



**TPC Benchmark™ C
Full Disclosure
Report**

**Unisys Corporation
Enterprise Systems**

Aquanta QR/2 Server

using

**Microsoft NT Server Enterprise Edition 4.0
and**

Microsoft SQL Server Enterprise Edition 7.0

**Second Edition
January 4th 1999**

Unisys Part Number 4494 8925-100

Second Edition – January 1999

Unisys Corporation 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. Unisys Corporation 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, Unisys Corporation and Microsoft Corporation provide no warranty on the pricing information in this document.

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 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 was obtained in a rigorously controlled environment, and therefore results obtained in other operating environments may vary significantly. Unisys Corporation and Microsoft Corporation do 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.

Copyright © 1999 Unisys Corporation.

All Rights Reserved. Permission is hereby granted to reproduce this document in whole or in part provided the copyright notice printed above is set forth in full text on the title page of each item reproduced.

Printed in USA, January 1999.

Unisys Corporation Part Number: 4494 8925-100

Unisys and Aquanta are registered trademarks of Unisys Corporation.

Intel, Pentium and Pentium II and Xeon are registered trademarks of Intel Corporation.

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

BEA and Tuxedo are registered trademarks of BEA Systems, Inc.

TPC Benchmark, tpmC and ipmC are trademarks of the Transaction Processing Performance Council.

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

Page	Issue
i through xii	-100
0-1 through 0-3	-000
0-4	Blank
1-1 through 1-1	-000
1-2	Blank
2-1 through 2-2	-000
3-1 through 3-3	-000
3-4	Blank
4-1 through 4-8	-000
5-1 through 5-8	-000
6-1 through 6-2	-000
7-1	-000
7-2	-100
8-1 through 8-1	-000
8-2	Blank
9-1 through 9-3	-000
9-4	Blank
A-1 through A-53	-000
A-54	Blank
B-1 through B-43	-000
B-44	Blank
C-1 through C-76	-000
D-1 through D-3	-000
D-4	Blank
E-1 through E-2	-000
F-1 through F-8	-000

Unisys uses an 11-digit document numbering system. The suffix of the document number (1234 5678-xyz) indicates the document level. The first digit of the suffix (x) designates a revision level; the second digit (y) designates an update level. For example, the first release of a document has a suffix of -100. A suffix of -130 designates the third update to revision 1. The third digit (z) is used to indicate an errata for a particular level and is not reflected in the page status summary.

Overview

This report documents the methodology and results of the TPC Benchmark C (TPC-C) conducted on the Unisys Corporation Aquanta QR/2 server. The operating system on the server was Microsoft Windows NT Server Enterprise Edition 4.0. The DBMS used was Microsoft SQL Server Enterprise Edition 7.0. The operating system on the clients was Microsoft Windows NT Server 4.0. The clients ran Microsoft's Internet Information Server 3.0 and Tuxedo 6.3 CFS for NT.

TPC Benchmark Metrics

The standard TPC Benchmark C metrics, ipmC (transactions per minute), price per ipmC (five year capital cost per measured ipmC), and the availability date are reported as required by the benchmark specification.

Executive Summary

The following pages contain the executive summary results of the benchmark.

Auditor

The benchmark configuration, environment, and methodology used to produce and validate the test results, along with the pricing model used to calculate the cost per ipmC, were audited by Richard Gimarc of Performance Metrics, Inc. to verify compliance with the relevant TPC specification.

Unisys Corporation

**Aquanta QR2 Server
C/S**

TPC-C Rev 3.4
4-Jan-1999

Description	Style	Third Party	Brand Pricing	Unit Price	Qty.	Extended Price	5 Years Maint.
Server Hardware							
SYS: Aquanta QR2, w/ CDRom, 0 Proc, 0MB Mem	QR2000101-BAS			\$10,820	1	\$10,820	\$3,936
PROC: 1x450MHz Pentium II 12MB Cache	XER2450-2MB			\$6,800	4	\$27,200	\$9,648
ACC: Voltage Regulator Module, Processor	XEO24001-VRM			\$46	6	\$276	
MEM: 128 MB Memory Upgrade	DIM5072-128			\$459	32	\$14,688	\$8,448
DISK: 4GB Drive, Ultra SCSI SCA	HDS417-CX1			\$746	1	\$746	\$264
ETHERNET: 100Mbit/sec, PCI 32-bit	ETH10100051-PCI			\$117	1	\$117	
CDROM: 14-32x Speed, SCSI	CDR1432-SI			\$161	1	\$161	\$48
MONITOR: 15-inch Color	EVG2100-P			\$320	1	\$320	
KEYBD: 104 Key Spacemaker	PCK104-SKB			\$31	1	\$31	
MOUSE: 2 Button PS2	PWM1-PS2			\$23	1	\$23	
CTRL: RAID 3-Ch PCI w/ 32MB Mem&Bat, BU+10% spares	DACT164PVX	Mylex		\$2,058	10	\$20,580	\$500
					Subtotal	\$74,962	\$22,844
Storage Hardware							
DISK: 4GB Drive, 10K, SCA + 10% spares	OSD4203-W45			\$625	77	\$48,125	spared
DISK: 9GB Drive, 10K, SCA + 10% spares	OSD9203-W45			\$930	154	\$143,220	spared
CAB: Disk, 7 SCA w/ 050 I/F & Cat-Cbl, 0 Disks, 3U	OSM310050-U05			\$1,345	14	\$18,830	\$10,752
CAB: Disk, 7 SCA w/ 057 I/F, 0 Disks, 3U	OSM310057-U05			\$1,350	14	\$18,900	\$10,752
CAB: Disk, 7 SCA w/ 100 I/F, 0MB, 0 Disks, 3U	OSM310100-U05			\$2,727	2	\$5,454	\$3,456
MEM: 32MB OSM cache	OSM1000-C32			\$150	2	\$300	\$216
PWR: 2nd Power Supply Upgrade, OSM Cabinet	OSM3000-APM			\$261	2	\$522	\$264
PWR: 3000 VA UPS, 3U	UP030001-SXR			\$2,239	1	\$2,239	\$936
PWR: Distribution Unit, 6 OSM Cabinets	RM1-PDU			\$379	6	\$2,274	\$3,600
CBL: SCSI 68-pin HD->YHD Conn's	CBL2210-OSM			\$90	16	\$1,440	
CAB: Rackmount Kit for Disk Cages	OSM3000-RMK			\$84	30	\$2,520	
CAB: Rack Cabinet, w/ fill pnls, 36U	CAB361-SXR			\$1,384	3	\$4,152	
CAB: Bezel Kit 36U	BEZ361-CAB			\$206	3	\$618	
CAB: Stabilizer kit 0U	WGT39681-SXR			\$110	3	\$330	
PNL: L&R side panels 36U	PAN3621-SXR			\$192	3	\$576	
					Subtotal	\$249,500	\$29,976
Server Software							
Microsoft NT Server Enterprise Edition 4.0, Incl 25 CALS		Microsoft		\$3,999	1	\$3,999	\$0
Microsoft SQL Server Enterprise Edition 7.0, unlimited user license		Microsoft		\$28,999	1	\$28,999	\$10,475
					Subtotal	\$32,998	\$10,475
Client Hardware							
SYS: NetServer LC3, w/1 450MHz Proc & CDRom, 0MB Mem	D7026-AV			\$1,910	3	\$5,730	\$3,600
PROC: 1x450MHz Pentium II/512KB Cache UPG	D7032-AV			\$1,138	3	\$3,414	\$1,512
MEM: 128 MB SDRAM Memory Upgrade	D6098-AV			\$358	12	\$4,296	
DISK: 4GB SCSI 3.5 Internal	D4910-AV			\$316	3	\$948	\$1,764
ETHERNET: 10/100TX Mbit/sec, PCI 32-bit	D5013-AV			\$68	15	\$1,020	
ETHERNET: 100Mbit/sec, PCI 32-bit, Quad	SF1001-ET4			\$1,212	3	\$3,636	\$1,584
MONITOR: 15-inch Color	EVG2100-P			\$320	3	\$960	
					Subtotal	\$20,004	\$8,460
Client Software							
Microsoft Windows NT Server 4.0, Incl 5 CALS		Microsoft		\$809	3	\$2,427	\$0
Microsoft Visual C++ Professional 5.0		Microsoft		\$499	1	\$499	\$0
TUXEDO Core Functional Services 6.3 for NT		BEA		\$3,000	3	\$9,000	\$6,750
					Subtotal	\$11,926	\$6,750
User Connectivity							
Ethernet Hub, 8-Port 100TX TrueFast + 10% spares	NX-H8TX	Netlux		\$205	3	\$615	spared
Ethernet Hub, 8-Port 10Base-T + 1-Port BNC + 10% spares	DEH2924	Compex		\$33	2353	\$77,649	spared
					Subtotal	\$78,264	\$0
					Total	\$467,654	\$78,505
						(\$32,389)	(\$7,094)

Notes:

1. HW Maintenance - First 36 months that are included in Unisys warranty are upgraded to service level: Standard Performance-Gold. Last 24 months are also at service level: Standard Performance-Gold.
2. All Microsoft maintenance is covered by the maintenance cost of Microsoft SQL Server.
3. 10% or minimum 2 spares are added in place of onsite service (products have a five year return-to-vendor warranty)
4. Pricing: 1 = Western Micro, 2 = Mylex, 3 = Microsoft, 4 = BEA, 5 = Netlux, 6 = DataComm/Whse

The benchmark results and test methodology were audited by Richard Gimarc of Performance Metrics, Inc.

Pricing 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 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 benchmarks 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 \$506,676
TPC-C Throughput 21,354.55
\$/tpmc \$23.73

NUMERICAL QUANTITIES SUMMARY

Unisys Aquanta QR/2 Server

Microsoft SQL Server Enterprise Edition 7.0

MQTl, Computed Maximum Qualified Throughput: **21354.55**
 % throughput difference, reported & reproducibility runs: 0.25%

Transaction Mix

New Order	44.88%
Payment	43.06%
Delivery	4.04%
Stock-Level	4.01%
Order-Status	4.02%

Response Times

Transaction	Average	Maximum	90th %ile
New-Order	0.43	4.10	0.65
Payment	0.29	3.88	0.49
Delivery	0.12	0.71	0.12
Stock-Level	2.23	5.53	3.01
Order Status	0.33	3.79	0.54
Menu	0.12	3.06	0.12
Delivery (Deferred)	0.51	2.33	0.76

Response time delay added for emulated components (seconds)

RT Response time	0.1
Menu Response time	0.1

Keying/Think Time Times (seconds)

Transaction	Minimum	Average	Maximum
New-Order	18.00/0	18.01/12.04	18.07/120.3
Payment	3.00/0	3/12.03	3.04/120.31
Delivery	2.00/0	2/5.03	2.04/50.6
Stock-Level	2.00/0	2/5.01	2.04/44.85
Order-Status	2.00/0	2/10.1	2.02/100.71

Test Duration

Ramp up time	23 minutes
Measurement interval (M)	20 minutes
Transactions (all types) completed during measurement interval	951,653
Ramp-down time	39 minutes

Checkpointing:

Number of checkpoints	1
Checkpoint interval	20 minutes

Table of Contents

Abstract	iv
Table of Contents	viii
Preface	xii
0. General Items.....	0-1
0.1. Order and Titles.....	0-1
0.2. Executive Summary Statement.....	0-1
0.3. Numerical Quantities Summary.....	0-1
0.4. Application Code Disclosure.....	0-1
0.5. Benchmark Sponsor	0-2
0.6. Parameter Settings.....	0-2
0.7. Configuration Diagrams	0-2
1. Clause 1: Logical Database Design	1-1
1.1. Table Definitions.....	1-1
1.2. Physical Organization of the Database	1-1
1.3. Insert and/or Delete Operations.....	1-1
1.4. Partitioning.....	1-1
1.5. Replication, Duplication or Additions.....	1-1
2. Clause 2: Transaction & Terminal Profiles	2-1
2.1. Random Number Generation.....	2-1
2.2. Input/Output Screen Layout	2-1
2.3. Priced Terminal Feature Verification	2-1
2.4. Presentation Managers or Intelligent Terminal	2-1
2.5. Transaction Statistics.....	2-1
2.6. Queuing Mechanism of Delivery.....	2-2
3. Clause 3: Transaction & System Properties	3-1
3.1. Transaction System Properties (ACID).....	3-1
3.2. Atomicity.....	3-1
3.2.1. Completed Transaction.....	3-1
3.2.2. Aborted Transactions	3-1
3.3. Consistency	3-1
3.4. Isolation.....	3-2

3.5.	Durability	3-2
3.5.1.	Loss of Log Disk and Loss of Data Disk.....	3-2
3.5.2.	Instantaneous Interruption and Loss of Memory	3-3
4.	Clause 4: Scaling & Database Population	4-1
4.1.	Initial Cardinality of Tables	4-1
4.2.	Constant Values.....	4-1
4.3.	Database Layout.....	4-2
4.4.	DBMS: Data Model and DBMS Interface/Access Language.....	4-2
4.5.	DBMS Partitions/Replications	4-2
4.6.	DBMS Space Requirements.....	4-2
5.	Clause 5: Performance Metrics & Response Time.....	5-1
5.1.	Measured Throughput (tpmC).....	5-1
5.2.	Response Times	5-1
5.3.	Keying and Think Times.....	5-1
5.4.	Response Time-Frequency Distribution Curves.....	5-2
5.5.	New Order Think Time-Frequency Distribution Curve.....	5-4
5.6.	Response Time versus Throughput Performance Curve	5-5
5.7.	New-Order Throughput vs. Time	5-5
5.8.	Determination of “Steady State”	5-6
5.9.	Work Performed During Steady State	5-6
5.10.	Reproducibility	5-7
5.11.	Measurement Interval Duration.....	5-7
5.12.	Regulation of Transaction Mix.....	5-7
5.13.	Transaction Statistics.....	5-7
5.14.	Checkpoint Statistics	5-8
6.	Clause 6: SUT, Driver & Communications Definition	6-1
6.1.	Remote Terminal Emulator (RTE) Description	6-1
6.2.	Emulated Components	6-1
6.3.	Functional Diagrams	6-1
6.4.	Network Configuration.....	6-1
6.5.	Network Bandwidth	6-1
6.6.	Operator Intervention	6-2
7.	Clause 7: Pricing.....	7-1
7.1.	Pricing.....	7-1
7.1.1.	System Pricing.....	7-1
7.1.2.	Maintenance Pricing.....	7-1
7.1.3.	Discounts.....	7-1
7.2.	Availability.....	7-2
7.3.	Measured tpmC, Price/Performance, and Availability Date	7-2

7.4. Country-Specific Pricing.....	7-2
7.5. Usage Pricing	7-2
8. Clause 8 : Full Disclosure Availability.....	8-1
8.1. Availability.....	8-1
9. Clause 9 : Audit	9-1
9.1. Auditor's Report.....	9-1
Appendix A - Client/Server Source	A-1
Appendix B - Database Design.....	B-1
Appendix C - Tunable Parameters	C-1
Appendix D - RTE Code.....	D-1
Disk Storage	E-1
Appendix F - Third-Party Price Quotations	F-1

Figures

Figure 0.1: Benchmarked Configuration	0-3
Figure 0.2: Priced Configuration	0-3
Figure 5.1: New Order Response Time Distribution	5-2
Figure 5.2: Payment Response Time Distribution	5-2
Figure 5.3: Order Status Response Time Distribution	5-3
Figure 5.4: Delivery Response Time Distribution	5-3
Figure 5.5: Stock Level Response Time Distribution	5-4
Figure 5.6: New Order Think Time Distribution	5-4
Figure 5.7: Response Time versus Throughput	5-5
Figure 5.8: Throughput (rpmC) versus Time	5-5

Tables

Table 4.1: Initial Cardinality of Database Table	4-1
Table 4.2: Constant C for NURand	4-1
Table 4.3: Disk Cage Configuration	4-3
Table 4.4: RAID Adapter Disk Configuration	4-5
Table 4.5: Disk Administrator Configuration	4-8
Table 5.1: Response Time Data	5-1
Table 5.2: Keying Times	5-1
Table 5.3: Think Times	5-1
Table 5.4: Transaction Statistics	5-8

Document Structure

The TPC Benchmark C Standard Specification requires test sponsors to publish, submit to the TPC, and make available to the public, a full disclosure report for any result to be considered compliant with the specification. The required contents of the full disclosure report are specified in Clause 8.

This report is submitted to satisfy the specification's requirement for full disclosure. It documents the compliance of the benchmark implementation and execution reported for the Unisys Corporation Aquanta QR/2 Server using Microsoft Windows NT 4.0 Enterprise Edition and Microsoft SQL Server Enterprise Edition 7.0.

TPC Benchmark C Overview

The TPC Benchmark™ C Standard Specification Revision 3.4 was developed by the Transaction Processing Council (TPC). It is the intent of the TPC to develop a suite of benchmarks to measure the performance of computer systems executing a wide range of applications. Unisys and Microsoft Corporations are active participants in the TPC to define and develop such a suite of benchmarks.

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

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

The performance metric reported by TPC-C is a "business throughput" measuring the number of orders processed per minute. Multiple transactions are used to simulate the business activity of processing an order, and each transaction is subject to a response time constraint. The performance metric for this benchmark is expressed in transactions-per-minute-C (tpmC). To be compliant with the TPC-C standard, all references to tpmC results must include the tpmC rate, the associated price-per-tpmC, and the availability date of the priced configuration.

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

0.

General Items

0.1. Order and Titles

The order and titles of sections in the Test Sponsor's Full Disclosure report must correspond with the order and titles of sections from the TPC-C standard specification (i.e., this document). The intent is to make it as easy as possible for readers to compare and contrast material in different Full Disclosure reports.

The order and titles of the sections in this report correspond with those from the TPC-C standard specification.

0.2. Executive Summary Statement

The TPC Executive Summary Statement must be included near the beginning of the Full Disclosure report.

The TPC Executive Summary Statement is included near the beginning of this report.

0.3. Numerical Quantities Summary

The numerical quantities listed below must be summarized near the beginning of the Full Disclosure Report :

- *measurement interval in minutes,*
- *number of checkpoints in the measurement interval,*
- *checkpoint interval in minutes,*
- *number of transactions (all types) completed within the measurement interval,*
- *computed Maximum Qualified Throughput in tpmC,*
- *percentage difference between reported throughput and throughput obtained in reproducibility run,*
- *ninetieth percentile, average and maximum response times for the New-Order, Payment, Order-Status, Stock-Level, Delivery (deferred and interactive) and Menu transactions,*
- *time in seconds added to response time to compensate for delays associated with emulated components,*
- *percentage of transaction mix for each transaction type.*

These numerical quantities are summarized near the beginning of this report.

0.4. Application Code Disclosure

The applicable 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 and output functions.

Appendix A contains the client application code used in this TPC-C benchmark. Appendix B contains the SQL stored procedures which implement the TPC-C transactions.

0.5. Benchmark Sponsor

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

This TPC benchmark C was sponsored by Unisys Corporation. The benchmark test was developed by Microsoft and Unisys. The benchmark was conducted at Unisys, Mission Viejo, California.

0.6. 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 but not limited to:

- *Data Base tuning options*
- *Recovery/commit options*
- *Consistency/locking options*
- *Operating system and application configuration parameters*

Appendix C contains the configuration and system parameters used in running these TPC-C tests. It also contains all the client and server OS, and SQL Server tunable parameters.

0.7. Configuration Diagrams

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

- *Number and type of processors.*
- *Size of allocated memory, and any specific mapping/partitioning of memory unique to the test.*
- *Number and type of disk units (and controllers, if applicable).*
- *Number of channels or bus connections to disk units, including their protocol type.*
- *Number of LAN (e.g., Ethernet) connections, including routers, workstations, terminals, etc., that were physically used in the test or are incorporated into the pricing structure (see Clause 8.1.8).*
- *Type and the run-time execution location of software components (e.g., DBMS, client processes, transaction monitors, software drivers, etc.).*

The Remote Terminal Emulator (RTE) software used for these TPC-C tests is proprietary to Unisys. The benchmarked configuration of the RTE and Aquanta QR/2 server is illustrated in Figure 0.1. Tables 4.3, 4.4 and 4.5 contain a detailed explanation of the disk configuration.

The priced configuration for the Aquanta QR/2 server is shown in Figure 0.2.

Figure 0.1: Benchmarked Configuration

Aquantia QR/2 Server - Benchmarked Configuration

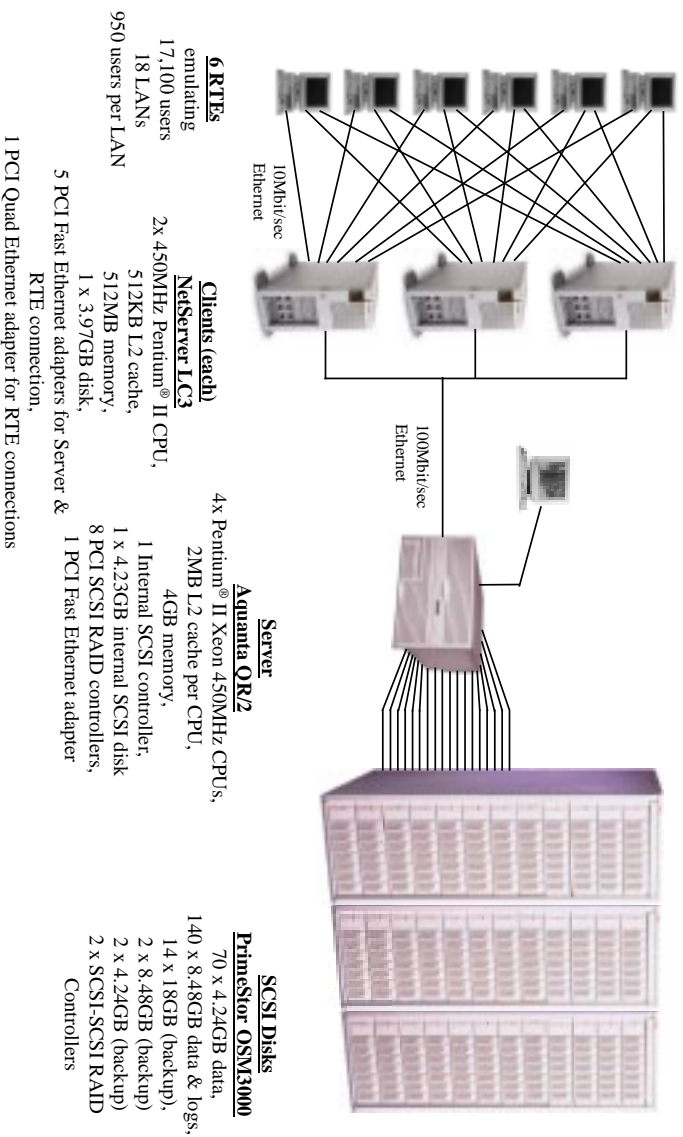
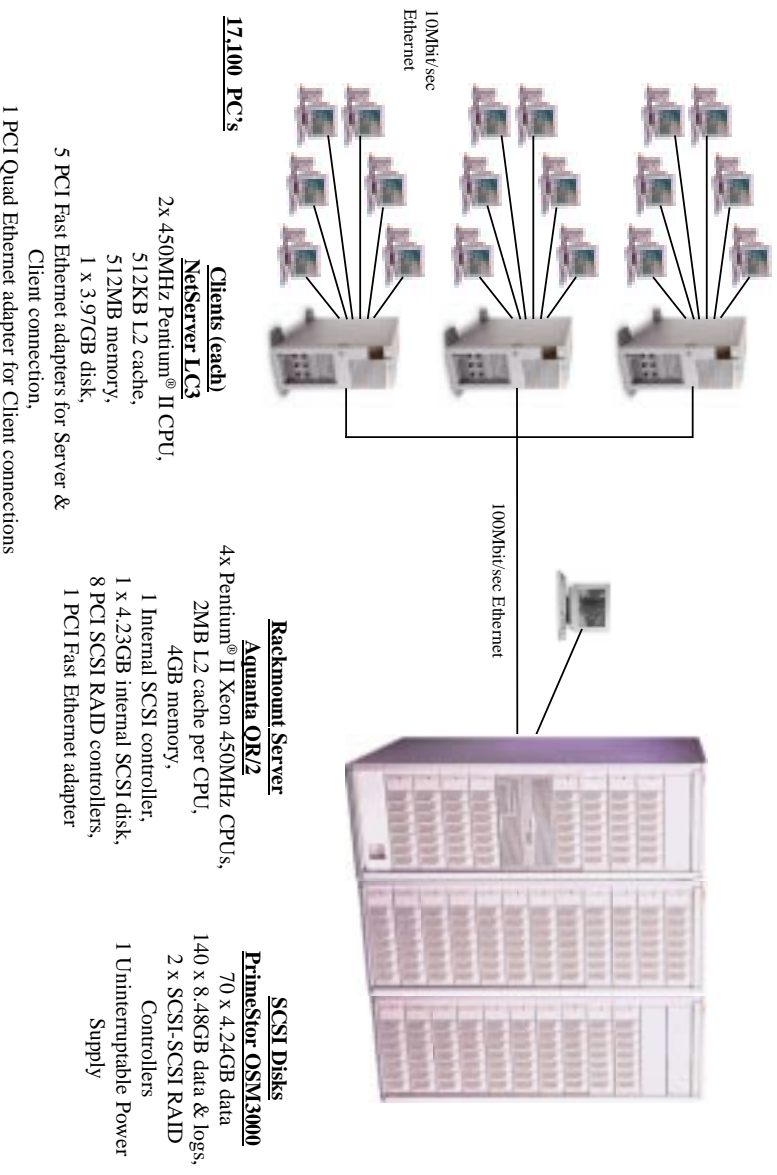


Figure 0.2: Priced Configuration

Aquantia QR/2 Server - Priced Configuration



1.

Clause 1: Logical Database Design

1.1. Table Definitions

Listings must be provided for all table definition statements and all other statements used to setup the data base.

Appendix B contains the SQL definitions of all the required database files, filegroups, tables, indexes and stored procedures, plus a listing of the program used to load the database and establish the required initial populations of each table.

1.2. Physical Organization of the Database

The physical organization of tables and indices, within the data base, must be disclosed.

The disk space was allocated to SQL Server according to the data in Table 4.4. The SQL definitions are contained in Appendix B.

1.3. Insert and/or Delete Operations

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

There were no restrictions on insert and/or delete operations to any of the tables.

1.4. Partitioning

While there are 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 for any table in this implementation.

1.5. Replication, Duplication or Additions

Replication of tables, if used, must be disclosed.

Additional and/or duplicate 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 implementation.

2. Clause 2: Transaction & Terminal Profiles

2.1. Random Number Generation

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

The drivers used the Unisys RTE program, which was independently audited. The initial population of the database was performed by the loader program from V4.01 of the Microsoft TPC-C toolkit, which was also independently audited. Furthermore, the auditor sampled various initial and runtime distributions produced by this implementation to verify correctness.

2.2. Input/Output Screen Layout

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

The screen layouts are based on those in Clauses 2.4.3, 2.5.3, 2.6.3, 2.7.3, and 2.8.3 of the TPC Benchmark C Standard Specification. There are some minor differences in appearance due to the use of a web client implementation.

2.3. 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).

This was verified by the auditor.

2.4. Presentation Managers or Intelligent Terminal

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

Application code running on the client implemented the TPC-C user interface. A listing of this code is included in Appendix A. No presentation manager was used on the client, as screen manipulation and data input/output was handled for each user by the Microsoft Internet Explorer web browser running on each user PC.

2.5. Transaction Statistics

The percentage of New-Order transactions that were rolled back as a result of an unused item number must be disclosed.

The number of items per order entered by New-Order transactions must be disclosed.

The percentage of home and remote Payment transactions must be disclosed.

The percentage of Payment and Order-Status transactions that used non-primary key (C_LAST) access to the database must be disclosed.

The percentage of Delivery transactions that were skipped as a result of an insufficient number of rows in the NEW-ORDER table must be disclosed.

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

Table 5.4 in Section 5 contains all these statistics.

2.6. Queuing Mechanism of Delivery

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

Tuxedo provides the queue for delivery servers. The client application process posts delivery transactions to the delivery queue using a Tuxedo asynchronous call with the TPNOReply option. Upon return from this call, the client application provides a 'delivery queued' response to the user. Delivery servers independently retrieve messages from their queue, submit them to the data base for processing, and log the result to a file upon completion. The source code for this delivery process is included in Appendix A.

3. Clause 3: Transaction & System Properties

3.1. 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.

The TPC Benchmark C Standard Specification defines a set of transaction processing system properties that a system under test (SUT) must support during the execution of the benchmark. Those properties are Atomicity, Consistency, Isolation, and Durability (ACID).

This section defines each of these properties, describes the steps taken to ensure that they were present during the test and describes a series of tests done to demonstrate compliance with the specification. All ACID property tests were executed successfully.

3.2. Atomicity

The system under test must guarantee that data base 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.

3.2.1. Completed Transaction

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number) and verify that the records in the CUSTOMER, DISTRICT, and WAREHOUSE tables have been changed appropriately.

The balances from a randomly selected warehouse, district, and customer row were retrieved by customer number from a script. A Payment transaction was submitted with the same warehouse, district and customer identifiers for a known amount. After completion of the Payment transaction, the balances of the selected warehouse, district, and customer were again retrieved to verify that the changes had been made correctly.

3.2.2. Aborted Transactions

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number) and substitute a ROLLBACK of the transaction for the COMMIT of the transaction. Verify that the records in the CUSTOMER, DISTRICT, and WAREHOUSE tables have NOT been changed.

The balances from a randomly selected warehouse, district, and customer row were retrieved by customer number from a script. A Payment transaction was submitted with the same warehouse, district and customer identifiers that issued a ROLLBACK command rather than a COMMIT. After the transaction completed, the balances of the selected warehouse, district, and customer were again retrieved to verify that no changes had been made to the database.

3.3. Consistency

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

The benchmark specification requires explicit demonstration of the following four consistency conditions:

1. The sum of the district balances in a warehouse is equal to the warehouse balance;
 2. For each district, the next order id minus one is equal to maximum order id in the ORDER table and equal to the maximum new order id in the NEW ORDER table;
 3. For each district, the maximum order id minus minimum order id in the ORDER table plus one equals the number of rows in the NEW-ORDER table for that district;
 4. For each district, the sum of the order line counts in the ORDER table equals the number of rows in the ORDER-LINE table for that district;
- In order to demonstrate this consistency, the following steps were taken:
1. Prior to the start of a benchmark run, the consistency of the database was verified by testing successfully conditions 1-4 described above with a script.
 2. A run under full user load was executed for over 10 minutes with a checkpoint during the run.
 3. After completion of that test, the consistency of the database was again verified by successfully testing using the same consistency script as in step 1.

3.4. 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.

The benchmark specification defines seven required tests to be performed to demonstrate that required levels of transaction isolation are met. These tests, described in Clauses 3.4.2.1 - 3.4.2.7, were all performed from a script and verified by the auditor. In Isolation Test 7, Case A was observed. In addition, the phantom tests and stock level tests were executed and verified to be successful.

3.5. Durability

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

Three durability tests were executed to satisfy the requirements of the specification. The test for loss of memory and instantaneous interruption was combined and performed with a fully scaled database with 17,100 emulated users. The loss of log and loss of data tests were performed on the same system, using a ten warehouse database with 100 emulated users. To the best of our knowledge, these tests prove that the fully scaled configuration used for the throughput test would also meet all durability tests.

3.5.1. Loss of Log Disk and Loss of Data Disk

The following steps were taken (using a ten warehouse database on the same system) to demonstrate durability in the case of loss of a log disk and of data disk. The same log disks and controllers were used for the log as for the fully scaled database. Two separate data disks were on each of two existing controllers.

1. The database was backed up to extra disks on a dump device.
2. The D_NEXT_O_ID fields for all rows in the district table were summed up to determine the initial count of orders present in the database.
3. The RTE was started with 100 users. On the driver systems, committed and rolled back New-Order transactions were recorded in a “success” file.
4. After ten minutes of running at steady state, a hot-pluggable log disk was removed from the disk cabinet, with no effect on NT or SQL server.

5. After 5 additional minutes of operation, a hot-pluggable data disk was removed from the disk cabinet.
6. NT and SQL Server encountered IO errors due to the missing disk and recorded these errors in the NT event log and SQL Server error log, respectively. The RTEs also recorded errors.
7. First, the RTEs and clients were stopped, then SQL Server was used to take a dump of the transaction log to the dump device.
8. Next, SQL server was shutdown, then restarted, and scripts were executed to drop the database and all its devices. Then, SQL Server was shutdown again and the SUT shutdown.
9. A data disk was inserted in the disk cabinet to replace the one removed. The RAID controller was used to recreate the stripe set containing the new data disk. (The missing log drive was not replaced.)
10. The SUT was restarted, and Disk Administrator was used to assign the proper drive letter to the new volume. SQL Server was then restarted and a new (empty) database created as part of the restore database process. That process loaded the initial database into the new database, but did not perform any recovery. Next the transaction log was restored, followed by transaction recovery. The latter step restored all committed transactions to the database.
11. Consistency condition 3 of Clause 3.3.2.3 was executed to verify database consistency.
12. Step 2 was repeated to determine the total number of orders. This number was subtracted from the count obtained previously in Step 2 to determine the number of additional orders added to the database.
13. The contents of the “success” files on the drivers were sampled to verify that the records in the “success” file for committed New-Order transactions had corresponding records in the ORDER. Moreover, the counts were matched with those obtained in step 12.

3.5.2. Instantaneous Interruption and Loss of Memory

Instantaneous interruption and loss of memory tests were combined because the loss of power erased the contents of memory. This failure was induced by removing the primary power to the System Under Test while the benchmark was executing.

1. The D_NEXT_O_ID fields for all rows in the district table were summed up to determine the initial count of orders present in the database (count1).
2. On the driver systems, committed and rolled back New-Order transaction were recorded in a “success” file.
3. The benchmark was executed at full load with all emulated users for a minimum of 10 minutes.
4. Shortly after execution of a checkpoint completed, the system’s primary power was turned off.
5. After transaction failures were noted by the RTEs, the RTEs and clients were shutdown.
6. Power was restored to the SUT, the system rebooted, SQL Server was restarted, and automatic database recovery was performed. The database recovery used the transaction log to reapply all committed transactions and rollback any (in progress) uncommitted transactions, so that the database disks were correct.
7. After recovery finished, Consistency Condition of Clause 3.3.2.3 (no gaps in NO_O_ID) was executed to verify that the database was consistent..
8. Next, samples of the contents of the “success” file from the drivers were compared against corresponding rows of the ORDER table to verify that records in the “success” file for committed New-Order transactions had corresponding records in the ORDER table.
9. Finally, step 1 was repeated to determine the total number of orders (count2). Count2 minus count1 was not less than the number of committed New-Order records in the “success” file.

4.

Clause 4: Scaling & Database Population

4.1. Initial Cardinality of Tables

The Cardinality (e.g., the 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 (see Clause 4.2.2 and the Auditor's attestation letter) the cardinality of the WAREHOUSE table as initially configured and the number of rows deleted must be disclosed.

The TPC-C database for this test was configured with 1,728 warehouses. The cardinality of each table in the database is listed in Table 4.1

Table 4.1: Initial Cardinality of Database Table

Table	Occurrences
Warehouse	1,728
District	17,280
Customer	51,840,000
History	51,840,000
Order	51,840,000
New-Order	15,552,000
Order Line	518,403,182
Stock	172,800,000
Item	100,000

18 rows were deleted from the warehouse table before executing the measurement runs.

4.2. Constant Values

The following values were used as the constant C input values to the NURand function during Build and Run time for this implementation.

Table 4.2: Constant C for NURand

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

4.3. Database Layout

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

Tables 4.3, 4.4 and 4.5 list the distribution of the database over 140 disks and the transaction log over 7 mirrored pairs of disks for the benchmark configuration. In addition, there was one disk containing Windows NT Enterprise Edition and SQL Server Enterprise Edition code and the Master database plus the paging file. Database backup used 14 extra disks. The 10 warehouse durability database used 4 extra disks. These 18 disks were unused during the benchmark and were excluded from the priced configuration.

4.4. DBMS: Data Model and DBMS Interface/Access Language

A statement must be provided that describes:

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

Microsoft SQL Server Enterprise Edition 7.0 is a relational DBMS.

The client software interfaced to SQL Server through Stored Procedures invoked through Remote Procedure Calls embedded in the C application code. Specifically, DBLIB and TCP/IP sockets were used.

4.5. DBMS Partitions/Replications

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

No table partitioning or replication was done.

4.6. DBMS Space Requirements

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

Appendix E lists the space requirements for the 180-day space as well as the logical log space for eight hours.

Table 4.3: Disk Cage Configuration

Disk Cage Configuration																
Adapter	Channel	Id	Id	Id	Id	Id	Id	Id	Id	Id	Id	Id	Rack #			
1	0	0	1	2	3	4	5	6	9GB	9GB	9GB	9GB	1			
		8	9	10	11	12	13	14	9GB	9GB	9GB	9GB	2			
		1	0	1	2	3	4	5	6	9GB	9GB	9GB	9GB	3		
			8	9	10	11	12	13	14	9GB	9GB	9GB	9GB	4		
		2	0	0	1	2	3	4	5	6	9GB	9GB	9GB	9GB	5	
				8	9	10	11	12	13	14	9GB	9GB	9GB	9GB	6	
	1			0	1	2	3	4	5	6	9GB	9GB	9GB	9GB	7	
				8	9	10	11	12	13	14	9GB	9GB	9GB	9GB	8	
	3		0	0	1	2	3	4	5	6	4GB	4GB	4GB	4GB	9	
				8	9	10	11	12	13	14	4GB	4GB	4GB	4GB	10	
				1	0	1	2	3	4	5	6	4GB	4GB	4GB	4GB	11
					8	9	10	11	12	13	14	4GB	4GB	4GB	4GB	12
		4		0	0	1	2	3	4	5	6	9GB	9GB	9GB	9GB	13
					0	1	2	3	4	5	6	9GB	9GB	9GB	9GB	14
5			0	0	1	2	3	4	5	6	4GB	4GB	4GB	4GB	15	
				8	9	10	11	12	13	14	4GB	4GB	4GB	4GB	16	
		1	0	1	2	3	4	5	6	9GB	9GB	9GB	9GB	17		
			8	9	10	11	12	13	14	9GB	9GB	9GB	9GB	18		

Disk Cage Configuration

Adapter	Channel	Id	Id	Id	Id	Id	Id	Id	Id	Id	Id	Rack #				
6	0	0	1	2	3	4	5	6								
		4GB	4GB	4GB	4GB	4GB	4GB	4GB	4GB					19		
		8	9	10	11	12	13	14						14		
		4GB	4GB	4GB	4GB	4GB	4GB	4GB	4GB					20		
		1	0	1	2	3	4	5	6							
		4GB	4GB	4GB	4GB	4GB	4GB	4GB	4GB					21		
	1	8	9	10	11	12	13	14								
			4GB	4GB	4GB	4GB	4GB	4GB	4GB					22		
			2	3	4	5	6									
			18GB	18GB	18GB	18GB	18GB	18GB	18GB							
		2	empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	*		
				empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	*		
			8	empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	*	
					empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	*	
7	0		0	1	2	3	4	5	6							
			9GB	9GB	9GB	9GB	9GB	9GB	9GB	9GB					23	
		8	9	10	11	12	13	14								
		9GB	9GB	9GB	9GB	9GB	9GB	9GB	9GB					24		
		1	0	1	2	3	4	5	6							
		9GB	9GB	9GB	9GB	9GB	9GB	9GB	9GB					25		
		8	9	10	11	12	13	14								
		9GB	9GB	9GB	9GB	9GB	9GB	9GB	9GB					26		
		2	0	1	2	3	4	5	6							
		18GB	18GB	18GB	18GB	18GB	18GB	18GB	18GB					*		
		8	empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	*	
				empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	*	
	0			9GB	8	9	10	11	12	13	14					
					9GB	9GB	9GB	9GB	9GB	9GB	9GB	9GB				27
	1			9GB	8	9	10	11	12	13	14					
					9GB	9GB	9GB	9GB	9GB	9GB	9GB	9GB				28
			2	9GB	0	1	2	3	4	5	6					
					9GB	9GB	9GB	9GB	9GB	9GB	9GB	9GB				29
				8	9GB	8	9	10	11	12	13	14				
						9GB	9GB	9GB	9GB	9GB	9GB	9GB	9GB			
	2		18GB	0	1	2	3	4	5	6						
				18GB	18GB	18GB	18GB	18GB	18GB	18GB	18GB				*	
		8	empty	8	9	10	11	12	13	14						
				empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	empty	*	

Table 4.4: RAID Adapter Disk Configuration

RAID Adapter Disk Configuration						
Adapter	ID	Channel 0	Channel 1	Channel 2	RAID Configuration	Drive Letters
1	0	A0	C0		Arrange Packs A - D as RAID 0	E: and S:
	1	A1	C1			
	2	A2	C2			
	3	A3	C3			
	4	A4	C4			
	5	A5	C5			
	6	A6	C6			
	8	B0	D0			
	9	B1	D1			
	10	B2	D2			
	11	B3	D3			
	12	B4	D4			
	13	B5	D5			
	14	B6	D6			
	2	0	A0	C0		
1		A1	C1			
2		A2	C2			
3		A3	C3			
4		A4	C4			
5		A5	C5			
6		A6	C6			
8		B0	D0			
9		B1	D1			
10		B2	D2			
11		B3	D3			
12		B4	D4			
13		B5	D5			
14		B6	D6			
3		0	A0	C0		Arrange Packs A - D as RAID 0
	1	A1	C1			
	2	A2	C2			
	3	A3	C3			
	4	A4	C4			
	5	A5	C5			
	6	A6	C6			
	8	B0	D0			
	9	B1	D1			
	10	B2	D2			
	11	B3	D3			
	12	B4	D4			
	13	B5	D5			
	14	B6	D6			

RAID Adapter Disk Configuration

Adapter	ID	Channel 0	Channel 1	Channel 2	RAID Configuration	Drive Letters
4	0	A0	A1		Arrange Pack A as RAID 1 (log)	L:
	1					
	2					
	3					
	4					
	5					
	6					
5	0	A0	C0		Arrange Packs A - D as RAID 0	H: and P:
	1	A1	C1			
	2	A2	C2			
	3	A3	C3			
	4	A4	C4			
	5	A5	C5			
	6	A6	C6			
	8	B0	D0			
	9	B1	D1			
	10	B2	D2			
	11	B3	D3			
	12	B4	D4			
	13	B5	D5			
	14	B6	D6			
	6	0	A0	C0		
1		A1	C1			
2		A2	C2			
3		A3	C3			
4		A4	C4			
5		A5	C5			
6		A6	C6			
8		B0	D0			
9		B1	D1			
10		B2	D2			
11		B3	D3			
12		B4	D4			
13		B5	D5			
14		B6	D6			

RAID Adapter Disk Configuration

Adapter	ID	Channel 0	Channel 1	Channel 2	RAID Configuration	Drive Letters
7	0	A0	C0	E0	Arrange Packs A - D as RAID 0	J: and N:
	1	A1	C1	E1		
	2	A2	C2	E2		
	3	A3	C3	E3		
	4	A4	C4	E4		
	5	A5	C5	E5		
	6	A6	C6	E6	Arrange Pack E as RAID 5 (backup)	T:
	8	B0	D0			
	9	B1	D1			
	10	B2	D2			
	11	B3	D3			
	12	B4	D4			
	13	B5	D5			
	14	B6	D6			
8	0	A0	C0	E0	Arrange Packs A - D as RAID 0	K: and M:
	1	A1	C1	E1		
	2	A2	C2	E2		
	3	A3	C3	E3		
	4	A4	C4	E4		
	5	A5	C5	E5		
	6	A6	C6	E6	Arrange Pack E as RAID 5 (backup)	U:
	8	B0	D0			
	9	B1	D1			
	10	B2	D2			
	11	B3	D3			
	12	B4	D4			
	13	B5	D5			
	14	B6	D6			

Table 4.5: Disk Administrator Configuration

Disk Administrator Configuration			
Disk 0 4338 MB	C: SYSTEM FAT 2047 MB	Z: testfiles NTFS 2291 MB	unused 0 MB
Disk 1 243124 MB	E: unknown 15005 MB	S: unknown 7005 MB	unused 221114 MB
Disk 2 243124 MB	F: unknown 15005 MB	R: unknown 7005 MB	unused 221114 MB
Disk 3 121492 MB	G: unknown 15005 MB	Q: unknown 7005 MB	unused 99482 MB
Disk 4 607807 MB	L: unknown 55000 MB	P: unknown 7005 MB	unused 5780 MB
Disk 5 182308 MB	H: unknown 15005 MB	O: unknown 7005 MB	unused 160298 MB
Disk 6 121492 MB	I: unknown 15005 MB	N: unknown 7005 MB	unused 99482 MB
Disk 7 243124 MB	J: unknown 15005 MB	M: unknown 7005 MB	unused 221114 MB
Disk 8 104532 MB	T: BACK1 NTFS 104532 MB		unused 0 MB
Disk 9 243124 MB	K: unknown 15005 MB		unused 221114 MB
Disk 10 104532 MB	U: BACK2 NTFS 104532 MB		unused 0 MB
CD-ROM 0	D:		

5. Clause 5: Performance Metrics & Response Time

5.1. Measured Throughput (tpmC)

Measured tpmC must be reported.

The measured tpmC was 21,354.55.

5.2. 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 5.1 : Response Time Data

Transaction	Average	Maximum	90th %ile
New-Order	0.43	4.10	0.65
Payment	0.29	3.88	0.49
Delivery	0.12	0.71	0.12
Stock-Level	2.23	5.53	3.01
Order Status	0.33	3.79	0.54
Menu	0.12	3.06	0.12
Delivery (Deferred)	0.51	2.33	0.76

5.3. Keying and Think Times

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

Table 5.2: Keying Times

Transaction	Minimum	Average	Maximum
New-Order	18.00	18.01	18.07
Payment	3.00	3.00	3.04
Delivery	2.00	2.00	2.04
Stock-Level	2.00	2.00	2.04
Order Status	2.00	2.00	2.02

Table 5.3: Think Times

Transaction	Minimum	Average	Maximum
New-Order	0.00	12.04	120.30
Payment	0.00	12.03	120.31
Delivery	0.00	5.03	50.60
Stock-Level	0.00	5.01	44.85
Order Status	0.00	10.10	100.71

5.4. Response Time Frequency Distribution Curves

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

Figure 5.1: New Order Response Time Distribution

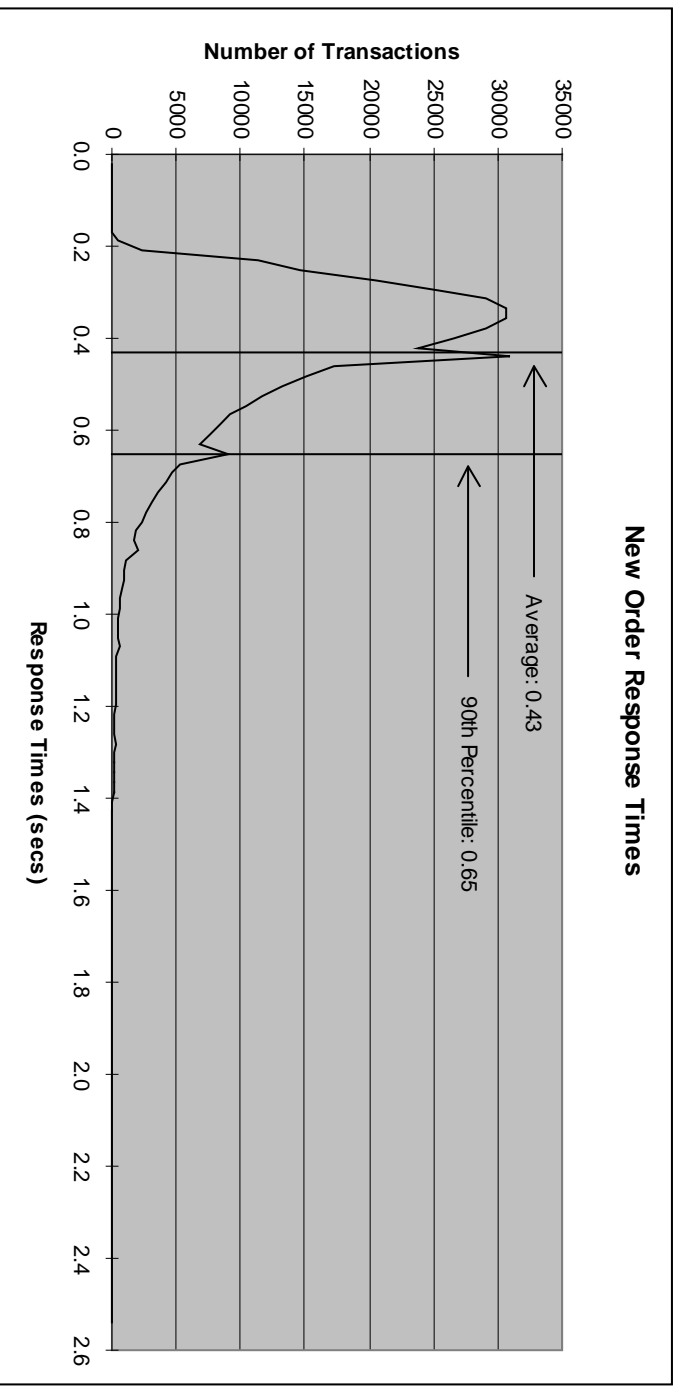


Figure 5.2: Payment Response Time Distribution

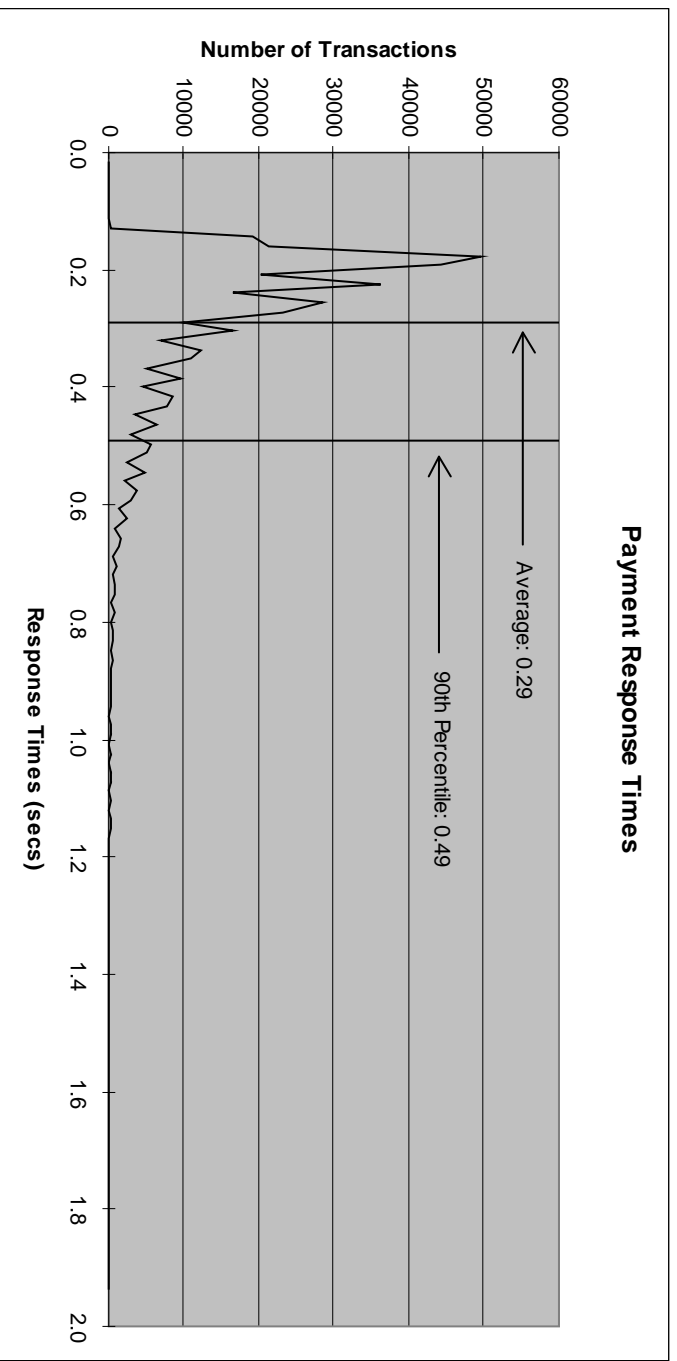


Figure 5.3: Order Status Response Time Distribution

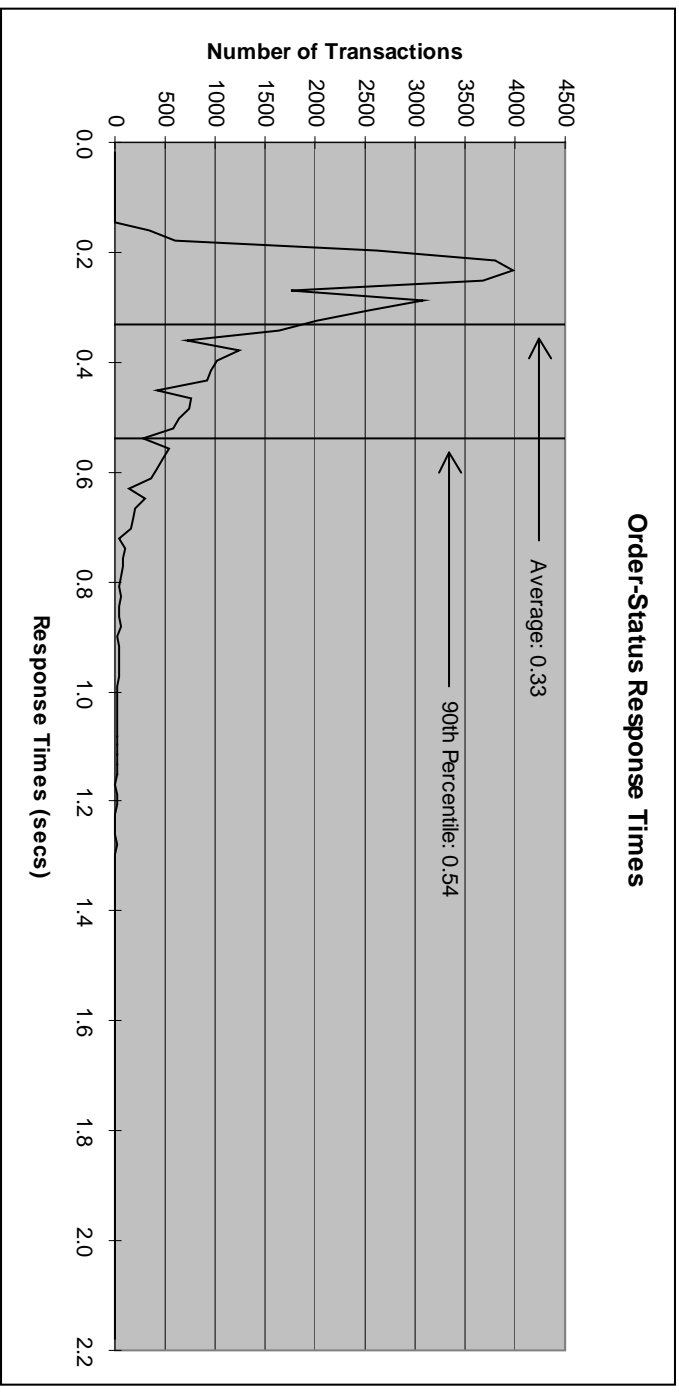


Figure 5.4: Delivery Response Time Distribution

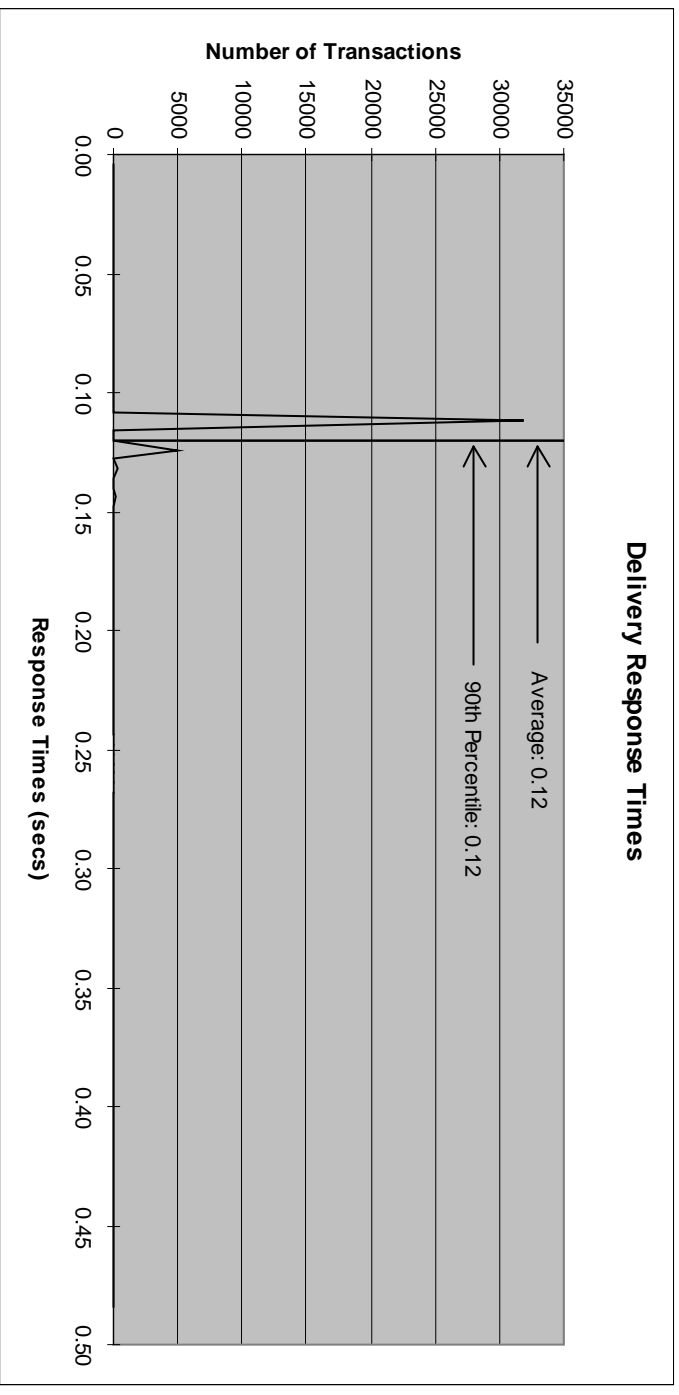
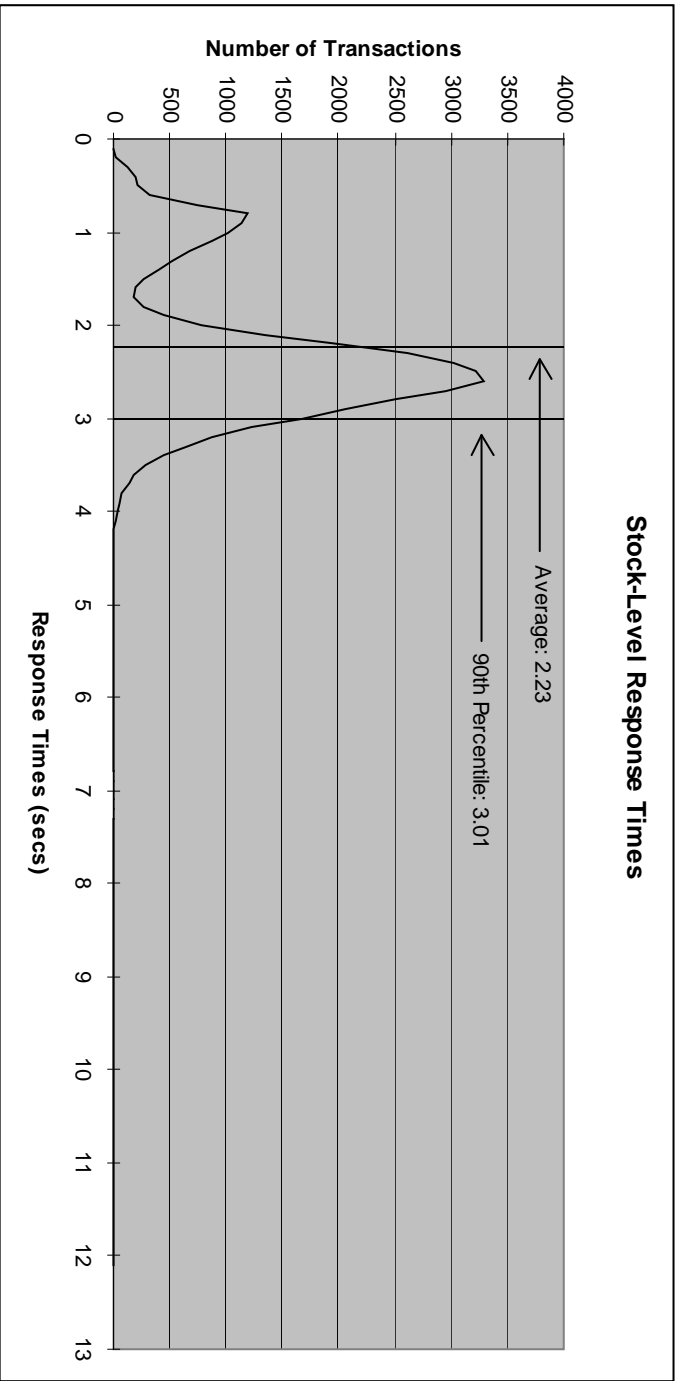


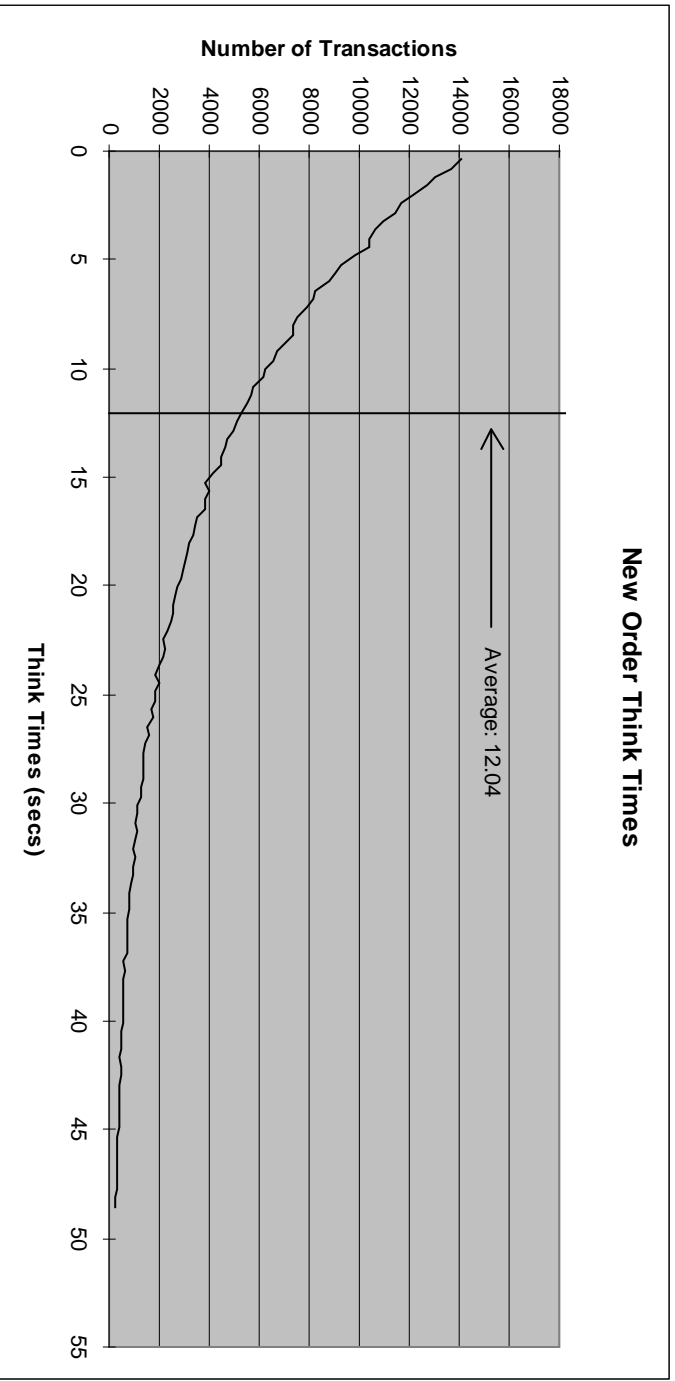
Figure 5.5: Stock Level Response Time Distribution



5.5. New Order Think Time Frequency Distribution Curve

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

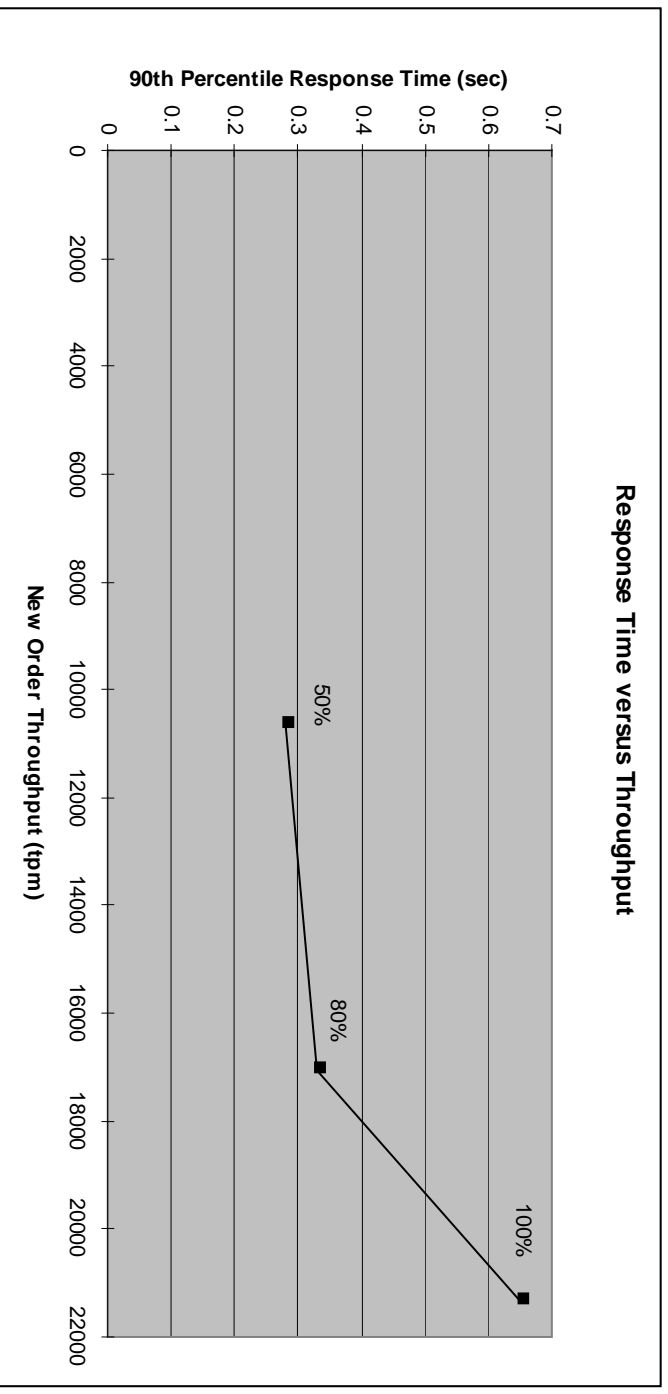
Figure 5.6: New Order Think Time Distribution



5.6. Response Time versus Throughput Performance Curve

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

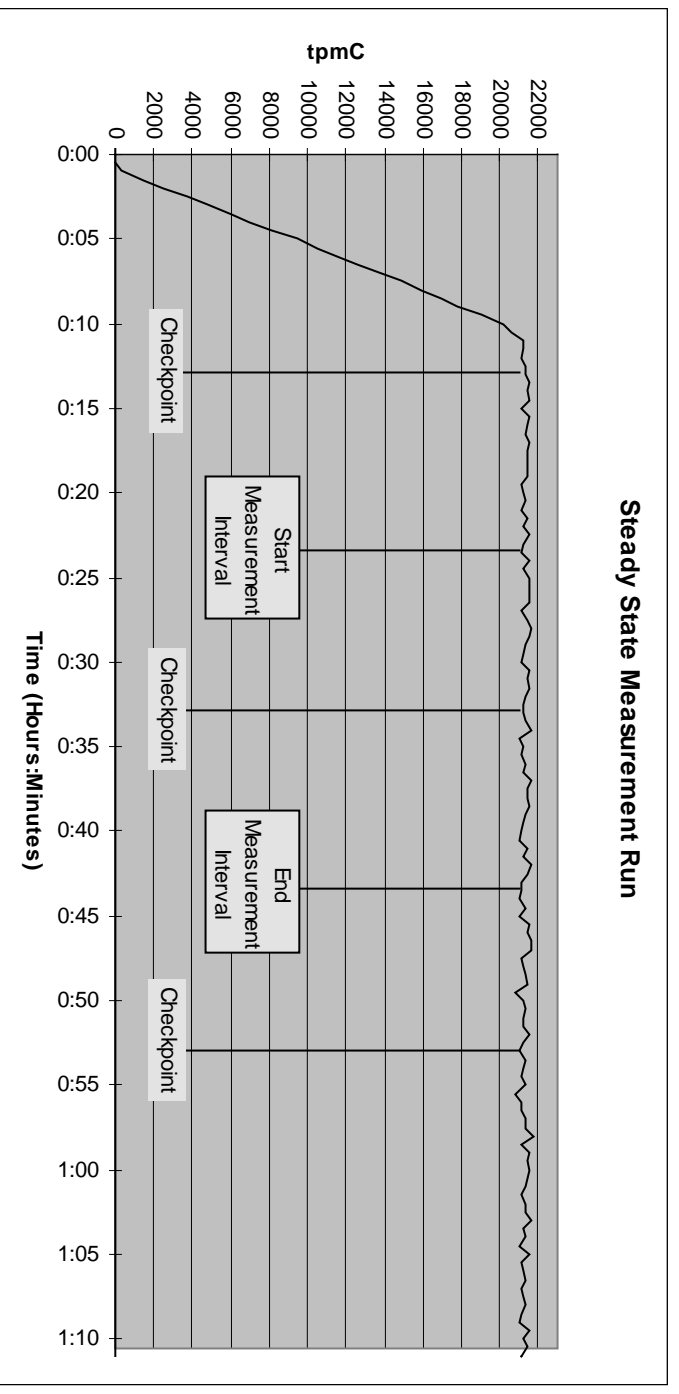
Figure 5.7: Response Time versus Throughput



5.7. New-Order Throughput vs. Time

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

Figure 5.8: Throughput (tpmC) versus Time



5.8. Determination of ‘Steady State’

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

The transaction throughput rate (tpmC) and response time were relatively constant after the initial ‘ramp up’ period. The throughput and response time behavior were determined by examining data reported for each 30-second interval over the duration of the benchmark. Ramp-up, steady state, and ramp-down regions are discernible in the graph presented in Figure 5.8.

5.9. 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 selects a transaction type from the menu and prepares to request the appropriate blank form. A timestamp is taken before the form request is sent and after the response is returned. The difference between the two is saved off as the menu response time. The RTE then generates input data for the transaction to create a completed form and waits the appropriate key time. A timestamp is taken before the completed form is sent and after the response is returned. The difference between these two is saved off as the transaction response time. Both response times are padded with a 0.1 second delay per spec to account for the web browser delay. The appropriate transaction data and response times are logged and the RTE waits the required think time interval before repeating the process. Each RTE driver maintains its own log file. Log file contents are consolidated for the reports.

The RTE emulates web browsers (not terminals) in this client-server implementation. The RTE sends and receives HTML formatted data using HTTP through Ethernet LANs to a client application running on the client machine. The client application processes the request, sends the transaction to a Tuxedo TPC-C application server queue, waits for the transaction response (except for delivery), and returns an appropriately formatted HTML form back to the (emulated) web browser (RTE). The Tuxedo TPC-C application server retrieves a message from its queue, invokes request processing via a stored procedure on the database server using Microsoft SQL Server DDLIB and RPC through sockets over another Ethernet LAN, accepts the response, and returns a result to the client application (via Tuxedo). For delivery transactions, the client application does not wait for the Tuxedo TPC-C delivery server to respond. Each delivery server logs its results to its own file. The delivery report files are consolidated for reports.

To perform checkpoints at specific intervals, SQL Server’s checkpoint interval was set to the maximum allowable value and a utility was written to schedule checkpoints at parameter-specified intervals and record the start and end time of each checkpoint. The checkpoint script was started manually on one of the client machines after the RTE had all users logged in and sending transactions and a steady state had been achieved. Using this information, the positioning of the checkpoint within the measurement interval was verified to be clear of the guard zones.

At each checkpoint, SQL Server wrote to disk all database pages in memory that had been updated but not yet physically written to the disk. Upon completion of the checkpoint, SQL Server also wrote records to the transaction log indicating that a checkpoint had completed.

5.10. Reproducibility

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

In a repeat test, carried out in the same manner as the primary test, a throughput of 21,301.90 tpmC was achieved on the same database during a 20-minute, steady state run. All required transaction statistics were met. See the Auditor's attestation letter for details.

5.11. Measurement Interval Duration

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

The measurement interval was 20 minutes.

5.12. 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 weighed random distribution which could not be adjusted during the run.

5.13. 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 entered 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 5.4 shows this information.

Table 5.4: Transaction Statistics

Transaction Type	Statistics	Value
New Order	Rolledback transactions	1.00%
	Home warehouse	99.01%
	Remote warehouse	0.99%
	Average Items per Order	10.00
Payment	Home warehouse	85.00%
	Remote warehouse	15.00%
	Non-primary key access	59.96%
Order Status	Non-primary key access	59.77%
Delivery	Skipped transactions (Interactive)	0
	Skipped transaction counts (Deferred)	0
	Skipped District counts (Deferred)	0
Transaction Mix	New Order	44.88%
	Payment	43.06%
	Delivery	4.04%
	Stock-Level Order-Status	4.01% 4.02%

5.14. Checkpoint Statistics

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.

There is one checkpoint in the measurement interval. The checkpoint starts 571 seconds into the measurement interval. The checkpoint interval is 20 minutes (from the start of one to the start of the next) and a checkpoint lasts approximately 5.2 minutes. In conformance with Clause 5.2.2 there is no checkpoint within a span of 5 minutes before or after the beginning or end of the measurement interval.

6. Clause 6: SUT, Driver & Communications Definition

6.1. Remote Terminal Emulator (RTE) Description

The RTE input parameters, code fragments, functions, etc. used to generate each transaction input field must be disclosed.

The RTE used is proprietary to Unisys. Appendix D contains the profile used as input to this RTE.

6.2. Emulated Components

It must be demonstrated that the functionality and performance of the components being emulated in the Driver System are equivalent to that of the priced system.

There were no emulated components in the benchmark configuration other than the emulated web browsers on the users' PCs.

6.3. Functional Diagrams

A complete functional diagram of both benchmark 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.

Section 0.7 describes and shows functional diagrams of the benchmarked and priced systems.

6.4. Network Configuration

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

Figures 0.1 and 0.2 in Section 0.7 also diagram the network configurations of the benchmark and configured systems and represent the RTEs connected via LAN replacing the user PCs that are directly connected via LAN.

6.5. Network Bandwidth

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

Ethernet local area networks (LAN) are used in the priced and tested configurations. The database server (SUT) contains a single 10/100 megabit per second LAN adapter connecting it to the client systems. This LAN segment is run at 100 megabits per second in both the priced and tested configuration.

Each client contains five 10/100 megabit per second LAN adapters and one quad LAN adapter that supports four 10/100 megabit per second LAN segments. One 10/100 megabit per second LAN adapter connects to a LAN segment that communicates with the SUT at 100 megabits per second in both the priced and tested configuration. Six of the

remaining eight LAN segments were run at 10 megabits per second in both the priced and tested configurations. Two LAN segments were not used in both the priced and tested configuration.

In the priced configuration, each client is connected to workstations (PCs running web browsers) spread over six 10 megabit per second LAN segments.

In the tested configuration, each client is connected to RTE driver systems emulating web browsers spread over six 10 megabit per second LAN segments.

6.6. Operator Intervention

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

No operator intervention was required to sustain eight hours of operation at the reported throughput.

7.

Clause 7: Pricing

7.1. 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 date. If package-pricing is used, vendor part number of the package and a description uniquely identifying each of the components of the package must be disclosed. Pricing source(s) and effective date(s) 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.

System pricing should include subtotals for the following components: Server Hardware, Server Software, Client Hardware, Client Software, and Network Components used for terminal connection (see Clause 7.2.2.3). Clause 6.1 describes the Server and Client components.

System pricing must include line item indication where non-sponsoring companies' brands are used. System pricing must also include line item indication of third party pricing.

A detailed list of hardware and software components along with their part numbers and prices are given in the Executive Summary near the beginning of this document.

7.1.1. System Pricing

Each priced configuration consists of an integrated system package, additional options, and components. Prices for all products are US list prices. A three year warranty is standard with this class of Unisys server products.

7.1.2. Maintenance Pricing

The five year support pricing for Unisys Corporation Open Business Server products is based on a 36-month warranty on hardware, upgraded to service level Performance-Gold, plus an additional 24 months of support at service level Performance-Gold. Microsoft and BEA support pricing is based on 60 months of monthly support costs.

Unisys Corporation Standard Performance-Gold Support: four hour maximum response, onsite support for hardware provides service from 8:00 A.M. to 5:00 P.M., Monday through Friday. Service requests made as late as 5:00 P.M. will receive a response the same day.

Netlux and Data Comm Warehouse provide return-to-factory replacement within seven days. Server disks are covered by Western Micro's seven day return-to-factory warranty. Appropriate spares are included in the priced configuration. Mylex's 36-month, seven day return-to-factory warranty is extended to 60 months, and appropriate spares and upgrade price are included in the priced configuration.

7.1.3. Discounts

Unisys provides a standard pre-pay discount for maintenance service of the client, server and storage components of the priced configuration.

Western Micro provides a standard dollar-volume discount to the client, server and storage components of the priced configuration.

7.2. Availability

The committed delivery date for general availability (availability date) of products used in the price calculation must be reported. When the priced system includes products with different availability dates, the reported availability date for the priced system must be the date at which all components are committed to be available.

The hardware, software and support/maintenance products priced in this benchmark are detailed on page vi.

All components will be available by January 5th, 1999.

7.3. Measured tpmC, Price/Performance, and Availability Date

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.

Unisys Corporation Aquanta QR2 Server with Microsoft Windows NT Server Enterprise Edition 4.0 and SQL Server Enterprise Edition 7.0, achieved 21,354.55 tpmC at \$23.73 per tpmC. All components will be available by January 5, 1999.

7.4. Country-Specific Pricing

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

None.

7.5. 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:

- One (1) Microsoft Windows NT Server Enterprise Edition 4.0 license
- One (1) Microsoft SQL 7.0 Server Enterprise Edition license
- Three (3) Microsoft Windows NT Server 4.0 Licenses
- One (1) Microsoft Visual C++ Professional 5.0
- Three (3) BEA Tuxedo 6.3 CFS for NT licenses

Microsoft SQL Server & Internet Information Server and BEA Tuxedo were priced for an unlimited number of users.

8.

Clause 8 : Full Disclosure Availability

8.1. Availability

The Full Disclosure Report must be readily available to the public at a reasonable charge, similar to charges for similar documents by that test sponsor.

Copies of this Full Disclosure Report may be obtained by contacting:

TPC Benchmark Administrator
Systems Analysis, Modeling & Measurement Group
Unisys Corporation, M/S 262
25725 Jeronimo Road
Mission Viejo, CA 92691
USA

9.

Clause 9 : Audit

9.1. 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 on the Unisys Aquanta QR/2 Server was audited by Richard Gimarc, a TPC certified auditor of:

Performance Metrics Inc.,
2229 Benita Drive, Suite 101,
Rancho Cordova, CA 95670.

(916) 635-2822 Fax: (916) 858-0109
e-mail: Richard@PerfMetrics.com

The attestation letter is shown on the next page.

PERFORMANCE METRICS INC.
TPC Certified Auditors

December 22, 1998

Jerrold Buggert
 Director of Modeling and Measurement
 Unisys Corporation
 25725 Jeronimo Road
 Mission Viejo, CA 92691

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

Platform: Unisys Aquanta QR/2 Server
 Database Manager: Microsoft SQL Server Enterprise Edition 7.0
 Operating System: Microsoft Windows NT Server Enterprise Edition 4.0 (SP4)
 Transaction Manager: BEA TUXEDO CFS 6.3 for NT

CPU's	Memory	Disks	New-Order Response Time @ 90%	tpmC
Server: Unisys Aquanta QR/2 Server				
4 Pentium II Xeon @ 450 MHz	Main: 4 GB L2 Cache: 2 MB	73 @ 4.23 GB 142 @ 8.48 GB 14 @ 17.01 GB	0.65 sec.	21,354.55
3 Clients: Unisys NetServer LC3				
2 Pentium II @ 450 MHz	Main: 512 MB L2 Cache: 512 KB	1 @ 3.97 GB	n/a	n/a

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 was properly sized and populated.
- The database was properly scaled with 1,728 warehouses. Only 1,710 warehouses were used during measurement.
- The ACID properties were met.

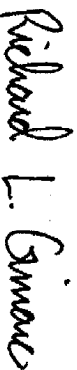
2229 Benita Drive, Suite 101, Rancho Cordova, CA 95670
 (916) 635-2822 Fax: (916) 858-0109 e-mail: Richard@PerfMetrics.com

PERFORMANCE METRICS INC.
TPC Certified Auditors

- The durability data loss and log loss tests were performed on a 10-warehouse database.
- Input data was generated according to the specified percentages.
- Eight hours of mirrored log space was configured on the measured system.
- Eight hours of dynamic table growth space was configured on the measured system.
- The following server disks contained backup and other data and were not active during measurement: two 4.23 GB disks, two 8.48 GB disks, and fourteen 17.01 GB disks. These 18 disks were not included in the priced configuration.
- The 180-day space calculation was verified. The measured configuration has sufficient storage to satisfy this requirement.
- Measurement cycle times included a 0.1 second menu and a 0.1 second response time delay for an emulated Web browser.
- The steady state portion of the test was 20 minutes.
- One checkpoint was taken during the steady state portion of the test.
- Checkpoints were verified to be clear of the guard zones.
- There were 17,100 user contexts present on the system.
- Each emulated user started with a different random number seed.
- The NURand constants used for database load and at run time were verified.
- System pricing was checked for major components and maintenance.

Additional Audit Notes: (none)

Regards,



Richard L. Gimarc
Auditor

2229 Benita Drive, Suite 101, Rancho Cordova, CA 95670
(916) 635-2822 Fax: (916) 858-0109 e-mail: Richard@PerfMetrics.com

Page 2

Appendix A - Client/Server Source

CLIENT MAKEFILE

```
# Microsoft Developer Studio Generated NMAKE File, Format Version 4.20
# ** DO NOT EDIT **

# TARGETTYPE "Win32 (x86) Dynamic-Link Library" 0x0102

!IF "$(CFG)" == ""
CFG=tpcc - Win32 Debug
!MESSAGE No configuration specified. Defaulting to tpcc - Win32 Debug.
!ENDIF

!IF "$(CFG)" != "tpcc - Win32 Release" && "$(CFG)" != "tpcc - Win32 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 /f "tpcc.mak" CFG="tpcc - Win32 Debug"
!MESSAGE
!MESSAGE Possible choices for configuration are:
!MESSAGE
!MESSAGE "tpcc - Win32 Release" (based on "Win32 (x86) Dynamic-Link
Library")
!MESSAGE "tpcc - Win32 Debug" (based on "Win32 (x86) Dynamic-Link
Library")
!MESSAGE
!ERROR An invalid configuration is specified.
!ENDIF

!IF "$(OS)" == "Windows_NT"
NULL=
!ELSE
NULL=nul
!ENDIF
#####
#####
# Begin Project
# PROP Target_Last_Scanned "tpcc - Win32 Release"
CPP=cl.exe
RSC=rc.exe
MTL=mktyplib.exe

!IF "$(CFG)" == "tpcc - Win32 Release"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 0
# PROP BASE Output_Dir "Release"
# PROP BASE Intermediate_Dir "Release"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 0
# PROP Output_Dir "Release"
```

```
# PROP Intermediate_Dir "Release"
# PROP Target_Dir ""
OUTDIR=.\Release
INTDIR=.\Release

ALL : "$(OUTDIR)\tpcc.dll"

CLEAN :
    -@erase "$(INTDIR)\diagio.obj"
    -@erase "$(INTDIR)\term.obj"
    -@erase "$(INTDIR)\timesupp.obj"
    -@erase "$(INTDIR)\tmon.obj"
    -@erase "$(INTDIR)\TPCC.OBJ"
    -@erase "$(INTDIR)\tpcchandler.obj"
    -@erase "$(OUTDIR)\tpcc.dll"
    -@erase "$(OUTDIR)\tpcc.exp"
    -@erase "$(OUTDIR)\tpcc.lib"

"$ (OUTDIR)" :
    if not exist "$(OUTDIR)/$(NULL)" mkdir "$(OUTDIR)"

# ADD BASE CPP /nologo /MT /W3 /GX /O2 /D "WIN32" /D "NDEBUG" /D
" _WINDOWS" /YX /c
# ADD CPP /nologo /MT /W3 /GX /O2 /D "WIN32" /D "NDEBUG" /D "_WINDOWS" /YX
/c
CPP_PROJ=/nologo /MT /W3 /GX /O2 /D "WIN32" /D "NDEBUG" /D "_WINDOWS" \
/Fp"$(INTDIR)/tpcc.pch" /YX /Fo"$(INTDIR)/" /c
CPP_OBJ=.\Release/
CPP_SBRS=.\.
# ADD BASE MTL /nologo /D "NDEBUG" /win32
# ADD MTL /nologo /D "NDEBUG" /win32
MTL_PROJ=/nologo /D "NDEBUG" /win32
# ADD BASE RSC /l 0x409 /d "NDEBUG"
# ADD RSC /l 0x409 /d "NDEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
BSC32_FLAGS=/nologo /o"$(OUTDIR)/tpcc.bsc"
BSC32_SBRS= \

LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib
odbc32.lib odbccp32.lib /nologo /subsystem:windows /dll /machine:I386
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib libtux.lib libbuft.lib libtux2.lib libfml.lib libfml32.lib
libgp.lib /nologo /subsystem:windows /dll /machine:I386
# SUBTRACT LINK32 /verbose /nodefaultlib
LINK32_FLAGS=kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib \
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib \
odbccp32.lib libtux.lib libbuft.lib libtux2.lib libfml.lib libfml32.lib \
libgp.lib /nologo /subsystem:windows /dll /incremental:no \
/pdb:"$(OUTDIR)/tpcc.pdb" /machine:I386 /def:". \tpcc.def" \
/out:"$(OUTDIR)/tpcc.dll" /implib:"$(OUTDIR)/tpcc.lib"
DEF_FILE= \
```

```

        ".\tpcc.def"
LINK32_OBJS= \
    "$ (INTDIR)\diagio.obj" \
    "$ (INTDIR)\term.obj" \
    "$ (INTDIR)\timesupp.obj" \
    "$ (INTDIR)\tmon.obj" \
    "$ (INTDIR)\TPCC.OBJ" \
    "$ (INTDIR)\tpcchandler.obj"

"$ (OUTDIR)\tpcc.dll" : "$ (OUTDIR)" $(DEF_FILE) $(LINK32_OBJS)
    $(LINK32) @<<
    $(LINK32_FLAGS) $(LINK32_OBJS)
<<

!ELSEIF "$(CFG)" == "tpcc - Win32 Debug"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 1
# PROP BASE Output_Dir "Debug"
# PROP BASE Intermediate_Dir "Debug"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 1
# PROP Output_Dir "Debug"
# PROP Intermediate_Dir "Debug"
# PROP Target_Dir ""
OUTDIR=.\Debug
INTDIR=.\Debug

ALL : "$ (OUTDIR)\tpcc.dll"

CLEAN :
    -@erase "$(INTDIR)\diagio.obj"
    -@erase "$(INTDIR)\term.obj"
    -@erase "$(INTDIR)\timesupp.obj"
    -@erase "$(INTDIR)\tmon.obj"
    -@erase "$(INTDIR)\TPCC.OBJ"
    -@erase "$(INTDIR)\tpcchandler.obj"
    -@erase "$(INTDIR)\vc40.idb"
    -@erase "$(INTDIR)\vc40.pdb"
    -@erase "$(OUTDIR)\tpcc.dll"
    -@erase "$(OUTDIR)\tpcc.exp"
    -@erase "$(OUTDIR)\tpcc.ilk"
    -@erase "$(OUTDIR)\tpcc.lib"
    -@erase "$(OUTDIR)\tpcc.pdb"

"$ (OUTDIR)" :
    if not exist "$ (OUTDIR)/$(NULL)" mkdir "$ (OUTDIR)"

# ADD BASE CPP /nologo /MTd /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D
" _WINDOWS" /YX /c
# ADD CPP /nologo /MT /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D
" _WINDOWS" /YX /c
CPP_PROJ=/nologo /MT /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D
" _WINDOWS" \
    /Fp"$ (INTDIR)\tpcc.pch" /YX /Fo"$ (INTDIR)/" /Fd"$ (INTDIR)/" /c
CPP_OBJS=.\Debug\
CPP_SBRS=.\.
# ADD BASE MTL /nologo /D "_DEBUG" /win32
# ADD MTL /nologo /D "_DEBUG" /win32
MTL_PROJ=/nologo /D "_DEBUG" /win32

```

```

# ADD BASE RSC /l 0x409 /d "_DEBUG"
# ADD RSC /l 0x409 /d "_DEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
BSC32_FLAGS=/nologo /o"$ (OUTDIR)\tpcc.bsc"
BSC32_SBRS= \

LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib
odbc32.lib odbccp32.lib /nologo /subsystem:windows /dll /debug
/machine:I386
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib libtux.lib libbuft.lib libtux2.lib libfml.lib libfml32.lib
libgp.lib /nologo /subsystem:windows /dll /debug /machine:I386
# SUBTRACT LINK32 /verbose /nodefaultlib
LINK32_FLAGS=kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib\
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib\
odbccp32.lib libtux.lib libbuft.lib libtux2.lib libfml.lib libfml32.lib\
libgp.lib /nologo /subsystem:windows /dll /incremental:yes\
/pdb:"$ (OUTDIR)\tpcc.pdb" /debug /machine:I386 /def:".\tpcc.def"\
/out:"$ (OUTDIR)\tpcc.dll" /implib:"$ (OUTDIR)\tpcc.lib"
DEF_FILE= \
    ".\tpcc.def"
LINK32_OBJS= \
    "$ (INTDIR)\diagio.obj" \
    "$ (INTDIR)\term.obj" \
    "$ (INTDIR)\timesupp.obj" \
    "$ (INTDIR)\tmon.obj" \
    "$ (INTDIR)\TPCC.OBJ" \
    "$ (INTDIR)\tpcchandler.obj"

"$ (OUTDIR)\tpcc.dll" : "$ (OUTDIR)" $(DEF_FILE) $(LINK32_OBJS)
    $(LINK32) @<<
    $(LINK32_FLAGS) $(LINK32_OBJS)
<<

!ENDIF

.c{$ (CPP_OBJS)}.obj:
    $(CPP) $(CPP_PROJ) $<

.cpp{$ (CPP_OBJS)}.obj:
    $(CPP) $(CPP_PROJ) $<

.cxx{$ (CPP_OBJS)}.obj:
    $(CPP) $(CPP_PROJ) $<

.c{$ (CPP_SBRS)}.sbr:
    $(CPP) $(CPP_PROJ) $<

.cpp{$ (CPP_SBRS)}.sbr:
    $(CPP) $(CPP_PROJ) $<

.cxx{$ (CPP_SBRS)}.sbr:
    $(CPP) $(CPP_PROJ) $<

#####
#####

```

```

# Begin Target

# Name "tpcc - Win32 Release"
# Name "tpcc - Win32 Debug"

!IF "$(CFG)" == "tpcc - Win32 Release"

!ELSEIF "$(CFG)" == "tpcc - Win32 Debug"

!ENDIF

#####
#####
# Begin Source File

SOURCE=.\term.c
DEP_CPP_TERM=\
    ".\diagio.h"\
    ".\term.h"\
    ".\timesupp.h"\

"$ (INTDIR)\term.obj" : $(SOURCE) $(DEP_CPP_TERM_) "$ (INTDIR) "

# End Source File
#####
#####
# Begin Source File

SOURCE=.\timesupp.c
DEP_CPP_TIMES=\
    ".\timesupp.h"\

"$ (INTDIR)\timesupp.obj" : $(SOURCE) $(DEP_CPP_TIMES) "$ (INTDIR) "

# End Source File
#####
#####
# Begin Source File

SOURCE=.\TPCC.C
DEP_CPP_TPCC=\
    ".\diagio.h"\
    ".\term.h"\
    ".\tmon.h"\
    ".\tpcc.h"\
    ".\tpcchandler.h"\

"$ (INTDIR)\TPCC.OBJ" : $(SOURCE) $(DEP_CPP_TPCC_) "$ (INTDIR) "

# End Source File
#####
#####
# Begin Source File

SOURCE=.\tpcchandler.c

```

```

DEP_CPP_TPCCCH=\
    ".\diagio.h"\
    ".\term.h"\
    ".\tmon.h"\
    ".\tpcc.h"\
    ".\tpcchandler.h"\

"$ (INTDIR)\tpcchandler.obj" : $(SOURCE) $(DEP_CPP_TPCCCH) "$ (INTDIR) "

# End Source File
#####
#####
# Begin Source File

SOURCE=.\tpcc.def

!IF "$(CFG)" == "tpcc - Win32 Release"

!ELSEIF "$(CFG)" == "tpcc - Win32 Debug"

!ENDIF

# End Source File
#####
#####
# Begin Source File

SOURCE=.\tmon.c
DEP_CPP_TMON=\
    ".\tmon.h"\
    {$ (INCLUDE) }\atmi.h"\
    {$ (INCLUDE) }\sys\types.h"\
    {$ (INCLUDE) }\tmenv.h"\

"$ (INTDIR)\tmon.obj" : $(SOURCE) $(DEP_CPP_TMON_) "$ (INTDIR) "

# End Source File
#####
#####
# Begin Source File

SOURCE=.\diagio.c
DEP_CPP_DIAGI=\
    ".\diagio.h"\

"$ (INTDIR)\diagio.obj" : $(SOURCE) $(DEP_CPP_DIAGI) "$ (INTDIR) "

# End Source File
# End Target
# End Project
#####
#####

```

tpcc.def

```
EXPORTS
  GetExtensionVersion
  HttpExtensionProc
```

tpcc.h

```
// tpcc.h

#include <time.h>

// TPCCHandler return codes
#define TPCCSEND 1
#define TPCCSENDEND 2
#define TPCCENDNOW 3

// TPCC Service return codes
#define SVC_BADITEMID 1
#define SVC_NOERROR 0
#define SVCERR_DEADLOCK -1
#define SVCERR_NOCUSTOMER -2
#define SVCERR_NOORDERS -3
#define SVCERR_DBLIB -4

// Min/Max transaction data definitions
#define MIN_DID 1
#define MAX_DID 10
#define MIN_OL 5
#define MAX_OL 15
#define MIN_QUANTITY 1
#define MAX_QUANTITY 10
#define MIN_ITEM_ID 1
#define MAX_ITEM_ID 10000
#define MIN_CUST_ID 1
#define MAX_CUST_ID 3000
#define MIN_CARRIER 1
#define MAX_CARRIER 10
#define MIN_THRESHOLD 10
#define MAX_THRESHOLD 20

// pTPCC->iStatusId codes
#define INVALID_IID 1
#define STATUS_OK 0
#define ERR_CMD_UNKNOWN -10
#define ERRTXT_CMD_UNKNOWN "Unrecognized Command"
#define ERR_ALREADY_LOGGEDIN -11
#define ERRTXT_ALREADY_LOGGEDIN "Already Logged In"
#define ERR_TERMID -12
#define ERRTXT_TERMID "TermId or SyncId in Error"
#define ERR_FORM_UNKNOWN -13
#define ERRTXT_FORM_UNKNOWN "Unrecognized FormId"
#define ERR_WID_INVALID -14
#define ERR_DID_INVALID -15
#define ERR_MISSING_KEY -16
#define ERR_NOT_NUMERIC -17
#define ERR_THRESHOLD_RANGE -18
#define ERR_EMBEDDED_EMPTY_OL -19
#define ERR_QUANTITY_INVALID -20
#define ERR_OL_INVALID -21
```

```
#define ERR_OL_COUNT -22
#define ERR_TM_INTERFACE -23
#define ERR_SERVICE_RSLT -24
#define ERR_INPUT_TOOLONG -25
#define ERR_IDANDNAME_EMPTY -26
#define ERR_IDANDNAME_ENTERED -27
#define ERR_AMOUNT_BADFORM -28
#define ERR_AMOUNT_INVALID -29
#define ERR_CARRIER_INVALID -30
#define ERR_TERM_ALLOC -31

#define STATUS_LEN 200
#define NAME_LEN 16
#define ADDR_LEN 20
#define STATE_LEN 2
#define ZIP_LEN 9

#define MAX_MSG_SZ 5000
#define CTEXT "Content-length: "
#define HTTPHDR "Connection: keep-alive\r\nContent-type: text/html\r\n" \
               "Content-length: \r\n\r\n"

typedef struct
{
  int year;
  int quarter;
  int month;
  int dayofyear;
  int day;
  int week;
  int weekday;
  int hour;
  int minute;
  int second;
  int millisecond;
} DBDATEREC;

typedef struct
{
  short ol_supply_w_id;
  long ol_i_id;
  char ol_i_name[25];
  short ol_quantity;
  char ol_brand_generic[2];
  double ol_i_price;
  double ol_amount;
  short ol_stock;
} OL_NEW_ORDER_DATA;

typedef struct
{
  short w_id;
  short d_id;
  long c_id;
  short o_ol_cnt;
  char c_last[NAME_LEN + 1];
  char c_credit[3];
  double c_discount;
  double w_tax;
  double d_tax;
  long o_id;
```

```

short o_commit_flag;
DBDATEREC o_entry_d;
short o_all_local;
double total_amount;
char execution_status[STATUS_LEN];
OL_NEW_ORDER_DATA Ol[MAX_OL];
} 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;
DBDATEREC h_date;
char w_street_1[ADDR_LEN + 1];
char w_street_2[ADDR_LEN + 1];
char w_city[ADDR_LEN + 1];
char w_state[STATE_LEN + 1];
char w_zip[ZIP_LEN + 1];
char d_street_1[ADDR_LEN + 1];
char d_street_2[ADDR_LEN + 1];
char d_city[ADDR_LEN + 1];
char d_state[STATE_LEN + 1];
char d_zip[ZIP_LEN + 1];
char c_first[NAME_LEN + 1];
char c_middle[3];
char c_last[NAME_LEN + 1];
char c_street_1[ADDR_LEN + 1];
char c_street_2[ADDR_LEN + 1];
char c_city[ADDR_LEN + 1];
char c_state[STATE_LEN + 1];
char c_zip[ZIP_LEN + 1];
char c_phone[16];
DBDATEREC c_since;
char c_credit[3];
double c_credit_lim;
double c_discount;
double c_balance;
char c_data[200+1];
char execution_status[STATUS_LEN];
} PAYMENT_DATA;

```

```
typedef struct
```

```

{
long ol_i_id;
short ol_supply_w_id;
short ol_quantity;
double ol_amount;
DBDATEREC ol_delivery_d;
} OL_ORDER_STATUS_DATA;

```

```
typedef struct
```

```

{
short w_id;
short d_id;
long c_id;
char c_first[NAME_LEN + 1];
char c_middle[3];

```

```

char c_last[NAME_LEN + 1];
double c_balance;
long o_id;
DBDATEREC o_entry_d;
short o_carrier_id;
OL_ORDER_STATUS_DATA OlOrderStatusData[MAX_OL];
short o_ol_cnt;
char execution_status[STATUS_LEN];
} ORDER_STATUS_DATA;

```

```
typedef struct
```

```

{
short w_id;
short o_carrier_id;
long o_id[10];
int iComplete;
SYSTEMTIME QTime; // time delivery was queued
SYSTEMTIME EndTime; // time delivery completed
char execution_status[STATUS_LEN];
} DELIVERY_DATA;

```

```
typedef struct
```

```

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

```

```
typedef struct
```

```

{
LPVOID ConnID; // Active Connection Id
SHORT sWId; // TPCC Warehouse Id
SHORT sDId; // TPCC District Id
INT iSyncId; // TPCC Sync Id
INT iTermId; // TPCC Term Id
UINT uFormId; // TPCC Form Id
INT iStatusId; // TPCC Status Id
CHAR ErrTxt[500]; // Error text
CHAR szWork[200]; // Thread work area
CHAR szHeader[100]; // HTTP work area
CHAR * RecvMsg; // HTML message from ECB
CHAR SendMsg[MAX_MSG_SZ]; // HTML work area
TMON_STATE tsTMon; // TMon Interface
} TPCC_STATE;

```

tpcc.c

```

// tpcc.c
//
// Copyright Unisys, 1997
//
#include <windows.h>
#include <stdio.h>
#include <malloc.h>
#include <stdlib.h>
#include <string.h>
#include <winreg.h>
#include <httplib.h>

```

```

#include "tmon.h"
#include "diagio.h"
#include "term.h"
#include "tpcchandler.h"

#define EXTN_VERSION MAKELONG(HSE_VERSION_MINOR,HSE_VERSION_MAJOR)
#define TLS_NULL 0xFFFFFFFF
DWORD dwTlsInx;
CHAR * pTitle = "IIS TPCC DLL";
CRITICAL_SECTION csDllMain;

// Diagnostic logging settings
BOOL bEventLog = TRUE;
BOOL bConsole = FALSE;
UINT uDiagLevel = DIAG_INFO;

// TMon Interface Settings
INT iTMMaxMsg = 0;

// Term Interface Settings
INT iMaxTerms = 3000;

static CHAR * szTPCCError =
    HTTPHdr "<HTML>"
    "<HEAD><TITLE>Welcome To TPC-C</TITLE></HEAD><BODY>"
    "<B>TPCC Extension Error (TPCC Array Not Allocated)</B><BR>"
    "</BODY></HTML>";

static CHAR * szTMinInitError =
    HTTPHdr "<HTML>"
    "<HEAD><TITLE>Welcome To TPC-C</TITLE></HEAD><BODY>"
    "<B>TPCC Extension Error (TMinInit Failed)</B><BR>"
    "</BODY></HTML>";
INT ihHdrLen = 0;
INT icTextLen = 0;

BOOL ThreadAttach(TPCC_STATE * pTPCC,CHAR * pDiag);
VOID ThreadDetach(TPCC_STATE * pTPCC);
VOID SendResponse(EXTENSION_CONTROL_BLOCK * pECB,CHAR * pMsg,CHAR *
pWork);
BOOL ReadRegistry(VOID);

//=====
//
// Function name: DllMain
//
//=====
BOOL APIENTRY DllMain(HANDLE hInst, ULONG ul_reason_for_call,
LPVOID lpReserved)
{
    TPCC_STATE * pTPCC = NULL;
    CHAR szDiag[MAX_DIAG_SZ];
    UINT iTMMaxSz = 0;
    switch(ul_reason_for_call)
    {
        case DLL_PROCESS_ATTACH:
            // Process initialization

            InitializeCriticalSection(&csDllMain);
            ReadRegistry();

```

```

DiagIoInit(pTitle,bConsole,bEventLog,uDiagLevel);
sprintf(szDiag,
    "EventLog = %d, Console = %d, DiagLevel = %d\n"
    "MaxTerms = %d\n",
    bEventLog,bConsole,uDiagLevel,iMaxTerms);
DiagIoWrite(szDiag,DIAG_FORCE);
dwTlsInx = TlsAlloc();
if (dwTlsInx == TLS_NULL)
{
    sprintf(szDiag,"PAttach(%d): Tls Alloc Failed (%d)\n",
        GetCurrentThreadId(),GetLastError());
    DiagIoWrite(szDiag,DIAG_ERROR);
    return(FALSE);
};
if (TermInit(iMaxTerms))
    return(FALSE);
iTMMMaxSz = max(iTMMMaxSz,sizeof(NEW_ORDER_DATA));
iTMMMaxSz = max(iTMMMaxSz,sizeof(PAYMENT_DATA));
iTMMMaxSz = max(iTMMMaxSz,sizeof(ORDER_STATUS_DATA));
iTMMMaxSz = max(iTMMMaxSz,sizeof(DELIVERY_DATA));
iTMMMaxSz = max(iTMMMaxSz,sizeof(STOCK_LEVEL_DATA));
iTMMMaxSz += 10;
TMonInit(iTMMMaxSz);
ihHdrLen = strlen(HTTPHdr);
icTextLen = strlen(CTEXT);
break;
case DLL_THREAD_ATTACH:
    // Move ThreadAttach call to HttpExt since the DllMain call
    // for Thread Attach did not reliably come before the first
    // call to HttpExtProc.
    break;
    case DLL_THREAD_DETACH:
        ThreadDetach(pTPCC);
        break;
    case DLL_PROCESS_DETACH:
        ThreadDetach(pTPCC);
        DeleteCriticalSection(&csDllMain);
        TMonTerm();
        TermTerm();
        TlsFree(dwTlsInx);
        dwTlsInx = TLS_NULL;
        DiagIoTerm();
        break;
};
return TRUE;
}; // DllMain

//=====
//
// Function name: ThreadAttach
//
// Result:
// FALSE Thread state structure initialized
// TRUE Thread state structure initialization failure
//
//=====
BOOL ThreadAttach(TPCC_STATE * pTPCC,CHAR * pDiag)
{
    BOOL bRslt;
    UINT uLabelNoOp;
    EnterCriticalSection(&csDllMain);

```



```

try
{
    pTPCC = (TPCC_STATE *) calloc(1, sizeof(TPCC_STATE));
    if (pTPCC == NULL)
    {
        sprintf(pDiag, "ThrAtt(%ld): pTPCC Alloc Failed (%ld)\n",
            GetCurrentThreadId(), GetLastError());
        DiagIoWrite(pDiag, DIAG_ERROR);
        bRslt = TRUE;
        goto TAttachXit;
    };
    TlsSetValue(dwTlsInx, pTPCC);
    pTPCC->tsTMon.pTMDData = NULL;
    pTPCC->tsTMon.pszErrTxt = pTPCC->ErrTxt;
    if (TMinInit(&pTPCC->tsTMon))
    {
        sprintf(pDiag, "ThrAtt(%ld): TMinInit %s\n",
            GetCurrentThreadId(), pTPCC->ErrTxt);
        DiagIoWrite(pDiag, DIAG_ERROR);
        bRslt = TRUE;
        goto TAttachXit;
    };
    bRslt = FALSE;
TAttachXit:
    uLabelNoOp = 0;
}
finally
{
    LeaveCriticalSection(&csDllMain);
};

return(bRslt);
}; // ThreadAttach

//=====
//
// Function name: ThreadDetach
//
//=====
VOID ThreadDetach(TPCC_STATE * pTPCC)
{
    EnterCriticalSection(&csDllMain);
    try
    {
        pTPCC = TlsGetValue(dwTlsInx);
        if (pTPCC != NULL)
        {
            TMDone(&pTPCC->tsTMon);
            free(pTPCC);
            pTPCC = NULL;
            TlsSetValue(dwTlsInx, pTPCC);
        };
    }
    finally
    {
        LeaveCriticalSection(&csDllMain);
    };
}; // ThreadDetach

//=====

```

```

//
// Function name: GetExtensionVersion
//
//=====
BOOL WINAPI GetExtensionVersion(HSE_VERSION_INFO *pVersion)
{
    pVersion->dwExtensionVersion = EXTN_VERSION;
    strncpy(pVersion->lpszExtensionDesc, pTitle, HSE_MAX_EXT_DLL_NAME_LEN);
    return TRUE;
}; // GetExtensionVersion

//=====
//
// Function name: HttpExtensionProc
//
// Returns:
// HSE_STATUS_SUCCESS          send msg, drop connection
// HSE_STATUS_SUCCESS_AND_KEEP_CONN  send msg, keep connection
//
//=====
DWORD WINAPI HttpExtensionProc(EXTENSION_CONTROL_BLOCK * pECB)
{
    TPCC_STATE * pTPCC;
    DWORD dwRslt = HSE_STATUS_SUCCESS;
    UINT uRslt;

    pTPCC = TlsGetValue(dwTlsInx);
    if (pTPCC == NULL)
    {
        CHAR szWork[200];
        ThreadAttach(pTPCC, szWork);
        pTPCC = TlsGetValue(dwTlsInx);
        if (pTPCC == NULL)
        {
            SendResponse(pECB, szTPCCError, szWork);
            goto HttpXit;
        };
    };
    if (pTPCC->tsTMon.pTMDData == NULL)
        SendResponse(pECB, szTMinInitError, pTPCC->szHeader);
    TPCCclear(pTPCC);
    pTPCC->ConnID = pECB->ConnID;
    pTPCC->RecvMsg = pECB->lpszQueryString;
    uRslt = TPCCHandler(pTPCC);
    switch (uRslt)
    {
        case TPCCSEND:
            SendResponse(pECB, pTPCC->SendMsg, pTPCC->szHeader);
            dwRslt = HSE_STATUS_SUCCESS_AND_KEEP_CONN;
            break;
        case TPCCSENDEND:
            SendResponse(pECB, pTPCC->SendMsg, pTPCC->szHeader);
            break;
        case TPCCENDNOW:
        default:
            break;
    }; // switch (TPCCHandler result)

HttpXit:

```

```

        return(dwRslt);
}; // HttpExtensionProc

//=====
//
// Function name: SendResponse
//
//=====
VOID SendResponse(EXTENSION_CONTROL_BLOCK * pECB, CHAR * pMsg, CHAR * pWork)
{
    DWORD dwMsgBytes;
    CHAR * pCL;
    dwMsgBytes = strlen(pMsg);
    pCL=strstr(pMsg, CTEXT);
    dwMsgBytes -= iHHdrLen;
    sprintf(pWork, "%4ld", dwMsgBytes);
    pCL += iCTextLen;
    strncpy(pCL, pWork, 4);
    (*pECB->ServerSupportFunction)
        (pECB->ConnID,
         HSE_REQ_SEND_RESPONSE_HEADER,
         NULL,
         &dwMsgBytes,
         (LPDWORD)pMsg);
}; // SendResponse

//=====
//
// Function name: ReadRegistry
//
// Sets global operational parameters from registry if they exist.
// Otherwise, compiled in defaults apply.
//
// Result:
// FALSE Registry entry found
// TRUE Registry entry does not exist
//
//=====
BOOL ReadRegistry(VOID)
{
    HKEY hkTPCC;
    DWORD dwMax;
    DWORD dwRT;
    INT i;
    CHAR szValue[100];
    if (RegOpenKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\\Unisys\\TPCC", 0,
        KEY_READ, &hkTPCC) != ERROR_SUCCESS )
        return(TRUE);
    dwMax = sizeof(szValue);
    if (RegQueryValueEx(hkTPCC, "EVENTLOG", 0, &dwRT, szValue, &dwMax)
        == ERROR_SUCCESS)
    {
        if (abs(atoi(szValue) == 0))
            bEventLog = FALSE;
        else
            bEventLog = TRUE;
    };
    dwMax = sizeof(szValue);
    if (RegQueryValueEx(hkTPCC, "CONSOLE", 0, &dwRT, szValue, &dwMax)
        == ERROR_SUCCESS )

```

```

    {
        if (abs(atoi(szValue) == 0))
            bConsole = FALSE;
        else
            bConsole = TRUE;
    };
    dwMax = sizeof(szValue);
    if (RegQueryValueEx(hkTPCC, "DIAGLEVEL", 0, &dwRT, szValue, &dwMax)
        == ERROR_SUCCESS )
    {
        i = atoi(szValue);
        if (i < DIAG_FORCE)
            i = DIAG_FORCE;
        else
            if (i > DIAG_INFO)
                i = DIAG_INFO;
        uDiagLevel = i;
    };
    dwMax = sizeof(szValue);
    if (RegQueryValueEx(hkTPCC, "MAXTERMS", 0, &dwRT, szValue, &dwMax)
        == ERROR_SUCCESS )
    {
        iMaxTerms = abs(atoi(szValue));
    };
    RegCloseKey(hkTPCC);
    return(FALSE);
}; // ReadRegistry

```

tpcchandler.h

```

// tpcchandler.h

#include "tpcc.h"

BOOL TPCCclear(TPCC_STATE * pTPCC);
UINT TPCCHandler(TPCC_STATE * pTPCC);

```

tpcchandler.c

```

// tpcchandler.c
//
// Copyright Unisys, 1997
//
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include "tmon.h"
#include "diagio.h"
#include "tpcchandler.h"
#include "term.h"

// pTPCC->iFormId - TPCC forms enumeration.
#define FORM_NULL 0
#define FORM_LOGON 1
#define FORM_MENU 2
#define FORM_NEWORDER 3

```

```

#define FORM_PAYMENT      4
#define FORM_DELIVERY    5
#define FORM_ORDERSTATUS 6
#define FORM_STOCKLEVEL  7
#define FORM_EXIT        8
#define FORM_MAX         9

```

```

// CMD= HTML Command Enumeration and Name
#define CMD_NULL          0
#define CMD_PROCESS      1
#define CMD_NEWORDER_FORM 2
#define CMD_PAYMENT_FORM 3
#define CMD_DELIVERY_FORM 4
#define CMD_ORDERSTATUS_FORM 5
#define CMD_STOCKLEVEL_FORM 6
#define CMD_EXIT         7
#define CMD_SUBMIT       8
#define CMD_MENU_FORM    9
#define CMD_MAX          10

```

```

static CHAR * szCmds[] =
{
    "Unknown",
    "Process",
    "..NewOrder..",
    "..Payment..",
    "..Delivery..",
    "..Order-Status..",
    "..Stock-Level..",
    "..Exit..",
    "Submit",
    "Menu"
};

```

```

static CHAR * szFormLogin =
HTTPHdr "<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=\"TERMIN\" VALUE=\"-2\">"
"<INPUT TYPE=\"hidden\" NAME=\"SYCID\" 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>";

```

```

static CHAR * szMenuList =
"<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..\">";

```

```

static CHAR * HTMLTrailer =
"</BODY></HTML>";

```

```

static CHAR * TERMINDTOKEN = "TERMIN=";

```

```

static CHAR * SYNCIDTOKEN = "SYCID=";
static CHAR * FORMIDTOKEN = "FORMID=";
static CHAR * STATUSIDTOKEN = "STATUSID=";
static CHAR * CMDTOKEN = "CMD=";
static CHAR * NEWORDER_SERVICE = "NEWORDER";
static CHAR * PAYMENT_SERVICE = "PAYMENT";
static CHAR * ORDERSTATUS_SERVICE = "ORDERSTS";
static CHAR * DELIVERY_SERVICE = "DELIVERY";
static CHAR * STOCKLEVEL_SERVICE = "STOCKLVL";
static CHAR * ZIPPIC = "XXXXX-XXXX";

```

```

BOOL ProcessLogin(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC);
BOOL ProcessForm(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC);
BOOL ProcessNewOrder(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC);
BOOL ProcessPayment(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC);
BOOL ProcessDelivery(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC);
BOOL ProcessOrderStatus(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC);
BOOL ProcessStockLevel(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC);
VOID FormatLogin(CHAR * pMsg, CHAR * pAddText);
BOOL GetHidden(CHAR * pMsg, UINT * uFormId, INT * iSyncId, INT * iTermId);
BOOL GetCmd(CHAR * pMsg, CHAR * pWork, UINT uLen);
BOOL GetLongKey(LONG * lRslt, CHAR * pHTML, CHAR * pKey, TPCC_STATE * pTPCC);
BOOL GetIntKey(INT * iRslt, CHAR * pHTML, CHAR * pKey, TPCC_STATE * pTPCC);
BOOL GetShortKey(SHORT * sRslt, CHAR * pHTML, CHAR * pKey, TPCC_STATE * pTPCC);
BOOL GetStringKey(CHAR * szRslt, CHAR * pHTML, CHAR * pKey,
                  TPCC_STATE * pTPCC, UINT uMax);
BOOL GetAmountKey(DOUBLE * dRslt, CHAR * pHTML, CHAR * pKey,
                  TPCC_STATE * pTPCC);
BOOL GetKeyValue(CHAR * pHTML, CHAR * pKey, CHAR * pValue, UINT uMax);
VOID FormatLogin(CHAR * pOut, CHAR * pAddText);
VOID FormatMenu(CHAR * pOut, TPCC_STATE * pTPCC);
VOID FormatNewOrder(CHAR * pOut, TPCC_STATE * pTPCC);
VOID FormatPayment(CHAR * pOut, TPCC_STATE * pTPCC);
VOID FormatDelivery(CHAR * pOut, TPCC_STATE * pTPCC);
VOID FormatOrderStatus(CHAR * pOut, TPCC_STATE * pTPCC);
VOID FormatStockLevel(CHAR * pOut, TPCC_STATE * pTPCC);
VOID FormatFormHdr(CHAR * pOut, CHAR * pTitle, TPCC_STATE * pTPCC);
VOID FormatRespHdr(CHAR * pOut, CHAR * pTitle, TPCC_STATE * pTPCC);
VOID FormatHTMLString(CHAR * pOut, CHAR * pIn, UINT uLen);
VOID FormatString(CHAR * pOut, CHAR * pPic, CHAR * pIn);
VOID UtilStrCpy(CHAR * pDest, CHAR * pSrc, INT n);
BOOL CheckNumeric(CHAR * pNum);

```

```

//=====
//
// Function name: TPCCclear
//
//=====
BOOL TPCCclear(TPCC_STATE * pTPCC)
{
    pTPCC->ConnID = 0;
    pTPCC->sWid = 0;
    pTPCC->sDId = 0;
    pTPCC->iSyncId = 0;
    pTPCC->iTermId = -2;
    pTPCC->uFormId = FORM_NULL;
    pTPCC->iStatusId = 0;
    pTPCC->tsTMon.lTMDDataLen = 0;
    strcpy(pTPCC->ErrTxt, "");
    return (FALSE);
}

```

```

}; // TPCCClear

//=====
//
// Function name: TPCCHandler
//
//=====
UINT TPCCHandler(TPCC_STATE * pTPCC)
{
    INT iSyncId;
    INT iTermId;
    UINT uCmdId;
    UINT uRslt = TPCCSENDEND; // default error handling
    TERM_STATE * pTerm;

    pTPCC->iStatusId = STATUS_OK;
    if (GetHidden(pTPCC->RecvMsg, &pTPCC->uFormId, &iSyncId, &iTermId))
    {
        uRslt = TPCCSEND;
        FormatLogin(pTPCC->SendMsg, pTPCC->ErrTxt);
        goto HdlrXit;
    };
    if (iTermId > 0)
    {
        pTerm = TermGet(iTermId);
        if (pTerm == NULL)
        {
            uRslt = TPCCSEND;
            strcpy(pTPCC->ErrTxt, "Invalid Term Id");
            FormatLogin(pTPCC->SendMsg, pTPCC->ErrTxt);
            goto HdlrXit;
        };
        if (pTerm->ConnID != pTPCC->ConnID)
        {
            uRslt = TPCCSEND;
            strcpy(pTPCC->ErrTxt, "TermId vs ConnId Mismatch");
            FormatLogin(pTPCC->SendMsg, pTPCC->ErrTxt);
            goto HdlrXit;
        };
        pTPCC->sWid = pTerm->sWid;
        pTPCC->sDId = pTerm->sDId;
        pTPCC->iSyncId = pTerm->iSyncId;
        pTPCC->iTermId = pTerm->iTermId;
    };
    uCmdId = GetCmd(pTPCC->RecvMsg, pTPCC->szWork, sizeof(pTPCC->szWork));
    // Except for Submit(log in), sWid must already be set
    if (pTPCC->sWid == 0 && uCmdId != CMD_SUBMIT)
    {
        strcpy(pTPCC->ErrTxt, "Must log in first!");
        FormatLogin(pTPCC->SendMsg, pTPCC->ErrTxt);
        uRslt = TPCCSEND;
        goto HdlrXit;
    };
    // Check for multiple log in attempts
    if (pTPCC->sWid != 0 && uCmdId == CMD_SUBMIT)
    {
        strcpy(pTPCC->ErrTxt, ERRTXT_ALREADY_LOGGEDIN);
        pTPCC->iStatusId = ERR_ALREADY_LOGGEDIN;
        FormatMenu(pTPCC->SendMsg, pTPCC);
        uRslt = TPCCSEND;
        goto HdlrXit;
    };
}

```

```

};
// If not logging in, validate hidden fields
if (uCmdId != CMD_SUBMIT)
{
    if (iTermId != pTPCC->iTermId || iTermId != iSyncId)
    {
        sprintf(pTPCC->ErrTxt, "%s: Received %ld, %ld (%ld)",
            ERRTXT_TERMID, iTermId, iSyncId, pTPCC->iTermId);
        pTPCC->iStatusId = ERR_TERMID;
        FormatMenu(pTPCC->SendMsg, pTPCC);
        goto HdlrXit;
    };
};
// Process the command
switch (uCmdId)
{
    case CMD_SUBMIT:
        ProcessLogin(pTPCC->RecvMsg, pTPCC->SendMsg, pTPCC);
        break;
    case CMD_MENU_FORM:
        FormatMenu(pTPCC->SendMsg, pTPCC);
        break;
    case CMD_PROCESS:
        ProcessForm(pTPCC->RecvMsg, pTPCC->SendMsg, pTPCC);
        break;
    case CMD_NEWORDER_FORM:
        FormatNewOrder(pTPCC->SendMsg, pTPCC);
        break;
    case CMD_PAYMENT_FORM:
        FormatPayment(pTPCC->SendMsg, pTPCC);
        break;
    case CMD_DELIVERY_FORM:
        FormatDelivery(pTPCC->SendMsg, pTPCC);
        break;
    case CMD_ORDERSTATUS_FORM:
        FormatOrderStatus(pTPCC->SendMsg, pTPCC);
        break;
    case CMD_STOCKLEVEL_FORM:
        FormatStockLevel(pTPCC->SendMsg, pTPCC);
        break;
    case CMD_EXIT:
        TermFree(pTPCC->iTermId);
        strcpy(pTPCC->ErrTxt, "Logged Off");
        FormatLogin(pTPCC->SendMsg, pTPCC->ErrTxt);
        goto HdlrXit;
    default:
        strcpy(pTPCC->ErrTxt, ERRTXT_CMD_UNKNOWN);
        pTPCC->iStatusId = ERR_CMD_UNKNOWN;
        if (pTPCC->sWid == 0)
            FormatLogin(pTPCC->SendMsg, pTPCC->ErrTxt);
        else
            FormatMenu(pTPCC->SendMsg, pTPCC);
        break;
}; // switch (uCmdId)

uRslt = TPCCSEND;

HdlrXit:

return(uRslt);

```

```

}; // TPCCHandler

//=====
//
// Function name: ProcessLogin
//
// ProcessLogin extracts WId and DId from the incoming form. Assumes
// log in has not previously completed (sWId == 0 already verified).
//
// Result:
// FALSE - log in successful, sWId and sDId set in pTPCC,
//         pOut contains menu.
// TRUE - log in failed, pOut contains log in form with
//        error message.
//
//=====
BOOL ProcessLogin(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC)
{
    SHORT sWId;
    SHORT sDId;
    TERM_STATE * pTerm;

    if (GetShortKey(&sWId, pIn, "w_id", pTPCC))
    {
        FormatLogin(pOut, pTPCC->ErrTxt);
        return(TRUE);
    };
    if (sWId < 1)
    {
        sprintf(pTPCC->ErrTxt, "Warehouse Id (%d) Invalid", sWId);
        pTPCC->iStatusId = ERR_WID_INVALID;
        FormatLogin(pOut, pTPCC->ErrTxt);
        return(TRUE);
    };
    if (GetShortKey(&sDId, pIn, "d_id", pTPCC))
    {
        FormatLogin(pOut, pTPCC->ErrTxt);
        return(TRUE);
    };
    if (sDId < MIN_DId || sDId > MAX_DId)
    {
        sprintf(pTPCC->ErrTxt, "DId Out of Range(%ld,%ld) - %ld",
            MIN_DId, MAX_DId, sDId);
        pTPCC->iStatusId = ERR_DID_INVALID;
        FormatLogin(pOut, pTPCC->ErrTxt);
        return(TRUE);
    };
    pTerm = TermAlloc();
    if (pTerm == NULL)
    {
        sprintf(pTPCC->ErrTxt, "Unable to Allocate Terminal Entry");
        pTPCC->iStatusId = ERR_TERM_ALLOC;
        FormatLogin(pOut, pTPCC->ErrTxt);
        return(TRUE);
    };
    pTerm->ConnID = pTPCC->ConnID;
    pTerm->iSyncId = pTerm->iTermId;
    pTerm->sWId = abs(sWId);
    pTerm->sDId = abs(sDId);
    pTPCC->iTermId = pTerm->iTermId;
    pTPCC->iSyncId = pTerm->iSyncId;

```

```

    pTPCC->sWId = pTerm->sWId;
    pTPCC->sDId = pTerm->sDId;
    FormatMenu(pOut, pTPCC);
    return(FALSE);
}; // ProcessLogin

//=====
//
// Function name: ProcessForm
//
// ProcessForm uses pTPCC->uFormId to determine which form input is
// present and ready for processing. Actual processing is done by
// the form specific routine.
//
// Result:
// FALSE - form processed, pOut contains response.
// TRUE - error processing form input, pOut contains reason.
//
//=====
BOOL ProcessForm(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC)
{
    switch (pTPCC->uFormId )
    {
        case FORM_NEWORDER:
            return(ProcessNewOrder(pIn, pOut, pTPCC));
        case FORM_PAYMENT:
            return(ProcessPayment(pIn, pOut, pTPCC));
        case FORM_DELIVERY:
            return(ProcessDelivery(pIn, pOut, pTPCC));
        case FORM_ORDERSTATUS:
            return(ProcessOrderStatus(pIn, pOut, pTPCC));
        case FORM_STOCKLEVEL:
            return(ProcessStockLevel(pIn, pOut, pTPCC));
        default:
            sprintf(pTPCC->ErrTxt, "%s (%ld)",
                ERRTXT_FORM_UNKNOWN, pTPCC->uFormId);
            pTPCC->iStatusId = ERR_FORM_UNKNOWN;
            FormatMenu(pOut, pTPCC);
            break;
    }
    return(TRUE);
}; // ProcessForm

//=====
//
// Function name: ProcessNewOrder
//
// ProcessNewOrder extracts the input data fields from pIn, processes
// the data, and returns a response in pOut.
//
// Result:
// FALSE - NewOrder processed successfully.
// TRUE - NewOrder processing failed.
//
//=====
BOOL ProcessNewOrder(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC)
{
    NEW_ORDER_DATA * pnod;
    TMON_STATE * pTMon;
    CHAR szKey[20];
    CHAR szCredit[14];

```

```

CHAR * ptr;
UINT u;
BOOL bDone = FALSE;
BOOL bTMRslt;
BOOL bTPRslt;
INT iTPRslt;

pTMon = &pTPCC->tsTMon;
pTMon->lTMDDataLen = sizeof(NEW_ORDER_DATA);
memset(pTMon->pTMDData, 0, pTMon->lTMDDataLen);
pnod = (NEW_ORDER_DATA *) pTMon->pTMDData;
pnod->w_id = pTPCC->sWid;
if (GetShortKey(&pnod->d_id, pIn, "DID*", pTPCC))
{
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (pnod->d_id < MIN_Did || pnod->d_id > MAX_Did)
{
    sprintf(pTPCC->ErrTxt, "DID Out of Range(%ld,%ld) - %ld",
        MIN_Did, MAX_Did, pnod->d_id);
    pTPCC->iStatusId = ERR_DID_INVALID;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (GetLongKey(&pnod->c_id, pIn, "CID*", pTPCC))
{
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
pnod->o_ol_cnt = 0;
ptr = pIn;
for(u=0; u < MAX_OL; u++)
{
    sprintf(szKey, "SP%2.2d*", u);
    ptr = strstr(ptr, szKey);
    if (GetShortKey(&pnod->Ol[u].ol_supply_w_id, ptr, szKey, pTPCC))
    {
        FormatMenu(pOut, pTPCC);
        return(TRUE);
    };
    sprintf(szKey, "IID%2.2d*", u);
    if (GetLongKey(&pnod->Ol[u].ol_i_id, ptr, szKey, pTPCC))
    {
        FormatMenu(pOut, pTPCC);
        return(TRUE);
    };
    sprintf(szKey, "Qty%2.2d*", u);
    if (GetShortKey(&pnod->Ol[u].ol_quantity, ptr, szKey, pTPCC))
    {
        FormatMenu(pOut, pTPCC);
        return(TRUE);
    };
    if (pnod->Ol[u].ol_i_id != 0)
    {
        // Check for prior blank lines
        if (bDone)
        {
            strcat(pTPCC->ErrTxt, "Embedded Empty Order Lines");
            pTPCC->iStatusId = ERR_EMBEDDED_EMPTY_OL;
            FormatMenu(pOut, pTPCC);
        }
    }
}

```

```

return(TRUE);
};
if (pnod->Ol[u].ol_supply_w_id < 1)
{
    sprintf(pTPCC->ErrTxt,
        "Order Line %ld Contains Invalid Wid %d",
        u, pnod->Ol[u].ol_supply_w_id);
    pTPCC->iStatusId = ERR_WID_INVALID;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (pnod->Ol[u].ol_quantity < MIN_QUANTITY ||
    pnod->Ol[u].ol_quantity > MAX_QUANTITY)
{
    sprintf(pTPCC->ErrTxt,
        "Order Line %ld Contains Invalid Qty %d",
        u, pnod->Ol[u].ol_quantity);
    pTPCC->iStatusId = ERR_QUANTITY_INVALID;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
pnod->o_ol_cnt++;
} // if (ol_i_id !=0)
else
{
    if (pnod->Ol[u].ol_supply_w_id != 0)
    {
        sprintf(pTPCC->ErrTxt,
            "Order Line %ld WID Supplied with No Item", u);
        pTPCC->iStatusId = ERR_OL_INVALID;
        FormatMenu(pOut, pTPCC);
        return(TRUE);
    };
    if (pnod->Ol[u].ol_quantity != 0)
    {
        sprintf(pTPCC->ErrTxt,
            "Order Line %ld Qty Supplied with No Item", u);
        pTPCC->iStatusId = ERR_OL_INVALID;
        FormatMenu(pOut, pTPCC);
        return(TRUE);
    };
    bDone = TRUE;
}; // empty order line
}; // for (u < MAX_OL)

if (pnod->o_ol_cnt < MIN_OL)
{
    sprintf(pTPCC->ErrTxt, "Too Few Order Lines %d", pnod->o_ol_cnt);
    pTPCC->iStatusId = ERR_OL_COUNT;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
bTMRslt = TMTran(NEWORDER_SERVICE, pTMon, &bTPRslt, &iTPRslt);
pnod = (NEW_ORDER_DATA *) pTMon->pTMDData;
if (bTMRslt)
{
    pTPCC->iStatusId = ERR_TM_INTERFACE;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
// Exclude invalid item id case

```

```

if (bTPRslt && iTPRslt < SVC_NOERROR)
{
    sprintf(pTPCC->ErrTxt,
        "New Order Service Returned Error(%ld): %s",
        iTPRslt,pnod->execution_status);
    pTPCC->iStatusId = ERR_SERVICE_RSLT;
    FormatMenu(pOut,pTPCC);
    return(TRUE);
};
if (iTPRslt == SVC_BADITEMID)
    pTPCC->iStatusId = INVALID_IID;

FormatRespHdr(pOut,"TPC-C New Order",pTPCC);
sprintf(pOut + strlen(pOut),
    "<PRE>
    Warehouse: %4.4d District: %2.2d
    pnod->w_id, pnod->d_id);
if (!bTPRslt)
{
    sprintf(pOut + strlen(pOut),
        "Date: %2.2d-%2.2d-%4.4d %2.2d:%2.2d:%2.2d <BR>",
        pnod->o_entry_d.day, pnod->o_entry_d.month,
        pnod->o_entry_d.year, pnod->o_entry_d.hour,
        pnod->o_entry_d.minute, pnod->o_entry_d.second);
}
else
{
    sprintf(pOut + strlen(pOut), "Date:<BR>");
};
FormatHTMLString(pTPCC->szWork, pnod->c_last, NAME_LEN);
FormatHTMLString(szCredit, pnod->c_credit, 2);
sprintf(pOut + strlen(pOut),
    "Customer: %4.4d Name: %s Credit: %s ",
    pnod->c_id, pTPCC->szWork, szCredit);
if (!bTPRslt)
{
    sprintf(pOut + strlen(pOut),
        "%%Disc: %5.2f <BR>", pnod->c_discount * 100);
    sprintf(pOut + strlen(pOut),
        "Order Number: %8.8d Number of Lines: %2.2d W_tax: %5.2f
D_tax: %5.2f <BR><BR>",
        pnod->o_id, pnod->o_ol_cnt, pnod->w_tax * 100, pnod->d_tax * 100);
    strcat(pOut, " Supp_W Item_Id Item Name Qty Stock
B/G Price Amount<BR>");
    for (u = 0; u < (UINT) pnod->o_ol_cnt; u++)
    {
        FormatHTMLString(pTPCC->szWork, pnod->ol[u].ol_i_name, 24);
        sprintf(pOut + strlen(pOut),
            " %4.4d %6.6d %s %2.2d %3.3d %1.1s %6.2f
$%7.2f <BR>",
            pnod->ol[u].ol_supply_w_id, pnod->ol[u].ol_i_id,
            pTPCC->szWork, pnod->ol[u].ol_quantity, pnod->ol[u].ol_stock,
            pnod->ol[u].ol_brand_generic, pnod->ol[u].ol_i_price,
            pnod->ol[u].ol_amount );
    }
} // if (!bTPRslt)
else
{
    strcat(pOut, "%Disc:<BR>");
    sprintf(pOut + strlen(pOut),

```

```

        "Order Number: %8.8d Number of Lines: W_tax:
D_tax:<BR><BR>",
        pnod->o_id);
    strcat(pOut,
        " Supp_W Item_Id Item Name Qty Stock B/G
Price Amount<BR>");
    u = 0;
};
for(; u < MAX_OL; u++)
    strcat(pOut, "<BR>");
if (!bTPRslt)
{
    sprintf(pOut + strlen(pOut),
        "Execution Status: %24.24s Total: $%8.2f ",
        pnod->execution_status, pnod->total_amount);
}
else
{
    sprintf(pOut + strlen(pOut),
        "Execution Status: %24.24s Total:",
        pnod->execution_status);
};
sprintf(pOut + strlen(pOut),
    "</PRE><HR><BR>%s</FORM>%s", szMenuList, HTMLTrailer);

return(FALSE);
}; // ProcessNewOrder

//=====
//
// Function name: ProcessPayment
//
// ProcessPayment extracts the input data fields from pIn, processes
// the data, and returns a response in pOut.
//
// Result:
// FALSE - Payment processed successfully.
// TRUE - Payment processing failed.
//=====
BOOL ProcessPayment(CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC)
{
    PAYMENT_DATA * ppd;
    TMON_STATE * pTMon;
    BOOL bTMRslt;
    BOOL bTPRslt;
    INT iTPRslt;
    CHAR * pCredit;
    INT icDLines;
    CHAR szWork2[60];
    CHAR szWork3[60];
    CHAR szWork4[60];
    CHAR szZip1[20];
    CHAR szZip2[20];
    INT i;

    pTMon = &pTPCC->tsTMon;
    pTMon->lTMDDataLen = sizeof(PAYMENT_DATA);
    memset(pTMon->pTMDData, 0, pTMon->lTMDDataLen);
    ppd = (PAYMENT_DATA *) pTMon->pTMDData;

```

```

ppd->w_id = pTPCC->SWid;
// Get and validate DID
if (GetShortKey(&ppd->d_id,pIn, "DID*", pTPCC)
{
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (ppd->d_id < MIN_DID || ppd->d_id > MAX_DID)
{
    sprintf(pTPCC->ErrTxt, "DId Out of Range(%ld,%ld) - %ld",
        MIN_DID, MAX_DID, ppd->d_id);
    pTPCC->iStatusId = ERR_DID_INVALID;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
// Get and validate customer Id and name
if (GetLongKey(&ppd->c_id,pIn, "CID*", pTPCC)
{
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (GetStringKey(ppd->c_last,pIn, "CLT*", pTPCC, NAME_LEN)
{
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (ppd->c_id == 0 && ppd->c_last[0] == 0)
{
    strcpy(pTPCC->ErrTxt, "Error - Customer Id and Name Empty");
    pTPCC->iStatusId = ERR_IDANDNAME_EMPTY;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (ppd->c_id != 0 && ppd->c_last[0] != 0)
{
    strcpy(pTPCC->ErrTxt,
        "Error - Specify Customer Id or Name, not Both");
    pTPCC->iStatusId = ERR_IDANDNAME_ENTERED;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
// Get and validate customer DID
if (GetShortKey(&ppd->c_d_id,pIn, "CDI*", pTPCC)
{
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (ppd->c_d_id < MIN_DID || ppd->c_d_id > MAX_DID)
{
    sprintf(pTPCC->ErrTxt, "Cust DId Out of Range(%ld,%ld) - %ld",
        MIN_DID, MAX_DID, ppd->d_id);
    pTPCC->iStatusId = ERR_DID_INVALID;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
// Get and validate customer WId
if (GetShortKey(&ppd->c_w_id,pIn, "CWI*", pTPCC)
{
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};

```

```

if (ppd->c_w_id < 1)
{
    sprintf(pTPCC->ErrTxt,
        "Payment Contains Invalid Customer WId %d",
        ppd->c_w_id);
    pTPCC->iStatusId = ERR_WID_INVALID;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
// Get and validate amount
if (GetAmountKey(&ppd->h_amount,pIn, "HAM*", pTPCC)
{
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (ppd->h_amount <= 0)
{
    sprintf(pTPCC->ErrTxt,
        "Payment Amount Negative or Missing");
    pTPCC->iStatusId = ERR_AMOUNT_INVALID;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
bTMRslt = TMTran(PAYMENT_SERVICE, pTMon, &bTPRslt, &iTPRslt);
ppd = (PAYMENT_DATA *) pTMon->pTMDData;
if (bTMRslt)
{
    pTPCC->iStatusId = ERR_TM_INTERFACE;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
if (bTPRslt)
{
    sprintf(pTPCC->ErrTxt,
        "Payment Service Returned Error(%ld): %s",
        iTPrslt, ppd->execution_status);
    pTPCC->iStatusId = ERR_SERVICE_RSLT;
    FormatMenu(pOut, pTPCC);
    return(TRUE);
};
FormatRespHdr(pOut, "TPC-C Payment", pTPCC);
sprintf(pOut + strlen(pOut),
    "<PRE>
    Date: %2.2d-%2.2d-%4.4d %2.2d:%2.2d:%2.2d <BR><BR>"
    "Warehouse: %4.4d
    District: %2.2d<BR>",
    ppd->h_date.day, ppd->h_date.month,
    ppd->h_date.year, ppd->h_date.hour,
    ppd->h_date.minute, ppd->h_date.second,
    ppd->w_id, ppd->d_id);
FormatHTMLString(szWork2, ppd->w_street_1, ADDR_LEN);
FormatHTMLString(szWork3, ppd->d_street_1, ADDR_LEN);
sprintf(pOut + strlen(pOut),
    "%s
    %s<BR>", szWork2, szWork3);
FormatHTMLString(szWork2, ppd->w_street_2, ADDR_LEN);
FormatHTMLString(szWork3, ppd->d_street_2, ADDR_LEN);
sprintf(pOut + strlen(pOut),
    "%s
    %s<BR>", szWork2, szWork3);
FormatHTMLString(pTPCC->szWork, ppd->w_city, ADDR_LEN);
FormatHTMLString(szWork2, ppd->d_city, ADDR_LEN);

```



```

FormatHTMLString (szWork3,ppd->w_state,STATE_LEN);
FormatHTMLString (szWork4,ppd->d_state,STATE_LEN);
FormatString (szZip1,ZIPPIC,ppd->w_zip);
FormatString (szZip2,ZIPPIC,ppd->d_zip);
sprintf (pOut + strlen(pOut),
"%s %s %10.10s %s %s %10.10s<BR><BR>",
pTPCC->szWork,szWork3,szZip1,szWork2,szWork4,szZip2);
FormatHTMLString (szWork2,ppd->c_first,NAME_LEN);
FormatHTMLString (szWork3,ppd->c_middle,2);
FormatHTMLString (szWork4,ppd->c_last,NAME_LEN);
sprintf (pOut + strlen(pOut),
"Customer: %4.4d Cust-Warehouse: %4.4d Cust-District: %2.2d<BR>"
"Name: %s %s %s Since: %2.2d-%2.2d-%4.4d<BR>",
ppd->c_id,ppd->c_w_id,ppd->c_d_id,
szWork2,szWork3,szWork4,
ppd->c_since.day,ppd->c_since.month,ppd->c_since.year);
FormatHTMLString (pTPCC->szWork,ppd->c_street_1,ADDR_LEN);
FormatHTMLString (szWork2,ppd->c_credit,2);
FormatHTMLString (szWork3,ppd->d_street_2,ADDR_LEN);
sprintf (pOut + strlen(pOut),
" %s Credit: %s<BR>"
"%s %s %%Disc: %5.2f<BR>",
pTPCC->szWork,szWork2,szWork3,ppd->c_discount * 100);
FormatHTMLString (szWork2,ppd->c_city,ADDR_LEN);
FormatHTMLString (szWork3,ppd->c_state,STATE_LEN);
FormatString (szZip1,ZIPPIC,ppd->c_zip);
FormatString (szWork4,"XXXXXX-XXX-XXX-XXXX",ppd->c_phone);
sprintf (pOut + strlen(pOut),
" %s %s %10.10s Phone: %-19.19s<BR><BR>"
"Amount Paid: $%7.2f New Cust Balance: $%14.2f<BR>"
"Credit Limit: $%13.2f<BR><BR>",
szWork2,szWork3,szZip1,szWork4,
ppd->h_amount,ppd->c_balance,ppd->c_credit_lim);
pCredit = ppd->c_credit;
if (*pCredit == 'B' && *(pCredit + 1) == 'C')
{
pCredit = ppd->c_data;
iCDLines = strlen(pCredit) / 50;
for(i = 0; i < 4; i++, pCredit += 50)
{
if (i <= iCDLines)
UtilStrCpy (szWork2,pCredit,50);
else
szWork2[0] = 0;
FormatHTMLString (szWork3,szWork2,50);
if (!i)
sprintf (pOut + strlen(pOut),
"Cust-Data: %s<BR>",szWork3);
else
sprintf (pOut + strlen(pOut),
"%s<BR>",szWork3);
};
}
else
strcat (pOut,"Cust-Data: <BR><BR><BR><BR>");
sprintf (pOut + strlen(pOut),
"</PRE><HR><BR>%s</FORM>%s",szMenuList,HTMLTrailer);

return (FALSE);
}; // ProcessPayment

```

```

//=====
//
// Function name: ProcessDelivery
//
// ProcessDelivery extracts the input data fields from pIn, processes
// the data, and returns a response in pOut.
//
// Result:
// FALSE - Delivery processed successfully.
// TRUE - Delivery processing failed.
//=====
BOOL ProcessDelivery (CHAR * pIn,CHAR * pOut,TPCC_STATE * pTPCC)
{
DELIVERY_DATA * pdd;
TMON_STATE * pTMon;
BOOL bTMRslt;

pTMon = &pTPCC->tsTMon;
pTMon->lTMDDataLen = sizeof (DELIVERY_DATA);
memset (pTMon->pTMDData,0,pTMon->lTMDDataLen);
pdd = (DELIVERY_DATA *) pTMon->pTMDData;
pdd->w_id = pTPCC->swId;
// Get and validate carrier id
if (GetShortKey (&pdd->o_carrier_id,pIn,"OCD*",pTPCC)
{
FormatMenu (pOut,pTPCC);
return (TRUE);
};
if (pdd->o_carrier_id < MIN_CARRIER ||
pdd->o_carrier_id > MAX_CARRIER)
{
sprintf (pTPCC->ErrTxt,"Carrier Id Out of Range(%ld,%ld) - %ld",
MIN_CARRIER,MAX_CARRIER,pdd->o_carrier_id);
pTPCC->iStatusId = ERR_CARRIER_INVALID;
FormatMenu (pOut,pTPCC);
return (TRUE);
};
GetLocalTime (&pdd->QTime);
bTMRslt = TMPost (DELIVERY_SERVICE,pTMon);
if (bTMRslt)
{
pTPCC->iStatusId = ERR_TM_INTERFACE;
FormatMenu (pOut,pTPCC);
return (TRUE);
};
strcpy (pdd->execution_status,"Delivery has been queued.");
FormatRespHdr (pOut,"TPC-C Delivery",pTPCC);
sprintf (pOut + strlen(pOut),
"<PRE> Delivery<BR>"
"Warehouse: %4.4d<BR><BR>"
"Carrier Number: %2.2d<BR><BR>"
"Execution Status: %25.25s<BR>",
pdd->w_id,pdd->o_carrier_id,pdd->execution_status);
sprintf (pOut + strlen(pOut),
"</PRE><HR><BR>%s</FORM>%s",szMenuList,HTMLTrailer);

return (FALSE);
}; // ProcessDelivery

```

```

//=====
//
// Function name: ProcessOrderStatus
//
// ProcessOrderStatus extracts the input data fields from pIn,
// processes the data, and returns a response in pOut.
//
// Result:
// FALSE - OrderStatus processed successfully.
// TRUE - OrderStatus processing failed.
//=====
BOOL ProcessOrderStatus(CHAR * pIn,CHAR * pOut,TPCC_STATE * pTPCC)
{
    ORDER_STATUS_DATA * posd;
    TMON_STATE * pTMon;
    INT i;
    CHAR szWork2[50];
    CHAR szWork3[50];
    BOOL bTMRslt;
    BOOL bTPRslt;
    INT iTPrslt;

    pTMon = &pTPCC->tsTMon;
    pTMon->lTMDDataLen = sizeof(ORDER_STATUS_DATA);
    memset(pTMon->pTMDData,0,pTMon->lTMDDataLen);
    posd = (ORDER_STATUS_DATA *) pTMon->pTMDData;
    posd->w_id = pTPCC->swId;
    if (GetShortKey(&posd->d_id,pIn,"DID*",pTPCC))
    {
        FormatMenu(pOut,pTPCC);
        return(TRUE);
    };
    if (posd->d_id < MIN_DID || posd->d_id > MAX_DID)
    {
        sprintf(pTPCC->ErrTxt,"DID Out of Range(%ld,%ld) - %ld",
            MIN_DID,MAX_DID,posd->d_id);
        pTPCC->iStatusId = ERR_DID_INVALID;
        FormatMenu(pOut,pTPCC);
        return(TRUE);
    };
    if (GetLongKey(&posd->c_id,pIn,"CID*",pTPCC))
    {
        FormatMenu(pOut,pTPCC);
        return(TRUE);
    };
    if (GetStringKey(posd->c_last,pIn,"CLT*",pTPCC,NAME_LEN))
    {
        FormatMenu(pOut,pTPCC);
        return(TRUE);
    };
    if (posd->c_id == 0 && posd->c_last[0] == 0)
    {
        strcpy(pTPCC->ErrTxt,"Error - Customer Id and Name Empty");
        pTPCC->iStatusId = ERR_IDANDNAME_EMPTY;
        FormatMenu(pOut,pTPCC);
        return(TRUE);
    };
    if (posd->c_id != 0 && posd->c_last[0] != 0)
    {

```

```

        strcpy(pTPCC->ErrTxt,
            "Error - Specify Customer Id or Name, not Both");
        pTPCC->iStatusId = ERR_IDANDNAME_ENTERED;
        FormatMenu(pOut,pTPCC);
        return(TRUE);
    };
    bTMRslt = TMTran(ORDERSTATUS_SERVICE,pTMon,&bTPRslt,&iTPrslt);
    posd = (ORDER_STATUS_DATA *) pTMon->pTMDData;
    if (bTMRslt)
    {
        pTPCC->iStatusId = ERR_TM_INTERFACE;
        FormatMenu(pOut,pTPCC);
        return(TRUE);
    };
    if (bTPRslt)
    {
        sprintf(pTPCC->ErrTxt,
            "Order Status Service Returned Error(%ld): %s",
            iTPrslt,posd->execution_status);
        pTPCC->iStatusId = ERR_SERVICE_RSLT;
        FormatMenu(pOut,pTPCC);
        return(TRUE);
    };
    FormatRespHdr(pOut,"TPC-C Order-Status",pTPCC);
    sprintf(pOut + strlen(pOut),
        "<PRE>                                Order-Status<BR>"
        "Warehouse: %4.4d  District: %2.2d<BR>",
        posd->w_id,posd->d_id);
    FormatHTMLString(pTPCC->szWork,posd->c_first,NAME_LEN);
    FormatHTMLString(szWork2,posd->c_middle,2);
    FormatHTMLString(szWork3,posd->c_last,NAME_LEN);
    sprintf(pOut + strlen(pOut),
        "Customer: %4.4d  Name: %s %s %s<BR>"
        "Cust-Balance: $%9.2f<BR><BR>",
        posd->c_id,pTPCC->szWork,szWork2,szWork3,posd->c_balance);
    sprintf(pOut + strlen(pOut),
        "Order-Number: %8.8d  Entry-Date: %2.2d-%2.2d-%4.4d
%2.2d:%2.2d:%2.2d  Carrier-Number: %2.2d<BR>"
        "Supply-W  Item-Id  Qty  Amount  Delivery-Date<BR>",
        posd->o_id,posd->o_entry_d.day,posd->o_entry_d.month,
        posd->o_entry_d.year,posd->o_entry_d.hour,
        posd->o_entry_d.minute,posd->o_entry_d.second,
        posd->o_carrier_id);
    for(i = 0; i < posd->o_ol_cnt; i++)
    {
        sprintf(pOut + strlen(pOut),
            " %4.4d      %6.6d      %2.2d      %$8.2f      %2.2d-%2.2d-
%4.4d<BR>",
            posd->OlOrderStatusData[i].ol_supply_w_id,
            posd->OlOrderStatusData[i].ol_i_id,
            posd->OlOrderStatusData[i].ol_quantity,
            posd->OlOrderStatusData[i].ol_amount,
            posd->OlOrderStatusData[i].ol_delivery_d.day,
            posd->OlOrderStatusData[i].ol_delivery_d.month,
            posd->OlOrderStatusData[i].ol_delivery_d.year);
    };
    sprintf(pOut + strlen(pOut),
        "<BR></PRE><HR><BR>%s</FORM>%s",szMenuList,HTMLTrailer);

    return(FALSE);

```

```

}; // ProcessOrderStatus

//=====
//
// Function name: ProcessStockLevel
//
// ProcessStockLevel extracts the input data fields from pIn,
// processes the data, and returns a response in pOut.
//
// Result:
// FALSE - StockLevel processed successfully.
// TRUE - StockLevel processing failed.
//
//=====
BOOL ProcessStockLevel (CHAR * pIn, CHAR * pOut, TPCC_STATE * pTPCC)
{
    STOCK_LEVEL_DATA * psld;
    TMON_STATE * pTMon;
    BOOL bTMRslt;
    BOOL bTPRslt;
    INT iTPRslt;

    pTMon = &pTPCC->tsTMon;
    pTMon->lTMDDataLen = sizeof(STOCK_LEVEL_DATA);
    memset(pTMon->pTMDData, 0, pTMon->lTMDDataLen);
    psld = (STOCK_LEVEL_DATA *) pTMon->pTMDData;
    psld->w_id = pTPCC->sWId;
    psld->d_id = pTPCC->sDId;
    psld->low_stock = 0;
    psld->execution_status[0] = 0;
    if (GetShortKey(&psld->thresh_hold, pIn, "TT*", pTPCC))
    {
        FormatMenu(pOut, pTPCC);
        return(TRUE);
    };
    if (psld->thresh_hold < MIN_THRESHOLD ||
        psld->thresh_hold > MAX_THRESHOLD)
    {
        sprintf(pTPCC->ErrTxt, "Threshold Out of Range(%ld,%ld) - %ld",
            MIN_THRESHOLD, MAX_THRESHOLD, psld->thresh_hold);
        pTPCC->iStatusId = ERR_THRESHOLD_RANGE;
        FormatMenu(pOut, pTPCC);
        return(TRUE);
    };
    bTMRslt = TMTran(STOCKLEVEL_SERVICE, pTMon, &bTPRslt, &iTPRslt);
    psld = (STOCK_LEVEL_DATA *) pTMon->pTMDData;
    if (bTMRslt)
    {
        pTPCC->iStatusId = ERR_TM_INTERFACE;
        FormatMenu(pOut, pTPCC);
        return(TRUE);
    };
    if (bTPRslt)
    {
        sprintf(pTPCC->ErrTxt,
            "Stock Level Service Returned Error(%ld): %s",
            iTPRslt, psld->execution_status);
        pTPCC->iStatusId = ERR_SERVICE_RSLT;
        FormatMenu(pOut, pTPCC);
        return(TRUE);
    };
};

```

```

};

FormatRespHdr(pOut, "TPC-C Stock Level", pTPCC);
sprintf(pOut + strlen(pOut),
    "<PRE>
    Stock-Level<BR><BR>"
    "Warehouse: %4.4d District: %2.2d<BR><BR>"
    "Stock Level Threshold: %2.2d<BR><BR>"
    "low stock: %3.3ld</PRE><BR><HR>"
    "%s</FORM>%s",
    pTPCC->sWId, pTPCC->sDId, psld->thresh_hold, psld->low_stock,
    szMenuList, HTMLTrailer);

return(FALSE);
}; // ProcessStockLevel

//=====
//
// Function name: GetHidden
//
//=====
BOOL GetHidden(CHAR * pMsg, UINT * uFormId, INT * iSyncId, INT * iTermId)
{
    CHAR * pPtr;
    BOOL bRslt = TRUE;

    // Extract TERMID
    pPtr = strstr(pMsg, TERMIDTOKEN);
    if (pPtr == NULL)
        goto xit;
    pPtr += strlen(TERMIDTOKEN);
    *iTermId = atoi(pPtr);

    // Extract SYNCID
    pPtr = strstr(pMsg, SYNCIDTOKEN);
    if (pPtr == NULL)
        goto xit;
    pPtr += strlen(SYNCIDTOKEN);
    *iSyncId = atoi(pPtr);

    // Extract FORMID
    pPtr = strstr(pMsg, FORMIDTOKEN);
    if (pPtr == NULL)
        goto xit;
    pPtr += strlen(FORMIDTOKEN);
    *uFormId = abs(atoi(pPtr));

    bRslt = FALSE;

xit:

    return(bRslt);
}; // GetHidden

//=====
//
// Function name: GetCmd
//
//=====
BOOL GetCmd(CHAR * pMsg, CHAR * pWork, UINT uLen)

```

```

{
  UINT u;
  CHAR * ptr;
  CHAR * pUpd;

  // Check for CMD key
  if (!(ptr = strstr(pMsg,CMDTOKEN)))
    return(CMD_NULL);
  ptr += sizeof(CMDTOKEN);
  pUpd = pWork;
  while (*ptr && *ptr != '&')
    *pUpd++ = *ptr++;
  *pUpd = 0;

  // Convert command name into command index
  for(u=0; u < CMD_MAX; u++)
  {
    if (!strcmp(szCmds[u],pWork))
      return(u);
  };

  // Command string not found
  return(CMD_NULL);
}; // GetCmd

//=====
//
// Function name: GetLongKey
//
//=====
BOOL GetLongKey(LONG * lRslt,CHAR * pHTML,CHAR * pKey,TPCC_STATE * pTPCC)
{
  if (GetKeyValue(pHTML,pKey,pTPCC->szWork,sizeof(pTPCC->szWork)))
  {
    sprintf(pTPCC->ErrTxt,"Error - Missing %s Key",pKey);
    pTPCC->iStatusId = ERR_MISSING_KEY;
    return(TRUE);
  };
  if (pTPCC->szWork[0] != 0 )
  {
    if (CheckNumeric(pTPCC->szWork))
    {
      sprintf(pTPCC->ErrTxt,"Error - %s Value Not Numeric",pKey);
      pTPCC->iStatusId = ERR_NOT_NUMERIC;
      return(TRUE);
    };
  };
  *lRslt = atol(pTPCC->szWork);
  return(FALSE);
}; // GetLongKey

//=====
//
// Function name: GetIntKey
//
//=====
BOOL GetIntKey(INT * iRslt,CHAR * pHTML,CHAR * pKey,TPCC_STATE * pTPCC)
{
  if (GetKeyValue(pHTML,pKey,pTPCC->szWork,sizeof(pTPCC->szWork)))
  {

```

```

    sprintf(pTPCC->ErrTxt,"Error - Missing %s Key",pKey);
    pTPCC->iStatusId = ERR_MISSING_KEY;
    return(TRUE);
  };
  if (pTPCC->szWork[0] != 0 )
  {
    if (CheckNumeric(pTPCC->szWork))
    {
      sprintf(pTPCC->ErrTxt,"Error - %s Value Not Numeric",pKey);
      pTPCC->iStatusId = ERR_NOT_NUMERIC;
      return(TRUE);
    };
  };
  *iRslt = atoi(pTPCC->szWork);
  return(FALSE);
}; // GetIntKey

//=====
//
// Function name: GetShortKey
//
//=====
BOOL GetShortKey(SHORT * sRslt,CHAR * pHTML,CHAR * pKey,TPCC_STATE *
pTPCC)
{
  if (GetKeyValue(pHTML,pKey,pTPCC->szWork,sizeof(pTPCC->szWork)))
  {
    sprintf(pTPCC->ErrTxt,"Error - Missing %s Key",pKey);
    pTPCC->iStatusId = ERR_MISSING_KEY;
    return(TRUE);
  };
  if (pTPCC->szWork[0] != 0 )
  {
    if (CheckNumeric(pTPCC->szWork))
    {
      sprintf(pTPCC->ErrTxt,"Error - %s Value Not Numeric",pKey);
      pTPCC->iStatusId = ERR_NOT_NUMERIC;
      return(TRUE);
    };
  };
  *sRslt = (SHORT) atoi(pTPCC->szWork);
  return(FALSE);
}; // GetShortKey

//=====
//
// Function name: GetStringKey
//
//=====
BOOL GetStringKey(CHAR * szRslt,CHAR * pHTML,CHAR * pKey,
TPCC_STATE * pTPCC,UINT uMax)
{
  UINT uLen;
  if (GetKeyValue(pHTML,pKey,pTPCC->szWork,sizeof(pTPCC->szWork)))
  {
    sprintf(pTPCC->ErrTxt,"Error - Missing %s Key",pKey);
    pTPCC->iStatusId = ERR_MISSING_KEY;
    return(TRUE);
  };
  uLen = strlen(pTPCC->szWork);
  if (uLen > uMax)

```

```

{
    sprintf(pTPCC->ErrTxt,
        "Error - %s Key Input (%ld) Too Long (%ld)"
        ,pKey,uLen,uMax);
    pTPCC->iStatusId = ERR_INPUT_TOOLONG;
    return(TRUE);
};
_strupr(pTPCC->szWork);
strcpy(szRslt,pTPCC->szWork);
return(FALSE);
}; // GetStringKey

//=====
//
// Function name: GetAmountKey
//
//=====
BOOL GetAmountKey(DOUBLE * dRslt,CHAR * pHTML,CHAR * pKey,
                  TPCC_STATE * pTPCC)
{
    CHAR * ptr;
    BOOL bInvalid = FALSE;

    if (GetKeyValue(pHTML,pKey,pTPCC->szWork,sizeof(pTPCC->szWork)))
    {
        sprintf(pTPCC->ErrTxt,"Error - Missing %s Key",pKey);
        pTPCC->iStatusId = ERR_MISSING_KEY;
        return(TRUE);
    };
    ptr = pTPCC->szWork;
    while(*ptr)
    {
        if (*ptr == '.')
        {
            ptr++;
            if (!*ptr)
                break;
            if (*ptr < '0' || *ptr > '9')
            {
                bInvalid = TRUE;
                break;
            };
            ptr++;
            if (!*ptr)
                break;
            if (*ptr < '0' || *ptr > '9')
            {
                bInvalid = TRUE;
                break;
            };
            ptr++;
            if (*ptr)
            {
                bInvalid = TRUE;
                break;
            };
            break;
        }
        else
            if (*ptr < '0' || *ptr > '9')
            {

```

```

                bInvalid = TRUE;
                break;
            };
            ptr++;
        }; // while(!*ptr)

    if (!bInvalid)
        *dRslt = atof(pTPCC->szWork);
    else
    {
        sprintf(pTPCC->ErrTxt,
            "Error - Invalid Amount Format (%s)",pTPCC->szWork);
        pTPCC->iStatusId = ERR_AMOUNT_BADFORM;
    };

    return(bInvalid);
}; // GetAmountKey

//=====
//
// Function name: GetKeyValue
// This function parses an HTTP formatted string for specific key
// values. HTTP keys terminate with '='. HTTP values terminate
// with an '&' or '\0'.
//
// Result:
// FALSE - Key found, string value return in pValue
// TRUE - Key not found
//
//=====
BOOL GetKeyValue(CHAR * pHTML,CHAR * pKey,CHAR * pValue,UINT uMax)
{
    CHAR * ptr;
    if (!(ptr=strstr(pHTML,pKey)))
        return(TRUE);
    if (!(ptr=strchr(ptr,'=')))
        return(TRUE);
    ptr++;
    uMax--;
    while (*ptr && *ptr != '&' && uMax)
    {
        *pValue++ = *ptr++;
        uMax--;
    };
    *pValue = 0;
    return(FALSE);
}; // GetKeyValue

//=====
//
// Function name: FormatLogin
//
//=====
VOID FormatLogin(CHAR * pOut,CHAR * pAddText)
{
    sprintf(pOut,"%s<BR>%s<BR>%s",szFormLogin,pAddText,HTMLTrailer);
}; // FormatLogin

//=====
//

```

```

// Function name: FormatMenu
//
//=====
VOID FormatMenu(CHAR * pOut,TPCC_STATE * pTPCC)
{
    sprintf(pOut,
        "%s<HTML><HEAD><TITLE>TPC-C MainMenu</TITLE></HEAD><BODY>"
        "Select Desired Transaction.<BR><HR>"
        "<FORM ACTION=\"tpcc.dll\"METHOD=\"GET\">"
        "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">"
        "<INPUT TYPE=\"hidden\" NAME=\"TERMIN\" VALUE=\"%d\">"
        "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">"
        "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
        "%s</FORM><BR>%s<BR>%s",
        HTTPHdr,pTPCC->iStatusId,pTPCC->iTermId,pTPCC->iSyncId,FORM_MENU,
        szMenuList,pTPCC->ErrTxt,HTMLTrailer);
}; // FormatMenu

//=====
//
// Function name: FormatNewOrder
//
//=====
VOID FormatNewOrder(CHAR * pOut,TPCC_STATE * pTPCC)
{
    pTPCC->uFormId = FORM_NEWORDER;
    FormatFormHdr(pOut,"TPC-C New Order",pTPCC);
    sprintf(pOut + strlen(pOut),
        "<PRE>
        New Order<BR>"
        "Warehouse: %4.4d District: <INPUT NAME=\"DID*\" SIZE=1>"
        "Date:<BR>"
        "Customer: <INPUT NAME=\"CID*\" SIZE=4> Name:
        Credit: %Disc:<BR>"
        "Order Number: Number of Lines: W_tax:
        D_tax:<BR><BR>"
        " Supp_W Item_Id Item Name Qty Stock B/G Price
        Amount<BR>"
        "<INPUT NAME=\"SP00*\" SIZE=4> <INPUT NAME=\"IID00*\" SIZE=6>"
        "<INPUT NAME=\"Qty00*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP01*\" SIZE=4> <INPUT NAME=\"IID01*\" SIZE=6>"
        "<INPUT NAME=\"Qty01*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP02*\" SIZE=4> <INPUT NAME=\"IID02*\" SIZE=6>"
        "<INPUT NAME=\"Qty02*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP03*\" SIZE=4> <INPUT NAME=\"IID03*\" SIZE=6>"
        "<INPUT NAME=\"Qty03*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP04*\" SIZE=4> <INPUT NAME=\"IID04*\" SIZE=6>"
        "<INPUT NAME=\"Qty04*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP05*\" SIZE=4> <INPUT NAME=\"IID05*\" SIZE=6>"
        "<INPUT NAME=\"Qty05*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP06*\" SIZE=4> <INPUT NAME=\"IID06*\" SIZE=6>"
        "<INPUT NAME=\"Qty06*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP07*\" SIZE=4> <INPUT NAME=\"IID07*\" SIZE=6>"
        "<INPUT NAME=\"Qty07*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP08*\" SIZE=4> <INPUT NAME=\"IID08*\" SIZE=6>"
        "<INPUT NAME=\"Qty08*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP09*\" SIZE=4> <INPUT NAME=\"IID09*\" SIZE=6>"
        "<INPUT NAME=\"Qty09*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP10*\" SIZE=4> <INPUT NAME=\"IID10*\" SIZE=6>"
        "<INPUT NAME=\"Qty10*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP11*\" SIZE=4> <INPUT NAME=\"IID11*\" SIZE=6>"
        "<INPUT NAME=\"Qty11*\" SIZE=1><BR>"

```

```

        "<INPUT NAME=\"SP12*\" SIZE=4> <INPUT NAME=\"IID12*\" SIZE=6>"
        "<INPUT NAME=\"Qty12*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP13*\" SIZE=4> <INPUT NAME=\"IID13*\" SIZE=6>"
        "<INPUT NAME=\"Qty13*\" SIZE=1><BR>"
        "<INPUT NAME=\"SP14*\" SIZE=4> <INPUT NAME=\"IID14*\" SIZE=6>"
        "<INPUT NAME=\"Qty14*\" SIZE=1><BR>"
        "Execution Status:
        Total:<BR><HR>"
        "<INPUT TYPE=\"submit\"NAME=\"CMD\" VALUE=\"Process\">"
        "<INPUT TYPE=\"submit\"NAME=\"CMD\" VALUE=\"Menu\">"
        "</FORM>%s",
        pTPCC->sWId,HTMLTrailer);
}; // FormatNewOrder

//=====
//
// Function name: FormatPayment
//
//=====
VOID FormatPayment(CHAR * pOut,TPCC_STATE * pTPCC)
{
    pTPCC->uFormId = FORM_PAYMENT;
    FormatFormHdr(pOut,"TPC-C Payment",pTPCC);
    sprintf(pOut + strlen(pOut),
        "<PRE>
        Payment<BR>"
        "Date:<BR><BR>"
        "Warehouse: %4.4d
        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\">"
        "</FORM>%s",
        pTPCC->sWId,HTMLTrailer);
}; // FormatPayment

//=====
//
// Function name: FormatDelivery
//
//=====
VOID FormatDelivery(CHAR * pOut,TPCC_STATE * pTPCC)
{
    pTPCC->uFormId = FORM_DELIVERY;
    FormatFormHdr(pOut,"TPC-C Delivery",pTPCC);
    sprintf(pOut + strlen(pOut),
        "<PRE>
        Delivery<BR>"
        "Warehouse: %4.4d<BR><BR>"
        "Carrier Number: <INPUT NAME=\"OCD*\" SIZE=1><BR><BR>"
        "Execution Status:<BR></PRE><HR>"
        "<INPUT TYPE=\"submit\"NAME=\"CMD\" VALUE=\"Process\">"

```

```

        "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Menu\">"
        "</FORM>%s",
        pTPCC->sWId,HTMLTrailer);
}; // FormatDelivery

//=====
//
// Function name: FormatOrderStatus
//
//=====
VOID FormatOrderStatus (CHAR * pOut,TPCC_STATE * pTPCC)
{
    pTPCC->uFormId = FORM_ORDERSTATUS;
    FormatFormHdr (pOut,"TPC-C Order-Status",pTPCC);
    sprintf (pOut + strlen (pOut),
        "<PRE>
        Warehouse: %4.4d
        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\">"
        "</FORM>%s",
        pTPCC->sWId,HTMLTrailer);
}; // FormatOrderStatus

//=====
//
// Function name: FormatStockLevel
//
//=====
VOID FormatStockLevel (CHAR * pOut,TPCC_STATE * pTPCC)
{
    pTPCC->uFormId = FORM_STOCKLEVEL;
    FormatFormHdr (pOut,"TPC-C Stock Level",pTPCC);
    sprintf (pOut + strlen (pOut),
        "<PRE>
        Warehouse: %4.4d District: %2.2d<BR><BR>"
        "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\">"
        "</FORM>%s",
        pTPCC->sWId,pTPCC->sDId,HTMLTrailer);
}; // FormatStockLevel

//=====
//
// Function name: FormatFormHdr
//
//=====
VOID FormatFormHdr (CHAR * pOut,CHAR * pTitle,TPCC_STATE * pTPCC)
{
    sprintf (pOut,
        "%s<HTML><HEAD><TITLE>%s</TITLE></HEAD>"
        "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"

```

```

        "<INPUT TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">"
        "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
        "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
        "<INPUT TYPE=\"hidden\" NAME=\"TERMIN\" VALUE=\"%d\">"
        "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
        HTTPHdr,pTitle,pTPCC->uFormId,pTPCC->iTermId,pTPCC->iSyncId);
}; // FormatFormHdr

//=====
//
// Function name: FormatRespHdr
//
//=====
VOID FormatRespHdr (CHAR * pOut,CHAR * pTitle,TPCC_STATE * pTPCC)
{
    sprintf (pOut,
        "%s<HTML><HEAD><TITLE>%s</TITLE></HEAD>"
        "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
        "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">"
        "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
        "<INPUT TYPE=\"hidden\" NAME=\"TERMIN\" VALUE=\"%d\">"
        "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">",
        HTTPHdr,pTitle,pTPCC->iStatusId,pTPCC->uFormId,
        pTPCC->iTermId,pTPCC->iSyncId);
}; // FormatRespHdr

//=====
//
// Function name: FormatHTMLString
//
// Encodes HTML special characters. If necessary, space fills
// to pOut to total uLen characters.
//
//=====
VOID FormatHTMLString (CHAR * pOut,CHAR * pIn,UINT uLen)
{
    while (uLen && *pIn)
    {
        *pOut++ = *pIn++;
        uLen--;
    }; // while (uLen && *pIn)
    while (uLen-- > 0)
        *pOut++ = ' ';
}; // FormatHTMLString

//=====
//
// Function name: FormatString
//
// Encodes formatted string for HTML transmission.
//
//=====
VOID FormatString (CHAR * pOut,CHAR * pPic,CHAR * pIn)
{
    while (*pPic)
    {
        if (*pPic == 'X' )
        {
            if (*pIn)
                *pOut++ = *pIn++;

```

```

        else
            *pOut++ = ' ';
    }
    else
        *pOut++ = *pPic;
        pPic++;
    };
    *pOut = 0;
}; // FormatString

```

```

//=====
// FUNCTION: UtilStrCpy
//
// Copies n characters from string pSrc to pDst and places a null
// null character at the end of the destination string. Unlike
// strncpy this function ensures that the result string is always
// null terminated.
//
//=====
VOID UtilStrCpy(CHAR * pDest,CHAR * pSrc,INT n)
{
    strncpy(pDest,pSrc,n);
    pDest[n] = '\0';
    return;
}; // UtilStrCpy

```

```

//=====
//
// Function name: CheckNumeric
//
// Result
// FALSE - string is all numeric
// TRUE - sting contains non-numeric characters
//
//=====
BOOL CheckNumeric(CHAR * pNum)
{
    if (*pNum == 0 )
        return(TRUE);
    while (*pNum && isdigit(*pNum))
        pNum++;
    return(*pNum);
}; // CheckNumeric

```

term.h

```

// term.h

#include <sys\timeb.h>

#define TMILLI_TIMEOUT 3600000 // One hour

typedef struct
{
    BOOL bInUse; // In use flag
    INT iTermId; // TermId
    LPVOID ConnID; // Connection Id
    INT iSyncId; // Sync Id

```

```

    SHORT sWId; // TPC WareHouse Id
    SHORT sDId; // TPC District Id
    struct _timeb tbLastAccess; // Last activity timestamp
} TERM_STATE;

```

```

BOOL TermInit(INT iSetMaxTerm);
VOID TermTerm(VOID);
TERM_STATE * TermAlloc(VOID);
TERM_STATE * TermGet(INT iTermId);
BOOL TermFree(INT iTermId);

```

term.c

```

// term.c
//
// Copyright Unisys, 1997
//
#include <windows.h>
#include <stdio.h>
#include "diagio.h"
#include "timesupp.h"
#include "term.h"

TERM_STATE * pTArray;
INT iNextTerm = 0;
INT iMaxTerm = 0;
CRITICAL_SECTION csTerm;

VOID TermMaint(VOID);

//=====
//
// Function name: TermInit
// Creates and initializes the first TERMINITIAL TArray entries.
// Initializes critical section to control access to TArray. Assumes
// access to function is single threaded, no other threads will start
// until this function completes and that function is called once
// (DLL_PROCESS_ATTACH).
//
// Returns:
// FALSE TArray allocated and initialized
// TRUE TArray allocation failure
//
//=====
BOOL TermInit(INT iSetMaxTerm)
{
    INT iTermId;
    CHAR szDiag[MAX_DIAG_SZ];
    if (pTArray != NULL)
    {
        sprintf(szDiag,"TermInit(%ld): TArray Already Initialized\n",
            GetCurrentThreadId());
        DiagIoWrite(szDiag,DIAG_ERROR);
        return(TRUE);
    };
    InitializeCriticalSection(&csTerm);
    iMaxTerm = iSetMaxTerm;
    pTArray = (TERM_STATE *) malloc(sizeof(TERM_STATE) * (iMaxTerm + 1));
    if (pTArray == NULL)
    {

```



```

    sprintf(szDiag,"TermInit(%ld): malloc failed (%ld)\n",
        GetCurrentThreadId(),GetLastError());
    DiagIoWrite(szDiag,DIAG_ERROR);
    return(TRUE);
}
for (iTermId = 1; iTermId <= iMaxTerm; iTermId++)
    TermFree(iTermId);
iNextTerm = 1;
return(FALSE);
}; // TermInit

//=====
//
// Function name: TermTerm
// Frees TArray and deletes csTerm critical section. Assumes access
// to function is single threaded and no other threads are actively
// accessing TArray entries (DLL_PROCESS_DETACH).
//
//=====
VOID TermTerm(VOID)
{
    DeleteCriticalSection(&csTerm);
    if (pTArray != NULL)
        free(pTArray);
    iNextTerm = 0;
    iMaxTerm = 0;
}; // TermTerm

//=====
//
// Function name: TermAlloc
// Allocates empty TArray. Uses iNextTerm to start search.
//
// Returns:
// > 0 TArray entry index (iTermId)
// < 0 Empty TArray entry not available
//
//=====
TERM_STATE * TermAlloc(VOID)
{
    INT iTermId = -1;
    if (pTArray == NULL)
    {
        CHAR szDiag[MAX_DIAG_SZ];
        sprintf(szDiag,"TermAlloc(%ld): Term Array Not Allocated\n",
            GetCurrentThreadId());
        DiagIoWrite(szDiag,DIAG_ERROR);
        return(NULL);
    };
    EnterCriticalSection(&csTerm);
    _try
    {
        while(iNextTerm <= iMaxTerm)
        {
            if (!pTArray[iNextTerm].bInUse)
            {
                pTArray[iNextTerm].bInUse = TRUE;
                _ftime(&pTArray[iNextTerm].tbLastAccess);
                iTermId = iNextTerm;
                iNextTerm++;
                break;
            }
        }
    }
}

```

```

};
iNextTerm++;
}; // while(iNextTerm <= iMaxTerm) (1st Try)
if (iTermId <= 0)
{
    // No entry found. Perform maint and try again
    TermMaint();
    iNextTerm = 1;
    while(iNextTerm <= iMaxTerm)
    {
        if (!pTArray[iNextTerm].bInUse)
        {
            pTArray[iNextTerm].bInUse = TRUE;
            _ftime(&pTArray[iNextTerm].tbLastAccess);
            iTermId = iNextTerm;
            iNextTerm++;
            break;
        };
        iNextTerm++;
    }; // while(iNextTerm <= iMaxTerm) (2nd Try)
}; // if (iTermId <= 0)
if (iTermId <= 0)
    iNextTerm = 1;
}
finally
{
    LeaveCriticalSection(&csTerm);
};

if (iTermId > 0)
    return(&pTArray[iTermId]);
else
    return(NULL);
}; // TermAlloc

//=====
//
// Function name: TermMaint
// Clears entries whose last access time exceeds TMILLI_TIMEOUT.
// Assumes caller has entered csTerm.
//
//=====
VOID TermMaint(VOID)
{
    INT iTermId;
    TMILLI tmElapsed;
    // Free entries that have timed out
    for (iTermId = 1; iTermId <= iMaxTerm; iTermId++)
    {
        if (pTArray[iTermId].bInUse)
        {
            tmElapsed = TimebElapsed(&pTArray[iTermId].tbLastAccess);
            if (tmElapsed > TMILLI_TIMEOUT)
                TermFree(iTermId);
        };
    };
}; // TermMaint

//=====

```

```

//
// Function name: TermGet
// Returns pointer to TArray slot at iTermId.
//
// Returns:
// FALSE TArray entry made available
// TRUE iTermId invalid.
//
//=====
TERM_STATE * TermGet(INT iTermId)
{
    TERM_STATE * pTerm;
    TMILLI tmElapsed;
    if (iTermId <= 0 || iTermId > iMaxTerm)
    {
        CHAR szDiag[MAX_DIAG_SZ];
        sprintf(szDiag, "TermGet(%ld): Invalid TermId (%ld)\n",
            GetCurrentThreadId(), iTermId);
        DiagIoWrite(szDiag, DIAG_ERROR);
        return(NULL);
    };
    pTerm = &pTArray[iTermId];
    if (!pTerm->bInUse)
        return(NULL);
    tmElapsed = TimebElapsed(&pTerm->tbLastAccess);
    if (tmElapsed > TMILLI_TIMEOUT)
        return(NULL); // Entry destined to be freed by maint
    _ftime(&pTArray[iTermId].tbLastAccess);
    return(&pTArray[iTermId]);
}; // TermGet

//=====
//
// Function name: TermFree
// Initializes contents of TArray slot at iTermId.
//
// Returns:
// FALSE TArray entry made available
// TRUE iTermId invalid.
//
//=====
BOOL TermFree(INT iTermId)
{
    TERM_STATE * pTerm;
    if (iTermId <= 0 || iTermId > iMaxTerm)
    {
        CHAR szDiag[MAX_DIAG_SZ];
        sprintf(szDiag, "TermFree(%ld): Invalid TermId (%ld)\n",
            GetCurrentThreadId(), iTermId);
        DiagIoWrite(szDiag, DIAG_ERROR);
        return(TRUE);
    };
    pTerm = &pTArray[iTermId];
    pTerm->ConnID = 0;
    pTerm->SWID = 0;
    pTerm->SDID = 0;
    pTerm->iSyncId = 0;
    pTerm->iTermId = iTermId;
    TimebClear(&pTerm->tbLastAccess);
    pTerm->bInUse = FALSE;
}; // TermFree

```

tmon.h

```

// tmon.h
typedef struct
{
    CHAR * pszErrTxt; // Error text
    CHAR * pTMDData; // TM buffer area
    LONG lTMDDataLen; // TM buffer len
} TMON_STATE;

VOID TMonInit(INT iSetMaxMsg);
VOID TMonTerm(VOID);
BOOL TMinit(TMON_STATE * pTMon);
VOID TMDone(TMON_STATE * pTMon);
BOOL TMTran(CHAR * pService, TMON_STATE * pTMon,
            BOOL * bTPRsult, INT * iTPRsult);
BOOL TMPost(CHAR * pService, TMON_STATE * pTMon);

```

tmon.c

```

// tmon.c
//
// Copyright Unisys, 1997
//
#include <windows.h>
#include <stdio.h>
#include <atmi.h>
#include "tmon.h"

INT iTMMaxSz;

//=====
//
// Function name: TMonInit
//
//=====
VOID TMonInit(INT iSetMaxMsg)
{
    iTMMaxSz = iSetMaxMsg;
}; // TMonInit

//=====
//
// Function name: TMonTerm
//
//=====
VOID TMonTerm(VOID)
{
}; // TMonTerm

//=====
//
// Function name: TMinit
//
// Result:
// FALSE Initialization completed successfully
// TRUE Initialization failed
//

```

```

//=====
BOOL TMinit(TMON_STATE * pTMon)
{
    BOOL bRslt = FALSE;
    TPINIT * tpinfo;

    // Must have ErrTxt message area set before init
    if (pTMon->pszErrTxt == NULL)
        return(TRUE);
    tpinfo = (TPINIT *) tmalloc("TPINIT",NULL,TPINITNEED(20));
    memset(tpinfo,0,sizeof(TPINIT));
    tpinfo->flags=TPMULTICONTEXTS;
    sprintf(tpinfo->cltname,"tpcc%d",GetCurrentThreadId());

    if (tpinit(tpinfo) == -1)
    {
        sprintf(pTMon->pszErrTxt,"TPInit Failed(%ld)",tperrno);
        bRslt = TRUE;
    }
    else
    {
        pTMon->pTMDData = tmalloc("CARRAY",NULL,iTMMMaxSz);
        if (pTMon->pTMDData == NULL)
        {
            sprintf(pTMon->pszErrTxt,"TPAlloc Failed(%ld)",tperrno);
            bRslt = TRUE;
        };
    };

    return(bRslt);
}; // TMinit

//=====
//
// Function name: TMDone
//
//=====
VOID TMDone(TMON_STATE * pTMon)
{
    tpfree(pTMon->pTMDData);
    tpterm();
}; // TMDone

//=====
//
// Function name: TMTran
//
// Result:
// FALSE    call completed. bTPRslt contains outcome (FALSE tran
//           success). iTPRslt contains application returned
//           result code.
// TRUE     TM interface error, ErrTxt has diagnostic.
//
//=====
BOOL TMTran(CHAR * pService,TMON_STATE * pTMon,
            BOOL * bTPRslt,INT * iTPRslt)
{
    BOOL bRslt = FALSE;
    INT iGRply;

```

```

    iGRply = tpcall(pService,pTMon->pTMDData,iTMMMaxSz,
        &pTMon->pTMDData,&pTMon->lTMDDataLen,TPNOTIME | TPSIGRSTRT);
    if (iGRply != -1)
    {
        *iTPRslt = tpurcode;
        *bTPRslt = FALSE;
    }
    else
    if (tperrno == TPESVCFAIL)
    {
        *iTPRslt = tpurcode;
        *bTPRslt = TRUE;
    }
    else
    {
        sprintf(pTMon->pszErrTxt,"TPCall Failed (%ld)",tperrno);
        bRslt = TRUE;
    };
    return(bRslt);
}; // TMTran

//=====
//
// Function name: TMPost
//
// Result:
// FALSE    transaction submitted with no response expected
// TRUE     tpacall failed, ErrTxt has diagnostic
//
//=====
BOOL TMPost(CHAR * pService,TMON_STATE * pTMon)
{
    BOOL bRslt = FALSE;
    INT iCD;

    iCD = tpacall(pService,pTMon->pTMDData,iTMMMaxSz,TPNOREPLY);
    if (iCD == -1)
    {
        sprintf(pTMon->pszErrTxt,"TPACall Failed (%ld)",tperrno);
        bRslt = TRUE;
    };
    return(bRslt);
}; // TMPost

```

timesupp.h

```

// timesupp.h
#include <windows.h>
#include <time.h>
#include <sys\timeb.h>

#define TIMEBSEED_MOD 10000
#define TIMEBSEED_SHIFT 1000
#define TIMEB_STRING_SZ 23
#define TIMEB_STRING_DATESZ 10
#define TIMEB_STRING_TIMEOFFSET 11
#define TIMEB_STRING_TIMESZ 12

typedef ULONG TMILLI;

```

```

TMILLI TimebDiff(struct _timeb * p_tb1, struct _timeb * p_tb2);
VOID TimebCopy(struct _timeb * p_tbDest, struct _timeb * p_tbSource);
TMILLI TimebElapsed(struct _timeb * p_tb1);
VOID TimebClear(struct _timeb * p_tb1);
CHAR * TimebToString(struct _timeb * p_tb1, CHAR * psz, BOOL bMillis);
BOOL TimebFromString(struct _timeb * p_tb1, CHAR * psz);
VOID TimebAddSecs(struct _timeb * p_tb1, INT iSeconds);
ULONG TimebSeed(VOID);

```

timesupp.c

```

// timesupp.c
//
// Copyright Unisys, 1997
//

#include <stdio.h>
#include "timesupp.h"

//=====
//
// Function name: TimebCopy
// Structure contents copy of _timeb source to _timeb dest.
//
//=====
VOID TimebCopy(struct _timeb * p_tbDest, struct _timeb * p_tbSource)
{
    p_tbDest->time = p_tbSource->time;
    p_tbDest->millitm = p_tbSource->millitm;
    p_tbDest->dstflag = p_tbSource->dstflag;
    p_tbDest->timezone = p_tbSource->timezone;
}; // TimebCopy

//=====
//
// Function name: TimebDiff
// Time difference in milliseconds between _timeb_t1 and _timeb_t2.
//
//=====
TMILLI TimebDiff(struct _timeb * p_tb1, struct _timeb * p_tb2)
{
    LONG lRslt;
    lRslt = ((p_tb2->time - p_tb1->time) * 1000) +
            (p_tb2->millitm - p_tb1->millitm);
    if (lRslt < 0)
        return(0);
    else
        return((TMILLI) lRslt);
}; // TimebDiff

//=====
//
// Function name: TimebElapsed
//
//=====
TMILLI TimebElapsed(struct _timeb * p_tb1)
{

```

```

    struct _timeb _tb2;
    _ftime(&_tb2);
    return (TimebDiff(p_tb1, &_tb2));
}; // TimebElapsed

//=====
//
// Function name: TimebClear
//
//=====
VOID TimebClear(struct _timeb * p_tb1)
{
    p_tb1->time = 0;
    p_tb1->millitm = 0;
}; // TimebClear

//=====
//
// Function name: TimebToString
// Converts timeb to yyyy:mm:dd,hh:mm:ss.sss format
//
//=====
CHAR * TimebToString(struct _timeb * p_tb1, CHAR * psz, BOOL bMillis)
{
    struct tm * ptm;
    ptm = localtime(&p_tb1->time);
    sprintf(psz, "%4.4d/%2.2d/%2.2d,%2.2d:%2.2d:%2.2d",
            ptm->tm_year + 1900, ptm->tm_mon + 1, ptm->tm_mday,
            ptm->tm_hour, ptm->tm_min, ptm->tm_sec);
    if (bMillis)
        sprintf(psz + strlen(psz), ".%3.3d", p_tb1->millitm);
    return(psz);
}; // TimebToString

//=====
//
// Function name: TimebFromString
// Converts yyyy:mm:dd,hh:mm:ss.sss (TimebToString) format to timeb
//
//=====
BOOL TimebFromString(struct _timeb * p_tb1, CHAR * psz)
{
    struct tm tmTime;
    struct tm * ptm;
    UINT uLen;

    ptm = &tmTime;
    uLen = strlen(psz);
    if (uLen < (TIMEB_STRING_SZ - 4)) // millis are optional
    {
        p_tb1->time = 0;
        p_tb1->millitm = 0;
        return (TRUE);
    };
    // Clear fields that won't be set
    ptm->tm_wday = 0;
    ptm->tm_yday = 0;
    ptm->tm_isdst = -1;
    // Set tm struct fields from string

```

```

ptm->tm_year = (atoi(psz)) - 1900;
psz += 5;
ptm->tm_mon = (atoi(psz)) - 1;
psz += 3;
ptm->tm_mday = atoi(psz);
psz += 3;
ptm->tm_hour = atoi(psz);
psz += 3;
ptm->tm_min = atoi(psz);
psz +=3;
ptm->tm_sec = atoi(psz);
if (uLen >= TIMEB_STRING_SZ) // Millis present
{
    psz += 3;
    p_tbl->millitm = atoi(psz);
};
p_tbl->time = mktime(ptm);
return (FALSE);
}; // TimebFromString

//=====
//
// Function name: TimebAddSecs
//
//=====
VOID TimebAddSecs(struct _timeb * p_tbl,INT iSeconds)
{
    p_tbl->time += iSeconds;
}; // TimebAddSecs

```

diagio.h

```

// diagio.h

// Environment variable defaults
#define DEFAULTDIAGLEVEL DIAG_INFO
#define DEFAULTTEVENTLOG 0

#define DIAGNOSTICS TRUE
#define MAX_DIAG_SZ 2000

// Severity level of diagnostic report
#define DIAG_FORCE 1
#define DIAG_ERROR 2
#define DIAG_STATE 3
#define DIAG_INFO 4

VOID DiagIoInit (CHAR * pDiagId,BOOL bConsole,BOOL bEvent,UINT uLevel);
VOID DiagIoTerm (VOID);
VOID DiagIoWrite (CHAR * pDiagBuffer, UINT uSeverity);

```

diagio.c

```

// diagio.c
//
// Copyright Unisys, 1997
//
#include <windows.h>
#include <stdio.h>

```

```

#include "diagio.h"

CRITICAL_SECTION csDiagIo;
HANDLE hEventLog = NULL;
UINT uDiagLevel;
BOOL bEventLog;
BOOL bConsoleLog;
CHAR * pDiagHdr;
CHAR * pEventHost;
CHAR * pErrHdr =
    {"*** ERROR *** ERROR *** ERROR *** ERROR *** ERROR ***"};

INT WriteEventLog (CHAR * pDMsgs[],UINT uMsgCnt,UINT uSeverity);

//=====
//
// Function name: DiagIoInit
//
//=====
VOID DiagIoInit (CHAR * pDiagId,BOOL bConsole,BOOL bEvent,UINT uLevel)
{
    if (DIAGNOSTICS)
    {
        InitializeCriticalSection (&csDiagIo);

        uDiagLevel = uLevel;
        bEventLog = bEvent;
        bConsoleLog = bConsole;
        pEventHost = (CHAR *) malloc(10);
        strcpy(pEventHost,""); // local host
        pDiagHdr = (CHAR *) malloc(strlen(pDiagId) + 1);
        strcpy(pDiagHdr,pDiagId);
        if (bEventLog)
        {
            hEventLog = RegisterEventSource (pEventHost,pDiagId);
            if (hEventLog == NULL)
            {
                bEventLog = FALSE;
                if (bConsoleLog)
                    fprintf (stdout,
                        "%s: Event Log Register Failed (%ld)\n"
                        "Event Log Will NOT be Used\n",
                        pDiagHdr,GetLastError());
            }
            else
            {
                if (bConsoleLog)
                    fprintf (stdout,"%s: Event Logging to LocalHost as %s\n",
                        pDiagHdr,pDiagHdr);
            }
        }; // if bEventLog
    }; // if Diagnostics
}; // DiagIoInit

//=====
//
// Function name: DiagIoTerm
//
//=====
VOID DiagIoTerm (VOID)

```

```

{
    if (DIAGNOSTICS)
    {
        DeleteCriticalSection(&csDiagIo);
        if (hEventLog != NULL)
            DeregisterEventSource(hEventLog);
        free(pDiagHdr);
        free(pEventHost);
    };
}; // DiagIoTerm

//=====
//
// Function name: DiagIoWrite
//
//=====
VOID DiagIoWrite(CHAR * pDiagBuffer, UINT uSeverity)
{
    CHAR * pDMsgs[3];
    UINT uMsgCnt = 0;
    INT iERslt = 0;
    if (DIAGNOSTICS)
    {
        if (uDiagLevel >= uSeverity)
        {
            EnterCriticalSection(&csDiagIo);
            try
            {
                if (uSeverity == DIAG_ERROR)
                {
                    pDMsgs[0] = pDiagHdr;
                    pDMsgs[1] = pErrHdr;
                    pDMsgs[2] = pDiagBuffer;
                    uMsgCnt = 3;
                }
                else
                {
                    pDMsgs[0] = pDiagHdr;
                    pDMsgs[1] = pDiagBuffer;
                    uMsgCnt = 2;
                };
                if (bEventLog)
                    iERslt = WriteEventLog(pDMsgs, uMsgCnt, uSeverity);
                if (bConsoleLog)
                {
                    if (uMsgCnt == 3)
                        fprintf(stdout, "\n%s:
%s\n%s", pDMsgs[0], pDMsgs[1], pDMsgs[2]);
                    else
                        fprintf(stdout, "\n%s: %s", pDMsgs[0], pDMsgs[1]);
                    if (iERslt != 0)
                        fprintf(stdout,
                            "EventLog Write Failed (%ld), No Longer in Use\n",
                            iERslt);
                };
            }
            finally
            {
                LeaveCriticalSection(&csDiagIo);
            };
        }; // if uDiagLevel >= uSeverity
    }
};

```

```

}; // if Diagnostics
}; // DiagIoWrite

INT WriteEventLog(CHAR * pDMsgs[], UINT uMsgCnt, UINT uSeverity)
{
    WORD wType;
    WORD wCount;
    wCount = uMsgCnt;
    switch (uSeverity)
    {
        case DIAG_ERROR:
            wType = EVENTLOG_ERROR_TYPE;
            break;
        default:
            wType = EVENTLOG_INFORMATION_TYPE;
            break;
    };
    if (wType != 0)
    {
        if (!ReportEvent(hEventLog, // event log handle
            wType, // event type
            0, // category zero
            uSeverity, // no event identifier
            NULL, // no user security identifier
            wCount, // # of substitution strings
            0, // no binary data
            (LPCTSTR *) pDMsgs, // address of string array
            NULL)) // address of binary
        {
            DeregisterEventSource(hEventLog);
            hEventLog = NULL;
            bEventLog = FALSE;
            return(GetLastError());
        }; // ReportEvent failed
    }; // if wType != 0
    return(0);
}; // WriteEventLog

```

SERVER MAKEFILES

```

SVR = tpccsvr
SRC = \webrte\tpcctux\tpccsvr.c
DBG = /f "/Zi"
$(SVR).exe: $(SRC)
    erase $(SVR).exe
    $(TUXDIR)\bin\buildserver /f "$(SRC)" /o $(SVR).exe /s
NEWORDER:NEWORDER /s PAYMENT:PAYMENT /s ORDERSTS:ORDERSTS /s
STOCKLVL:STOCKLVL -l i:\mssql7\devtools\lib\ntwdblib.lib
    copy $(SVR).exe $(APPDIR)

```

```

SVR = tpccdelv
SRC = \webrte\tpcctux\tpccdelv.c
DBG = /f "/Zi"
$(SVR).exe: $(SRC)
    erase $(SVR).exe
    $(TUXDIR)\bin\buildserver /f "$(SRC)" /o $(SVR).exe /s
DELIVERY:DELIVERY -l i:\mssql7\devtools\lib\ntwdblib.lib

```

```
copy $(SVR).exe $(APPDIR)
```

tpccsvr.h

```
// tpccsvr.h
//
// Copyright Unisys, 1997
// Copyright Microsoft, 1996
```

```
#include "tpcc.h"
```

```
#define DEFCLPACKSIZE 2000
#define DEADLOCKWAIT 10
#define LOGFILE_NAME "delilog"
```

```
// 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
```

tpcc.h

```
// tpcc.h
```

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

```
// TPCCHandler return codes
```

```
#define TPCCSEND 1
#define TPCCSENDEND 2
#define TPCCENDNOW 3
```

```
// TPC Service return codes
```

```
#define SVC_BADITEMID 1
#define SVC_NOERROR 0
#define SVCERR_DEADLOCK -1
#define SVCERR_NOCUSTOMER -2
#define SVCERR_NOORDERS -3
#define SVCERR_DBLIB -4
```

```
// Min/Max transaction data definitions
```

```
#define MIN_DID 1
#define MAX_DID 10
#define MIN_OL 5
#define MAX_OL 15
#define MIN_QUANTITY 1
#define MAX_QUANTITY 10
#define MIN_ITEM_ID 1
#define MAX_ITEM_ID 100000
#define MIN_CUST_ID 1
#define MAX_CUST_ID 3000
#define MIN_CARRIER 1
#define MAX_CARRIER 10
#define MIN_THRESHOLD 10
```

```
#define MAX_THRESHOLD 20
```

```
// pTPCC->iStatusId codes
```

```
#define INVALID_IID 1
#define STATUS_OK 0
#define ERR_CMD_UNKNOWN -10
#define ERRTEXT_CMD_UNKNOWN "Unrecognized Command"
#define ERR_ALREADY_LOGGEDIN -11
#define ERRTEXT_ALREADY_LOGGEDIN "Already Logged In"
#define ERR_TERMID -12
#define ERRTEXT_TERMID "TermId or SyncId in Error"
#define ERR_FORM_UNKNOWN -13
#define ERRTEXT_FORM_UNKNOWN "Unrecognized FormId"
#define ERR_WID_INVALID -14
#define ERR_DID_INVALID -15
#define ERR_MISSING_KEY -16
#define ERR_NOT_NUMERIC -17
#define ERR_THRESHOLD_RANGE -18
#define ERR_EMBEDDED_EMPTY_OL -19
#define ERR_QUANTITY_INVALID -20
#define ERR_OL_INVALID -21
#define ERR_OL_COUNT -22
#define ERR_TM_INTERFACE -23
#define ERR_SERVICE_RSLT -24
#define ERR_INPUT_TOOLONG -25
#define ERR_IDANDNAME_EMPTY -26
#define ERR_IDANDNAME_ENTERED -27
#define ERR_AMOUNT_BADFORM -28
#define ERR_AMOUNT_INVALID -29
#define ERR_CARRIER_INVALID -30
#define ERR_TERM_ALLOC -31
```

```
#define STATUS_LEN 200
#define NAME_LEN 16
#define ADDR_LEN 20
#define STATE_LEN 2
#define ZIP_LEN 9
```

```
#define MAX_MSG_SZ 5000
```

```
typedef struct
```

```
{
    short ol_supply_w_id;
    long ol_i_id;
    char ol_i_name[25];
    short ol_quantity;
    char ol_brand_generic[2];
    double ol_i_price;
    double ol_amount;
    short ol_stock;
} OL_NEW_ORDER_DATA;
```

```
typedef struct
```

```
{
    short w_id;
    short d_id;
    long c_id;
    short o_ol_cnt;
    char c_last[NAME_LEN + 1];
    char c_credit[3];
    double c_discount;
```

```

double w_tax;
double d_tax;
long o_id;
short o_commit_flag;
DBDATEREC o_entry_d;
short o_all_local;
double total_amount;
char execution_status[STATUS_LEN];
OL_NEW_ORDER_DATA Ol[MAX_OL];
} 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;
    DBDATEREC h_date;
    char w_street_1[ADDR_LEN + 1];
    char w_street_2[ADDR_LEN + 1];
    char w_city[ADDR_LEN + 1];
    char w_state[STATE_LEN + 1];
    char w_zip[ZIP_LEN + 1];
    char d_street_1[ADDR_LEN + 1];
    char d_street_2[ADDR_LEN + 1];
    char d_city[ADDR_LEN + 1];
    char d_state[STATE_LEN + 1];
    char d_zip[ZIP_LEN + 1];
    char c_first[NAME_LEN + 1];
    char c_middle[3];
    char c_last[NAME_LEN + 1];
    char c_street_1[ADDR_LEN + 1];
    char c_street_2[ADDR_LEN + 1];
    char c_city[ADDR_LEN + 1];
    char c_state[STATE_LEN + 1];
    char c_zip[ZIP_LEN + 1];
    char c_phone[16];
    DBDATEREC c_since;
    char c_credit[3];
    double c_credit_lim;
    double c_discount;
    double c_balance;
    char c_data[200+1];
    char execution_status[STATUS_LEN];
} PAYMENT_DATA;

```

```
typedef struct
```

```

{
    long ol_i_id;
    short ol_supply_w_id;
    short ol_quantity;
    double ol_amount;
    DBDATEREC ol_delivery_d;
} OL_ORDER_STATUS_DATA;

```

```
typedef struct
```

```

{
    short w_id;
    short d_id;

```

```

long c_id;
char c_first[NAME_LEN + 1];
char c_middle[3];
char c_last[NAME_LEN + 1];
double c_balance;
long o_id;
DBDATEREC o_entry_d;
short o_carrier_id;
OL_ORDER_STATUS_DATA OlOrderStatusData[MAX_OL];
short o_ol_cnt;
char execution_status[STATUS_LEN];
} ORDER_STATUS_DATA;

```

```
typedef struct
```

```

{
    short w_id;
    short o_carrier_id;
    long o_id[10];
    int iComplete;
    SYSTEMTIME QTime; // time delivery was queued
    SYSTEMTIME EndTime; // time delivery completed
    char execution_status[STATUS_LEN];
} DELIVERY_DATA;

```

```
typedef struct
```

```

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

```

tpccsvr.c

```
// tpccsvr.c
```

```
//
```

```
// Copyright Unisys, 1997
```

```
// Copyright Microsoft, 1996
```

```

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

```

```

#include <atmi.h>
#include <userlog.h>

#include "tpccsvr.h"

```

```

char    szServer[32]    = "tpccserver";
char    szUser[32]      = { 0 };
char    szPassword[32] = { 0 };
char    szDatabase[32] = "tpcc";
char    szService[16]  = "tpccsvr";
char    szWork[200];

```



```

PDBPROCESS    dbproc;
int    spid;                // spid assigned from dblib
BOOL   bFailed;
BOOL   bDeadlock;
short  DeadlockRetry = (short)3;

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);
int SQLStockLevel(STOCK_LEVEL_DATA *psld);
int SQLNewOrder(NEW_ORDER_DATA * pnod);
int SQLPayment(PAYMENT_DATA *ppd);
int SQLOrderStatus(ORDER_STATUS_DATA * pOrderStatus);
void UtilStrCpy(char * pDest, char * pSrc, int n);
VOID GetArgs(INT argc, CHAR **argv);

//=====
//
// Function name: tpsvrinit
//
//=====
tpsvrinit(int argc, char *argv[])
{
    GetArgs(argc, argv);
    sprintf(szWork, "%s Started, DBServer=%s, DB=%s",
            szService, szServer, szDatabase);
    userlog(szWork);
    if (SQLInit(szServer, szDatabase, szUser, szPassword))
        return(-1);
    userlog("Database open, initialization complete");
    return(0);
}; // tpsvrinit

//=====
//
// Function name: tpsvrdone
//
//=====
void tpsvrdone()
{
    userlog("Shutdown request for tpcctux server");
    dbclose(dbproc);
    dbexit();
}; // tpsvrdone

//=====
//
// Function name: NEWORDER
//
// Entry point called by tuxedo for NEWORDER service requests.
//
//=====
void NEWORDER(TPSVCINFO * svcinfo)
{
    int iRslt;
    NEW_ORDER_DATA * pnod;

    pnod = (NEW_ORDER_DATA *) svcinfo->data;
    iRslt = SQLNewOrder(pnod);

```

```

// Check for DBLib termination error
if (bFailed)
{
    strcpy(pnod->execution_status, szWork);
    tpreturn(TPFAIL, SVCERR_DBLIB, svcinfo->data, svcinfo->len, 0);
}
else
if (iRslt == 0)
    tpreturn(TPSUCCESS, 0, svcinfo->data, svcinfo->len, 0);
else
    tpreturn(TPFAIL, iRslt, svcinfo->data, svcinfo->len, 0);
}; // NEWORDER

//=====
//
// Function name: PAYMENT
//
// Entry point called by tuxedo for PAYMENT service requests.
//
//=====
void PAYMENT(TPSVCINFO * svcinfo)
{
    int iRslt;
    PAYMENT_DATA * ppd;

    ppd = (PAYMENT_DATA *) svcinfo->data;

    iRslt = SQLPayment(ppd);

    if (bFailed)
    {
        strcpy(ppd->execution_status, szWork);
        tpreturn(TPFAIL, SVCERR_DBLIB, svcinfo->data, svcinfo->len, 0);
    }
    else
    if (iRslt == 0)
        tpreturn(TPSUCCESS, 0, svcinfo->data, svcinfo->len, 0);
    else
        tpreturn(TPFAIL, iRslt, svcinfo->data, svcinfo->len, 0);
}; // PAYMENT

//=====
//
// Function name: ORDERSTS
//
// Entry point called by tuxedo for ORDERSTS service requests.
//
//=====
void ORDERSTS(TPSVCINFO * svcinfo)
{
    int iRslt;
    ORDER_STATUS_DATA * posd;

    posd = (ORDER_STATUS_DATA *) svcinfo->data;
    iRslt = SQLOrderStatus(posd);

    // Check for DBLib termination error
    if (bFailed)
    {
        strcpy(posd->execution_status, szWork);
        tpreturn(TPFAIL, SVCERR_DBLIB, svcinfo->data, svcinfo->len, 0);
    }

```

```

}
else
if (iRslt == 0)
    tpreturn(TPSUCCESS,0,svcinfn->data,svcinfn->len,0);
else
    tpreturn(TPFAIL,iRslt,svcinfn->data,svcinfn->len,0);
}; // ORDERSTS

//=====
//
// Function name: STOCKLVL
//
// Entry point called by tuxedo for STOCKLVL service requests.
//
//=====
void STOCKLVL(TPSVCINFO * svcinfn)
{
    int iRslt;
    STOCK_LEVEL_DATA * psld;

    psld = (STOCK_LEVEL_DATA *) svcinfn->data;
    iRslt = SQLStockLevel(psld);

    // Check for DBLib termination error
    if (bFailed)
    {
        strcpy(psld->execution_status,szWork);
        tpreturn(TPFAIL,SVCERR_DBLIB,svcinfn->data,svcinfn->len,0);
    }
    else
    if (iRslt == 0)
        tpreturn(TPSUCCESS,0,svcinfn->data,svcinfn->len,0);
    else
        tpreturn(TPFAIL,iRslt,svcinfn->data,svcinfn->len,0);
}; // STOCKLVL

//=====
//
// Function name: SQLInit
//
// Set global dbproc and spid.
//
// Result:
// FALSE - database open, dbproc valid
// TRUE - database open failed
//
//=====
BOOL SQLInit(CHAR * pSvr,CHAR * pDB,CHAR * pUsr,CHAR * pPW,CHAR * pSvc)
{
    char szApp[32];
    char server[256];
    char database[256];
    char user[256];
    char password[256];
    LOGINREC *login;

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

```

```

dbproc = NULL;
strcpy(server,pSvr);
strcpy(database,pDB);
strcpy(user,pUsr);
strcpy(password,pPW);
sprintf(szApp,"%s%d",pSvc,_getpid());

login = dblogin();
if (!*user)
    DBSETLUSER(login,"sa");
else
    DBSETLUSER(login,user);
DBSETLPWD(login,password);
DBSETLHOST(login,szApp);
DBSETLVERSION(login,DBVER60);
// DBSETLPACKET(login,(unsigned short)DEFCLPACKSIZE);

if ((dbproc = dbopen(login,server)) == NULL)
{
    userlog("dbopen failed");
    return TRUE;
};
// Use the the right database
dbuse(dbproc,database);
dbcmd(dbproc,"select @@spid");
dbsqlxexec(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");
dbsqlxexec(dbproc);
while (dbresults(dbproc) != NO_MORE_RESULTS)
{
    while (dbnextrow(dbproc) != NO_MORE_ROWS)
        ;
};

//rollback transaction on abort
dbcmd(dbproc,"set XACT_ABORT ON");
dbsqlxexec(dbproc);
while (dbresults(dbproc) != NO_MORE_RESULTS)
{
    while (dbnextrow(dbproc) != NO_MORE_ROWS)
        ;
};

return(FALSE);
}; // SQLInit

//=====
// FUNCTION: err_handler
//
// 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_CANCEL
//
// COMMENTS: None
//
//=====
int err_handler(DBPROCESS *dbproc, int severity, int dberr, int oserr,
char *dberrstr, char *oserrstr)
{
    if ((dbproc == NULL) || (DBDEAD(dbproc)))
    {
        userlog("ErrHandler: DBPROC is invalid");
        return INT_CANCEL;
    };
    if (bFailed)
        return INT_CANCEL;
    if (oserr != DBNOERR)
    {
        sprintf(szWork, "ErrHandler: OSErr(%ld) - %s", oserr, oserrstr);
        userlog(szWork);
        bFailed = TRUE;
    };

    return INT_CANCEL;
}; // err_handler

//=====
// FUNCTION: msg_handler
//
// 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 operation
// 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)
{
    if ((msgno == 5701) || (msgno == 2528) ||
        (msgno == 5703) || (msgno == 6006))
        return INT_CONTINUE;

```

```

// deadlock message
if (msgno == 1205)
{
    // set the deadlock indicator
    bDeadlock = TRUE;
    return INT_CONTINUE;
};

if (bFailed)
    return INT_CANCEL;

if (msgno == 0)
    return INT_CONTINUE;
else
{
    sprintf(szWork, "MsgHandler: MsgNo(%ld) - %s", msgno, msgtext);
    userlog(szWork);
    bFailed = TRUE;
};

return INT_CANCEL;
}; // msg_handler

//=====
// FUNCTION: SQLStockLevel
//
// Handles the stock level transaction.
//
// ARGUMENTS:
// STOCK_LEVEL_DATA StockLevel input / output data structure
// dbdata (global)
// bDeadlock (global)
//
// RETURNS:
// SVC_NOERROR success
// !SVC_NOERROR failure
//
// COMMENTS: None
//
//=====
int SQLStockLevel(STOCK_LEVEL_DATA * psld)
{
    int tryit;
    short num_deadlocks = 0;
    RETCODE rc;
    BYTE * pData;

    bFailed = FALSE;
    bDeadlock = FALSE;

    for (tryit=0; tryit < DeadlockRetry; tryit++)
    {
        if (dbrpcinit(dbproc, "tpcc_stocklevel", 0) == SUCCEEDED)
        {
            dbrpcparam(dbproc, NULL, 0, SQLINT2, -1, -1,
                (BYTE *) &psld->w_id);
            dbrpcparam(dbproc, NULL, 0, SQLINT1, -1, -1,
                (BYTE *) &psld->d_id);
            dbrpcparam(dbproc, NULL, 0, SQLINT2, -1, -1,
                (BYTE *) &psld->thresh_hold);

```

```

if (dbrpcexec(dbproc) == SUCCEEDED)
{
    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))
                    psld->low_stock = *((long *) pData);
            };
        }; // if (DBROWS(dbproc))
    }; // while (dbresults)
}; // if (dbrpcexec)
}; // if (dbrpcinit)
if (bDeadlock)
{
    num_deadlocks++;
    bDeadlock = FALSE;
    userlog("StockLevel Deadlock Retry (%d)",num_deadlocks);
    Sleep(10 * tryit);
}
else
{
    strcpy(psld->execution_status,"Transaction committed.");
    return(SVC_NOERROR);
};
}; // for (tryit)

// If we reached here, it means we quit after MAX_RETRY deadlocks
strcpy(psld->execution_status,"Hit deadlock max.");
userlog("StockLevel Deadlock Failure (%d)",num_deadlocks);
return(SVCERR_DEADLOCK);
}; // SQLStockLevel

//=====
// FUNCTION: SQLNewOrder
//
// Handles the new order transaction.
//
// ARGUMENTS:
// NEW_ORDER_DATA    NewOrder structure for input/output data
// dbdata (global)
// bDeadlock (global)
//
// RETURNS:
// SVC_NOERROR success
// !SVC_NOERROR failure
//
// COMMENTS: None
//
//=====
int SQLNewOrder(NEW_ORDER_DATA * pnod)
{
    RETCODE rc;
    int i;
    DBINT commit_flag;

```

```

short num_deadlocks = 0;
int tryit;
DBDATETIME datetime;
BYTE * pData;

bFailed = FALSE;
bDeadlock = FALSE;

for (tryit=0; tryit < DeadlockRetry; tryit++)
{
    if (dbrpcinit(dbproc,"tpcc_neworder",0) == SUCCEEDED)
    {
        dbrpcparam(dbproc, NULL, 0, SQLINT2, -1, -1,
                   (BYTE *) &pnod->w_id);
        dbrpcparam(dbproc, NULL, 0, SQLINT1, -1, -1,
                   (BYTE *) &pnod->d_id);
        dbrpcparam(dbproc, NULL, 0, SQLINT4, -1, -1,
                   (BYTE *) &pnod->c_id);
        dbrpcparam(dbproc, NULL, 0, SQLINT1, -1, -1,
                   (BYTE *) &pnod->o_ol_cnt);

        pnod->o_all_local = 1;
        for (i = 0; i < pnod->o_ol_cnt; i++)
        {
            if (pnod->o_all_local &&
                pnod->Ol[i].ol_supply_w_id != pnod->w_id )
                pnod->o_all_local = 0;
        };
        dbrpcparam(dbproc, NULL, 0, SQLINT1, -1, -1,
                   (BYTE *) &pnod->o_all_local);

        for (i = 0; i < pnod->o_ol_cnt; i++)
        {
            dbrpcparam(dbproc, NULL, 0, SQLINT4, -1, -1,
                       (BYTE *) &pnod->Ol[i].ol_i_id);
            dbrpcparam(dbproc, NULL, 0, SQLINT2, -1, -1,
                       (BYTE *) &pnod->Ol[i].ol_supply_w_id);
            dbrpcparam(dbproc, NULL, 0, SQLINT2, -1, -1,
                       (BYTE *) &pnod->Ol[i].ol_quantity);
        };

        if (dbrpcexec(dbproc) == SUCCEEDED)
        {
            pnod->total_amount=0;
            // Get results from order line
            for (i = 0; i<pnod->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(pnod->Ol[i].ol_i_name,pData,dbdatlen(dbproc, 1));
                            if(pData=dbdata(dbproc, 2))
                                pnod->Ol[i].ol_stock = *(DBSMALLINT *) pData);
                            if(pData=dbdata(dbproc, 3))

```

```

        UtilStrCpy(pnod-
>Ol[i].ol_brand_generic,pData,dbdatlen(dbproc, 3));
        if(pData=dbdata(dbproc, 4))
dbconvert (dbproc,SQLNUMERIC,pData,sizeof(DBNUMERIC),
        SQLFLTNT,(CHAR *) &pnod->Ol[i].ol_i_price,8);
        if(pData=dbdata(dbproc, 5))
dbconvert (dbproc,SQLNUMERIC,pData,sizeof(DBNUMERIC),
        SQLFLTNT,(CHAR *) &pnod->Ol[i].ol_amount,8);
        pnod->total_amount = pnod->total_amount + pnod-
>Ol[i].ol_amount;
    }; // while (dbnextrow)
}; // if (DBROWS && dbnumcols)
}; // if (dbresults)
}; // for (o_ol_cnt)
while ((rc = dbresults(dbproc)) != NO_MORE_RESULTS) &&
    (rc != FAIL)
{
    if (DBROWS(dbproc) && (dbnumcols(dbproc) == 8))
    {
        while ((rc = dbnextrow(dbproc)) != NO_MORE_ROWS) &&
            (rc != FAIL)
        {
            if(pData=dbdata(dbproc, 1))
            dbconvert (dbproc,SQLNUMERIC,pData,sizeof(DBNUMERIC),
                SQLFLTNT,(CHAR *) &pnod->w_tax,8);
            if(pData=dbdata(dbproc, 2))
            dbconvert (dbproc,SQLNUMERIC,pData,sizeof(DBNUMERIC),
                SQLFLTNT,(CHAR *) &pnod->d_tax,8);
            if(pData=dbdata(dbproc, 3))
                pnod->o_id = (*(DBINT *) pData);
            if(pData=dbdata(dbproc, 4))
                UtilStrCpy(pnod->c_last,pData,dbdatlen(dbproc,4));
            if(pData=dbdata(dbproc, 5))
            dbconvert (dbproc,SQLNUMERIC,pData,sizeof(DBNUMERIC),
                SQLFLTNT,(CHAR *) &pnod->c_discount,8);
            if(pData=dbdata(dbproc, 6))
                UtilStrCpy(pnod-
>c_credit,pData,dbdatlen(dbproc,6));
            if(pData=dbdata(dbproc, 7))
            {
                datetime = *((DBDATETIME *) pData);
                dbdatecrack(dbproc,&pnod->o_entry_d,&datetime);
            };
            if(pData=dbdata(dbproc, 8))
                commit_flag = (*(DBTINYINT *) pData);
        }; // while (dbnextrow)
    }; // if (DBROWS && dbnumcols)
}; // while (dbresults)
}; // if (dbrpcexec)
}; // if (dbrpcinit)
if (bDeadlock)
{
    num_deadlocks++;
    bDeadlock = FALSE;
    userlog("NewOrder Deadlock Retry (%d)",num_deadlocks);
    Sleep(10 * tryit);
}
else
{

```

```

        if (commit_flag == 1)
        {
            pnod->total_amount = pnod->total_amount *
                ((1 + pnod->w_tax + pnod->d_tax) * (1 - pnod->c_discount));
            strcpy(pnod->execution_status,"Transaction committed.");
            return(SVC_NOERROR);
        }
        else
        {
            strcpy(pnod->execution_status,"Item number is not valid.");
            return(SVC_BADITEMID);
        }
    }; // !bDeadlock
}; // for (tryit)

// If we reached here, it means we quit after MAX_RETRY deadlocks
strcpy(pnod->execution_status,"Hit deadlock max.");
userlog("NewOrder Deadlock Failure (%d)",num_deadlocks);
return(SVCERR_DEADLOCK);
}; // SQLNewOrder

//=====
// FUNCTION: SQLPayment
//
// Handles the payment transaction.
//
// ARGUMENTS:
// PAYMENT_DATA Payment input/output data structure
// dbdata (global)
// bDeadlock (global)
//
// RETURNS:
// SVC_NOERROR success
// !SVC_NOERROR failure
//
// COMMENTS: None
//
//=====
int SQLPayment(PAYMENT_DATA *ppd)
{
    RETCODE rc;
    int tryit;
    short num_deadlocks = 0;
    DBDATETIME datetime;
    BYTE * pData;

    bFailed = FALSE;
    bDeadlock = FALSE;

    for (tryit=0; tryit < DeadlockRetry; tryit++)
    {
        if (dbrpcinit(dbproc,"tpcc_payment",0) == SUCCEED)
        {
            dbrpcparam(dbproc,NULL,0,SQLINT2,-1,-1,(BYTE *) &ppd->w_id);
            dbrpcparam(dbproc,NULL,0,SQLINT2,-1,-1,(BYTE *) &ppd->c_w_id);
            dbrpcparam(dbproc,NULL,0,SQLFLT8,-1,-1,(BYTE *) &ppd->h_amount);
            dbrpcparam(dbproc,NULL,0,SQLINT1,-1,-1,(BYTE *) &ppd->d_id);
            dbrpcparam(dbproc,NULL,0,SQLINT1,-1,-1,(BYTE *) &ppd->c_d_id);
            dbrpcparam(dbproc,NULL,0,SQLINT4,-1,-1,(BYTE *) &ppd->c_id);
            if (ppd->c_id == 0)

```

```

    {
        dbrpcparam(dbproc, NULL, 0, SQLCHAR, -1, strlen(ppd->c_last), ppd-
>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))
                    ppd->c_id = *((DBINT *) pData);
                if (pData=dbdata(dbproc, 2))
                    UtilStrCpy(ppd->c_last, pData, dbdatlen(dbproc, 2));
                if (pData=dbdata(dbproc, 3))
                {
                    datetime = *((DBDATETIME *) pData);
                    dbdatecrack(dbproc, &ppd->h_date, &datetime);
                };
                if (pData=dbdata(dbproc, 4))
                    UtilStrCpy(ppd->w_street_1, pData, dbdatlen(dbproc, 4));
                if (pData=dbdata(dbproc, 5))
                    UtilStrCpy(ppd->w_street_2, pData, dbdatlen(dbproc, 5));
                if (pData=dbdata(dbproc, 6))
                    UtilStrCpy(ppd->w_city, pData, dbdatlen(dbproc, 6));
                if (pData=dbdata(dbproc, 7))
                    UtilStrCpy(ppd->w_state, pData, dbdatlen(dbproc, 7));
                if (pData=dbdata(dbproc, 8))
                    UtilStrCpy(ppd->w_zip, pData, dbdatlen(dbproc, 8));
                if (pData=dbdata(dbproc, 9))
                    UtilStrCpy(ppd->d_street_1, pData, dbdatlen(dbproc, 9));
                if (pData=dbdata(dbproc, 10))
                    UtilStrCpy(ppd-
>d_street_2, pData, dbdatlen(dbproc, 10));
                if (pData=dbdata(dbproc, 11))
                    UtilStrCpy(ppd->d_city, pData, dbdatlen(dbproc, 11));
                if (pData=dbdata(dbproc, 12))
                    UtilStrCpy(ppd->d_state, pData, dbdatlen(dbproc, 12));
                if (pData=dbdata(dbproc, 13))
                    UtilStrCpy(ppd->d_zip, pData, dbdatlen(dbproc, 13));
                if (pData=dbdata(dbproc, 14))
                    UtilStrCpy(ppd->c_first, pData, dbdatlen(dbproc, 14));
                if (pData=dbdata(dbproc, 15))
                    UtilStrCpy(ppd->c_middle, pData, dbdatlen(dbproc, 15));
                if (pData=dbdata(dbproc, 16))
                    UtilStrCpy(ppd-
>c_street_1, pData, dbdatlen(dbproc, 16));
                if (pData=dbdata(dbproc, 17))
                    UtilStrCpy(ppd-
>c_street_2, pData, dbdatlen(dbproc, 17));
                if (pData=dbdata(dbproc, 18))
                    UtilStrCpy(ppd->c_city, pData, dbdatlen(dbproc, 18));
                if (pData=dbdata(dbproc, 19))
                    UtilStrCpy(ppd->c_state, pData, dbdatlen(dbproc, 19));
                if (pData=dbdata(dbproc, 20))
                    UtilStrCpy(ppd->c_zip, pData, dbdatlen(dbproc, 20));

```

```

                if (pData=dbdata(dbproc, 21))
                    UtilStrCpy(ppd->c_phone, pData, dbdatlen(dbproc, 21));
                if (pData=dbdata(dbproc, 22))
                {
                    datetime = *((DBDATETIME *) pData);
                    dbdatecrack(dbproc, &ppd->c_since, &datetime);
                };
                if (pData=dbdata(dbproc, 23))
                    UtilStrCpy(ppd->c_credit, pData, dbdatlen(dbproc, 23));
                if (pData=dbdata(dbproc, 24))
                    dbconvert(dbproc, SQLNUMERIC, pData, sizeof(DBNUMERIC),
SQLFLTN, (CHAR *) &ppd->c_credit_lim, 8);
                if (pData=dbdata(dbproc, 25))
                    dbconvert(dbproc, SQLNUMERIC, pData, sizeof(DBNUMERIC),
SQLFLTN, (CHAR *) &ppd->c_discount, 8);
                if (pData=dbdata(dbproc, 26))
                    dbconvert(dbproc, SQLNUMERIC, pData, sizeof(DBNUMERIC),
SQLFLTN, (CHAR *) &ppd->c_balance, 8);
                if (pData=dbdata(dbproc, 27))
                    UtilStrCpy(ppd->c_data, pData, dbdatlen(dbproc, 27));
            }; // while (dbnextrow)
        }; // if (DBROWS && dbnumcols)
    }; // while (dbresults)
}; // if (dbrpcexe)
if (bDeadlock)
{
    num_deadlocks++;
    bDeadlock = FALSE;
    userlog("Payment Deadlock Retry (%d)", num_deadlocks);
    Sleep(10 * tryit);
}
else
{
    if (ppd->c_id == 0)
    {
        strcpy(ppd->execution_status, "Invalid Customer id, name.");
        return(SVCERR_NOCUSTOMER);
    }
    else
        strcpy(ppd->execution_status, "Transaction committed.");
    return(SVC_NOERROR);
}; // !bDeadlock
}; // for (tryit)

// If we reached here, it means we quit after MAX_RETRY deadlocks
strcpy(ppd->execution_status, "Hit deadlock max.");
userlog("Payment Deadlock Failure (%d)", num_deadlocks);
return(SVCERR_DEADLOCK);
}; // SQLPayment

//=====
// FUNCTION: SQLOrderStatus
//
// Handles the Order Status transaction.
//
// ARGUMENTS:
// ORDER_STATUS_DATA      Payment input/output data structure
// dbdata (global)
// bDeadlock (global)
//

```

```

// RETURNS:
//   SVC_NOERROR success
//   !SVC_NOERROR failure
//
// COMMENTS:   None
//
//=====
int SQLOrderStatus(ORDER_STATUS_DATA * posd)
{
    RETCODE rc;
    int tryit;
    short num_deadlocks = 0;
    int i;
    DBDATETIME datetime;
    BYTE * pData;

    bFailed = FALSE;
    bDeadlock = FALSE;

    for (tryit=0; tryit < DeadlockRetry; tryit++)
    {
        if (dbrpcinit(dbproc,"tpcc_orderstatus", 0) == SUCCEED)
        {
            dbrpcparam(dbproc,NULL,0,SQLINT2,-1,-1,(BYTE *) &posd->w_id);
            dbrpcparam(dbproc,NULL,0,SQLINT1,-1,-1,(BYTE *) &posd->d_id);
            dbrpcparam(dbproc,NULL,0,SQLINT4,-1,-1,(BYTE *) &posd->c_id);
            if (posd->c_id == 0)
            {
                dbrpcparam(dbproc,NULL,0,SQLCHAR,-1,strlen(posd->c_last),posd->
                >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))
                            posd->OlOrderStatusData[i].ol_supply_w_id =
                            (*DBSMALLINT *) pData);
                        if(pData=dbdata(dbproc,2))
                            posd->OlOrderStatusData[i].ol_i_id = (*DBINT *)
                            pData);
                        if(pData=dbdata(dbproc,3))
                            posd->OlOrderStatusData[i].ol_quantity =
                            (*DBSMALLINT *) pData);
                        if(pData=dbdata(dbproc,4))
                            dbconvert(dbproc,SQLNUMERIC,pData,sizeof(DBNUMERIC),
                            SQLFLTN,(CHAR *) &posd->
                            >OlOrderStatusData[i].ol_amount,8);
                        if(pData=dbdata(dbproc,5))
                        {
                            datetime = *((DBDATETIME *) pData);
                            dbdatecrack(dbproc,&posd->
                            >OlOrderStatusData[i].ol_delivery_d,&datetime);

```

```

};
        i++;
    }; // while (dbnextrow)
    posd->o_ol_cnt = i;
} // if (DBROWS && dbnumcols == 5)
else
if (DBROWS(dbproc) && (dbnumcols(dbproc) == 8))
{
    while (((rc = dbnextrow(dbproc)) != NO_MORE_ROWS) && (rc !=
    FAIL))
    {
        if(pData=dbdata(dbproc,1))
            posd->c_id = (*(DBINT *) pData);
        if(pData=dbdata(dbproc,2))
            UtilStrCpy(posd->c_last,pData,dbdatlen(dbproc,2));
        if(pData=dbdata(dbproc,3))
            UtilStrCpy(posd->c_first,pData,dbdatlen(dbproc,3));
        if(pData=dbdata(dbproc,4))
            UtilStrCpy(posd->c_middle,pData,dbdatlen(dbproc,4));
        if(pData=dbdata(dbproc,5))
        {
            datetime = *((DBDATETIME *) pData);
            dbdatecrack(dbproc,&posd->o_entry_d,&datetime);
        };
        if(pData=dbdata(dbproc,6))
            posd->o_carrier_id = (*(DBSMALLINT *) pData);
        if(pData=dbdata(dbproc,7))
            dbconvert(dbproc,SQLNUMERIC,pData,sizeof(DBNUMERIC),
            SQLFLTN,(CHAR *) &posd->c_balance,8);
        if(pData=dbdata(dbproc,8))
            posd->o_id = (*(DBINT *) pData);
    }; // while (dbnextrow)
}; // if (DBROWS && dbnumcols == 8)
if (i==0)
    return(SVCERR_NOORDERS); // "No orders found for customer"
}; // while (dbresults)
}; // if (dbrpcexec)
if (bDeadlock)
{
    num_deadlocks++;
    bDeadlock = FALSE;
    userlog("OrderStatus Deadlock Retry (%d)",num_deadlocks);
    Sleep(10 * tryit);
}
else
{
    if (posd->c_id == 0 && posd->c_last[0] == 0)
    {
        strcpy(posd->execution_status,"Invalid Customer id,name.");
        return(SVCERR_NOCUSTOMER);
    }
    else
        strcpy(posd->execution_status,"Transaction committed.");
    return(SVC_NOERROR);
}; // !bDeadlock
}; // for (tryit)

// If we reached here, it means we quit after MAX_RETRY deadlocks
strcpy(posd->execution_status,"Hit deadlock max.");
userlog("OrderStatus Deadlock Failure (%d)",num_deadlocks);
return(SVCERR_DEADLOCK);

```

```

}; // SQLOrderStatus

//=====
// FUNCTION: UtilStrCpy
//
// Copies n characters from string pSrc to pDst and places a null
// null character at the end of the destination string. Unlike
// strncpy this function ensures that the result string is always
// null terminated.
//
//=====
void UtilStrCpy(char * pDest, char * pSrc, int n)
{
    strncpy(pDest, pSrc, n);
    pDest[n] = '\0';
    return;
}; // UtilStrCpy

//=====
//
// Function name: GetArgs
//
//=====
VOID GetArgs(INT argc, CHAR **argv)
{
    INT j;
    CHAR * ptr;
    BOOL bRslt = TRUE;

    for (j = 1; j < argc; ++j)
    {
        ptr = argv[j];
        switch (ptr[1])
        {
            case 's':
            case 'S':
                strcpy(szServer,ptr+2);
                break;

            case 'd':
            case 'D':
                strcpy(szDatabase,ptr+2);
                break;

        }; // switch(ptr[1])
    }; // for (j = 1; j < argc; ++j)
}; // GetArgs

```

tpccdelv.c

```

// tpccdelv.// tpccdelv.c
//
// Copyright Unisys, 1997
// Copyright Microsoft, 1996

#include <windows.h>
#include <malloc.h>
#include <stdarg.h>
#include <stdio.h>

```

```

#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <sys\timeb.h>

#include <atmi.h>
#include <userlog.h>

#include "tpccsvr.h"

int    iServerNo = 0;
char   szServer[32] = "tpccdelv";
char   szUser[32] = { 0 };
char   szPassword[32] = { 0 };
char   szDatabase[32] = "tpcc";
char   szService[16] = "tpccdelv";
char   szWork[200];

PDBPROCESS    dbproc;
int    spid; // spid assigned from dblib
BOOL   bFailed;
BOOL   bDeadlock;
short  DeadlockRetry = (short)10;

FILE *fpLog;
char  szLogTitle[32];
BOOL  bFlush = FALSE; // flush after every write

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);
void WriteLog(DELIVERY_DATA * pdd);
BOOL OpenLogFile(void);
void CalculateElapsed(int * pElapsed,LPSYSTEMTIME lpBegin,
                     LPSYSTEMTIME lpEnd);
void UtilStrCpy(char * pDest, char * pSrc, int n);
void GetArgs(INT argc, CHAR **argv);

//=====
//
// Function name: tpsvrrinit
//
//=====
tpsvrrinit(int argc, char *argv[])
{
    GetArgs(argc,argv);
    iServerNo = _getpid();
    sprintf(szWork,"%s%d Started, DBServer=%s, DB=%s",
            szService,iServerNo,szServer,szDatabase);
    userlog(szWork);
    if (OpenLogFile())
        return(-1);
    if (SQLInit(szServer,szDatabase,szUser,szPassword))
        return(-1);
    userlog("Database open, initialization complete");
    return(0);
}; // tpsvrrinit

//=====
//

```



```

// Function name: tpsvrdone
//
//=====
void tpsvrdone()
{
    userlog("Shutdown request for tpccdelv server");
    if ( fpLog )
        fclose(fpLog);
    dbclose(dbproc);
    dbexit();
}; // tpsvrdone

//=====
//
// Function name: DELIVERY
//
// Entry point called by tuxedo for DELIVERY service requests.
//
//=====
void DELIVERY(TPSVCINFO * svcinfo)
{
    int iRslt;
    DELIVERY_DATA * pdd;

    pdd = (DELIVERY_DATA *) svcinfo->data;
    iRslt = SQLDelivery(pdd);
    WriteLog(pdd);

    // Check for DBLib termination error
    if (bFailed)
    {
        strcpy(pdd->execution_status,szWork);
        userlog(szWork);
        tpreturn(TPFAIL,SVCERR_DBLIB,svcinfo->data,svcinfo->len,0);
    }
    else
    if (iRslt == 0)
        tpreturn(TPSUCCESS,0,svcinfo->data,svcinfo->len,0);
    else
        tpreturn(TPFAIL,iRslt,svcinfo->data,svcinfo->len,0);
}; // DELIVERY

//=====
//
// Function name: SQLInit
//
// Set global dbproc and spid.
//
// Result:
// FALSE - database open, dbproc valid
// TRUE - database open failed
//
//=====
BOOL SQLInit(CHAR * pSvr,CHAR * pDB,CHAR * pUsr,CHAR * pPW,CHAR * pSvc)
{
    char szApp[32];
    char server[256];
    char database[256];
    char user[256];
    char password[256];

```

```

LOGINREC *login;

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

dbproc = NULL;
strcpy(server,pSvr);
strcpy(database,pDB);
strcpy(user,pUsr);
strcpy(password,pPW);
sprintf(szApp,"%s%d",pSvc,_getpid());

login = dblogin();
if (!*user )
    DBSETLUSER(login,"sa");
else
    DBSETLUSER(login,user);
DBSETLPWD(login,password);
DBSETLHOST(login,szApp);
DBSETLVERSION(login, DBVER60);
// DBSETLPACKET(login,(unsigned short)DEFCLPACKSIZE);

if ((dbproc = dbopen(login,server)) == NULL)
{
    userlog("dbopen failed");
    return TRUE;
};
// Use the the right database
dbuse(dbproc,database);
dbcmd(dbproc,"select @@spid");
dbsqlxexec(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");
dbsqlxexec(dbproc);
while (dbresults(dbproc) != NO_MORE_RESULTS)
{
    while (dbnextrow(dbproc) != NO_MORE_ROWS)
        ;
};

//rollback transaction on abort
dbcmd(dbproc,"set XACT_ABORT ON");
dbsqlxexec(dbproc);
while (dbresults(dbproc) != NO_MORE_RESULTS)
{
    while (dbnextrow(dbproc) != NO_MORE_ROWS)
        ;
};

return(FALSE);
}; // SQLInit

```

```

//=====
// FUNCTION: err_handler
//
//   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_CANCEL
//
// COMMENTS:   None
//=====
int err_handler(DBPROCESS *dbproc, int severity, int dberr, int oserr,
char *dberrstr, char *oserrstr)
{
    if ((dbproc == NULL) || (DBDEAD(dbproc)))
    {
        userlog("ErrHandler: DBPROC is invalid");
        return INT_CANCEL;
    };
    if (bFailed)
        return INT_CANCEL;
    if (oserr != DBNOERR)
    {
        sprintf(szWork, "ErrHandler: OSErr(%ld) - %s", oserr, oserrstr);
        userlog(szWork);
        bFailed = TRUE;
    };

    return INT_CANCEL;
}; // err_handler

//=====
// FUNCTION: msg_handler
//
//   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 operation
//             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)

```

```

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

    // deadlock message
    if (msgno == 1205)
    {
        // set the deadlock indicator
        bDeadlock = TRUE;
        return INT_CONTINUE;
    };

    if (bFailed)
        return INT_CANCEL;

    if (msgno == 0)
        return INT_CONTINUE;
    else
    {
        sprintf(szWork, "MsgHandler: MsgNo(%ld) - %s", msgno, msgtext);
        userlog(szWork);
        bFailed = TRUE;
    };

    return INT_CANCEL;
}; // msg_handler

//=====
// FUNCTION: SQLDelivery
//
// ARGUMENTS:
//   pdd        delivery transaction structure
//   dbdata (global)
//   bDeadlock (global)
//
// RETURNS:
//   SVC_NOERROR success
//   !SVC_NOERROR failure
//
// COMMENTS:   None
//=====
int SQLDelivery(DELIVERY_DATA * pdd)
{
    RETCODE rc;
    int i;
    short num_deadlocks = 0;
    int tryit;
    DBDATETIME datetime;
    BYTE * pData;

    bFailed = FALSE;
    bDeadlock = FALSE;
    pdd->iComplete = 0;

    for (tryit=0; tryit < DeadlockRetry; tryit++)
    {
        if (dbrpcinit(dbproc, "tpcc_delivery", 0) == SUCCEEDED)

```

```

    {
        dbrpcparam(dbproc,NULL,0,SQLINT2,-1,-1,(BYTE *) &pdd->w_id);
        dbrpcparam(dbproc,NULL,0,SQLINT1,-1,-1,(BYTE *) &pdd-
>o_carrier_id);

        if (dbrpcexec(dbproc) == SUCCEEDED)
        {
            while ((rc = dbresults(dbproc)) != NO_MORE_RESULTS) && (rc !=
FAIL))
            {
                while ((rc = dbnextrow(dbproc)) != NO_MORE_ROWS) && (rc !=
FAIL))
                {
                    for (i = 0; i < 10; i++)
                    {
                        if(pData = dbdata(dbproc,i + 1))
                            pdd->o_id[i] = *((DBINT *)pData);
                        else
                            pdd->o_id[i] = 0;
                    };
                }; // while (dbnextrow)
            }; // while (dbresults)
        }; // if (dbrpcexec)
    }; // if (dbrpcinit)
    if (bDeadlock)
    {
        num_deadlocks++;
        bDeadlock = FALSE;
        userlog("Delivery Deadlock Retry (%d)",num_deadlocks);
        Sleep(10 * tryit);
    }
    else
    {
        GetLocalTime(&pdd->EndTime);
        pdd->iComplete = 1;
        strcpy(pdd->execution_status,"Transaction committed.");
        return(SVC_NOERROR);
    };
}; // for (tryit)

// If we reached here, it means we quit after MAX_RETRY deadlocks
strcpy(pdd->execution_status,"Hit deadlock max.");
userlog("Delivery Deadlock Failure (%d)",num_deadlocks);
return(SVCERR_DEADLOCK);

}; // SQLDelivery

//=====
// FUNCTION: WriteLog
//
// Writes the delivery results to a log file.
//
// ARGUMENTS:
// pDelivery delivery information.
//
// RETURNS:
//
// COMMENTS:
// Record format:
// QTime,EndTime,Elapsed,w_id,o_carrier_id,o_id1, ... o_id10
//

```

```

//=====
void WriteLog(DELIVERY_DATA * pdd)
{
    int elapsed = 9999999;
    if (pdd->iComplete)
        CalculateElapsed(&elapsed, &pdd->QTime, &pdd->EndTime);
    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",
pdd->EndTime.wYear - 1900,pdd->EndTime.wMonth,pdd->EndTime.wDay,
pdd->QTime.wHour,pdd->QTime.wMinute,
pdd->QTime.wSecond,pdd->QTime.wMilliseconds,
pdd->EndTime.wHour,pdd->EndTime.wMinute,
pdd->EndTime.wSecond,pdd->EndTime.wMilliseconds,
elapsed,pdd->w_id,pdd->o_carrier_id,
pdd->o_id[0],pdd->o_id[1],pdd->o_id[2],pdd->o_id[3],pdd->o_id[4],
pdd->o_id[5],pdd->o_id[6],pdd->o_id[7],pdd->o_id[8],pdd->o_id[9] );
    if (bFlush)
        fflush(fpLog);
}; // WriteLog

//=====
// FUNCTION: OpenLogFile
//
// Opens the delivery log file.
//
// ARGUMENTS:
// None.
//
// RETURNS:
// FALSE Log file successfully opened
// TRUE Failed to open log file
//
// COMMENTS:
//
//=====
BOOL OpenLogFile(void)
{
    sprintf(szLogTitle,"%s%d",LOGFILE_NAME,iServerNo);
    fpLog = fopen(szLogTitle,"ab");
    if (!fpLog)
    {
        sprintf(szWork,"LogFile %s Open Failed (%d)",
szLogTitle,GetLastError());
        userlog(szWork);
        return(TRUE);
    };
    return(FALSE);
}; // OpenLogFile

//=====
// FUNCTION: CalculateElapsed
//
// Calculates the elapsed time of the delivery transaction.
//
// ARGUMENTS:
// lpBegin time delivery was queued
// lpEnd time delivery update completed
//
// RETURNS:
// int pElapsed elapsed time result (in milliseconds)
//

```

```

//
// COMMENTS:
// None
//
//=====
void CalculateElapsed(int * pElapsed,LPSYSTEMTIME lpBegin,
                    LPSYSTEMTIME lpEnd)
{
    int tmBegin;
    int tmEnd;

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

    // Check for day boundry, this will function for 24 hour period but
    // will fail over a 48 hours period.
    if (*pElapsed < 0)
        *pElapsed = *pElapsed + (24 * 60 * 60 * 1000);
    return;
}; // CalculateElapsed

//=====
// FUNCTION: UtilStrCpy
//
// Copies n characters from string pSrc to pDst and places a null
// 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 strcpy this function ensures that the result string is
// always null terminated.
//
//=====
void UtilStrCpy(char * pDest, char * pSrc, int n)
{
    strcpy(pDest, pSrc, n);
    pDest[n] = '\0';
    return;
}; // UtilStrCpy

//=====
// Function name: GetArgs
//
//=====
void GetArgs(INT argc, CHAR **argv)
{
    INT j;
    CHAR * ptr;
    BOOL bRslt = TRUE;

    for (j = 1; j < argc; ++j)

```

```

{
    ptr = argv[j];
    switch (ptr[1])
    {
        case 's':
        case 'S':
            strcpy(szServer,ptr+2);
            break;

        case 'd':
        case 'D':
            strcpy(szDatabase,ptr+2);
            break;

        case 'F':
        case 'f':
            bFlush = TRUE; //turn on delilog flush when written.
            break;
    }; // switch(ptr[1])
}; // for (j = 1; j < argc; ++j)
}; // GetArgs

```

DELIVERY REPORT MAKEFILE

```

# Microsoft Developer Studio Generated NMAKE File, Format Version 4.20
# ** DO NOT EDIT **

# TARGETTYPE "Win32 (x86) Console Application" 0x0103

!IF "$(CFG)" == ""
CFG=delirpt - Win32 Debug
!MESSAGE No configuration specified. Defaulting to delirpt - Win32 Debug.
!ENDIF

!IF "$(CFG)" != "delirpt - Win32 Release" && "$(CFG)" !=\
"delirpt - Win32 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 /f "delirpt.mak" CFG="delirpt - Win32 Debug"
!MESSAGE
!MESSAGE Possible choices for configuration are:
!MESSAGE
!MESSAGE "delirpt - Win32 Release" (based on "Win32 (x86) Console
Application")
!MESSAGE "delirpt - Win32 Debug" (based on "Win32 (x86) Console
Application")
!MESSAGE
!ERROR An invalid configuration is specified.
!ENDIF

!IF "$(OS)" == "Windows_NT"
NULL=
!ELSE
NULL=nul
!ENDIF

```

```
#####
#####
# Begin Project
CPP=cl.exe
RSC=rc.exe

!IF "$(CFG)" == "delirpt - Win32 Release"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 0
# PROP BASE Output_Dir "delirpt_"
# PROP BASE Intermediate_Dir "delirpt_"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 0
# PROP Output_Dir "delirpt_"
# PROP Intermediate_Dir "delirpt_"
# PROP Target_Dir ""
OUTDIR=.\delirpt_
INTDIR=.\delirpt_

ALL : "$(OUTDIR)\delirpt.exe"

CLEAN :
-@erase "$(INTDIR)\DELIRPT.OBJ"
-@erase "$(OUTDIR)\delirpt.exe"

"$(OUTDIR)" :
if not exist "$(OUTDIR)/$(NULL)" mkdir "$(OUTDIR)"

# ADD BASE CPP /nologo /W3 /GX /O2 /D "WIN32" /D "NDEBUG" /D "_CONSOLE"
/YX /c
# ADD CPP /nologo /W3 /GX /O2 /D "WIN32" /D "NDEBUG" /D "_CONSOLE" /YX /c
CPP_PROJ=/nologo /ML /W3 /GX /O2 /D "WIN32" /D "NDEBUG" /D "_CONSOLE" \
/Fp"$(INTDIR)/delirpt.pch" /YX /Fo"$(INTDIR)/" /c
CPP_OBJS=.\delirpt_
CPP_SBRS=.\
# ADD BASE RSC /l 0x409 /d "NDEBUG"
# ADD RSC /l 0x409 /d "NDEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
BSC32_FLAGS=/nologo /o"$(OUTDIR)/delirpt.bsc"
BSC32_SBRS= \

LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib
odbc32.lib odbccp32.lib /nologo /subsystem:console /machine:I386
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib /nologo /subsystem:console /machine:I386
LINK32_FLAGS=kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib\
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib\
odbccp32.lib /nologo /subsystem:console /incremental:no\
/pdb:"$(OUTDIR)/delirpt.pdb" /machine:I386 /out:"$(OUTDIR)/delirpt.exe"
LINK32_OBJS= \
- "$(INTDIR)\DELIRPT.OBJ"

"$(OUTDIR)\delirpt.exe" : "$(OUTDIR)" $(DEF_FILE) $(LINK32_OBJS)
$(LINK32) @<<
```

```
$(LINK32_FLAGS) $(LINK32_OBJS)
<<
!ELSEIF "$(CFG)" == "delirpt - Win32 Debug"

# PROP BASE Use_MFC 0
# PROP BASE Use_Debug_Libraries 1
# PROP BASE Output_Dir "Debug"
# PROP BASE Intermediate_Dir "Debug"
# PROP BASE Target_Dir ""
# PROP Use_MFC 0
# PROP Use_Debug_Libraries 1
# PROP Output_Dir "Debug"
# PROP Intermediate_Dir "Debug"
# PROP Target_Dir ""
OUTDIR=.\Debug
INTDIR=.\Debug

ALL : "$(OUTDIR)\delirpt.exe"

CLEAN :
-@erase "$(INTDIR)\DELIRPT.OBJ"
-@erase "$(INTDIR)\vc40.idb"
-@erase "$(INTDIR)\vc40.pdb"
-@erase "$(OUTDIR)\delirpt.exe"
-@erase "$(OUTDIR)\delirpt.ilc"
-@erase "$(OUTDIR)\delirpt.pdb"

"$(OUTDIR)" :
if not exist "$(OUTDIR)/$(NULL)" mkdir "$(OUTDIR)"

# ADD BASE CPP /nologo /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D
"_CONSOLE" /YX /c
# ADD CPP /nologo /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D "_CONSOLE"
/YX /c
CPP_PROJ=/nologo /MLd /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D
"_CONSOLE" \
/Fp"$(INTDIR)/delirpt.pch" /YX /Fo"$(INTDIR)/" /Fd"$(INTDIR)/" /c
CPP_OBJS=.\Debug\
CPP_SBRS=.\
# ADD BASE RSC /l 0x409 /d "_DEBUG"
# ADD RSC /l 0x409 /d "_DEBUG"
BSC32=bscmake.exe
# ADD BASE BSC32 /nologo
# ADD BSC32 /nologo
BSC32_FLAGS=/nologo /o"$(OUTDIR)/delirpt.bsc"
BSC32_SBRS= \

LINK32=link.exe
# ADD BASE LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib
odbc32.lib odbccp32.lib /nologo /subsystem:console /debug /machine:I386
# ADD LINK32 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib
odbccp32.lib /nologo /subsystem:console /debug /machine:I386
LINK32_FLAGS=kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib\
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib\
odbccp32.lib /nologo /subsystem:console /incremental:yes\
/pdb:"$(OUTDIR)/delirpt.pdb" /debug /machine:I386
/out:"$(OUTDIR)/delirpt.exe"
LINK32_OBJS= \
```

```

    "$(INTDIR)\DELIRPT.OBJ"

"$ (OUTDIR)\delirpt.exe" : "$(OUTDIR)" $(DEF_FILE) $(LINK32_OBJS)
    $(LINK32) @<<
    $(LINK32_FLAGS) $(LINK32_OBJS)
<<

!ENDIF

.c{$(CPP_OBJS)}.obj:
    $(CPP) $(CPP_PROJ) $<

.cpp{$(CPP_OBJS)}.obj:
    $(CPP) $(CPP_PROJ) $<

.cxx{$(CPP_OBJS)}.obj:
    $(CPP) $(CPP_PROJ) $<

.c{$(CPP_SBRs)}.sbr:
    $(CPP) $(CPP_PROJ) $<

.cpp{$(CPP_SBRs)}.sbr:
    $(CPP) $(CPP_PROJ) $<

.cxx{$(CPP_SBRs)}.sbr:
    $(CPP) $(CPP_PROJ) $<

#####
#####
# Begin Target

# Name "delirpt - Win32 Release"
# Name "delirpt - Win32 Debug"

!IF "$(CFG)" == "delirpt - Win32 Release"

!ELSEIF "$(CFG)" == "delirpt - Win32 Debug"

!ENDIF

#####
#####
# Begin Source File

SOURCE=.\DELIRPT.C

"$ (INTDIR)\DELIRPT.OBJ" : $(SOURCE) "$(INTDIR)"

# End Source File
# End Target
# End Project
#####
#####

                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
                                //insufficient 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 = 4;
                                //delirpt version
int      versionMM = 0;
int      versionLS = 0;

```

```

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
CHAR szLogFileTitle[100];
#define DEFAULTLOGTITLE "delilog."

//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 delirtp 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      iTotResponse;
    int      iLines;
    double  fAverage;
    char    szDelivery[128];

    ResetLogFile();

    iTotResponse = 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 ( !LogEOF(LOGFILE_READ_EOF) )
    {
        if ( CheckTimes(&reportLine) )
            continue;
        iLines++;
        iTotResponse += 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)iTotResponse /
(double)iLines)/(double)1000;
    printf("Total Deliveries:      %10.0f\n", (float)iLines);
    printf("Total Response Times:    %10.3f\n",
((float)iTotResponse/(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      iTotBuckets;
    double  iTot;

```



```

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;
    }
}

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");

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 ( CheckTimes(&reportLine) )
            continue;
        psBuckets[reportLine.response]++;
        iTotal++;
        if ( iMaxSeconds < reportLine.response )
            iMaxSeconds = reportLine.response;
    }
}

printf("Max Response Time = %d.%d\n",
(iMaxSeconds / 1000), (iMaxSeconds % 1000));

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 ( !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 if the delilog record falls
 *           within 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
 *           start and end times.
 *           TRUE if the report line is
 *           within the
 *           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(szLogFileTitle, "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;
    }
    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
 */

```

```

 *
 * 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 log file."
        },
        { 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\n", 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;
    UINT uLogTitleLen;

    iStartTime = 0;
    iEndTime = 0;
    iReport = 4;
    strcpy(szLogFileTitle,DEFAULTLOGTITLE);

    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 'F':
        case 'f':
            uLogTitleLen = strlen(argv[i] - 2);
            if (uLogTitleLen > 0 && uLogTitleLen <
sizeof(szLogFileTitle))
            {
                strcpy(szLogFileTitle,argv[i]+2);
                printf("Log File Title set to %s",szLogFileTitle);
            };
            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
\n");
    printf("-E End Time HH:MM:SS:MMM
\n");
    printf("-R 1)Average Response, 2)90th 3) Skipped 4) 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 7.0\n");
    printf("*\n");
    printf("** HTML TPC-C BENCHMARK KIT: Delivery Report\n");
    printf("** Version %d.%2.2d.%3.3d\n");
    printf("*\n");
    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;
}

```


Appendix B - Database Design

Build Scripts

BACKUP.SQL

```
-- File:      BACKUP.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates backup of tpcc database

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

backup database tpcc to tpccback1, tpccback2, tpccback3, tpccback4 with
init, stats = 5

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

CREATEDB.SQL

```
-- File:      CREATEDB.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates tpcc database and backup files
--           for 1728 warehouses.

use master
go

-- remove any existing database and backup files

exec sp_dbremove tpcc, dropdev
exec sp_dropdevice 'tpccback1', delfile
exec sp_dropdevice 'tpccback2', delfile
exec sp_dropdevice 'tpccback3', delfile
exec sp_dropdevice 'tpccback4', delfile
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

-- create main database files

create database tpcc on
```

```
(name="MSSQL70_tpcc_root",filename="C:\MSSQL7\Data\tpcc_root.mdf",
size=10MB, FILEGROWTH=0)
log on
(name="MSSQL70_tpcc_log",filename="L:",size=54996MB, FILEGROWTH=0)

-- create filegroups

alter database tpcc add filegroup MSSQL70_misc_fg
alter database tpcc add filegroup MSSQL70_cs_fg

-- add files to filegroups

alter database tpcc add file
(name="MSSQL70_misc1",filename="M:",size=7000MB, FILEGROWTH=0),
(name="MSSQL70_misc2",filename="N:",size=7000MB, FILEGROWTH=0),
(name="MSSQL70_misc3",filename="O:",size=7000MB, FILEGROWTH=0),
(name="MSSQL70_misc4",filename="P:",size=7000MB, FILEGROWTH=0),
(name="MSSQL70_misc5",filename="Q:",size=7000MB, FILEGROWTH=0),
(name="MSSQL70_misc6",filename="R:",size=7000MB, FILEGROWTH=0),
(name="MSSQL70_misc7",filename="S:",size=7000MB, FILEGROWTH=0)
to filegroup MSSQL70_misc_fg

alter database tpcc add file
(name="MSSQL70_cs1",filename="E:",size=15000MB, FILEGROWTH=0),
(name="MSSQL70_cs2",filename="F:",size=15000MB, FILEGROWTH=0),
(name="MSSQL70_cs3",filename="G:",size=15000MB, FILEGROWTH=0),
(name="MSSQL70_cs4",filename="H:",size=15000MB, FILEGROWTH=0),
(name="MSSQL70_cs5",filename="I:",size=15000MB, FILEGROWTH=0),
(name="MSSQL70_cs6",filename="J:",size=15000MB, FILEGROWTH=0),
(name="MSSQL70_cs7",filename="K:",size=15000MB, FILEGROWTH=0)
to filegroup MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)
go

-- create backup devices

exec sp_addumpdevice 'disk','tpccback1','T:\tpccback1.dmp'
exec sp_addumpdevice 'disk','tpccback2','U:\tpccback2.dmp'
exec sp_addumpdevice 'disk','tpccback3','T:\tpccback3.dmp'
exec sp_addumpdevice 'disk','tpccback4','U:\tpccback4.dmp'
go

-- create backup devices



### DBOPT1.SQL



-- File:      DBOPT1.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Sets database options for data load

use master
```

```

go
exec sp_dboption tpcc,'select into/bulkcopy',true
exec sp_dboption tpcc,'trunc. log on chkpt.',true
go

```

```

use tpcc
go
checkpoint
go

```

DBOPT2.SQL

```

-- File:          DBOPT2.SQL
--               Microsoft TPC-C Benchmark Kit Ver. 4.01
--               Copyright Microsoft, 1996
-- Purpose:      Resets database options after data load

```

```

use master
go

```

```

sp_dboption tpcc,'select ',false
go

```

```

sp_dboption tpcc,'trunc. ',false
go

```

```

use tpcc
go

```

```

checkpoint
go

```

```

sp_configure allow,1
go

```

```

reconfigure with override
go

```

```

/*                               */
/* Set option values for user-defined indexes */
/*                               */

```

```

sp_indexoption 'customer','AllowPageLocks',FALSE
go
sp_indexoption 'district','AllowPageLocks',FALSE
go
sp_indexoption 'warehouse','AllowPageLocks',FALSE
go
sp_indexoption 'stock','AllowPageLocks',FALSE
go
sp_indexoption 'order_line','AllowRowLocks',FALSE
go
sp_indexoption 'orders','AllowRowLocks',FALSE
go
sp_indexoption 'new_order','AllowRowLocks',FALSE

```

```

go
sp_indexoption 'item','AllowRowLocks',FALSE
go
sp_indexoption 'item','AllowPageLocks',FALSE
go

```

```

Print ' '
Print '*****'
Print 'Pre-specified Locking Hierarchy:'
Print '  Lockflag = 0 ==> No pre-specified hierarchy'
Print '  Lockflag = 1 ==> Lock at Page-level then Table-level'
Print '  Lockflag = 2 ==> Lock at Row-level then Table-level'
Print '  Lockflag = 3 ==> Lock at Table-level'
Print ' '

```

```

select name,lockflags
from sysindexes
where object_id("warehouse")=id or
      object_id("district")=id or
      object_id("customer")=id or
      object_id("stock")=id or
      object_id("orders")=id or
      object_id("order_line")=id or
      object_id("history")=id or
      object_id("new_order")=id or
      object_id("item")=id
order by lockflags asc
go

```

```

sp_configure allow,0
go

```

```

reconfigure with override
go

```

```

exec sp_dboption tpcc, 'auto update statistics', FALSE
exec sp_dboption tpcc, 'auto create statistics', FALSE
go

```

```

exec sp_tableoption "district","pintable",true
exec sp_tableoption "warehouse","pintable",true
exec sp_tableoption "new_order","pintable",true
exec sp_tableoption "item","pintable",true
go

```

IDXCUSCL.SQL

```

-- File:          IDXCUSCL.SQL
--               Microsoft TPC-C Benchmark Kit Ver. 4.00
--               Copyright Microsoft, 1996
-- Purpose:      Creates clustered index on customer table

```

```

use tpcc
go

```

```

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

```

```

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

create unique clustered index customer_c1 on customer(c_w_id, c_d_id,
c_id)
    on MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go

```

IDXCUSNC.SQL

```

-- File:      IDXCUSNC.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates non-clustered index on customer table

```

```

use tpcc
go

```

```

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

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

create unique nonclustered index customer_nc1 on customer(c_w_id, c_d_id,
c_last, c_first, c_id)
    on MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go

```

IDXDISCL.SQL

```

-- File:      IDXDISCL.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates clustered index on district table

```

```

use tpcc
go

```

```

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

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

create unique clustered index district_c1 on district(d_w_id, d_id)
    on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go

```

IDXITMCL.SQL

```

-- File:      IDXITMCL.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates clustered index on item table

```

```

use tpcc
go

```

```

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

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

create unique clustered index item_c1 on item(i_id)
    on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go

```

IDXNODCL.SQL

```

-- File:      IDXNODCL.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates clustered index on new_order table

```

```

use tpcc
go

```

```

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert (varchar(30),@startdate,9)

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

create unique clustered index new_order_c1 on new_order(no_w_id, no_d_id,
no_o_id)
    on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert (varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go

```

IDXODLCL.SQL

```

-- File:      IDXODLCL.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates clustered index on order_line table

```

```

use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert (varchar(30),@startdate,9)

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

create unique clustered index order_line_c1 on order_line(ol_w_id,
ol_d_id, ol_o_id, ol_number)
    on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert (varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go

```

IDXORDCL.SQL

```

-- File:      IDXORDCL.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates clustered index on orders table

```

```

use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert (varchar(30),@startdate,9)

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

create unique clustered index orders_c1 on orders(o_w_id, o_d_id, o_id)
    on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert (varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go

```

IDXORDNC.SQL

```

-- File:      IDXORDNC.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates non-clustered index on orders table

```

```

use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert (varchar(30),@startdate,9)

if exists ( select name from sysindexes where name = 'orders_nc1' )
    drop index orders.orders_nc1

create unique nonclustered index orders_nc1 on orders(o_w_id, o_d_id,
o_c_id, o_id)
    on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert (varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go

```

IDXSTKCL.SQL

```

-- File:      IDXSTKCL.SQL

```

```
--          Microsoft TPC-C Benchmark Kit Ver. 4.00
--          Copyright Microsoft, 1996
-- Purpose:  Creates clustered index on stock table

use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert (varchar(30),@startdate,9)

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

create unique clustered index stock_c1 on stock(s_i_id, s_w_id)
    on MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert (varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXWARCL.SQL

```
-- File:      IDXWARCL.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates clustered index on warehouse table

use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert (varchar(30),@startdate,9)

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

create unique clustered index warehouse_c1 on warehouse(w_id)
    on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert (varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

RESTORE.SQL

```
-- File:      RESTORE.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Loads database backup from backup files

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert (varchar(30),@startdate,9)

restore database tpcc from tpccback1, tpccback2, tpccback3, tpccback4 with
replace, stats = 5

select @enddate = getdate()
select "End date: ", convert (varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

TABLES .SQL

```
-- File:      TABLES.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates TPC-C tables

use tpcc
go

if exists ( select name from sysobjects where name = 'warehouse' )
    drop table warehouse
go
create table warehouse
(
    w_id                smallint,
    w_name              char(10),
    w_street_1          char(20),
    w_street_2          char(20),
    w_city              char(20),
    w_state             char(2),
    w_zip               char(9),
    w_tax               numeric(4,4),
    w_ytd               numeric(12,2)
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name = 'district' )
    drop table district
go
create table district
(
    d_id                tinyint,
    d_w_id              smallint,
    d_name              char(10),
    d_street_1          char(20),
```

```

        d_street_2          char(20),
        d_city              char(20),
        d_state             char(2),
        d_zip               char(9),
        d_tax               numeric(4,4),
        d_ytd               numeric(12,2),
        d_next_o_id        int
    ) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name = 'customer' )
    drop table customer
go
create table customer
(
    c_id                    int,
    c_d_id                  tinyint,
    c_w_id                  smallint,
    c_first                 char(16),
    c_middle                char(2),
    c_last                  char(16),
    c_street_1              char(20),
    c_street_2              char(20),
    c_city                  char(20),
    c_state                 char(2),
    c_zip                   char(9),
    c_phone                 char(16),
    c_since                 datetime,
    c_credit                char(2),
    c_credit_lim            numeric(12,2),
    c_discount              numeric(4,4),
    c_balance               numeric(12,2),
    c_ytd_payment           numeric(12,2),
    c_payment_cnt           smallint,
    c_delivery_cnt          smallint,
    c_data                  char(500)
) on MSSQL70_cs_fg
go

if exists ( select name from sysobjects where name = 'history' )
    drop table history
go
create table history
(
    h_c_id                  int,
    h_c_d_id                tinyint,
    h_c_w_id                smallint,
    h_d_id                  tinyint,
    h_w_id                  smallint,
    h_date                  datetime,
    h_amount                numeric(6,2),
    h_data                  char(24)
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name = 'new_order' )
    drop table new_order
go
create table new_order
(
    no_o_id                 int,

```

```

        no_d_id             tinyint,
        no_w_id             smallint
    ) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name = 'orders' )
    drop table orders
go
create table orders
(
    o_id                    int,
    o_d_id                  tinyint,
    o_w_id                  smallint,
    o_c_id                  int,
    o_entry_d               datetime,
    o_carrier_id            tinyint,
    o_ol_cnt                tinyint,
    o_all_local              tinyint
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name = 'order_line' )
    drop table order_line
go
create table order_line
(
    ol_o_id                 int,
    ol_d_id                 tinyint,
    ol_w_id                 smallint,
    ol_number                tinyint,
    ol_i_id                 int,
    ol_supply_w_id           smallint,
    ol_delivery_d            datetime,
    ol_quantity              smallint,
    ol_amount                numeric(6,2),
    ol_dist_info             char(24)
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name = 'item' )
    drop table item
go
create table item
(
    i_id                    int,
    i_im_id                 int,
    i_name                   char(24),
    i_price                  numeric(5,2),
    i_data                   char(50)
) on MSSQL70_misc_fg
go

if exists ( select name from sysobjects where name = 'stock' )
    drop table stock
go
create table stock
(
    s_i_id                  int,
    s_w_id                  smallint,
    s_quantity               smallint,
    s_dist_01                char(24),

```

```

s_dist_02          char(24),
s_dist_03          char(24),
s_dist_04          char(24),
s_dist_05          char(24),
s_dist_06          char(24),
s_dist_07          char(24),
s_dist_08          char(24),
s_dist_09          char(24),
s_dist_10          char(24),
s_ytd              int,
s_order_cnt        smallint,
s_remote_cnt       smallint,
s_data             char(50)
) on MSSQL70_cs_fg
go

```

VERIFYTPCCLOAD

```

use tpcc
print 'WAREHOUSE'
select rows from sysindexes where id=object_id("warehouse")
print 'DISTRICT = (10 * No of warehouses)'
select rows from sysindexes where id=object_id("district")
print 'ITEM = 100,000'
select rows from sysindexes where id=object_id("item")
print 'CUSTOMER = (30,000 * No of warehouses)'
select rows from sysindexes where id=object_id("customer")
print 'ORDERS = (30,000 * No of warehouses)'
select rows from sysindexes where id=object_id("orders")
print 'HISTORY = (30,000 * No of warehouses)'
select rows from sysindexes where id=object_id("history")
print 'STOCK = (100,000 * No of warehouses)'
select rows from sysindexes where id=object_id("stock")
print 'ORDER_LINE = (300,000 * No of warehouses + some change)'
select rows from sysindexes where id=object_id("order_line")
print 'NEW_ORDER = (9000 * No of warehouses)'
select rows from sysindexes where id=object_id("new_order")
print '*****Index Check*****'
use tpcc
go
sp_helpindex customer
go
sp_helpindex stock
go
sp_helpindex district
go
sp_helpindex item
go
sp_helpindex new_order
go
sp_helpindex orders
go
sp_helpindex order_line
go
sp_helpindex warehouse
go

```

Stored Procedures

DELIVERY.SQL

```

-- File:      DELIVERY.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates delivery transaction stored procedure

use tpcc
go

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 top 1 @o_id = no_o_id
        from new_order (serializable uplock)
        where no_w_id = @w_id and
              no_d_id = @d_id
        order by no_o_id asc

        if (@@rowcount <> 0)
        begin

-- claim the order for this district

```

```

delete new_order
where no_w_id = @w_id and
      no_d_id = @d_id and
      no_o_id = @o_id

-- set carrier_id on this order (and get customer id)

update orders
  set o_carrier_id = @o_carrier_id,
      @c_id        = o_c_id
where o_w_id = @w_id and
      o_d_id = @d_id and
      o_id     = @o_id

-- set date in all lineitems for this order (and sum amounts)

update order_line
  set ol_delivery_d = getdate(),
      @total        = @total + ol_amount
where ol_w_id = @w_id and
      ol_d_id = @d_id and
      ol_o_id = @o_id

-- accumulate lineitem amounts for this order into customer

update customer
  set c_balance      = c_balance + @total,
      c_delivery_cnt = c_delivery_cnt + 1

where c_w_id = @w_id and
      c_d_id = @d_id and
      c_id   = @c_id

end

select @oid1 = case @d_id when 1 then @o_id else @oid1 end,
       @oid2 = case @d_id when 2 then @o_id else @oid2 end,
       @oid3 = case @d_id when 3 then @o_id else @oid3 end,
       @oid4 = case @d_id when 4 then @o_id else @oid4 end,
       @oid5 = case @d_id when 5 then @o_id else @oid5 end,
       @oid6 = case @d_id when 6 then @o_id else @oid6 end,
       @oid7 = case @d_id when 7 then @o_id else @oid7 end,
       @oid8 = case @d_id when 8 then @o_id else @oid8 end,
       @oid9 = case @d_id when 9 then @o_id else @oid9 end,
       @oid10 = case @d_id when 10 then @o_id else @oid10 end

end

commit tran d

-- return delivery data to client

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

```

```

@oid10
go

NEWORD.SQL

-- File:      NEWORD.SQL
-- Microsoft TPC-C Benchmark Kit Ver. 4.00
-- Copyright Microsoft, 1996
-- Purpose:   Creates new order transaction stored procedure
--
-- Modified 9/21/98 - Jamie Reding - Microsoft Corporation
-- Reordered @rowcount check so that invalid supply warehouse
id,
-- as well as invalid item id, is detected and causes explicit
-- transaction rollback.
--
use tpcc
go

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

create proc tpcc_neworder
    @w_id      smallint,
    @d_id      tinyint,
    @c_id      int,
    @o_ol_cnt  tinyint,
    @o_all_local tinyint,
    @i_id1 int = 0, @s_w_id1 smallint =
0, @ol_qty1 smallint = 0,
    @i_id2 int = 0, @s_w_id2 smallint =
0, @ol_qty2 smallint = 0,
    @i_id3 int = 0, @s_w_id3 smallint =
0, @ol_qty3 smallint = 0,
    @i_id4 int = 0, @s_w_id4 smallint =
0, @ol_qty4 smallint = 0,
    @i_id5 int = 0, @s_w_id5 smallint =
0, @ol_qty5 smallint = 0,
    @i_id6 int = 0, @s_w_id6 smallint =
0, @ol_qty6 smallint = 0,
    @i_id7 int = 0, @s_w_id7 smallint =
0, @ol_qty7 smallint = 0,
    @i_id8 int = 0, @s_w_id8 smallint =
0, @ol_qty8 smallint = 0,
    @i_id9 int = 0, @s_w_id9 smallint =
0, @ol_qty9 smallint = 0,
    @i_id10 int = 0, @s_w_id10 smallint
= 0, @ol_qty10 smallint = 0,
    @i_id11 int = 0, @s_w_id11 smallint
= 0, @ol_qty11 smallint = 0,
    @i_id12 int = 0, @s_w_id12 smallint
= 0, @ol_qty12 smallint = 0,
    @i_id13 int = 0, @s_w_id13 smallint
= 0, @ol_qty13 smallint = 0,
    @i_id14 int = 0, @s_w_id14 smallint
= 0, @ol_qty14 smallint = 0,

```



```

= 0, @ol_qty15 smallint = 0
                                @i_id15 int = 0, @s_w_id15 smallint

as
declare @w_tax                numeric(4,4),
        @d_tax                numeric(4,4),
        @c_last               char(16),
        @c_credit             char(2),
        @c_discount           numeric(4,4),
        @i_price              numeric(5,2),
        @i_name               char(24),
        @i_data               char(50),
        @o_entry_d            datetime,
        @remote_flag          int,
        @s_quantity           smallint,
        @s_data               char(50),
        @s_dist               char(24),
        @li_no                int,
        @o_id                  int,
        @commit_flag          tinyint,
        @li_id                int,
        @li_s_w_id            smallint,
        @li_qty               smallint,
        @ol_number            int,
        @c_id_local           int

begin

    begin transaction n

-- get district tax and next available order id and update
-- plus initialize local variables

    update district
    set      @d_tax            = d_tax,
            @o_id             = d_next_o_id,
            d_next_o_id      = d_next_o_id + 1,
            @o_entry_d        = getdate(),
            @li_no            = 0,
            @commit_flag      = 1
    where   d_w_id            = @w_id and
            d_id              = @d_id

-- process orderlines

    while (@li_no < @o_ol_cnt)
    begin

        select @li_no = @li_no + 1

-- set i_id, s_w_id, and qty for this lineitem

        select @li_id = case @li_no
            when 1 then @i_id1
            when 2 then @i_id2
            when 3 then @i_id3
            when 4 then @i_id4
            when 5 then @i_id5
            when 6 then @i_id6
            when 7 then @i_id7

```

```

        when 8 then @i_id8
        when 9 then @i_id9
        when 10 then @i_id10
        when 11 then @i_id11
        when 12 then @i_id12
        when 13 then @i_id13
        when 14 then @i_id14
        when 15 then @i_id15
    end,

    @li_s_w_id = case @li_no
        when 1 then @s_w_id1
        when 2 then @s_w_id2
        when 3 then @s_w_id3
        when 4 then @s_w_id4
        when 5 then @s_w_id5
        when 6 then @s_w_id6
        when 7 then @s_w_id7
        when 8 then @s_w_id8
        when 9 then @s_w_id9
        when 10 then @s_w_id10
        when 11 then @s_w_id11
        when 12 then @s_w_id12
        when 13 then @s_w_id13
        when 14 then @s_w_id14
        when 15 then @s_w_id15
    end,

    @li_qty = case @li_no
        when 1 then @ol_qty1
        when 2 then @ol_qty2
        when 3 then @ol_qty3
        when 4 then @ol_qty4
        when 5 then @ol_qty5
        when 6 then @ol_qty6
        when 7 then @ol_qty7
        when 8 then @ol_qty8
        when 9 then @ol_qty9
        when 10 then @ol_qty10
        when 11 then @ol_qty11
        when 12 then @ol_qty12
        when 13 then @ol_qty13
        when 14 then @ol_qty14
        when 15 then @ol_qty15
    end

-- get item data (no one updates item)

    select @i_price = i_price,
           @i_name   = i_name,
           @i_data   = i_data
    from   item (tablock repeatableread)
    where  i_id = @li_id

-- update stock values

    update stock
    set      s_ytd            = s_ytd + @li_qty,
           @s_quantity       = s_quantity -
@li_qty +

```



```

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 Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates order status transaction stored procedure

use tpcc
go

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

create proc tpcc_orderstatus @w_id          smallint,
                             @d_id          tinyint,
                             @c_id          int,
                             @c_last       char(16) = ""
as

declare @c_balance      numeric(12,2),
        @c_first        char(16),
        @c_middle       char(2),
        @o_id           int,
        @o_entry_d      datetime,
        @o_carrier_id   smallint,
        @cnt            smallint

begin tran o

    if (@c_id = 0)
        begin

-- get customer id and info using last name

            select @cnt = (count(*)+1)/2
            from customer (repeatableread)
            where c_last = @c_last and
                  c_w_id = @w_id and
                  c_d_id = @d_id

            set rowcount @cnt

```

```

select @c_id = c_id,
       @c_balance = c_balance,
       @c_first   = c_first,
       @c_last    = c_last,
       @c_middle  = c_middle
from customer (repeatableread)
where c_last = @c_last and
      c_w_id = @w_id and
      c_d_id = @d_id
order by c_w_id, c_d_id, c_last, c_first

set rowcount 0
end

else
    begin

-- get customer info if by id

        select @c_balance = c_balance,
               @c_first   = c_first,
               @c_middle  = c_middle,
               @c_last    = c_last
        from customer (repeatableread)
        where c_id   = @c_id and
              c_d_id = @d_id and
              c_w_id = @w_id

        select @cnt = @@rowcount

    end

-- if no such customer

    if (@cnt = 0)
        begin
            raiserror("Customer not found",18,1)
            goto custnotfound
        end

-- get order info

        select @o_id = o_id,
               @o_entry_d = o_entry_d,
               @o_carrier_id = o_carrier_id
        from orders (serializable)
        where o_c_id = @c_id and
              o_d_id = @d_id and
              o_w_id = @w_id
        order by o_id asc

-- select order lines for the current order

        select ol_supply_w_id,
               ol_i_id,
               ol_quantity,
               ol_amount,
               ol_delivery_d
        from order_line (repeatableread)
        where ol_o_id = @o_id and

```

```

        ol_d_id = @d_id and
        ol_w_id = @w_id

```

```

custnotfound:

```

```

commit tran o

```

```

-- return data to client

```

```

select @c_id,
        @c_last,
        @c_first,
        @c_middle,
        @o_entry_d,
        @o_carrier_id,
        @c_balance,
        @o_id

```

```

go

```

PAYMENTS.SQL

```

-- File:      PAYMENT.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates payment transaction stored procedure

```

```

use tpcc
go

```

```

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

```

```

create proc tpcc_payment @w_id          smallint,
                        @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),
        @data       char(500),
        @c_data     char(500),
        @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 (repeatableread)
                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 (repeatableread)
                where c_last = @c_last and
                       c_w_id = @c_w_id and
                       c_d_id = @c_d_id
                order by c_last, c_first

```

```

                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,
        @data          = c_data,
        @c_id_local    = c_id
where c_id      = @c_id and
      c_w_id   = @c_w_id and
      c_d_id   = @c_d_id

-- if customer has bad credit get some more info

      if (@c_credit = "BC")
      begin

--      compute new info

          select @c_data = 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(@data, 1, 458)

--      update customer info

          update customer set
            c_data = @c_data
          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,1,200)
        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

```

STOCKLEV.SQL

```
-- File:      STOCKLEV.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Creates stock level transaction stored procedure

use tpcc
go

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

create proc tpcc_stocklevel    @w_id          smallint,
                              @d_id          tinyint,
                              @threshold    smallint
as

    declare @o_id_low int,
            @o_id_high int

    select @o_id_low = (d_next_o_id - 20),
           @o_id_high = (d_next_o_id - 1)
    from district
    where d_w_id = @w_id and
          d_id   = @d_id

    select count(distinct(s_i_id))
    from stock, order_line
    where ol_w_id = @w_id and
          ol_d_id = @d_id and
          ol_o_id between @o_id_low and @o_id_high and
          s_w_id = ol_w_id and
          s_i_id = ol_i_id and
          s_quantity < @threshold

go
```

Loader Source

GETARGS.C

```
// File:      GETARGS.C
//           Microsoft TPC-C Kit Ver. 4.00
//           Copyright Microsoft, 1996, 1997, 1998
// Purpose:   Source file for command line processing

// Includes
#include "tpcc.h"

//=====
//
```

```
// Function name: GetArgsLoader
//
//=====

void GetArgsLoader(int argc, char **argv, TPCCLDR_ARGS *pargs)
{
    int      i;
    char     *ptr;

#ifdef DEBUG
    printf("[%ld]DBG: Entering GetArgsLoader()\n", (int)
GetCurrentThreadId());
#endif

    /* init args struct with some useful values */
    pargs->server      = SERVER;
    pargs->user        = USER;
    pargs->password    = PASSWORD;
    pargs->database    = DATABASE;
    pargs->batch       = BATCH;
    pargs->num_warehouses = UNDEF;
    pargs->tables_all  = TRUE;
    pargs->table_item   = FALSE;
    pargs->table_warehouse = FALSE;
    pargs->table_customer = FALSE;
    pargs->table_orders = FALSE;
    pargs->loader_res_file = LOADER_RES_FILE;
    pargs->pack_size     = DEFLDPACKSIZE;
    pargs->starting_warehouse = DEF_STARTING_WAREHOUSE;
    pargs->build_index   = BUILD_INDEX;
    pargs->index_order   = INDEX_ORDER;
    pargs->index_script_path = INDEX_SCRIPT_PATH;
    pargs->scale_down    = SCALE_DOWN;

    /* check for zero command line args */
    if ( argc == 1 )
        GetArgsLoaderUsage();

    for ( i = 1; i < argc; ++i )
    {
        if (argv[i][0] != '-' && argv[i][0] != '/')
        {
            printf("\nUnrecognized command");
            GetArgsLoaderUsage();
            exit(1);
        }

        ptr = argv[i];

        switch (ptr[1])
        {
            case 'h': /* Fall throught */
            case 'H':
                GetArgsLoaderUsage();
                break;

            case 'D':
                pargs->database = ptr+2;
                break;

            case 'P':
```

```

        pargs->password = ptr+2;
        break;
    case 'S':
        pargs->server = ptr+2;
        break;
    case 'U':
        pargs->user = ptr+2;
        break;
    case 'b':
        pargs->batch = atol(ptr+2);
        break;
    case 'W':
        pargs->num_warehouses = atol(ptr+2);
        break;
    case 's':
        pargs->starting_warehouse = atol(ptr+2);
        break;
    case 't':
    {
        pargs->tables_all = FALSE;
        if (strcmp(ptr+2,"item") == 0)
            pargs->table_item = TRUE;
        else if (strcmp(ptr+2,"warehouse")
== 0)
            pargs->table_warehouse =
TRUE;
        else if (strcmp(ptr+2,"customer") ==
0)
            pargs->table_customer = TRUE;
        else if (strcmp(ptr+2,"orders") ==
0)
            pargs->table_orders = TRUE;
        else
        {
            printf("\nUnrecognized command");
            GetArgsLoaderUsage();
            exit(1);
        }
        break;
    }
    case 'f':
        pargs->loader_res_file = ptr+2;
        break;
    case 'p':
        pargs->pack_size = atol(ptr+2);
        break;
    case 'i':
        pargs->build_index = atol(ptr+2);
        break;
    case 'o':
        pargs->index_order = atol(ptr+2);
        break;
    case 'c':
        pargs->scale_down = atol(ptr+2);
        break;
    case 'd':
        pargs->index_script_path = ptr+2;
        break;
    default:
        GetArgsLoaderUsage();
        exit(-1);
        break;
    }
}

/* check for required args */
if (pargs->num_warehouses == UNDEF )
{
    printf("Number of Warehouses is required\n");
    exit(-2);
}

return;
}

//=====
// Function name: GetArgsLoaderUsage
//=====
void GetArgsLoaderUsage()
{
#ifdef DEBUG
    printf("[%ld]DBG: Entering GetArgsLoaderUsage()\n", (int)
GetCurrentThreadId());
#endif

    printf("TPCCLDR:\n\n");
    printf("Parameter
Default\n");
    printf("-----\n");
    printf("-W Number of Warehouses to Load           Required
\n");
    printf("-S Server                               %s\n",
SERVER);
    printf("-U Username                               %s\n",
USER);
    printf("-P Password                               %s\n",
PASSWORD);
    printf("-D Database                               %s\n",
DATABASE);
    printf("-b Batch Size
%ld\n", (long) BATCH);
}

```

```

    printf("-p TDS packet size
%ld\n", (long) DEFLDPACKSIZE);
    printf("-f Loader Results Output Filename
%s\n", LOADER_RES_FILE);
    printf("-s Starting Warehouse
%ld\n", (long) DEF_STARTING_WAREHOUSE);
    printf("-i Build Option (data = 0, data and index = 1)
%ld\n", (long) BUILD_INDEX);
    printf("-o Cluster Index Build Order (before = 1, after = 0)
%ld\n", (long) INDEX_ORDER);
    printf("-c Build Scaled Database (normal = 0, tiny = 1)
%ld\n", (long) SCALE_DOWN);
    printf("-d Index Script Path
%s\n", INDEX_SCRIPT_PATH);
    printf("-t Table to Load          all
tables \n");
    printf("    [item|warehouse|customer|orders]\n");
    printf("    Notes: \n");
    printf("    - the '-t' parameter may be included multiple times to
\n");
    printf("    specify multiple tables to be loaded \n");
    printf("    - 'item' loads ITEM table \n");
    printf("    - 'warehouse' loads WAREHOUSE, DISTRICT, and STOCK tables
\n");
    printf("    - 'customer' loads CUSTOMER and HISTORY tables \n");
    printf("    - 'orders' load NEW-ORDER, ORDERS, ORDER-LINE tables
\n");

    printf("\nNote: Command line switches are case sensitive.\n");

    exit(0);
}

```

RANDOM.C

```

// File:          RANDOM.C
//               Microsoft TPC-C Kit Ver. 4.00
//               Copyright Microsoft, 1996, 1997, 1998
// Purpose:       Random number generation routines for database
loader

// Includes
#include "tpcc.h"
#include "math.h"

// Defines
#define A          16807
#define M          2147483647
#define Q          127773      /* M div A */
#define R          2836       /* M mod A */
#define Thread    __declspec(thread)

// Globals
long  Thread Seed = 0;      /* thread local seed */

/*****
*****

```

```

*
*
* random -
*
* Implements a GOOD pseudo random number generator. This generator
*
* will/should? run the complete period before repeating.
*
*
* Copied from:
*
* Random Numbers Generators: Good Ones Are Hard to Find.
*
* Communications of the ACM - October 1988 Volume 31 Number 10
*
*
* Machine Dependencies:
*
* long must be 2 ^ 31 - 1 or greater.
*
*
* ****/
/*****/
* seed - load the Seed value used in irand and drand. Should be used
before *
* first call to irand or drand.
*
* ****/
*****/

void seed(long val)
{
#ifdef DEBUG
    printf("[%ld]DBG: Entering seed()..\n", (int) GetCurrentThreadId());
    printf("Old Seed %ld New Seed %ld\n",Seed, val);
#endif

    if ( val < 0 )
        val = abs(val);

    Seed = val;
}

/*****/
*****/
*
*
* irand - returns a 32 bit integer pseudo random number with a period of
*
* 1 to 2 ^ 32 - 1.
*

```



```

*
*
* parameters:
*
*     none.
*
*
*
* returns:
*
*     32 bit integer - defined as long ( see above ).
*
*
* side effects:
*
*     seed get recomputed.
*
*****
***/
long irand()
{
    register long    s;        /* copy of seed */
    register long    test;    /* test flag */
    register long    hi;      /* tmp value for speed */
    register long    lo;      /* tmp value for speed */

#ifdef DEBUG
    printf("[%ld]DBG: Entering irand()...\n", (int) GetCurrentThreadId());
#endif

    s = Seed;
    hi = s / Q;
    lo = s % Q;

    test = A * lo - R * hi;
    if ( test > 0 )
        Seed = test;
    else
        Seed = test + M;

    return( Seed );
}

/*****
*
* drand - returns a double pseudo random number between 0.0 and 1.0.
*
*     See irand.
*
*****
***/
double drand()
{
#ifdef DEBUG
    printf("[%ld]DBG: Entering drand()...\n", (int) GetCurrentThreadId());

```

```

#endif
    return( (double)irand() / 2147483647.0);
}

//=====
// Function   : RandomNumber
//
// Description:
//=====
long RandomNumber(long lower, long upper)
{
    long rand_num;

#ifdef DEBUG
    printf("[%ld]DBG: Entering RandomNumber()...\n", (int)
GetCurrentThreadId());
#endif

    if ( upper == lower ) /* pgd 08-13-96 perf enhancement */
        return lower;

    upper++;

    if ( upper <= lower )
        rand_num = upper;
    else
        rand_num = lower + irand() % (upper - lower); /* pgd 08-13-
96 perf enhancement */

#ifdef DEBUG
    printf("[%ld]DBG: RandomNumber between %ld & %ld ==> %ld\n",
(int) GetCurrentThreadId(), lower, upper,
rand_num);
#endif

    return rand_num;
}

#if 0
//Original code pgd 08/13/96
long RandomNumber(long lower,
long upper)
{
    long rand_num;

#ifdef DEBUG
    printf("[%ld]DBG: Entering RandomNumber()...\n", (int)
GetCurrentThreadId());
#endif

    upper++;

    if ((upper <= lower))
        rand_num = upper;

```

```

else
    rand_num = lower + irand() % ((upper > lower) ? upper -
lower : upper);

#ifdef DEBUG
    printf("[%ld]DBG: RandomNumber between %ld & %ld ==> %ld\n",
           (int) GetCurrentThreadId(), lower, upper,
rand_num);
#endif

    return rand_num;
}
#endif

//=====
// Function : NURand
//
// Description:
//=====
long NURand(int iConst,
           long x,
           long y,
           long C)
{
    long rand_num;

#ifdef DEBUG
    printf("[%ld]DBG: Entering NURand()...\n", (int)
GetCurrentThreadId());
#endif

    rand_num = (((RandomNumber(0,iConst) | RandomNumber(x,y)) + C) % (y-
x+1))+x;

#ifdef DEBUG
    printf("[%ld]DBG: NURand: num = %d\n", (int) GetCurrentThreadId(),
rand_num);
#endif

    return rand_num;
}

```

STRINGS.C

```

// File: STRINGS.C
// Microsoft TPC-C Kit Ver. 4.00
// Copyright Microsoft, 1996, 1997, 1998
// Purpose: Source file for database loader string functions

// Includes
#include "tpcc.h"
#include <string.h>
#include <ctype.h>

//=====

```

```

//
// Function name: MakeAddress
//
//=====
void MakeAddress(char *street_1,
                char *street_2,
                char *city,
                char *state,
                char *zip)
{
#ifdef DEBUG
    printf("[%ld]DBG: Entering MakeAddress()\n", (int)
GetCurrentThreadId());
#endif

    MakeAlphaString (10, 20, ADDRESS_LEN, street_1);
    MakeAlphaString (10, 20, ADDRESS_LEN, street_2);
    MakeAlphaString (10, 20, ADDRESS_LEN, city);
    MakeAlphaString ( 2,  2, STATE_LEN, state);
    MakeZipNumberString( 9,  9, ZIP_LEN, zip);

#ifdef DEBUG
    printf("[%ld]DBG: MakeAddress: street_1: %s, street_2: %s, city: %s,
state: %s, zip: %s\n",
           (int) GetCurrentThreadId(), street_1, street_2,
city, state, zip);
#endif

    return;
}

//=====
//
// Function name: LastName
//
//=====
void LastName(int num,
             char *name)
{
    static char *n[] =
    {
        "BAR" , "OUGHT", "ABLE" , "PRI" , "PRES",
        "ESE" , "ANTI" , "CALLY", "ATION", "EING"
    };

#ifdef DEBUG
    printf("[%ld]DBG: Entering LastName()\n", (int) GetCurrentThreadId());
#endif

    if ((num >= 0) && (num < 1000))
    {
        strcpy(name, n[(num/100)%10]);
        strcat(name, n[(num/10)%10]);
        strcat(name, n[(num/1)%10]);

        if (strlen(name) < LAST_NAME_LEN)

```

```

        {
            PaddString(LAST_NAME_LEN, name);
        }
    }
    else
    {
        printf("\nError in LastName()... num <%ld> out of range
(0,999)\n", num);
        exit(-1);
    }

#ifdef DEBUG
    printf("[%ld]DBG: LastName: num = [%d] ==> [%d] [%d] [%d]\n",
           (int) GetCurrentThreadId(), num, num/100,
           (num/10)%10, num%10);
    printf("[%ld]DBG: LastName: String = %s\n", (int)
GetCurrentThreadId(), name);
#endif

    return;
}

//=====
//
// Function name: MakeAlphaString
//
//=====

//philipdu 08/13/96 Changed MakeAlphaString to use A-Z, a-z, and 0-9 in
//accordance with spec see below:
//The spec says:
//4.3.2.2 The notation random a-string [x .. y]
//(respectively, n-string [x .. y]) represents a string of random
alphanumeric
//(respectively, numeric) characters of a random length of minimum x,
maximum y,
//and mean (y+x)/2. Alphanumerics are A..Z, a..z, and 0..9. The only
other
//requirement is that the character set used "must be able to represent a
minimum
//of 128 different characters". We are using 8-bit chars, so this is a
non issue.
//It is completely unreasonable to stuff non-printing chars into the text
fields.
//-CLevine 08/13/96

int MakeAlphaString( int x, int y, int z, char *str)
{
    int len;
    int i;
    static char chArray[] =
"0123456789ABCDEFGHIJKLMNQRSTUUVWXYZabcdefghijklmnopqrstuvwxyz";
    static int chArrayMax = 61;

#ifdef DEBUG
    printf("[%ld]DBG: Entering MakeAlphaString()\n", (int)
GetCurrentThreadId());

```

```

#endif

    len= RandomNumber(x, y);

    for (i=0; i<len; i++)
        str[i] = chArray[RandomNumber(0, chArrayMax)];
    if ( len < z )
        memset(str+len, ' ', z - len);
    str[len] = 0;

    return len;
}

//=====
//
// Function name: MakeOriginalAlphaString
//
//=====

int MakeOriginalAlphaString(int x,
                            int y,
                            int z,
                            char *str,
                            int percent)
{
    int len;
    int val;
    int start;

#ifdef DEBUG
    printf("[%ld]DBG: Entering MakeOriginalAlphaString()\n", (int)
GetCurrentThreadId());
#endif

    // verify percentage is valid
    if ((percent < 0) || (percent > 100))
    {
        printf("MakeOriginalAlphaString: Invalid percentage: %d\n",
percent);
        exit(-1);
    }

    // verify string is at least 8 chars in length
    if ((x + y) <= 8)
    {
        printf("MakeOriginalAlphaString: string length must be >=
8\n");
        exit(-1);
    }

    // Make Alpha String
    len = MakeAlphaString(x,y, z, str);

    val = RandomNumber(1,100);
    if (val <= percent)
    {
        start = RandomNumber(0, len - 8);
        strncpy(str + start, "ORIGINAL", 8);
    }
}

```

```

#ifdef DEBUG
    printf("[%ld]DBG: MakeOriginalAlphaString: : %s\n",
           (int) GetCurrentThreadId(), str);
#endif

    return strlen(str);
}

//=====
//
// Function name: MakeNumberString
//
//=====
int MakeNumberString(int x, int y, int z, char *str)
{
    char tmp[16];

    //MakeNumberString is always called MakeZipNumberString(16, 16,
    16, string)

    memset(str, '0', 16);
    itoa(RandomNumber(0, 99999999), tmp, 10);
    memcpy(str, tmp, strlen(tmp));

    itoa(RandomNumber(0, 99999999), tmp, 10);
    memcpy(str+8, tmp, strlen(tmp));

    str[16] = 0;

    return 16;
}

//=====
//
// Function name: MakeZipNumberString
//
//=====
int MakeZipNumberString(int x, int y, int z, char *str)
{
    char tmp[16];

    //MakeZipNumberString is always called MakeZipNumberString(9, 9,
    9, string)

    strcpy(str, "000011111");

    itoa(RandomNumber(0, 9999), tmp, 10);
    memcpy(str, tmp, strlen(tmp));

    return 9;
}

//=====
//
// Function name: InitString
//
//=====

```

```

void InitString(char *str, int len)
{
#ifdef DEBUG
    printf("[%ld]DBG: Entering InitString()\n", (int)
    GetCurrentThreadId());
#endif

    memset(str, ' ', len);
    str[len] = 0;
}

//=====
// Function name: InitAddress
//
// Description:
//
//=====
void InitAddress(char *street_1, char *street_2, char *city, char *state,
char *zip)
{
    memset(street_1, ' ', ADDRESS_LEN+1);
    memset(street_2, ' ', ADDRESS_LEN+1);
    memset(city, ' ', ADDRESS_LEN+1);

    street_1[ADDRESS_LEN+1] = 0;
    street_2[ADDRESS_LEN+1] = 0;
    city[ADDRESS_LEN+1] = 0;

    memset(state, ' ', STATE_LEN+1);
    state[STATE_LEN+1] = 0;

    memset(zip, ' ', ZIP_LEN+1);
    zip[ZIP_LEN+1] = 0;
}

//=====
//
// Function name: PaddString
//
//=====
void PaddString(int max, char *name)
{
    int len;

    len = strlen(name);
    if ( len < max )
        memset(name+len, ' ', max - len);
    name[max] = 0;

    return;
}

//
// File: TIME.C
// Microsoft TPC-C Kit Ver. 4.00

```

```

//          Copyright Microsoft, 1996, 1997, 1998
// Purpose:  Source file for time functions

// Includes
#include "tpcc.h"

// Globals
static long start_sec;

//=====
//
// Function name: TimeNow
//
//=====

long TimeNow()
{
    long    time_now;
    struct  _timeb el_time;

#ifdef DEBUG
    printf("[%ld]DBG: Entering TimeNow()\n", (int) GetCurrentThreadId());
#endif

    _ftime(&el_time);

    time_now = ((el_time.time - start_sec) * 1000) + el_time.millitm;

    return time_now;
}

```

TPCC.H

```

// File:      TPCC.H
//           Microsoft TPC-C Kit Ver. 4.00
//           Copyright Microsoft, 1996, 1997, 1998
// Purpose:   Header file for TPC-C database loader

// Build number of TPC Benchmark Kit
#define TPCKIT_VER    "4.00"

// General headers
#include <windows.h>
#include <winbase.h>
#include <stdlib.h>
#include <stdio.h>
#include <process.h>
#include <stddef.h>
#include <stdarg.h>
#include <string.h>
#include <time.h>
#include <sys\timeb.h>
#include <sys\types.h>

// ODBC headers

```

```

#include <sql.h>
#include <sqlext.h>
#include <odbcss.h>

// General constants
#define MILLI          1000
#define FALSE          0
#define TRUE           1
#define UNDEF          -1
#define MINPRINTASCII 32
#define MAXPRINTASCII 126

// Default environment constants
#define SERVER          ""
#define DATABASE        "tpcc"
#define USER            "sa"
#define PASSWORD        ""

// Default loader arguments
#define BATCH           10000
#define DEFLOADPACKSIZE 32768
#define ORDERS_PER_DIST 3000
#define LOADER_RES_FILE "logs\\load.out"
#define LOADER_NURAND_C 123
#define DEF_STARTING_WAREHOUSE 1
#define BUILD_INDEX    1 // build both
data and indexes
#define INDEX_ORDER    1 // build
indexes before load
#define SCALE_DOWN     0 // build a normal
scale database
#define INDEX_SCRIPT_PATH "scripts"

typedef struct
{
    char    *server;
    char    *database;
    char    *user;
    char    *password;
    BOOL    tables_all; // set
if loading all tables
    BOOL    table_item; // set
if loading ITEM table specifically
    BOOL    table_warehouse; // set if
loading WAREHOUSE, DISTRICT, and STOCK
    BOOL    table_customer; // set
if loading CUSTOMER and HISTORY
    BOOL    table_orders; // set if
loading NEW-ORDER, ORDERS, ORDER-LINE
    long    num_warehouses;
    long    batch;
    long    verbose;
    long    pack_size;
    char    *loader_res_file;
    char    *synch_servername;
    long    case_sensitivity;
    long    starting_warehouse;
    long    build_index;
    long    index_order;
    long    scale_down;
    char    *index_script_path;
}

```

```

} TPCCLDR_ARGS;

// String length constants
#define SERVER_NAME_LEN      20
#define DATABASE_NAME_LEN   20
#define USER_NAME_LEN       20
#define PASSWORD_LEN        20
#define TABLE_NAME_LEN    20
#define I_DATA_LEN          50
#define I_NAME_LEN          24
#define BRAND_LEN           1
#define LAST_NAME_LEN       16
#define W_NAME_LEN          10
#define ADDRESS_LEN         20
#define STATE_LEN           2
#define ZIP_LEN              9
#define S_DIST_LEN          24
#define S_DATA_LEN          50
#define D_NAME_LEN          10
#define FIRST_NAME_LEN      16
#define MIDDLE_NAME_LEN     2
#define PHONE_LEN           16
#define CREDIT_LEN          2
#define C_DATA_LEN          500
#define H_DATA_LEN          24
#define DIST_INFO_LEN       24
#define MAX_OL_NEW_ORDER_ITEMS 15
#define MAX_OL_ORDER_STATUS_ITEMS 15
#define STATUS_LEN          25
#define OL_DIST_INFO_LEN    24
#define C_SINCE_LEN         23
#define H_DATE_LEN          23
#define OL_DELIVERY_D_LEN   23
#define O_ENTRY_D_LEN       23

// Functions in random.c
void seed();
long irand();
double drand();
void WUCreate();
short WURand();
long RandomNumber(long lower, long upper);

// Functions in getargs.c;
void GetArgsLoader();
void GetArgsLoaderUsage();

// Functions in time.c
long TimeNow();

// Functions in strings.c
void MakeAddress();
void LastName();
int MakeAlphaString();
int MakeOriginalAlphaString();
int MakeNumberString();
int MakeZipNumberString();
void InitString();
void InitAddress();
void PaddString();

```

TPCCLDR.C

```

// File:          TPCCLDR.C
//               Microsoft TPC-C Kit Ver. 4.00
//               Copyright Microsoft, 1996, 1997, 1998
// Purpose:       Source file for TPC-C database loader

// Includes
#include "tpcc.h"
#include "search.h"

// Defines
#define MAXITEMS          100000
#define MAXITEMS_SCALE_DOWN 100
#define CUSTOMERS_PER_DISTRICT 3000
#define CUSTOMERS_SCALE_DOWN 30
#define DISTRICT_PER_WAREHOUSE 10
#define ORDERS_PER_DISTRICT 3000
#define ORDERS_SCALE_DOWN 30
#define MAX_CUSTOMER_THREADS 2
#define MAX_ORDER_THREADS 3
#define MAX_MAIN_THREADS 4

// Functions declarations
void HandleErrorDBC (SQLHDBC hdbc1);

long NURand();
void LoadItem();
void LoadWarehouse();

void Stock();
void District();

void LoadCustomer();
void CustomerBufInit();
void CustomerBufLoad();
void LoadCustomerTable();
void LoadHistoryTable();

void LoadOrders();
void OrdersBufInit();
void OrdersBufLoad();
void LoadOrdersTable();
void LoadNewOrderTable();
void LoadOrderLineTable();
void GetPermutation();
void CheckForCommit();
void OpenConnections();
void BuildIndex();
void FormatDate ();

// Shared memory structures
typedef struct
{
    long ol;
    long ol_i_id;
    short ol_supply_w_id;
}

```

```

short      ol_quantity;
double     ol_amount;
char       ol_dist_info[DIST_INFO_LEN+1];
char       ol_delivery_d[OL_DELIVERY_D_LEN+1];
} ORDER_LINE_STRUCT;

typedef struct
{
    long      o_id;
    short     o_d_id;
    short     o_w_id;
    long      o_c_id;
    short     o_carrier_id;
    short     o_ol_cnt;
    short     o_all_local;
    ORDER_LINE_STRUCT o_ol[15];
} ORDERS_STRUCT;

typedef struct
{
    long      c_id;
    short     c_d_id;
    short     c_w_id;
    char      c_first[FIRST_NAME_LEN+1];
    char      c_middle[MIDDLE_NAME_LEN+1];
    char      c_last[LAST_NAME_LEN+1];
    char      c_street_1[ADDRESS_LEN+1];
    char      c_street_2[ADDRESS_LEN+1];
    char      c_city[ADDRESS_LEN+1];
    char      c_state[STATE_LEN+1];
    char      c_zip[ZIP_LEN+1];
    char      c_phone[PHONE_LEN+1];
    char      c_credit[CREDIT_LEN+1];
    double    c_credit_lim;
    double    c_discount;
// fix to avoid ODBC float to numeric conversion problem.
// double    c_balance;
    char      c_balance[6];

    double    c_ytd_payment;
    short     c_payment_cnt;
    short     c_delivery_cnt;
    char      c_data[C_DATA_LEN+1];
    double    h_amount;
    char      h_data[H_DATA_LEN+1];
} CUSTOMER_STRUCT;

typedef struct
{
    char      c_last[LAST_NAME_LEN+1];
    char      c_first[FIRST_NAME_LEN+1];
    long      c_id;
} CUSTOMER_SORT_STRUCT;

typedef struct
{
    long      time_start;
} LOADER_TIME_STRUCT;

// Global variables

```

```

char       szLastError[300];
HENV       henv;

HDBC       i_hdbc1;           // for ITEM table
HDBC       w_hdbc1;           // for WAREHOUSE,
DISTRICT, STOCK
HDBC       c_hdbc1;           // for CUSTOMER
HDBC       c_hdbc2;           // for HISTORY
HDBC       o_hdbc1;           // for ORDERS
HDBC       o_hdbc2;           // for NEW-ORDER

HDBC       o_hdbc3;           // for ORDER-LINE

HSTMT      i_hstmt1;
HSTMT      w_hstmt1;
HSTMT      c_hstmt1, c_hstmt2;
HSTMT      o_hstmt1, o_hstmt2, o_hstmt3;

ORDERS_STRUCT orders_buf[ORDERS_PER_DISTRICT];
CUSTOMER_STRUCT customer_buf[CUSTOMERS_PER_DISTRICT];
long         orders_rows_loaded;
long         new_order_rows_loaded;
long         order_line_rows_loaded;
long         history_rows_loaded;
long         customer_rows_loaded;
long         stock_rows_loaded;
long         district_rows_loaded;
long         item_rows_loaded;
long         warehouse_rows_loaded;
long         main_time_start;
long         main_time_end;
long         max_items;
long         customers_per_district;
long         orders_per_district;
long         first_new_order;
long         last_new_order;

TPCCCLR_ARGS *aptr, args;

//=====
//
// Function name: main
//
//=====

int main(int argc, char **argv)
{
    DWORD     dwThreadId[MAX_MAIN_THREADS];
    HANDLE    hThread[MAX_MAIN_THREADS];
    FILE      *fLoader;
    char      buffer[255];
    int       i;

    for (i=0; i<MAX_MAIN_THREADS; i++)
        hThread[i] = NULL;

```

```

printf("\n*****");
printf("\n*                               *");
printf("\n* Microsoft SQL Server             *");
printf("\n*                               *");
printf("\n* TPC-C BENCHMARK KIT: Database loader *");
printf("\n* Version %s                        *",
TPCKIT_VER);
printf("\n*                               *");
printf("\n*****\n\n");
);

// process command line arguments

aptr = &args;
GetArgsLoader(argc, argv, aptr);

printf("Build interface is ODBC.\n");

if (aptr->build_index == 0)
    printf("Data load only - no index creation.\n");
else
    printf("Data load and index creation.\n");

if (aptr->index_order == 0)
    printf("Clustered indexes will be created after bulk
load.\n");
else
    printf("Clustered indexes will be created before bulk
load.\n");

// set database scale values
if (aptr->scale_down == 1)
{
    printf("*** Scaled Down Database ***\n");
    max_items = MAXITEMS_SCALE_DOWN;
    customers_per_district = CUSTOMERS_SCALE_DOWN;
    orders_per_district = ORDERS_SCALE_DOWN;
    first_new_order = 0;
    last_new_order = 30;
}
else
{
    max_items = MAXITEMS;
    customers_per_district = CUSTOMERS_PER_DISTRICT;
    orders_per_district = ORDERS_PER_DISTRICT;
    first_new_order = 2100;
    last_new_order = 3000;
}

// open connections to SQL Server
OpenConnections();

// open file for loader results
fLoader = fopen(aptr->loader_res_file, "w");

if (fLoader == NULL)
{
    printf("Error, loader result file open failed.");
    exit(-1);
}

```

```

// start loading data

sprintf(buffer, "TPC-C load started for %ld warehouses.\n", aptr-
>num_warehouses);

printf("%s", buffer);
fprintf(fLoader, "%s", buffer);

main_time_start = (TimeNow() / MILLI);

// start parallel load threads

if (aptr->tables_all || aptr->table_item)
{
    fprintf(fLoader, "\nStarting loader threads for: item\n");
    hThread[0] = CreateThread(NULL,
                                0,
(LPTHREAD_START_ROUTINE) LoadItem,
                                NULL,
                                0,
&dwThreadID[0]);
    if (hThread[0] == NULL)
    {
        printf("Error, failed in creating creating thread =
0.\n");
        exit(-1);
    }
    if (aptr->tables_all || aptr->table_warehouse)
    {
        fprintf(fLoader, "Starting loader threads for:
warehouse\n");
        hThread[1] = CreateThread(NULL,
                                    0,
(LPTHREAD_START_ROUTINE) LoadWarehouse,
                                    NULL,
                                    0,
&dwThreadID[1]);
        if (hThread[1] == NULL)
        {
            printf("Error, failed in creating creating thread =
1.\n");
            exit(-1);
        }
        if (aptr->tables_all || aptr->table_customer)
        {
            fprintf(fLoader, "Starting loader threads for:
customer\n");

```



```

        hThread[2] = CreateThread(NULL,
                                0,
(LPTHREAD_START_ROUTINE) LoadCustomer,
                                NULL,
                                0,
&dwThreadID[2]);
        if (hThread[2] == NULL)
        {
            printf("Error, failed in creating creating main
thread = 2.\n");
            exit(-1);
        }
        if (aptr->tables_all || aptr->table_orders)
        {
            fprintf(fLoader, "Starting loader threads for: orders\n");
            hThread[3] = CreateThread(NULL,
                                0,
(LPTHREAD_START_ROUTINE) LoadOrders,
                                NULL,
                                0,
&dwThreadID[3]);
            if (hThread[3] == NULL)
            {
                printf("Error, failed in creating creating main
thread = 3.\n");
                exit(-1);
            }
            // Wait for threads to finish...
            for (i=0; i<MAX_MAIN_THREADS; i++)
            {
                if (hThread[i] != NULL)
                {
                    WaitForSingleObject( hThread[i], INFINITE );
                    CloseHandle(hThread[i]);
                    hThread[i] = NULL;
                }
            }
            main_time_end = (TimeNow() / MILLI);
            sprintf(buffer, "\nTPC-C load completed successfully in %ld
minutes.\n",
                    (main_time_end - main_time_start)/60);
            printf("%s",buffer);
            fprintf(fLoader, "%s", buffer);
            fclose(fLoader);

```

```

        SQLFreeEnv(henv);
        exit(0);
        return 0;
    }
    //=====
    //
    // Function name: LoadItem
    //
    //=====
void LoadItem()
{
    long          i_id;
    long          i_im_id;
    char          i_name[I_NAME_LEN+1];
    double        i_price;
    char          i_data[I_DATA_LEN+1];
    char          name[20];
    long          time_start;
    RETCODE       rc;
    DBINT         rcint;
    char          bcphint[128];

    // Seed with unique number
    seed(1);

    printf("Loading item table...\n");

    // if build index before load
    if ((aptr->build_index == 1) && (aptr->index_order == 1))
        BuildIndex("idxitmcl");

    InitString(i_name, I_NAME_LEN+1);
    InitString(i_data, I_DATA_LEN+1);

    sprintf(name, "%s..%s", aptr->database, "item");

    rc = bcp_init(i_hdbc1, name, NULL, "logs\\item.err", DB_IN);
    if (rc != SUCCEED)
        HandleErrorDBC(i_hdbc1);

    if ((aptr->build_index == 1) && (aptr->index_order == 1))
    {
        sprintf(bcphint, "tablock, order (i_id), ROWS_PER_BATCH =
100000");
        rc = bcp_control(i_hdbc1, BCPHINTS, (void*) bcphint);
        if (rc != SUCCEED)
            HandleErrorDBC(i_hdbc1);
    }

    rc = bcp_bind(i_hdbc1, (BYTE *) &i_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT4, 1);
    if (rc != SUCCEED)
        HandleErrorDBC(i_hdbc1);

    rc = bcp_bind(i_hdbc1, (BYTE *) &i_im_id, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT4, 2);

```

```

    if (rc != SUCCEED)
        HandleErrorDBC(i_hdbc1);

    rc = bcp_bind(i_hdbc1, (BYTE *) i_name, 0, I_NAME_LEN, NULL, 0, 0,
3);
    if (rc != SUCCEED)
        HandleErrorDBC(i_hdbc1);

    rc = bcp_bind(i_hdbc1, (BYTE *) &i_price, 0, SQL_VARLEN_DATA,
NULL, 0, SQLFLT8, 4);
    if (rc != SUCCEED)
        HandleErrorDBC(i_hdbc1);

    rc = bcp_bind(i_hdbc1, (BYTE *) i_data, 0, I_DATA_LEN, NULL, 0, 0,
5);
    if (rc != SUCCEED)
        HandleErrorDBC(i_hdbc1);

    time_start = (TimeNow() / MILLI);

    item_rows_loaded = 0;

    for (i_id = 1; i_id <= max_items; i_id++)
    {
        i_im_id = RandomNumber(1L, 10000L);

        MakeAlphaString(14, 24, I_NAME_LEN, i_name);

        i_price = ((float) RandomNumber(100L, 10000L))/100.0;

        MakeOriginalAlphaString(26, 50, I_DATA_LEN, i_data, 10);

        rc = bcp_sendrow(i_hdbc1);
        if (rc != SUCCEED)
            HandleErrorDBC(i_hdbc1);

        item_rows_loaded++;
        CheckForCommit(i_hdbc1, i_hstmt1, item_rows_loaded, "item",
&time_start);
    }

    rcint = bcp_done(i_hdbc1);
    if (rcint < 0)
        HandleErrorDBC(i_hdbc1);

    printf("Finished loading item table.\n");

    SQLFreeStmt(i_hstmt1, SQL_DROP);
    SQLDisconnect(i_hdbc1);
    SQLFreeConnect(i_hdbc1);

    // if build index after load
    if ((aptr->build_index == 1) && (aptr->index_order == 0))
        BuildIndex("idxitmcl");
}

//=====
//

```

```

// Function : LoadWarehouse
//
// Loads WAREHOUSE table and loads Stock and District as Warehouses are
// created
//
//=====
void LoadWarehouse()
{
    short    w_id;
    char     w_name[W_NAME_LEN+1];
    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];
    double   w_tax;
    double   w_ytd;
    char     name[20];
    long     time_start;
    RETCODE rc;
    DBINT    rcint;
    char     bcphint[128];

    // Seed with unique number
    seed(2);

    printf("Loading warehouse table...\n");

    // if build index before load...
    if ((aptr->build_index == 1) && (aptr->index_order == 1))
        BuildIndex("idxwarcl");

    InitString(w_name, W_NAME_LEN+1);
    InitAddress(w_street_1, w_street_2, w_city, w_state, w_zip);

    sprintf(name, "%s..%s", aptr->database, "warehouse");

    rc = bcp_init(w_hdbc1, name, NULL, "logs\\whouse.err", DB_IN);
    if (rc != SUCCEED)
        HandleErrorDBC(w_hdbc1);

    if ((aptr->build_index == 1) && (aptr->index_order == 1))
    {
        sprintf(bcphint, "tablock, order (w_id), ROWS_PER_BATCH =
%d", aptr->num_warehouses);
        rc = bcp_control(w_hdbc1, BCPHINTS, (void*) bcphint);
        if (rc != SUCCEED)
            HandleErrorDBC(w_hdbc1);
    }

    rc = bcp_bind(w_hdbc1, (BYTE *) &w_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 1);
    if (rc != SUCCEED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) w_name, 0, W_NAME_LEN, NULL, 0, 0,
2);
    if (rc != SUCCEED)

```

```

        HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) w_street_1, 0, ADDRESS_LEN, NULL,
0, 0, 3);
        if (rc != SUCCEEDED)
            HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) w_street_2, 0, ADDRESS_LEN, NULL,
0, 0, 4);
        if (rc != SUCCEEDED)
            HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) w_city, 0, ADDRESS_LEN, NULL, 0,
0, 5);
        if (rc != SUCCEEDED)
            HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) w_state, 0, STATE_LEN, NULL, 0, 0,
6);
        if (rc != SUCCEEDED)
            HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) w_zip, 0, ZIP_LEN, NULL, 0, 0, 7);
        if (rc != SUCCEEDED)
            HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) &w_tax, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 8);
        if (rc != SUCCEEDED)
            HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) &w_ytd, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 9);
        if (rc != SUCCEEDED)
            HandleErrorDBC(w_hdbc1);

        time_start = (TimeNow() / MILLI);

        warehouse_rows_loaded = 0;

        for (w_id = (short)aptr->starting_warehouse; w_id <= aptr-
>num_warehouses; w_id++)
        {
            MakeAlphaString(6,10, W_NAME_LEN, w_name);

            MakeAddress(w_street_1, w_street_2, w_city, w_state,
w_zip);

            w_tax = ((float) RandomNumber(0L,2000L))/10000.00;

            w_ytd = 300000.00;

            rc = bcp_sendrow(w_hdbc1);
            if (rc != SUCCEEDED)
                HandleErrorDBC(w_hdbc1);

            warehouse_rows_loaded++;
            CheckForCommit(w_hdbc1, i_hstmt1, warehouse_rows_loaded,
"warehouse", &time_start);
        }

```

```

        rcint = bcp_done(w_hdbc1);
        if (rcint < 0)
            HandleErrorDBC(w_hdbc1);

        printf("Finished loading warehouse table.\n");

        // if build index after load...
        if ((aptr->build_index == 1) && (aptr->index_order == 0))
            BuildIndex("idxwarcl");

        stock_rows_loaded = 0;
        district_rows_loaded = 0;

        District();
        Stock();
    }

//=====
//
// Function    : District
//
//=====

void District()
{
    short    d_id;
    short    d_w_id;
    char     d_name[D_NAME_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];
    double   d_tax;
    double   d_ytd;
    char     name[20];
    long     d_next_o_id;
    long     time_start;
    int      w_id;
    RETCODE rc;
    DBINT    rcint;
    char     bcphint[128];

    // Seed with unique number
    seed(4);

    printf("Loading district table...\n");

    // build index before load
    if ((aptr->build_index == 1) && (aptr->index_order == 1))
        BuildIndex("idxdiscl");

    InitString(d_name, D_NAME_LEN+1);
    InitAddress(d_street_1, d_street_2, d_city, d_state, d_zip);
    sprintf(name, "%s..%s", aptr->database, "district");

    rc = bcp_init(w_hdbc1, name, NULL, "logs\\district.err", DB_IN);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

```

```

    if ((aptr->build_index == 1) && (aptr->index_order == 1))
    {
        sprintf(bcphint, "tablock, order (d_w_id, d_id),
ROWS_PER_BATCH = %u", (aptr->num_warehouses * 10));
        rc = bcp_control(w_hdbc1, BCPHINTS, (void*) bcphint);
        if (rc != SUCCEEDED)
            HandleErrorDBC(w_hdbc1);
    }

    rc = bcp_bind(w_hdbc1, (BYTE *) &d_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 1);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) &d_w_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 2);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) d_name, 0, D_NAME_LEN, NULL, 0, 0,
3);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) d_street_1, 0, ADDRESS_LEN, NULL,
0, 0, 4);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) d_street_2, 0, ADDRESS_LEN, NULL,
0, 0, 5);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) d_city, 0, ADDRESS_LEN, NULL, 0,
0, 6);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) d_state, 0, STATE_LEN, NULL, 0, 0,
7);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) d_zip, 0, ZIP_LEN, NULL, 0, 0, 8);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) &d_tax, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 9);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) &d_ytd, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 10);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) &d_next_o_id, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT4, 11);

```

```

    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);

    d_ytd = 30000.0;

    d_next_o_id = orders_per_district+1;

    time_start = (TimeNow() / MILLI);

    for (w_id = aptr->starting_warehouse; w_id <= aptr-
>num_warehouses; w_id++)
    {
        d_w_id = w_id;

        for (d_id = 1; d_id <= DISTRICT_PER_WAREHOUSE; d_id++)
        {
            MakeAlphaString(6,10,D_NAME_LEN, d_name);

            MakeAddress(d_street_1, d_street_2, d_city,
d_state, d_zip);

            d_tax = ((float) RandomNumber(0L,2000L))/10000.00;

            rc = bcp_sendrow(w_hdbc1);
            if (rc != SUCCEEDED)
                HandleErrorDBC(w_hdbc1);

            district_rows_loaded++;
            CheckForCommit(w_hdbc1, w_hstmt1,
district_rows_loaded, "district", &time_start);
        }
    }

    rcint = bcp_done(w_hdbc1);
    if (rcint < 0)
        HandleErrorDBC(w_hdbc1);

    printf("Finished loading district table.\n");

    // if build index after load...
    if ((aptr->build_index == 1) && (aptr->index_order == 0))
        BuildIndex("idxdiscl");

    return;
}

//=====
//
// Function : Stock
//
//=====

void Stock()
{
    long      s_i_id;
    short     s_w_id;
    short     s_quantity;
    char      s_dist_01[S_DIST_LEN+1];
    char      s_dist_02[S_DIST_LEN+1];
    char      s_dist_03[S_DIST_LEN+1];

```

```

char      s_dist_04[S_DIST_LEN+1];
char      s_dist_05[S_DIST_LEN+1];
char      s_dist_06[S_DIST_LEN+1];
char      s_dist_07[S_DIST_LEN+1];
char      s_dist_08[S_DIST_LEN+1];
char      s_dist_09[S_DIST_LEN+1];
char      s_dist_10[S_DIST_LEN+1];
long      s_ytd;
short     s_order_cnt;
short     s_remote_cnt;
char      s_data[S_DATA_LEN+1];
short     len;
char      name[20];
long      time_start;
RETCODE rc;
DBINT     rcint;
char      bcphint[128];

// Seed with unique number
seed(3);

// if build index before load...
if ((aptr->build_index == 1) && (aptr->index_order == 1))
    BuildIndex("idxstkcl");

sprintf(name, "%s..%s", aptr->database, "stock");

rc = bcp_init(w_hdbc1, name, NULL, "logs\\stock.err", DB_IN);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

if ((aptr->build_index == 1) && (aptr->index_order == 1))
{
    sprintf(bcphint, "tablock, order (s_i_id, s_w_id),
ROWS_PER_BATCH = %u", (aptr->num_warehouses * 10000));
    rc = bcp_control(w_hdbc1, BCPHINTS, (void*) bcphint);
    if (rc != SUCCEEDED)
        HandleErrorDBC(w_hdbc1);
}

rc = bcp_bind(w_hdbc1, (BYTE *) &s_i_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT4, 1);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

    bcp_bind(w_hdbc1, (BYTE *) &s_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

    rc = bcp_bind(w_hdbc1, (BYTE *) &s_quantity, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 3);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_01, 0, S_DIST_LEN, NULL, 0,
0, 4);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

```

```

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_02, 0, S_DIST_LEN, NULL, 0,
0, 5);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_03, 0, S_DIST_LEN, NULL, 0,
0, 6);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_04, 0, S_DIST_LEN, NULL, 0,
0, 7);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_05, 0, S_DIST_LEN, NULL, 0,
0, 8);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_06, 0, S_DIST_LEN, NULL, 0,
0, 9);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_07, 0, S_DIST_LEN, NULL, 0,
0, 10);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_08, 0, S_DIST_LEN, NULL, 0,
0, 11);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_09, 0, S_DIST_LEN, NULL, 0,
0, 12);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_10, 0, S_DIST_LEN, NULL, 0,
0, 13);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) &s_ytd, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT4, 14);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) &s_order_cnt, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 15);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

rc = bcp_bind(w_hdbc1, (BYTE *) &s_remote_cnt, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 16);
if (rc != SUCCEEDED)
    HandleErrorDBC(w_hdbc1);

```

```

17); rc = bcp_bind(w_hdbc1, (BYTE *) s_data, 0, S_DATA_LEN, NULL, 0, 0,
if (rc != SUCCEED)
    HandleErrorDBC(w_hdbc1);

s_ytd = s_order_cnt = s_remote_cnt = 0;

time_start = (TimeNow() / MILLI);

printf("...Loading stock table\n");

for (s_i_id=1; s_i_id <= max_items; s_i_id++)
{
    for (s_w_id = (short)aptr->starting_warehouse; s_w_id <=
aptr->num_warehouses; s_w_id++)
    {
        s_quantity = (short)RandomNumber(10L,100L);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_01);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_02);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_03);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_04);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_05);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_06);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_07);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_08);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_09);
        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_10);

        len = MakeOriginalAlphaString(26,50, S_DATA_LEN,
s_data,10);

        rc = bcp_sendrow(w_hdbc1);
        if (rc != SUCCEED)
            HandleErrorDBC(w_hdbc1);

        stock_rows_loaded++;
        CheckForCommit(w_hdbc1, w_hstmt1,
stock_rows_loaded, "stock", &time_start);
    }
}

rcint = bcp_done(w_hdbc1);
if (rcint < 0)
    HandleErrorDBC(w_hdbc1);

printf("Finished loading stock table.\n");

SQLFreeStmt(w_hstmt1, SQL_DROP);
SQLDisconnect(w_hdbc1);
SQLFreeConnect(w_hdbc1);

// if build index after load...
if ((aptr->build_index == 1) && (aptr->index_order == 0))
    BuildIndex("idxstkcl");

return;
}

```

```

//=====
//
// Function : LoadCustomer
//
//=====

void LoadCustomer()
{
    LOADER_TIME_STRUCT customer_time_start;
    LOADER_TIME_STRUCT history_time_start;
    short w_id;
    short d_id;
    DWORD dwThreadId[MAX_CUSTOMER_THREADS];
    HANDLE hThread[MAX_CUSTOMER_THREADS];
    char name[20];
    RETCODE rc;
    DBINT rcint;
    char bcphint[128];
    char cmd[256];
    // SQLRETURN rc_1;
    // SQLSMALLINT recnum, MsgLen;
    // SQLCHAR SqlState[6],
Msg[SQL_MAX_MESSAGE_LENGTH];
    // SQLINTEGER NativeError;

    // Seed with unique number
    seed(5);

    printf("Loading customer and history tables...\n");

    // if build index before load...
    if ((aptr->build_index == 1) && (aptr->index_order == 1))
        BuildIndex("idxcuscl");

    // Initialize bulk copy
    sprintf(name, "%s..%s", aptr->database, "customer");

    rc = bcp_init(c_hdbc1, name, NULL, "logs\\customer.err", DB_IN);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

    if ((aptr->build_index == 1) && (aptr->index_order == 1))
    {
        sprintf(bcphint, "tablock, order (c_w_id, c_d_id, c_id),
ROWS_PER_BATCH = %u", (aptr->num_warehouses * 30000));
        rc = bcp_control(c_hdbc1, BCPHINTS, (void*) bcphint);
        if (rc != SUCCEED)
            HandleErrorDBC(c_hdbc1);
    }

    sprintf(name, "%s..%s", aptr->database, "history");

    rc = bcp_init(c_hdbc2, name, NULL, "logs\\history.err", DB_IN);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    sprintf(bcphint, "tablock");
    rc = bcp_control(c_hdbc2, BCPHINTS, (void*) bcphint);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);
}

```

```

customer_rows_loaded    = 0;
history_rows_loaded     = 0;

CustomerBufInit();

customer_time_start.time_start = (TimeNow() / MILLI);
history_time_start.time_start = (TimeNow() / MILLI);

for (w_id = (short)aptr->starting_warehouse; w_id <= aptr-
>num_warehouses; w_id++)
{
    for (d_id = 1; d_id <= DISTRICT_PER_WAREHOUSE; d_id++)
    {
        CustomerBufLoad(d_id, w_id);

        // Start parallel loading threads here...

        // Start customer table thread

        printf("...Loading customer table for: d_id = %d,
w_id = %d\n", d_id, w_id);

        hThread[0] = CreateThread(NULL,
                                0,
                                (LPTHREAD_START_ROUTINE) LoadCustomerTable,
                                &customer_time_start,
                                0,
                                &dwThreadID[0]);

        if (hThread[0] == NULL)
        {
            printf("Error, failed in creating creating
thread = 0.\n");
            exit(-1);
        }

        // Start History table thread

        printf("...Loading history table for: d_id = %d,
w_id = %d\n", d_id, w_id);

        hThread[1] = CreateThread(NULL,
                                0,
                                (LPTHREAD_START_ROUTINE) LoadHistoryTable,
                                &history_time_start,
                                0,
                                &dwThreadID[1]);

        if (hThread[1] == NULL)
        {
            printf("Error, failed in creating creating
thread = 1.\n");
            exit(-1);
        }
    }
}

```

```

    }

    WaitForSingleObject( hThread[0], INFINITE );
    WaitForSingleObject( hThread[1], INFINITE );

    if (CloseHandle(hThread[0]) == FALSE)
    {
        printf("Error, failed in closing customer
thread handle with errno: %d\n", GetLastError());
    }

    if (CloseHandle(hThread[1]) == FALSE)
    {
        printf("Error, failed in closing history
thread handle with errno: %d\n", GetLastError());
    }

}

// flush the bulk connection
rcint = bcp_done(c_hdbc1);
if (rcint < 0)
    HandleErrorDBC(c_hdbc1);

rcint = bcp_done(c_hdbc2);
if (rcint < 0)
    HandleErrorDBC(c_hdbc2);

printf("Finished loading customer table.\n");

// if build index after load...
if ((aptr->build_index == 1) && (aptr->index_order == 0))
    BuildIndex("idxcuscl");

// build non-clustered index
if (aptr->build_index == 1)
    BuildIndex("idxcusnc");

// Output the NURAND used for the loader into C_FIRST for C_ID =
1,
// C_W_ID = 1, and C_D_ID = 1
sprintf(cmd, "isql -S%s -U%s -P%s -d%s -e -Q\"update customer set
c_first = 'C_LOAD = %d' where c_id = 1 and c_w_id = 1 and c_d_id = 1\" >
logs\\nurand_load.log",
        aptr->server,
        aptr->user,
        aptr->password,
        aptr->database,
        LOADER_NURAND_C);

system(cmd);

SQLFreeStmt(c_hstmt1, SQL_DROP);
SQLDisconnect(c_hdbc1);
SQLFreeConnect(c_hdbc1);

SQLFreeStmt(c_hstmt2, SQL_DROP);
SQLDisconnect(c_hdbc2);

```

```

        SQLFreeConnect(c_hdbc2);

return;
}

//=====
//
// Function   : CustomerBufInit
//
//=====
void CustomerBufInit()
{
    int    i;

    for (i=0;i<customers_per_district;i++)
    {
        customer_buf[i].c_id = 0;
        customer_buf[i].c_d_id = 0;
        customer_buf[i].c_w_id = 0;

        strcpy(customer_buf[i].c_first,"");
        strcpy(customer_buf[i].c_middle,"");
        strcpy(customer_buf[i].c_last,"");
        strcpy(customer_buf[i].c_street_1,"");
        strcpy(customer_buf[i].c_street_2,"");
        strcpy(customer_buf[i].c_city,"");
        strcpy(customer_buf[i].c_state,"");
        strcpy(customer_buf[i].c_zip,"");
        strcpy(customer_buf[i].c_phone,"");
        strcpy(customer_buf[i].c_credit,"");

        customer_buf[i].c_credit_lim = 0;
        customer_buf[i].c_discount = (float) 0;

        // fix to avoid ODBC float to numeric conversion problem.
        // customer_buf[i].c_balance = 0;
        strcpy(customer_buf[i].c_balance,"");

        customer_buf[i].c_ytd_payment = 0;
        customer_buf[i].c_payment_cnt = 0;
        customer_buf[i].c_delivery_cnt = 0;

        strcpy(customer_buf[i].c_data,"");

        customer_buf[i].h_amount = 0;

        strcpy(customer_buf[i].h_data,"");
    }
}

//=====
//
// Function   : CustomerBufLoad

```

```

//
// Fills shared buffer for HISTORY and CUSTOMER
//=====
void CustomerBufLoad(int d_id, int w_id)
{
    long
    CUSTOMER_SORT_STRUCT    c[CUSTOMERS_PER_DISTRICT];

    for (i=0;i<customers_per_district;i++)
    {
        if (i < 1000)
            LastName(i, c[i].c_last);
        else
            LastName(NURand(255,0,999,LOADER_NURAND_C),
c[i].c_last);

        MakeAlphaString(8,16,FIRST_NAME_LEN, c[i].c_first);

        c[i].c_id = i+1;
    }

    printf("...Loading customer buffer for: d_id = %d, w_id = %d\n",
        d_id, w_id);

    for (i=0;i<customers_per_district;i++)
    {
        customer_buf[i].c_d_id = d_id;
        customer_buf[i].c_w_id = w_id;
        customer_buf[i].h_amount = 10.0;

        customer_buf[i].c_ytd_payment = 10.0;

        customer_buf[i].c_payment_cnt = 1;
        customer_buf[i].c_delivery_cnt = 0;

        // Generate CUSTOMER and HISTORY data

        customer_buf[i].c_id = c[i].c_id;

        strcpy(customer_buf[i].c_first, c[i].c_first);
        strcpy(customer_buf[i].c_last, c[i].c_last);

        customer_buf[i].c_middle[0] = 'O';
        customer_buf[i].c_middle[1] = 'E';

        MakeAddress(customer_buf[i].c_street_1,
                    customer_buf[i].c_street_2,
                    customer_buf[i].c_city,
                    customer_buf[i].c_state,
                    customer_buf[i].c_zip);

        MakeNumberString(16, 16, PHONE_LEN,
customer_buf[i].c_phone);

        if (RandomNumber(1L, 100L) > 10)
            customer_buf[i].c_credit[0] = 'G';
        else

```



```

        customer_buf[i].c_credit[0] = 'B';
        customer_buf[i].c_credit[1] = 'C';

        customer_buf[i].c_credit_lim = 50000.0;
        customer_buf[i].c_discount = ((float) RandomNumber(0L,
5000L)) / 10000.0;

        // fix to avoid ODBC float to numeric conversion problem.

        // customer_buf[i].c_balance = -10.0;
        strcpy(customer_buf[i].c_balance, "-10.0");

        MakeAlphaString(500, 500, C_DATA_LEN,
customer_buf[i].c_data);

        // Generate HISTORY data
        MakeAlphaString(12, 24, H_DATA_LEN,
customer_buf[i].h_data);
    }
}

//=====
//
// Function   : LoadCustomerTable
//
//=====

void LoadCustomerTable(LOADER_TIME_STRUCT *customer_time_start)
{
    int         i;
    long        c_id;
    short       c_d_id;
    short       c_w_id;
    char        c_first[FIRST_NAME_LEN+1];
    char        c_middle[MIDDLE_NAME_LEN+1];
    char        c_last[LAST_NAME_LEN+1];
    char        c_street_1[ADDRESS_LEN+1];
    char        c_street_2[ADDRESS_LEN+1];
    char        c_city[ADDRESS_LEN+1];
    char        c_state[STATE_LEN+1];
    char        c_zip[ZIP_LEN+1];
    char        c_phone[PHONE_LEN+1];
    char        c_credit[CREDIT_LEN+1];
    double      c_credit_lim;
    double      c_discount;

    // fix to avoid ODBC float to numeric conversion problem.

    // double      c_balance;
    char        c_balance[6];

    double      c_ytd_payment;
    short       c_payment_cnt;
    short       c_delivery_cnt;
    char        c_data[C_DATA_LEN+1];
    char        c_since[C_SINCE_LEN+1];
    RETCODE     rc;

    rc = bcp_bind(c_hdbc1, (BYTE *) &c_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);

```

```

        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) &c_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) &c_w_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 3);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) c_first, 0, FIRST_NAME_LEN, NULL, 0,
0, 4);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) c_middle, 0, MIDDLE_NAME_LEN, NULL, 0,
0, 5);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) c_last, 0, LAST_NAME_LEN, NULL, 0, 0,
6);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) c_street_1, 0, ADDRESS_LEN, NULL, 0,
0, 7);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) c_street_2, 0, ADDRESS_LEN, NULL, 0, 0,
8);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) c_city, 0, ADDRESS_LEN, NULL, 0, 0,
9);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) c_state, 0, STATE_LEN, NULL, 0, 0,
10);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) c_zip, 0, ZIP_LEN, NULL, 0, 0, 11);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) c_phone, 0, PHONE_LEN, NULL, 0, 0,
12);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

        rc = bcp_bind(c_hdbc1, (BYTE *) &c_since, 0, C_SINCE_LEN, NULL, 0,
SQLCHARACTER, 13);
        if (rc != SUCCEEDED)
            HandleErrorDBC(c_hdbc1);

```

```

rc = bcp_bind(c_hdbc1, (BYTE *) c_credit, 0, CREDIT_LEN, NULL, 0, 0,
14);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

rc = bcp_bind(c_hdbc1, (BYTE *) &c_credit_lim, 0, SQL_VARLEN_DATA,
NULL, 0, SQLFLT8, 15);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

rc = bcp_bind(c_hdbc1, (BYTE *) &c_discount, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 16);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

    // fix to avoid ODBC float to numeric conversion problem.

// rc = bcp_bind(c_hdbc1, (BYTE *) &c_balance, 0, SQL_VARLEN_DATA,
NULL, 0, SQLFLT8, 17);
// if (rc != SUCCEED)
//     HandleErrorDBC(c_hdbc1);
rc = bcp_bind(c_hdbc1, (BYTE *) c_balance, 0, 5, NULL, 0,
SQLCHARACTER, 17);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

rc = bcp_bind(c_hdbc1, (BYTE *) &c_ytd_payment, 0, SQL_VARLEN_DATA,
NULL, 0, SQLFLT8, 18);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

rc = bcp_bind(c_hdbc1, (BYTE *) &c_payment_cnt, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 19);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

rc = bcp_bind(c_hdbc1, (BYTE *) &c_delivery_cnt, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 20);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

rc = bcp_bind(c_hdbc1, (BYTE *) c_data, 0, 500, NULL, 0, 0, 21);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

for (i = 0; i < customers_per_district; i++)
{
    c_id = customer_buf[i].c_id;
    c_d_id = customer_buf[i].c_d_id;
    c_w_id = customer_buf[i].c_w_id;

    strcpy(c_first, customer_buf[i].c_first);
    strcpy(c_middle, customer_buf[i].c_middle);
    strcpy(c_last, customer_buf[i].c_last);
    strcpy(c_street_1, customer_buf[i].c_street_1);
    strcpy(c_street_2, customer_buf[i].c_street_2);
    strcpy(c_city, customer_buf[i].c_city);
    strcpy(c_state, customer_buf[i].c_state);

```

```

strcpy(c_zip, customer_buf[i].c_zip);
strcpy(c_phone, customer_buf[i].c_phone);
strcpy(c_credit, customer_buf[i].c_credit);

FormatDate(&c_since);

c_credit_lim = customer_buf[i].c_credit_lim;
c_discount = customer_buf[i].c_discount;

// fix to avoid ODBC float to numeric conversion problem.

// c_balance = customer_buf[i].c_balance;
strcpy(c_balance, customer_buf[i].c_balance);

c_ytd_payment = customer_buf[i].c_ytd_payment;
c_payment_cnt = customer_buf[i].c_payment_cnt;
c_delivery_cnt = customer_buf[i].c_delivery_cnt;

strcpy(c_data, customer_buf[i].c_data);

// Send data to server
rc = bcp_sendrow(c_hdbc1);
if (rc != SUCCEED)
    HandleErrorDBC(c_hdbc1);

customer_rows_loaded++;
CheckForCommit(c_hdbc1, c_hstmt1, customer_rows_loaded,
"customer", &customer_time_start->time_start);
}

}

//=====
//
// Function : LoadHistoryTable
//
//=====

void LoadHistoryTable(LOADER_TIME_STRUCT *history_time_start)
{
    int i;
    long c_id;
    short c_d_id;
    short c_w_id;
    double h_amount;
    char h_data[H_DATA_LEN+1];
    char h_date[H_DATE_LEN+1];
    RETCODE rc;

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

```

```

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 3);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 4);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 5);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) &h_date, 0, H_DATE_LEN, NULL, 0,
SQLCHARACTER, 6);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) &h_amount, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 7);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) h_data, 0, H_DATA_LEN, NULL, 0, 0, 8);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    for (i = 0; i < customers_per_district; i++)
    {
        c_id = customer_buf[i].c_id;
        c_d_id = customer_buf[i].c_d_id;
        c_w_id = customer_buf[i].c_w_id;
        h_amount = customer_buf[i].h_amount;
        strcpy(h_data, customer_buf[i].h_data);

        FormatDate(&h_date);

        // send to server
        rc = bcp_sendrow(c_hdbc2);
        if (rc != SUCCEED)
            HandleErrorDBC(c_hdbc2);

        history_rows_loaded++;
        CheckForCommit(c_hdbc2, c_hstmt2, history_rows_loaded,
"history", &history_time_start->time_start);
    }
}

//=====
//
// Function : LoadOrders
//
//=====
void LoadOrders()

```

```

{
    LOADER_TIME_STRUCT    orders_time_start;
    LOADER_TIME_STRUCT    new_order_time_start;
    LOADER_TIME_STRUCT    order_line_time_start;
    short                  w_id;
    short                  d_id;
    DWORD                  dwThreadId[MAX_ORDER_THREADS];
    HANDLE                  hThread[MAX_ORDER_THREADS];
    char                    name[20];
    RETCODE                 rc;
    char                    bcphint[128];

    // seed with unique number
    seed(6);

    printf("Loading orders...\n");

    // if build index before load...
    if ((aptr->build_index == 1) && (aptr->index_order == 1))
    {
        BuildIndex("idxordcl");
        BuildIndex("idxnodcl");
        BuildIndex("idxodlcl");
    }

    // initialize bulk copy
    sprintf(name, "%s..%s", aptr->database, "orders");

    rc = bcp_init(o_hdbc1, name, NULL, "logs\\orders.err", DB_IN);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc1);

    if ((aptr->build_index == 1) && (aptr->index_order == 1))
    {
        sprintf(bcphint, "tablock, order (o_w_id, o_d_id, o_id),
ROWS_PER_BATCH = %u", (aptr->num_warehouses * 30000));
        rc = bcp_control(o_hdbc1, BCPHINTS, (void*) bcphint);
        if (rc != SUCCEED)
            HandleErrorDBC(o_hdbc1);
    }

    sprintf(name, "%s..%s", aptr->database, "new_order");

    rc = bcp_init(o_hdbc2, name, NULL, "logs\\neword.err", DB_IN);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc2);

    if ((aptr->build_index == 1) && (aptr->index_order == 1))
    {
        sprintf(bcphint, "tablock, order (no_w_id, no_d_id,
no_o_id), ROWS_PER_BATCH = %u", (aptr->num_warehouses * 9000));
        rc = bcp_control(o_hdbc2, BCPHINTS, (void*) bcphint);
        if (rc != SUCCEED)
            HandleErrorDBC(o_hdbc2);
    }

    sprintf(name, "%s..%s", aptr->database, "order_line");

    rc = bcp_init(o_hdbc3, name, NULL, "logs\\ordline.err", DB_IN);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);
}

```

```

    if ((aptr->build_index == 1) && (aptr->index_order == 1))
    {
        sprintf(bcphint, "tablock, order (ol_w_id, ol_d_id,
ol_o_id, ol_number), ROWS_PER_BATCH = %u", (aptr->num_warehouses *
300000));
        rc = bcp_control(o_hdbc3, BCPHINTS, (void*) bcphint);
        if (rc != SUCCEED)
            HandleErrorDBC(o_hdbc3);
    }

    orders_rows_loaded      = 0;
    new_order_rows_loaded  = 0;
    order_line_rows_loaded  = 0;

    OrdersBufInit();

    orders_time_start.time_start = (TimeNow() / MILLI);
    new_order_time_start.time_start = (TimeNow() / MILLI);
    order_line_time_start.time_start = (TimeNow() / MILLI);

    for (w_id = (short)aptr->starting_warehouse; w_id <= aptr-
>num_warehouses; w_id++)
    {
        for (d_id = 1; d_id <= DISTRICT_PER_WAREHOUSE; d_id++)
        {
            OrdersBufLoad(d_id, w_id);

            // start parallel loading threads here...

            // start Orders table thread
            printf("...Loading Order Table for: d_id = %d, w_id
= %d\n", d_id, w_id);

            hThread[0] = CreateThread(NULL,
0,
(LPTHREAD_START_ROUTINE) LoadOrdersTable,
&orders_time_start,
0,
&dwThreadID[0]);

            if (hThread[0] == NULL)
            {
                printf("Error, failed in creating creating
thread = 0.\n");
                exit(-1);
            }

            // start NewOrder table thread

            printf("...Loading New-Order Table for: d_id = %d,
w_id = %d\n", d_id, w_id);

            hThread[1] = CreateThread(NULL,
0,

```

```

(LPTHREAD_START_ROUTINE) LoadNewOrderTable,
&new_order_time_start,
0,
&dwThreadID[1]);

            if (hThread[1] == NULL)
            {
                printf("Error, failed in creating creating
thread = 1.\n");
                exit(-1);
            }

            // start Order-Line table thread

            printf("...Loading Order-Line Table for: d_id = %d,
w_id = %d\n", d_id, w_id);

            hThread[2] = CreateThread(NULL,
0,
(LPTHREAD_START_ROUTINE) LoadOrderLineTable,
&order_line_time_start,
0,
&dwThreadID[2]);

            if (hThread[2] == NULL)
            {
                printf("Error, failed in creating creating
thread = 2.\n");
                exit(-1);
            }

            WaitForSingleObject( hThread[0], INFINITE );
            WaitForSingleObject( hThread[1], INFINITE );
            WaitForSingleObject( hThread[2], INFINITE );

            if (CloseHandle(hThread[0]) == FALSE)
            {
                printf("Error, failed in closing Orders
thread handle with errno: %d\n", GetLastError());
            }

            if (CloseHandle(hThread[1]) == FALSE)
            {
                printf("Error, failed in closing NewOrder
thread handle with errno: %d\n", GetLastError());
            }

            if (CloseHandle(hThread[2]) == FALSE)
            {
                printf("Error, failed in closing OrderLine
thread handle with errno: %d\n", GetLastError());
            }
        }
    }
}

```

```

    printf("Finished loading orders.\n");

return;
}

//=====
//
// Function   : OrdersBufInit
//
// Clears shared buffer for ORDERS, NEWORDER, and ORDERLINE
//
//=====

void OrdersBufInit()
{
    int     i;
    int     j;

    for (i=0;i<orders_per_district;i++)
    {
        orders_buf[i].o_id = 0;
        orders_buf[i].o_d_id = 0;
        orders_buf[i].o_w_id = 0;
        orders_buf[i].o_c_id = 0;
        orders_buf[i].o_carrier_id = 0;
        orders_buf[i].o_ol_cnt = 0;
        orders_buf[i].o_all_local = 0;

        for (j=0;j<=14;j++)
        {
            orders_buf[i].o_ol[j].ol = 0;
            orders_buf[i].o_ol[j].ol_i_id = 0;
            orders_buf[i].o_ol[j].ol_supply_w_id = 0;
            orders_buf[i].o_ol[j].ol_quantity = 0;
            orders_buf[i].o_ol[j].ol_amount = 0;
            strcpy(orders_buf[i].o_ol[j].ol_dist_info,"");
        }
    }
}

//=====
//
// Function   : OrdersBufLoad
//
// Fills shared buffer for ORDERS, NEWORDER, and ORDERLINE
//
//=====

void OrdersBufLoad(int d_id, int w_id)
{
    int     cust[ORDERS_PER_DIST+1];
    long    o_id;
    short   ol;

```

```

    printf("...Loading Order Buffer for: d_id = %d, w_id = %d\n",
           d_id, w_id);

    GetPermutation(cust, ORDERS_PER_DIST);

    for (o_id=0;o_id<orders_per_district;o_id++)
    {
        // Generate ORDER and NEW-ORDER data

        orders_buf[o_id].o_d_id = d_id;
        orders_buf[o_id].o_w_id = w_id;
        orders_buf[o_id].o_id = o_id+1;
        orders_buf[o_id].o_c_id = cust[o_id+1];
        orders_buf[o_id].o_ol_cnt = (short)RandomNumber(5L, 15L);

        if (o_id < first_new_order)
        {
            orders_buf[o_id].o_carrier_id =
(short)RandomNumber(1L, 10L);
            orders_buf[o_id].o_all_local = 1;
        }
        else
        {
            orders_buf[o_id].o_carrier_id = 0;
            orders_buf[o_id].o_all_local = 1;
        }

        for (ol=0; ol<orders_buf[o_id].o_ol_cnt; ol++)
        {
            orders_buf[o_id].o_ol[ol].ol = ol+1;
            orders_buf[o_id].o_ol[ol].ol_i_id =
RandomNumber(1L, max_items);
            orders_buf[o_id].o_ol[ol].ol_supply_w_id = w_id;
            orders_buf[o_id].o_ol[ol].ol_quantity = 5;
            MakeAlphaString(24, 24, OL_DIST_INFO_LEN,
&orders_buf[o_id].o_ol[ol].ol_dist_info);

            // Generate ORDER-LINE data
            if (o_id < first_new_order)
            {
                orders_buf[o_id].o_ol[ol].ol_amount = 0;
                // Added to insure ol_delivery_d set
properly during load

                FormatDate(&orders_buf[o_id].o_ol[ol].ol_delivery_d);
            }
            else
            {
                orders_buf[o_id].o_ol[ol].ol_amount =
RandomNumber(1,999999)/100.0;
                // Added to insure ol_delivery_d set
properly during load

                // odbc datetime format

                strcpy(orders_buf[o_id].o_ol[ol].ol_delivery_d,"1899-12-31
12:00:00.000");

```

```

    }
}

//=====
//
// Function   : LoadOrdersTable
//
//=====

void LoadOrdersTable(LOADER_TIME_STRUCT *orders_time_start)
{
    int          i;
    long         o_id;
    short        o_d_id;
    short        o_w_id;
    long         o_c_id;
    short        o_carrier_id;
    short        o_ol_cnt;
    short        o_all_local;
    char         o_entry_d[O_ENTRY_D_LEN+1];
    RETCODE      rc;
    DBINT        rcint;

    // bind ORDER data
    rc = bcp_bind(o_hdbc1, (BYTE *) &o_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 3);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_c_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 4);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_entry_d, 0, O_ENTRY_D_LEN,
NULL, 0, SQLCHARACTER, 5);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_carrier_id, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 6);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_ol_cnt, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 7);
    if (rc != SUCCEED)

```

```

        HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_all_local, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 8);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc1);

    for (i = 0; i < orders_per_district; i++)
    {
        o_id          = orders_buf[i].o_id;
        o_d_id        = orders_buf[i].o_d_id;
        o_w_id        = orders_buf[i].o_w_id;
        o_c_id        = orders_buf[i].o_c_id;
        o_carrier_id  = orders_buf[i].o_carrier_id;
        o_ol_cnt      = orders_buf[i].o_ol_cnt;
        o_all_local   = orders_buf[i].o_all_local;

        FormatDate(&o_entry_d);

        // send data to server
        rc = bcp_sendrow(o_hdbc1);
        if (rc != SUCCEED)
            HandleErrorDBC(o_hdbc1);

        orders_rows_loaded++;
        CheckForCommit(o_hdbc1, o_hstmt1, orders_rows_loaded,
"orders", &orders_time_start->time_start);
    }

    // rcint = bcp_batch(o_hdbc1);
    // if (rcint < 0)
    //     HandleErrorDBC(o_hdbc1);

    if ((o_w_id == aptr->num_warehouses) && (o_d_id == 10))
    {
        rcint = bcp_done(o_hdbc1);
        if (rcint < 0)
            HandleErrorDBC(o_hdbc1);

        SQLFreeStmt(o_hstmt1, SQL_DROP);
        SQLDisconnect(o_hdbc1);
        SQLFreeConnect(o_hdbc1);

        // if build index after load...
        if ((aptr->build_index == 1) && (aptr->index_order == 0))
            BuildIndex("idxordc1");

        // build non-clustered index
        if (aptr->build_index == 1)
            BuildIndex("idxordnc");
    }
}

//=====
//
// Function   : LoadNewOrderTable
//
//=====

```

```

void LoadNewOrderTable(LOADER_TIME_STRUCT *new_order_time_start)
{
    int            i;
    long           o_id;
    short          o_d_id;
    short          o_w_id;
    RETCODE        rc;
    DBINT          rcint;

    // Bind NEW-ORDER data

    rc = bcp_bind(o_hdbc2, (BYTE *) &o_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc2);

    rc = bcp_bind(o_hdbc2, (BYTE *) &o_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc2);

    rc = bcp_bind(o_hdbc2, (BYTE *) &o_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 3);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc2);

    for (i = first_new_order; i < last_new_order; i++)
    {
        o_id    = orders_buf[i].o_id;
        o_d_id  = orders_buf[i].o_d_id;
        o_w_id  = orders_buf[i].o_w_id;

        rc = bcp_sendrow(o_hdbc2);
        if (rc != SUCCEED)
            HandleErrorDBC(o_hdbc2);

        new_order_rows_loaded++;
        CheckForCommit(o_hdbc2, o_hstmt2, new_order_rows_loaded,
"new_order", &new_order_time_start->time_start);
    }

    // rcint = bcp_batch(o_hdbc2);
    // if (rcint < 0)
    //     HandleErrorDBC(o_hdbc2);

    if ((o_w_id == aptr->num_warehouses) && (o_d_id == 10))
    {
        rcint = bcp_done(o_hdbc2);
        if (rcint < 0)
            HandleErrorDBC(o_hdbc2);

        SQLFreeStmt(o_hstmt2, SQL_DROP);
        SQLDisconnect(o_hdbc2);
        SQLFreeConnect(o_hdbc2);

        // if build index after load...
        if ((aptr->build_index == 1) && (aptr->index_order == 0))
            BuildIndex("idxnodcl");
    }
}

```

```

}
//=====
//
// Function   : LoadOrderLineTable
//
//=====
void LoadOrderLineTable(LOADER_TIME_STRUCT *order_line_time_start)
{
    int            i,j;
    long           o_id;
    short          o_d_id;
    short          o_w_id;
    long           ol;
    long           ol_i_id;
    short          ol_supply_w_id;
    short          ol_quantity;
    double         ol_amount;
    char           ol_dist_info[DIST_INFO_LEN+1];
    char           ol_delivery_d[OL_DELIVERY_D_LEN+1];
    RETCODE        rc;
    DBINT          rcint;

    // bind ORDER-LINE data
    rc = bcp_bind(o_hdbc3, (BYTE *) &o_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &o_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &o_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 3);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 4);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol_i_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 5);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol_supply_w_id, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 6);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol_delivery_d, 0,
OL_DELIVERY_D_LEN, NULL, 0, SQLCHARACTER, 7);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);
}

```

```

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol_quantity, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 8);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol_amount, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 9);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) ol_dist_info, 0, DIST_INFO_LEN, NULL,
0, 0, 10);
    if (rc != SUCCEED)
        HandleErrorDBC(o_hdbc3);

    for (i = 0; i < orders_per_district; i++)
    {
        o_id    = orders_buf[i].o_id;
        o_d_id  = orders_buf[i].o_d_id;
        o_w_id  = orders_buf[i].o_w_id;

        for (j=0; j < orders_buf[i].o_ol_cnt; j++)
        {
            ol            = orders_buf[i].o_ol[j].ol;
            ol_i_id      = orders_buf[i].o_ol[j].ol_i_id;
            ol_supply_w_id =
orders_buf[i].o_ol[j].ol_supply_w_id;
            ol_quantity  = orders_buf[i].o_ol[j].ol_quantity;
            ol_amount    = orders_buf[i].o_ol[j].ol_amount;

            strcpy(ol_delivery_d,orders_buf[i].o_ol[j].ol_delivery_d);

            strcpy(ol_dist_info,orders_buf[i].o_ol[j].ol_dist_info);

            rc = bcp_sendrow(o_hdbc3);
            if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

            order_line_rows_loaded++;
            CheckForCommit(o_hdbc3, o_hstmt3,
order_line_rows_loaded, "order_line", &order_line_time_start->time_start);
        }
    }

    // rcint = bcp_batch(o_hdbc3);
    // if (rcint < 0)
    //     HandleErrorDBC(o_hdbc3);

    if ((o_w_id == aptr->num_warehouses) && (o_d_id == 10))
    {
        rcint = bcp_done(o_hdbc3);
        if (rcint < 0)
            HandleErrorDBC(o_hdbc3);

        SQLFreeStmt(o_hstmt3, SQL_DROP);
        SQLDisconnect(o_hdbc3);
        SQLFreeConnect(o_hdbc3);
    }

```

```

// if build index after load...
if ((aptr->build_index == 1) && (aptr->index_order == 0))
    BuildIndex("idxodlcl");
}
}

//=====
//
// Function    : GetPermutation
//
//=====

void GetPermutation(int perm[], int n)
{
    int i, r, t;

    for (i=1;i<=n;i++)
        perm[i] = i;

    for (i=1;i<=n;i++)
    {
        r = RandomNumber(i,n);
        t = perm[i];
        perm[i] = perm[r];
        perm[r] = t;
    }
}

//=====
//
// Function    : CheckForCommit
//
//=====

void CheckForCommit(HDBC hdbc,
                    HSTMT hstmt,
                    int rows_loaded,
                    char *table_name,
                    long *time_start)
{
    long    time_end, time_diff;
           // DBINT    rcint;

    if ( !(rows_loaded % aptr->batch) )
    {
        // rcint = bcp_batch(hdbc);
        // if (rcint < 0)
        //     HandleErrorDBC(hdbc);

        time_end = (TimeNow() / MILLI);
        time_diff = time_end - *time_start;

        printf("-> Loaded %ld rows into %s in %ld sec - Total = %d
(%.2f rps)\n",

```



```

        aptr->batch,
        table_name,
            time_diff,
        rows_loaded,
            (float) aptr->batch / (time_diff ? time_diff
: 1L));
    }
    *time_start = time_end;
}
return;
}

//=====
//
// Function   : OpenConnections
//
//=====

void OpenConnections()
{
    RETCODE      rc;

    char          szDriverString[300];
    char          szDriverStringOut[1024];
    SQLSMALLINT   cbDriverStringOut;

    SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &henv );
    SQLSetEnvAttr(henv, SQL_ATTR_ODBC_VERSION, (void*)SQL_OV_ODBC3, 0
);

    SQLAllocHandle(SQL_HANDLE_DBC, henv , &i_hdbc1);
    SQLAllocHandle(SQL_HANDLE_DBC, henv , &w_hdbc1);
    SQLAllocHandle(SQL_HANDLE_DBC, henv , &c_hdbc1);
    SQLAllocHandle(SQL_HANDLE_DBC, henv , &c_hdbc2);
    SQLAllocHandle(SQL_HANDLE_DBC, henv , &o_hdbc1);
    SQLAllocHandle(SQL_HANDLE_DBC, henv , &o_hdbc2);
    SQLAllocHandle(SQL_HANDLE_DBC, henv , &o_hdbc3);

    SQLSetConnectAttr(i_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
    SQLSetConnectAttr(w_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
    SQLSetConnectAttr(c_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
    SQLSetConnectAttr(c_hdbc2, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
    SQLSetConnectAttr(o_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
    SQLSetConnectAttr(o_hdbc2, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
    SQLSetConnectAttr(o_hdbc3, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );

    // Open connections to SQL Server
    // Connection 1

```

```

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
        aptr->server,
        aptr->user,
        aptr->password,
        aptr->database );

    rc = SQLSetConnectOption (i_hdbc1, SQL_PACKET_SIZE, aptr-
>pack_size);
    if (rc != SUCCEED)
        HandleErrorDBC(i_hdbc1);

    rc = SQLDriverConnect ( i_hdbc1,
        NULL,
        (SQLCHAR*)&szDriverString[0]
,
        SQL_NTS,
        (SQLCHAR*)&szDriverStringOut[0],
        sizeof(szDriverStringOut),
        &cbDriverStringOut,
        SQL_DRIVER_NOPROMPT );

    if (rc != SUCCEED)
        HandleErrorDBC(i_hdbc1);

    // Connection 2

    sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
        aptr->server,
        aptr->user,
        aptr->password,
        aptr->database );

    rc = SQLSetConnectOption (w_hdbc1, SQL_PACKET_SIZE, aptr-
>pack_size);
    if (rc != SUCCEED)
        HandleErrorDBC(w_hdbc1);

    rc = SQLDriverConnect ( w_hdbc1,
        NULL,
        (SQLCHAR*)&szDriverString[0] ,
        SQL_NTS,
        (SQLCHAR*)&szDriverStringOut[0],
        sizeof(szDriverStringOut),
        &cbDriverStringOut,
        SQL_DRIVER_NOPROMPT
);

    if (rc != SUCCEED)
        HandleErrorDBC(w_hdbc1);

    // Connection 3

    sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
        aptr->server,
        aptr->user,

```

```

        aptr->password,
        aptr->database );

    rc = SQLSetConnectOption ( c_hdbc1, SQL_PACKET_SIZE, aptr-
>pack_size);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

    rc = SQLDriverConnect ( c_hdbc1,
                            NULL,
                            (SQLCHAR*)&szDriverString[0] ,
                            SQL_NTS,
                            (SQLCHAR*)&szDriverStringOut[0] ,
                            sizeof(szDriverStringOut),
                            &cbDriverStringOut,
                            SQL_DRIVER_NOPROMPT
);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc1);

    // Connection 4

    sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
            aptr->server,
            aptr->user,
            aptr->password,
            aptr->database );

    rc = SQLSetConnectOption (c_hdbc2, SQL_PACKET_SIZE, aptr-
>pack_size);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    rc = SQLDriverConnect ( c_hdbc2,
                            NULL,
                            (SQLCHAR*)&szDriverString[0] ,
                            SQL_NTS,
                            (SQLCHAR*)&szDriverStringOut[0] ,
                            sizeof(szDriverStringOut),
                            &cbDriverStringOut,
                            SQL_DRIVER_NOPROMPT
);
    if (rc != SUCCEED)
        HandleErrorDBC(c_hdbc2);

    // Connection 5

    sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
            aptr->server,
            aptr->user,
            aptr->password,
            aptr->database );

```

```

        rc = SQLSetConnectOption (o_hdbc1, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)
            HandleErrorDBC(o_hdbc1);

        rc = SQLDriverConnect ( o_hdbc1,
                                NULL,
                                (SQLCHAR*)&szDriverString[0] ,
                                SQL_NTS,
                                (SQLCHAR*)&szDriverStringOut[0] ,
                                sizeof(szDriverStringOut),
                                &cbDriverStringOut,
                                SQL_DRIVER_NOPROMPT
);
        if (rc != SUCCEED)
            HandleErrorDBC(o_hdbc1);

        // Connection 6

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                aptr->server,
                aptr->user,
                aptr->password,
                aptr->database );

        rc = SQLSetConnectOption (o_hdbc2, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)
            HandleErrorDBC(o_hdbc2);

        rc = SQLDriverConnect ( o_hdbc2,
                                NULL,
                                (SQLCHAR*)&szDriverString[0] ,
                                SQL_NTS,
                                (SQLCHAR*)&szDriverStringOut[0] ,
                                sizeof(szDriverStringOut),
                                &cbDriverStringOut,
                                SQL_DRIVER_NOPROMPT
);
        if (rc != SUCCEED)
            HandleErrorDBC(o_hdbc2);

        // Connection 7

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                aptr->server,
                aptr->user,
                aptr->password,
                aptr->database );

        rc = SQLSetConnectOption (o_hdbc3, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)

```

```

        HandleErrorDBC(o_hdbc3);

rc = SQLDriverConnect ( o_hdbc3,
                        NULL,
                        (SQLCHAR*)&szDriverString[0] ,
                        SQL_NTS,
                        (SQLCHAR*)&szDriverStringOut[0],
                        sizeof(szDriverStringOut),
                        &cbDriverStringOut,
                        SQL_DRIVER_NOPROMPT
);
    if (rc != SUCCEEDED)
        HandleErrorDBC(o_hdbc3);
}

//=====
//
// Function name: BuildIndex
//
//=====

void BuildIndex(char *index_script)
{
    char cmd[256];

    printf("Starting index creation: %s\n",index_script);

    sprintf(cmd, "isql -S%s -U%s -P%s -e -i%s\\%s.sql > logs\\%s.log",
            aptr->server,
            aptr->user,
            aptr->password,
            aptr->index_script_path,
            index_script,
            index_script);

    system(cmd);

    printf("Finished index creation: %s\n",index_script);
}

void HandleErrorDBC (SQLHDBC hdbc1)
{
    SQLCHAR      SqlState[6], Msg[SQL_MAX_MESSAGE_LENGTH];
    SQLINTEGER   NativeError;
    SQLSMALLINT  i, MsgLen;
    SQLRETURN    rc2;
    char         timebuf[128];
    char         datebuf[128];
    FILE         *fp1;

    i = 1;
    while (( rc2 = SQLGetDiagRec(SQL_HANDLE_DBC , hdbc1, i, SqlState ,
    &NativeError,
                                Msg, sizeof(Msg) , &MsgLen )) !=
SQL_NO_DATA )

```

```

{
    sprintf( szLastError , "%s" , Msg );
    _strtime(timebuf);
    _strdate(datebuf);
    printf( "[%s : %s] %s\n" , datebuf, timebuf, szLastError);
    fp1 = fopen("logs\\tpccldr.err","w");
    if (fp1 == NULL)
        printf("ERROR: Unable to open errorlog file.\n");
    else
    {
        fprintf(fp1, "[%s : %s] %s\n" , datebuf, timebuf,
szLastError);
        fclose(fp1);
    }
    i++;
}

void FormatDate ( char* szTimeCOutput )
{
    struct tm when;
    time_t now;

    time( &now );
    when = *localtime( &now );

    mktime( &when );

    // odbc datetime format
    strftime( szTimeCOutput , 30 , "%Y-%m-%d %H:%M:%S.000", &when );

    return;
}

```


Appendix C - Tunable Parameters

Microsoft SQL Server Startup Parameters

```
C:\MSSQL\BINN\SQLSERVR.EXE -c -x -t3502 -g38
```

Where:

- -c Start SQL Server independently of the Service Control Manager
- -x Disables the keeping of CPU time and cache hit ratio statistics
- -t3502 Writes a message to the SQL Server Errorlog showing the beginning and ending time of each checkpoint
- -g38 Specifies the amount of virtual address space, in MB, SQL Server will leave available for memory allocations, excluding the buffer pool and thread stacks, such as dynamically-loaded DLLs, extended procedure calls, etc.. If this option is not specified, SQL Server will use a value that is suitable for a wide range of runtime environments. Use of this option may be appropriate in 2GB (3GB Enterprise Edition) configurations in which the memory usage requirements of SQL Server are atypical and the virtual address space of the SQL Server process is totally in use. Incorrect use of this option can lead to conditions under which SQL Server may not start or may encounter runtime errors.

SQL Server Stack Size

The default stack size for Microsoft SQL Server 7.0 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: 131072 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.

BOOT.INI

The /3gb switch was added to the boot.ini file to cause Windows NT Enterprise Edition to allow 3GB of user and 1GB of kernel virtual address space, rather than the usual 2GB of virtual address space for each.

Microsoft SQL Server Configuration Parameters

```
1> 2> 3> 4> 5> 6> 7> 8> 9> 10> 11>
-- File:      VERSION.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Returns SQL Server version string
```

```
print " "
select convert(char(30), getdate(),9)
print " "
```

```
-----
Dec 10 1998 11:40:20:560AM
```

```
(1 row affected)
```

```
1> 2> 3>
select @@version
```

```
-----
-----
-----
-----
-----
Microsoft SQL Server 7.00 - 7.00.618 (Intel X86)
Nov 8 1998 02:05:55
Copyright (c) 1988-1998 Microsoft Corporation
Enterprise Edition on Windows NT 4.0 (Build 1381: Service Pack 4)
```

```
(1 row affected)
1> 2>
1> 2> 3> 4> 5> 6> 7> 8> 9> 10>
-- File:      CONFIG.SQL
--           Microsoft TPC-C Benchmark Kit Ver. 4.00
--           Copyright Microsoft, 1996
-- Purpose:   Collects SQL Server configuration parameters
```

```
print " "
select convert(char(30), getdate(),9)
print " "
```

```
-----
Dec 10 1998 11:40:21:373AM
```

```
(1 row affected)
```

```
1> 2> 3> DBCC execution completed. If DBCC printed error messages, contact
your system administrator.
Configuration option changed. Run the RECONFIGURE statement to install.
```

```
sp_configure "show advanced",1
1> 2> reconfigure with override
1> 2> sp_configure
```

name	minimum	maximum	config_value	run_value

affinity mask				
0	2147483647	15	15	
allow updates				
0	1	1	1	
cost threshold for parallelism				
0	32767	5	5	
cursor threshold				
-1	2147483647	-1	-1	
default language				
0	9999	0	0	
default sortorder id				
0	255	50	50	
extended memory size (MB)				
0	2147483647	0	0	
fill factor (%)				
0	100	0	0	
index create memory (KB)				
704	1600000	0	0	
language in cache				
3	100	3	3	
language neutral full-text				
0	1	0	0	
lightweight pooling				
0	1	1	1	
locks				
5000	2147483647	0	0	
max async IO				
1	255	255	255	
max degree of parallelism				
0	32	1	1	

max server memory (MB)			
4	2147483647	2147483647	2147483647
max text repl size (B)			
0	2147483647	65536	65536
max worker threads			
10	1024	175	175
media retention			
0	365	0	0
min memory per query (KB)			
512	2147483647	512	512
min server memory (MB)			
0	2147483647	0	0
nested triggers			
0	1	0	0
network packet size (B)			
512	65535	4096	4096
open objects			
0	2147483647	0	0
priority boost			
0	1	1	1
query governor cost limit			
0	2147483647	0	0
query wait (s)			
-1	2147483647	-1	-1
recovery interval (min)			
0	32767	32767	32767
remote access			
0	1	0	0
remote login timeout (s)			
0	2147483647	5	5
remote proc trans			
0	1	0	0
remote query timeout (s)			
0	2147483647	0	0
resource timeout (s)			
5	2147483647	10	10
scan for startup procs			
0	1	1	1
set working set size			
0	1	0	0
show advanced options			
0	1	1	1
spin counter			
1	2147483647	10000	10000
time slice (ms)			
50	1000	100	100
two digit year cutoff			
1753	9999	2049	2049
Unicode comparison style			
0	2147483647	0	0
Unicode locale id			
0	2147483647	33280	33280
user connections			
0	32767	0	0
user options			
0	4095	0	0

```
1>
```

Internal RAID Configuration Parameters

```
*****
*           MYLEX Disk Array Controller - Configuration Utility
*
*           Version 4.76
*
*****

CONFIGURATION INFORMATION OF :
=====

3 Channel - 15 Target  DAC1164P  #1    Firmware version 5.06

PHYSICAL PACK INFORMATION :
=====

Number of Packs = 4

Pack 0 : [0:0]  [0:1]  [0:2]  [0:3]  [0:4]  [0:5]  [0:6]
Pack 1 : [0:8]  [0:9]  [0:10] [0:11] [0:12] [0:13] [0:14]
Pack 2 : [1:0]  [1:1]  [1:2]  [1:3]  [1:4]  [1:5]  [1:6]
Pack 3 : [1:8]  [1:9]  [1:10] [1:11] [1:12] [1:13] [1:14]

SYSTEM DRIVE INFORMATION :
=====

Number of System Drives = 1

Sys Drv #   Phy. Size   Raid Level   Eff. Size   Write Policy
=====
0           243124 MB     0            243124 MB   Write Thru

*****
*           MYLEX Disk Array Controller - Configuration Utility
*
*           Version 4.76
*
*****

CONFIGURATION INFORMATION OF :
=====

3 Channel - 15 Target  DAC1164P  #2    Firmware version 5.06

PHYSICAL PACK INFORMATION :
=====

Number of Packs = 4
```

```
Pack 0 : [0:0]  [0:1]  [0:2]  [0:3]  [0:4]  [0:5]  [0:6]
Pack 1 : [0:8]  [0:9]  [0:10] [0:11] [0:12] [0:13] [0:14]
Pack 2 : [1:0]  [1:1]  [1:2]  [1:3]  [1:4]  [1:5]  [1:6]
Pack 3 : [1:8]  [1:9]  [1:10] [1:11] [1:12] [1:13] [1:14]
```

```
SYSTEM DRIVE INFORMATION :
=====
```

Number of System Drives = 1

Sys Drv #	Phy. Size	Raid Level	Eff. Size	Write Policy
0	243124 MB	0	243124 MB	Write Thru

```
*****
*           MYLEX Disk Array Controller - Configuration Utility
*
*           Version 4.76
*
*****
```

```
CONFIGURATION INFORMATION OF :
=====
```

3 Channel - 15 Target DAC1164P #3 Firmware version 5.06

```
PHYSICAL PACK INFORMATION :
=====
```

Number of Packs = 4

```
Pack 0 : [0:0]  [0:1]  [0:2]  [0:3]  [0:4]  [0:5]  [0:6]
Pack 1 : [0:8]  [0:9]  [0:10] [0:11] [0:12] [0:13] [0:14]
Pack 2 : [1:0]  [1:1]  [1:2]  [1:3]  [1:4]  [1:5]  [1:6]
Pack 3 : [1:8]  [1:9]  [1:10] [1:11] [1:12] [1:13] [1:14]
```

```
SYSTEM DRIVE INFORMATION :
=====
```

Number of System Drives = 1

Sys Drv #	Phy. Size	Raid Level	Eff. Size	Write Policy
0	121492 MB	0	121492 MB	Write Thru

```
*****
*           MYLEX Disk Array Controller - Configuration Utility
*
*           Version 4.76
*
*****
```

CONFIGURATION INFORMATION OF :
=====

3 Channel - 15 Target DAC1164P #4 Firmware version 5.06

PHYSICAL PACK INFORMATION :
=====

Number of Packs = 1

Pack 0 : [0:0] [1:0]

SYSTEM DRIVE INFORMATION :
=====

Number of System Drives = 1

Sys Drv #	Phy. Size	Raid Level	Eff. Size	Write Policy
0	1231578 MB	1	60789 MB	Write Thru

* MYLEX Disk Array Controller - Configuration Utility
*
* Version 4.76
*

CONFIGURATION INFORMATION OF :
=====

3 Channel - 15 Target DAC1164P #5 Firmware version 5.06

PHYSICAL PACK INFORMATION :
=====

Number of Packs = 4

Pack 0 : [0:0] [0:1] [0:2] [0:3] [0:4] [0:5] [0:6]
Pack 1 : [0:8] [0:9] [0:10] [0:11] [0:12] [0:13] [0:14]
Pack 2 : [1:0] [1:1] [1:2] [1:3] [1:4] [1:5] [1:6]
Pack 3 : [1:8] [1:9] [1:10] [1:11] [1:12] [1:13] [1:14]

SYSTEM DRIVE INFORMATION :
=====

Number of System Drives = 1

Sys Drv #	Phy. Size	Raid Level	Eff. Size	Write Policy
0	1231578 MB	1	60789 MB	Write Thru

0 182308 MB 0 182308 MB Write Thru

* MYLEX Disk Array Controller - Configuration Utility
*
* Version 4.76
*

CONFIGURATION INFORMATION OF :
=====

3 Channel - 15 Target DAC1164P #6 Firmware version 5.06

PHYSICAL PACK INFORMATION :
=====

Number of Packs = 4

Pack 0 : [0:0] [0:1] [0:2] [0:3] [0:4] [0:5] [0:6]
Pack 1 : [0:8] [0:9] [0:10] [0:11] [0:12] [0:13] [0:14]
Pack 2 : [1:0] [1:1] [1:2] [1:3] [1:4] [1:5] [1:6]
Pack 3 : [1:8] [1:9] [1:10] [1:11] [1:12] [1:13] [1:14]

SYSTEM DRIVE INFORMATION :
=====

Number of System Drives = 1

Sys Drv #	Phy. Size	Raid Level	Eff. Size	Write Policy
0	121492 MB	0	121492 MB	Write Thru

* MYLEX Disk Array Controller - Configuration Utility
*
* Version 4.76
*

CONFIGURATION INFORMATION OF :
=====

3 Channel - 15 Target DAC1164P #7 Firmware version 5.06

PHYSICAL PACK INFORMATION :
=====

Number of Packs = 5


```
Pack 0 : [0:0] [0:1] [0:2] [0:3] [0:4] [0:5] [0:6]
Pack 1 : [0:8] [0:9] [0:10] [0:11] [0:12] [0:13] [0:14]
Pack 2 : [1:0] [1:1] [1:2] [1:3] [1:4] [1:5] [1:6]
Pack 3 : [1:8] [1:9] [1:10] [1:11] [1:12] [1:13] [1:14]
Pack 4 : [2:0] [2:1] [2:2] [2:3] [2:4] [2:5] [2:6] [2:8]
```

SYSTEM DRIVE INFORMATION :
=====

Number of System Drives = 2

Sys Drv #	Phy. Size	Raid Level	Eff. Size	Write Policy
0	243124 MB	0	243124 MB	Write Thru
1	121954 MB	5	104537 MB	Write Back

```
*****
*           MYLEX Disk Array Controller - Configuration Utility
*
*                               Version 4.76
*
```

CONFIGURATION INFORMATION OF :
=====

3 Channel - 15 Target DAC1164P #8 Firmware version 5.06

PHYSICAL PACK INFORMATION :
=====

Number of Packs = 5

```
Pack 0 : [0:0] [0:1] [0:2] [0:3] [0:4] [0:5] [0:6]
Pack 1 : [0:8] [0:9] [0:10] [0:11] [0:12] [0:13] [0:14]
Pack 2 : [1:0] [1:1] [1:2] [1:3] [1:4] [1:5] [1:6]
Pack 3 : [1:8] [1:9] [1:10] [1:11] [1:12] [1:13] [1:14]
Pack 4 : [2:0] [2:1] [2:2] [2:3] [2:4] [2:5] [2:6] [2:8]
```

SYSTEM DRIVE INFORMATION :
=====

Number of System Drives = 2

Sys Drv #	Phy. Size	Raid Level	Eff. Size	Write Policy
0	243124 MB	0	243124 MB	Write Thru
1	121954 MB	5	104537 MB	Write Back

External RAID Configuration Parameters

```
*****
*           Unisys Ultra-Wide RAID Controller   OSM1000-C32
*
*****
```

CPU type: 5x86-133 (WB)

Firmware version 1.31G
Bootcode version 1.12B

Total cache: 32 MB

```
- Cache      Write Back:      enabled
              optimization:  sequential (128K stripe size)

- Raid       Rebuild Priority: low
              Write Priority: on Initialization: disabled
                              on Rebuild:      disabled
                              on Normal:      disabled
```

Logical Volume Partition table

Volume ID1	Capacity	RAID	# drives
52098 MB	RAID 0	6	

Host LUN Assignment

SCSI Chl	LUN	LVIDx	PortIdx	Capacity
0	0	0	0	52098 MB

Physical Drives

Id	Slot	Chl	Id	Capacity	Status	XferRate	Vendor/Product
		Firmware					
		0	0	8683 MB	online	41.7 MB	UNISYS
00640ST19101W		8B04	1	8683 MB	online	41.7 MB	UNISYS
00640ST19101W		8B04	2	8683 MB	online	41.7 MB	UNISYS
00640ST19101W		8B04	3	8683 MB	online	41.7 MB	UNISYS
00640ST19101W		8B04	4	8683 MB	online	41.7 MB	UNISYS
00640ST19101W		8B04	5	8683 MB	online	41.7 MB	UNISYS
00640ST19101W		8B04					

Configuration of Log Drives

A single Mylex DAC1164P RAID controller was used in the SUT for the mirrored log drives. Half of the drives were in one disk cage connected to one channel of the controller and half were in a second disk cage connected to a second channel of the controller. The controller implemented the RAID 1 mirroring across the two channels. Write caching was disabled on both the controller and on all the physical drives themselves.

One OSM1000-100 SCSI-to-SCSI RAID controller was used in each of the two log disk cages. Each of these controllers implemented RAID 0 striping on the seven 9GB drives that were in each disk cage, so that the Mylex controller in the SUT saw just two large 'disks'. Each of the IFT controllers had a 32MB cache. Configuration options were set for Write Back caching and Optimized for Sequential IO. The IFT controllers used an algorithm that ensured that cached write data was held for no more than a fraction of a minute before being written to the physical drives.

For the priced configuration, each of the disk cages contained two redundant power supplies. Only one was required to be functional to keep the ITF controller and disk drives operational. A UPS was priced to provide power to one power supply in each disk cage. The second power supply in each disk cage was connected to normal wall power. Thus neither interruption of power or failure of the UPS would affect the two log disk cages (or their IFT controllers and disks). Since the two disk cages were completely independent of each other, this configuration ensured that there was no single point of failure in writing to the log.

NT Server Configuration Information

Microsoft Diagnostics Report For \\AVALON4

OS Version Report

Microsoft (R) Windows NT (TM) Server
Version 4.0 (Build 1381: Service Pack 4) x86 Multiprocessor Free
Registered Owner: SAM&M, Unisys Corporation
Product Number: 70234-810-6895975-67328

System Report

System: AT/AT COMPATIBLE
Hardware Abstraction Layer: MPS 1.4 - APIC platform
BIOS Date: 10/29/98
BIOS Version: AC450NX - PhoenixBIOS 4.0 Releas

Processor list:

0: x86 Family 6 Model 5 Stepping 3 GenuineIntel ~450 Mhz

1: x86 Family 6 Model 5 Stepping 3 GenuineIntel ~450 Mhz
2: x86 Family 6 Model 5 Stepping 3 GenuineIntel ~450 Mhz
3: x86 Family 6 Model 5 Stepping 3 GenuineIntel ~450 Mhz

Video Display Report

BIOS Date: 05/22/96
BIOS Version: CL-GD5436/46 PCI VGA BIOS Version 1.25

Adapter:

Setting: 1024 x 768 x 65536
75 Hz
Type: cirrus compatible display adapter
String: Cirrus Logic Compatible
Memory: 2 MB
Chip Type: Cirrus Logic 5446
DAC Type: Integrated RAMDAC

Driver:

Vendor: Microsoft Corporation
File(s): cirrus.sys, vga.dll, cirrus.dll, vga256.dll, vga64K.dll
Version: 4.00, 4.0.0

Drives Report

C:\ (Local - FAT) SYSTEM Total: 2,096,160 KB, Free: 1,143,968 KB
Serial Number: F035 - 8AA4
Bytes per cluster: 512
Sectors per cluster: 64
Filename length: 255
T:\ (Local - NTFS) BACK1 Total: 107,040,748 KB, Free: 39,749,216 KB
Serial Number: DCE7 - 57BA
Bytes per cluster: 512
Sectors per cluster: 8
Filename length: 255
U:\ (Local - NTFS) BACK2 Total: 107,040,748 KB, Free: 39,722,552 KB
Serial Number: 5CF3 - BAAD
Bytes per cluster: 512
Sectors per cluster: 8
Filename length: 255
Z:\ (Local - NTFS) testfiles Total: 2,345,488 KB, Free: 829,204 KB
Serial Number: B0C5 - 33C8
Bytes per cluster: 512
Sectors per cluster: 8
Filename length: 255

Memory Report

Handles: 2,445
Threads: 112
Processes: 15

Physical Memory (K)
Total: 3,767,720
Available: 525,720
File Cache: 11,692

Kernel Memory (K)

Total: 15,204
 Paged: 8,956
 Nonpaged: 6,248

Commit Charge (K)
 Total: 3,106,648
 Limit: 3,893,332
 Peak: 3,127,472

Pagefile Space (K)
 Total: 273,408
 Total in use: 10,692
 Peak: 11,428

C:\pagefile.sys
 Total: 273,408
 Total in use: 10,692
 Peak: 11,428

Services Report

```
-----
Alerter                               Stopped (Manual)
  C:\WINNT\System32\services.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Shared Process
  Service Dependencies:
    LanmanWorkstation
Computer Browser                       Stopped (Manual)
  C:\WINNT\System32\services.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Shared Process
  Service Dependencies:
    LanmanWorkstation
    LanmanServer
    LmHosts
ClipBook Server                        Stopped (Manual)
  C:\WINNT\system32\clipsrv.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Own Process
  Service Dependencies:
    NetDDE
PCI Hot Plug Service                   Stopped (Disabled)
  C:\WINNT\System32\cpqphps.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Own Process
DHCP Client (TDI)                     Stopped (Disabled)
  C:\WINNT\System32\services.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Shared Process
  Service Dependencies:
    Tcpip
    Afd
```

```
NetBT
EventLog (Event log)                  Running (Automatic)
  C:\WINNT\system32\services.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Shared Process
Server                                 Running (Automatic)
  C:\WINNT\System32\services.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Shared Process
  Group Dependencies:
    TDI
Workstation (NetworkProvider)         Running (Automatic)
  C:\WINNT\System32\services.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Shared Process
  Group Dependencies:
    TDI
License Logging Service                Stopped (Manual)
  C:\WINNT\System32\llssrv.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Own Process
TCP/IP NetBIOS Helper                 Stopped (Manual)
  C:\WINNT\System32\services.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Shared Process
  Group Dependencies:
    NetworkProvider
Messenger                             Stopped (Manual)
  C:\WINNT\System32\services.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Shared Process
  Service Dependencies:
    LanmanWorkstation
    NetBios
MSDTC (MS Transactions)               Stopped (Manual)
  C:\WINNT\System32\msdtc.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Own Process
  Service Dependencies:
    RPCSS
    NTLMSSP
MSSQLServer                           Stopped (Manual)
  C:\MSSQL7\binn\sqlservr.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Own Process
Network DDE (NetDDEGroup)            Stopped (Manual)
  C:\WINNT\system32\netdde.exe
  Service Account Name: LocalSystem
  Error Severity: Normal
  Service Flags: Shared Process
  Service Dependencies:
```

NetDDESDM					
Network DDE DSDM	Stopped	(Manual)		Service Account Name: LocalSystem	
C:\WINNT\system32\netdde.exe				Error Severity: Normal	
Service Account Name: LocalSystem				Service Flags: Own Process	
Error Severity: Normal				Service Dependencies:	
Service Flags: Shared Process				Tcpip	
Net Logon (RemoteValidation)	Stopped	(Manual)		EventLog	
C:\WINNT\System32\lsass.exe				SNMP Trap Service	Stopped (Disabled)
Service Account Name: LocalSystem				C:\WINNT\System32\snmptrap.exe	
Error Severity: Normal				Service Account Name: LocalSystem	
Service Flags: Shared Process				Error Severity: Normal	
Service Dependencies:				Service Flags: Own Process	
LanmanWorkstation				Service Dependencies:	
LmHosts				Tcpip	
NT LM Security Support Provider	Running	(Manual)		EventLog	
C:\WINNT\System32\SERVICES.EXE				Spooler (SpoolerGroup)	Stopped (Manual)
Service Account Name: LocalSystem				C:\WINNT\system32\spoolss.exe	
Error Severity: Normal				Service Account Name: LocalSystem	
Service Flags: Shared Process				Error Severity: Normal	
Plug and Play (PlugPlay)	Running	(Automatic)		Service Flags: Own Process, Interactive	
C:\WINNT\system32\services.exe				SQLServerAgent	Stopped (Manual)
Service Account Name: LocalSystem				C:\MSSQL7\bin\sqlagent.exe	
Error Severity: Normal				Service Account Name: LocalSystem	
Service Flags: Shared Process				Error Severity: Normal	
Protected Storage	Running	(Automatic)		Service Flags: Own Process	
C:\WINNT\System32\pstores.exe				Service Dependencies:	
Service Account Name: LocalSystem				MSSQLServer	
Error Severity: Normal				Telephony Service	Stopped (Manual)
Service Flags: Own Process, Interactive				C:\WINNT\system32\tapisrv.exe	
Service Dependencies:				Service Account Name: LocalSystem	
RpcSs				Error Severity: Normal	
Directory Replicator	Stopped	(Manual)		Service Flags: Own Process	
C:\WINNT\System32\lmrepl.exe				UPS	Stopped (Manual)
Service Account Name: LocalSystem				C:\WINNT\System32\ups.exe	
Error Severity: Normal				Service Account Name: LocalSystem	
Service Flags: Own Process				Error Severity: Normal	
Service Dependencies:				Service Flags: Own Process	
LanmanWorkstation					
LanmanServer				Drivers Report	
Remote Procedure Call (RPC) Locator	Stopped	(Manual)		-----	
C:\WINNT\System32\LOCATOR.EXE				Abiosdsk (Primary disk)	Stopped (Disabled)
Service Account Name: LocalSystem				Error Severity: Ignore	
Error Severity: Normal				Service Flags: Kernel Driver, Shared Process	
Service Flags: Own Process				AFD Networking Support Environment (TDI)	Running (Automatic)
Service Dependencies:				C:\WINNT\System32\drivers\afd.sys	
LanmanWorkstation				Error Severity: Normal	
Rdr				Service Flags: Kernel Driver, Shared Process	
Remote Procedure Call (RPC) Service	Running	(Automatic)		Aha154x (SCSI miniport)	Stopped (Disabled)
C:\WINNT\system32\RpcSs.exe				Error Severity: Normal	
Service Account Name: LocalSystem				Service Flags: Kernel Driver, Shared Process	
Error Severity: Normal				Aha174x (SCSI miniport)	Stopped (Disabled)
Service Flags: Own Process				Error Severity: Normal	
Schedule	Stopped	(Manual)		Service Flags: Kernel Driver, Shared Process	
C:\WINNT\System32\AtSvc.Exe				aic78xx (SCSI miniport)	Stopped (Boot)
Service Account Name: LocalSystem				C:\WINNT\System32\DRIVERS\aic78xx.sys	
Error Severity: Normal				Error Severity: Normal	
Service Flags: Own Process				Service Flags: Kernel Driver, Shared Process	
SNMP	Stopped	(Disabled)		Always (SCSI miniport)	Stopped (Disabled)
C:\WINNT\System32\snmp.exe					

Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
ami0nt (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
amsint (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Arrow (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
atapi (SCSI miniport) Running (Boot)
C:\WINNT\System32\DRIVERS\atapi.sys
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Atdisk (Primary disk) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
ati (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Beep (Base) Running (System)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
BusLogic (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Busmouse (Pointer Port) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Cdaudio (Filter) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Cdfs (File system) Running (Disabled)
Error Severity: Normal
Service Flags: File System Driver, Shared Process
Group Dependencies:
SCSI CDROM Class
Cdrom (SCSI CDROM Class) Running (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Group Dependencies:
SCSI miniport
Changer (Filter) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
cirrus (Video) Running (System)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Cpqarray (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
cpqfws2e (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
dac960nt (SCSI miniport) Running (Boot)
C:\WINNT\System32\drivers\dac960nt.sys
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process

dce376nt (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Dellds (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Dell_DGX (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Disk (SCSI Class) Running (Boot)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Group Dependencies:
SCSI miniport
Diskperf (Filter) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
DptScsi (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
dte329x (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Intel(R) PRO NDIS Driver (NDIS) Running (Automatic)
C:\WINNT\System32\drivers\E100BNT.SYS
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
em (Base) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
et4000 (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Fastfat (Boot file system) Running (Disabled)
Error Severity: Normal
Service Flags: File System Driver, Shared Process
Fd16_700 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Fd7000ex (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Fd8xx (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
flashpnt (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Floppy (Primary disk) Running (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Ftdisk (Filter) Running (Boot)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
gamdrv (SCSI Class) Stopped (Boot)
C:\WINNT\System32\drivers\gamdrv.sys
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
PCI Hot Plug Driver Stopped (Disabled)

System32\DRIVERS\hotplug.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 i8042 Keyboard and PS/2 Mouse Port Driver (Keyboard Port) Running (System)
 System32\DRIVERS\i8042prt.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 Inport (Pointer Port) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 Jazzg300 (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 Jazzg364 (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 Jzvx1484 (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 Keyboard Class Driver (Keyboard Class) Running (System)
 System32\DRIVERS\kbdclass.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 KSecDD (Base) Running (System)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 macdisk (Filter) Running (Boot)
 C:\WINNT\System32\drivers\macdisk.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 megaraid (SCSI Miniport) Stopped (Boot)
 C:\WINNT\System32\drivers\megaraid.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 mga (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 mga_mil (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 mitsumi (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 mkecr5xx (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 Modem (Extended base) Stopped (Manual)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 Mouse Class Driver (Pointer Class) Running (System)
 System32\DRIVERS\mouclass.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 mraid (Primary disk) Stopped (Boot)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 mraid35x (Primary disk) Stopped (Disabled)
 Error Severity: Ignore

Service Flags: Kernel Driver, Shared Process
 Msfs (File system) Running (System)
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process
 Mup (Network) Running (Manual)
 C:\WINNT\System32\drivers\mup.sys
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process
 NetBEUI Protocol (PNP_TDI) Running (Automatic)
 C:\WINNT\System32\drivers\nbf.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 Ncr53c9x (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 ncr77c22 (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 Ncrc700 (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 Ncrc710 (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 Microsoft NDIS System Driver (NDIS) Running (System)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 NetBIOS Interface (NetBIOSGroup) Stopped (Manual)
 C:\WINNT\System32\drivers\netbios.sys
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process
 Group Dependencies:
 TDI
 WINS Client (TCP/IP) (PNP_TDI) Stopped (Disabled)
 C:\WINNT\System32\drivers\netbt.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 Service Dependencies:
 Tcpip
 NetDetect Stopped (Manual)
 C:\WINNT\system32\drivers\netdetect.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 Npfs (File system) Running (System)
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process
 Ntfs (File system) Running (Disabled)
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process
 Null (Base) Running (System)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 Oliscsi (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process
 Parallel (Extended base) Running (Automatic)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process
 Service Dependencies:

Parport
Group Dependencies:
Parallel arbitrator
Parport (Parallel arbitrator) Running (Automatic)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
ParVdm (Extended base) Running (Automatic)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Service Dependencies:
Parport
Group Dependencies:
Parallel arbitrator
PCIDump (PCI Configuration) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Pcmcia (System Bus Extender) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
PnP ISA Enabler Driver (Base) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
PortFltr (port) Stopped (Manual)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Group Dependencies:
SCSI miniport
psidisp (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Ql10wnt (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
qv (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Rdr (Network) Running (Manual)
C:\WINNT\System32\drivers\rdr.sys
Error Severity: Normal
Service Flags: File System Driver, Shared Process
s3 (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Scsiiprnt (Extended base) Stopped (Automatic)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Group Dependencies:
SCSI miniport
Scsiscan (SCSI Class) Running (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Group Dependencies:
SCSI miniport
Serial (Extended base) Running (Automatic)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Sermouse (Pointer Port) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process

Sfloppy (Primary disk) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Group Dependencies:
SCSI miniport
Simbad (Filter) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
slcd32 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Sparrow (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Spock (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Srv (Network) Running (Manual)
C:\WINNT\System32\drivers\srv.sys
Error Severity: Normal
Service Flags: File System Driver, Shared Process
symc810 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Symc8XX (SCSI Miniport) Stopped (Boot)
C:\WINNT\System32\drivers\Symc8XX.sys
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Sym_hi (SCSI Miniport) Running (Boot)
C:\WINNT\System32\drivers\Sym_hi.sys
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Sysdrv (Extended Base) Running (Automatic)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
T128 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
T13B (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
TCP/IP Service (PNP_TDI) Running (Automatic)
C:\WINNT\System32\drivers\tcpip.sys
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
tga (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
tmv1 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Ultra124 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Ultra14f (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Ultra24f (SCSI miniport) Stopped (Disabled)
Error Severity: Normal

```

Service Flags: Kernel Driver, Shared Process
update (Base) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
v7vram (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
VgaSave (Video Save) Stopped (System)
C:\WINNT\System32\drivers\vga.sys
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
VgaStart (Video Init) Stopped (System)
C:\WINNT\System32\drivers\vga.sys
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Wd33c93 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
wd90c24a (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
wdvga (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
weitek9 (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Xga (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process

```

IRQ and Port Report

Devices	Vector	Level	Affinity
MPS 1.4 - APIC platform	8	8	0x0000000f
MPS 1.4 - APIC platform	0	0	0x0000000f
MPS 1.4 - APIC platform	1	1	0x0000000f
MPS 1.4 - APIC platform	2	2	0x0000000f
MPS 1.4 - APIC platform	3	3	0x0000000f
MPS 1.4 - APIC platform	4	4	0x0000000f
MPS 1.4 - APIC platform	5	5	0x0000000f
MPS 1.4 - APIC platform	6	6	0x0000000f
MPS 1.4 - APIC platform	7	7	0x0000000f
MPS 1.4 - APIC platform	8	8	0x0000000f
MPS 1.4 - APIC platform	9	9	0x0000000f
MPS 1.4 - APIC platform	10	10	0x0000000f
MPS 1.4 - APIC platform	11	11	0x0000000f
MPS 1.4 - APIC platform	12	12	0x0000000f
MPS 1.4 - APIC platform	13	13	0x0000000f
MPS 1.4 - APIC platform	14	14	0x0000000f
MPS 1.4 - APIC platform	15	15	0x0000000f
MPS 1.4 - APIC platform	16	16	0x0000000f
MPS 1.4 - APIC platform	17	17	0x0000000f
MPS 1.4 - APIC platform	18	18	0x0000000f
MPS 1.4 - APIC platform	19	19	0x0000000f
MPS 1.4 - APIC platform	20	20	0x0000000f

MPS 1.4 - APIC platform	21	21	0x0000000f
MPS 1.4 - APIC platform	22	22	0x0000000f
MPS 1.4 - APIC platform	23	23	0x0000000f
MPS 1.4 - APIC platform	24	24	0x0000000f
MPS 1.4 - APIC platform	25	25	0x0000000f
MPS 1.4 - APIC platform	26	26	0x0000000f
MPS 1.4 - APIC platform	27	27	0x0000000f
MPS 1.4 - APIC platform	28	28	0x0000000f
MPS 1.4 - APIC platform	29	29	0x0000000f
MPS 1.4 - APIC platform	30	30	0x0000000f
MPS 1.4 - APIC platform	31	31	0x0000000f
MPS 1.4 - APIC platform	32	32	0x0000000f
MPS 1.4 - APIC platform	33	33	0x0000000f
MPS 1.4 - APIC platform	34	34	0x0000000f
MPS 1.4 - APIC platform	35	35	0x0000000f
MPS 1.4 - APIC platform	36	36	0x0000000f
MPS 1.4 - APIC platform	37	37	0x0000000f
MPS 1.4 - APIC platform	38	38	0x0000000f
MPS 1.4 - APIC platform	39	39	0x0000000f
MPS 1.4 - APIC platform	40	40	0x0000000f
MPS 1.4 - APIC platform	41	41	0x0000000f
MPS 1.4 - APIC platform	42	42	0x0000000f
MPS 1.4 - APIC platform	43	43	0x0000000f
MPS 1.4 - APIC platform	44	44	0x0000000f
MPS 1.4 - APIC platform	45	45	0x0000000f
MPS 1.4 - APIC platform	46	46	0x0000000f
MPS 1.4 - APIC platform	47	47	0x0000000f
MPS 1.4 - APIC platform	61	61	0x0000000f
MPS 1.4 - APIC platform	65	65	0x0000000f
MPS 1.4 - APIC platform	80	80	0x0000000f
MPS 1.4 - APIC platform	193	193	0x0000000f
MPS 1.4 - APIC platform	225	225	0x0000000f
MPS 1.4 - APIC platform	253	253	0x0000000f
MPS 1.4 - APIC platform	254	254	0x0000000f
MPS 1.4 - APIC platform	255	255	0x0000000f
i8042prt	1	1	0xffffffff
i8042prt	12	12	0xffffffff
Serial	4	4	0x00000000
Serial	3	3	0x00000000
E100B	28	28	0x6db6db6d
Floppy	6	6	0x00000000
atapi	0	14	0x00000000
dac960nt	32	32	0x00000000
dac960nt	32	32	0x00000000
dac960nt	32	32	0x00000000
dac960nt	32	32	0x00000000
dac960nt	32	32	0x00000000
dac960nt	32	32	0x00000000
dac960nt	32	32	0x00000000
dac960nt	32	32	0x00000000
Sym_hi	40	40	0x00000000
Sym_hi	41	41	0x00000000

Devices	Physical Address	Length
MPS 1.4 - APIC platform	0x00000000	0x000000010
MPS 1.4 - APIC platform	0x00000020	0x000000002
MPS 1.4 - APIC platform	0x00000040	0x000000004
MPS 1.4 - APIC platform	0x00000048	0x000000004

MPS 1.4 - APIC platform	0x00000061	0x0000000001
MPS 1.4 - APIC platform	0x00000070	0x0000000002
MPS 1.4 - APIC platform	0x00000080	0x0000000010
MPS 1.4 - APIC platform	0x00000092	0x0000000001
MPS 1.4 - APIC platform	0x000000a0	0x0000000002
MPS 1.4 - APIC platform	0x000000c0	0x0000000010
MPS 1.4 - APIC platform	0x000000f0	0x0000000010
i8042prt	0x00000060	0x0000000001
i8042prt	0x00000064	0x0000000001
Parport	0x00000378	0x0000000003
Serial	0x000003f8	0x0000000007
Serial	0x000002f8	0x0000000007
Floppy	0x000003f0	0x0000000006
Floppy	0x000003f7	0x0000000001
atapi	0x000001f0	0x0000000008
atapi	0x000003f6	0x0000000001
dac960nt	0x00003000	0x0000000080
dac960nt	0x00004000	0x0000000080
dac960nt	0x00006000	0x0000000080
dac960nt	0x00007000	0x0000000080
dac960nt	0x00008000	0x0000000080
dac960nt	0x00009000	0x0000000080
dac960nt	0x0000a000	0x0000000080
dac960nt	0x0000b000	0x0000000080
Sym_hi	0x00002000	0x0000000100
Sym_hi	0x00002400	0x0000000100
cirrus	0x000003b0	0x000000000c
cirrus	0x000003c0	0x0000000020

DMA and Memory Report

```
-----
Devices          Channel   Port
-----
Floppy           2         0
-----
```

```
-----
Devices          Physical Address  Length
-----
MPS 1.4 - APIC platform 0xfec10000 0x00000400
MPS 1.4 - APIC platform 0xfec10000 0x00000400
E100B             0xef200000 0x0000001e
dac960nt         0xe6110000 0x00000080
dac960nt         0xea000000 0x02000000
dac960nt         0xe6210000 0x00000080
dac960nt         0xec000000 0x02000000
dac960nt         0xee110000 0x00000080
dac960nt         0xf0000000 0x02000000
dac960nt         0xee210000 0x00000080
dac960nt         0xf2000000 0x02000000
dac960nt         0xee310000 0x00000080
dac960nt         0xf4000000 0x02000000
dac960nt         0xf6110000 0x00000080
dac960nt         0xf8000000 0x02000000
dac960nt         0xf6210000 0x00000080
dac960nt         0xfa000000 0x02000000
dac960nt         0xf6310000 0x00000080
dac960nt         0xfc000000 0x02000000
Sym_hi           0xe6004000 0x00000400
-----
```

```
Sym_hi           0xe6000000 0x00002000
Sym_hi           0xe6004400 0x00000400
Sym_hi           0xe6002000 0x00002000
cirrus           0x000a0000 0x00020000
cirrus           0xe8000000 0x01000000
```

Environment Report

System Environment Variables

```
ComSpec=C:\WINNT\system32\cmd.exe
HOME=C:/
NTRESKIT=Z:\NTRESKIT
NUMBER_OF_PROCESSORS=4
OS=Windows_NT
Os2LibPath=C:\WINNT\system32\os2\dll;
```

```
Path=C:\MKS\mksnt;C:\WINNT\system32;C:\WINNT;Z:\NTRESKIT;Z:\NTRESKIT\Perl;
z:\emon\bin;C:\MSSQL7\BINN
PROCESSOR_ARCHITECTURE=x86
PROCESSOR_IDENTIFIER=x86 Family 6 Model 5 Stepping 3, GenuineIntel
PROCESSOR_LEVEL=6
PROCESSOR_REVISION=0503
ROOTDIR=C:/MKS
SHELL=C:/MKS/mksnt/sh.exe
TMPDIR=C:/TEMP
windir=C:\WINNT
```

Environment Variables for Current User

```
TEMP=C:\TEMP
TMP=C:\TEMP
```

Network Report

```
-----
Your Access Level: Admin & Local
Workgroup or Domain: WORKGROUP
Network Version: 4.0
LanRoot: WORKGROUP
Logged On Users: 1
Current User (1): Administrator
  Logon Domain: AVALON4
  Logon Server: AVALON4

Transport: Nbf_E100B1, 00-A0-C9-C5-45-C4, VC's: 1, Wan: Wan

Character Wait: 3,600
Collection Time: 250
Maximum Collection Count: 16
Keep Connection: 600
Maximum Commands: 5
Session Time Out: 45
Character Buffer Size: 512
```

Maximum Threads: 17
 Lock Quota: 6,144
 Lock Increment: 10
 Maximum Locks: 500
 Pipe Increment: 10
 Maximum Pipes: 500
 Cache Time Out: 40
 Dormant File Limit: 45
 Read Ahead Throughput: 4,294,967,295
 Mailslot Buffers: 3
 Server Announce Buffers: 20
 Illegal Datagrams: 5
 Datagram Reset Frequency: 60
 Log Election Packets: False
 Use Opportunistic Locking: True
 Use Unlock Behind: True
 Use Close Behind: True
 Buffer Pipes: True
 Use Lock, Read, Unlock: True
 Use NT Caching: True
 Use Raw Read: True
 Use Raw Write: True
 Use Write Raw Data: True
 Use Encryption: True
 Buffer Deny Write Files: True
 Buffer Read Only Files: True
 Force Core Creation: True
 512 Byte Max Transfer: False
 Bytes Received: 429,484
 SMB's Received: 405
 Paged Read Bytes Requested: 409,600
 Non Paged Read Bytes Requested: 366,019
 Cache Read Bytes Requested: 366,019
 Network Read Bytes Requested: 377,271
 Bytes Transmitted: 406,770
 SMB's Transmitted: 405
 Paged Read Bytes Requested: 0
 Non Paged Read Bytes Requested: 374,423
 Cache Read Bytes Requested: 0
 Network Read Bytes Requested: 373,479
 Initially Failed Operations: 0
 Failed Completion Operations: 0
 Read Operations: 17
 Random Read Operations: 1
 Read SMB's: 15
 Large Read SMB's: 8
 Small Read SMB's: 0
 Write Operations: 14
 Random Write Operations: 0
 Write SMB's: 93
 Large Write SMB's: 8
 Small Write SMB's: 0
 Raw Reads Denied: 0
 Raw Writes Denied: 0
 Network Errors: 0
 Sessions: 6
 Failed Sessions: 0
 Reconnects: 0
 Core Connects: 0

LM 2.0 Connects: 0
 LM 2.x Connects: 0
 Windows NT Connects: 4
 Server Disconnects: 2
 Hung Sessions: 0
 Use Count: 3
 Failed Use Count: 0
 Current Commands: 0
 Server File Opens: 12
 Server Device Opens: 0
 Server Jobs Queued: 0
 Server Session Opens: 0
 Server Sessions Timed Out: 0
 Server Sessions Errored Out: 0
 Server Password Errors: 0
 Server Permission Errors: 0
 Server System Errors: 0
 Server Bytes Sent: 6,947
 Server Bytes Received: 8,321
 Server Average Response Time: 0
 Server Request Buffers Needed: 0
 Server Big Buffers Needed: 0

NT Server Registry Information

Software\Microsoft

Key Name: SOFTWARE\Microsoft\MMC
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 11:42 AM

Key Name: SOFTWARE\Microsoft\MMC\NodeTypes
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 11:42 AM

Key Name: SOFTWARE\Microsoft\MMC\Settings
 Class Name: <NO CLASS>
 Last Write Time: 9/4/98 - 8:39 AM

Value 0
 Name: Help File Index
 Type: REG_DWORD
 Data: 0x2

Key Name: SOFTWARE\Microsoft\MMC\SnapIns
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 11:42 AM

11d0- Key Name: SOFTWARE\Microsoft\MMC\SnapIns\{00100100-1816-
 8EF5-00AA0062C58F}
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:13 PM
 Value 0
 Name: About
 Type: REG_SZ

Data: {00100101-1816-11d0-8EF5-00AA0062C58F}

Value 1
 Name: NameString
 Type: REG_SZ
 Data: Microsoft SQL Enterprise Manager

11d0- Key Name: SOFTWARE\Microsoft\MMC\SnapIns\{00100100-1816-8EF5-00AA0062C58F}\About
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:13 PM
 Value 0
 Name: <NO NAME>
 Type: REG_SZ
 Data: {00100101-1816-11d0-8EF5-00AA0062C58F}

11d0- Key Name: SOFTWARE\Microsoft\MMC\SnapIns\{00100100-1816-8EF5-00AA0062C58F}\NameString
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:13 PM
 Value 0
 Name: <NO NAME>
 Type: REG_SZ
 Data: Microsoft SQL Enterprise Manager

11d0- Key Name: SOFTWARE\Microsoft\MMC\SnapIns\{00100100-1816-8EF5-00AA0062C58F}\NodeTypes
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:13 PM

11d0- Key Name: SOFTWARE\Microsoft\MMC\SnapIns\{00100100-1816-8EF5-00AA0062C58F}\NodeTypes\{00100200-1816-F5-00AA0062C58F}
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:13 PM

11d0- Key Name: SOFTWARE\Microsoft\MMC\SnapIns\{00100100-1816-8EF5-00AA0062C58F}\Standalone
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:13 PM

Key Name: SOFTWARE\Microsoft\MSDTC
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:13 PM
 Value 0
 Name: MaxLogSize
 Type: REG_DWORD
 Data: 0x200

Key Name: SOFTWARE\Microsoft\MSDTC\Setup
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:13 PM
 Value 0
 Name: InstallCode
 Type: REG_DWORD
 Data: 0

Value 1
 Name: InstallState
 Type: REG_DWORD
 Data: 0x1

Value 2
 Name: MajorVersion
 Type: REG_DWORD
 Data: 0x20000

Value 3
 Name: MinorVersion
 Type: REG_DWORD
 Data: 0x2f8

Key Name: SOFTWARE\Microsoft\MSDTC\Setup\ExitStatus
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:13 PM
 Value 0
 Name: CompletionComment
 Type: REG_SZ
 Data: Source = DtcComplete, ExitType = Success, Successfulfu
 1 Install

Value 1
 Name: ErrorCode
 Type: REG_DWORD
 Data: 0

Value 2
 Name: ExitCode
 Type: REG_DWORD
 Data: 0

Value 3
 Name: Source
 Type: REG_DWORD
 Data: 0x1

Key Name: SOFTWARE\Microsoft\MSSQLServer
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 1:00 PM

Key Name: SOFTWARE\Microsoft\MSSQLServer\Client
 Class Name: <NO CLASS>

Last Write Time: 9/2/98 - 3:13 PM

Key Name: SOFTWARE\Microsoft\MicrosoftSQLServer\Client\ConnectTo
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:57 PM

Value 0
 Name: DSQUERY
 Type: REG_SZ
 Data: DBMSSOCN

Key Name: SOFTWARE\Microsoft\MicrosoftSQLServer\Client\DB-Lib
 Class Name: <NO CLASS>
 Last Write Time: 9/21/98 - 3:37 PM

Value 0
 Name: AutoAnsiToOem
 Type: REG_SZ
 Data: on

Key Name: SOFTWARE\Microsoft\MicrosoftSQLServer\Client\TDS
 Class Name: <NO CLASS>
 Last Write Time: 9/4/98 - 4:50 PM

Value 0
 Name: <NO NAME>
 Type: REG_SZ
 Data: 7.0

Value 1
 Name: .
 Type: REG_SZ
 Data: 7.0

Value 2
 Name: Avalon4
 Type: REG_SZ
 Data: 7.0

Key Name: SOFTWARE\Microsoft\MicrosoftSQLServer\ClientSetup
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:16 PM

Value 0
 Name: SQLPath
 Type: REG_SZ
 Data: C:\MSSQL7

Key Name: SOFTWARE\Microsoft\MicrosoftSQLServer\MSSQLServer
 Class Name: <NO CLASS>
 Last Write Time: 9/2/98 - 3:16 PM

Value 0
 Name: AuditLevel
 Type: REG_DWORD
 Data: 0

Value 1
 Name: BackupDirectory
 Type: REG_SZ

Data: C:\MSSQL7\BACKUP

Value 2
 Name: DefaultCompStyle
 Type: REG_SZ
 Data: 0

Value 3
 Name: DefaultDomain
 Type: REG_SZ
 Data: AVALON4

Value 4
 Name: DefaultLocaleID
 Type: REG_SZ
 Data: 8200

Value 5
 Name: DefaultLogin
 Type: REG_SZ
 Data: guest

Value 6
 Name: DefaultSortID
 Type: REG_SZ
 Data: 50

Value 7
 Name: ListenOn
 Type: REG_MULTI_SZ
 Data: SSNMPN70, \\.\pipe\sql\query
 SSMSSO70,1433

Value 8
 Name: LoginMode
 Type: REG_DWORD
 Data: 0

Value 9
 Name: Map#
 Type: REG_SZ
 Data: -

Value 10
 Name: Map\$
 Type: REG_SZ
 Data:

Value 11
 Name: Map_
 Type: REG_SZ
 Data: \

Value 12
 Name: ResourceMgrID
 Type: REG_SZ
 Data: {4002167A-42B2-11D2-B9ED-00A0C9C545C4}

```

Value 13
  Name:      RWSTListenAddress
  Type:      REG_SZ
  Data:

Value 14
  Name:      SetHostName
  Type:      REG_DWORD
  Data:      0

Value 15
  Name:      Tapeloadwaittime
  Type:      REG_DWORD
  Data:      0xffffffff

Key Name:
SOFTWARE\Microsoft\MSSQLServer\MSSQLServer\CurrentVersion
  Class Name:      <NO CLASS>
  Last Write Time: 9/2/98 - 3:13 PM
  Value 0
    Name:      checksum
    Type:      REG_BINARY
    Data:
00000000 36 35 32 32 63 31 35 63 - 32 37 64 65 64 62 32 38
6522c15c
27dedb28
00000010 61 61 30 33 34 34 64 64 - 31 31 36 38 65 32 66 38
aa0344dd
1168e2f8
00000020 35 31 39 35 62 34 30 62 - 65 39 34 31 62 31 32 35
5195b40b
e941b125
00000030 62 37 66 32 64 36 39 36 - 36 36 34 62 64 63 63 32
b7f2d696
664bdcc2
00000040 31 31 30 62 36 36 65 30 - 34 31 38 61 31 63 33 39
110b66e0
418a1c39
00000050 38 30 65 33 61 34 66 63 - 63 36 65 34 38 64 38 38
80e3a4fc
c6e48d88
00000060 37 32 35 32 32 66 31 38 - 63 63 31 34 34 35 38 61
72522f18
cc14458a
00000070 62 31 65 34 37 36 39 63 - 36 66 33 32 66 65 32 38
b1e4769c
6f32fe28
00000080 31 64 65 36 00
1de6.

Value 1
  Name:      CurrentVersion
  Type:      REG_SZ
  Data:      7.00.549

Value 2
  Name:      RegisteredOwner
  Type:

```

```

Data:      SAM&M

Value 3
  Name:      SerialNumber
  Type:      REG_DWORD
  Data:      0x81530040

Key Name:
SOFTWARE\Microsoft\MSSQLServer\MSSQLServer\Parameters
  Class Name:      <NO CLASS>
  Last Write Time: 9/2/98 - 3:13 PM
  Value 0
    Name:      SQLArg0
    Type:      REG_SZ
    Data:      -dC:\MSSQL7\data\master.mdf

Value 1
  Name:      SQLArg1
  Type:      REG_SZ
  Data:      -eC:\MSSQL7\log\ERRORLOG

Value 2
  Name:      SQLArg2
  Type:      REG_SZ
  Data:      -lC:\MSSQL7\data\mastlog.ldf

Key Name:
SOFTWARE\Microsoft\MSSQLServer\MSSQLServer\RPCNetLib
  Class Name:      <NO CLASS>
  Last Write Time: 9/2/98 - 3:13 PM
  Value 0
    Name:      Security
    Type:      REG_SZ
    Data:

Key Name:      SOFTWARE\Microsoft\MSSQLServer\Providers
  Class Name:      <NO CLASS>
  Last Write Time: 9/2/98 - 3:16 PM
  Value 0
    Name:      AllowInProcess
    Type:      REG_DWORD
    Data:      0x1

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Providers\ADSDSOObject
  Class Name:      <NO CLASS>
  Last Write Time: 9/2/98 - 3:16 PM
  Value 0
    Name:      AllowInProcess
    Type:      REG_DWORD
    Data:      0x1

```

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Providers\DTSPackage
DSO
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: AllowInProcess
Type: REG_DWORD
Data: 0x1

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Providers\Microsoft.
Jet.OLEDB.4.0
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: AllowInProcess
Type: REG_DWORD
Data: 0x1

Key Name: SOFTWARE\Microsoft\MSSQLServer\Providers\MSDAORA
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: AllowInProcess
Type: REG_DWORD
Data: 0x1

Key Name: SOFTWARE\Microsoft\MSSQLServer\Providers\MSDASQL
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: AllowInProcess
Type: REG_DWORD
Data: 0x1

Key Name: SOFTWARE\Microsoft\MSSQLServer\Providers\MSIDX
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: AllowInProcess
Type: REG_DWORD
Data: 0x1

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Providers\MSQLImpPro
v
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: AllowInProcess
Type: REG_DWORD
Data: 0x1

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Providers\MSSEARCHSQ
L
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: AllowInProcess
Type: REG_DWORD
Data: 0x1

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Providers\SQLOLEDB
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: AllowInProcess
Type: REG_DWORD
Data: 0x1

Key Name: SOFTWARE\Microsoft\MSSQLServer\Replication
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 1:00 PM

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Replication\MergeRep
licationProvider
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 1:00 PM

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Replication\MergeRep
licationProvider\7.0
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 1:00 PM

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Replication\MergeRep
licationProvider\7.0\MsJet
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:20 PM
Value 0
Name: <NO NAME>
Type: REG_SZ
Data: {f159cf30-0db4-11d1-b272-00aa00b8de95}

Key Name: SOFTWARE\Microsoft\MSSQLServer\Setup
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:13 PM
Value 0
Name: SourcePath
Type: REG_SZ
Data: Z:\Sql70549.03p\x86\Data

Value 1

Name: SQLDataRoot
Type: REG_SZ
Data: C:\MSSQL7

Value 2
Name: SQLPath
Type: REG_SZ
Data: C:\MSSQL7

Key Name: SOFTWARE\Microsoft\MSSQLServer\SQL Service
Manager
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: Action Verify
Type: REG_DWORD
Data: 0

Value 1
Name: DefaultSvc
Type: REG_SZ
Data: MSSQLServer

Value 2
Name: Remote
Type: REG_DWORD
Data: 0x1

Value 3
Name: Services
Type: REG_MULTI_SZ
Data: MSSQLServer
SQLServerAgent
MSDTC

Key Name: SOFTWARE\Microsoft\MSSQLServer\SQLEW
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM

Key Name: SOFTWARE\Microsoft\MSSQLServer\SQLEW\Replication
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: PerfmonFile
Type: REG_SZ
Data: C:\MSSQL7\BINN\REPLMON.PMC

Key Name: SOFTWARE\Microsoft\MSSQLServer\SQLEW\Wizards
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: Web Assistant
Type: REG_SZ
Data: C:\MSSQL7\BINN\semwebwz.DLL^WebWizardEntry

Key Name: SOFTWARE\Microsoft\MSSQLServer\SQLServerAgent
Class Name: <NO CLASS>
Last Write Time: 9/2/98 - 3:16 PM
Value 0
Name: DownloadedMaxRows
Type: REG_DWORD
Data: 0x64

Value 1
Name: ErrorLogFile
Type: REG_SZ
Data: C:\MSSQL7\LOG\SQLAGENT.OUT

Value 2
Name: ErrorLoggingLevel
Type: REG_DWORD
Data: 0x3

Value 3
Name: JobHistoryMaxRows
Type: REG_DWORD
Data: 0x3e8

Value 4
Name: JobHistoryMaxRowsPerJob
Type: REG_DWORD
Data: 0x64

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

Value 6
Name: MSXServerName
Type: REG_SZ
Data:

Value 7
Name: NonAlertableErrors
Type: REG_SZ
Data: 1204,4002

Value 8
Name: RestartSQLServer
Type: REG_DWORD
Data: 0x1

Value 9
Name: ServerHost
Type: REG_SZ
Data:

Value 10
Name: WorkingDirectory
Type: REG_SZ
Data: C:\MSSQL7\JOBS

Key Name: SOFTWARE\Microsoft\Microsoft SQL Server\SQL Server Agent\Subsystems

Class Name: <NO CLASS>

Last Write Time: 9/2/98 - 3:16 PM

Value 0

Name: ActiveScripting

Type: REG_SZ

Data: C:\MSSQL7\BINN\SQLATXSS.DLL,NULL,ActiveScriptStart,ActiveScriptEvent,ActiveScriptStop,10

Value 1

Name: CmdExec

Type: REG_SZ

Data: C:\MSSQL7\BINN\SQLCMDSS.DLL,NULL,CmdExecStart,CmdEvent,CmdExecStop,10

Value 2

Name: Distribution

Type: REG_SZ

Data: C:\MSSQL7\BINN\SQLREPS.DLL,C:\MSSQL7\BINN\DISTRIB.EXE,ReplStart,ReplEvent,ReplStop,100

Value 3

Name: LogReader

Type: REG_SZ

Data: C:\MSSQL7\BINN\SQLREPS.DLL,C:\MSSQL7\BINN\LOGREAD.EXE,ReplStart,ReplEvent,ReplStop,25

Value 4

Name: Merge

Type: REG_SZ

Data: C:\MSSQL7\BINN\SQLREPS.DLL,C:\MSSQL7\BINN\REPLMERG.EXE,ReplStart,ReplEvent,ReplStop,100

Value 5

Name: Snapshot

Type: REG_SZ

Data: C:\MSSQL7\BINN\SQLREPS.DLL,C:\MSSQL7\BINN\SNAPSHOT.EXE,ReplStart,ReplEvent,ReplStop,100

Software\Intel\E100B

Key Name: SOFTWARE\Intel\E100B

Class Name: GenericClass

Last Write Time: 6/10/98 - 4:01 AM

Key Name: SOFTWARE\Intel\E100B\CurrentVersion

Class Name: GenericClass

Last Write Time: 6/10/98 - 4:01 AM

Value 0

Name: Description

Type: REG_SZ

Data: Intel EtherExpress PRO Adapter Driver

Value 1

Name: InstallDate

Type: REG_DWORD

Data: 0x357ec9fd

Value 2

Name: MajorVersion

Type: REG_DWORD

Data: 0x3

Value 3

Name: MinorVersion

Type: REG_DWORD

Data: 0

Value 4

Name: RefCount

Type: REG_DWORD

Data: 0x1

Value 5

Name: ServiceName

Type: REG_SZ

Data: E100B

Value 6

Name: SoftwareType

Type: REG_SZ

Data: driver

Value 7

Name: Title

Type: REG_SZ

Data: Intel EtherExpress PRO Adapter

Key Name: SOFTWARE\Intel\E100B\CurrentVersion\NetRules

Class Name: GenericClass

Last Write Time: 6/10/98 - 4:01 AM

Value 0

Name: bindable

Type: REG_MULTI_SZ

Data: E100Bdriver E100BAdapter non exclusive 100

Value 1

Name: bindform

Type: REG_SZ

Data: "E100BSys" yes no container

Value 2

Name: class

Type: REG_MULTI_SZ

Data: E100Bdriver basic

Value 3
 Name: InfName
 Type: REG_SZ
 Data: oemnad7.inf

Value 4
 Name: InfOption
 Type: REG_SZ
 Data: E100BEXP

Value 5
 Name: type
 Type: REG_SZ
 Data: E100BSys ndisDriver E100BDriver

Value 6
 Name: use
 Type: REG_SZ
 Data: driver

Services/dac960nt

Key Name: SYSTEM\CurrentControlSet\Services\dac960nt
 Class Name: <NO CLASS>
 Last Write Time: 6/17/98 - 6:48 PM

Value 0
 Name: ErrorControl
 Type: REG_DWORD
 Data: 0x1

Value 1
 Name: Group
 Type: REG_SZ
 Data: SCSI miniport

Value 2
 Name: ImagePath
 Type: REG_EXPAND_SZ
 Data: System32\drivers\dac960nt.sys

Value 3
 Name: PlugPlayServiceType
 Type: REG_DWORD
 Data: 0x1

Value 4
 Name: RequestedSystemResources
 Type: REG_RESOURCE_REQUIREMENTS_LIST
 Data:
 Interface Type: Internal
 Bus Number: 0
 Slot Number: 0
 List 0
 Descriptor 0
 Resource: Interrupt

Option: 0x00000000
 Disposition: Shared
 Type: Level Sensitive
 Minimum Vector: 0x20
 Maximum Vector: 0x20

Descriptor 1
 Resource: Memory
 Option: 0x00000001
 Disposition: Device Exclusive
 Type: Write Only
 Length: 0x2000
 Alignment: 0x2000
 Minimum Address: 0xfe606000
 Maximum Address: 0xfe607fff

Descriptor 2
 Resource: Memory
 Option: 0x00000009
 Disposition: Device Exclusive
 Type: Write Only
 Length: 0x2000
 Alignment: 0x2000
 Minimum Address: 0xfe606000
 Maximum Address: 0xfe607fff

Value 5
 Name: Start
 Type: REG_DWORD
 Data: 0

Value 6
 Name: Tag
 Type: REG_DWORD
 Data: 0x63

Value 7
 Name: Type
 Type: REG_DWORD
 Data: 0x1

Key Name: SYSTEM\CurrentControlSet\Services\dac960nt\Enum
 Class Name: <NO CLASS>
 Last Write Time: 6/17/98 - 6:46 PM

Value 0
 Name: 0
 Type: REG_SZ
 Data: Root\SCSIADAPTER\OEM1.INF&DAC960NT

Value 1
 Name: Count
 Type: REG_DWORD
 Data: 0x1

Value 2
 Name: NextInstance

Type: REG_DWORD
Data: 0x1

Services/Disk

Key Name: SYSTEM\CurrentControlSet\Services\Disk
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: DependOnGroup
Type: REG_MULTI_SZ
Data: SCSI miniport

Value 1
Name: ErrorControl
Type: REG_DWORD
Data: 0

Value 2
Name: Group
Type: REG_SZ
Data: SCSI Class

Value 3
Name: Start
Type: REG_DWORD
Data: 0

Value 4
Name: Tag
Type: REG_DWORD
Data: 0x2

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

Key Name: SYSTEM\CurrentControlSet\Services\Disk\Enum
Class Name: <NO CLASS>
Last Write Time: 6/17/98 - 6:46 PM
Value 0
Name: 0
Type: REG_SZ
Data: Root\LEGACY_DISK\0000

Value 1
Name: Count
Type: REG_DWORD
Data: 0x1

Value 2
Name: NextInstance
Type: REG_DWORD
Data: 0x1

Services/Diskperf

Key Name: SYSTEM\CurrentControlSet\Services\Diskperf
Class Name: <NO CLASS>
Last Write Time: 6/17/98 - 1:34 PM
Value 0
Name: ErrorControl
Type: REG_DWORD
Data: 0x1

Value 1
Name: Group
Type: REG_SZ
Data: Filter

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

Value 3
Name: Tag
Type: REG_DWORD
Data: 0x4

Value 4
Name: Type
Type: REG_DWORD
Data: 0x1

Key Name: SYSTEM\CurrentControlSet\Services\Diskperf\Enum
Class Name: <NO CLASS>
Last Write Time: 6/17/98 - 6:46 PM
Value 0
Name: 0
Type: REG_SZ
Data: Root\LEGACY_DISKPERF\0000

Value 1
Name: Count
Type: REG_DWORD
Data: 0x1

Value 2
Name: NextInstance
Type: REG_DWORD
Data: 0x1

Services\E100B

Key Name: SYSTEM\CurrentControlSet\Services\E100B

```

Class Name: <NO CLASS>
Last Write Time: 6/17/98 - 6:48 PM
Value 0
  Name: DisplayName
  Type: REG_SZ
  Data: Intel EtherExpress PRO Adapter

Value 1
  Name: ErrorControl
  Type: REG_DWORD
  Data: 0x1

Value 2
  Name: Group
  Type: REG_SZ
  Data: NDIS

Value 3
  Name: ImagePath
  Type: REG_EXPAND_SZ
  Data: \SystemRoot\System32\drivers\e100bnt.sys

Value 4
  Name: RequestedSystemResources
  Type: REG_RESOURCE_REQUIREMENTS_LIST
  Data:
    Interface Type: Internal
    Bus Number: 0
    Slot Number: 0
    List 0
      Descriptor 0
        Resource: Interrupt
        Option: 0x00000000
        Disposition: Shared
        Type: Level Sensitive
        Minimum Vector: 0x1c
        Maximum Vector: 0x1c

      Descriptor 1
        Resource: Memory
        Option: 0x00000001
        Disposition: Device Exclusive
        Type: Write Only
        Length: 0x1000
        Alignment: 0x1000
        Minimum Address: 0xfe306000
        Maximum Address: 0xfe306fff

      Descriptor 2
        Resource: Memory
        Option: 0x00000009
        Disposition: Device Exclusive
        Type: Write Only
        Length: 0x1000
        Alignment: 0x1000
        Minimum Address: 0xfe306000
        Maximum Address: 0xfe306fff

      Descriptor 3

```

```

Resource: Memory
Option: 0x00000008
Disposition: Device Exclusive
Type: Write Only
Length: 0x1000
Alignment: 0x1000
Minimum Address: 0xfe000000
Maximum Address: 0xfe0fffff

Descriptor 4
Resource: Port
Option: 0x00000001
Disposition: Device Exclusive
Type: Port
Length: 0x20
Alignment: 0x20
Minimum Address: 0x00003000
Maximum Address: 0x0000301f

Descriptor 5
Resource: Port
Option: 0x00000008
Disposition: Device Exclusive
Type: Port
Length: 0x20
Alignment: 0x20
Minimum Address: 0x00003000
Maximum Address: 0x0000301f

Descriptor 6
Resource: Memory
Option: 0x00000001
Disposition: Device Exclusive
Type: Read / Write
Length: 0x100000
Alignment: 0x100000
Minimum Address: 0xfe000000
Maximum Address: 0xfe0fffff

Descriptor 7
Resource: Memory
Option: 0x00000008
Disposition: Device Exclusive
Type: Read / Write
Length: 0x100000
Alignment: 0x100000
Minimum Address: 0xfe000000
Maximum Address: 0xfe0fffff

```

```

Value 5
  Name: Start
  Type: REG_DWORD
  Data: 0x2

Value 6
  Name: Type
  Type: REG_DWORD

```

```

Data:          0x1

Key Name:      SYSTEM\CurrentControlSet\Services\E100B\Enum
Class Name:    <NO CLASS>
Last Write Time: 6/17/98 - 6:46 PM
Value 0
  Name:        0
  Type:        REG_SZ
  Data:        Root\LEGACY_E100B\0000

Value 1
  Name:        Count
  Type:        REG_DWORD
  Data:        0x1

Value 2
  Name:        NextInstance
  Type:        REG_DWORD
  Data:        0x1

Key Name:      SYSTEM\CurrentControlSet\Services\E100B\Linkage
Class Name:    GenericClass
Last Write Time: 6/10/98 - 4:07 AM
Value 0
  Name:        Bind
  Type:        REG_MULTI_SZ
  Data:        \Device\E100B1

Value 1
  Name:        Export
  Type:        REG_MULTI_SZ
  Data:        \Device\E100B1

Value 2
  Name:        Route
  Type:        REG_MULTI_SZ
  Data:        "E100B1"

Key Name:      SYSTEM\CurrentControlSet\Services\E100B\Linkage\Disabled
Class Name:    GenericClass
Last Write Time: 6/10/98 - 4:07 AM
Value 0
  Name:        Bind
  Type:        REG_MULTI_SZ
  Data:

Value 1
  Name:        Export
  Type:        REG_MULTI_SZ
  Data:

```

```

Value 2
  Name:        Route
  Type:        REG_MULTI_SZ
  Data:

Key Name:      SYSTEM\CurrentControlSet\Services\E100B\Parameters
Class Name:    GenericClass
Last Write Time: 6/10/98 - 4:01 AM

Key Name:      SYSTEM\CurrentControlSet\Services\E100B\Security
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 4:01 AM
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  73 00 00 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
s.....
00000060  00 00 00 05 20 00 00 00 - 23 02 00 00 00 00 b6 80 ....
...
#.....
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 - 00 00 b6 80 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 00 00 b6 80 - 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 -
.....

```

Services\E100B1

Key Name: SYSTEM\CurrentControlSet\Services\E100B1
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:01 AM

Value 0
Name: ErrorControl
Type: REG_DWORD
Data: 0x1

Value 1
Name: Start
Type: REG_DWORD
Data: 0x3

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

Key Name: SYSTEM\CurrentControlSet\Services\E100B1\Linkage
Class Name: GenericClass
Last Write Time: 6/17/98 - 6:48 PM

Value 0
Name: Bind
Type: REG_MULTI_SZ
Data: \Device\E100B1

Value 1
Name: Export
Type: REG_MULTI_SZ
Data: \Device\E100B1

Value 2
Name: Route
Type: REG_MULTI_SZ
Data: "E100B1"

Key Name: SYSTEM\CurrentControlSet\Services\E100B1\Linkage\Disabled
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:01 AM

Key Name: SYSTEM\CurrentControlSet\Services\E100B1\Parameters
Class Name: GenericClass
Last Write Time: 6/12/98 - 11:46 AM
Value 0
Name: Adaptive_IFS
Type: REG_DWORD
Data: 0x1

Value 1
Name: BusNumber
Type: REG_DWORD
Data: 0x3

Value 2
Name: BusType
Type: REG_DWORD
Data: 0x5

Value 3
Name: BusTypeLocal
Type: REG_DWORD
Data: 0x5

Value 4
Name: Eid
Type: REG_DWORD
Data: 0xc9c545c4

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

Value 6
Name: MapRegisters
Type: REG_DWORD
Data: 0x40

Value 7
Name: MediaType
Type: REG_DWORD
Data: 0x1

Value 8
Name: MsPciScan
Type: REG_DWORD
Data: 0x1

Value 9
Name: NetworkAddress
Type: REG_SZ
Data: 0

Value 10
Name: NumCoalesce
Type: REG_DWORD
Data: 0x10

Value 11
Name: NumRfd
Type: REG_DWORD
Data: 0x40

Value 12
Name: NumTbdPerTcb
Type: REG_DWORD

Data: 0xc

Value 13
 Name: NumTcb
 Type: REG_DWORD
 Data: 0x20

Value 14
 Name: PerfOptims
 Type: REG_DWORD
 Data: 0x2

Value 15
 Name: ProposeIAFTAddress
 Type: REG_SZ
 Data: 00A0C9C545C4

Value 16
 Name: RxDmaCount
 Type: REG_DWORD
 Data: 0

Value 17
 Name: RxFifo
 Type: REG_DWORD
 Data: 0x8

Value 18
 Name: SlotNumber
 Type: REG_DWORD
 Data: 0x7

Value 19
 Name: Speed
 Type: REG_DWORD
 Data: 0x64

Value 20
 Name: Threshold
 Type: REG_DWORD
 Data: 0x10

Value 21
 Name: TxDmaCount
 Type: REG_DWORD
 Data: 0

Value 22
 Name: TxFifo
 Type: REG_DWORD
 Data: 0x8

Value 23
 Name: Txmitwait
 Type: REG_DWORD
 Data: 0x1

Value 24
 Name: UcodeSW

Type: REG_DWORD
 Data: 0x1

Value 25
 Name: UnderrunRetry
 Type: REG_DWORD
 Data: 0x1

Key Name:
 SYSTEM\CurrentControlSet\Services\E100B1\Parameters

\Tcpip
 Class Name: GenericClass
 Last Write Time: 6/17/98 - 6:48 PM

Value 0
 Name: DefaultGateway
 Type: REG_MULTI_SZ
 Data:

Value 1
 Name: EnableDHCP
 Type: REG_DWORD
 Data: 0

Value 2
 Name: IPAddress
 Type: REG_MULTI_SZ
 Data: 192.168.91.214

Value 3
 Name: IPInterfaceContext
 Type: REG_DWORD
 Data: 0x1

Value 4
 Name: IPInterfaceContextMax
 Type: REG_DWORD
 Data: 0x1

Value 5
 Name: LLInterface
 Type: REG_SZ
 Data:

Value 6
 Name: PPTPFiltering
 Type: REG_DWORD
 Data: 0

Value 7
 Name: RawIPAllowedProtocols
 Type: REG_MULTI_SZ
 Data: 0

Value 8
 Name: SubnetMask

Type: REG_MULTI_SZ
Data: 255.255.255.0

Value 9
Name: TCPAllowedPorts
Type: REG_MULTI_SZ
Data: 0

Value 10
Name: UDPAllowedPorts
Type: REG_MULTI_SZ
Data: 0

Value 11
Name: UseZeroBroadcast
Type: REG_DWORD
Data: 0

Services\macdisk

Key Name: SYSTEM\CurrentControlSet\Services\macdisk
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 11:57 AM

Value 0
Name: ErrorControl
Type: REG_DWORD
Data: 0x1

Value 1
Name: Group
Type: REG_SZ
Data: Filter

Value 2
Name: ImagePath
Type: REG_EXPAND_SZ
Data: System32\drivers\macdisk.sys

Value 3
Name: PlugPlayServiceType
Type: REG_DWORD
Data: 0x1

Value 4
Name: Start
Type: REG_DWORD
Data: 0

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

Value 6
Name: Type
Type: REG_DWORD
Data: 0x1

Key Name: SYSTEM\CurrentControlSet\Services\macdisk\Enum
Class Name: <NO CLASS>
Last Write Time: 6/17/98 - 6:46 PM

Value 0
Name: 0
Type: REG_SZ
Data: Root\SCSIADAPTER\OEM2.INF&MACDISK

Value 1
Name: Count
Type: REG_DWORD
Data: 0x1

Value 2
Name: NextInstance
Type: REG_DWORD
Data: 0x1

Key Name: SYSTEM\CurrentControlSet\Services\macdisk\Security
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 11:57 AM

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 46 00 69 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
F.i.....
00000060 00 00 00 05 20 00 00 00 - 23 02 00 00 6c 00 74 00
... #...l.t.
00000070 00 00 1c 00 ff 01 0f 00 - 01 02 00 00 00 00 05
.....
00000080 20 00 00 00 20 02 00 00 - 6c 00 74 00 00 00 1c 00 ...
...

```

1.t.....
00000090 ff 01 0f 00 01 02 00 00 - 00 00 00 05 20 00 00 00
.....
000000a0 25 02 00 00 6c 00 74 00 - 00 00 18 00 fd 01 02 00
%...l.t.
.....
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 -
.....

```

Services\MSDTC

```

Key Name: SYSTEM\CurrentControlSet\Services\MSDTC\Security
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 1:00 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 00 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
.....
00000060 00 00 00 05 20 00 00 00 - 23 02 00 00 00 00 00 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 - 00 00 00 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 00 00 00 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 -
.....

```

Services\MSSQLServer

```

Key Name: SYSTEM\CurrentControlSet\Services\MSSQLServer
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 1:01 PM

```

```

Value 0
Name: DisplayName
Type: REG_SZ
Data: MSSQLServer

```

```

Value 1
Name: ErrorControl
Type: REG_DWORD
Data: 0x1

```

```

Value 2
Name: ImagePath
Type: REG_EXPAND_SZ
Data: C:\MSSQL7\bin\sqlservr.exe

```

```

Value 3
Name: ObjectName
Type: REG_SZ
Data: LocalSystem

```

```

Value 4
Name: Start
Type: REG_DWORD
Data: 0x3

```

```

Value 5
Name: Type
Type: REG_DWORD
Data: 0x10

```

```

Key Name: SYSTEM\CurrentControlSet\Services\MSSQLServer\Enum
Class Name: <NO CLASS>
Last Write Time: 6/17/98 - 6:46 PM
Value 0
Name: 0
Type: REG_SZ
Data: Root\LEGACY_MSSQLSERVER\0000

```

```

Value 1
Name: Count

```



```

Type:          REG_DWORD
Data:          0x1

Value 2
Name:          NextInstance
Type:          REG_DWORD
Data:          0x1

Key Name:
SYSTEM\CurrentControlSet\Services\MSSQLServer\Perfo
rmance
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 1:01 PM
Value 0
Name:          Close
Type:          REG_SZ
Data:          CloseSQLPerformanceData

Value 1
Name:          Collect
Type:          REG_SZ
Data:          CollectSQLPerformanceData

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:          0x80a

Value 5
Name:          Last Help
Type:          REG_DWORD
Data:          0x80b

Value 6
Name:          Library
Type:          REG_SZ
Data:          SQLCTR70.DLL

Value 7
Name:          Open
Type:          REG_SZ
Data:          OpenSQLPerformanceData

Key Name:
SYSTEM\CurrentControlSet\Services\MSSQLServer\Secur
ity
Class Name:    <NO CLASS>

```

```

Last Write Time: 6/10/98 - 1:00 PM
Value
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 4e 00 54 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
N.T.....
00000060 00 00 00 05 20 00 00 00 - 23 02 00 00 5c 00 73 00
...
#...\s.
00000070 00 00 1c 00 ff 01 0f 00 - 01 02 00 00 00 00 05
.....
00000080 20 00 00 00 20 02 00 00 - 5c 00 73 00 00 00 1c 00
...
\s.....
00000090 ff 01 0f 00 01 02 00 00 - 00 00 00 05 20 00 00 00
.....
000000a0 25 02 00 00 5c 00 73 00 - 00 00 18 00 fd 01 02 00
%\s.
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 -
.....

Services\NDIS
Key Name:      SYSTEM\CurrentControlSet\Services\NDIS
Class Name:    <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name:          DisplayName
Type:          REG_SZ
Data:          Microsoft NDIS System Driver
Value 1

```

```

Name: ErrorControl
Type: REG_DWORD
Data: 0x1

Value 2
Name: Group
Type: REG_SZ
Data: NDIS

Value 3
Name: Start
Type: REG_DWORD
Data: 0x1

Value 4
Name: Type
Type: REG_DWORD
Data: 0x1

Key Name: SYSTEM\CurrentControlSet\Services\NDIS\Enum
Class Name: <NO CLASS>
Last Write Time: 6/17/98 - 6:46 PM
Value 0
Name: 0
Type: REG_SZ
Data: Root\LEGACY_NDIS\0000

Value 1
Name: Count
Type: REG_DWORD
Data: 0x1

Value 2
Name: NextInstance
Type: REG_DWORD
Data: 0x1

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\MediaTypes
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM

Key Name: SYSTEM\CurrentControlSet\Services\NDIS\NetDetect
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\3C592
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM

```

```

Value 0
Name: Id
Type: REG_DWORD
Data: 0x20596d50

Value 1
Name: Mask
Type: REG_DWORD
Data: 0xf0ffffff

Value 2
Name: token
Type: REG_SZ
Data: 3C592

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\3C597
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x70596d50

Value 1
Name: Mask
Type: REG_DWORD
Data: 0xf0ffffff

Value 2
Name: token
Type: REG_SZ
Data: 3C597

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\BONSAI
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x62110e

Value 1
Name: Mask
Type: REG_DWORD
Data: 0xffffffff

Value 2
Name: token
Type: REG_SZ
Data: BONSAI

```

Key Name:
 SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
 SA\C320TNT
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 3:56 AM
 Value 0
 Name: Id
 Type: REG_DWORD
 Data: 0x32530e
 Value 1
 Name: Mask
 Type: REG_DWORD
 Data: 0xffffffff
 Value 2
 Name: token
 Type: REG_SZ
 Data: C320TNT

Key Name:
 SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
 SA\DE425
 Class Name: <NO CLASS>
 Last Write Time: 10/10/96 - 1:09 AM
 Value 0
 Name: Id
 Type: REG_DWORD
 Data: 0x5042a310
 Value 1
 Name: Mask
 Type: REG_DWORD
 Data: 0xf0ffffff
 Value 2
 Name: token
 Type: REG_SZ
 Data: DE425

Key Name:
 SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
 SA\DEC300
 Class Name: <NO CLASS>
 Last Write Time: 10/10/96 - 1:09 AM
 Value 0
 Name: Id
 Type: REG_DWORD
 Data: 0x230a310
 Value 1
 Name: Mask
 Type: REG_DWORD
 Data: 0xffffffff
 Value 2
 Name: token

Type: REG_SZ
 Data: DEC300

Key Name:
 SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
 SA\DEC422
 Class Name: <NO CLASS>
 Last Write Time: 10/10/96 - 1:09 AM
 Value 0
 Name: Id
 Type: REG_DWORD
 Data: 0x2042a310
 Value 1
 Name: Mask
 Type: REG_DWORD
 Data: 0xf0ffffff
 Value 2
 Name: token
 Type: REG_SZ
 Data: DEC422

Key Name:
 SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
 SA\DURANGO
 Class Name: <NO CLASS>
 Last Write Time: 10/10/96 - 1:09 AM
 Value 0
 Name: Id
 Type: REG_DWORD
 Data: 0x260110e
 Value 1
 Name: Mask
 Type: REG_DWORD
 Data: 0xffffffff
 Value 2
 Name: token
 Type: REG_SZ
 Data: DURANGO

Key Name:
 SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
 SA\ELNK3EISA
 Class Name: <NO CLASS>
 Last Write Time: 10/10/96 - 1:09 AM
 Value 0
 Name: Id
 Type: REG_DWORD
 Data: 0x90506d50
 Value 1
 Name: Mask
 Type: REG_DWORD

```

Data:          0xf0ffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          ELNK3EISA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\ES3210
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name:          Id
Type:          REG_DWORD
Data:          0x12949

Value 1
Name:          Mask
Type:          REG_DWORD
Data:          0xffffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          ES3210

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\F70XX
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name:          Id
Type:          REG_DWORD
Data:          0x6690e

Value 1
Name:          Mask
Type:          REG_DWORD
Data:          0xffffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          F70XX

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\FL32
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name:          Id
Type:          REG_DWORD
Data:          0x1010d425

```

```

Value 1
Name:          Mask
Type:          REG_DWORD
Data:          0xffffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          FL32

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\FLNK
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name:          Id
Type:          REG_DWORD
Data:          0x776d50

Value 1
Name:          Mask
Type:          REG_DWORD
Data:          0xffffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          FLNK

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\J2577A
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name:          Id
Type:          REG_DWORD
Data:          0x4019f022

Value 1
Name:          Mask
Type:          REG_DWORD
Data:          0xf0ffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          J2577A

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\MAPLE
Class Name:    <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM

```

Value 0
Name: Id
Type: REG_DWORD
Data: 0x160110e

Value 1
Name: Mask
Type: REG_DWORD
Data: 0xffffffff

Value 2
Name: token
Type: REG_SZ
Data: MAPLE

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI

Class Name: SA\NE3200
<NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM

Value 0
Name: Id
Type: REG_DWORD
Data: 0x7cc3a

Value 1
Name: Mask
Type: REG_DWORD
Data: 0xffffffff

Value 2
Name: token
Type: REG_SZ
Data: NE3200

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI

Class Name: SA\NETFLEX3
<NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM

Value 0
Name: Id
Type: REG_DWORD
Data: 0x20f1110e

Value 1
Name: Mask
Type: REG_DWORD
Data: 0xf0ffffff

Value 2
Name: token
Type: REG_SZ
Data: NETFLEX3

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI

Class Name: SA\NETFLEX3.1
<NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM

Value 0
Name: Id
Type: REG_DWORD
Data: 0x40f1110e

Value 1
Name: Mask
Type: REG_DWORD
Data: 0xf0ffffff

Value 2
Name: token
Type: REG_SZ
Data: NETFLEX3

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI

Class Name: SA\NETFLX
<NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM

Value 0
Name: Id
Type: REG_DWORD
Data: 0x61110e

Value 1
Name: Mask
Type: REG_DWORD
Data: 0xffffffff

Value 2
Name: token
Type: REG_SZ
Data: NETFLX

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI

Class Name: SA\NF3500
<NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM

Value 0
Name: Id
Type: REG_DWORD
Data: 0x84633a

Value 1
Name: Mask
Type: REG_DWORD
Data: 0xffffffff

Value 2
Name: token

```

Type:          REG_SZ
Data:          NF3500

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\NPEISA.1
Class Name:    <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name:          id
Type:          REG_DWORD
Data:          0x2093a

Value 1
Name:          Mask
Type:          REG_DWORD
Data:          0xffffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          NPEISA

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\NPEISA.2
Class Name:    <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name:          id
Type:          REG_DWORD
Data:          0x3093a

Value 1
Name:          Mask
Type:          REG_DWORD
Data:          0xffffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          NPEISA

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\P1990
Class Name:    <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name:          Id
Type:          REG_DWORD
Data:          0x604f42

Value 1
Name:          Mask
Type:          REG_DWORD

```

```

Data:          0xffffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          P1990

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\RODAN
Class Name:    <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name:          Id
Type:          REG_DWORD
Data:          0x63110e

Value 1
Name:          Mask
Type:          REG_DWORD
Data:          0xffffffff

Value 2
Name:          token
Type:          REG_SZ
Data:          RODAN

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\SKETHNT
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name:          Id
Type:          REG_DWORD
Data:          0x2644d

Value 1
Name:          Mask
Type:          REG_DWORD
Data:          0xfffff

Value 2
Name:          token
Type:          REG_SZ
Data:          SKETHNT

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
SA\SKFENT
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name:          Id
Type:          REG_DWORD
Data:          0x1644d

```

```

Value 1
  Name:      Mask
  Type:      REG_DWORD
  Data:      0xfffff

Value 2
  Name:      token
  Type:      REG_SZ
  Data:      SKFENT

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
  Class Name:  SA\SMC8232
  Last Write Time: 6/10/98 - 3:56 AM
Value 0
  Name:      Id
  Type:      REG_DWORD
  Data:      0x80a34d

Value 1
  Name:      Mask
  Type:      REG_DWORD
  Data:      0xffffffff

Value 2
  Name:      token
  Type:      REG_SZ
  Data:      SMC8232

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
  Class Name:  SA\TLNK3E
  Last Write Time: 6/10/98 - 3:56 AM
Value 0
  Name:      Id
  Type:      REG_DWORD
  Data:      0x9c616d50

Value 1
  Name:      Mask
  Type:      REG_DWORD
  Data:      0xf0ffffff

Value 2
  Name:      token
  Type:      REG_SZ
  Data:      TLNK3E

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\EI
  Class Name:  SA\TLNK3EISA
  Last Write Time: 6/10/98 - 3:56 AM

Value 0
  Name:      Id
  Type:      REG_DWORD
  Data:      0x90616d50

Value 1
  Name:      Mask
  Type:      REG_DWORD
  Data:      0xf0ffffff

Value 2
  Name:      token
  Type:      REG_SZ
  Data:      TLNK3EISA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
  Class Name:  A
  Last Write Time: 10/10/96 - 1:09 AM

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
  Class Name:  A\AT1700
  Last Write Time: 6/10/98 - 3:56 AM
Value 0
  Name:      Id
  Type:      REG_DWORD
  Data:      0x6413

Value 1
  Name:      token
  Type:      REG_SZ
  Data:      AT1700

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
  Class Name:  A\EE16MC
  Last Write Time: 10/10/96 - 1:09 AM
Value 0
  Name:      Id
  Type:      REG_DWORD
  Data:      0x628b

Value 1
  Name:      token
  Type:      REG_SZ
  Data:      EE16MC

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
  Class Name:  A\E1INK527
  Last Write Time: 6/10/98 - 3:56 AM

```

Value 0
Name: Id
Type: REG_DWORD
Data: 0x41

Value 1
Name: token
Type: REG_SZ
Data: ELNK527

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\ELNK3MCA.1
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x627c

Value 1
Name: token
Type: REG_SZ
Data: ELNK3MCA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\ELNK3MCA.2
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x627d

Value 1
Name: token
Type: REG_SZ
Data: ELNK3MCA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\ELNK3MCA.3
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x61db

Value 1
Name: token
Type: REG_SZ
Data: ELNK3MCA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\ELNK3MCA.4
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM

Value 0
Name: Id
Type: REG_DWORD
Data: 0x62f6

Value 1
Name: token
Type: REG_SZ
Data: ELNK3MCA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\ELNK3MCA.5
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x62f7

Value 1
Name: token
Type: REG_SZ
Data: ELNK3MCA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\ELNKMC
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x6042

Value 1
Name: token
Type: REG_SZ
Data: ELNKMC

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\F30XX
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x70

Value 1


```

Name: token
Type: REG_SZ
Data: F30XX

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\HPMCA
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x63ca

Value 1
Name: token
Type: REG_SZ
Data: HPMCA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\IBMENIIN
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xffe0

Value 1
Name: token
Type: REG_SZ
Data: IBMENIIN

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\IBMTOKA
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xe000

Value 1
Name: token
Type: REG_SZ
Data: IBMTOKA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\IBMTOKMC
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0

```

```

Name: Id
Type: REG_DWORD
Data: 0xe001

Value 1
Name: token
Type: REG_SZ
Data: IBMTOKMC

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\IRMAtrac.1
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x5c1c

Value 1
Name: token
Type: REG_SZ
Data: IRMAtrac

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\IRMAtrac.2
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x5c1d

Value 1
Name: token
Type: REG_SZ
Data: IRMAtrac

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\NCR TOK
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x100

Value 1
Name: token
Type: REG_SZ
Data: NCR TOK

```

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\NPMCA
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x69
Value 1
Name: token
Type: REG_SZ
Data: NPMCA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\OCTK16.1
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xa84
Value 1
Name: token
Type: REG_SZ
Data: OCTK16

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\OCTK16.2
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xa85
Value 1
Name: token
Type: REG_SZ
Data: OCTK16

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\OCTK16.3
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xa86
Value 1

Name: token
Type: REG_SZ
Data: OCTK16

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\QUADENET.1
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x8f6d
Value 1
Name: token
Type: REG_SZ
Data: QUADENET

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\QUADENET.2
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x8f6a
Value 1
Name: token
Type: REG_SZ
Data: QUADENET

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\SKFMNT.1
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x83
Value 1
Name: token
Type: REG_SZ
Data: SKFMNT

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
A\SKFMNT.2
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0

```

Name: Id
Type: REG_DWORD
Data: 0xab

Value 1
Name: token
Type: REG_SZ
Data: SKFMNT

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x8fa0

Value 1
Name: token
Type: REG_SZ
Data: STREAMER

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x8fa2

Value 1
Name: token
Type: REG_SZ
Data: STREAMER

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x8fa8

Value 1
Name: token
Type: REG_SZ
Data: STREAMER

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
Class Name: A\STREAMER.4
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x8faa

Value 1
Name: token
Type: REG_SZ
Data: STREAMER

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
Class Name: A\TC$4046E
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x51

Value 1
Name: token
Type: REG_SZ
Data: TC$4046E

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
Class Name: A\UBPS
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x7012

Value 1
Name: token
Type: REG_SZ
Data: UBPS

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MC
Class Name: A\WAVELAN_MCA
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x6a14

Value 1

```

```

Name: token
Type: REG_SZ
Data: WAVELAN_MCA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\WD8003EA
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x67c0

Value 1
Name: token
Type: REG_SZ
Data: WD8003EA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\WD8003WA
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x67c2

Value 1
Name: token
Type: REG_SZ
Data: WD8003WA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\WD8013EPA
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x61c8

Value 1
Name: token
Type: REG_SZ
Data: WD8013EPA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\MCA\WD8013WPA
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0

```

```

Name: Id
Type: REG_DWORD
Data: 0x61c9

Value 1
Name: token
Type: REG_SZ
Data: WD8013WPA

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\3C590
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x590010b7

Value 1
Name: token
Type: REG_SZ
Data: 3C590

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\3C595
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x595010b7

Value 1
Name: token
Type: REG_SZ
Data: 3C595

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\3C905
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x905010b7

Value 1

```

```

Name: token
Type: REG_SZ
Data: 3C905

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
I\ALANE0
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x59009004

Value 1
Name: token
Type: REG_SZ
Data: ALANE0

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
I\AMDPCI
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x20001022

Value 1
Name: token
Type: REG_SZ
Data: AMDPCI

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
I\DC21040
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x21011

Value 1
Name: token
Type: REG_SZ
Data: DC21040

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
I\DC21041
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0

```

```

Name: Id
Type: REG_DWORD
Data: 0x141011

Value 1
Name: token
Type: REG_SZ
Data: DC21041

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
I\DC21140
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x91011

Value 1
Name: token
Type: REG_SZ
Data: DC21140

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
I\DC21142
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x191011

Value 1
Name: token
Type: REG_SZ
Data: DC21142

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
I\DEFFA
Class Name: <NO CLASS>
Last Write Time: 10/10/96 - 1:09 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xf1011

Value 1
Name: token
Type: REG_SZ
Data: DEFFA

```

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\E100BPCI
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x12298086

Value 1
Name: token
Type: REG_SZ
Data: E100BPCI

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\E10PCI
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x12268086

Value 1
Name: token
Type: REG_SZ
Data: E10PCI

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\LEC
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x100110b6

Value 1
Name: token
Type: REG_SZ
Data: LEC

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NCPF
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x111bc

Value 1
Name: token
Type: REG_SZ
Data: NCPF

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.1
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xf1300e11

Value 1
Name: token
Type: REG_SZ
Data: NETFLEX3

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.2
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xae320e11

Value 1
Name: token
Type: REG_SZ
Data: NETFLEX3

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.3
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xae340e11

Value 1
Name: token
Type: REG_SZ
Data: NETFLEX3

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.4
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM

Value 0
Name: Id
Type: REG_DWORD
Data: 0xae350e11

Value 1
Name: token
Type: REG_SZ
Data: NETFLEX3

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.5
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xae430e11

Value 1
Name: token
Type: REG_SZ
Data: NETFLEX3

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.6
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xae400e11

Value 1
Name: token
Type: REG_SZ
Data: NETFLEX3

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\NETFLEX3.7
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0xf1500e11

Value 1
Name: token
Type: REG_SZ
Data: NETFLEX3

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\O100PCI
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM

Value 0
Name: Id
Type: REG_DWORD
Data: 0x11108d

Value 1
Name: token
Type: REG_SZ
Data: O100PCI

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\OCE4XMP
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x13108d

Value 1
Name: token
Type: REG_SZ
Data: OCE4XMP

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\OCTK16
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x11108d

Value 1
Name: token
Type: REG_SZ
Data: OCTK16

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PCI\RNSFDDI
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x22001112

Value 1

```

Name: token
Type: REG_SZ
Data: RNSFDDI

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
Class Name: I\RTL8029
Last Write Time: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x802910ec

Value 1
Name: token
Type: REG_SZ
Data: RTL8029

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
Class Name: I\SKFPNT
Last Write Time: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x40001148

Value 1
Name: token
Type: REG_SZ
Data: SKFPNT

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
Class Name: I\SKTOKNT_PCI
Last Write Time: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0
Name: Id
Type: REG_DWORD
Data: 0x42001148

Value 1
Name: token
Type: REG_SZ
Data: SKTOKNT_PCI

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\NetDetect\PC
Class Name: I\STREAMER
Last Write Time: <NO CLASS>
Last Write Time: 6/10/98 - 3:56 AM
Value 0

```

```

Name: Id
Type: REG_DWORD
Data: 0x181014

Value 1
Name: token
Type: REG_SZ
Data: STREAMER

Key Name:
SYSTEM\CurrentControlSet\Services\NDIS\Parameters
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 2:17 PM
Value 0
Name: ProcessorAffinityMask
Type: REG_DWORD
Data: 0

Services\NetBIOS

Key Name: SYSTEM\CurrentControlSet\Services\NetBIOS
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: DependOnGroup
Type: REG_MULTI_SZ
Data: TDI

Value 1
Name: DependOnService
Type: REG_MULTI_SZ
Data:

Value 2
Name: DisplayName
Type: REG_SZ
Data: NetBIOS Interface

Value 3
Name: ErrorControl
Type: REG_DWORD
Data: 0x1

Value 4
Name: Group
Type: REG_SZ
Data: NetBIOSGroup

Value 5
Name: ImagePath
Type: REG_EXPAND_SZ
Data: \SystemRoot\System32\drivers\netbios.sys

Value 6
Name: Start

```



```

Type: REG_DWORD
Data: 0x3

Value 7
Name: Type
Type: REG_DWORD
Data: 0x2

Key Name: SYSTEM\CurrentControlSet\Services\NetBIOS\Enum
Class Name: <NO CLASS>
Last Write Time: 6/17/98 - 6:46 PM
Value 0
Name: 0
Type: REG_SZ
Data: Root\LEGACY_NETBIOS\0000

Value 1
Name: Count
Type: REG_DWORD
Data: 0x1

Value 2
Name: NextInstance
Type: REG_DWORD
Data: 0x1

Key Name:
SYSTEM\CurrentControlSet\Services\NetBIOS\Linkage
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: Bind
Type: REG_MULTI_SZ
Data: \Device\Nbf_E100B1

Value 1
Name: Export
Type: REG_MULTI_SZ
Data: \Device\Netbios\Nbf_E100B1

Value 2
Name: LanaMap
Type: REG_BINARY
Data:
00000000 01 01

Value 3
Name: Route
Type: REG_MULTI_SZ
Data: "Nbf" "E100B" "E100B1"

Key Name:
SYSTEM\CurrentControlSet\Services\NetBIOS\Linkage\D

```

```

Class Name: isabled
GenericClass
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: Bind
Type: REG_MULTI_SZ
Data: \Device\NetBT_E100B1

Value 1
Name: Export
Type: REG_MULTI_SZ
Data: \Device\Netbios\NetBT_E100B1

Value 2
Name: Route
Type: REG_MULTI_SZ
Data: "NetBT" "E100B" "E100B1"

Key Name:
SYSTEM\CurrentControlSet\Services\NetBIOS\Parameters
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:05 AM

Key Name:
SYSTEM\CurrentControlSet\Services\NetBIOS\Parameters\Winsock
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:05 AM
Value 0
Name: HelperDllName
Type: REG_EXPAND_SZ
Data: %SystemRoot%\system32\wshnetbs.dll

Value 1
Name: Mapping
Type: REG_BINARY
Data:
00000000 02 00 00 00 03 00 00 00 - 11 00 00 00 05 00 00 00
.....
00000010 00 00 00 00 11 00 00 00 - 02 00 00 00 00 00 00 00
.....

Value 2
Name: MaxSockAddrLength
Type: REG_DWORD
Data: 0x14

Value 3
Name: MinSockAddrLength
Type: REG_DWORD
Data: 0x14

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NetBIOS\Security
Class Name:      <NO CLASS>
Last Write Time: 6/10/98 - 4:05 AM
Value 0
  Name:          Security
  Type:          REG_BINARY
  Data:
00000000  01 00 14 80 cc 00 00 00 - d8 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 98 00 - 06 00 00 00 00 03 18 00
.....
00000040  00 00 02 00 01 01 00 00 - 00 00 00 01 00 00 00 00
.....
00000050  00 00 00 00 00 03 18 00 - ff 01 0f 00 01 02 00 00
.....
00000060  00 00 00 05 20 00 00 00 - 20 02 00 00 00 03 18 00 ....
...
00000070  ff 01 0f 00 01 01 00 00 - 00 00 00 05 12 00 00 00
.....
00000080  20 02 00 00 00 03 18 00 - 00 00 02 00 01 02 00 00
.....
00000090  00 00 00 05 20 00 00 00 - 23 02 00 00 00 03 18 00 ....
...
#.....
000000a0  9d 00 00 00 01 01 00 00 - 00 00 00 05 04 00 00 00
.....
000000b0  23 02 00 00 00 03 18 00 - 9d 00 00 00 01 02 00 00
#.....
000000c0  00 00 00 05 20 00 00 00 - 21 02 00 00 01 01 00 00 ....
...
!.....
000000d0  00 00 00 05 12 00 00 00 - 01 01 00 00 00 00 00 05
.....
000000e0  12 00 00 00
.....

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NetBIOSInformatio
n
Class Name:      GenericClass
Last Write Time: 6/10/98 - 4:05 AM

```

```

Value 0
  Name:          ErrorControl
  Type:          REG_DWORD
  Data:          0x1

```

```

Value 1
  Name:          Start
  Type:          REG_DWORD
  Data:          0x3

```

```

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

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NetBIOSInformatio
n\Linkage
Class Name:      GenericClass
Last Write Time: 6/10/98 - 4:05 AM

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NetBIOSInformatio
n\Linkage\Disabled
Class Name:      GenericClass
Last Write Time: 6/10/98 - 4:05 AM

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\NetBIOSInformatio
n\Parameters
Class Name:      GenericClass
Last Write Time: 6/10/98 - 4:07 AM

```

```

Value 0
  Name:          EnumExport1
  Type:          REG_DWORD
  Data:          0x1

```

```

Value 1
  Name:          EnumExport2
  Type:          REG_DWORD
  Data:          0x1

```

```

Value 2
  Name:          LanaNum1
  Type:          REG_DWORD
  Data:          0

```

```

Value 3
  Name:          LanaNum2
  Type:          REG_DWORD
  Data:          0x1

```

```

Value 4
  Name:          MaxLana
  Type:          REG_DWORD
  Data:          0x1

```

```
Value 5
```

Name: Route
Type: REG_MULTI_SZ
Data: "NetBT" "E100B" "E100B1"
"Nbf" "E100B" "E100B1"

Services\NetBT

Key Name: SYSTEM\CurrentControlSet\Services\NetBT
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: DependOnGroup
Type: REG_MULTI_SZ
Data:
Value 1
Name: DependOnService
Type: REG_MULTI_SZ
Data: Tcpip
Value 2
Name: DisplayName
Type: REG_SZ
Data: WINS Client (TCP/IP)
Value 3
Name: ErrorControl
Type: REG_DWORD
Data: 0x1
Value 4
Name: Group
Type: REG_SZ
Data: PNP_TDI
Value 5
Name: ImagePath
Type: REG_EXPAND_SZ
Data: \SystemRoot\System32\drivers\netbt.sys
Value 6
Name: Start
Type: REG_DWORD
Data: 0x2
Value 7
Name: Type
Type: REG_DWORD
Data: 0x1
Key Name: SYSTEM\CurrentControlSet\Services\NetBT\Adapters
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:05 AM

Key Name: SYSTEM\CurrentControlSet\Services\NetBT\Adapters\E100B1
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: NameServer
Type: REG_SZ
Data:

Value 1
Name: NameServerBackup
Type: REG_SZ
Data:

Key Name: SYSTEM\CurrentControlSet\Services\NetBT\Enum
Class Name: <NO CLASS>
Last Write Time: 6/17/98 - 6:48 PM
Value 0
Name: Count
Type: REG_DWORD
Data: 0

Value 1
Name: NextInstance
Type: REG_DWORD
Data: 0

Key Name: SYSTEM\CurrentControlSet\Services\NetBT\Linkage
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: Bind
Type: REG_MULTI_SZ
Data:

Value 1
Name: Export
Type: REG_MULTI_SZ
Data:

Value 2
Name: OtherDependencies
Type: REG_MULTI_SZ
Data: Tcpip

Value 3
Name: Route
Type: REG_MULTI_SZ
Data:

Key Name: SYSTEM\CurrentControlSet\Services\NetBT\Linkage\Disabled
Class Name: GenericClass

Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: Bind
Type: REG_MULTI_SZ
Data: \Device\E100B1

Value 1
Name: Export
Type: REG_MULTI_SZ
Data: \Device\NetBT_E100B1

Value 2
Name: Route
Type: REG_MULTI_SZ
Data: "E100B" "E100B1"

Key Name:
SYSTEM\CurrentControlSet\Services\NetBT\Parameters
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:07 AM

Value 0
Name: BcastNameQueryCount
Type: REG_DWORD
Data: 0x3

Value 1
Name: BcastQueryTimeout
Type: REG_DWORD
Data: 0x2ee

Value 2
Name: CacheTimeout
Type: REG_DWORD
Data: 0x927c0

Value 3
Name: EnabledDNS
Type: REG_DWORD
Data: 0

Value 4
Name: EnableLMHOSTS
Type: REG_DWORD
Data: 0x1

Value 5
Name: EnableProxy
Type: REG_DWORD
Data: 0

Value 6
Name: NameServerPort
Type: REG_DWORD
Data: 0x89

Value 7
Name: NameSrvQueryCount
Type: REG_DWORD
Data: 0x3

Value 8
Name: NameSrvQueryTimeout
Type: REG_DWORD
Data: 0x5dc

Value 9
Name: NbProvider
Type: REG_SZ
Data: _tcp

Value 10
Name: ScopeID
Type: REG_SZ
Data:

Value 11
Name: SessionKeepAlive
Type: REG_DWORD
Data: 0x36ee80

Value 12
Name: Size/Small/Medium/Large
Type: REG_DWORD
Data: 0x1

Value 13
Name: TransportBindName
Type: REG_SZ
Data: \Device\

Key Name: SYSTEM\CurrentControlSet\Services\NetBT\Security
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:05 AM

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 01 01 00 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
.....
00000060 00 00 00 05 20 00 00 00 - 23 02 00 00 00 00 05 ....
...
#.....
00000070 00 00 1c 00 ff 01 0f 00 - 01 02 00 00 00 00 05
.....
00000080 20 00 00 00 20 02 00 00 - 00 00 00 05 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 00 00 05 - 00 00 18 00 fd 01 02 00
%.....
000000b0 01 01 00 00 00 00 05 - 12 00 00 00 25 02 00 00
.....
....%...
000000c0 01 01 00 00 00 00 05 - 12 00 00 00 01 01 00 00
.....
000000d0 00 00 00 05 12 00 00 00 -
.....

```

Services\PROSet

```

Key Name: SYSTEM\CurrentControlSet\Services\PROSet
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:01 AM

```

```

Key Name: SYSTEM\CurrentControlSet\Services\PROSet\Adapters
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:01 AM

```

```

Value 0
Name: EPRO100
Type: REG_SZ
Data: Intel EtherExpress PRO Adapter

```

```

Key Name: SYSTEM\CurrentControlSet\Services\PROSet\EPRO100
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:01 AM

```

```

Value 0
Name: AdapterDescription
Type: REG_SZ
Data: EPRO100_GetAdapterDescription

```

```

Value 1
Name: Configure
Type: REG_SZ
Data: EPRO100_Configure

```

```

Value 2

```

```

Name: Detect
Type: REG_SZ
Data: EPRO100_Detect

```

```

Value 3
Name: DeviceExist
Type: REG_SZ
Data: EPRO100_DeviceExist

```

```

Value 4
Name: Diagnose
Type: REG_SZ
Data: EPRO100_Diagnose

```

```

Value 5
Name: DLL
Type: REG_SZ
Data: EPRO100.DLL

```

```

Value 6
Name: GetExtendedFeatures
Type: REG_SZ
Data: EPRO100_GetExtendedFeatures

```

```

Value 7
Name: Help
Type: REG_SZ
Data: E100SET.HLP

```

```

Value 8
Name: InstallAnyway
Type: REG_SZ
Data: EPRO100_InstallAnyway

```

```

Value 9
Name: RegistryKey
Type: REG_SZ
Data: EPRO100_GetRegistryKey

```

```

Value 10
Name: Summary
Type: REG_SZ
Data: EPRO100_Resource_Summary

```

```

Key Name: SYSTEM\CurrentControlSet\Services\PROSet\EPRO100\Parameters
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:01 AM

```

```

Value 0
Name: Adaptive_IFS
Type: REG_SZ
Data: 1,7,Adaptive Inter-Frame Spacing,0,2,1,0,255,1

```

```

Value 1
Name: BusNumber
Type: REG_SZ
Data: 0,7,BusNumber,0,2,0,0,16,1

```

Value 2	Name: BusType Type: REG_SZ Data: 0,7,BusType,0,2,5,2,5,1	Data: 0,3,Transmit Buffer Descriptors,0,2,64,1,65535,1
Value 3	Name: BusTypeLocal Type: REG_SZ Data: 0,7,BusTypeLocal,0,2,5,2,5,1	Value 14 Name: NumTbdPerTcb Type: REG_SZ Data: 0,4,Transmit Buffers per Frame,0,2,12,1,16,1
Value 4	Name: Eid Type: REG_SZ Data: 0,7,Eid,0,2,0,0,4294967295,1	Value 15 Name: NumTcb Type: REG_SZ Data: 1,7,Transmit Control Blocks,0,2,16,1,80,1
Value 5	Name: Fifo Type: REG_SZ Data: 0,3,Fifo Depth,0,2,12,0,15,1	Value 16 Name: Off Type: REG_SZ Data: 1,3,Off Timer,0,2,2,1,65535,1
Value 6	Name: ForceDpx Type: REG_SZ Data: 1,4,Duplex Mode,0,1,Auto,Auto,Half,Full	Value 17 Name: On Type: REG_SZ Data: 1,3,On Timer,0,2,32768,1,65535,1
Value 7	Name: MapRegisters Type: REG_SZ Data:	Value 18 Name: PerfOptims Type: REG_SZ Data: 0,4,PerfOptims,0,2,0,0,65535,1
Value 8	Name: MediaType Type: REG_SZ Data: 0,7,MediaType,0,2,1,1,1,1	Value 19 Name: RxDmaCount Type: REG_SZ Data: 0,4,RxDmaCount,0,2,0,0,63,1
Value 9	Name: MsPciScan Type: REG_SZ Data: 0,4,MsPciScan,0,2,1,0,1,1	Value 20 Name: RxFifo Type: REG_SZ Data: 0,4,Receive Fifo Depth,0,2,8,0,15,1
Value 10	Name: NetworkAddress Type: REG_SZ Data: 1,7,Locally Administered Address,0,5,0,0,1,1	Value 21 Name: Slot Type: REG_SZ Data:
Value 11	Name: NumCoalesce Type: REG_SZ Data: 1,7,Coalesce Buffers,0,2,8,1,32,1	Value 22 Name: Speed Type: REG_SZ Data: 1,7,Network Speed,0,4,Auto,Auto,0,10Mbps,10,100Mbps,100
Value 12	Name: NumRfd Type: REG_SZ Data: 1,7,Receive Buffers,0,2,32,1,1024,1	Value 23 Name: Threshold Type: REG_SZ Data: 0,7,Transmit Threshold,0,2,16,0,200,1
Value 13	Name: NumTbd Type: REG_SZ	Value 24 Name: TxDmaCount Type: REG_SZ Data: 0,4,TxDmaCount,0,2,0,0,63,1

Value 25
 Name: TxFifo
 Type: REG_SZ
 Data: 0,4,Transmit Fifo Depth,0,2,8,0,15,1

Value 26
 Name: Txmitwait
 Type: REG_SZ
 Data: 0,7,Txmitwait,0,2,1,0,255,1

Value 27
 Name: UcodeSW
 Type: REG_SZ
 Data: 0,7,UcodeSW,0,2,1,0,1,1

Value 28
 Name: UnderrunRetry
 Type: REG_SZ
 Data: 0,4,UnderrunRetry,0,2,1,0,3,1

Services\SNMP

Key Name: SYSTEM\CurrentControlSet\Services\SNMP
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 4:05 AM

Value 0
 Name: DependOnGroup
 Type: REG_MULTI_SZ
 Data:

Value 1
 Name: DependOnService
 Type: REG_MULTI_SZ
 Data: Tcpiip
 EventLog

Value 2
 Name: DisplayName
 Type: REG_SZ
 Data: SNMP

Value 3
 Name: ErrorControl
 Type: REG_DWORD
 Data: 0x1

Value 4
 Name: ImagePath
 Type: REG_EXPAND_SZ
 Data: %SystemRoot%\System32\snmp.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: 0x10

Key Name: SYSTEM\CurrentControlSet\Services\SNMP\Enum
 Class Name: <NO CLASS>
 Last Write Time: 6/17/98 - 6:46 PM

Value 0
 Name: 0
 Type: REG_SZ
 Data: Root\LEGACY_SNMP\0000

Value 1
 Name: Count
 Type: REG_DWORD
 Data: 0x1

Value 2
 Name: NextInstance
 Type: REG_DWORD
 Data: 0x1

Key Name: SYSTEM\CurrentControlSet\Services\SNMP\Linkage
 Class Name: GenericClass
 Last Write Time: 6/10/98 - 4:05 AM

Key Name: SYSTEM\CurrentControlSet\Services\SNMP\Linkage\Disab
 led
 Class Name: GenericClass
 Last Write Time: 6/10/98 - 4:05 AM

Key Name: SYSTEM\CurrentControlSet\Services\SNMP\Parameters
 Class Name: GenericClass
 Last Write Time: 6/10/98 - 4:05 AM

Key Name: SYSTEM\CurrentControlSet\Services\SNMP\Parameters\E
 nableAuthenticationTraps
 Class Name: GenericClass
 Last Write Time: 6/10/98 - 4:05 AM

Value 0
 Name: switch
 Type: REG_DWORD
 Data: 0x1

Key Name: SYSTEM\CurrentControlSet\Services\SNMP\Parameters\E
 xtensionAgents

Class Name:	GenericClass	00000000	01 00 14 80 c0 00 00 00 - cc 00 00 00 14 00 00 00
Last Write Time:	6/10/98 - 4:05 AM	
Value 0		
Name:	1	00000010	34 00 00 00 02 00 20 00 - 01 00 00 00 02 80 18 00
Type:	REG_SZ	4.....	
Data:		
		00000020	ff 01 0f 00 01 01 00 00 - 00 00 00 01 00 00 00 00
		
SYSTEM\CurrentControlSet\Services\SNMP\Parameters\PermittedManagers		00000030	20 02 00 00 02 00 8c 00 - 05 00 00 00 00 00 18 00
Class Name:	GenericClass	
Last Write Time:	6/10/98 - 4:05 AM	
Key Name:		00000040	8d 01 02 00 01 01 00 00 - 00 00 00 01 00 00 00 00
SYSTEM\CurrentControlSet\Services\SNMP\Parameters\RFC1156Agent		
Class Name:	GenericClass	00000050	12 00 00 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
Last Write Time:	6/10/98 - 4:06 AM	
Value 0		00000060	00 00 00 05 20 00 00 00 - 23 02 00 00 01 01 00 00
Name:	sysContact	...	
Type:	REG_SZ	#.....	
Data:	SAM&M	00000070	00 00 1c 00 ff 01 0f 00 - 01 02 00 00 00 00 00 05
Value 1		
Name:	sysLocation	00000080	20 00 00 00 20 02 00 00 - 01 01 00 00 00 00 1c 00 ...
Type:	REG_SZ	...	
Data:	MV Performance Lab	
Value 2		00000090	ff 01 0f 00 01 02 00 00 - 00 00 00 05 20 00 00 00
Name:	sysServices	
Type:	REG_DWORD	000000a0	25 02 00 00 01 01 00 00 - 00 00 18 00 fd 01 02 00
Data:	0x4c	%.....	
Key Name:		000000b0	01 01 00 00 00 00 00 05 - 12 00 00 00 25 02 00 00
SYSTEM\CurrentControlSet\Services\SNMP\Parameters\TrapConfiguration		
Class Name:	GenericClass%...	
Last Write Time:	6/10/98 - 4:05 AM	000000c0	01 01 00 00 00 00 00 05 - 12 00 00 00 01 01 00 00
Key Name:		
SYSTEM\CurrentControlSet\Services\SNMP\Parameters\ValidCommunities		000000d0	00 00 00 05 12 00 00 00 -
Class Name:	GenericClass	
Last Write Time:	6/10/98 - 4:05 AM		
Value 0			
Name:	1		
Type:	REG_SZ		
Data:	public		
Key Name:	SYSTEM\CurrentControlSet\Services\SNMP\Security		
Class Name:	<NO CLASS>		
Last Write Time:	6/10/98 - 4:05 AM		
Value 0			
Name:	Security		
Type:	REG_BINARY		
Data:			

		Services\SQLServerAgent	
Key Name:	SYSTEM\CurrentControlSet\Services\SQLServerAgent	Key Name:	SYSTEM\CurrentControlSet\Services\SQLServerAgent
Class Name:	<NO CLASS>	Class Name:	<NO CLASS>
Last Write Time:	6/10/98 - 1:01 PM	Last Write Time:	6/10/98 - 1:01 PM
Value 0		Value 0	
Name:	DependOnGroup	Name:	DependOnGroup
Type:	REG_MULTI_SZ	Type:	REG_MULTI_SZ
Data:		Data:	
Value 1		Value 1	
Name:	DependOnService	Name:	DependOnService
Type:	REG_MULTI_SZ	Type:	REG_MULTI_SZ
Data:	MSSQLServer	Data:	MSSQLServer

Value 2
 Name: DisplayName
 Type: REG_SZ
 Data: SQLServerAgent

Value 3
 Name: ErrorControl
 Type: REG_DWORD
 Data: 0x1

Value 4
 Name: ImagePath
 Type: REG_EXPAND_SZ
 Data: C:\MSSQL7\bin\sqlagent.exe

Value 5
 Name: ObjectName
 Type: REG_SZ
 Data: LocalSystem

Value 6
 Name: Start
 Type: REG_DWORD
 Data: 0x3

Value 7
 Name: Type
 Type: REG_DWORD
 Data: 0x10

Key Name:
 SYSTEM\CurrentControlSet\Services\SQLServerAgent\Se
 curity
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 1:01 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 00 00 00 00 1c 00 - fd 01 02 00 01 02 00 00


```

.....
00000060 00 00 00 05 20 00 00 00 - 23 02 00 00 00 89 ba fd ....
... #.....
00000070 00 00 1c 00 ff 01 0f 00 - 01 02 00 00 00 00 05
.....
.....
00000080 20 00 00 00 20 02 00 00 - 00 89 ba fd 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 00 89 ba fd - 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 -
.....

```

Services\Tcpip

Key Name: SYSTEM\CurrentControlSet\Services\Tcpip
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 4:06 AM
 Value 0
 Name: DisplayName
 Type: REG_SZ
 Data: TCP/IP Service
 Value 1
 Name: ErrorControl
 Type: REG_DWORD
 Data: 0x1
 Value 2
 Name: Group
 Type: REG_SZ
 Data: PNP_TDI
 Value 3
 Name: ImagePath
 Type: REG_EXPAND_SZ
 Data: \SystemRoot\System32\drivers\tcpip.sys
 Value 4
 Name: Start
 Type: REG_DWORD
 Data: 0x2
 Value 5

```

Name:                Type
Type:                REG_DWORD
Data:                0x1

Key Name:            SYSTEM\CurrentControlSet\Services\Tcpip\Enum
Class Name:          <NO CLASS>
Last Write Time:    6/17/98 - 6:46 PM
Value 0
Name:                0
Type:                REG_SZ
Data:                Root\LEGACY_TCPIP\0000

Value 1
Name:                Count
Type:                REG_DWORD
Data:                0x1

Value 2
Name:                NextInstance
Type:                REG_DWORD
Data:                0x1

Key Name:            SYSTEM\CurrentControlSet\Services\Tcpip\Linkage
Class Name:          GenericClass
Last Write Time:    6/10/98 - 4:07 AM
Value 0
Name:                Bind
Type:                REG_MULTI_SZ
Data:                \Device\E100B1

Value 1
Name:                Export
Type:                REG_MULTI_SZ
Data:                \Device\Tcpip\E100B1

Value 2
Name:                Route
Type:                REG_MULTI_SZ
Data:                "E100B" "E100B1"

Key Name:            SYSTEM\CurrentControlSet\Services\Tcpip\Linkage\Dis
                    abled
Class Name:          GenericClass
Last Write Time:    6/10/98 - 4:07 AM
Value 0
Name:                Bind
Type:                REG_MULTI_SZ
Data:

Value 1
Name:                Export
Type:                REG_MULTI_SZ
Data:

```

```

Value 2
Name:                Route
Type:                REG_MULTI_SZ
Data:

Key Name:            SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
Class Name:          GenericClass
Last Write Time:    6/10/98 - 11:28 AM
Value 0
Name:                DataBasePath
Type:                REG_EXPAND_SZ
Data:                %SystemRoot%\System32\drivers\etc

Value 1
Name:                Domain
Type:                REG_SZ
Data:                mv.unisys.com

Value 2
Name:                EnableSecurityFilters
Type:                REG_DWORD
Data:                0

Value 3
Name:                ForwardBroadcasts
Type:                REG_DWORD
Data:                0

Value 4
Name:                Hostname
Type:                REG_SZ
Data:                ava1on4

Value 5
Name:                IPEnableRouter
Type:                REG_DWORD
Data:                0

Value 6
Name:                KeepAliveInterval
Type:                REG_DWORD
Data:                0x2710

Value 7
Name:                NameServer
Type:                REG_SZ
Data:

Value 8
Name:                SearchList
Type:                REG_SZ
Data:

Value 9
Name:                TcpAverageRTT
Type:                REG_DWORD

```

```

Data:          0x3e8

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\
  PersistentRoutes
Class Name:    GenericClass
Last Write Time: 6/10/98 - 4:05 AM

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\
  Winsock
Class Name:    GenericClass
Last Write Time: 6/10/98 - 4:05 AM
Value 0
  Name:        HelperDllName
  Type:        REG_EXPAND_SZ
  Data:        %SystemRoot%\System32\wshtcpip.dll

Value 1
  Name:        Mapping
  Type:        REG_BINARY
  Data:
00000000  0b 00 00 00 03 00 00 00 - 02 00 00 00 01 00 00 00
.....
00000010  06 00 00 00 02 00 00 00 - 01 00 00 00 00 00 00 00
.....
00000020  02 00 00 00 00 00 00 00 - 06 00 00 00 00 00 00 00
.....
00000030  00 00 00 00 06 00 00 00 - 00 00 00 00 01 00 00 00
.....
00000040  06 00 00 00 02 00 00 00 - 02 00 00 00 11 00 00 00
.....
00000050  02 00 00 00 02 00 00 00 - 00 00 00 00 02 00 00 00
.....
00000060  00 00 00 00 11 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000070  11 00 00 00 00 00 00 00 - 02 00 00 00 11 00 00 00
.....
00000080  02 00 00 00 03 00 00 00 - 00 00 00 00
.....
....

Value 2
  Name:        MaxSockAddrLength
  Type:        REG_DWORD
  Data:        0x10

Value 3
  Name:        MinSockAddrLength
  Type:        REG_DWORD

```

```

Data:          0x10

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Performance
Class Name:    GenericClass
Last Write Time: 6/10/98 - 4:05 AM
Value 0
  Name:        Close
  Type:        REG_SZ
  Data:        CloseTcpIpPerformanceData

Value 1
  Name:        Collect
  Type:        REG_SZ
  Data:        CollectTcpIpPerformanceData

Value 2
  Name:        Library
  Type:        REG_SZ
  Data:        Perfctrs.dll

Value 3
  Name:        Open
  Type:        REG_SZ
  Data:        OpenTcpIpPerformanceData

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Security
Class Name:    <NO CLASS>
Last Write Time: 6/10/98 - 4:05 AM
Value 0
  Name:        Security
  Type:        REG_BINARY
  Data:
00000000  01 00 14 80 c0 00 00 00 - cc 00 00 00 14 00 00 00
.....
00000010  34 00 00 00 02 00 20 00 - 01 00 00 00 02 80 18 00
4.....
00000020  ff 01 0f 00 01 01 00 00 - 00 00 00 01 00 00 00 00
.....
00000030  20 02 00 00 02 00 8c 00 - 05 00 00 00 00 00 18 00
.....
00000040  8d 01 02 00 01 01 00 00 - 00 00 00 01 00 00 00 00
.....
00000050  6d 00 00 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
m.....
00000060  00 00 00 05 20 00 00 00 - 23 02 00 00 43 00 48 00
...
#...C.H.
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 - 43 00 48 00 00 00 1c 00 ...
...
C.H.....
00000090 ff 01 0f 00 01 02 00 00 - 00 00 00 05 20 00 00 00
.....
000000a0 25 02 00 00 43 00 48 00 - 00 00 18 00 fd 01 02 00
%...C.H.
.....
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\Tcpip\ServicePro
vider
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:05 AM
Value 0
Name: Class
Type: REG_DWORD
Data: 0x8

Value 1
Name: DnsPriority
Type: REG_DWORD
Data: 0x7d0

Value 2
Name: HostsPriority
Type: REG_DWORD
Data: 0x1f4

Value 3
Name: LocalPriority
Type: REG_DWORD
Data: 0x1f3

Value 4
Name: Name
Type: REG_SZ
Data: TCP/IP

Value 5
Name: NetbtPriority
Type: REG_DWORD
Data: 0x7d1

Value 6
Name: ProviderPath
Type: REG_EXPAND_SZ
Data: %SystemRoot%\System32\wsock32.dll

```

Services\WinSock

```

Key Name: SYSTEM\CurrentControlSet\Services\WinSock
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:05 AM
Value 0
Name: ErrorControl
Type: REG_DWORD
Data: 0x1

Value 1
Name: Start
Type: REG_DWORD
Data: 0x3

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

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock\Autodial
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 11:59 AM
Value 0
Name: AutodialDllName32
Type: REG_SZ
Data: wininet.dll

Value 1
Name: AutodialFcnName32
Type: REG_SZ
Data: InternetAutodialCallback

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock\Linkage
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:05 AM

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock\Linkage\D
isabled
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:05 AM

```

```

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock\Parameter
s
Class Name: GenericClass
Last Write Time: 6/10/98 - 4:05 AM
Value 0
Name: Transports
Type: REG_MULTI_SZ
Data: Tcpip

```

```

NetBIOS
Mig Key Name: SYSTEM\CurrentControlSet\Services\WinSock\Setup
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: Known Static Providers
Type: REG_MULTI_SZ
Data: Tcpip
      NwlnkIpx
      NwlnkSpx
      AppleTalk
      IsoTp
Value 1
Name: Provider List
Type: REG_MULTI_SZ
Data: Tcpip
      NetBIOS
Value 2
Name: Setup Version
Type: REG_DWORD
Data: 0x1009
Mig Key Name: SYSTEM\CurrentControlSet\Services\WinSock\Setup
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:06 AM
Mig Key Name: SYSTEM\CurrentControlSet\Services\WinSock\Setup
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: WinSock 1.1 Provider Data
Type: REG_BINARY
Data: 00000000 0e 10 00 00 11 00 00 00 - 14 00 00 00 14 00 00 00
.....
00000010 05 00 00 00 ff ff ff ff - 00 fa 00 00 66 00 00 00
.....
.....f...
00000020 09 12 00 00 11 00 00 00 - 14 00 00 00 14 00 00 00
.....
.....
00000030 02 00 00 00 ff ff ff ff - 00 fa 00 00 40 00 00 00
.....
.....@...

```

```

00000040 5c 00 44 00 65 00 76 00 - 69 00 63 00 65 00 5c 00
\D.e.v.
i.c.e.\.
00000050 4e 00 62 00 66 00 5f 00 - 45 00 31 00 30 00 30 00
N.b.f._.
E.1.0.0.
00000060 42 00 31 00 00 00 5c 00 - 44 00 65 00 76 00 69 00
B.1...\
D.e.v.i.
00000070 63 00 65 00 5c 00 4e 00 - 62 00 66 00 5f 00 45 00
c.e.\.N.
b.f._.E.
00000080 31 00 30 00 30 00 42 00 - 31 00 00 00
1.0.0.B.
1...
Value 1
Name: WinSock 2.0 Provider ID
Type: REG_BINARY
Data: 00000000 30 18 5f 8d 73 c2 cf 11 - 95 c8 00 80 5f 48 a1 92
0._.s...
...._H..
Mig Key Name: SYSTEM\CurrentControlSet\Services\WinSock\Setup
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:06 AM
Value 0
Name: WinSock 2.0 Provider ID
Type: REG_BINARY
Data: 00000000 a0 1a 0f e7 8b ab cf 11 - 8c a3 00 80 5f 48 a1 92
.....
...._H..
Mig Key Name: SYSTEM\CurrentControlSet\Services\WinSock\Setup
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:06 AM
Value 0
Name: AppleTalk
Type: REG_BINARY
Data: 00000000 a0 17 3b 2c df c6 cf 11 - 95 c8 00 80 5f 48 a1 92
...;....
...._H..
Value 1
Name: IsoTp
Type: REG_BINARY
Data: 00000000 b0 cb e4 89 c1 b9 cf 11 - 95 c8 00 80 5f 48 a1 92
.....
...._H..

```

Value 2
 Name: McsXns
 Type: REG_BINARY
 Data:
 00000000 b1 cb e4 89 c1 b9 cf 11 - 95 c8 00 80 5f 48 a1 92

_H..

Services\WinSock2

Key Name: SYSTEM\CurrentControlSet\Services\WinSock2
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 4:01 AM

Key Name: SYSTEM\CurrentControlSet\Services\WinSock2\Paramete
 rs
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 4:06 AM
 Value 0
 Name: Current_NameSpace_Catalog
 Type: REG_SZ
 Data: NameSpace_Catalog5

Value 1
 Name: Current_Protocol_Catalog
 Type: REG_SZ
 Data: Protocol_Catalog9

Value 2
 Name: WinSock_Registry_Version
 Type: REG_SZ
 Data: 2.0

Key Name: SYSTEM\CurrentControlSet\Services\WinSock2\Paramete
 rs\NameSpace_Catalog5
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 4:05 AM
 Value 0
 Name: Next_Provider_ID
 Type: REG_DWORD
 Data: 0x7d0

Value 1
 Name: Num_Catalog_Entries
 Type: REG_DWORD
 Data: 0x1

Key Name: SYSTEM\CurrentControlSet\Services\WinSock2\Paramete
 rs\NameSpace_Catalog5\Catalog_Entries
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 4:05 AM

Key Name: SYSTEM\CurrentControlSet\Services\WinSock2\Paramete

rs\NameSpace_Catalog5\Catalog_Entries\000000000001
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 4:05 AM
 Value 0
 Name: DisplayString
 Type: REG_SZ
 Data: TCP/IP

Value 1
 Name: Enabled
 Type: REG_DWORD
 Data: 0x1

Value 2
 Name: LibraryPath
 Type: REG_SZ
 Data: %SystemRoot%\System32\rnr20.dll

Value 3
 Name: ProviderId
 Type: REG_BINARY
 Data: 00000000 40 9d 05 22 9e 7e cf 11 - ae 5a 00 aa 00 a7 11 2b
 @...".~...
 .Z.....+

Value 4
 Name: StoresServiceClassInfo
 Type: REG_DWORD
 Data: 0x5e7

Value 5
 Name: SupportedNameSpace
 Type: REG_DWORD
 Data: 0xc

Value 6
 Name: Version
 Type: REG_DWORD
 Data: 0

Key Name: SYSTEM\CurrentControlSet\Services\WinSock2\Paramete

rs\Protocol_Catalog9
 Class Name: <NO CLASS>
 Last Write Time: 6/10/98 - 4:07 AM
 Value 0
 Name: Next_Catalog_Entry_ID
 Type: REG_DWORD
 Data: 0x3f2

Value 1
 Name: Next_Provider_ID
 Type: REG_DWORD

```

Data:          0x1
Value 2
Name:         Num_Catalog_Entries
Type:        REG_DWORD
Data:        0x5

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock2\Paramete
rs\Protocol_Catalog9\Catalog_Entries
Class Name:   <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock2\Paramete
rs\Protocol_Catalog9\Catalog_Entries\000000000001
Class Name:   <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name:        PackedCatalogItem
Type:        REG_BINARY
Data:
00000000  25 53 79 73 74 65 6d 52 - 6f 6f 74 25 5c 73 79 73
%SystemR
oot%\sys
00000010  74 65 6d 33 32 5c 6d 73 - 61 66 64 2e 64 6c 6c 00
tem32\ms
afd.dll.
00000020  61 66 64 2e 64 6c 6c 00 - 76 00 65 00 72 00 20 00
afd.dll.
v.e.r. .
00000030  6e 00 6f 00 64 00 65 00 - 73 00 2c 00 20 00 66 00
n.o.d.e.
s,. .f.
00000040  6f 00 72 00 20 00 77 00 - 68 00 69 00 63 00 68 00 o.r.
.w.
h.i.c.h.
00000050  20 00 74 00 68 00 65 00 - 72 00 65 00 20 00 61 00
.t.h.e.
r.e. .a.
00000060  72 00 65 00 20 00 73 00 - 65 00 70 00 61 00 72 00 r.e.
.s.
e.p.a.r.
00000070  61 00 74 00 65 00 20 00 - 69 00 74 00 65 00 6d 00
a.t.e. .
i.t.e.m.
00000080  73 00 20 00 74 00 6f 00 - 20 00 62 00 65 00 0d 00 s.
.t.o.
.b.e...
00000090  0a 00 3b 00 20 00 70 00 - 72 00 65 00 73 00 65 00 ..;.
.p.
r.e.s.e.
000000a0  6e 00 74 00 65 00 64 00 - 20 00 74 00 6f 00 20 00
n.t.e.d.
.t.o. .
000000b0  74 00 68 00 65 00 20 00 - 75 00 73 00 65 00 72 00
t.h.e. .

```

```

u.s.e.r.
000000c0  2e 00 20 00 20 00 54 00 - 68 00 65 00 73 00 65 00 . . .
.T.
h.e.s.e.
000000d0  20 00 63 00 6f 00 6d 00 - 62 00 69 00 6e 00 61 00
.c.o.m.
b.i.n.a.
000000e0  74 00 69 00 6f 00 6e 00 - 20 00 6e 00 6f 00 64 00
t.i.o.n.
.n.o.d.
000000f0  65 00 73 00 20 00 61 00 - 72 00 65 00 20 00 6f 00 e.s.
.a.
r.e. .o.
00000100  6e 00 6c 00 66 00 02 00 - 00 00 00 00 00 00 00 00
n.l.f...
.....
00000110  00 00 00 00 08 00 00 00 - a0 1a 0f e7 8b ab cf 11
.....
.....
00000120  8c a3 00 80 5f 48 a1 92 - e9 03 00 00 01 00 00 00
...._H..
.....
00000130  00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000140  00 00 00 00 00 00 00 00 - 00 00 00 00 02 00 00 00
.....
.....
00000150  02 00 00 00 10 00 00 00 - 10 00 00 00 01 00 00 00
.....
.....
00000160  06 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000170  00 00 00 00 00 00 00 00 - 4d 00 53 00 41 00 46 00
.....
.....
M.S.A.F.
00000180  44 00 20 00 54 00 63 00 - 70 00 69 00 70 00 20 00 D.
.T.c.
p.i.p. .
00000190  5b 00 54 00 43 00 50 00 - 2f 00 49 00 50 00 5d 00
[.T.C.P.
/.I.P.].
000001a0  00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000001b0  00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000001c0  00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000001d0  00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000001e0  00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
.....

```

```

000001f0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000200 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000210 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000220 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000230 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000240 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000250 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000260 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000270 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000280 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000290 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002a0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002b0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002c0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002d0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002e0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002f0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000300 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000310 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000320 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....

```

```

.....
00000330 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000340 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000350 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000360 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000370 00 00 00 00 00 00 00 00 -
.....

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock2\Paramete
rs\Protocol_Catalog9\Catalog_Entries\000000000002
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: PackedCatalogItem
Type: REG_BINARY
Data:
00000000 25 53 79 73 74 65 6d 52 - 6f 6f 74 25 5c 73 79 73
%SystemR
oot%\sys
00000010 74 65 6d 33 32 5c 6d 73 - 61 66 64 2e 64 6c 6c 00
tem32\ms
afd.dll.
00000020 61 66 64 2e 64 6c 6c 00 - 76 00 65 00 72 00 20 00
afd.dll.
v.e.r. .
00000030 6e 00 6f 00 64 00 65 00 - 73 00 2c 00 20 00 66 00
n.o.d.e.
s...f.
00000040 6f 00 72 00 20 00 77 00 - 68 00 69 00 63 00 68 00 o.r.
.w.
h.i.c.h.
00000050 20 00 74 00 68 00 65 00 - 72 00 65 00 20 00 61 00
.t.h.e.
r.e. .a.
00000060 72 00 65 00 20 00 73 00 - 65 00 70 00 61 00 72 00 r.e.
.s.
e.p.a.r.
00000070 61 00 74 00 65 00 20 00 - 69 00 74 00 65 00 6d 00
a.t.e. .
i.t.e.m.
00000080 73 00 20 00 74 00 6f 00 - 20 00 62 00 65 00 0d 00 s.
.t.o.
.b.e...
00000090 0a 00 3b 00 20 00 70 00 - 72 00 65 00 73 00 65 00 ..;.
.p.
r.e.s.e.
000000a0 6e 00 74 00 65 00 64 00 - 20 00 74 00 6f 00 20 00
n.t.e.d.

```



```

.....
00000320 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000330 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000340 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000350 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000360 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000370 00 00 00 00 00 00 00 00 -
.....

```

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock2\Paramete

rs\Protocol_Catalog9\Catalog_Entries\000000000003

```

Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: PackedCatalogItem
Type: REG_BINARY
Data:
00000000 25 53 79 73 74 65 6d 52 - 6f 6f 74 25 5c 73 79 73
%SystemR
oot%\sys
00000010 74 65 6d 33 32 5c 6d 73 - 61 66 64 2e 64 6c 6c 00
tem32\ms
afd.dll.
00000020 61 66 64 2e 64 6c 6c 00 - 76 00 65 00 72 00 20 00
afd.dll.
v.e.r. .
00000030 6e 00 6f 00 64 00 65 00 - 73 00 2c 00 20 00 66 00
n.o.d.e.
s.,. .f.
00000040 6f 00 72 00 20 00 77 00 - 68 00 69 00 63 00 68 00 o.r.
.w.
h.i.c.h.
00000050 20 00 74 00 68 00 65 00 - 72 00 65 00 20 00 61 00
.t.h.e.
r.e. .a.
00000060 72 00 65 00 20 00 73 00 - 65 00 70 00 61 00 72 00 r.e.
.s.
e.p.a.r.
00000070 61 00 74 00 65 00 20 00 - 69 00 74 00 65 00 6d 00
a.t.e. .
i.t.e.m.
00000080 73 00 20 00 74 00 6f 00 - 20 00 62 00 65 00 0d 00 s.
.t.o.
.b.e...
00000090 0a 00 3b 00 20 00 70 00 - 72 00 65 00 73 00 65 00 ..;.
.p.

```

```

r.e.s.e.
000000a0 6e 00 74 00 65 00 64 00 - 20 00 74 00 6f 00 20 00
n.t.e.d.
.t.o. .
000000b0 74 00 68 00 65 00 20 00 - 75 00 73 00 65 00 72 00
t.h.e. .
u.s.e.r.
000000c0 2e 00 20 00 20 00 54 00 - 68 00 65 00 73 00 65 00 . . .
.T.
h.e.s.e.
000000d0 20 00 63 00 6f 00 6d 00 - 62 00 69 00 6e 00 61 00
.c.o.m.
b.i.n.a.
000000e0 74 00 69 00 6f 00 6e 00 - 20 00 6e 00 6f 00 64 00
t.i.o.n.
.n.o.d.
000000f0 65 00 73 00 20 00 61 00 - 72 00 65 00 20 00 6f 00 e.s.
.a.
r.e. .o.
00000100 6e 00 6c 00 09 06 02 00 - 00 00 00 00 00 00 00 00
n.l.....
.....
00000110 00 00 00 00 0c 00 00 00 - a0 1a 0f e7 8b ab cf 11
.....
.....
00000120 8c a3 00 80 5f 48 a1 92 - eb 03 00 00 01 00 00 00
...._H..
.....
00000130 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000140 00 00 00 00 00 00 00 00 - 00 00 00 00 02 00 00 00
.....
.....
00000150 02 00 00 00 10 00 00 00 - 10 00 00 00 03 00 00 00
.....
.....
00000160 00 00 00 00 ff 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000170 bb ff 00 00 00 00 00 00 - 4d 00 53 00 41 00 46 00
.....
M.S.A.F.
00000180 44 00 20 00 54 00 63 00 - 70 00 69 00 70 00 20 00 D.
.T.c.
p.i.p. .
00000190 5b 00 52 00 41 00 57 00 - 2f 00 49 00 50 00 5d 00
[.R.A.W.
/.I.P.].
000001a0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000001b0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000001c0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
.....

```

```

000001d0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000001e0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000001f0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000200 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000210 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000220 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000230 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000240 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000250 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000260 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000270 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000280 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000290 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002a0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002b0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002c0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002d0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002e0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
000002f0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000300 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....

```

```

.....
00000310 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000320 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000330 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000340 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000350 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000360 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000370 00 00 00 00 00 00 00 00 -
.....

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock2\Paramete
rs\Protocol_Catalog9\Catalog_Entries\000000000004
Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: PackedCatalogItem
Type: REG_BINARY
Data:
00000000 25 53 79 73 74 65 6d 52 - 6f 6f 74 25 5c 73 79 73
%SystemR
oot%\sys
00000010 74 65 6d 33 32 5c 6d 73 - 61 66 64 2e 64 6c 6c 00
tem32\ms
afd.dll.
00000020 61 66 64 2e 64 6c 6c 00 - 76 00 65 00 72 00 20 00
afd.dll.
v.e.r. .
00000030 6e 00 6f 00 64 00 65 00 - 73 00 2c 00 20 00 66 00
n.o.d.e.
s.,. .f.
00000040 6f 00 72 00 20 00 77 00 - 68 00 69 00 63 00 68 00 o.r.
.w.
h.i.c.h.
00000050 20 00 74 00 68 00 65 00 - 72 00 65 00 20 00 61 00
.t.h.e.
r.e. .a.
00000060 72 00 65 00 20 00 73 00 - 65 00 70 00 61 00 72 00 r.e.
.s.
e.p.a.r.
00000070 61 00 74 00 65 00 20 00 - 69 00 74 00 65 00 6d 00
a.t.e. .
i.t.e.m.
00000080 73 00 20 00 74 00 6f 00 - 20 00 62 00 65 00 0d 00 s.
.t.o.

```

.b.e...	00000090	0a 00 3b 00 20 00 70 00 - 72 00 65 00 73 00 65 00	..;	E.Q.P.A.	000001c0	45 00 51 00 50 00 41 00 - 43 00 4b 00 45 00 54 00
.p.	r.e.s.e.	000000a0	6e 00 74 00 65 00 64 00 - 20 00 74 00 6f 00 20 00	C.K.E.T.	000001d0	20 00 31 00 00 00 00 00 - 00 00 00 00 00 00 00
n.t.e.d.	.t.o.	000000b0	74 00 68 00 65 00 20 00 - 75 00 73 00 65 00 72 00	.1.....	000001e0	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
t.h.e.	u.s.e.r.	000000c0	2e 00 20 00 20 00 54 00 - 68 00 65 00 73 00 65 00	000001f0	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.T.	h.e.s.e.	000000d0	20 00 63 00 6f 00 6d 00 - 62 00 69 00 6e 00 61 00	00000200	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.c.o.m.	b.i.n.a.	000000e0	74 00 69 00 6f 00 6e 00 - 20 00 6e 00 6f 00 64 00	00000210	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
t.i.o.n.	.n.o.d.	000000f0	65 00 73 00 20 00 61 00 - 72 00 65 00 20 00 6f 00	e.s.	00000220	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.a.	r.e.o.	00000100	6e 00 6c 00 0e 00 02 00 - 00 00 00 00 00 00 00	00000230	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
n.l.....	00000110	00 00 00 00 00 00 00 00 - 30 18 5f 8d 73 c2 cf 11	00000240	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.....	0.s...	00000120	95 c8 00 80 5f 48 a1 92 - f0 03 00 00 01 00 00 00	00000250	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
...._H..	00000130	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00	00000260	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.....	00000140	00 00 00 00 00 00 00 00 - 00 00 00 00 02 00 00 00	00000270	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.....	00000150	11 00 00 00 14 00 00 00 - 14 00 00 00 05 00 00 00	00000280	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.....	00000160	ff ff ff ff 00 00 00 00 - 00 00 00 00 00 00 00 00	00000290	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.....	00000170	00 fa 00 00 00 00 00 00 - 4d 00 53 00 41 00 46 00	000002a0	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.....	M.S.A.F.	00000180	44 00 20 00 4e 00 65 00 - 74 00 42 00 49 00 4f 00	D.	000002b0	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.N.e.	t.B.I.O.	00000190	53 00 20 00 5b 00 5c 00 - 44 00 65 00 76 00 69 00	S.	000002c0	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
.[\.	D.e.v.i.	000001a0	63 00 65 00 5c 00 4e 00 - 62 00 66 00 5f 00 45 00	000002d0	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
c.e.\.N.	b.f._.E.	000001b0	31 00 30 00 30 00 42 00 - 31 00 5d 00 20 00 53 00	000002e0	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
1.0.0.B.	1.] .S.			000002f0	00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00
					

```

.....
00000300 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000310 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000320 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000330 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000340 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000350 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000360 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000370 00 00 00 00 00 00 00 00 -
.....

```

Key Name:
SYSTEM\CurrentControlSet\Services\WinSock2\Paramete

rs\Protocol_Catalog9\Catalog_Entries\000000000005

```

Class Name: <NO CLASS>
Last Write Time: 6/10/98 - 4:07 AM
Value 0
Name: PackedCatalogItem
Type: REG_BINARY
Data:
00000000 25 53 79 73 74 65 6d 52 - 6f 6f 74 25 5c 73 79 73
%SystemR
oot%\sys
00000010 74 65 6d 33 32 5c 6d 73 - 61 66 64 2e 64 6c 6c 00
tem32\ms
afd.dll.
00000020 61 66 64 2e 64 6c 6c 00 - 76 00 65 00 72 00 20 00
afd.dll.
v.e.r. .
00000030 6e 00 6f 00 64 00 65 00 - 73 00 2c 00 20 00 66 00
n.o.d.e.
s.,. .f.
00000040 6f 00 72 00 20 00 77 00 - 68 00 69 00 63 00 68 00 o.r.
.w.
h.i.c.h.
00000050 20 00 74 00 68 00 65 00 - 72 00 65 00 20 00 61 00
.t.h.e.
r.e. .a.
00000060 72 00 65 00 20 00 73 00 - 65 00 70 00 61 00 72 00 r.e.
.s.
e.p.a.r.
00000070 61 00 74 00 65 00 20 00 - 69 00 74 00 65 00 6d 00
a.t.e. .

```

```

i.t.e.m.
00000080 73 00 20 00 74 00 6f 00 - 20 00 62 00 65 00 0d 00 s.
.t.o.
.b.e...
00000090 0a 00 3b 00 20 00 70 00 - 72 00 65 00 73 00 65 00 .;.
.p.
r.e.s.e.
000000a0 6e 00 74 00 65 00 64 00 - 20 00 74 00 6f 00 20 00
n.t.e.d.
.t.o. .
000000b0 74 00 68 00 65 00 20 00 - 75 00 73 00 65 00 72 00
t.h.e. .
u.s.e.r.
000000c0 2e 00 20 00 20 00 54 00 - 68 00 65 00 73 00 65 00 . . .
.T.
h.e.s.e.
000000d0 20 00 63 00 6f 00 6d 00 - 62 00 69 00 6e 00 61 00
.c.o.m.
b.i.n.a.
000000e0 74 00 69 00 6f 00 6e 00 - 20 00 6e 00 6f 00 64 00
t.i.o.n.
.n.o.d.
000000f0 65 00 73 00 20 00 61 00 - 72 00 65 00 20 00 6f 00 e.s.
.a.
r.e. .o.
00000100 6e 00 6c 00 09 02 02 00 - 00 00 00 00 00 00 00 00
n.l.....
.....
00000110 00 00 00 00 00 00 00 00 - 30 18 5f 8d 73 c2 cf 11
.....
0..s...
00000120 95 c8 00 80 5f 48 a1 92 - f1 03 00 00 01 00 00 00
...._H..
.....
00000130 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000140 00 00 00 00 00 00 00 00 - 00 00 00 00 02 00 00 00
.....
00000150 11 00 00 00 14 00 00 00 - 14 00 00 00 02 00 00 00
.....
00000160 ff ff ff ff 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
00000170 00 fa 00 00 00 00 00 00 - 4d 00 53 00 41 00 46 00
.....
M.S.A.F.
00000180 44 00 20 00 4e 00 65 00 - 74 00 42 00 49 00 4f 00 D.
.N.e.
t.B.I.O.
00000190 53 00 20 00 5b 00 5c 00 - 44 00 65 00 76 00 69 00 S.
.[.\.
D.e.v.i.
000001a0 63 00 65 00 5c 00 4e 00 - 62 00 66 00 5f 00 45 00
c.e.\.N.
b.f._.E.

```

```

000001b0 31 00 30 00 30 00 42 00 - 31 00 5d 00 20 00 44 00
1.O.O.B.
1.J. .D.
000001c0 41 00 54 00 41 00 47 00 - 52 00 41 00 4d 00 20 00
A.T.A.G.
R.A.M. .
000001d0 31 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
1.....
.....
000001e0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000001f0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000200 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000210 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000220 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000230 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000240 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000250 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000260 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000270 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000280 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000290 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000002a0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000002b0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000002c0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000002d0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
000002e0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....

```

```

.....
000002f0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000300 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000310 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000320 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000330 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000340 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000350 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000360 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
.....
.....
00000370 00 00 00 00 00 00 00 00 -
.....

```

NT Client Configuration Information

Microsoft Diagnostics Report For \\CLIENT1

OS Version Report

```

-----
Microsoft (R) Windows NT (TM) Server
Version 4.0 (Build 1381: Service Pack 4) x86 Multiprocessor Free
Registered Owner: Unisys, Unisys
Product Number: 01397-OEM-0018032-66306
-----

```

System Report

```

-----
System: AT/AT COMPATIBLE
Hardware Abstraction Layer: MPS 1.4 - APIC platform
BIOS Date: 07/18/98
BIOS Version: <unavailable>
-----

```

Processor list:

```

0: x86 Family 6 Model 5 Stepping 2 GenuineIntel ~449 Mhz
1: x86 Family 6 Model 5 Stepping 2 GenuineIntel ~449 Mhz

```

Video Display Report

BIOS Date: 05/21/97
BIOS Version: CL-GD5446 PCI VGA BIOS Version 1.33

Adapter:

Setting: 800 x 600 x 256
60 Hz
Type: cirrus compatible display adapter
String: Cirrus Logic Compatible
Memory: 1 MB
Chip Type: Cirrus Logic 5446
DAC Type: Integrated RAMDAC

Driver:

Vendor: Microsoft Corporation
File(s): cirrus.sys, vga.dll, cirrus.dll, vga256.dll, vga64K.dll
Version: 4.00, 4.0.0

Drives Report

C:\ (Local - NTFS) Total: 4,160,803 KB, Free: 3,377,864 KB
Serial Number: 6802 - BFA0
Bytes per cluster: 512
Sectors per cluster: 1
Filename length: 255

Memory Report

Handles: 1,258
Threads: 116
Processes: 15

Physical Memory (K)

Total: 523,696
Available: 478,036
File Cache: 13,212

Kernel Memory (K)

Total: 22,808
Paged: 8,564
Nonpaged: 14,244

Commit Charge (K)

Total: 24,980
Limit: 1,019,912
Peak: 321,392

Pagefile Space (K)

Total: 524,288
Total in use: 5,108
Peak: 119,092

C:\pagefile.sys

Total: 524,288
Total in use: 5,108
Peak: 119,092

Services Report

Alerter	Stopped	(Manual)
C:\WINNT\System32\services.exe		
Service Account Name: LocalSystem		
Error Severity: Normal		
Service Flags: Shared Process		
Service Dependencies: LanmanWorkstation		
Computer Browser	Running	(Automatic)
C:\WINNT\System32\services.exe		
Service Account Name: LocalSystem		
Error Severity: Normal		
Service Flags: Shared Process		
Service Dependencies: LanmanWorkstation LanmanServer LmHosts		
ClipBook Server	Stopped	(Manual)
C:\WINNT\system32\clipsrv.exe		
Service Account Name: LocalSystem		
Error Severity: Normal		
Service Flags: Own Process		
Service Dependencies: NetDDE		
DHCP Client (TDI)	Stopped	(Disabled)
C:\WINNT\System32\services.exe		
Service Account Name: LocalSystem		
Error Severity: Normal		
Service Flags: Shared Process		
Service Dependencies: Tcpiip Afd NetBT		
EventLog (Event log)	Running	(Automatic)
C:\WINNT\system32\services.exe		
Service Account Name: LocalSystem		
Error Severity: Normal		
Service Flags: Shared Process		
Adaptec Failover Backup Monitor	Stopped	(Manual)
C:\WINNT\System32\forbmon.exe		
Service Account Name: LocalSystem		
Error Severity: Normal		
Service Flags: Own Process		
Server	Running	(Automatic)
C:\WINNT\System32\services.exe		
Service Account Name: LocalSystem		
Error Severity: Normal		
Service Flags: Shared Process		
Group Dependencies: TDI		
Workstation (NetworkProvider)	Running	(Automatic)
C:\WINNT\System32\services.exe		
Service Account Name: LocalSystem		
Error Severity: Normal		
Service Flags: Shared Process		

Group Dependencies: TDI			Remote Command Server	Running	(Automatic)
License Logging Service	Stopped	(Manual)	C:\WINNT\System32\rcmdsvc.exe		
C:\WINNT\System32\llssrv.exe			Service Account Name: LocalSystem		
Service Account Name: LocalSystem			Error Severity: Normal		
Error Severity: Normal			Service Flags: Own Process		
Service Flags: Own Process			Service Dependencies:		
TCP/IP NetBIOS Helper	Running	(Automatic)	LanmanServer		
C:\WINNT\System32\services.exe			Directory Replicator	Stopped	(Manual)
Service Account Name: LocalSystem			C:\WINNT\System32\lmrepl.exe		
Error Severity: Normal			Service Account Name: LocalSystem		
Service Flags: Shared Process			Error Severity: Normal		
Group Dependencies:			Service Flags: Own Process		
NetworkProvider			Service Dependencies:		
Messenger	Stopped	(Manual)	LanmanWorkstation		
C:\WINNT\System32\services.exe			LanmanServer		
Service Account Name: LocalSystem			Remote Procedure Call (RPC) Locator	Stopped	(Manual)
Error Severity: Normal			C:\WINNT\System32\LOCATOR.EXE		
Service Flags: Shared Process			Service Account Name: LocalSystem		
Service Dependencies:			Error Severity: Normal		
LanmanWorkstation			Service Flags: Own Process		
NetBios			Service Dependencies:		
Network DDE (NetDDEGroup)	Stopped	(Manual)	LanmanWorkstation		
C:\WINNT\system32\netdde.exe			Rdr		
Service Account Name: LocalSystem			Remote Procedure Call (RPC) Service	Running	(Automatic)
Error Severity: Normal			C:\WINNT\system32\RpcSs.exe		
Service Flags: Shared Process			Service Account Name: LocalSystem		
Service Dependencies:			Error Severity: Normal		
NetDDESDM			Service Flags: Own Process		
Network DDE DSDM	Stopped	(Manual)	Schedule	Stopped	(Manual)
C:\WINNT\system32\netdde.exe			C:\WINNT\System32\AtSvc.Exe		
Service Account Name: LocalSystem			Service Account Name: LocalSystem		
Error Severity: Normal			Error Severity: Normal		
Service Flags: Shared Process			Service Flags: Own Process		
Net Logon (RemoteValidation)	Stopped	(Manual)	Spooler (SpoolerGroup)	Stopped	(Manual)
C:\WINNT\System32\lsass.exe			C:\WINNT\system32\spoolss.exe		
Service Account Name: LocalSystem			Service Account Name: LocalSystem		
Error Severity: Normal			Error Severity: Normal		
Service Flags: Shared Process			Service Flags: Own Process, Interactive		
Service Dependencies:			Telephony Service	Stopped	(Manual)
LanmanWorkstation			C:\WINNT\system32\tapisrv.exe		
LmHosts			Service Account Name: LocalSystem		
NT LM Security Support Provider	Running	(Manual)	Error Severity: Normal		
C:\WINNT\System32\SERVICES.EXE			Service Flags: Own Process		
Service Account Name: LocalSystem			TUXEDO IPC Helper	Stopped	(Automatic)
Error Severity: Normal			C:\TUXEDO\bin\tuxipc.exe		
Service Flags: Shared Process			Service Account Name: LocalSystem		
Plug and Play (PlugPlay)	Running	(Automatic)	Error Severity: Normal		
C:\WINNT\system32\services.exe			Service Flags: Own Process		
Service Account Name: LocalSystem			TListen (Port: 3050)	Stopped	(Manual)
Error Severity: Normal			C:\TUXEDO\bin\slisten.exe		
Service Flags: Shared Process			Service Account Name: LocalSystem		
Protected Storage	Running	(Automatic)	Error Severity: Normal		
c:\winnt\system32\pstores.exe			Service Flags: Own Process		
Service Account Name: LocalSystem			UPS	Stopped	(Manual)
Error Severity: Normal			C:\WINNT\System32\ups.exe		
Service Flags: Own Process, Interactive			Service Account Name: LocalSystem		
Service Dependencies:			Error Severity: Normal		
RpcSs			Service Flags: Own Process		
			World Wide Web Publishing Service	Stopped	(Manual)

C:\WINNT\System32\inetsrv\inetinfo.exe
 Service Account Name: LocalSystem
 Error Severity: Ignore
 Service Flags: Shared Process
 Service Dependencies:
 RPCSS
 NTLMSSP

Drivers Report

```
-----
Abiosdsk (Primary disk)                Stopped (Disabled)
  Error Severity: Ignore
  Service Flags: Kernel Driver, Shared Process
AFD Networking Support Environment (TDI) Running (Automatic)
  C:\WINNT\System32\drivers\afd.sys
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Ahal154x (SCSI miniport)                Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Ahal174x (SCSI miniport)                Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
aic78xx (SCSI miniport)                 Running (Boot)
  C:\WINNT\System32\DRIVERS\aic78xx.sys
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Always (SCSI miniport)                  Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
ami0nt (SCSI miniport)                  Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
amsint (SCSI miniport)                  Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Arrow (SCSI miniport)                   Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
atapi (SCSI miniport)                   Running (Boot)
  C:\WINNT\System32\DRIVERS\atapi.sys
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Atdisk (Primary disk)                   Stopped (Disabled)
  Error Severity: Ignore
  Service Flags: Kernel Driver, Shared Process
ati (Video)                              Stopped (Disabled)
  Error Severity: Ignore
  Service Flags: Kernel Driver, Shared Process
Beep (Base)                              Running (System)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
BusLogic (SCSI miniport)                Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Busmouse (Pointer Port)                 Stopped (Disabled)
  Error Severity: Ignore
```

```

  Service Flags: Kernel Driver, Shared Process
Cdaudio (Filter)                         Stopped (System)
  Error Severity: Ignore
  Service Flags: Kernel Driver, Shared Process
Cdfs (File system)                       Running (Disabled)
  Error Severity: Normal
  Service Flags: File System Driver, Shared Process
  Group Dependencies:
    SCSI CDROM Class
Cdrom (SCSI CDROM Class)                 Running (System)
  Error Severity: Ignore
  Service Flags: Kernel Driver, Shared Process
  Group Dependencies:
    SCSI miniport
Changer (Filter)                         Stopped (System)
  Error Severity: Ignore
  Service Flags: Kernel Driver, Shared Process
cirrus (Video)                           Running (System)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Cpqarray (SCSI miniport)                 Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
cpqfw2e (SCSI miniport)                  Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
dac960nt (SCSI miniport)                 Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
dce376nt (SCSI miniport)                 Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Dellds (SCSI miniport)                   Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Dell_DGX (Video)                         Stopped (Disabled)
  Error Severity: Ignore
  Service Flags: Kernel Driver, Shared Process
Disk (SCSI Class)                        Running (Boot)
  Error Severity: Ignore
  Service Flags: Kernel Driver, Shared Process
  Group Dependencies:
    SCSI miniport
Diskperf (Filter)                        Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
DptScsi (SCSI miniport)                  Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
dte329x (SCSI miniport)                  Stopped (Disabled)
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
Adaptec EMPCI Adapter Driver (NDIS)      Running (Automatic)
  C:\WINNT\System32\drivers\EMPCI.sys
  Error Severity: Normal
  Service Flags: Kernel Driver, Shared Process
et4000 (Video)                            Stopped (Disabled)
  Error Severity: Ignore
  Service Flags: Kernel Driver, Shared Process
```

Fastfat (Boot file system) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process

Fdl6_700 (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Fd7000ex (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Fd8xx (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

flashpnt (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Floppy (Primary disk) Running (System)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

Ftdisk (Filter) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

HP 10/100TX PCI Ethernet Adapter Driver (NDIS) Running (Automatic)
 C:\WINNT\System32\drivers\hptxnt.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

i8042 Keyboard and PS/2 Mouse Port Driver (Keyboard Port) Running (System)
 System32\DRIVERS\i8042prt.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Inport (Pointer Port) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

Jazzg300 (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

Jazzg364 (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

Jzvxl484 (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

Keyboard Class Driver (Keyboard Class) Running (System)
 System32\DRIVERS\kbdclass.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

KSecDD (Base) Running (System)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

mga (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

mga_mil (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

mitsumi (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

mkecr5xx (SCSI miniport) Stopped (Disabled)

Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Modem (Extended base) Stopped (Manual)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

Mouse Class Driver (Pointer Class) Running (System)
 System32\DRIVERS\mouclass.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Msfis (File system) Running (System)
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process

Mup (Network) Running (Manual)
 C:\WINNT\System32\drivers\mup.sys
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process

NetBEUI Protocol (PNP_TDI) Running (Automatic)
 C:\WINNT\System32\drivers\nbf.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Ncr53c9x (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

ncr77c22 (Video) Stopped (Disabled)
 Error Severity: Ignore
 Service Flags: Kernel Driver, Shared Process

Ncrc700 (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Ncrc710 (SCSI miniport) Stopped (Disabled)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Microsoft NDIS System Driver (NDIS) Running (System)
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

NetBIOS Interface (NetBIOSGroup) Stopped (Manual)
 C:\WINNT\System32\drivers\netbios.sys
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process

Group Dependencies:
 TDI

WINS Client(TCP/IP) (PNP_TDI) Stopped (Automatic)
 C:\WINNT\System32\drivers\netbt.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Service Dependencies:
 Tcpip

NetDetect Stopped (Manual)
 C:\WINNT\system32\drivers\netdetect.sys
 Error Severity: Normal
 Service Flags: Kernel Driver, Shared Process

Npfs (File system) Running (System)
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process

Ntfs (File system) Running (Disabled)
 Error Severity: Normal
 Service Flags: File System Driver, Shared Process

Null (Base) Running (System)
 Error Severity: Normal

Service Flags: Kernel Driver, Shared Process
Oliscsi (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Parallel (Extended base) Running (Automatic)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Service Dependencies:
Parport
Group Dependencies:
Parallel arbitrator
Parport (Parallel arbitrator) Running (Automatic)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
ParVdm (Extended base) Running (Automatic)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Service Dependencies:
Parport
Group Dependencies:
Parallel arbitrator
PCIDump (PCI Configuration) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Pcmcia (System Bus Extender) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
PnP ISA Enabler Driver (Base) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
psidisp (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Ql10wnt (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
qv (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Rdr (Network) Running (Manual)
C:\WINNT\System32\drivers\rdr.sys
Error Severity: Normal
Service Flags: File System Driver, Shared Process
s3 (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Scsiprnt (Extended base) Stopped (Automatic)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Group Dependencies:
SCSI miniport
Scsiscan (SCSI Class) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Group Dependencies:
SCSI miniport
Serial (Extended base) Running (Automatic)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process

Sermouse (Pointer Port) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Sfloppy (Primary disk) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Group Dependencies:
SCSI miniport
Simbad (Filter) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
slcd32 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Sparrow (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Spock (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Srv (Network) Running (Manual)
C:\WINNT\System32\drivers\srv.sys
Error Severity: Normal
Service Flags: File System Driver, Shared Process
symc810 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
T128 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
T13B (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
TCP/IP Service (PNP_TDI) Running (Automatic)
C:\WINNT\System32\drivers\tcpip.sys
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
tga (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
tmv1 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Ultra124 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Ultra14f (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
Ultra24f (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
update (Base) Stopped (System)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
v7vram (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
VgaSave (Video Save) Stopped (System)

```

C:\WINNT\System32\drivers\vga.sys
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
VgaStart (Video Init) Stopped (System)
C:\WINNT\System32\drivers\vga.sys
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Wd33c93 (SCSI miniport) Stopped (Disabled)
Error Severity: Normal
Service Flags: Kernel Driver, Shared Process
wd90c24a (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
wdvga (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
weitekp9 (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process
Xga (Video) Stopped (Disabled)
Error Severity: Ignore
Service Flags: Kernel Driver, Shared Process

```

IRQ and Port Report

Devices	Vector	Level	Affinity
MPS 1.4 - APIC platform	8	8	0x00000003
MPS 1.4 - APIC platform	0	0	0x00000003
MPS 1.4 - APIC platform	1	1	0x00000003
MPS 1.4 - APIC platform	2	2	0x00000003
MPS 1.4 - APIC platform	3	3	0x00000003
MPS 1.4 - APIC platform	4	4	0x00000003
MPS 1.4 - APIC platform	5	5	0x00000003
MPS 1.4 - APIC platform	6	6	0x00000003
MPS 1.4 - APIC platform	7	7	0x00000003
MPS 1.4 - APIC platform	8	8	0x00000003
MPS 1.4 - APIC platform	9	9	0x00000003
MPS 1.4 - APIC platform	10	10	0x00000003
MPS 1.4 - APIC platform	11	11	0x00000003
MPS 1.4 - APIC platform	12	12	0x00000003
MPS 1.4 - APIC platform	13	13	0x00000003
MPS 1.4 - APIC platform	14	14	0x00000003
MPS 1.4 - APIC platform	15	15	0x00000003
MPS 1.4 - APIC platform	16	16	0x00000003
MPS 1.4 - APIC platform	17	17	0x00000003
MPS 1.4 - APIC platform	18	18	0x00000003
MPS 1.4 - APIC platform	19	19	0x00000003
MPS 1.4 - APIC platform	20	20	0x00000003
MPS 1.4 - APIC platform	21	21	0x00000003
MPS 1.4 - APIC platform	22	22	0x00000003
MPS 1.4 - APIC platform	23	23	0x00000003
MPS 1.4 - APIC platform	24	24	0x00000003
MPS 1.4 - APIC platform	25	25	0x00000003
MPS 1.4 - APIC platform	26	26	0x00000003
MPS 1.4 - APIC platform	27	27	0x00000003
MPS 1.4 - APIC platform	28	28	0x00000003

MPS 1.4 - APIC platform	29	29	0x00000003
MPS 1.4 - APIC platform	30	30	0x00000003
MPS 1.4 - APIC platform	31	31	0x00000003
MPS 1.4 - APIC platform	32	32	0x00000003
MPS 1.4 - APIC platform	33	33	0x00000003
MPS 1.4 - APIC platform	34	34	0x00000003
MPS 1.4 - APIC platform	35	35	0x00000003
MPS 1.4 - APIC platform	36	36	0x00000003
MPS 1.4 - APIC platform	37	37	0x00000003
MPS 1.4 - APIC platform	38	38	0x00000003
MPS 1.4 - APIC platform	39	39	0x00000003
MPS 1.4 - APIC platform	40	40	0x00000003
MPS 1.4 - APIC platform	41	41	0x00000003
MPS 1.4 - APIC platform	42	42	0x00000003
MPS 1.4 - APIC platform	43	43	0x00000003
MPS 1.4 - APIC platform	44	44	0x00000003
MPS 1.4 - APIC platform	45	45	0x00000003
MPS 1.4 - APIC platform	46	46	0x00000003
MPS 1.4 - APIC platform	47	47	0x00000003
MPS 1.4 - APIC platform	61	61	0x00000003
MPS 1.4 - APIC platform	65	65	0x00000003
MPS 1.4 - APIC platform	80	80	0x00000003
MPS 1.4 - APIC platform	193	193	0x00000003
MPS 1.4 - APIC platform	225	225	0x00000003
MPS 1.4 - APIC platform	253	253	0x00000003
MPS 1.4 - APIC platform	254	254	0x00000003
MPS 1.4 - APIC platform	255	255	0x00000003
i8042prt	1	1	0xffffffff
i8042prt	12	12	0xffffffff
Serial	4	4	0x00000000
Serial	3	3	0x00000000
EMPCI	16	16	0x00000000
EMPCI	16	16	0x00000000
EMPCI	16	16	0x00010000
EMPCI	16	16	0x00000000
Floppy	6	6	0x00000000
HPTX	36	36	0x80e678e8
HPTX	8	8	0x80e5dba8
HPTX	12	12	0x80e40548
HPTX	16	16	0x00000000
HPTX	32	32	0x00000000
aic78xx	40	40	0x00000000
atapi	0	14	0x00000000

Devices	Physical Address	Length	

MPS 1.4 - APIC platform	0x00000000	0x000000010	
MPS 1.4 - APIC platform	0x00000020	0x000000002	
MPS 1.4 - APIC platform	0x00000040	0x000000004	
MPS 1.4 - APIC platform	0x00000048	0x000000004	
MPS 1.4 - APIC platform	0x00000061	0x000000001	
MPS 1.4 - APIC platform	0x00000070	0x000000002	
MPS 1.4 - APIC platform	0x00000080	0x000000010	
MPS 1.4 - APIC platform	0x00000092	0x000000001	
MPS 1.4 - APIC platform	0x000000a0	0x000000002	
MPS 1.4 - APIC platform	0x000000c0	0x000000010	
MPS 1.4 - APIC platform	0x000000f0	0x000000010	
i8042prt	0x00000060	0x000000001	
i8042prt	0x00000064	0x000000001	

```

Parport          0x00000378  0x0000000003
Serial           0x000003f8  0x0000000007
Serial           0x000002f8  0x0000000007
EMPCI            0x0000dc00  0x0000000080
EMPCI            0x0000d880  0x0000000080
EMPCI            0x0000d800  0x0000000080
EMPCI            0x0000d480  0x0000000080
Floppy           0x000003f0  0x0000000006
Floppy           0x000003f7  0x0000000001
HPTX             0x0000fca0  0x000000001c
HPTX             0x0000ece0  0x000000001c
HPTX             0x0000ecc0  0x000000001c
HPTX             0x0000eca0  0x000000001c
HPTX             0x0000fcc0  0x000000001c
aic78xx         0x0000f800  0x0000000100
ataapi           0x000001f0  0x0000000008
ataapi           0x000003f6  0x0000000001
cirrus           0x000003b0  0x000000000c
cirrus           0x000003c0  0x0000000020

```

DMA and Memory Report

```

-----
Devices          Channel   Port
-----
Floppy            2       0
-----
Devices          Physical Address  Length
-----
MPS 1.4 - APIC platform  0xfec00000  0x00000400
MPS 1.4 - APIC platform  0xfef00000  0x00000400
HPTX                0xfedfc000  0x0000001c
HPTX                0xfbffe000  0x0000001c
HPTX                0xfbffd000  0x0000001c
HPTX                0xfbffc000  0x0000001c
HPTX                0xfedfd000  0x0000001c
aic78xx             0xfedff000  0x00001000
cirrus               0x000a0000  0x00020000
cirrus               0xfc000000  0x02000000

```

Environment Report

System Environment Variables

```

APPDIR=c:\tuxedo\runtime
ComSpec=C:\WINNT\system32\cmd.exe
LIBPATH=c:\tuxedo\lib
NUMBER_OF_PROCESSORS=2
OS=Windows NT
Os2LibPath=C:\WINNT\system32\os2\dll;
Path=C:\WINNT\system32;C:\WINNT;C:\TUXEDO\bin;C:\MSSQL7\BINN
PROCESSOR_ARCHITECTURE=x86
PROCESSOR_IDENTIFIER=x86 Family 6 Model 5 Stepping 2, GenuineIntel
PROCESSOR_LEVEL=6
PROCESSOR_REVISION=0502

```

```

TMCONTEXTS=1
TUXCONFIG=c:\tuxedo\runtime\tuxconfig
TUXDIR=c:\tuxedo
windir=C:\WINNT

```

Environment Variables for Current User

```

TEMP=C:\TEMP
TMP=C:\TEMP

```

Network Report

```

-----
Your Access Level: Admin & Local
Workgroup or Domain: WORKGROUP
Network Version: 4.0
LanRoot: WORKGROUP
Logged On Users: 1
Current User (1): Administrator
  Logon Domain: CLIENT1
  Logon Server: CLIENT1

Transport: Nbf_HPTX1, 00-90-27-27-9D-BD, VC's: 1, Wan: Wan
Transport: Nbf_HPTX2, 00-90-27-27-9E-5E, VC's: 1, Wan: Wan
Transport: Nbf_HPTX3, 00-A0-C9-EA-AA-71, VC's: 1, Wan: Wan
Transport: Nbf_HPTX4, 00-A0-C9-EA-A9-30, VC's: 1, Wan: Wan
Transport: Nbf_HPTX5, 00-A0-C9-EA-CD-4D, VC's: 1, Wan: Wan
Transport: Nbf_EMPCI6, 00-00-92-A7-C2-BC, VC's: 1, Wan: Wan
Transport: Nbf_EMPCI7, 00-00-92-A7-C2-BD, VC's: 1, Wan: Wan
Transport: Nbf_EMPCI8, 00-00-92-A7-C2-BE, VC's: 0, Wan: Wan
Transport: Nbf_EMPCI9, 00-00-92-A7-C2-BF, VC's: 0, Wan: Wan

```

```

Character Wait: 3,600
Collection Time: 250
Maximum Collection Count: 16
Keep Connection: 600
Maximum Commands: 5
Session Time Out: 45
Character Buffer Size: 512
Maximum Threads: 17
Lock Quota: 6,144
Lock Increment: 10
Maximum Locks: 500
Pipe Increment: 10
Maximum Pipes: 500
Cache Time Out: 40
Dormant File Limit: 45
Read Ahead Throughput: 4,294,967,295
Mailslot Buffers: 3
Server Announce Buffers: 20
Illegal Datagrams: 5
Datagram Reset Frequency: 60
Log Election Packets: False
Use Opportunistic Locking: True
Use Unlock Behind: True
Use Close Behind: True
Buffer Pipes: True

```

```

Use Lock, Read, Unlock: True
Use NT Caching: True
Use Raw Read: True
Use Raw Write: True
Use Write Raw Data: True
Use Encryption: True
Buffer Deny Write Files: True
Buffer Read Only Files: True
Force Core Creation: True
512 Byte Max Transfer: False
Bytes Received: 884,333
SMB's Received: 9,777
Paged Read Bytes Requested: 0
Non Paged Read Bytes Requested: 0
Cache Read Bytes Requested: 0
Network Read Bytes Requested: 0
Bytes Transmitted: 1,065,121
SMB's Transmitted: 9,777
Paged Read Bytes Requested: 0
Non Paged Read Bytes Requested: 0
Cache Read Bytes Requested: 0
Network Read Bytes Requested: 0
Initially Failed Operations: 0
Failed Completion Operations: 0
Read Operations: 0
Random Read Operations: 0
Read SMB's: 0
Large Read SMB's: 0
Small Read SMB's: 0
Write Operations: 0
Random Write Operations: 0
Write SMB's: 0
Large Write SMB's: 0
Small Write SMB's: 0
Raw Reads Denied: 0
Raw Writes Denied: 0
Network Errors: 0
Sessions: 1,623
Failed Sessions: 0
Reconnects: 0
Core Connects: 0
LM 2.0 Connects: 0
LM 2.x Connects: 0
Windows NT Connects: 1,623
Server Disconnects: 0
Hung Sessions: 0
Use Count: 3,244
Failed Use Count: 0
Current Commands: 0
Server File Opens: 353
Server Device Opens: 0
Server Jobs Queued: 0
Server Session Opens: 2
Server Sessions Timed Out: 6
Server Sessions Errored Out: 6
Server Password Errors: 0
Server Permission Errors: 0
Server System Errors: 0
Server Bytes Sent: 26,603,937

```

```

Server Bytes Received: 6,151,433
Server Average Response Time: 0
Server Request Buffers Needed: 0
Server Big Buffers Needed: 0

```

Internet Information Server Registry Parameters

```

\registry\machine\system\currentcontrolset\services\inetinfo
Parameters
  BandwidthLevel = REG_DWORD 0xffffffff
  ListenBackLog = REG_DWORD 0x00000019
  DisableMemoryCache = REG_DWORD 0x00000001
  MemoryCacheSize = REG_DWORD 0x00000000
  PoolThreadLimit = REG_DWORD 0x000000b4
  ObjectCacheTTL = REG_DWORD 0xffffffff
  Filter
    FilterType = REG_DWORD 0x00000000
    NumGrantSites = REG_DWORD 0x00000000
    NumDenySites = REG_DWORD 0x00000000
  MimeMap
    text/html,htm,,h =
    image/gif,gif,,g =
    image/jpeg,jpg,,: =
    text/plain,txt,,0 =
    text/html,html,,h =
    image/jpeg,jpeg,,: =
    image/jpeg,jpe,,: =
    image/bmp,bmp,,: =
    application/octet-stream,*,,5 =
    application/pdf,pdf,,5 =
    application/octet-stream,bin,,5 =
    application/oda,oda,,5 =
    application/zip,zip,,9 =
    application/rtf,rtf,,5 =
    application/postscript,ps,,5 =
    application/postscript,ai,,5 =
    application/postscript,eps,,5 =
    application/mac-binhex40,hqx,,4 =
    application/msword,doc,,5 =
    application/msword,dot,,5 =
    application/winhelp,hlp,,5 =
    video/mpeg,mpeg,,; =
    video/mpeg,mpg,,; =
    video/mpeg,mpe,,; =
    video/x-msvideo,avi,,< =
    video/quicktime,qt,,; =
    video/quicktime,mov,,; =
    video/x-sgi-movie,movie,,< =
    x-world/x-vrml,wrl,,5 =
    x-world/x-vrml,xaf,,5 =
    x-world/x-vrml,xof,,5 =
    x-world/x-vrml,flr,,5 =
    x-world/x-vrml,wrz,,5 =
    application/x-director,dcr,,5 =
    application/x-director,dir,,5 =
    application/x-director,dxr,,5 =
    image/cis-cod,cod,,5 =

```

```

image/x-cmx,cmx,,5 =
application/envoy,evy,,5 =
application/x-msaccess,mdb,,5 =
application/x-mscardfile,crd,,5 =
application/x-msclip,clp,,5 =
application/octet-stream,exe,,5 =
application/x-msexcel,xla,,5 =
application/x-msexcel,xlc,,5 =
application/x-msexcel,xlm,,5 =
application/x-msexcel,xls,,5 =
application/x-msexcel,xlt,,5 =
application/x-msexcel,xlw,,5 =
application/x-msmediaview,m13,,5 =
application/x-msmediaview,m14,,5 =
application/x-msmoney,mny,,5 =
application/x-mspowerpoint,ppt,,5 =
application/x-msproject,mpp,,5 =
application/x-mspublisher,pub,,5 =
application/x-sterminal,trm,,5 =
application/x-msworks,wks,,5 =
application/x-mswrite,wri,,5 =
application/x-msmetafile,wmf,,5 =
application/x-csh,csh,,5 =
application/x-dvi,dvi,,5 =
application/x-hdf,hdf,,5 =
application/x-latex,latex,,5 =
application/x-netcdf,nc,,5 =
application/x-netcdf,cdf,,5 =
application/x-sh,sh,,5 =
application/x-tcl,tcl,,5 =
application/x-tex,tex,,5 =
application/x-texinfo,texinfo,,5 =
application/x-texinfo,txi,,5 =
application/x-troff,t,,5 =
application/x-troff,tr,,5 =
application/x-troff,roff,,5 =
application/x-troff-man,man,,5 =
application/x-troff-me,me,,5 =
application/x-troff-ms,ms,,5 =
application/x-wais-source,src,,7 =
application/x-bcpio,bcpio,,5 =
application/x-cpio,cpio,,5 =
application/x-gtar,gtar,,9 =
application/x-shar,shar,,5 =
application/x-sv4cpio,sv4cpio,,5 =
application/x-sv4crc,sv4crc,,5 =
application/x-tar,tar,,5 =
application/x-ustar,ustar,,5 =
audio/basic,au,,< =
audio/basic,snd,,< =
audio/x-aiff,aif,,< =
audio/x-aiff,aiff,,< =
audio/x-aiff,aifc,,< =
audio/x-wav,wav,,< =
audio/x-pn-realaudio,ram,,< =
image/ief,ief,,: =
image/tiff,tiff,,: =
image/tiff,tif,,: =
image/x-cmu-raster,ras,,: =

```

```

image/x-portable-anymap,pnm,,: =
image/x-portable-bitmap,pbm,,: =
image/x-portable-graymap,pgm,,: =
image/x-portable-pixmap,ppm,,: =
image/x-rgb,rgb,,: =
image/x-xbitmap,xbm,,: =
image/x-ypixmap,xpm,,: =
image/x-xwinddumpp,xwd,,: =
text/html,stm,,h =
text/plain,bas,,0 =
text/plain,c,,0 =
text/plain,h,,0 =
text/richtext,rtx,,0 =
text/tab-separated-values,tsv,,0 =
text/x-setext,etx,,0 =
application/x-perfmon,pmc,,5 =
application/x-perfmon,pma,,5 =
application/x-perfmon,pmr,,5 =
application/x-perfmon,pml,,5 =
application/x-perfmon,pmw,,5 =
Performance
Library = infoctrs.DLL
Open = OpenINFOPerformanceData
Close = CloseINFOPerformanceData
Collect = CollectINFOPerformanceData
Last Counter = REG_DWORD 0x00000756
Last Help = REG_DWORD 0x00000757
First Counter = REG_DWORD 0x00000738
First Help = REG_DWORD 0x00000739

```

World Wide Web Server Registry Parameters

```

\registry\machine\system\currentcontrolset\services\w3svc [17 1]
Type = REG_DWORD 0x00000020
Start = REG_DWORD 0x00000003
ErrorControl = REG_DWORD 0x00000000
ImagePath = REG_EXPAND_SZ C:\WINNT\System32\inetrv\inetinfo.exe
DisplayName = World Wide Web Publishing Service
DependOnService = REG_MULTI_SZ "RPCSS" \
"NTLMSSP"
DependOnGroup = REG_MULTI_SZ
ObjectName = LocalSystem
Parameters
MajorVersion = REG_DWORD 0x00000002
MinorVersion = REG_DWORD 0x00000000
AdminName = Administrator
AdminEmail = Admin@corp.com
MaxConnections = REG_DWORD 0x00002710
LogType = REG_DWORD 0x00000000
LogFileDirectory = REG_EXPAND_SZ c:\inetpub\wwwroot
LogFileTruncateSize = REG_DWORD 0x0012f368
LogFilePeriod = REG_DWORD 0x00000000
LogFileFormat = REG_DWORD 0x00000000
LogSqlDataSource = HTTPLOG
LogSqlTableName = Internetlog
LogSqlUserName = InternetAdmin
LogSqlPassword = sqllog

```

```

Authorization = REG_DWORD 0x00000005
AnonymousUserName = IUSR_CLIENT4
Default Load File = Default.htm
Dir Browse Control = REG_DWORD 0x4000001e
CheckForWAISDB = REG_DWORD 0x00000000
CacheExtensions = REG_DWORD 0x00000001
GlobalExpire = REG_DWORD 0xffffffff
ServerSideIncludesEnabled = REG_DWORD 0x00000001
ServerSideIncludesExtension = .stm
DebugFlags = REG_DWORD 0x00000008
ScriptTimeout = REG_DWORD 0x00000384
ConnectionTimeOut = REG_DWORD 0x00001c20
InstallPath = C:\WINNT\System32\inetsrv
SecurePort = REG_DWORD 0x000001bb
Filter DLLs = C:\WINNT\System32\inetsrv\sspicfilt.dll
AccessDeniedMessage = Error: Access is Denied.
NTAuthenticationProviders = NTLM
ServerComment =
ADCLaunch
    AdvancedDataFactory
    RDSServer.DataFactory
Script Map
    .idc = C:\WINNT\System32\inetsrv\httpodbc.dll
Virtual Roots
    /, = C:\InetPub\wwwroot,,5
    /Scripts, = C:\InetPub\scripts,,4
    /MSADC, = C:\Program Files\Common Files\System\MSADC,,5
    /iisadmin, = C:\WINNT\System32\inetsrv\iisadmin,,1
Performance
    Library = w3ctrs.DLL
    Open = OpenW3PerformanceData
    Close = CloseW3PerformanceData
    Collect = CollectW3PerformanceData
    Last Counter = REG_DWORD 0x00000790
    Last Help = REG_DWORD 0x00000791
    First Counter = REG_DWORD 0x00000758
    First Help = REG_DWORD 0x00000759
Security [17 1]
    Security = REG_BINARY 0x000000d8 0x80140001 0x000000c0 0x000000cc
0x00000014 0x00000034 0x00200002 0x00000001 0x00188002 0x000f01ff
0x00000101 0x01000000 0x00000000 0x00000220 0x008c0002 0x00000005
0x00180000 0x0002018d 0x00000101 \
    0x01000000 0x00000000 0x00000000 0x001c0000 0x000201fd
0x00000201 0x05000000 0x00000020 0x00000223 0x001400c8 0x001c0000
0x000f01ff 0x00000201 0x05000000 0x00000020 0x00000220 0x001400c8
0x001c0000 0x000f01ff 0x00000201 \
    0x05000000 0x00000020 0x00000225 0x001400c8 0x00180000
0x000201fd 0x00000101 0x05000000 0x00000012 0x00000225 0x00000101
0x05000000 0x00000012 0x00000101 0x05000000 0x00000012
Enum
    0 = Root\LEGACY_W3SVC\0000
    Count = REG_DWORD 0x00000001
    NextInstance = REG_DWORD 0x00000001

\registry\machine\software\unisys
TPCC
MAXTERMS = 10000

```

Tuxedo Configuration

Note: this configuration file is repeated on each of the other 2 clients with the exception of the Hostname, "CLIENT1", which is replaced by "CLIENT2" thru "CLIENT3".

```

*RESOURCES
IPCKEY          133133

MAXACCESSERS   400
MAXSERVERS     210
MAXSERVICES    1100
MODEL          SHM
MASTER         tpcctm
LDBAL          N
SCANUNIT       60
BLOCKTIME      60
BBLQUERY       60

*MACHINES
DEFAULT:

CLIENT1        LMID=tpcctm
                TUXDIR="c:\tuxedo"
                APPDIR="c:\tuxedo\runtime"
                TUXCONFIG="c:\tuxedo\runtime\tuxconfig"
                ULGPFIX="c:\tuxedo\runtime\ulog\ULOG"
                TYPE="WinNT"
                UID=0
                GID=0

*GROUPS
GRALL          LMID=tpcctm      GRPNO=1      OPENINFO=NONE

GRDEL          LMID=tpcctm      GRPNO=3      OPENINFO=NONE

*SERVERS
DEFAULT:
                CLOPT="-A -- -sAVALON4 -dtpcc"

tpccsvr       SRVGRP=GRALL
                SRVID=100
                MIN=78 MAX=200
                RQADDR=allq REPLYQ=Y

tpccdelv      SRVGRP=GRDEL
                SRVID=300
                MIN=9 MAX=20
                CLOPT="-A -- -sAVALON4 -dtpcc"
                RQADDR=delq REPLYQ=Y

*SERVICES

```


Appendix D - RTE Code

Admin Environment

```
if '%1'==' ' goto usage
if '%2'==' ' goto usage
if '%3'==' ' goto usage

:paramok

net time \\%1 /SET /Y

if %ERRORLEVEL% NEQ 0 pause

set WEBADMINCFG=web%2.cfg
set WEBMAXDRIVERS=%3
set WEBDIAGLEVEL=4
set WEBEVENTLOG=0
set WEBEVENTHOST=
set WEBCHECKLEVEL=2

webadmin.exe

goto end

:usage
@ECHO You must supply the following parameters:
@ECHO "webnnn.cmd <clock sync host name> <cfg file suffix> <driver count>"
pause

:end
```

Profiles used for Performance Run

Web1710.cfg

```
//
// Common Driver Configuration
//
INITBASEPORT 4300
INITSYNCMAX 4
INITPAUSE 1
INITRSCALE 400
INITTSCALE 100
INITRWID 1, 1710
INITFIXEDWID 1
INITCCLAST 208
INITCCID 208
INITCITEMID 208
//
```

```
// Configuration Driver 1
//
1 INITIPADDR 192.168.90.31
1 INITIISADDR 192.168.12.1
1 INITIISPORT 80
1 INITBROWSERS 950
1 INITMYWID 1,95
//
// Configuration Driver 2
//
2 INITIPADDR 192.168.90.31
2 INITIISADDR 192.168.22.2
2 INITIISPORT 80
2 INITBROWSERS 950
2 INITMYWID 96,190
//
// Configuration Driver 3
//
3 INITIPADDR 192.168.90.31
3 INITIISADDR 192.168.32.3
3 INITIISPORT 80
3 INITBROWSERS 950
3 INITMYWID 191,285
//
// Configuration Driver 4
//
4 INITIPADDR 192.168.90.32
4 INITIISADDR 192.168.33.3
4 INITIISPORT 80
4 INITBROWSERS 950
4 INITMYWID 286,380
//
// Configuration Driver 5
//
5 INITIPADDR 192.168.90.32
5 INITIISADDR 192.168.23.2
5 INITIISPORT 80
5 INITBROWSERS 950
5 INITMYWID 381,475
//
// Configuration Driver 6
//
6 INITIPADDR 192.168.90.32
6 INITIISADDR 192.168.13.1
6 INITIISPORT 80
6 INITBROWSERS 950
6 INITMYWID 476,570
//
// Configuration Driver 7
//
7 INITIPADDR 192.168.90.33
7 INITIISADDR 192.168.14.1
7 INITIISPORT 80
7 INITBROWSERS 950
```

```

7 INITMYWID 571,665
//
// Configuration Driver 8
//
8 INITIPADDR 192.168.90.33
8 INITIISADDR 192.168.24.2
8 INITIISPORT 80
8 INITBROWSERS 950
8 INITMYWID 666,760
//
// Configuration Driver 9
//
9 INITIPADDR 192.168.90.33
9 INITIISADDR 192.168.34.3
9 INITIISPORT 80
9 INITBROWSERS 950
9 INITMYWID 761,855
//
// Configuration Driver 10
//
10 INITIPADDR 192.168.90.34
10 INITIISADDR 192.168.35.3
10 INITIISPORT 80
10 INITBROWSERS 950
10 INITMYWID 856,950
//
// Configuration Driver 11
//
11 INITIPADDR 192.168.90.34
11 INITIISADDR 192.168.25.2
11 INITIISPORT 80
11 INITBROWSERS 950
11 INITMYWID 951,1045
//
// Configuration Driver 12
//
12 INITIPADDR 192.168.90.34
12 INITIISADDR 192.168.15.1
12 INITIISPORT 80
12 INITBROWSERS 950
12 INITMYWID 1046,1140
//
// Configuration Driver 13
//
13 INITIPADDR 192.168.90.35
13 INITIISADDR 192.168.16.1
13 INITIISPORT 80
13 INITBROWSERS 950
13 INITMYWID 1141,1235
//
// Configuration Driver 14
//
14 INITIPADDR 192.168.90.35
14 INITIISADDR 192.168.26.2
14 INITIISPORT 80
14 INITBROWSERS 950
14 INITMYWID 1236,1330
//
// Configuration Driver 15

```

```

//
15 INITIPADDR 192.168.90.35
15 INITIISADDR 192.168.36.3
15 INITIISPORT 80
15 INITBROWSERS 950
15 INITMYWID 1331,1425
//
// Configuration Driver 16
//
16 INITIPADDR 192.168.90.36
16 INITIISADDR 192.168.37.3
16 INITIISPORT 80
16 INITBROWSERS 950
16 INITMYWID 1426,1520
//
// Configuration Driver 17
//
17 INITIPADDR 192.168.90.36
17 INITIISADDR 192.168.27.2
17 INITIISPORT 80
17 INITBROWSERS 950
17 INITMYWID 1521,1615
//
// Configuration Driver 18
//
18 INITIPADDR 192.168.90.36
18 INITIISADDR 192.168.17.1
18 INITIISPORT 80
18 INITBROWSERS 950
18 INITMYWID 1616,1710
//

```

Driver Environment

```

if '%1'==' ' goto usage

:paramok

set WEBDRIVERNO=%1
set WEBADMBASEPORT=4300
set WEBDIAGLEVEL=2
set WEBEVENTLOG=1
set WEBEVENTHOST=
set WEBLOGLEVEL=1
set WEBSINGLETRAN=0
set WEBTPCCAUDIT=0
set WEBRTFUDGETM=110
set WEBNEWORDERPROB=4484
set WEBPAYMENTPROB=4307
set WEBORDERSTATUSPROB=403
set WEBDELIVERYPROB=403
set WEBSTOCKLEVELPROB=403
set WEBTTNEWORDER=12030
set WEBTTPAYMENT=12030
set WEBTTDELIVERY=5060
set WEBTTORDERSTATUS=10070
set WEBTTSTOCKLEVEL=5060

```

```
webdriver.exe
```

```
goto end
```

```
:usage
```

```
@ECHO You must supply the following parameters:
```

```
@ECHO "webdriver.cmd <driver number>"
```

```
pause
```

```
:end
```

```
exit
```


Appendix E - Disk Storage

TPC-C 180-Day Disk Space Requirements						
Warehouses	1728	tpmc	21,354.55			
Table	Initial Rows	Data KB	Index KB	Extra 5% KB	Total With 5% KB	
Warehouse	1,728	192	48	12	252	
District	17,280	1,920	64	99	2,083	
Customer	51,840,000	37,701,824	2,420,960	2,006,139	42,128,923	
History (D)	51,840,000	2,880,064	0		2,880,064	
Order (D)	51,840,000	1,588,968	877,696		2,466,664	
New-Order	15,552,000	245,888	704	12,330	258,922	
Order-Line (D)	518,403,182	32,400,200	80,720		32,480,920	
Item	100,000	9,528	88	481	10,097	
Stock	172,800,000	55,296,000	123,944	2,770,997	58,190,941	
Totals KB		130,124,584	3,504,224	4,790,058	138,418,866	
Db/Filegroup	Count	Size MB	MB Allocated	MB Loaded +5%	MB for 8 Hours	
master, model & msdb	22	22	22	22	22	
tempdb	10	10	10	10	10	
mssql70_tpc_c_root	1	10	10	10	10	
mssql70_cs_fg	7	15,000	105,000	97,969	97,969	
mssql70_misc_fg	7	7,000	49,000	37,206	45,819	
Total Allocated MB			154,042	135,217	143,829	
MB						
Dynamic Space MB	36,005	Sum of data for orders, order_line & history				
Static Space	99,170	Sum of data+index+5% - Dynamic Space				
Free Space	18,867	Total allocated space - (Dynamic & Static Spaces)				
Daily Growth	7,119	(Dynamic Space / (W * 62.5)) * tpmC				
Daily Spread	8,189	Free space - 1.5 * Daily growth (zero if negative)				
	0	SQL Server can be configured to eliminate Daily Spread				
180 Day Space MB	1,380,624	Static Space + 180 * (Daily Growth + Daily Spread)				
180 Day Space GB	1,348.27					
8 hr log GB	51.56	(need double for mirroring)				
Disk Capacity MB	4339	4,2373 GB	Capacity of 4GB disks			
	8683	8,4795 GB	Capacity of 9GB disks			
Space Usage	GB Needed	Disks Priced	GB Priced	4GB drives	9GB drives	
180-day space DB	1348.27 GB	70	296.61 GB	4GB drives		
		126	1068.42 GB	9GB drives		
Total DB		196	1365.03 GB			
8-hr log+mirror	103.13 GB	14	118.71 GB	9GB drives		
OS, SQL Server	4.24 GB	1	4.24 GB	4GB drives		
Total space	1455.64 GB	211	1487.98 GB			

TPC-C 180-Day Dynamic Table Growth Rates for 8 Hours							21,354.55 tpmC
Tables	Initial (KB)	Final (KB)	Change(KB)	Unused (KB)	KB / New-Order	8-Hr MB	
History	2,880,064	3,591,896	711,832	61,784	0.0683	3,496.73	
Orders	2,466,664	3,887,024	1,420,360	17,136	0.1364	3,774.02	
Order_line	32,480,920	39,309,560	6,828,640	6,648	0.6557	38,282.93	
Dynamic	37,827,648	46,788,480	8,960,832	85,568	0.8604	45,553.68	
New_order	246,592	412,448	165,856	22,032	0.0159	400.22	
Static							
Log	419,271	54,930,328	54,511,057		5.2341	52,802.22	51.565
SUM(d_next_o_id)	51,857,280	62,271,936	10,414,656				

Appendix F - Third-Party Price Quotations

DEC 21 1998 15:04 FR MICROSOFT RECP #1 425 936 7329 TO 919493805539 P.02/03
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
Tel: 425 004 0000
Fax 425 936 7329
<http://www.microsoft.com/>



December 21, 1998

Mr. Jerrold Buggett
Director, Systems Analysis, Modeling, Measurement
Unisys Corporation
25725 Jeronimo Road
Mission Viejo, CA 92691

via FAX # 949-380-5539

Dear Jerry,

Microsoft has received your request for permission to disclose the results of TPC-C benchmark tests conducted by Unisys with Microsoft SQL Server 7.0, Enterprise Edition on the following system:

Unisys Aquanta QR2 Server, 4-processors, Pentium II Xeon, 450 MHz
Test Results: 21,300 ipmC @ \$24/ipmC approximately

Microsoft hereby grants Unisys permission to disclose these results to third parties and acknowledges that Unisys has formally requested permission to do so in accordance with the license agreement for Microsoft SQL Server 7.0 software.

Best regards,



Sid Arora
Product Manager, Microsoft SQL Server
Applications Marketing

DEC 21 1998 15:04 FR MICROSOFT RECP #1 425 936 7329 TO 919493805539 P.03/03
One Microsoft Way Fax 425 936 7329
Redmond, WA 98052-6399 http://www.microsoft.com/

Microsoft

December 21, 1998

Mr. Jerrold Buggert
Director, Systems Analysis, Modeling, Measurement
Unisys Corporation
25725 Jeronimo Road
Mission Viejo, CA 92691
via FAX # 949-380-5539

Dear Jerry,

Here is the information you requested regarding pricing of certain Microsoft products:

Microsoft SQL Server 7.0, Enterprise Edition, unlimited user licence	\$28999
Microsoft Windows NT Server, Enterprise Edition 4.0, incl 25 CALs	\$3999
Windows NT Server 4.0 software, incl 5 CALs	\$809
Visual C++ Professional 5.0	\$499
5-yr maintenance for above software @ \$2095/yr	\$10475

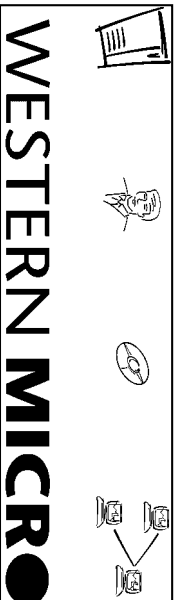
This quote is valid for the next 60 days. Please let me know if I can be of any further assistance.

Best regards,



Sid Arora
Product Manager, Microsoft SQL Server
Applications Marketing

** TOTAL PAGE 03 **



Western Micro Technology
 (800)937-8446

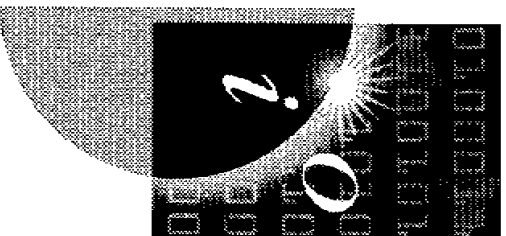
1/21/21/98

Quoted to: Jerry Bugger/Unisys for TPC.org
 Prepared by: Tony Jacobs

Qty.	Description	Style	Price	Extended Price
1	SYS: Aquantia QR/2, w/ CDRom, 0 Proc, 0MB Mem	QR2000101-BAS	\$10,820	\$10,820
4	PROC: 1x450MHz Pentium II/2MB Cache	XER2450-2MB	\$6,800	\$27,200
6	ACC: Voltage Regulator Module, Processor	XEO24001-VRM	\$46	\$276
32	MEM: 128 MB Memory Upgrade	DIM5072-128	\$459	\$14,688
1	DISK: 4GB Drive, Ultra SCSI SCA	HDS417-CX1	\$746	\$746
1	ETHERNET: 100Mbit/sec, PCI 32-bit	ETH1010051-PCI	\$117	\$117
1	CDROM: 14-32x Speed, SCSI	CDR1432-SI	\$161	\$161
1	MONITOR: 15-inch Color	EVG2100-P	\$320	\$320
1	KEYBD: 104 Key Spacesaver	PCK104-SKB	\$31	\$31
1	MOUSE: 2 Button PS2	PWM1-PS2	\$23	\$23
	Server Total			\$54,382
77	DISK: 4GB Drive, 10K, SCA	OSD4203-W45	\$625	\$48,125
154	DISK: 9GB Drive, 10K, SCA	OSD9203-W45	\$930	\$143,220
14	CAB: 7 SCA Disk Cage w/ 050 I/F & Cat Cbl, 3U	OSM310050-U05	\$1,345	\$18,830
14	CAB: 7 SCA Disk Cage w/ 057 I/F, 3U	OSM310057-U05	\$1,350	\$18,900
2	CAB: 7 SCA Disk Cage w/ 100 I/F, 0MB, 3U	OSM310100-U05	\$2,727	\$5,454
2	MEM: 32MB OSM Cache	OSM1000-C32	\$150	\$300
2	PWR: OSM 2nd Power Supply	OSM3000-APM	\$261	\$522
1	PWR: 3000 VA UPS, 3U	UPD30001-SXR	\$2,239	\$2,239
6	PWR: Distribution Unit, 6 OSM Cabinets	RM1-PDU	\$379	\$2,274
16	CBL: SCSI 68-pin HD Conn.	CBL2210-OSM	\$90	\$1,440
30	CAB: Rackmount Kit for Disk Cages	OSM3000-RMK	\$84	\$2,520
3	CAB: Rack Cabinet, w/ fill pnls, 36U	CAB361-SXR	\$1,384	\$4,152
3	CAB: Bezel kit 36U	BEZ3611-CAB	\$206	\$618
3	CAB: Stabilizer kit 0U	WGT39581-SXR	\$110	\$330
3	PNL: L&R side panels 36U	PAN3621-SXR	\$192	\$576
	Storage Total			\$249,500
3	SYS: NetServer LC3, w/1 450MHz Proc & CDROM, 0MI	D7026-AV	\$1,910	\$5,730
3	PROC: 1x450MHz Pentium II/512KB Cache UPG	D7032-AV	\$1,138	\$3,414
12	MEM: 128 MB SDRAM Memory Upgrade	D6098-AV	\$358	\$4,296
3	DISK: 4GB SCSI 3.5 Internal	D4910-AV	\$316	\$948
15	ETHERNET: 10/100TX Mbit/sec, PCI 32-bit	D5013-AV	\$68	\$1,020
3	ETHERNET: 100Mbit/sec, PCI 32-bit, Quad	SF1001-ET4	\$1,212	\$3,636
3	MONITOR: 15-inch Color	EVG2100-P	\$320	\$960
	Client Total			\$20,004
	Server, Storage and Client Total			\$323,886
	Discount based on total dollar volume			(\$32,389)
	Quote Total			\$291,497

Quote valid for 75 days.

Disks come with return to factory, 5 year warranty, 7 day replenishment



Quotation

To: Glenn Weeks	From: Michael Alam
Company: Unisys Corporation - Orange County Operations 25725 Jeronimo Rd Mission Viejo, CA 92531	Date: December 21, 1998
RE: Mylex ExtremeRAID DAC1164P	Total Pages: 1

Dear Mr. Weeks,

Mylex is pleased to submit the following quotation for the ExtremeRAID controller:

Part # / Description	MSRP
DAC1164P-3E-32-MY	\$2058 ea
PCI RAID Ultra 2 (LVD) DAC1164P 3E/2I CH 32MB W/Battery Back Up	

Notes:

Above prices are based on FOB, ex-factory, Fremont, California and firm for 90 days.

Lead time: 30 days ARO.

Product is covered by a 3 year warranty, and maintenance cost for additional 2 years is \$50.

Failed product will be repaired or replaced within 7 days.

Best regards,

Michael Alam, Regional Sales Manager

34551 Ardenvood Blvd.
Fremont, CA 94555-3807
Tel: 510.796-8100
Sales Fax: 510.745-8016
www.mylex.com

EQ01:390A

6CS508C661:01

5552809015

TJN:WDRJ 92:80 86 22-330



ENTERPRISE MIDDLEWARE SOLUTIONS

October 6, 1998

Mr. Jerrold Buggert
Director, Systems Analysis, Modeling, Measurement
Unisys Corporation
25725 Jeronimo Road
Mission Viejo, CA 92691
Fax (714) 380-5468

Dear Mr. Buggert:

Per your request I am enclosing the pricing information regarding TUXEDO 6.x that you requested. This pricing applies to Tuxedo 6.1, 6.2, 6.3 and 6.4. Please note that Tuxedo 6.4 is our most recent version of Tuxedo but that all 6.x releases are generally available. Core functionality services pricing is appropriate for your activities. As per the table below, server systems are classified in one of 5 tiers based on CPU type and capacity. The Acquanta GPS systems with 2 CPU capacity are classified as tier 1 systems, those with 4 CPU capacity are tier 2.

Tuxedo Core Functionality Services (CFS) Program Product Pricing and Description

TUX-CFS provides a basic level of middleware support for distributed computing, and is best used by organizations with substantial resources and knowledge for advanced distributed computing implementations.

TUX-CFS prices are server only and are based on the overall performance characteristics of the server and uses the same five tier computer classification as TUXEDO 6.x. Prices range from \$3,000 for Tier 1 to \$250,000 for Tier 5. Under this pricing option EVERY system running TUX-CFS at the user site must have a TUXEDO license installed and pay the appropriate per server license fees.

BEA Tux/CFS Unlimited User License Fees Per Server

Unlimited User License fees per Server	Number of Users	Dollar Amount	Maintenance (5 x 8) per year	Maintenance (7 x 24) per year
Tier 1 -- PC Servers with 1 or 2 CPUs, entry level RISC Uni-processor workstations and servers (Class 1and Class 2)	Unlimited	\$3,000.00	\$450.00	\$660.00
Tier 2 -- PC Servers with 3 or 4 CPUs, Midrange RISC Uni-processor servers and workstations (class 3)	Unlimited	\$12,000.00	\$1,800.00	\$2,640.00
Tier 3 -- Midrange Multiprocessors, up to 8 CPUs per system capacity (Class 4 and 5)	Unlimited	\$30,000.00	\$4,500.00	\$6,600.00

10/06/98

BEA SYSTEMS, INC.

Tier 4 -- Large (more than 8, less than 32 CPUs) and Mainframe Systems (Class 6)	Unlimited	\$100,000.00	\$15,000.00	\$22,000.00
Tier 5 -- Massively Parallel Systems, > 32 processors	Unlimited	\$250,000.00	\$37,500.00	\$55,000.00

Intel based server tier classifications:

Platform	Operating System	Tier 1	Tier 1	Tier 2	Tier 3	Tier 3
Intel Pentium/ Pentium Pro PCs	Interactive R3.2 ESIX SVR 4.0 SCO UNIX 3.2.2 and 3.2.4 SCO ODT 2.x,3.x Solaris x86 2.X UnixWare, Windows NT 3.5/4.0	All 386/486 PCs are Class 1	ALL Pentium and Pentium Pro PCs with 1 or 2 CPUs capacity are Tier 1	ALL Pentium and Pentium Pro PCs with 3 or 4 CPUs capacity are Tier 2		ALL Pentium and Pentium Pro PCs with 5,6,7, or 8 CPUs are Tier 3

Very Truly Yours,



Lewis D. Brentano,
Director, Market Planning

NOVEMBER 18th, 1998

RICK FREEMAN
 UNISYS CORPORATION
 25725 JERONIMO ROAD
 MISSION VIEJO, CALIFORNIA, 92691
 FAX: 949-380-5539

RICK
 HERE IS THE MODIFIED QUOTE AS REQUESTED BY GLEN WEEKS.

ITEM	QTY.	DESCRIPTION	UNIT	EXTENDED
DEH2924	2000+	COMPEX TP1008C 8 PORT 10BASE-T HUB WITH BNC UPLINK. LIFETIME WARRANTY.	\$ 33.00	\$ 66,000.00
DEH2648	2000+	LANTECH LTC WORKGROUP 8 PORT 10BASE-T HUB WITH SWITCH SELECTABLE 8 th PORT AS UPLINK OR DEDICATED 8 th PORT. 5 YEAR WARRANTY.	\$ 34.50	\$ 69,000.00
DEH4009	2000+	MACSENSE 8 PORT 10BASE-T HUB WITH BNC & RJ45 UPLINK PORTS. LIFETIME WARRANTY	\$ 33.95	\$ 67,900.00

THIS QUOTE IS VALID FOR 90 DAYS.
 THANK YOU FOR YOUR CONSIDERATION

BOB CHENEY
 ACCOUNT MANAGER
 DATACOMM WAREHOUSE
 800-328-2261 EXT: 22878
 732-942-2513 OR 732-905-5731 FAX
 CHENEYR@MMWHSE.COM



NETLUX

14180 Live Oak Ave., Unit E
Baldwin Park, Ca. 91760

1-800-789-1780
Phone#626-851-9737
Fax #626-851-9837

December 2, 1998

Rick Freeman
Unisys Corporation
25725 Jeronimo Road
Mission Viejo, CA 92691
Fax: (949) 380-5539
cc: (949) 380-5344

Quotation

Quantity	Part No.	Description	Unit Price	Total
3	NX-H8TXD	8-port 100Mbps FAST Ethernet Hub	\$205.00	\$ 615.00

Terms and Conditions:
FOB Origin
Quote Valid for 90 days
5 Year Warranty

Sincerely,
Martin Parry
NETLUX