150W
    Initialized for Monitoring .  Yes
      Reference time . . . . . 359842
      Sectors read . . . . . 4222641195
      Hard read errors . . . . . 0
      Read errors retry  . . . . 0
      ECC read errors  . . . . . 0
      Sectors written  . . . . . 374675823
      Hard write errors  . . . . 0
      Write errors retry . . . . 0
      Seek count . . . . . 301472
      Seek errors  . . . . . 0
      Spin cycles  . . . . . 30
      Spin up time . . . . . 0
      Seek time track  . . . . . 75%
      Seek time third  . . . . . 75%
      Seek time full . . . . . 75%
      Reallocated sectors  . . . 160
      Recovers read failed . . . 0
      Bus faults . . . . . 16

    Hard Drive  2
    SCSI Bus . . . . . 1 (internal/external)
    SCSI ID  . . . . . 3
```

Serial Number 00583246
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 385825
Sectors read 4239461476
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 390917621
Hard write errors 0
Write errors retry 0
Seek count 319968
Seek errors 0
Spin cycles 47
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 75%
Reallocated sectors . . . 14
Recovers read failed . . . 0
Bus faults 12

Hard Drive 3
SCSI Bus . . . . . 1 (internal/external)
SCSI ID 4
Serial Number 00640660
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 359834
Sectors read 4221446482
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 374039602
Hard write errors 0
Write errors retry 0
Seek count 291632
Seek errors 0
Spin cycles 31
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors . . . 312
Recovers read failed . . . 0
Bus faults 12

Hard Drive 4
SCSI Bus . . . . . 1 (internal/external)
SCSI ID 5
Serial Number 00554822
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 385824
Sectors read 4249562078
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 402050371
Hard write errors 0
Write errors retry 0
Seek count 314864
Seek errors 0
Spin cycles 47
Spin up time 0
Seek time track 78%
Seek time third 76%
Seek time full 75%
Reallocated sectors . . . 476
Recovers read failed . . . 0
Bus faults 18

Hard Drive 5
SCSI Bus . . . . . 1 (internal/external)
SCSI ID 6
Serial Number 00652828
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 359773
Sectors read 4210911619
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 353342832
Hard write errors 0
Write errors retry 0
Seek count 287809
Seek errors 0
Spin cycles 34
Spin up time 0
Seek time track 78%
Seek time third 74%
Seek time full 75%
Reallocated sectors . . . 89
Recovers read failed . . . 0
Bus faults 12

Hard Drive 6
SCSI Bus . . . . . 2 (external)
SCSI ID 2
Serial Number 00571754
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 385822
Sectors read 4248164468
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 403227192
Hard write errors 0
Write errors retry 0
Seek count 332640
Seek errors 0
Spin cycles 48
Spin up time 0
Seek time track 75%
Seek time third 75%
Seek time full 75%
Reallocated sectors . . . 139
Recovers read failed . . . 0
Bus faults 13

Hard Drive 7
SCSI Bus . . . . . 2 (external)
SCSI ID 3
Serial Number 00578637
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 385825
Sectors read 4237705681
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 392334064
Hard write errors 0
Write errors retry 0
Seek count 327712
Seek errors 0
Spin cycles 46
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors . . . 131
Recovers read failed . . . 0
Bus faults 12

Hard Drive 8
SCSI Bus . . . . . 2 (external)
SCSI ID 4
Serial Number 00719968
Firmware Revision 1 6215
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes
Reference time 359834
Sectors read 4219241764
Hard read errors 0
Read errors retry 0
ECC read errors 0
Sectors written 374594183
Hard write errors 0
Write errors retry 0
Seek count 296001
Seek errors 0
Spin cycles 25
Spin up time 0
Seek time track 78%
Seek time third 75%
Seek time full 75%
Reallocated sectors . . . 108
Recovers read failed . . . 0
Bus faults 12

Hard Drive 9
SCSI Bus . . . . . 2 (external)
SCSI ID 5
Serial Number 00468105
Firmware Revision 1 6213
Model Number . . . . . COMPAQ ST15150W
Initialized for Monitoring . Yes

```
    Reference time . . . . . 385825
    Sectors read . . . . . 4234862008
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 391470050
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 321200
    Seek errors . . . . . 0
    Spin cycles . . . . . 46
    Spin up time . . . . . 0
    Seek time track . . . . . 78%
    Seek time third . . . . . 75%
    Seek time full . . . . . 72%
    Reallocated sectors . . . 459
    Recovers read failed . . . 0
    Bus faults . . . . . 12

Logical Drive 3 . . . . . 36535 Megabyte
Fault Tolerance . . . . . None
OS Format . . . . . Multi-Sector Distribution
Drive geometry (Cyl, Hds, Sec) 8745, 255, 32
Array Accelerator . . . . . Enabled

  Hard Drive 1
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 2
  Serial Number . . . . . 00241651
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 359843
    Sectors read . . . . . 4222641195
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 374675823
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 301648
    Seek errors . . . . . 0
    Spin cycles . . . . . 30
    Spin up time . . . . . 0
    Seek time track . . . . . 75%
    Seek time third . . . . . 75%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 160
    Recovers read failed . . . 0
    Bus faults . . . . . 16

  Hard Drive 2
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 3
  Serial Number . . . . . 00583246
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 385826
    Sectors read . . . . . 4239461476
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 390917621
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 320144
    Seek errors . . . . . 0
    Spin cycles . . . . . 47
    Spin up time . . . . . 0
    Seek time track . . . . . 78%
    Seek time third . . . . . 76%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 14
    Recovers read failed . . . 0
    Bus faults . . . . . 12

  Hard Drive 3
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 4
  Serial Number . . . . . 00640660
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 359835
    Sectors read . . . . . 4221446482
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 374039602
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 291808
    Seek errors . . . . . 0
    Spin cycles . . . . . 31
    Spin up time . . . . . 0
    Seek time track . . . . . 78%
    Seek time third . . . . . 75%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 312
    Recovers read failed . . . 0
    Bus faults . . . . . 12

  Hard Drive 4
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 5
  Serial Number . . . . . 00554822
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 385825
    Sectors read . . . . . 4249562078
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 402050371
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 315040
    Seek errors . . . . . 0
    Spin cycles . . . . . 47
    Spin up time . . . . . 0
    Seek time track . . . . . 78%
    Seek time third . . . . . 76%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 476
    Recovers read failed . . . 0
    Bus faults . . . . . 18

  Hard Drive 5
  SCSI Bus . . . . . 1 (internal/external)
  SCSI ID . . . . . 6
  Serial Number . . . . . 00652828
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 359774
    Sectors read . . . . . 4210911619
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 353342832
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 287985
    Seek errors . . . . . 0
    Spin cycles . . . . . 34
    Spin up time . . . . . 0
    Seek time track . . . . . 78%
    Seek time third . . . . . 74%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 89
    Recovers read failed . . . 0
    Bus faults . . . . . 12

  Hard Drive 6
  SCSI Bus . . . . . 2 (external)
  SCSI ID . . . . . 2
  Serial Number . . . . . 00571754
  Firmware Revision 1 . . . . 6213
  Model Number . . . . . COMPAQ ST15150W
  Initialized for Monitoring . Yes
    Reference time . . . . . 385823
    Sectors read . . . . . 4248164468
    Hard read errors . . . . . 0
    Read errors retry . . . . 0
    ECC read errors . . . . . 0
    Sectors written . . . . . 403227192
    Hard write errors . . . . 0
    Write errors retry . . . . 0
    Seek count . . . . . 332816
    Seek errors . . . . . 0
    Spin cycles . . . . . 48
    Spin up time . . . . . 0
    Seek time track . . . . . 75%
    Seek time third . . . . . 75%
    Seek time full . . . . . 75%
    Reallocated sectors . . . 139
    Recovers read failed . . . 0
    Bus faults . . . . . 13

  Hard Drive 7
  SCSI Bus . . . . . 2 (external)
  SCSI ID . . . . . 3
  Serial Number . . . . . 00578637
  Firmware Revision 1 . . . . 6213
```

```
Model Number . . . . . .  COMPAQ  ST15150W
Initialized for Monitoring .  Yes
  Reference time . . . . . .  385826
  Sectors read . . . . . .  4237705681
  Hard read errors . . . . .  0
  Read errors retry  . . . .  0
  ECC read errors  . . . . .  0
  Sectors written  . . . . .  392334064
  Hard write errors  . . . .  0
  Write errors retry . . . .  0
  Seek count . . . . . .  327888
  Seek errors  . . . . . .  0
  Spin cycles  . . . . . .  46
  Spin up time . . . . . .  0
  Seek time track  . . . . .  78%
  Seek time third  . . . . .  75%
  Seek time full . . . . . .  75%
  Reallocated sectors  . . .  131
  Recovers read failed . . .  0
  Bus faults . . . . . .  12

Hard Drive  8
SCSI Bus . . . . . .  2 (external)
SCSI ID  . . . . . .  4
Serial Number  . . . . . .  00719968
Firmware Revision 1  . . . .  6215
Model Number . . . . . .  COMPAQ  ST15150W
Initialized for Monitoring .  Yes
  Reference time . . . . . .  359835
  Sectors read . . . . . .  4219241764
  Hard read errors . . . . .  0
  Read errors retry  . . . .  0
  ECC read errors  . . . . .  0
  Sectors written  . . . . .  374594183
  Hard write errors  . . . .  0
  Write errors retry . . . .  0
  Seek count . . . . . .  296177
  Seek errors  . . . . . .  0
  Spin cycles  . . . . . .  25
  Spin up time . . . . . .  0
  Seek time track  . . . . .  78%
  Seek time third  . . . . .  75%
  Seek time full . . . . . .  75%
  Reallocated sectors  . . .  108
  Recovers read failed . . .  0
  Bus faults . . . . . .  12

Hard Drive  9
SCSI Bus . . . . . .  2 (external)
SCSI ID  . . . . . .  5
Serial Number  . . . . . .  00468105
Firmware Revision 1  . . . .  6213
Model Number . . . . . .  COMPAQ  ST15150W
Initialized for Monitoring .  Yes
  Reference time . . . . . .  385826
  Sectors read . . . . . .  4234862008
  Hard read errors . . . . .  0
  Read errors retry  . . . .  0
  ECC read errors  . . . . .  0
  Sectors written  . . . . .  391470050
  Hard write errors  . . . .  0
  Write errors retry . . . .  0
  Seek count . . . . . .  321376
  Seek errors  . . . . . .  0
  Spin cycles  . . . . . .  46
  Spin up time . . . . . .  0
  Seek time track  . . . . .  78%
  Seek time third  . . . . .  75%
  Seek time full . . . . . .  72%
  Reallocated sectors  . . .  459
  Recovers read failed . . .  0
  Bus faults . . . . . .  12

Graphics Mode  . . . . . .  03 (80-Column Text)


Primary Monitor attached to  . .  Cirrus CL-GD5430 Graphics Controller
                                  with Video Graphics Color Monitor

Total Video Memory . . . . . .  1024 Kbytes

Base Memory
  System Total . . . . . .  638 Kbytes
  Amount Free  . . . . . .  596 Kbytes (611008 Bytes)

Extended Memory
  Amount Free  . . . . . .  1041856 Kbytes

Expanded Memory
  LIM Driver Support . . . . . .  LIM driver not loaded

Operating System . . . . . .  MS-DOS version 7.00 (from diskette)

Status of file: C:\IO.SYS
  Date and Time  . . . . . .  01/31/97  17:46:02
  Size . . . . . .  0 Bytes

Status of file: C:\MSDOS.SYS
  Date and Time  . . . . . .  01/31/97  17:46:02
  Size . . . . . .  0 Bytes

-------------------------------------------------------------------------------
** Dump of C:\CONFIG.SYS
-------------------------------------------------------------------------------
** End of file

Status of file: C:\COMMAND.COM
*** File does not exist

-------------------------------------------------------------------------------
** Dump of C:\AUTOEXEC.BAT
-------------------------------------------------------------------------------
** End of file

Environment variables
  PATH=
  PROMPT=$P$G
  COMSPEC=A:\COMMAND.COM
  CMDLINE=inspect /u
End of environment
Critical Error Log
==================
Correctable Memory Error Log
============================

System Configuration Utility . .  Version 2.33

Extended Non-volatile Memory
-------- ------------ ------
Slot . . . . . .  0
  Slot Type  . . . . . .  Embedded
  Board ID . . . . . .  CPQ0551
  CFG File Extension
    Revision Level . . . . . .  2.21
  Type Entry(s)  . . . . . .  MSD,FPYCTL
    IRQ Entry(s):
      IRQ  6, Not Shared, Edge Triggered
    DMA Channel(s):
      Channel 2, Not Shared
      Timing: Type B
      Transfer Size: 8-bit (byte)
    Port Range(s):
      03F0h - 03F5h, Not Shared
      03F6h - 03F7h, Shared

  Type Entry(s)  . . . . . .  MSD,UNIT0,FPYDRV;TYP=4
  Type Entry(s)  . . . . . .  MSD,UNIT1,FPYDRV;TYP=0
  Type Entry(s)  . . . . . .  MSD
    IRQ Entry(s):
      IRQ 14, Not Shared, Edge Triggered
    Port Range(s):
      01F0h - 01F7h, Not Shared
      03F6h - 03F7h, Shared
      11F1h, Not Shared

  Memory Entry(s):
                                  Range              Size
                                  -----              ----
    ROM: Other,    Cacheable      896K -       1M      128K

  Type Entry(s)  . . . . . .  MEM;COMPAQ
  Memory Entry(s):
                                  Range              Size
                                  -----              ----
    RAM: System,   Cacheable      0K -       640K      640K

  Type Entry(s)  . . . . . .  MEM;COMPAQ
  Memory Entry(s):
                                  Range              Size
                                  -----              ----
    RAM: System,   Cacheable      1M -        16M       15M

  Type Entry(s)  . . . . . .  MEM;COMPAQ
  Memory Entry(s):
                                  Range              Size
                                  -----              ----
    RAM: System,   Cacheable      16M -       64M       48M
    RAM: System,   Cacheable      64M -      128M       64M
    RAM: System,   Cacheable      128M -     192M       64M
    RAM: System,   Cacheable      192M -     256M       64M
    RAM: System,   Cacheable      256M -     320M       64M
    RAM: System,   Cacheable      320M -     384M       64M
    RAM: System,   Cacheable      384M -     448M       64M
    RAM: System,   Cacheable      448M -     512M       64M

  Type Entry(s)  . . . . . .  MEM;COMPAQ
```

```
Memory Entry(s):
                        Range              Size
                        -----              ----
  RAM: System,  Cacheable       512M -     576M        64M
  RAM: System,  Cacheable       576M -     640M        64M
  RAM: System,  Cacheable       640M -     704M        64M
  RAM: System,  Cacheable       704M -     768M        64M
  RAM: System,  Cacheable       768M -     832M        64M
  RAM: System,  Cacheable       832M -     896M        64M
  RAM: System,  Cacheable       896M -     960M        64M
  RAM: System,  Cacheable       960M -    1024M        64M

Type Entry(s) . . . . . MEM;COMPAQ
Type Entry(s) . . . . . COM,ASY
Type Entry(s) . . . . . COM,ASY
Type Entry(s) . . . . . PAR
Type Entry(s) . . . . . PTR,8042
  IRQ Entry(s):
    IRQ 12, Not Shared, Edge Triggered
Type Entry(s) . . . . . OTH,CPQCSM
Free Form Text . . . . . 0D 03 03 03 01 78 3C 02 50 0C 00 00
Type Entry(s) . . . . . OTH,A20
Type Entry(s) . . . . . OTH,SOFTNMI
Type Entry(s) . . . . . OTH,FLSFNMI
Type Entry(s) . . . . . OTH,BUSNMI
Type Entry(s) . . . . . OTH,DSKTDMA
Type Entry(s) . . . . . OTH,REFRESH
Type Entry(s) . . . . . OTH,PERR
Type Entry(s) . . . . . OTH,SIMMSPD;AUTO
Type Entry(s) . . . . . OTH,TABLE;DEFAULT6
Type Entry(s) . . . . . OTH,CURREV
Free Form Text . . . . . 8C 4C 18
Type Entry(s) . . . . . OTH,PREREV
Free Form Text . . . . . 6B 4C 18
Type Entry(s) . . . . . OTH,CPR,NMI
Free Form Text . . . . . 01 00 0A 00 D8 0C C1 C5
  Port Range(s):
    7C80h - 7C83h, Not Shared
    8C80h - 8C83h, Not Shared
    9C80h - 9C83h, Not Shared
    AC80h - AC83h, Not Shared
    BC80h - BC83h, Not Shared
    CC80h - CC83h, Not Shared
    DC80h - DC83h, Not Shared
    EC80h - EC83h, Not Shared
    FC80h - FC83h, Not Shared

Memory Entry(s):
                        Range              Size
                        -----              ----
  RAM: Virtual,  Non-Cacheable  2060M -  2109441K         1K

  IRQ Entry(s):
    IRQ 13, Not Shared, Edge Triggered
Type Entry(s) . . . . . ISA;MAP
Free Form Text . . . . . 61 62 63 64 85 86 87 E0 E0 E0 E0 E0 E0
E0
                        E0 01 02 03 04 00 00 00 00 00 00 00 00
                        00 00
Type Entry(s) . . . . . ISA;PCIMAP
Free Form Text . . . . . 01 00 68 02 00 58 03 00 50 04 01 68 05 01
                        58 06 01 50

Empty Slot(s) . . . . . 1 2 3 4

Revisions Table
===============

Current Revisions
  Date . . . . . 3/4/97

Previous Revisions
  Date . . . . . 3/3/97

Memory Allocation (including INSPECT)
  PSP   SIZE    NAME         TRAPPED INTERRUPTS
  ----  ------  -----------  ----------------------------------------
  088F  007200  COMMAND.COM   FBh 2Fh 2Eh 24h 23h 22h
  0A5A  208720  INSPECT.EXE   FFh EFh EAh 3Fh 00h

System Configuration Memory
  00 - 0F :   47 00 59 00   14 00 01 21   04 97 26 82   50 80 00 00
  10 - 1F :   40 00 00 00   03 80 02 00   3C 00 00 00   00 00 00 00
  20 - 2F :   00 00 00 00   7F 20 00 40   00 70 00 00   00 10 02 60
  30 - 3F :   00 3C 19 80   00 00 XX XX   XX XX XX XX   XX XX XX
XX

BIOS Data Area
  40:0000 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 80 9F
  40:0010 :   27 00 00 7E   02 00 00 00   00 00 1E 00   1E 00 00 00
  40:0020 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
  40:0030 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 01 01
  40:0040 :   25 00 04 00   00 29 00 01   02 03 50 00   00 10 00 00
  40:0050 :   00 14 00 00   00 00 00 00   00 00 00 00   00 00 00 00
  40:0060 :   0E 0D 00 D4   03 29 30 C2   11 05 77 00   7F FF 0E 00
  40:0070 :   00 00 00 12   00 01 00 00   14 14 14 14   01 01 01 01
  40:0080 :   1E 00 3E 00   18 10 00 60   F9 11 0B 01   00 00 00 05
  40:0090 :   17 00 00 00   28 00 10 00   00 00 00 00   00 00 00 00
  40:00A0 :   00 00 00 00   00 00 00 00   53 5B 00 C0   00 00 00 00
  40:00B0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
  40:00C0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
  40:00D0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
  40:00E0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00
  40:00F0 :   00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00


Interrupt Vector Table (including INSPECT)
  00 - 03 :   0A6A:0555     0070:0465     07C6:0016     0070:0465
  04 - 07 :   0070:0465     F000:FF54     F000:93CC     F000:9BD0
  08 - 0B :   07C6:001F     07C6:0028     F000:9BD0     F000:9BD0
  0C - 0F :   F000:9BD0     F000:9BD0     07C6:009A     0070:0465
  10 - 13 :   C000:329C     F000:F84D     F000:F841     0070:03EE
  14 - 17 :   F000:E739     0206:0240     0070:042D     F000:EFD2
  18 - 1B :   F000:24F0     0887:002F     F000:FE6E     0070:045F
  1C - 1F :   F000:FF53     C000:1F24     0000:0522     C000:6743
  20 - 23 :   00C9:0FA8     00C9:0FB2     088F:0314     088F:016D
  24 - 27 :   088F:0178     00C9:0FBC     00C9:0FC6     00C9:0FD0
  28 - 2B :   00C9:106F     0070:0466     00C9:106F     00C9:106F
  2C - 2F :   00C9:106F     00C9:106F     088F:0162     0890:01CC
  30 - 33 :   C90F:E4EA     F000:9B00     00C9:106F     00C9:106F
  34 - 37 :   00C9:106F     00C9:106F     00C9:106F     00C9:106F
  38 - 3B :   00C9:106F     00C9:106F     00C9:106F     00C9:106F
  3C - 3F :   00C9:106F     00C9:106F     00C9:106F     1B25:04F5
  40 - 43 :   F000:EC59     0000:0000     F000:F065     C000:6343
  44 - 47 :   F000:9BD0     F000:9BD0     0000:0000     F000:9BD0
  48 - 4B :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
  4C - 4F :   F000:9BD0     F000:9BD0     F000:9BD0     0070:04FC
  50 - 53 :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
  54 - 57 :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
  58 - 5B :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
  5C - 5F :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
  60 - 63 :   0000:0000     0000:0000     0000:0000     0000:0000
  64 - 67 :   0000:0000     0000:0000     0000:0000     0000:0000
  68 - 6B :   F000:9BD0     F000:9BD0     F000:9BD0     F000:9BD0
  6C - 6F :   F000:9BD0     C000:329C     F000:9BD0     F000:9BD0
  70 - 73 :   07C6:0035     F000:9C1F     07C6:00B2     F000:9BD0
  74 - 77 :   07C6:00E2     F000:9C28     07C6:00FA     F000:9BD0
  78 - 7B :   0000:0000     0000:0000     0000:0000     0000:0000
  7C - 7F :   0000:0000     0000:0000     0000:0000     0000:0000
  80 - 83 :   0000:0000     0000:0000     0000:0000     0000:0000
  84 - 87 :   0000:0000     0000:0000     0000:0000     0000:0000
  88 - 8B :   0000:0000     0000:0000     0000:0000     0000:0000
  8C - 8F :   0000:0000     0000:0000     0000:0000     0000:0000
  90 - 93 :   0000:0000     0000:0000     0000:0000     0000:0000
  94 - 97 :   0000:0000     0000:0000     0000:0000     0000:0000
  98 - 9B :   0000:0000     0000:0000     0000:0000     0000:0000
  9C - 9F :   0000:0000     0000:0000     0000:0000     0000:0000
  A0 - A3 :   0000:0000     0000:0000     0000:0000     0000:0000
  A4 - A7 :   0000:0000     0000:0000     0000:0000     0000:0000
  A8 - AB :   0000:0000     0000:0000     0000:0000     0000:0000
  AC - AF :   0000:0000     0000:0000     0000:0000     0000:0000
  B0 - B3 :   0000:0000     0000:0000     0000:0000     0000:0000
  B4 - B7 :   0000:0000     0000:0000     0000:0000     0000:0000
  B8 - BB :   0000:0000     0000:0000     0000:0000     0000:0000
  BC - BF :   0000:0000     0000:0000     0000:0000     0000:0000
  C0 - C3 :   0000:0300     0000:1200     0000:0000     0000:0000
  C4 - C7 :   0000:0000     0000:0000     0000:0000     0000:0000
  C8 - CB :   0000:0000     0000:0000     0000:0000     0000:0000
  CC - CF :   0000:0000     0000:0000     0000:0000     0000:0000
  D0 - D3 :   0000:0000     0000:0000     0000:0000     0000:0000
  D4 - D7 :   0000:0000     0000:0000     0000:0000     0000:0000
  D8 - DB :   0000:0000     0000:0000     0000:0000     0000:0000
  DC - DF :   0000:0000     0000:0000     0000:0000     0000:0000
  E0 - E3 :   0000:0000     0000:0000     EE00:0000     D6BA:EE00
  E4 - E7 :   F886:0040     F886:0006     EF6C:0006     F886:F886
  E8 - EB :   F886:0006     EF6C:0006     0086:F000     0046:0040
  EC - EF :   FB00:0046     0020:0203     0000:0327     2003:10DE
  F0 - F3 :   0008:0202     7C73:C000     03DA:0E06     568B:03CE
  F4 - F7 :   0008:6C73     03C4:0203     03C4:0000     0000:0ED0
  F8 - FB :   0000:03C4     FF10:020F     0000:0000     00F4:91C7
  FC - FF :   0000:5BFC     0000:0A27     62BA:0003     0246:F000

PCI Devices Information
  Signature . . . . . PCI
  Config Mechanism #1 . . . . . Supported
  Config Mechanism #2 . . . . . Not Supported
  Spec Cycle for Config #1 . . . Supported
  Spec Cycle for Config #2 . . . Not Supported
  BIOS Interface Version . . . . 2.10
  Last PCI Bus Number . . . . . 7
  Number of PCI Devices . . . . 9

  Bus Number . . . . . 1
  Device Number . . . . . 6
  Function Number . . . . . 00h
  Slot Number . . . . . 0
```

Vendor ID 1013h
Device ID 00A0h
Revision ID 22h
Device Type . . . . . VGA Compatible Controller
Programming Interface 00h
Expansion ROM Base Address . . FF000000h
IRQ Line 255
IRQ Pin . . . . . INTA#
Memory Address Base 41000000h
Memory Address Length 1000000h
IO Address Base 0h
IO Address Length 400h

Bus Number 1
Device Number 7
Function Number 00h
Slot Number 0
Vendor ID 0E11h
Device ID AE43h
Revision ID 10h
Device Type . . . . . Other Network Controller
Programming Interface 00h
Expansion ROM Base Address . . FFFF0000h
IRQ Line 5
IRQ Pin . . . . . INTA#
IO Address Base 6400h
IO Address Length 10h
Memory Address Base 40001100h
Memory Address Length 10h

Bus Number 1
Device Number 9
Function Number 00h
Slot Number 0
Vendor ID 1000h
Device ID 000Fh
Revision ID 03h
Device Type . . . . . SCSI Bus Controller
Programming Interface 00h
Expansion ROM Base Address . . 0h
IRQ Line 10
IRQ Pin . . . . . INTA#
IO Address Base 6000h
IO Address Length 100h
Memory Address Base 40001000h
Memory Address Length 100h
Memory Address Base 40000000h
Memory Address Length 1000h

Bus Number 2
Device Number 0
Function Number 00h
Slot Number 0
Vendor ID 0E11h
Device ID AE10h
Revision ID 03h
Device Type . . . . . Other Mass Storage Controller
Programming Interface 00h
Expansion ROM Base Address . . FFFF0000h
IRQ Line 4
IRQ Pin . . . . . INTA#
IO Address Base 7000h
IO Address Length 100h
Memory Address Base 42000000h
Memory Address Length 100h
Memory Address Base 44000000h
Memory Address Length 2000000h

Bus Number 3
Device Number 0
Function Number 00h
Slot Number 0
Vendor ID 0E11h
Device ID AE10h
Revision ID 03h
Device Type . . . . . Other Mass Storage Controller
Programming Interface 00h
Expansion ROM Base Address . . FFFF0000h
IRQ Line 9
IRQ Pin . . . . . INTA#
IO Address Base 8000h
IO Address Length 100h
Memory Address Base 46000000h
Memory Address Length 100h
Memory Address Base 48000000h
Memory Address Length 2000000h

Bus Number 4
Device Number 0
Function Number 00h
Slot Number 0
Vendor ID 0E11h
Device ID AE10h
Revision ID 02h
Device Type . . . . . Other Mass Storage Controller
Programming Interface 00h
Expansion ROM Base Address . . FFFF0000h
IRQ Line 11
IRQ Pin . . . . . INTA#
IO Address Base 9000h
IO Address Length 100h
Memory Address Base 4A000000h
Memory Address Length 100h
Memory Address Base 4C000000h
Memory Address Length 2000000h

Bus Number 5
Device Number 0
Function Number 00h
Slot Number 0
Vendor ID 0E11h
Device ID AE10h
Revision ID 03h
Device Type . . . . . Other Mass Storage Controller
Programming Interface 00h
Expansion ROM Base Address . . FFFF0000h
IRQ Line 15
IRQ Pin . . . . . INTA#
IO Address Base A000h
IO Address Length 100h
Memory Address Base 4E000000h
Memory Address Length 100h
Memory Address Base 50000000h
Memory Address Length 2000000h

Bus Number 6
Device Number 0
Function Number 00h
Slot Number 0
Vendor ID 0E11h
Device ID AE10h
Revision ID 03h
Device Type . . . . . Other Mass Storage Controller
Programming Interface 00h
Expansion ROM Base Address . . FFFF0000h
IRQ Line 3
IRQ Pin . . . . . INTA#
IO Address Base B000h
IO Address Length 100h
Memory Address Base 52000000h
Memory Address Length 100h
Memory Address Base 54000000h
Memory Address Length 2000000h

Bus Number 7
Device Number 0
Function Number 00h
Slot Number 0
Vendor ID 0E11h
Device ID AE10h
Revision ID 02h
Device Type . . . . . Other Mass Storage Controller
Programming Interface 00h
Expansion ROM Base Address . . FFFF0000h
IRQ Line 7
IRQ Pin . . . . . INTA#
IO Address Base C000h
IO Address Length 100h
Memory Address Base 56000000h
Memory Address Length 100h
Memory Address Base 58000000h
Memory Address Length 2000000h

Compaq ProLiant 2500 is a trademark of Compaq Computer Corporation.

Microsoft SQL Server 6.5 Configuration Parameters

```
name                                 minimum
maximum     config_value run_value
----------------------------------- ----------- -----
------ ------------ -----------
```

```
affinity mask                   0
2147483647  0           0
allow updates                   0           1
0           0
backup buffer size              1           32
1           1
backup threads                  0           32
6           6
cursor threshold                -1
2147483647  -1          -1
database size                   2           10000
2           2
default language                0           9999
0           0
default sortorder id            0           255
52          52
fill factor                     0           100
0           0
free buffers                    20
524288      1024        1024
hash buckets                    4999
265003      265003      265003
language in cache               3           100
3           3
LE threshold maximum            2
500000      200         200
LE threshold minimum            2
500000      20          20
LE threshold percent            1           100
0           0
locks                           5000
2147483647  5000        5000
LogLRU buffers                  0
2147483647  256         256
logwrite sleep (ms)             -1          500
0           0
max async IO                    1           1024
8           8
max lazywrite IO                1           1024
12          12
max text repl size              0
2147483647  65536       65536
max worker threads              10          1024
40          40
media retention                 0           365
0           0
memory                          2800
1048576     457114      457114
nested triggers                 0           1
1           1
network packet size             512         32767
4096        4096
open databases                  5           32767
10          10
open objects                    100
2147483647  200         200
priority boost                  0           1
0           0
procedure cache                 1           99
2           2
Protection cache size           1           8192
15          15
RA cache hit limit              1           255
4           4
RA cache miss limit             1           255
3           3
RA delay                        0           500
15          15
RA pre-fetches                  1           1000
3           3
RA slots per thread             1           255
5           5
RA worker threads               0           255
0           0
recovery flags                  0           1
0           0
recovery interval               1           32767
32767       32767
remote access                   0           1
0           0
remote conn timeout             -1          32767
10          10
remote login timeout            0
2147483647  5           5
remote proc trans               0           1
0           0
remote query timeout            0
2147483647  0           0
remote sites                    0           256
10          0
resource timeout                5
2147483647  10          10
set working set size            0           1
0           0
show advanced options           0           1
1           1
SMP concurrency                 -1          64
-1          -1
sort pages                      64          511
64          64
spin counter                    1
2147483647  10000       10000
tempdb in ram (MB)              0           2044
2           2
time slice                      50          1000
100         100
user connections                5           32767
4180        4180
user options                    0           4095
0           0

(1 row(s) affected)
```

Appendix D:
180-Day Space

Warehouses:	415
tpmC:	4864

Table	Rows	Data pages (KB)	Index pages (KB)	Overhead	Extra 5%	Total with 5%
Warehouse	415	830	8	0	42	
District	4,150	8,300	38	0	417	*8,*
Item	100,000	9,100	46	0	457	*9,*
Customer	12,450,000	8,301,660	644,328	0	447,299	*9,393,.*
New_order	3,735,000	41,500	254	0	2,088	*43,*
Stock	41,500,000	13,836,100	76,450	0	695,628	*14,608,.*
History (D)	12,450,000	622,502	0	0	0	*622,*
Orders (D)	12,450,000	323,700	1,956	0	0	*325,*
Order_line (D)	124,503,957	6,920,948	45,238	0	0	*6,966,.*
Totals (in MB)		**29,360**	**750**	**0**	**1,119**	**31,**
As loaded	***30,110.31***					6802.916
As needed for 5%	***31,229.38***					1024.882
As needed for 8 hours	**32,678.72**					

DBspaces	# of S	size in MB	Total Allocated	Tables
Master	1	25	25	
Model (included in Master)				
Msdb	1	8	8	
bigdev	*4*	*5,280*	21,120	
bigdev	*1*	*3,080*	3,080	
oldev	*1*	*9,200*	9,200	
miscdev	*1*	*1,400*	1,400	
TOTAL ALLOCATED			**34,833**	

	These are in MB	
Dynamic space	*7,683*	Sum of Data + Bitmap for Order, Order_Line and History
Static space	*23,580*	Sum of all data,index,bitmap (including the rootdbs) + 5% - above dynamic space
Free space	*3,571*	Total space allocated to DBMS - dynamic and static
Daily growth	*1,441*	(Dynamic space/(W * 62.5))* tpmC
Daily spread	*1,410*	Free space - 1.5 * Daily growth (zero if negative)
		This can be reconfigured to eliminate daily spread, zero assumed
180 day space (MB)	*282,911*	Static space + 180 (daily growth + daily spread)
180 day in GB	***276.28***	
8 hr log space (GB)	***12.35*** (excludes RAID)	

	Space needed	Disk size	Disks Priced	GB
180 day space	*276.28 GB*	*3.9974 GB*	*79*	*315.79*
		8.6760 GB	*0*	*0.00*
				315.79
Logical logs (w/ mirrors)	*24.69 GB*	*8.6760 GB*	4	*34.70*
OS, file sys, Swap	*1.50 GB*	*3.9974 GB*	1	*4.00*
Total	***302.47 GB***		**84**	***354.50***

Appendix E:
Third Party Letters

Redmond, WA 98052-6399 Fax 206 936 7329

Microsoft

March 31, 1997

Mr. Michael Nikolaiev
Systems Division
Compaq Computer Corporation
P.O. Box 692000, MS 090308
Houston, TX 77269-2000

via FAX: (713) 514-8375

Dear Mike,

Here is the information you requested regarding pricing of certain Microsoft products:

Microsoft SQL Server 6.5 software, incl 5 CALs	$1399
Microsoft SQL Server Internet Connector License	$2999
Microsoft SQL Workstation (includes programmers toolkit)	$499
Windows NT Server 4.0 software, incl 5 CALs (5 copies @ $809 each)	$4045
Visual C++ 32-bit edition (subscription)	$499
5-yr maintenance for above software @ $2095/yr	$10475

This quote is valid for the next 90 days. Please let me know if I can be of any further assistance.

Best regards,

Sid Arora
Product Manager, Microsoft SQL Server
Personal and Business Systems Group

Advanced Solutions

9777 W. Gulf Bank, #B Suite 1000
Houston, Tx 77040-3113
Phone (713) 849-2828
Fax (713) 849-2850

Item Description	Stock Number	Price Each	QTY	Total	5 Year Servi	Total Price
ProLiant 2500 6/200 - 512K Model 1H	307550-001	$6,655.00	1	$6,655.00	$2,329.25	$8,984.25
ProLiant 2500 6/200-512K Option Kit	300906-001	$2,905.00	1	$2,905.00	$1,016.75	$3,921.75
256 MB DIMM Kit	271910-001	$5,238.00	4	$20,952.00	$7,333.20	$28,285.20
SMART-2/P SCSI Array Controller	194753-001	$2,138.00	6	$12,828.00	$4,489.80	$17,317.80
Enhanced ProLiant Storage System	189600-001	$863.00	12	$10,356.00	$3,624.60	$13,980.60
2.1 GB Pluggable SCSI-2 Drive	199876-001	$744.00	1	$744.00	$260.40	$1,004.40
4.3 GB Pluggable SCSI-2 Drive	146742-006	$1,227.00	79	$96,933.00	$33,926.55	$130,859.55
9.1 GB Pluggable SCSI-2 Drive	199882-001	$2,127.00	4	$8,508.00	$2,977.80	$11,485.80
Compaq Uninteruptible Power Supply T2000	242688-005	$803.00	2	$1,606.00	$632.10	$2,438.10
4/16GB TurboDAT Drive	142161-001	$1,084.00	1	$1,084.00	$379.40	$1,463.40
ProLiant 800 6/160 - Model 4300	273750-003	$3,347.00	3	$10,041.00	$3,514.35	$13,555.35
64-MB Memory Kit (1x64-MB, 60 ns, ECC)	225483-001	$625.00	6	$3,750.00	$1,312.50	$5,062.50
NetFlex 3 PCI Ethernet Controller	169810-001	$162.00	6	$972.00	$340.20	$1,312.20
Compaq V50 Color Monitor	264150-001	$373.00	4	$1,492.00	$522.20	$2,014.20
Microsoft Windows NT Server v. 4.0		$809.00	4	$3,236.00		$3,236.00
Microsoft SQL Server 6.5 plus User Licenses (w		$1,399.00	1	$1,399.00	$10,475.00	$11,874.00
Microsoft Internet Connector License		$2,999.00	1	$2,999.00		$2,999.00
Microsoft SQL Workstation		$499.00	1	$499.00		$499.00
Microsoft Visual C++ v. 4.0		$499.00	1	$499.00		$499.00
NetLux 5-Port 100TX TrueFAST Ethernet Hub	NetLux	$658.00	3	$1,974.00	5 yr warranty	$1,974.00
CentreCOM 24-Port Ethernet Hub	CentreCOM	$215.00	192	$41,280.00	5 yr warranty	$41,280.00
		Totals:		$230,912.00	$73,134.10	$304,046.10

Prices include a large volume discount and may vary when items are purchased seperately.
Service prices based upon a service contract for all items listed.
The above quotation is valid for 60 days.

Jim Cockrill
Systems Consultant
(713) 849-2828