TPC Benchmark C®
Full Disclosure Report

Compaq AlphaServer GS320 Model 6/731 Client/Server System Using Oracle v8.1.7 Enterprise Edition for Tru64 UNIX ® and Compaq Tru64 UNIX 5.1®

Company Name	System Name	Database Software	Operating System Software
Compaq Computer Corporation	**Compaq AlphaServer GS320 Model 6/731 32 CPU Client/Server System**	**Oracle v8.1.7 Enterprise Edition for Tru64 UNIX**	**Compaq Tru64 UNIX V5.1**

Availability Date: February 2, 2001

Total System Cost	TPC-C Throughput	Price Performance
- Hardware - Software - 5-Years Maintenance **$8,641,121**	Sustained maximum throughput of system running TPC Benchmark C expressed in transactions per minute	Total system cost/ TPC-C® throughput **155,179/$55.68**

First Printing November 2000

Compaq Computer Corporation believes that the information in this document is accurate as of its publication date; such information is subject to change without notice. Compaq Computer Corporation is not responsible for any inadvertent errors.

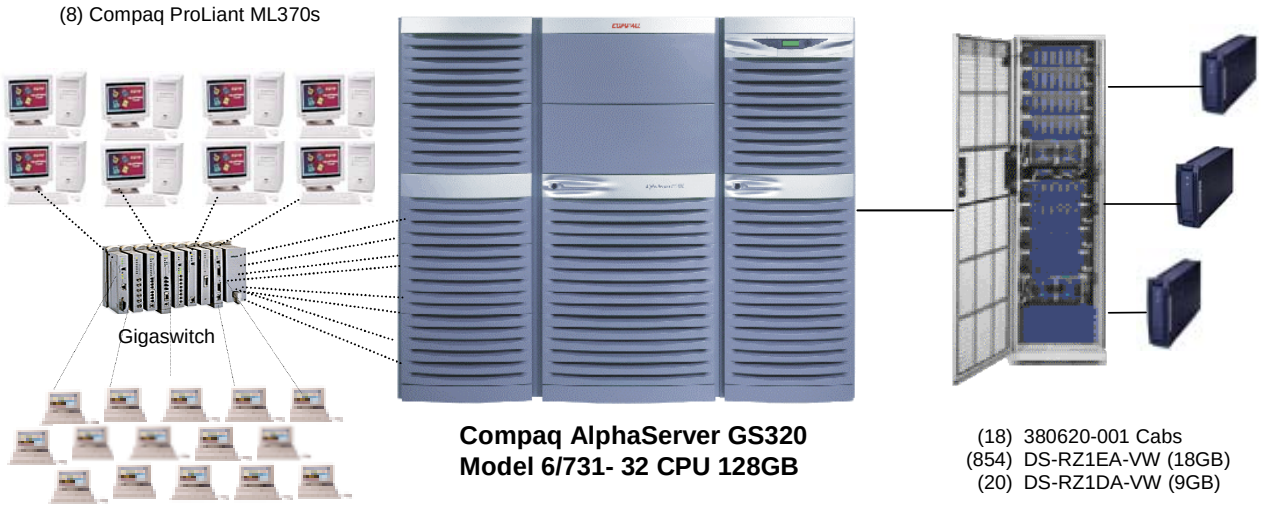
Compaq conducts its business in a manner that conserves the environment and protects the safety and health of its employees, customers, and the community.

The performance information in this document is for guidance only. System performance is highly dependent on many factors, including system hardware, system and user software, and user application characteristics. Customer applications must be carefully evaluated before estimating performance. Compaq Computer Corporation does not warrant or represent that a user can or will achieve similar performance expressed in transactions per minute (tpmC) or normalized price/performance ($/tpmC). No warranty on system performance or price/performance is expressed or implied in this document.

COMPAQ ORACLE®	Compaq AlphaServer GS320 Model 6/731 **32 CPU Client/Server System**	TPC-C Rev. 3.5 **Report Date**: November 10, 2000

Total System Cost	TPC-C Throughput	Price/Performance	Availability Date
$8,641,121	155,179.25	$55.68	February 2, 2001

Processors	Database	Operating System	Other Software	Number of Users
(32) 731MHz Alphachip 21264A	Oracle v8.1.7 Enterprise Edition for Tru64 UNIX	Compaq Tru64 UNIX V5.1	Compaq DB Web Connector V1.1	135,040



	System Components Database Server, Compaq AlphaServer GS320 Model 6/731		Client Components (Each of 8) Compaq ProLiant ML370	
	Qty	**Type**	**Qty**	**Type**
Processors	32	6/731 MHz Alphachip 21264A	2	600 MHz Intel Pentium III
Cache Memory		4MB Cache	1	512KB Cache
Memory	32	4 GB Memory	1	128MB Memory
Total Memory	128	GB Memory	2	256MB Memory
Disk Controllers	52	Fiber Storage Controllers		640MB Total
Disks	20	9.1GB Disks	1	18GB Disk
	854	18.1GB Disks		
Total Disk Storage	15,639.4	GB		

COMPAQ	AlphaServer GS320 Model 6/731 32 CPU C/S System	TPC-C REV 3.5 EXECUTIVE SUMMARY PAGE 2 OF 2 Report Date: Nov.10, 2000

Description	Part Number	Third Party Brand	Unit Price	Qty	Extended Price	5 yr. Maint. Price
Server Hardware			Pricing			
Compaq AlphaServer GS320 6/731 Model 32	DA-320EC-AA	1	666,000	1	666,000	476,962
GS160/320 ADD-on CPU UNIX SMP	3X-KN8AA-AD	1	23,000	31	713,000	Inc.
GS80/160/320 2GB MEMORY OPT	3X-KN8AA-CB	1	26,000	32	832,000	Inc.
GS80/160/320 2GB MEMORY UPG	3X-KN8AA-CU	1	22,000	32	704,000	Inc.
PCI Fibre Adapter	DS-KGPSA-CA	1	4,000	52	208,000	Inc.
PCI to Ultra SCSI Adapter	KZPBA-CA	1	460	2	920	Inc.
D shf 180w Dif Blw Mtrc blue	DS-BA356-JF	1	1,464	2	2,928	Inc.
16Bit SCSI for KZPSC/BA356	BN31S-1E	1	116	2	232	Inc.
1600 Watt Bulk Power Converter	H7506-AA	1	10,000	4	40,000	Inc.
GS160/320 PCI Expansion Box	3X-DWWPA-BA	1	12,000	8	96,000	Inc.
GS80/160/320 Expansion Cab	H9A20-AA	1	5,000	2	10,000	Inc.
15M Fibre Op Cable 50/125	234457-B23	1	227	52	11,804	Inc.
PCI to gbE SX Adapter	DEGPA-SA	1	1,949	8	15,592	Inc.
PCO Mempru Channel Two Controller	CCMAB-AA	1	2,990	1	2,990	Inc.
SC-SC DUAL FO Cbl, MM, PP, 4.5M	BN34B-4E	1	96	8	768	0
ESA 12000 Storage Array	380620-001	1	88,629	18	1,595,322	285,642
15M Fibre Op Cable 50/125	234457-B23	1	227	144	32,688	0
9.1GB 7200rpm UWSCSI Disk**	DS-RZ1DA-VW	1	650	21	13,650	Spared
18.2GB 7200rpm UWSCSI Disk**	DS-RZ1EA-VW	1	1,195.45	940	1,123,723	Spared
16 port Fibre Switch	DS-DSGGB-AB	1	35,714	13	464,282	175,448
Short Wave Optical GBIC	3380561-B21	1	492	196	96,432	Inc.
12/24gb 4MM DAT Tape	TLZ10-LB	1	1,135	1	1,135	1,274
GS-Router 16 Slot Chassis	DGSRA-BA	1	9,995	2	19,990	5,736
GS-/R 64MB Switch Control Pro	DGSRC-BA	1	8,995	2	17,990	5,736
GS/Rtr 2-P Geeth SX 16mB Line Card	DGSRS-AB	1	7,995	8	63,960	22,944
GS/R 16-Slot Swtiching Fabric Module	DGSRD-AA	1	4,995	4	19,980	5,400
GS/Rtr 8-P Fast Ethernet 16mb Line Card	DGSRT-AB	1	5,995	17	101,915	34,408
VT510, WHITE, NORTH AMER, NO KEY	VT510-AA	1	395	1	395	0
US & Canada/English KB	PCXLA-NA	1	25	1	25	0
				Subtotal	**6,855,721**	**1,013,550**
Server Software						
5YR TRU64UNIX BRZ9x5	FM-W16U9-60	1	34,074	1		34,074
5YR GS160/320 TRU64 UNIX SMP	FM-W16SM-60	1	2,592	31		80,352
5YR Tru64Unix O/S & LP	FM-CDDST-60	1	13,624	1		13,624
HSG80 ACSsf All Lic/PCRM Pkg	128697-B21	1	7,000	72	504,000	
GIGAsw/Rtr All Lic/MCD Pkg	QB-64PAA-WA	1	3,995	1	3,995	
5YR 5x9 HS*80 SW Dual Controller	FM-DBAL6-60	1	9,360	36		336,960
5YR 5x9 HS*90 Platform SW	FM-PBAT1-60	1	1,514	36		54,504
TRU64 UNIX Alpha CDRM	QA-MT4AA-H8	1	395	1	395	
Oracle v 8.1.7 Enterprise Edition for Tru64 UNIX	Oracle	2	1,224,158	1	1,224,158	1,224,158
				Subtotal	**1,732,548**	**1,743,672**
Client Hardware						
ML370T01 P600-256 128 US	195293-001	1	3,855	8	30,840	21,920
Processor 600Mhz	128292-B21	1	1,005	8	8,040	Inc.
256MB SDRAM DIMM Memory	128278-B21	1	1,205	16	19,280	Inc.
Ethernet Dual Channel	338456-B21	1	285	8	2,280	Inc.
Gigabit Daughter CD Upgrade	338456-B23	1	570	8	4,560	Inc.
18GB Drive	388144-B22	1	556	8	4,448	Inc.
SC-SC DUAL FO Cbl, MM, PP, 4.5M	BN34B-4E	1	96	8	768	Inc.
V700 15" Color Monitor**	325900-001	1	125	10	1,250	Spared
				Subtotal	**71,466**	**21,920**
Client Software						
Application Optimizer WNT Server Edition	QB-641AA-SA	1	450	8	3,600	2,880
				Subtotal	**3,600**	**2,880**
User Connectivity						
8 Port Desktop Hub/Plastic***	CT1008D/P	3	18	18568	334,224	Spared
				Subtotal	**334,224**	
Notes: 10% Spares**						
ICS Discount: 38% on Server Hardware/Software based on volume.					**($2,826,887)**	
ICS Discount: 20% on FM and FR numbers based on volume.						**($311,573)**
Subtotal					**$6,170,672**	**$2,470,449**
Five Year Cost of Ownership						**$8,641,121**
1=IC System Solutions, 2=Oracle, 3= NetCruiser Technologies					**tpmC Rating:**	**155,179.25**
Audited by InfoSizing (www.infosizing.com)					**$ / tpmC:**	**$55.68**

Prices used in TPC benchmarks reflect the actual prices a customer would pay for a one-time purchase of the stated components. Individually negotiated discounts are not permitted. Special prices based on assumptions about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed components. For complete details, see the pricing sections of the TPC benchmark 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.

Numeric Quantities Summary
AlphaServer GS320 6/731 Client/Server

MQTH

MQTH, computed Maximum Qualified Throughput	155179
% throughput difference, reported and reproducibility runs	.001

Response Times in seconds

Transaction	90th percentile	Average	Maximum
New-Order	1.826	1.050	9.952
Payment	1.780	1.037	9.567
Order-Status	1.676	0.947	8.727
Delivery (interactive)	1.190	0.625	9.176
Delivery (deferred)	1.141	0.611	7.907
Stock-Level	1.989	1.223	9.755
Menu	1.202	0.607	9.234

Transaction Mix in percent of total transactions

Transaction	Total Occurrences	Percentage
New-Order	18621511	44.980%
Payment	17803546	43.004%
Order-Status	1658365	4.006%
Delivery (interactive)	1658112	4.005%
Stock-Level	1657747	4.004%

Keying/Think Times in seconds

Transaction	Minimum	Average	Maximum
New-Order	18.000/0.000	18.001/13.904	18.192/140.005
Payment	3.000/0.000	3.000/12.004	3.172/120.007
Order-Status	2.000/0.000	2.001/10.003	2.163/ 99.977
Delivery (interactive)	2.000/0.000	2.001/ 5.002	2.163/ 49.963
Stock-Level	2.000/0.000	2.001/ 5.003	2.163/ 49.920

Emulation Delay in seconds

Transaction	Response Time	Menu Response Time
New-Order	0.10	0.10
Payment	0.10	0.10
Order-Status	0.10	0.10
Delivery (interactive)	0.10	0.10
Stock-Level	0.10	0.10

Test Duration

Ramp up time	60 minutes
Measurement interval	120:00.00 minutes
Number of checkpoints	4
Number of New Order transactions	18621511
Number of transactions (all types) completed in measurement interval	41399281

Abstract

This report documents the compliance of Compaq Computer Corporation's and Oracle Corporation's TPC Benchmark C tests on the Compaq AlphaServer GS320 Model 6/731 – 32 CPU Client/Server system with Version 3.5 of the *TPC Benchmark C Standard Specification*. Eight Compaq ProLiant ML370 computers were used as the front-end clients. One of these systems was also used as a development system.

Two standard metrics, transactions-per-minute-C (tpmC) and price per tpmC ($/tpmC) are reported, in accordance with the TPC Benchmark C Standard. The independent auditor's report by InfoSizing is appended at the end of this report.

Table of Contents

Preface

This report documents the compliance of Compaq Computer Corporation's and Oracle Corporation and TPC Benchmark C (TPC-C) testing on the Compaq AlphaServer GS320 Model 6/731 32 CPU client/server system with Version 3.5 of the TPC Benchmark C Standard Specification[1]. The TPC-C Standard represents an effort by Compaq Computer Corporation and other members of the Transaction Processing Performance Council (TPC) to create industry-wide benchmarks for evaluating the performance and price/performance of transaction processing systems.

These tests were run using Compaq Database Web Connector V1.1 transaction processing monitor, Oracle V8.1.7 Enterprise Edition for Tru64 UNIX, and Compaq Tru64 UNIX V5.1 operating system. Eight Compaq ProLiant 1600 computers were used as the front-end clients. One of these systems was also used as a development system.

About the TPC-C Benchmark

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
- 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.

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

[1] *TPC Benchmark C Standard Specification*, Transaction Processing Performance Council, Version. 3.5.

Document Structure

This *TPC Benchmark C Full Disclosure Report* is organized as follows:

- The main body of the document lists each item in Clause 8 of the TPC-C Standard and explains how each requirement is satisfied.
- Appendix A contains the source code of the TPC-C application code used to implement the TPC-C transactions.
- Appendix B contains the database definition and population code used in the tests.
- Appendix C contains the Remote Terminal Emulator (RTE) code used to generate and record transactions.
- Appendix D contains the tunable parameters.
- Appendix E contains the independent auditor's report on the compliance of this disclosure with the benchmark specifications.
- Appendix F contains price quotes.

Additional Copies

To request additional copies of this report, please contact:

Administrator, TPC Benchmark Reports
High Performance Servers Division
Benchmark Performance Engineering
Compaq Computer Corporation
110 Spit Brook Road (ZK02-3/M31)
Nashua, NH 03062
USA

FAX Number: 603-884-6082

TPC Benchmark C Full Disclosure

The *TPC Benchmark C Standard Specification* requires test sponsors to publish, and make available to the public, a full disclosure report for the results to be considered compliant with the Standard. The required contents of the full disclosure report are specified in Clause 8. This report is intended to document the compliance of the benchmark tests with each item listed in Clause 8 of the *TPC Benchmark C Standard Specification*.

In the *Standard Specification*, the main headings in Clause 8 are keyed to the other clauses. The headings in this report use the same sequence, so that they correspond to the titles or subjects referred to in Clause 8.

Each section in this report begins with the text of the corresponding item from Clause 8 of the Standard Specification, printed in italic type. The plain type text that follows explains how the tests comply with the TPC Benchmark C requirement. In sections where Clause 8 requires extensive listings, the section refers to the appropriate appendix at the end of this report.

1. General Items

1.1 Order and Titles

The order and titles of sections in the Test Sponsor's Full Disclosure Report must correspond with the order and titles for the TPC-C standard specification. 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 sections in this report correspond with that of the TPC-C standard specification.

1.2 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 at the beginning of this report.

1.3 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,*
- *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, and*
- *percentage of transaction mix for each transaction type.*

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

1.4 Application Program

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

Appendix A contains the Compaq C application code and the Oracle anonymous block SQL code.

1.5 Sponsor

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

This benchmark test was sponsored by Compaq Computer Corporation and Oracle Corporation, and attested to by InfoSizing.

1.6 Parameters and Options

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

- *Database tuning options*
- *Recovery/locking options*
- *Operating system and application configuration parameters*

The Test Sponsor has elected to provide a listing of all parameters and options. Appendix C contains the tunable parameters used in the TPC-C tests.

1.7 Configuration

Provide diagrams of both the measured and priced configuration, 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 or memory unique to the test.*
- *Number and type of disk drive 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.*
- *Type and the run-time execution location of software components (e.g., DBMS, client processes, transaction monitors, software drivers, etc.)*

The TPC-C terminal users were emulated by Compaq's Portable Remote Terminal Emulator (PRTE) software, which ran on four AlphaServer ES40 6/500 computers. Each emulated terminal was connected using HTTP (HyperText Transport Protocol) to a client node.

The clients consisted of 8 Compaq ProLiant ML370 computers running the TPC-C web client screen and application software. Each emulated terminal ran the TPC-C client application and made a request to the Oracle V8.1.7 Enterprise Edition for Tru64 UNIX using OCI calls.

The system consisted of 1 AlphaServer GS320 with 32 CPUs and 128GB of memory. The measured server was configured with 744 DS-RZ1DA–VW (9.1GB) disks and 24 DS-RZ1EA-VW (18.1GB) disks. The priced configuration consists of 22 DS-RZ1DA-VW and 940 DS-RZ1EA-VW disks, which includes an additional 10% spareable disks.

Measured Configuration

The following figure represents the measured configuration. The benchmark system used a remote terminal emulator (RTE) to initiate transactions and measure response times of transactions, as well as record various data for each transaction.

Compaq AlphaServer GS320
Model 6/731 – 32 CPU 128GB

8 Compaq ProLiant 1600's

18 380620-001 Cabs
744 DS-RZ1DA-VW (9.1GB)
24 DS-RZ1EA-VW (18.1GB)

4 AlphaServer ES40 6/500 PRTE Systems

Priced Configuration

The following figure depicts the priced system, whose cost determines the normalized price per tpmC reported for this test.

Compaq AlphaServer GS320
Model 6/731 – 32 CPUs 128GB

8 Compaq ProLiant ML370s

18 380620-001 Cabs
854 DS-RZ1EA-VW (18GB)
20 DS-RZ1DA-VW (9GB)

133 Ethernet Segments
on 10/100 Mbps
Fast Ethernet
1,016 Users per Segment
Total PC'S -135,040

2. Logical Database Design Related Items

2.1 Table Definitions

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

Appendix B contains the database definition files that were used to set up the database.

2.2 Table Organization

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

The scripts used to build and load the database are in Appendix B.

2.3 Insert/Delete Operations

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

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

2.4 Disclosure of Partitioning

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

Horizontal partioning was used on the warehouse, district, new_order, history, order, and tables using functionality provided by Oracle V8.1.7 Enterprise Edition for Tru64 UNIX. For further details, see Appendix B.

2.5 Replication of Tables

Replication of tables, if used, must be disclosed.

No tables were replicated in this benchmark test.

2.6 Additional and/or Duplicated Attributes in any Table

Additional and/or duplicated attributes in any table must be disclosed along with a statement on the impact on performance.

No attributes were added or replicated in this benchmark test.

3. Transaction and Terminal Profiles Related Items

3.1 Random Number Generation

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

Random numbers were generated using the drand48() and lrand48() UNIX calls. These functions generate pseudo randum numbers using the linear congruential algorithm and 48-bit integer arithmetic. The random number generators are initially seeded using the srand48() call.

3.2 Terminal Input/Output Screen Layouts

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

The screen layouts match the *TPC Benchmark C Standard Specification*.

3.3 Features in Priced Terminals

The method used to verify that the priced terminals provide all the features described in Clause 2.2.2.4 must be explained.

Each of the five transaction types were tested by the auditor from an Intel 486DX2/66 running Windows NT 4.0 SP2 and Netscape Navigator V3.0. The auditor verified that all the features specified in Clause 2.2.2.4 were provided. Any PC configured with any WWW browser will properly display the TPC-C screens.

3.4 Presentation Managers

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

The code to generate the input and menu screens and display the results runs on the front-end clients. The data is passed to the user's PC using the HTML (HyperText Markup Language) format, which can be displayed with any Web browser, such as Netscape Navigator or Internet Explorer. Both of these products come as standard software with many operating systems.

3.5 Home and Remote Order-Lines

The percentage of home and remote order-lines in the New-Order transactions must be provided.

The table in Section 3.10 shows the percentage of home and remote transactions that occurred during the measurement period for the New-Order transactions.

3.6 New Order Roll Back Transition

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

The table in Section 3.10 shows the percentage of New-Order transactions that were rolled back due to an invalid item number being entered.

3.7 Number of Order-Lines

The table in Section 3.10 shows the average number of items ordered per New-Order transaction.

3.8 Home and Remote Payment Transactions

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

The table in Section 3.10 shows the percentage of home and remote transactions that occurred during the measurement period for the Payment transactions.

3.9 Non-Primary Key Access in Payment and Order-Status

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

The table in Section 3.10 shows the percentage of non-primary key accesses to the database by the Payment and Order-Status transactions.

3.10 Skipped Deliveries

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

The following table summarizes the data required for disclosure from Sections 3.5 through 3.10. The range of acceptable and the measured results are listed.

Description	Section	Acceptable Requirement	Measured Result
% home order lines in New Order	3.5	(98.05 - 99.05)	99.00%
% remote order lines in New Order	3.5	(0.95 – 1.05)	1.00%
% New Order transactions rolled back	3.6	(0.9 - 1.1)	0.99%
Average number of items per order	3.7	(9.5 - 10.5)	10.00
% home Payment transactions	3.8	(84.0 - 86.0)	84.99%
% remote Payment transactions	3.8	(14.0 - 16.0)	15.01%
% non-primary key access, Payment	3.9	(57.0-63.0)	60.01%
% non-primary key access, Order Status	3.9	(57.0-63.0)	59.96%
% skipped deliveries	3.10	(0.0 - 1.0)	0

3.11 Transaction Mix

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

The following table summarizes the transaction mix.

Transaction Mix in percent of total transactions

Transaction	Total Occurrences	Percentage
New-Order	18621511	44.980%
Payment	17803546	43.004%
Order-Status	1658365	4.006%
Delivery (interactive)	1658112	4.005%
Stock-Level	1657747	4.004%

3.12 Queuing Mechanism

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

The delivery transactions were transmitted to a separate delisrv process using a pipe. The pipe acts as a FIFO queue. The delisrv has multiple threads that can do the delivery transaction in the database. The delivery data is written to a file. The number of threads in the delisrv is configurable.

4. Transaction and System Properties Related Items

4.1 ACID Properties

The results of the ACID test 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.

Clause 3 of the TPC Benchmark C Standard Specification lists specific tests to ensure the Atomicity, Consistency, Isolation, and Durability (ACID) properties of the SUT. The following subsections show how the tests required in Clause 3 were performed and the results verified. All mechanisms needed to ensure full ACID properties were enabled during both the measurement and test periods. A fully-sized, 13,504 warehouse database was used for the Consistency and system crash, Atomicity, Isolation, loss of log, and loss of data file Durability tests.

Case A was followed for the execution of Isolation Test 7, as described in the TPC-C Standard Specification, Clause 3.4.2.7.

4.2 Atomicity Tests

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

4.2.1 Atomicity of Completed Transaction

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number as specified in Clause 2.5.1.2) and verify that the records in the CUSTOMER, DISTRICT, and WAREHOUSE tables have been changed appropriately.

The following test was performed, and it verified the atomicity of completed transactions:

1. Select a random warehouse, district, customer, and payment amount.
2. For the selected warehouse, district, and customer, record their contents.
3. Perform a PAYMENT transaction, and COMMIT the transaction.
4. For the selected warehouse, district, and customer, verify the following:
   A. W_YTD in the warehouse is increased by the payment amount.
   B. D_YTD in the district is increased by the payment amount.
   C. C_YTD_PAYMENT in the customer is increased by the payment amount, C_BALANCE is decreased by the payment amount, and C_PAYMENT_CNT is incremented by 1.

4.2.2 Atomicity of Aborted Transaction

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number as specified in Clause 2.5.1.2) 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 following test was performed, and verified the atomicity of aborted transactions:

1. Select a random warehouse, district, customer, and payment amount.
2. For the selected warehouse, district, and customer, record their contents.
3. Perform a PAYMENT transaction and substitute a ROLLBACK for the COMMIT.
4. For the selected warehouse, district, and customer, verify that records have not been changed.

4.3 Consistency Tests

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

The consistency of the database was checked before and after the power fail Durability test. The following consistency conditions were run and verified that all four conditions were met:

4.3.1 Consistency Condition 1

Entries in the WAREHOUSE and DISTRICT tables satisfy:
W_YTD = sum(D_YTD) for each warehouse defined by (W_ID = D_W_ID)

4.3.2 Consistency Condition 2

Entries in the DISTRICT, ORDER, and NEW-ORDER tables satisfy:
D_NEXT_O_ID - 1 = max(O_ID) = max(NO_O_ID) for each district defined by
(D_W_ID = O_W_ID = NO_W_ID) and (D_ID = O_D_ID = NO_D_ID)

4.3.3 Consistency Condition 3

Entries in the NEW-ORDER table satisfy:
max(NO_O_ID) - min(NO_O_ID) = [# of rows in NEW-ORDER of this district] for each district defined by NO_W_ID and NO_D_ID

4.3.4 Consistency Condition 4

Entries in the ORDER and ORDER-LINE tables satisfy:
sum(O_OL_CNT) = [# of rows in ORDER-LINE of this district] for each district defined by (O_W_ID = OL_W_ID) and (O_D_ID = OL_D_ID).

4.4 Isolation Tests

The TPC Benchmark C Standard Revision 3.3.3 defines seven required tests to be performed to demonstrate that the required levels of transaction isolation are met. All seven required tests were performed successfully. In addition to those seven tests, two more tests specified by the auditor were performed successfully. These additional tests demonstrated phantom protection within any mix of TPC-C transactions.

4.4.1 Isolation Test 1

This test demonstrates isolation for read-write conflicts of Order-Status and New-Order transactions.
The following steps were performed to satisfy the test of isolation for Order-Status and New-Order transactions.

1. 1st terminal: Start a New-Order transaction with the necessary inputs. The transaction is delayed for 20 seconds just prior to the Commit.
2. 2nd terminal: Start an Order-Status transaction for the same customer as used in the New-Order transaction.
3. 2nd terminal: The Order-Status transaction attempts to read the file but is locked out by the New-Order transaction waiting to complete.
4. 1st terminal: The New-Order transaction is released, and the Commit is executed releasing the record. With the CUSTOMER record now released, the Order-Status transaction can now complete.
5. 2nd terminal: Verify that the Order-Status transaction delays for approximately 20 seconds and that the results displayed for the Order-Status transaction match the input for the New-Order transaction.

4.4.2 Isolation Test 2

This test demonstrates isolation for read-write conflicts of Order-Status and New-Order transactions when the New-Order transaction is rolled back.

The following steps were performed to satisfy the test of isolation for Order-Status and a rolled back New-Order transactions.

1. 1st terminal: Perform an Order-Status transaction for a random customer. The transaction is completed.
2. 2nd terminal: Perform a New-Order transaction for the same customer in the Order-Status transaction in Step 1; include an invalid item number in the last order. The transaction is delayed for 20 seconds just prior to the roll back.
3. 2nd terminal: Start another Order-Status transaction for the same customer used in the New-Order transaction. The Order-Status transaction attempts to read the CUSTOMER table but is locked by the New-Order transaction.
4. 2nd terminal: Roll back the New-Order transaction. With the CUSTOMER record now released, the Order-Status transaction completes.
5. Verify the results returned from the 1st and 2nd Order-Status transactions match.

4.4.3 Isolation Test 3

This test demonstrates isolation for write-write conflicts of two New-Order transactions.
The following steps were performed to verify isolation of two New-Order transactions.

1. 1st terminal: Start a New-Order transaction using the necessary inputs. The transaction is delayed for 20 seconds just prior to the Commit.

2. 2nd terminal: Start a second New-Order transaction for the same customer used by the 1st terminal. This transaction is forced to wait while the 1st terminal holds a lock on the CUSTOMER record requested by the 2nd terminal.
3. 1st terminal: The New-Order transaction is allowed to complete and Commit the transaction. With the CUSTOMER record released, the 2nd terminal New-Order transaction will complete.
4. Verify that the order number from the 2nd terminal New-Order transaction is 1 greater than the order number from the 1st terminal. Verify that D_NEXT_O_ID is 1 greater than the order number of the New-Order transaction from the 2nd terminal.

4.4.4 Isolation Test 4

This test demonstrates isolation for write-write conflicts of two New-Order transactions when one transaction is rolled back.

The following steps were performed to verify the isolation of two New-Order transactions after one is rolled back.

1. 1st terminal: Start a New-Order transaction using the necessary inputs to cause a roll back (invalid item number). The transaction is delayed for 20 seconds just prior to the roll back.
2. 2nd terminal: Start a second New-Order transaction for the same customer used by the 1st terminal. This transaction is forced to wait while the 1st terminal holds a lock on the CUSTOMER record requested by the 2nd terminal.
3. 1st terminal: The New-Order transaction is allowed to complete, and the transaction is rolled back due to the invalid item number.
4. 2nd terminal: With the CUSTOMER record released, the 2nd terminal New-Order transaction will complete normally.
5. Verify that the order number from the 2nd terminal New-Order transaction is equal to the next order number before either New-Order transaction was started. Verify that D_NEXT_O_ID is 1 greater than D_NEXT_O_ID before either transaction was started and is 1 greater than the order number for the 2nd New-Order transaction.

4.4.5 Isolation Test 5

This test demonstrates isolation for write-write conflicts of Payment and Delivery transactions.

The following steps were performed to successfully conduct this test.

1. 1st terminal: A Delivery transaction is started. The Delivery transaction batch job is started with code added to the transaction to delay the transaction just prior to the Commit. (Note: This delay is only used during the Isolation tests.)
2. 2nd terminal: Start a Payment transaction for a customer that will have an order delivered in the transaction started in Step 1. The Payment transaction is forced to wait while the Delivery transaction holds a lock on the CUSTOMER record.
3. 1st terminal: After the delay time period is expired for the Delivery transaction, the transaction is Committed.
4. 2nd terminal: With the CUSTOMER record released, the Payment transaction is now able to complete.
5. The record is viewed from the CUSTOMER table for the customer and C_BALANCE is verified to reflect the changes from the Delivery and Payment transactions.

4.4.6 Isolation Test 6

This test demonstrates isolation for write-write conflicts of Payment and Delivery transactions when the Delivery transaction is rolled back.

The following steps were performed to successfully conduct this test.

1. 1st terminal: Start a Delivery transaction using the Delivery batch job with the delay coded in to halt the transaction just prior to the roll back.
2. 2nd terminal: Start a Payment transaction for a customer that will have an order delivered in the transaction started in Step 1. The Payment transaction is forced to wait while the Delivery transaction holds a lock on the CUSTOMER record.
3. 1st terminal: The Delivery transaction is rolled back after the time period expires.
4. 2nd terminal: With the CUSTOMER record released, the Payment transaction is now able to complete.
5. The record is viewed from the CUSTOMER table for the customer, and C_BALANCE is verified to reflect he change only from the Payment transaction.

4.4.7 Isolation Test 7

This test demonstrates repeatable reads for the New-Order transaction while an interactive transaction updates the price of an item.

The following test demonstrates that this test was conducted successfully.

1. 1st terminal: Using interactive SQL, view and record prices for two items (x, y) selected at random.
2. 2nd terminal: A New-Order transaction is started that contains item x twice and y once. This transaction is stopped after reading the price of item x from the item file the first time.
3. 1st terminal: Using interactive SQL, an update transaction is started for items x and y, increasing their price by 10%. Case A, transaction 3 stalls.
4. 2nd terminal: The New-Order transaction is allowed to complete. It is verified that the prices for items x and y are the same throughout the entire transaction and that they match the results of Step 1.
5. 1st terminal: After the New-Order transaction completes, the update transaction completes and is committed.

4.4.8 Isolation Test 8

This test demonstrates isolation for phantom protection between New-Order and Order-Status transactions.

The execution of the above test proceeded as follows.

1. For a randomly selected warehouse and district, the NO_D_ID column in NEW_ORDER is changed to 11.
2. 1st terminal: A Delivery transaction is started for the selected warehouse. The earliest New-Order for the selected warehouse and district is printed out and should be NULL. The delivery is paused for 30 seconds after reading the New-Order table.
3. 2nd terminal: A New-Order transaction is started for the selected warehouse and district. It is blocked by the other transaction.
4. 1st terminal: After the 30 seconds pause, the Delivery transaction re-reads the NEW_ORDER entries for the selected warehouse and district and verifies that no new orders exist. The Delivery transaction commits, showing nothing delivered for the selected district.
5. 2nd terminal: The blocked New-Order completes only after the Delivery commits.
6. The NO_D_IDs that were set to 11 are restored to their original values.

4.4.9 Isolation Test 9

This test demonstrates isolation for phantom protection between New-Order and Delivery transactions.
The execution of the above test proceeded as follows.

1. 1st terminal: An Order-Status transaction is started for a randomly selected customer. It is paused for 30 seconds after reading the ORDERS table for that customer. The most recent order is recorded.
2. 2nd terminal: A New-Order transaction is started for the same customer. It is blocked by the paused Order-Status transaction.
3. 1st terminal: After the 30 seconds pause, the Order-Status transaction re-reads the ORDERS entry for the selected customer and verifies that the most recent order is the same in Step 1. The Order-Status commits.
4. 2nd terminal: The blocked New-Order transaction completes only after the Order-Status transaction finishes.

4.5 Durability Tests

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

- *Permanent irrecoverable failure of any single durable medium containing database, ABTH files/tables, or recovery log data.*
- *Instantaneous interruption (system crash/system hang) in processing which requires system reboot to recover.*
- *Failure of all or part of memory (loss of contents).*

Tests were conducted for each of the preceding types of failures and successfully demonstrated that the durability properties were met.

4.5.1 Failure of Durable Medium containing TPC-C Database Tables and Failure of Durable Medium containing Recovery Log Data

This test was conducted on a 13,504 warehouse database with 16,880 users. The following steps were performed successfully.

1. The database file was copied to an extra disk using 'dd'.
2. The current number of orders in the database was counted, giving ORDER_COUNT_BEFORE.
3. A test was started and allowed to run at steady state for 3 minutes.
4. One of the log disks from a mirrored pair was removed. The test continued to run, using the remaining disk. An error was logged in the RAID manager software.
5. The test was allowed to run for 3 more minutes.
6. One of the data disks was removed. Errors were reported by the operating system.
7. The emulated users received errors.
8. Oracle V8.1.7 Enterprise Edition for Tru64 UNIX was stopped.
9. The backup of the database file was restored to a replacement disk.
10. Oracle V8.1.7 Enterprise Edition for Tru64 UNIX was restarted indicating that media recovery was necessary.
11. The saved transaction log was replayed.
12. The current number of orders in the database was counted giving ORDER_COUNT_AFTER. It was verified that ORDER_COUNT_AFTER - ORDER_COUNT_BEFORE was greater than or equal to the number of committed orders recorded by the RTE.
13. Several orders recorded by the RTE were checked in the database to make sure they existed.

4.5.2 Failure of Memory and Instantaneous Interruption

This test was conducted on the fully scaled 13,504 warehouses database using 135,040 emulated terminals.

1. The current number of orders in the database was counted, giving ORDER_COUNT_BEFORE.
   1a. Consistency was verified.
2. A test was started and allowed to run at steady state for at least 5 minutes. Then the system was powered off.
3. The test was aborted on the RTE.
4. The system was powered back on and rebooted.
5. Oracle V8.1.7 Enterprise Edition for Tru64 UNIX was restarted and recovered the database from the transaction log.
6. The current number of orders in the database was counted giving ORDER_COUNT_AFTER. It was verified that ORDER_COUNT_AFTER - ORDER_COUNT_BEFORE was greater than or equal to the number of committed orders recorded by the RTE.
7. Several orders recorded by the RTE were checked in the database to make sure they existed.
8. Consistency was verified again.

5. Scaling and Database Population Related Items

5.1 Cardinalities of the Database Tables

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

The initial cardinalities of the database tables are shown in the following table.

Table Name	Cardinality
Warehouse	13,504
District	135,040
Customer	405,120,000
History	405,120,000
New-Order	121,536,000
Order	405,120,000
Order-Line	4,051,443,648
Stock	1,350,400,000
Item	100000

5.2 Distribution of Tables and Log

The distribution of tables and logs across all media must be explicitly depicted for the tested and priced systems.
The following benchmark configuration tables over all the disks of the priced system is an extension of the distribution in the tested system. System was configured with enough mirrored log disk capacity to support 8 hours of run time.

Distribution of Data on the AlphaServer GS320 Model 6/731

control_001		100%	ordl_2_0	/dev/rdisk/dsk71b	1.28%
control_001.bck2		100%	ordl_2_1	/dev/rdisk/dsk71d	1.28%
control_002	/dev/rdisk/dsk32b	100%	ordl_2_2	/dev/rdisk/dsk71e	1.28%

control_003	/dev/rdisk/dsk56b	100%	ordl_2_3	/dev/rdisk/dsk71f	1.28%
cust_0_0	/dev/rdisk/dsk7f	1.56%	ordl_2_4	/dev/rdisk/dsk74b	1.28%
cust_0_1	/dev/rdisk/dsk8f	1.56%	ordl_2_5	/dev/rdisk/dsk74d	1.28%
cust_0_10	/dev/rdisk/dsk34f	1.56%	ordl_2_6	/dev/rdisk/dsk74e	1.28%
cust_0_11	/dev/rdisk/dsk35f	1.56%	ordl_2_7	/dev/rdisk/dsk74f	1.28%
cust_0_12	/dev/rdisk/dsk38f	1.56%	ordl_2_8	/dev/rdisk/dsk70b	1.28%
cust_0_13	/dev/rdisk/dsk39f	1.56%	ordl_3_0	/dev/rdisk/dsk102b	1.28%
cust_0_14	/dev/rdisk/dsk40f	1.56%	ordl_3_1	/dev/rdisk/dsk102d	1.28%
cust_0_15	/dev/rdisk/dsk41f	1.56%	ordl_3_2	/dev/rdisk/dsk102e	1.28%
cust_0_16	/dev/rdisk/dsk56f	1.56%	ordl_3_3	/dev/rdisk/dsk102f	1.28%
cust_0_17	/dev/rdisk/dsk57f	1.56%	ordl_3_4	/dev/rdisk/dsk105b	1.28%
cust_0_18	/dev/rdisk/dsk58f	1.56%	ordl_3_5	/dev/rdisk/dsk105d	1.28%
cust_0_19	/dev/rdisk/dsk59f	1.56%	ordl_3_6	/dev/rdisk/dsk105e	1.28%
cust_0_2	/dev/rdisk/dsk9f	1.56%	ordl_3_7	/dev/rdisk/dsk105f	1.28%
cust_0_20	/dev/rdisk/dsk62f	1.56%	ordl_3_8	/dev/rdisk/dsk101b	1.28%
cust_0_21	/dev/rdisk/dsk63f	1.56%	ordl_4_0	/dev/rdisk/dsk120b	1.28%
cust_0_22	/dev/rdisk/dsk64f	1.56%	ordl_4_1	/dev/rdisk/dsk120d	1.28%
cust_0_23	/dev/rdisk/dsk65f	1.56%	ordl_4_2	/dev/rdisk/dsk120e	1.28%
cust_0_24	/dev/rdisk/dsk250f	1.56%	ordl_4_3	/dev/rdisk/dsk120f	1.28%
cust_0_25	/dev/rdisk/dsk88f	1.56%	ordl_4_4	/dev/rdisk/dsk123b	1.28%
cust_0_26	/dev/rdisk/dsk243f	1.56%	ordl_4_5	/dev/rdisk/dsk123d	1.28%
cust_0_27	/dev/rdisk/dsk90f	1.56%	ordl_4_6	/dev/rdisk/dsk123e	1.28%
cust_0_28	/dev/rdisk/dsk93f	1.56%	ordl_4_7	/dev/rdisk/dsk123f	1.28%
cust_0_29	/dev/rdisk/dsk94f	1.56%	ordl_4_8	/dev/rdisk/dsk119b	1.28%
cust_0_3	/dev/rdisk/dsk10f	1.56%	ordl_5_0	/dev/rdisk/dsk150b	1.28%
cust_0_30	/dev/rdisk/dsk95f	1.56%	ordl_5_1	/dev/rdisk/dsk150d	1.28%
cust_0_31	/dev/rdisk/dsk96f	1.56%	ordl_5_2	/dev/rdisk/dsk150e	1.28%
cust_0_32	/dev/rdisk/dsk111f	1.56%	ordl_5_3	/dev/rdisk/dsk150f	1.28%
cust_0_33	/dev/rdisk/dsk112f	1.56%	ordl_5_4	/dev/rdisk/dsk153b	1.28%
cust_0_34	/dev/rdisk/dsk113f	1.56%	ordl_5_5	/dev/rdisk/dsk153d	1.28%
cust_0_35	/dev/rdisk/dsk114f	1.56%	ordl_5_6	/dev/rdisk/dsk153e	1.28%
cust_0_36	/dev/rdisk/dsk129f	1.56%	ordl_5_7	/dev/rdisk/dsk153f	1.28%
cust_0_37	/dev/rdisk/dsk130f	1.56%	ordl_5_8	/dev/rdisk/dsk149b	1.28%
cust_0_38	/dev/rdisk/dsk266f	1.56%	ordl_6_0	/dev/rdisk/dsk174b	1.28%
cust_0_39	/dev/rdisk/dsk132f	1.56%	ordl_6_1	/dev/rdisk/dsk174d	1.28%
cust_0_4	/dev/rdisk/dsk17f	1.56%	ordl_6_2	/dev/rdisk/dsk174e	1.28%
cust_0_40	/dev/rdisk/dsk135f	1.56%	ordl_6_3	/dev/rdisk/dsk174f	1.28%
cust_0_41	/dev/rdisk/dsk136f	1.56%	ordl_6_4	/dev/rdisk/dsk177b	1.28%
cust_0_42	/dev/rdisk/dsk137f	1.56%	ordl_6_5	/dev/rdisk/dsk177d	1.28%
cust_0_43	/dev/rdisk/dsk138f	1.56%	ordl_6_6	/dev/rdisk/dsk177e	1.28%
cust_0_44	/dev/rdisk/dsk141f	1.56%	ordl_6_7	/dev/rdisk/dsk177f	1.28%
cust_0_45	/dev/rdisk/dsk142f	1.56%	ordl_6_8	/dev/rdisk/dsk173b	1.28%
cust_0_46	/dev/rdisk/dsk143f	1.56%	ordl_7_0	/dev/rdisk/dsk244b	1.28%
cust_0_47	/dev/rdisk/dsk144f	1.56%	ordl_7_1	/dev/rdisk/dsk244d	1.28%
cust_0_48	/dev/rdisk/dsk285f	1.56%	ordl_7_2	/dev/rdisk/dsk245b	1.28%
cust_0_49	/dev/rdisk/dsk160f	1.56%	ordl_7_3	/dev/rdisk/dsk245d	1.28%
cust_0_5	/dev/rdisk/dsk14f	1.56%	ordl_7_4	/dev/rdisk/dsk207b	1.28%
cust_0_50	/dev/rdisk/dsk161f	1.56%	ordl_7_5	/dev/rdisk/dsk207d	1.28%
cust_0_51	/dev/rdisk/dsk162f	1.56%	ordl_7_6	/dev/rdisk/dsk207e	1.28%
cust_0_52	/dev/rdisk/dsk165f	1.56%	ordl_7_7	/dev/rdisk/dsk207f	1.28%
cust_0_53	/dev/rdisk/dsk166f	1.56%	ordl_7_8	/dev/rdisk/dsk201b	1.28%
cust_0_54	/dev/rdisk/dsk167f	1.56%	ordr_0_0	/dev/rdisk/dsk28f	12.50%
cust_0_55	/dev/rdisk/dsk168f	1.56%	ordr_1_0	/dev/rdisk/dsk52f	12.50%
cust_0_56	/dev/rdisk/dsk246d	1.56%	ordr_2_0	/dev/rdisk/dsk265f	12.50%
cust_0_57	/dev/rdisk/dsk185f	1.56%	ordr_3_0	/dev/rdisk/dsk107f	12.50%
cust_0_58	/dev/rdisk/dsk189f	1.56%	ordr_4_0	/dev/rdisk/dsk125f	12.50%
cust_0_59	/dev/rdisk/dsk190f	1.56%	ordr_5_0	/dev/rdisk/dsk155f	12.50%
cust_0_6	/dev/rdisk/dsk15f	1.56%	ordr_6_0	/dev/rdisk/dsk179f	12.50%
cust_0_60	/dev/rdisk/dsk286f	1.56%	ordr_7_0	/dev/rdisk/dsk209f	12.50%
cust_0_61	/dev/rdisk/dsk192f	1.56%	roll_0_1	/dev/rdisk/dsk20b	3.57%
cust_0_62	/dev/rdisk/dsk193f	1.56%	roll_0_2	/dev/rdisk/dsk20d	3.57%
cust_0_63	/dev/rdisk/dsk198f	1.56%	roll_0_3	/dev/rdisk/dsk21b	3.57%
cust_0_64	/dev/rdisk/dsk263f	1.56%	roll_0_4	/dev/rdisk/dsk21d	3.57%
cust_0_7	/dev/rdisk/dsk16f	1.56%	roll_1_1	/dev/rdisk/dsk44b	3.57%
cust_0_8	/dev/rdisk/dsk32f	1.56%	roll_1_2	/dev/rdisk/dsk44d	3.57%
cust_0_9	/dev/rdisk/dsk253f	1.56%	roll_1_3	/dev/rdisk/dsk242b	3.57%
dist_0_0	/dev/rdisk/dsk28d	12.50%	roll_1_4	/dev/rdisk/dsk242d	3.57%
dist_1_0	/dev/rdisk/dsk52d	12.50%	roll_2_1	/dev/rdisk/dsk68b	3.57%
dist_2_0	/dev/rdisk/dsk265d	12.50%	roll_2_2	/dev/rdisk/dsk68d	3.57%
dist_3_0	/dev/rdisk/dsk107d	12.50%	roll_2_3	/dev/rdisk/dsk69b	3.57%

dist_4_0	/dev/rdisk/dsk125d	12.50%	roll_2_4	/dev/rdisk/dsk69d	3.57%
dist_5_0	/dev/rdisk/dsk155d	12.50%	roll_3_1	/dev/rdisk/dsk99b	3.57%
dist_6_0	/dev/rdisk/dsk179d	12.50%	roll_3_2	/dev/rdisk/dsk99d	3.57%
dist_7_0	/dev/rdisk/dsk209d	12.50%	roll_3_3	/dev/rdisk/dsk252b	3.57%
hist_0_0	/dev/rvol/vol_a2	12.50%	roll_3_4	/dev/rdisk/dsk252d	3.57%
hist_1_0	/dev/rvol/vol_a6	12.50%	roll_4_1	/dev/rdisk/dsk117b	3.57%
hist_2_0	/dev/rvol/vol_a10	12.50%	roll_4_2	/dev/rdisk/dsk117d	3.57%
hist_3_0	/dev/rvol/vol_a13	12.50%	roll_4_3	/dev/rdisk/dsk118b	3.57%
hist_4_0	/dev/rvol/vol_a17	12.50%	roll_4_4	/dev/rdisk/dsk118d	3.57%
hist_5_0	/dev/rvol/vol_a21	12.50%	roll_5_1	/dev/rdisk/dsk147b	3.57%
hist_6_0	/dev/rvol/vol_a25	12.50%	roll_5_2	/dev/rdisk/dsk147d	3.57%
hist_7_0	/dev/rvol/vol_a29	12.50%	roll_5_3	/dev/rdisk/dsk148b	3.57%
icust1_0_0	/dev/rdisk/dsk22e	6.25%	roll_5_4	/dev/rdisk/dsk148d	3.57%
icust1_0_1	/dev/rdisk/dsk46e	6.25%	roll_6_1	/dev/rdisk/dsk171b	3.57%
icust1_0_2	/dev/rdisk/dsk70e	6.25%	roll_6_2	/dev/rdisk/dsk171d	3.57%
icust1_0_3	/dev/rdisk/dsk101e	6.25%	roll_6_3	/dev/rdisk/dsk172b	3.57%
icust1_0_4	/dev/rdisk/dsk119e	6.25%	roll_6_4	/dev/rdisk/dsk172d	3.57%
icust1_0_5	/dev/rdisk/dsk149e	6.25%	roll_7_1	/dev/rdisk/dsk263b	3.57%
icust1_0_6	/dev/rdisk/dsk173e	6.25%	roll_7_2	/dev/rdisk/dsk263d	3.57%
icust1_0_7	/dev/rdisk/dsk201e	6.25%	roll_7_3	/dev/rdisk/dsk200b	3.57%
icust2_0_0	/dev/rdisk/dsk22d	6.25%	roll_7_4	/dev/rdisk/dsk200d	3.57%
icust2_0_1	/dev/rdisk/dsk46d	6.25%	stok_0_0	/dev/rdisk/dsk7e	1.25%
icust2_0_2	/dev/rdisk/dsk70d	6.25%	stok_0_1	/dev/rdisk/dsk8e	1.25%
icust2_0_3	/dev/rdisk/dsk101d	6.25%	stok_0_10	/dev/rdisk/dsk32e	1.25%
icust2_0_4	/dev/rdisk/dsk119d	6.25%	stok_0_11	/dev/rdisk/dsk253e	1.25%
icust2_0_5	/dev/rdisk/dsk149d	6.25%	stok_0_12	/dev/rdisk/dsk34e	1.25%
icust2_0_6	/dev/rdisk/dsk173d	6.25%	stok_0_13	/dev/rdisk/dsk35e	1.25%
icust2_0_7	/dev/rdisk/dsk201d	6.25%	stok_0_14	/dev/rdisk/dsk38e	1.25%
idist_0_0	/dev/rdisk/dsk29d	12.50%	stok_0_15	/dev/rdisk/dsk39e	1.25%
idist_1_0	/dev/rdisk/dsk53d	12.50%	stok_0_16	/dev/rdisk/dsk40e	1.25%
idist_2_0	/dev/rdisk/dsk77d	12.50%	stok_0_17	/dev/rdisk/dsk41e	1.25%
idist_3_0	/dev/rdisk/dsk108d	12.50%	stok_0_18	/dev/rdisk/dsk44e	1.25%
idist_4_0	/dev/rdisk/dsk126d	12.50%	stok_0_19	/dev/rdisk/dsk242e	1.25%
idist_5_0	/dev/rdisk/dsk156d	12.50%	stok_0_2	/dev/rdisk/dsk9e	1.25%
idist_6_0	/dev/rdisk/dsk180d	12.50%	stok_0_20	/dev/rdisk/dsk56e	1.25%
idist_7_0	/dev/rdisk/dsk214d	12.50%	stok_0_21	/dev/rdisk/dsk57e	1.25%
iitem_0_0	/dev/rdisk/dsk22f	100%	stok_0_22	/dev/rdisk/dsk58e	1.25%
iordr1_0_0	/dev/rvol/vol_a3	6.25%	stok_0_23	/dev/rdisk/dsk59e	1.25%
iordr1_1_0	/dev/rvol/vol_a7	6.25%	stok_0_24	/dev/rdisk/dsk62e	1.25%
iordr1_2_0	/dev/rvol/vol_a11	6.25%	stok_0_25	/dev/rdisk/dsk63e	1.25%
iordr1_3_0	/dev/rvol/vol_a14	6.25%	stok_0_26	/dev/rdisk/dsk64e	1.25%
iordr1_4_0	/dev/rvol/vol_a18	6.25%	stok_0_27	/dev/rdisk/dsk65e	1.25%
iordr1_5_0	/dev/rvol/vol_a22	6.25%	stok_0_28	/dev/rdisk/dsk68e	1.25%
iordr1_6_0	/dev/rvol/vol_a26	6.25%	stok_0_29	/dev/rdisk/dsk69e	1.25%
iordr1_7_0	/dev/rvol/vol_a30	6.25%	stok_0_3	/dev/rdisk/dsk10e	1.25%
iordr2_0_0	/dev/rvol/vol_a4	6.25%	stok_0_30	/dev/rdisk/dsk250e	1.25%
iordr2_1_0	/dev/rvol/vol_a8	6.25%	stok_0_31	/dev/rdisk/dsk88e	1.25%
iordr2_2_0	/dev/rvol/vol_a12	6.25%	stok_0_32	/dev/rdisk/dsk243e	1.25%
iordr2_3_0	/dev/rvol/vol_a15	6.25%	stok_0_33	/dev/rdisk/dsk90e	1.25%
iordr2_4_0	/dev/rvol/vol_a19	6.25%	stok_0_34	/dev/rdisk/dsk93e	1.25%
iordr2_5_0	/dev/rvol/vol_a23	6.25%	stok_0_35	/dev/rdisk/dsk94e	1.25%
iordr2_6_0	/dev/rvol/vol_a27	6.25%	stok_0_36	/dev/rdisk/dsk95e	1.25%
iordr2_7_0	/dev/rvol/vol_a31	6.25%	stok_0_37	/dev/rdisk/dsk96e	1.25%
istok_0_0	/dev/rdisk/dsk29f	12.50%	stok_0_38	/dev/rdisk/dsk99e	1.25%
istok_0_1	/dev/rdisk/dsk53f	12.50%	stok_0_39	/dev/rdisk/dsk252e	1.25%
istok_0_2	/dev/rdisk/dsk77f	12.50%	stok_0_4	/dev/rdisk/dsk17e	1.25%
istok_0_3	/dev/rdisk/dsk108f	12.50%	stok_0_40	/dev/rdisk/dsk111e	1.25%
istok_0_4	/dev/rdisk/dsk126f	12.50%	stok_0_41	/dev/rdisk/dsk112e	1.25%
istok_0_5	/dev/rdisk/dsk156f	12.50%	stok_0_42	/dev/rdisk/dsk113e	1.25%
istok_0_6	/dev/rdisk/dsk180f	12.50%	stok_0_43	/dev/rdisk/dsk114e	1.25%
istok_0_7	/dev/rdisk/dsk214f	12.50%	stok_0_44	/dev/rdisk/dsk129e	1.25%
item_0_0	/dev/rdisk/dsk21f	100%	stok_0_45	/dev/rdisk/dsk130e	1.25%
iware_0_0	/dev/rdisk/dsk29e	12.50%	stok_0_46	/dev/rdisk/dsk266e	1.25%
iware_1_0	/dev/rdisk/dsk53e	12.50%	stok_0_47	/dev/rdisk/dsk132e	1.25%
iware_2_0	/dev/rdisk/dsk77e	12.50%	stok_0_48	/dev/rdisk/dsk117e	1.25%
iware_3_0	/dev/rdisk/dsk108e	12.50%	stok_0_49	/dev/rdisk/dsk118e	1.25%
iware_4_0	/dev/rdisk/dsk126e	12.50%	stok_0_5	/dev/rdisk/dsk14e	1.25%
iware_5_0	/dev/rdisk/dsk156e	12.50%	stok_0_50	/dev/rdisk/dsk135e	1.25%
iware_6_0	/dev/rdisk/dsk180e	12.50%	stok_0_51	/dev/rdisk/dsk136e	1.25%
iware_7_0	/dev/rdisk/dsk214e	12.50%	stok_0_52	/dev/rdisk/dsk137e	1.25%

log0_1	/dev/rdisk/dsk267g	6.25%	stok_0_53	/dev/rdisk/dsk138e	1.25%
log0_2	/dev/rdisk/dsk267h	6.25%	stok_0_54	/dev/rdisk/dsk141e	1.25%
log0_3	/dev/rdisk/dsk90h	6.25%	stok_0_55	/dev/rdisk/dsk142e	1.25%
log0_4	/dev/rdisk/dsk93g	6.25%	stok_0_56	/dev/rdisk/dsk143e	1.25%
log1_1	/dev/rdisk/dsk268g	6.25%	stok_0_57	/dev/rdisk/dsk144e	1.25%
log1_2	/dev/rdisk/dsk268h	6.25%	stok_0_58	/dev/rdisk/dsk147e	1.25%
log2_1	/dev/rdisk/dsk269g	6.25%	stok_0_59	/dev/rdisk/dsk148e	1.25%
log2_2	/dev/rdisk/dsk269h	6.25%	stok_0_6	/dev/rdisk/dsk15e	1.25%
log3_1	/dev/rdisk/dsk270g	6.25%	stok_0_60	/dev/rdisk/dsk285e	1.25%
log3_2	/dev/rdisk/dsk270h	6.25%	stok_0_61	/dev/rdisk/dsk160e	1.25%
log4_1	/dev/rdisk/dsk271g	6.25%	stok_0_62	/dev/rdisk/dsk161e	1.25%
log4_2	/dev/rdisk/dsk271h	6.25%	stok_0_63	/dev/rdisk/dsk162e	1.25%
log5_1	/dev/rdisk/dsk281g	6.25%	stok_0_64	/dev/rdisk/dsk165e	1.25%
log5_2	/dev/rdisk/dsk281h	6.25%	stok_0_65	/dev/rdisk/dsk166e	1.25%
log6_1	/dev/rdisk/dsk273g	6.25%	stok_0_66	/dev/rdisk/dsk167e	1.25%
log6_2	/dev/rdisk/dsk273h	6.25%	stok_0_67	/dev/rdisk/dsk168e	1.25%
log7_1	/dev/rdisk/dsk287g	6.25%	stok_0_68	/dev/rdisk/dsk171e	1.25%
log7_2	/dev/rdisk/dsk287h	6.25%	stok_0_69	/dev/rdisk/dsk172e	1.25%
nord_0_0	/dev/rvol/vol_a1	12.50%	stok_0_7	/dev/rdisk/dsk16e	1.25%
nord_1_0	/dev/rvol/vol_a5	12.50%	stok_0_70	/dev/rdisk/dsk246b	1.25%
nord_2_0	/dev/rvol/vol_a9	12.50%	stok_0_71	/dev/rdisk/dsk185e	1.25%
nord_3_0	/dev/rvol/vol_a32	12.50%	stok_0_72	/dev/rdisk/dsk189e	1.25%
nord_4_0	/dev/rvol/vol_a16	12.50%	stok_0_73	/dev/rdisk/dsk190e	1.25%
nord_5_0	/dev/rvol/vol_a20	12.50%	stok_0_74	/dev/rdisk/dsk286e	1.25%
nord_6_0	/dev/rvol/vol_a24	12.50%	stok_0_75	/dev/rdisk/dsk192e	1.25%
nord_7_0	/dev/rvol/vol_a28	12.50%	stok_0_76	/dev/rdisk/dsk193e	1.25%
ordl_0_0	/dev/rdisk/dsk23b	1.28%	stok_0_77	/dev/rdisk/dsk198e	1.25%
ordl_0_1	/dev/rdisk/dsk23d	1.28%	stok_0_78	/dev/rdisk/dsk247b	1.25%
ordl_0_2	/dev/rdisk/dsk23e	1.28%	stok_0_79	/dev/rdisk/dsk200e	1.25%
ordl_0_3	/dev/rdisk/dsk23f	1.28%	stok_0_8	/dev/rdisk/dsk20e	1.25%
ordl_0_4	/dev/rdisk/dsk26b	1.28%	stok_0_80	/dev/rdisk/dsk200f	1.25%
ordl_0_5	/dev/rdisk/dsk26d	1.28%	stok_0_9	/dev/rdisk/dsk21e	1.25%
ordl_0_6	/dev/rdisk/dsk26e	1.28%	system_001	/dev/rdisk/dsk41b	100%
ordl_0_7	/dev/rdisk/dsk26f	1.28%	temp_perm	/dev/rdisk/dsk65b	33.30%
ordl_0_8	/dev/rdisk/dsk22b	1.28%	temp_perm_2	/dev/rdisk/dsk65d	33.30%
ordl_1_0	/dev/rdisk/dsk264b	1.28%	temp_perm_3	/dev/rdisk/dsk65g	33.30%
ordl_1_1	/dev/rdisk/dsk264d	1.28%	ware_0_0	/dev/rdisk/dsk28b	12.50%
ordl_1_2	/dev/rdisk/dsk264e	1.28%	ware_1_0	/dev/rdisk/dsk52b	12.50%
ordl_1_3	/dev/rdisk/dsk264f	1.28%	ware_2_0	/dev/rdisk/dsk265b	12.50%
ordl_1_4	/dev/rdisk/dsk50b	1.28%	ware_3_0	/dev/rdisk/dsk107b	12.50%
ordl_1_5	/dev/rdisk/dsk50d	1.28%	ware_4_0	/dev/rdisk/dsk125b	12.50%
ordl_1_6	/dev/rdisk/dsk50e	1.28%	ware_5_0	/dev/rdisk/dsk155b	12.50%
ordl_1_7	/dev/rdisk/dsk50f	1.28%	ware_6_0	/dev/rdisk/dsk179b	12.50%
ordl_1_8	/dev/rdisk/dsk46b	1.28%	ware_7_0	/dev/rdisk/dsk209b	12.50%

5.3 Type of Database

A statement must be provided that describes:

The data model implemented by the DBMS used (e.g., relational, network, hierarchical)

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

The database used for this testing was Oracle V8.1.7 Enterprise Edition for Tru64 UNIX from Oracle Corporation. Oracle V8.1.7 Enterprise Edition for Tru64 UNIX is a relational DBMS. SQL stored procedures were used and were invoked from the clients using OCI calls.

5.4 Database Partitions/Replications

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

Horizontal partitioning was used on the warehouse, district, new_order, history, order, and orderline tables The functionality for this was provided by Oracle V8.1.7 Enterprise Edition for Tru64 UNIX. For further details of the partitioning of the database, see Appendix B.

5.5 180-Days Space Requirement

The calculations for arriving at the 180-day space computations, as defined in Clause 4.2.3 must be disclosed.

TPM 155179.258
Warehouses 13504

SEGMENT	TYPE	TSPACE	BLOCKS	FIVE_PCT	DAILY_GROW	TOTAL
CUSTCLUSTER	CLUSTER	CUST	202560012	10128001	0	212688013
DIST	TABLE PARTITION	DIST	139264	6963	0	146227
HIST	TABLE PARTITION	HIST	12390400	0	2278120	12390576
ICUST1	INDEX	ICUST1	5017600	250880	0	5268480
ICUST2	INDEX	ICUST2	11212800	560640	0	11773440
IDIST	INDEX PARTITION	IDIST	4096	205	0	4301
IITEM	INDEX	ITEMS	1068	53	0	1121
INORD	INDEX PARTITION	INORD	1075200	53760	0	1128960
IORDL	INDEX PARTITION	IORDL	162201600	0	29822659	162203906
IORDR1	INDEX PARTITION	IORDR1	5017600	250880	0	5268480
IORDR2	INDEX PARTITION	IORDR2	8294400	414720	0	8709120
ISTOK	INDEX	ISTOK	14796800	734720	0	15429120
ITEMCLUSTER	CLUSTER	ITEMS	7168	358	0	7526
IWARE	INDEX PARTITION	IWARE	3464	173	0	3637
ORDR	TABLE PARTITION	ORDR	8704000	0	1600332	8704124
STOKCLUSTER	CLUSTER	STOK	270924010	13546201	0	284470211
WARE	TABLE PARTITION	WARE	16384	819		17203
SYSTEM	SYS	SYSTEM	1024000	0	0	1024000
TOTAL			703389866	25948373	33701111	729238445

Dynamic space	183,296,000	
Static space	546,042,239	
Free space	-99,794	
Daily growth	33,701,111	
Daily spread	0	Oracle may be configured such that daily spread is 0
180-day space (blk.)	6,612,242,219	
Block size (bytes)	2,048	
180-day (GB)	12,611.85	
Log block size	1,024	
Log blocks/N_O txn.	13.50	Log blocks used per New-Order transactions
8-hour log (GB)	958.98	

Total Space (GB) Required on System (except for logs)

180 day (GB)	12,611.85
OS + Oracle	50.01
Swap	46.39
Total	12,708.25

Space Allocated for Data (Excluding logs)

Drive Type	Number	Cap. (GB)	Total Capacity (GB)	Space Needed	Disks Needed	
RZ1DA-VW	12	8.48	101.70		0	Data Disks
RZ1EA-VW	738	16.96	12,515.56		0	Data Disks
Total			12,617.26	-5.41	0	

Space Allocated for OS and Swap						
RZ1DA-VW	8	8.48	67.80		0	OS
RZ1EA-VW	2	16.96	33.92		0	Swap Disks
Total			101.72	0.00	0	OS + Swap Disks
Space Allocated for Logs						
Drive Type	Number	Cap. (GB)	Capacity (GB)			
RZ1EA-VW	57	16.96	966.65	0.00	0	Mirrored Log Disks
	114	16.96	1933.44		0	UnMirrored Log Disks
Total Priced devices	**RZ1EA-VW**	**RZ1DA-VW**				
For Configuration	854	20		mirrored logs + space allocated+OS and Swap		
Spares	86	2		10% extra-minimum of 2 drives		
	940	22				

6. Performance Metrics and Response Time Related Items

6.1 Reporting All Data

Measured tpmC must be reported
All the data required by Clause 5 is reported below in Section 5.2 through 5.10. The measured tpmC for the Compaq AlphaServer GS320 C/S configuration was 144,331.95 tpmC.

6.2 Response Times

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

Response Times in seconds

Response Times in seconds

Transaction	90th percentile	Average	Maximum
New-Order	1.826	1.050	9.952
Payment	1.780	1.037	9.567
Order-Status	1.676	0.947	8.727
Delivery (interactive)	1.190	0.625	9.176
Delivery (deferred)	1.141	0.611	7.907
Stock-Level	1.989	1.223	9.755
Menu	1.202	0.607	9.234

6.3 Think and Keying Times

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

Keying/Think Times in seconds

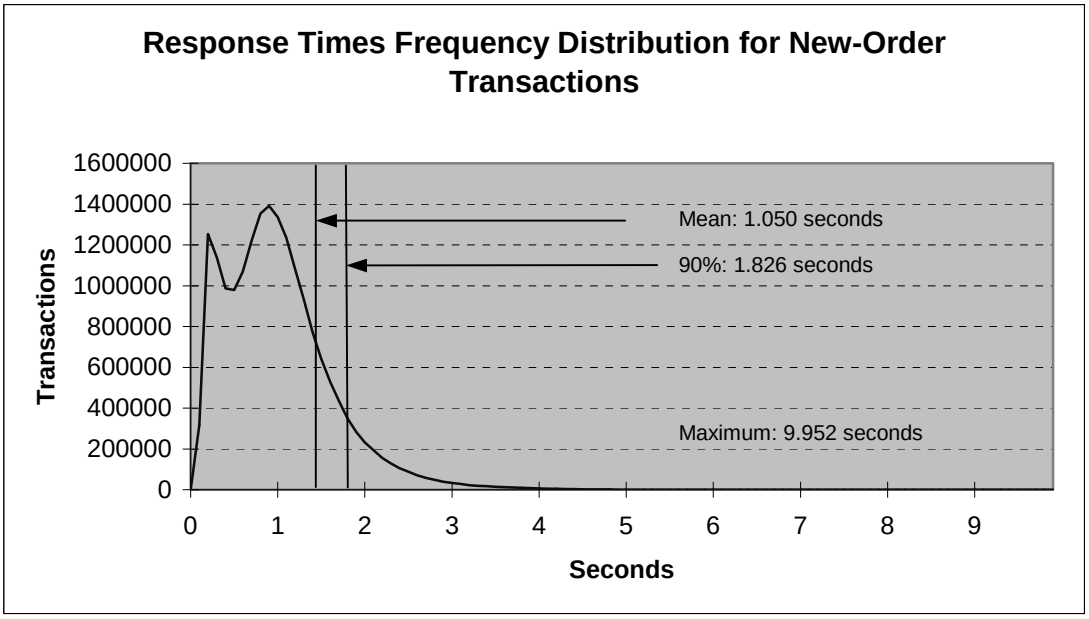
Keying/Think Times in seconds

Transaction	Minimum	Average	Maximum
New-Order	18.000/0.000	18.001/13.904	18.192/140.005
Payment	3.000/0.000	3.000/12.004	3.172/120.007
Order-Status	2.000/0.000	2.001/10.003	2.163/ 99.977
Delivery (interactive)	2.000/0.000	2.001/ 5.002	2.163/ 49.963
Stock-Level	2.000/0.000	2.001/ 5.003	2.163/ 49.920

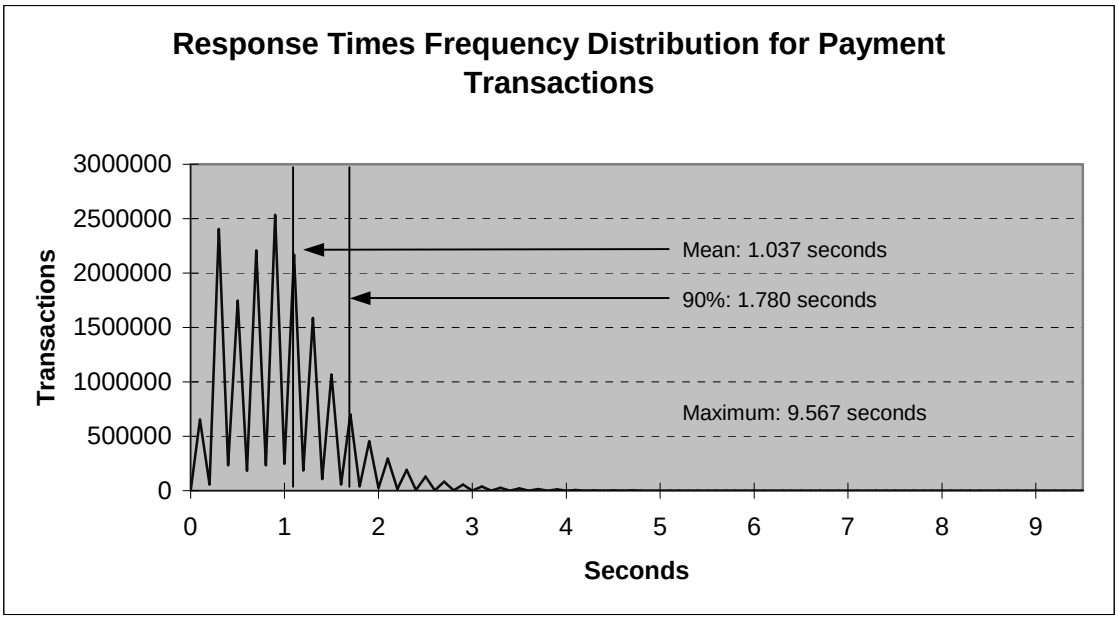
6.4 Response Times Frequency Distribution

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

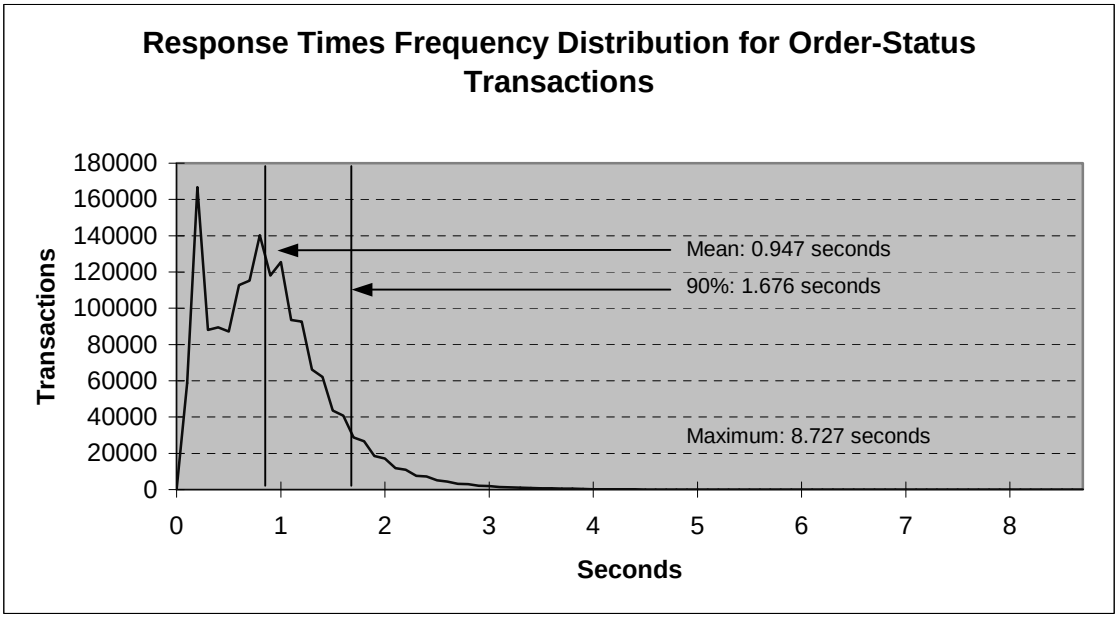
Response Times Frequency Distribution for New-Order Transactions



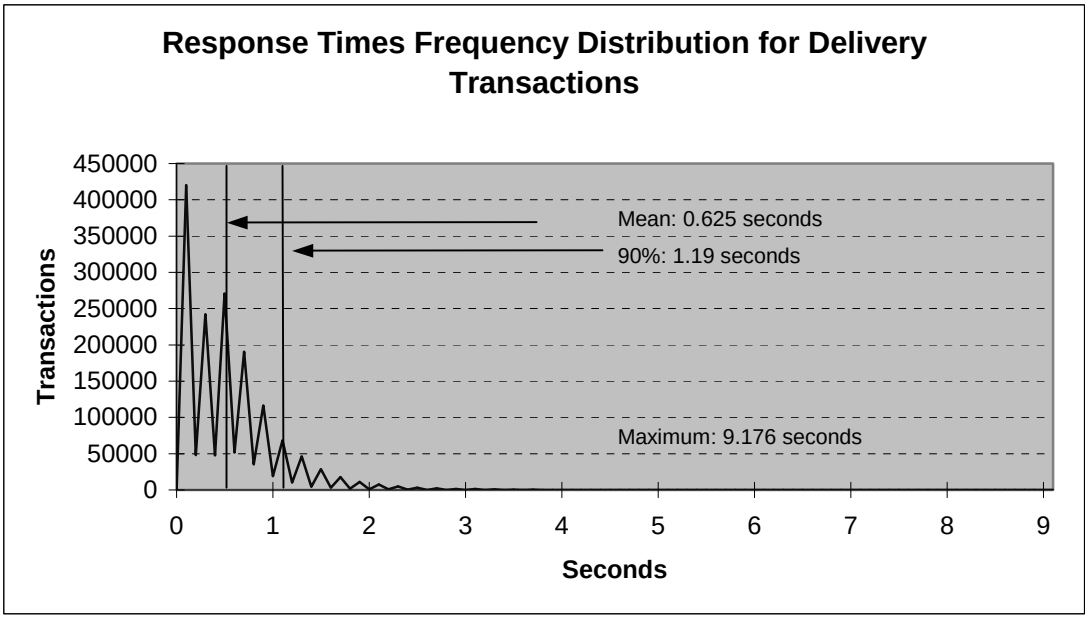
Response Times Frequency Distribution for Payment Transactions



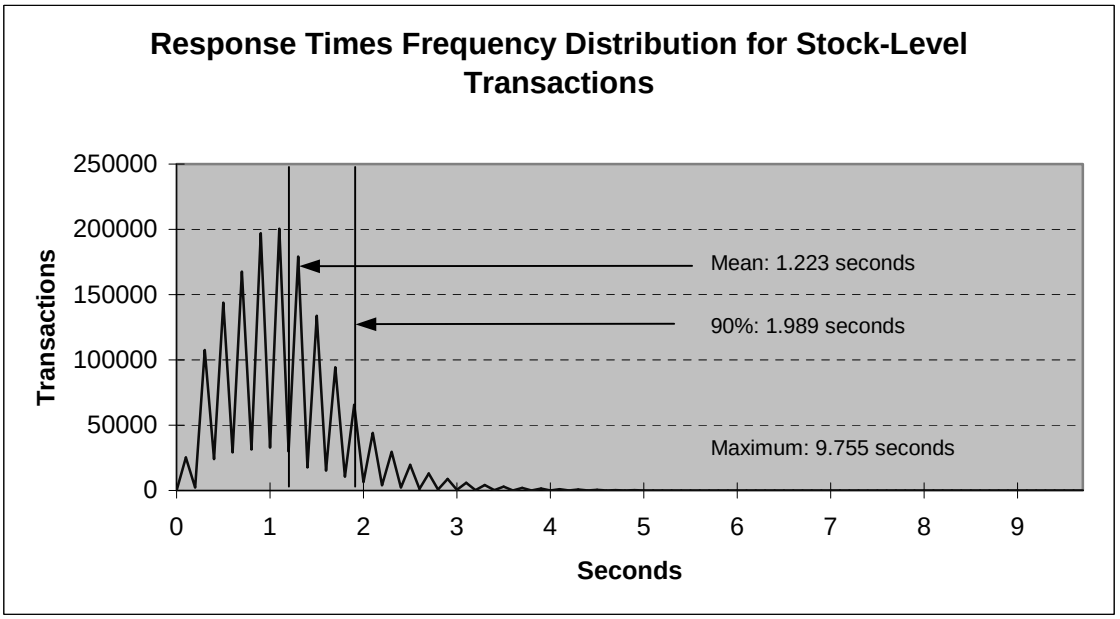
Response Times Frequency Distribution for Order-Status Transactions



Response Times Frequency Distribution for Delivery Transactions



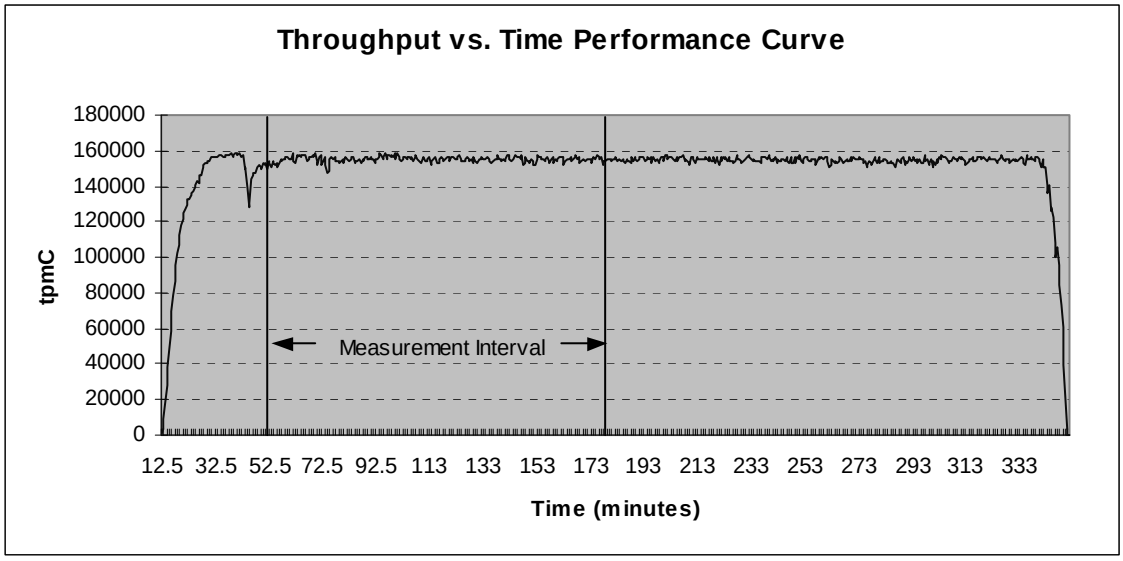
Response Times Frequency Distribution for Stock-Level Transactions



6.5 Response Time versus Throughput Performance Curve

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

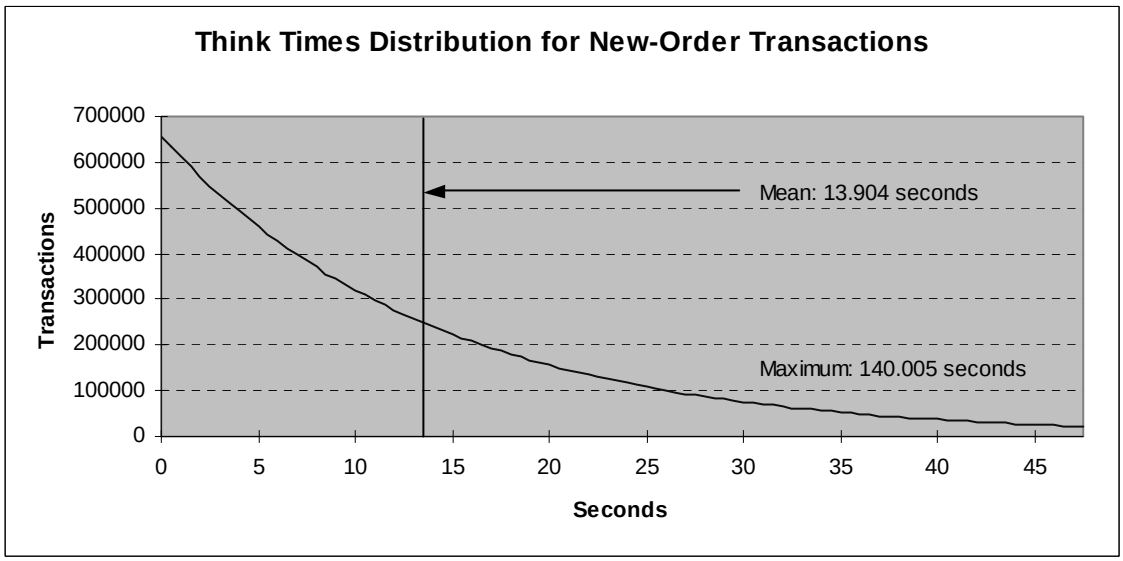
New-Order Response Time versus Throughput



6.6 Think Times Frequency Distribution

Think times frequency distribution curves (see Clause 5.6.3) must be reported for the New Order Transaction.

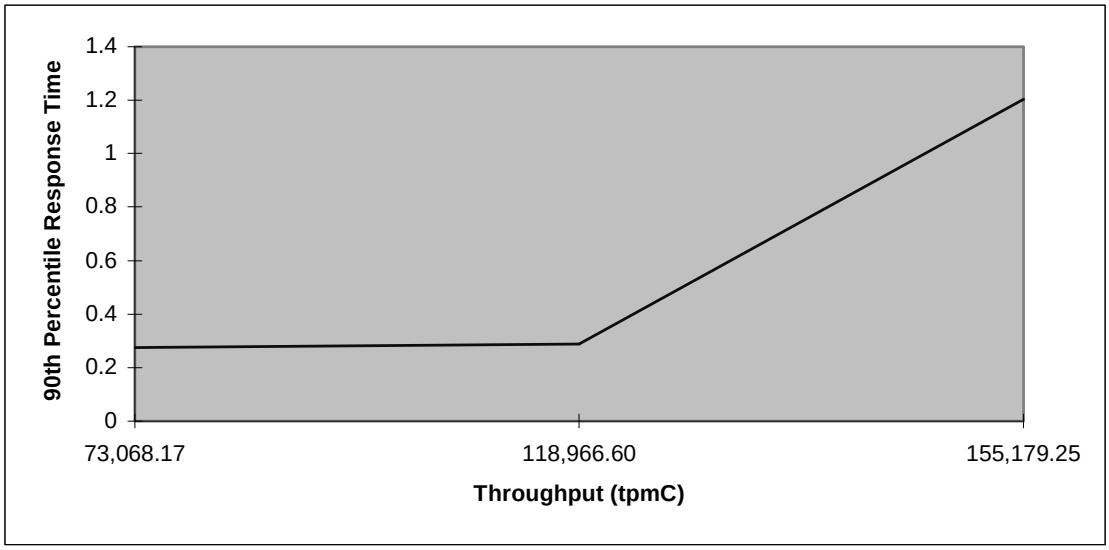
Think Times Distribution for New-Order Transactions



6.7 New-Order Throughput vs. Elapsed Time

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

Throughput vs. Time Performance Curve



6.8 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.

Confirmation that the SUT has reached steady state prior to the beginning of the data collection measurement interval is based on a visual inspection of the plots of tpmC versus time.

The graph in Section 6.7 plots the average tpmC versus time, averaged over 30 second intervals, and shows that steady state was reached before the data was collected. The ramp-up and steady-state stages are clearly visible.

6.8.1 Transaction Flow

For each of the TPC Benchmark C transaction types, the following steps are executed.

Compaq DB Web Connector RT V1.1 for Windows 2000 server was used as the transaction manager (TM). Each transaction was divided into two pieces. The front-end portion handled all screen I/O, while the back-end portion handled all database operations. Both front-end and back-end pieces ran on the client system.

The front-end portion communicates with the back-end portion through shared memory. The back-end portion communicates with the server system over gigaswitch using Oracle V8.1.7 Enterprise Edition for Tru64 UNIX OCI calls. Compaq DB Web Connector routes the transaction and balances the load according to values in the registry governing the number of processing procedure servers. Following is the transaction flow description.

- Each TPC-C user invokes the TPC-C main front-end portion.
- The front-end portion displays the TPC-C transaction menu on the user terminal.

- The TPC-C user chooses the transaction type and proceeds to fill the screen fields required for that transaction.
- The TPC-C front-end accepts all values entered by the user and transmits those values to one of the TPC-C back-end processing procedures. Each of the back-end processing procedures can process any of the transaction types.
- A TPC-C back-end processing procedure receives a workspace and proceeds to execute all database operations related to the transaction type specified. All the information entered on the user terminal is contained in the Compaq DB Web Connector workspace.
- Once the transaction is committed, the TPC-C back-end processing procedure loads the workspace with the transaction output and returns control to the Compaq DB Web Connector transaction manager.
- Compaq DB Web Connector passes the workspace back to the TPC-C main front-end.

6.8.2 Database Transaction

All database operations are performed by the TPC-C back-end programs. The process is described next.

- Using Oracle V8.1.7 Enterprise Edition for Tru64 UNIX OCI calls, the TPC-C back-end program interacts with Oracle V8.1.7 Enterprise Edition for Tru64 UNIX to perform SQL data manipulations such as update, select, delete, and insert, as required by the transaction. After all database operations are performed for a transaction, the transaction is committed.
- Oracle V8.1.7 Enterprise Edition for Tru64 UNIX proceeds to update the database as follows:

When Oracle V8.1.7 Enterprise Edition for Tru64 UNIX changes a database table with an update, insert, or delete operation, the change is initially made in memory, not on disk. When there is not enough space in the memory buffer to read in or write additional data pages, Oracle will make space by flushing some modified pages to disk. Modified pages are also written to disk when a checkpoint occurs. Before a change is made to the database, it is first recorded in the transaction log. This ensures that the database can be recovered completely in the event of a failure. Using the transaction log, transactions that started but did not complete prior to a failure can be undone, and transactions recorded as complete in the transaction log but not yet written to disk can be redone. For each of the TPC Benchmark C transaction types, the following steps are executed.

Compaq DB Web Connector RT V1.1 for Windows 2000 server was used as the transaction manager (TM). Each transaction was divided into two pieces. The front-end portion handled all screen I/O, while the back-end portion handled all database operations. Both front-end and back-end pieces ran on the client system.

The front-end portion communicates with the back-end portion through shared memory. The back-end portion communicates with the server system over gigaswitch using Oracle V8.1.7 Enterprise Edition for Tru64 UNIX OCI calls. Compaq DB Web Connector routes the transaction and balances the load according to values in the registry governing the number of processing procedure servers. The transaction flow is described below.

- Each TPC-C user invokes the TPC-C main front-end portion.
- The front-end portion displays the TPC-C transaction menu on the user terminal.
- The TPC-C user chooses the transaction type and proceeds to fill the screen fields required for that transaction.
- The TPC-C front-end accepts all values entered by the user and transmits those values to one of the TPC-C back-end processing procedures. Each of the back-end processing procedures can process any of the transaction types.
- A TPC-C back-end processing procedure receives a workspace and proceeds to execute all database operations related to the transaction type specified. All the information entered on the user terminal is contained in the Compaq DB Web Connector workspace.
- Once the transaction is committed, the TPC-C back-end processing procedure loads the workspace with the transaction output and returns control to the Compaq DB Web Connector transaction manager.
- Compaq DB Web Connector passes the workspace back to the TPC-C main front-end.

6.9 Work Performed During Steady State

A description of how the work normally performed during a sustained test actually occurred during the measurement interval must be reported.

For each of the TPC Benchmark C transaction types, the following steps are executed.

Each emulated user starts an Internet browser and asks to attach to the application on the desired client. The application formats the menus, input forms and data output using HTML (HyperText Markup Language). The HTML strings are transmitted over TCP/IP back to the client, where they can be displayed by any Web Browser software. The application on the client is run under the control of the Internet Information Server.

Transactions are submitted by the RTE in accordance with the rules of the TPC-C benchmark. The emulated user chooses a transaction from the menu. The RTE records the time it takes from selecting the menu item to receiving the requested form. Data is generated for input to the form, then the user waits the specified keying time. The submit is sent and the RTE records the time it takes for the transaction to be processed and all the output data to be returned. The user then waits for the randomly generated think time before starting the process over again. All timings taken by the RTE generate a start and end timestamp. Keying and think times are calculated as the difference between end-time of a timing to the start of the next.

The database records transactions in the database tables and also the transaction log. Writes to the database may stay in Oracle's in-memory data cache for a while before being written to disk. Checkpoints are initiated once the log files were filled and allowed to roll over.

6.10 Determining Reproducibility

A description of the method used to determine the reproducibility of the measurement results must be reported.
The experiment at the maximum targeted tpmC level was repeated once to ensure reproducibility. The computed tpmC for each experiment was within .0044% of the reported tpmC, and all 90th percentile response times were under their respective limits.

6.11 Duration of Measurement Period

A statement of the duration of the measurement period for the reported maximum qualified throughput (tpmC) must be included.

Each experiment was run for a minimum of 190 minutes. The data collection period was 120 minutes and started approximately 60 minutes after all simulated users had begun executing transactions with a 10 minute ramp down.

6.12 Method of Regulation of the 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 weighted distribution method was used. The RTE adjusted the transaction mix dynamically in accordance with clause 5.4.2.1.

6.13 Percentage of the Total Mix

The percentage of the total mix for each transaction type must be disclosed.
See Section 3.10.

6.14 Percentage of New-Order Transactions Rolled Back

The percentage of New-Order transactions rolled back as a result of invalid item numbers must be disclosed.
See Section 3.10.

6.15 Average Number of Order-Lines

The average number of order-lines entered per New-Order transaction must be disclosed.
See Section 3.10.

6.16 Percentage of Remote Order-Lines

The percentage of remote order-lines entered per New-order transaction must be disclosed.
See Section 3.10.

6.17 Percentage of Remote Payment Transactions

The percentage of remote Payment transactions must be disclosed.
See Section 3.10.

6.18 Percentage of Customer Selections

The percentage of customer selections by customer last name in the Payment and Order-Status transactions must be disclosed.
See Section 3.10.

6.19 Percentage of Delivery Transactions

The percentage of Delivery transactions skipped due to there being fewer than necessary orders in the New-Order table must be disclosed.
See Section 3.10.

6.20 Number of Checkpoints

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

A checkpoint is the process of writing all modified data pages to disk. The TPC-C benchmark on the Compaq AlphaServer GS320 Model 6/731 C/S system was set up to checkpoint within every 30 minutes. One checkpoint occurs during the warm-up period, and 4 checkpoints occurred during the measurement period.

7. SUT, Driver, and Communication Definition Related Items

7.1 Description of RTE

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

In order to simulate terminal users and record response times, a proprietary emulation package, PRTE, was used to simulate terminal users, generate random data, and record response times. This package runs on a processor that is distinct from the system under test.

7.2 Driver Functionality and Performance

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

Due to the large number of PCs and associated hardware that would be required to run these tests, Remote Terminal Emulator was used to emulate the connected PCs and LAN.

As configured for this test, the driver software emulates the traffic that would be observed from the users' PCs connected by Ethernet to the front-end clients using HTTP (HyperText Transfer Protocol) over TCP/IP.

7.3 Functional Diagrams and Details of Driver System

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

The diagrams in Section 1.7 show the tested and priced benchmark configurations.

7.4 Network Configurations and Driver System

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

Section 1.7 in this report has a picture of the network configurations of both the tested service and the proposed (target) services.

In the measured configuration, both the client machines (Compaq ProLiant ML370 and AlphaServer GS320) are connected into one Gigaswitch. The emulated PC's are connected on 10/100 Ethernet LANs. The total number of PCs are 135,040.

7.5 Network Bandwidth

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

The Ethernet used in the local area network (LAN) between the emulated terminals and the front-end systems complies with the IEEE 802.3 standard and has a bandwidth of 10 megabits per second (Mbps).

The network between the front-end clients and the server has a bandwidth of 2GB/S. The network connecting the PCs into the gigaswitch is 10/100 Mbps.

7.6 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.

8. Pricing Related Items

8.1 Hardware and Software Components

A detailed list of hardware and software used in the priced system. Each 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, contents of the package must be disclosed.

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

The detailed list of all hardware and software for the priced configuration is listed in the system pricing summary.

8.1.1 Hardware Pricing

Compaq Computer Corporation's TPC Benchmark C tests used packaged hardware systems whenever possible to simplify configurations to the fewest number of line items.

All hardware and software products have been priced to satisfy the 5 year warranty and 5 x 8 with 4 hour response requirements.

The hardware prices for the Compaq AlphaServer GS320 6/731 32 CPUs 128GB memory and their associated components (e.g., memory, storage subsystem, etc.) are based on IC System Solutions, Data Comm Warehouse and Compaq Computer Corporation price quotations.

8.1.2 Software Pricing

The priced system uses the following software products:

- Compaq Tru64 UNIX operating system
- Compaq TruCluster
- Compaq DB Web Connector
- Compaq C compiler
- Oracle V8.1.7 Enterprise Edition for Tru64 UNIX with Parallel Server and Partitioning Options
- Microsoft 2000 Server Operating System

The license purchase includes 1 year of warranty service. Four years of additional software warranty is provided for a total of 5 years extended warranty. The software warranty and service levels are the same as the service level for the hardware system on which the software operates.

The level of post-warranty software service is Layered Product Support (LPS).

8.1.3 Warranty Pricing

In addition to the base warranty, additional warranty has been priced to satisfy the 5-year TPC-C warranty requirements of all products.

8.1.4 Price Discounts

See Appendix F.

8.2 Availability Status

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

The Compaq AlphaServer GS320 Model 6/731 and all associated hardware used in the measurement as well as Compaq's Tru64 UNIX 5.1 are currently available. Oracle V8.1.7 Enterprise Edition for Tru64 UNIX will be available February 2, 2001.

8.3 Performance and Price/Performance

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

The following table shows the measured tpmC and price/tpmC results for the tested systems:

CPU Model	Configuration	Software	tpmC	Price per tpmC $tpmC
Compaq AlphaServer GS160 6/731	32 CPU	Compaq Tru64 UNIX v5.1 Oracle V8.1.7 Enterprise Edition for Tru64 UNIX	155,179	$55.68/tpmC

8.4 Country Specific Pricing

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

None.

8.5 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.*

None.

9. Audit Related Items

9.1 Audit

If the bench benchmark has been independently audited, the auditor's name, address, phone number, and a brief audit summary report indicating compliance must be included in the Full Disclosure Report. A statement should be included, specifying when the complete audit report will become available and who to contact in order to obtain a copy.

Appendix E contains the complete independent auditor's letter by Francois Raab of InfoSizing for the test described in this report.

Appendix A

crestdl.c

```
/*+*****************************************************************************
 *                                                                           *
 *  COPYRIGHT (c) 1997, 2000 BY                                              *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                                     *
 *                                                                           *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                             *
 *                                                                           *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND SHOULD NOT BE CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                             *
 *                                                                           *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                           *
 *                                                                           *

*****************************************************************************
*****-*/
#ifdef _WIN32
# include <windows.h>
#endif /* _WIN32 */

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


#include <tpccstruct.h>
#include <tpcc_acmsxp.h>

#define MAX(a,b) ((a)>(b)?(a):(b))

void usage( char *prog );

int
main(int argc, char *argv[])
{
  FILE *fptr;
  char *dir;
  int  iMaxSize;

  dir = getenv( "USR_OBJ" );

  if( 2 != argc ) {
    usage( argv[0] );
  }

  if( NULL == ( fptr = fopen( argv[1],"w" )))
  {
    fprintf( stderr, "\nCould not open file %s\n", argv[1] );
    exit( 1 );
  }

  iMaxSize = 0;
  iMaxSize = MAX(iMaxSize,sizeof(DeliveryData));
  iMaxSize = MAX(iMaxSize,sizeof(NewOrderData));
  iMaxSize = MAX(iMaxSize,sizeof(OrderStatusData));
  iMaxSize = MAX(iMaxSize,sizeof(PaymentData));
  iMaxSize = MAX(iMaxSize,sizeof(StockLevelData));

  fprintf(fptr, "RECORD io_login_wksp\n");
  fprintf(fptr, "\tlogin_data TEXT SIZE %d;\n",sizeof( LoginData ));
  fprintf(fptr, "END RECORD;\n\n");
  fprintf(fptr, "RECORD io_dy_wksp\n");
  fprintf(fptr, "\tdy_data TEXT SIZE %d;\n",sizeof( DeliveryData ));
  fprintf(fptr, "END RECORD;\n\n");
  fprintf(fptr, "RECORD io_no_wksp\n");
  fprintf(fptr, "\tno_data TEXT SIZE %d;\n",sizeof( NewOrderData ));
  fprintf(fptr, "END RECORD;\n\n");
  fprintf(fptr, "RECORD io_pt_wksp\n");
  fprintf(fptr, "\tpt_data TEXT SIZE %d;\n",sizeof( PaymentData ));
  fprintf(fptr, "END RECORD;\n\n");
  fprintf(fptr, "RECORD io_os_wksp\n");
  fprintf(fptr, "\tos_data TEXT SIZE %d;\n",sizeof( OrderStatusData ));
  fprintf(fptr, "END RECORD;\n\n");
  fprintf(fptr, "RECORD io_sl_wksp\n");
  fprintf(fptr, "\tsl_data TEXT SIZE %d;\n",sizeof( StockLevelData ));
  fprintf(fptr, "END RECORD;\n\n");
  fprintf(fptr, "RECORD io_gc_wksp\n");
  fprintf(fptr, "\tgc_data TEXT SIZE %d;\n", iMaxSize );
  fprintf(fptr, "END RECORD;\n\n");
  fprintf(fptr, "RECORD int_gc_wksp\n");
  fprintf(fptr, "\ttrans INTEGER;\n");
  fprintf(fptr, "END RECORD;\n\n");

  fclose(fptr);
  printf( "\n File %s generated.\n", argv[1] );

  return 0;
}

void
usage( char *prog )
{
  fprintf( stderr, "usage: %s <output file name>\n", prog );

  exit( 1 );
}
```

deli_cli.c

```
/*+*****************************************************************************
 *                                                                           *
 *  COPYRIGHT (c) 1997, 2000 BY                                              *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                                     *
 *                                                                           *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                             *
 *                                                                           *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND SHOULD NOT BE CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                             *
 *                                                                           *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
```

```
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                      *
 *                                                      *

**********************************************************************
*****-*/

/*+
 * Abstract: This file contains functions for the delivery server client code.
 *
 * Author:  W Carr
 * Creation Date: July 1997
 *
 * Modified history:
 *
 *
*/

#define DELI_CLI_C

#ifdef _WIN32
# include <windows.h>
#else
# include <unistd.h>
# include <errno.h>
# include <sys/time.h>
#endif

#include <time.h>

#include <tpccstruct.h>
#include <tpccapi.h>
#include <tpccerr.h>

#ifdef _WIN32
# include <buf.h>
#endif
#include <deli_srv.h>
#include <transpool.h>

#ifdef _WIN32
static BOOL gbUsePipe = FALSE;
#else
static BOOL gbUsePipe = TRUE;
#endif

static BOOL gbDisplayCompletions = TRUE;

/* FUNCTION: int DELIClientStartup( void )
 *
 * PURPOSE:             This function prepares the delivery client for processing.
 *
 * ARGUMENTS:           None
 *
 * RETURNS:             int                     Error code if unsuccessful
 *                                              ERR_SUCCESS if no error

 *
 * COMMENTS:            This code presumes that the server portion of the code
 *                      has been started first, otherwise, a call to the delivery
 *                      function will fail.
 *
 */

int
DELIClientStartup( pDeliveryTransportData pDeliveryTransport,
                                pLoginData pLogin, int loginDelay, char *Path )
{
  int status;

#ifdef _WIN32
  /* Let the caller decide on use of pipe. */
  gbUsePipe = pDeliveryTransport->use_pipe;
  if( gbUsePipe )
  {
    gbDisplayCompletions = FALSE;
  }
  else
  {
    gbDisplayCompletions = pDeliveryTransport->display_completions;
  }
#endif

  status = DELIServerStartup( pDeliveryTransport, pLogin, loginDelay, Path );
  if( ERR_SUCCESS != status )
    return( status );

  return( ERR_SUCCESS );
}

/* FUNCTION: int DELIClientShutdown( void )
 *
 * PURPOSE:             This function cleans up allocated objects to allow for
 *                      termination of the delivery subsystem.
 *
 * ARGUMENTS:           None
 *
 * RETURNS:             int                     Error code if unsuccessful
 *                                              ERR_SUCCESS if no error

 *
 * COMMENTS:            None
 *
 */

int
DELIClientShutdown( void )
{
  return( DELIServerShutdown( ));
}

/* FUNCTION: int TPCCDelivery( pDeliveryData *pppDelivery,
 *                      pDeliveryData
pCompletedDeliveries[DELIVERY_RESPONSE_COUNT] )
 *
 * PURPOSE:             Writes the input portion the the delivery structure to the
 *                      server.
 *
 * ARGUMENTS:           DeliveryData            *pDelivery   Delivery data i/o struct
ptr
 *
 * RETURNS:             int             ERR_DB_SUCCESS                  success
 *                                      ERR_DB_NOT_COMMITED             other
error
 *
 * COMMENTS:
 *
 */

int
TPCCDelivery( pDeliveryData *ppDelivery,
                        pDeliveryData
pCompletedDeliveries[DELIVERY_RESPONSE_COUNT] )
{
  pDeliveryData                                 pDeliveryCopy;
  size_t                                bw;
  size_t                                br;
  int                                           ii;
  int                                           status;

  /* Pull any completions off the output queue.  We do this first to help */
  /* prevent the deadlock situation where both sides of the delivery system */
  /* write before reading. */
  ii = 0;
  if( gbDisplayCompletions ) {
    while( ii < DELIVERY_RESPONSE_COUNT ) {
#ifdef _WIN32
      status = bufread( &pCompletedDeliveries[ii], sizeof( pDeliveryData ),
                                                &br, 0, outputbuf );
      if( BUF_READTIMEOUT == status ) {
                /* there was not a completed transaction waiting */
                break;
      }
      else if( BUF_SUCCESS != status )
                return ERR_DELIVERY_OUTPUT_PIPE_READ;
#else
      pCompletedDeliveries[ii] = ReserveTransactionStruct( DELIVERY_TRANS );
      br = sizeof( DeliveryData );
      status = DELIQueueRead( outputbuf,
```

```
                                        (void *)pCompletedDeliveries[ii], &br,
FALSE );
    if( ERR_SUCCESS != status )
    {
            return( status );
    }
    if( 0 == br )
    {
            UnreserveTransactionStruct( DELIVERY_TRANS,
&pCompletedDeliveries[ii] );
            break;
    }
#endif
    ii++;
   }
  }

  /* Zero out this and all remaining delivery records. */
  while( ii < DELIVERY_RESPONSE_COUNT )
  {
   pCompletedDeliveries[ii++] = NULL;
  }

  /* get local queue time if nothing supplied */
  if( 0 == (*ppDelivery)->queue_time )
   time( &(*ppDelivery)->queue_time );

  /* Overload the delta time to be the local queue time in milliseconds. */
  /* On the server side, we will get the tick count and subtract to  */
  /* calcuate the delta.
               */
  StartTime( (*ppDelivery)->delta_time );

  if( gbUsePipe ) {
#ifdef _WIN32
    if( 0 == WriteFile( ghPipeInputWrite, (*ppDelivery),
                                        sizeof(DeliveryDataInput), &bw, NULL )) {
     status = GetLastError( );
     return ERR_DB_DELIVERY_NOT_QUEUED;
    }
#else
#pragma message ("FIXME: When the delivery queue fills on Unix, you can reach a
deadlock.  There are two ways around this.  First, don't block and send an error
message to the caller that they should set the msg-tql parameter in the ipc section of
the sysconfigtab.  Second, use mmap() to create a shared memory section and port the
NT buf code so that the buffer size can be managed by application code.")
    status = DELIQueueWrite( inputbuf, (void *)*ppDelivery,
                                        sizeof( DeliveryDataInput ), TRUE );
    if( ERR_SUCCESS != status )
    {
     if( ERR_DB_DELIVERY_NOT_QUEUED == status )
     {
            TPCCErr( "Check the setting of msg-tql in the ipc: section of the
sysconfigtab.\r\n" );
     }
     return( status );
    }
#endif
  }
#ifdef _WIN32
  else {
    /* We need to post a copy of the delivery data to be processed so that */
    /* the original data structure can be returned as having been posted. */
    pDeliveryCopy = ReserveTransactionStruct( DELIVERY_TRANS );
    memcpy( (char *)pDeliveryCopy, (char *)(*ppDelivery),
              sizeof( DeliveryDataInput ));

    /* Since a copy of the delivery transaction data structure has been */
    /* reserved, we only wish to pass its address to the delivery server. */
    /* Since the buffer code assumes that you wish to copy what is at the */
    /* pointer value (it will dereference the pointer) what we want to pass */
    /* is a pointer to the pointer. */
    if( BUF_SUCCESS != bufwrite( &pDeliveryCopy, sizeof( pDeliveryData ),
                                        &bw, INFINITE, inputbuf ))
     return ERR_DB_DELIVERY_NOT_QUEUED;

  }
#endif

  /* Fake out the transaction pool code that the inbound transaction did all */
  /* the steps of the process. */
  TRANSACTION_DEBUG_STAGE( *ppDelivery,

IN_LH|IN_RH|IN_DB|LEAVING_DB|LEAVING_RH|LEAVING_LH );

  return ERR_DB_SUCCESS;
}
```

deli_cli.h

```
#ifndef DELI_CLI_H
#define DELI_CLI_H
/*+*****************************************************************************
*******
 *                                                                  *
 *  COPYRIGHT (c) 1997, 2000 BY                                     *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                            *
 *                                                                  *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                    *
 *                                                                  *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                    *
 *                                                                  *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                  *
 *                                                                  *

*****************************************************************************
******-*/
/*+
 * Abstract: This file contains definitions and declarations for the
 *                              delivery server client code.
 *
 * Author: W Carr
 * Creation Date: July 1997
 *
 * Modified history:
 *
 *
*/

int DELIClientStartup( pDeliveryTransportData pDeliveryTransport,
                                pLoginData pLogin, int loginDelay, char *Path );
int DELIClientShutdown( void );

#endif /* DELI_CLI_H */
```

deli_srv.c

```
/*+*****************************************************************************
*******
 *                                                                  *
 *  COPYRIGHT (c) 1997, 2000 BY                                     *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
```

```
 *  ALL RIGHTS RESERVED.                                        *
 *                                                              *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                *
 *                                                              *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                *
 *                                                              *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                              *
 *                                                              *

**********************************************************************
*****-*/

/*+
 * Abstract: This file contains functions for the delivery server queue.
 *
 * Author:  W Carr
 * Creation Date: July 1997
 *
 * Modified history:
 *
 *
*/

#define DELI_SRV_C

#ifdef _WIN32
# include <windows.h>
# include <winsock.h>
# include <process.h>
#else
# include <unistd.h>
# include <errno.h>
# include <signal.h>
# include <fcntl.h>
# include <sys/file.h>
# include <sys/time.h>
# include <sys/wait.h>
# include <sys/msg.h>
# define __stdcall
# define GetCurrentThreadId getpid
#endif
#include <time.h>
#include <stdio.h>
#include <stdlib.h>
#ifdef _WIN32
# include <crtdbg.h>
# include <buf.h>
#endif

#include <tpccstruct.h>
#include <tpccapi.h>
#include <tpccerr.h>

#include <transpool.h>
#include <deli_srv.h>

static int      giDeliInitStatus = ERR_SUCCESS;
static long     gDeliThreadStartCtr;
#ifdef USE_PROCESSES
static BOOL gbUsePipe = TRUE;                   /* Use OS pipe delivery code */
#else /* USE_PROCESSES */
static BOOL gbUsePipe = FALSE;                  /* Use OS pipe delivery code */
static HANDLE           gDeliThreadStartEvent;
/* Used for OS pipe deliveries */
static HANDLE           hPipeInputRead = INVALID_HANDLE_VALUE;
static BOOL bDone;                                      /* termination request
flag */
#endif /* USE_PROCESSES */
static BOOL gbDisplayCompletions = TRUE;        /* Send comp. back to UI? */

static FILE   *fpLog = NULL;                            /* pointer to log file */

static BOOL gbWriteDeliLog = FALSE;                     /* log transactions to
file ? */
static BOOL gbFlushDeliLog = FALSE;                     /* Flush log info when
written*/
static BOOL gbDirectConnect = FALSE;            /* Use transport or connect */

/* function prototypes */
void                    DELIErrorMessage(int iError);
void                    DELILog( pDeliveryData pDelivery );
unsigned        __stdcall DELIThread( void *ptr );
#ifdef USE_PROCESSES
# define        DELIVERY_MESSAGE 1
# define        DELIVERY_SHUTDOWN -1
# define        DELIVERY_STARTUP_SUCCESS -2
# define        DELIVERY_STARTUP_FAILURE -3
# define        INPUT_QUEUE_ID 71759
# define        OUTPUT_QUEUE_ID 10492
# define        DELI_PARAMS_NAME "/tmp/deli_server.params"
# define        DELI_LOCK_NAME "/tmp/deli_server.lock"
typedef struct _DELIMsgStruct
{
  long int              iType;
  DeliveryData                  Delivery;
} DELIMsgStruct, *pDELIMsgStruct;
typedef struct _DELIServerStruct
{
  int                           RunId;
  int                           CurrentServerCount;
  int                           CurrentClientCount;
} DELIServerStruct, *pDELIServerStruct;
int                     DELIServerShutdownInternal( pDELIServerStruct pdss );
void                    DELISigchldHandler( int );
#if 1
# define        CRASH *segv = 0
int                             *segv = NULL;
#else
int                             num = 1;
int                             denom = 0;
# define        CRASH num/denom;
#endif
#endif

/* FUNCTION: void DELIErrorMessage(int iError)
 *
 * PURPOSE:             This function writes an error message to the error log file.
 *
 * ARGUMENTS:           int                     iError          error id to be logged
 *
 * RETURNS:             None
 *
 * COMMENTS:            None
 *
 */

void
DELIErrorMessage(int iError)
{
  TPCCErr( "*Error(%d): %s\r\n", iError, TPCCErrString( iError ));
  return;
}

/* FUNCTION: void DELILog( pDeliveryData pDelivery )
 *
 * PURPOSE:             Writes the delivery results to the delivery log file.
 *
 * ARGUMENTS:           LPSYSTEMTIME    lpBegin                 Local
delivery start time.
 *                      pDeliveryData           pDelivery       Delivery data to be
written.
 *
```

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

void
DELILog( pDeliveryData pDelivery )
{
  struct tm             start;
  struct tm             *end;
  time_t                endt;
  unsigned              delta_time_seconds;
  unsigned              delta_time_milliseconds;

  delta_time_seconds = pDelivery->delta_time / 1000;
  delta_time_milliseconds = pDelivery->delta_time - (delta_time_seconds * 1000);

  memcpy( &start, localtime( &pDelivery->queue_time ), sizeof( start ));
  endt = pDelivery->queue_time + delta_time_seconds;
  end = localtime( &endt );

  fprintf( fpLog,
                "%2.2d/%2.2d/%2.2d,"
                "%2.2d:%2.2d:%2.2d:%3.3d,"
                "%2.2d:%2.2d:%2.2d:%3.3d,"
                "%8.8d,"
                "%5.5d,%2.2d,"
                "%4.4d,%4.4d,%4.4d,%4.4d,%4.4d,"
                "%4.4d,%4.4d,%4.4d,%4.4d,%4.4d\r\n",
                start.tm_year, start.tm_mon, start.tm_mday,
                start.tm_hour, start.tm_min, start.tm_sec, 0,
                end->tm_hour, end->tm_min, end->tm_sec, delta_time_milliseconds,
                pDelivery->delta_time,
                pDelivery->w_id, pDelivery->o_carrier_id,
                pDelivery->o_id[0], pDelivery->o_id[1],
                pDelivery->o_id[2], pDelivery->o_id[3],
                pDelivery->o_id[4], pDelivery->o_id[5],
                pDelivery->o_id[6], pDelivery->o_id[7],
                pDelivery->o_id[8], pDelivery->o_id[9] );

  if ( gbFlushDeliLog )
    fflush(fpLog);

  return;
}


#ifdef USE_PROCESSES
int
DELIQueueRead( int pipe, void *data, unsigned long *plen, BOOL block )
{
  int                           status;
  int                           err;
  DELIMsgStruct                 DeliMsg;
  int                           retlen;
  int                           flags;

  flags = ( block ) ? 0 : IPC_NOWAIT;
  retlen =  msgrcv( pipe, &DeliMsg, *plen, DELIVERY_MESSAGE, flags );
  if( -1 == retlen && ENOMSG == errno )
  {
    *plen = 0;
    status = ERR_SUCCESS;
  }
  else if( -1 == retlen || *plen != retlen )
  {
    err = errno;
    status = ERR_DELIVERY_PIPE_READ;
  }
  else
  {
    memcpy( data, &DeliMsg.Delivery, retlen );
    status = ERR_SUCCESS;
  }

  return( status );
}
#endif /* USE_PROCESSES */


#ifdef USE_PROCESSES
int
DELIQueueWrite( int pipe, void *data, unsigned long len, BOOL block )
{
  int                           status;
  int                           err;
  DELIMsgStruct                 DeliMsg;
  int                           retlen;
  int                           flags;

  DeliMsg.iType = DELIVERY_MESSAGE;
  memcpy( &DeliMsg.Delivery, data, len );

  flags = ( block ) ? 0 : IPC_NOWAIT;
  retlen = msgsnd( pipe, &DeliMsg, len, flags );
  if( -1 == retlen && EAGAIN == errno )
  {
    status = ERR_DB_DELIVERY_NOT_QUEUED;
  }
  else if( -1 == retlen )
  {
    err = errno;
CRASH;
    status = ERR_DB_DELIVERY_NOT_QUEUED;
  }
  else
  {
    status = ERR_SUCCESS;
  }

  return( status );
}
#endif /* USE_PROCESSES */


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

int
DELIServerShutdown( void )
{
  int                           status;
#ifdef USE_PROCESSES
  int                           err;
  int                           lock;
  int                           paramsFD;
  DELIServerStruct      dss;
  int                           size;
#endif /* USE_PROCESSES */

  status = ERR_SUCCESS;

#ifdef USE_PROCESSES
  if( -1 == ( lock = open( DELI_LOCK_NAME, O_WRONLY | O_CREAT, 0666 )))
  {
    err = errno;
    status = ERR_DELIVERY_PARAMS_FILE;
  }
  else
  {
    flock( lock, LOCK_EX );             /* Exclusive lock */

    if( -1 == ( paramsFD = open( DELI_PARAMS_NAME, O_RDWR, 0666 )))
    {
      status = ERR_DELIVERY_SHUTDOWN;
    }
    else
    {
      /* A params file exists.  Use its content to shutdown the servers. */
```

```
    size = sizeof( DELIServerStruct );
    if( size != read( paramsFD, &dss, size ))
    {
            err = errno;
            status = ERR_DELIVERY_PARAMS_FILE;
    }
    else if( 0 == --dss.CurrentClientCount )
    {
            status = DELIServerShutdownInternal( &dss );
            unlink( DELI_PARAMS_NAME );
            unlink( DELI_LOCK_NAME );
    }
    else
    {
            if( (off_t) -1 == lseek( paramsFD, 0, SEEK_SET ) ||
               size != write( paramsFD, &dss, size ))
            {
              err = errno;
              status = ERR_DELIVERY_PARAMS_FILE;
            }
    }
    close( paramsFD );
   }

   close( lock );
  }
#else /* USE_PROCESSES */
  if( gbUsePipe ) {
    CloseHandle( hPipeInputRead );
    CloseHandle( ghPipeInputWrite );
  }
  else {
    bufclose( inputbuf );
    bufclose( outputbuf );
  }
#endif /* USE_PROCESSES */

  if ( fpLog )
    fclose(fpLog);

  fpLog = NULL;

  return( status );
}

#ifdef USE_PROCESSES
int
DELIServerShutdownInternal( pDELIServerStruct pdss )
{
  int                   status;
  int                   inerr;
  int                   outerr;
  int                   ii;
  int                   Count;
  DeliveryDataInput     DeliveryClose;

  if( -1 != inputbuf )
  {
    DeliveryClose.o_carrier_id = DELIVERY_SHUTDOWN;
    Count = pdss->CurrentServerCount;
    for( ii = 0; ii < Count; ii++ ) {
      status = DELIQueueWrite( inputbuf, (void *)&DeliveryClose,
                               sizeof( DeliveryClose ), FALSE );
      if( ERR_SUCCESS != status && ERR_DB_DELIVERY_NOT_QUEUED !=
status )
      {
            return( status );
      }
    }
    if( -1 == msgctl( inputbuf, IPC_RMID, NULL ))
    {
      inerr = errno;
CRASH;
      return( ERR_DELIVERY_PIPE_DESTROY );
    }
  }
/* #if
MSG_QUEUES_SEEM_TO_DELETE_THEMSELVES_ON_EXIT_OF_LAST_REA
DER */
#if 1
  if( -1 != outputbuf )
  {
    if( -1 == msgctl( outputbuf, IPC_RMID, NULL ))
    {
      outerr = errno;
CRASH;
      return( ERR_DELIVERY_PIPE_DESTROY );
    }
  }
#endif

  return( ERR_SUCCESS );
}
#endif

/* FUNCTION: int DELIServerStartup( pDeliveryTransportData pDeliveryTransport,
                                    pLoginData pLogin, int
loginDelay,
                                    char *Path )
 *
 * PURPOSE:          This function prepares the delivery subsystem for execution.
 *
 * ARGUMENTS:        None
 *
 * RETURNS:          int         iError              Error code if
unsuccessfull
 *                               ERR_SUCCESS         No error successfull
code
 *
 *
 * COMMENTS:         None
 *
 */

int
DELIServerStartup( pDeliveryTransportData pDeliveryTransport,
                   pLoginData pLogin, int loginDelay, char *Path )
{
  char                  LogName[PATH_MAX];
#ifdef USE_PROCESSES
  int                   ii;
  int                   err;
  int                   inerr;
  int                   outerr;
  int                   childPid;
  int                   status;
  int                   lock;
  int                   paramsFD;
  DELIServerStruct      dss;
  int                   ParamsSize = sizeof( DELIServerStruct );
  int                   size;
#else /* USE_PROCESSES */
  int                   ii;
  int                   iError;
  size_t                inputbufsize;
  size_t                outputbufsize;
  unsigned              tid;
  unsigned long                 ulhThread;
  HANDLE                hThread;
#endif /* USE_PROCESSES */

  gDeliThreadStartCtr = pDeliveryTransport->num_threads;
  gbWriteDeliLog = pDeliveryTransport->write_log;
  gbFlushDeliLog = pDeliveryTransport->flush_log;
  gbDirectConnect = !pDeliveryTransport->use_transport;
  strcat( strcpy( LogName, Path ), "delilog." );

#ifdef USE_PROCESSES
  status = ERR_SUCCESS;
  if( -1 == ( lock = open( DELI_LOCK_NAME, O_WRONLY | O_CREAT, 0666 )))
  {
    err = errno;
CRASH;
    status = ERR_DELIVERY_PARAMS_FILE;
  }
  else
  {
    flock( lock, LOCK_EX );             /* Exclusive lock */
```

```
   if( -1 == ( paramsFD = open( DELI_PARAMS_NAME, O_RDWR | O_CREAT,
0666 )))
   {
    err = errno;
    status = ERR_DELIVERY_PARAMS_FILE;
   }
   else
   {
    /* A params file exists.  Check if its for the current or an old run. */
    size = read( paramsFD, &dss, ParamsSize );
    if(( 0 != size ) && ( ParamsSize != size ))
    {
                /* File exists, but content is invalid. */
                status = ERR_DELIVERY_PARAMS_FILE;
    }
    else if(( 0 != size ) && ( pLogin->databaseLogin.iRunId == dss.RunId ))
    {
                /* servers have been started for this run, update client count */
                dss.CurrentClientCount++;
    }
    else
    {
                status = ERR_SUCCESS;
                if(( 0 != size ) && ( pLogin->databaseLogin.iRunId != dss.RunId ))
                {
                 /* we have to cleanup a previous run */
                 if( -1 == ( inputbuf = msgget( INPUT_QUEUE_ID, IPC_R )))
                 {
                  inerr = errno;
                 }
                 if( -1 == ( outputbuf = msgget( OUTPUT_QUEUE_ID, IPC_R )))
                 {
                  outerr = errno;
                 }
                 if(( -1 == inputbuf && ENOENT != inerr ) ||
                    ( -1 == outputbuf && ENOENT != outerr ))
                 {
CRASH;
                  status = ERR_DELIVERY_PIPE_OPEN;
                 }
                 else if( -1 != inputbuf || -1 != outputbuf )
                 {
                  status = DELIServerShutdownInternal( &dss );
                 }
                 else
                 {
                  status = ERR_SUCCESS;
                 }
                }
                if( ERR_SUCCESS == status )
                {
                 /* Start the servers. */
                 if( -1 == msgget( INPUT_QUEUE_ID,
IPC_CREAT|IPC_EXCL|IPC_R|IPC_W ) ||
                     -1 == msgget( OUTPUT_QUEUE_ID,
IPC_CREAT|IPC_EXCL|IPC_R|IPC_W ))
                 {
                  err = errno;
CRASH;
                  status = ERR_DELIVERY_PIPE_CREATE;
                 }
                 else
                 {
                  signal( SIGCHLD, DELISigchldHandler );
                  for( ii = 0; ii < pDeliveryTransport->num_threads; ii++ )
                  {
                   if( -1 == ( childPid = fork( )))
                   {
                                err = errno;
                                status = ERR_CANT_START_DELIVERY_THREAD;
                   }
                   else if( 0 < childPid )
                   {
                                /* we are the parent */
                                Sleep( loginDelay );
                   }
                   else
                   {
                                /* we are the child */
                                close( paramsFD );
                                close( lock );
                                if( -1 == ( inputbuf = msgget( INPUT_QUEUE_ID, IPC_R
)) ||
                                   -1 == ( outputbuf = msgget( OUTPUT_QUEUE_ID,
IPC_W )))
                                {
CRASH;
                                 status = ERR_DELIVERY_PIPE_OPEN;
                                }
                                /* Open a log file if requested. */
                                else if( gbWriteDeliLog && NULL ==
(fpLog=fopen(LogName, "wb")))
                                {
                                 status = ERR_CANNOT_CREATE_RESULTS_FILE;
                                }
                                else
                                {
                                 status = DELIThread( pLogin );
                                }
                                exit( status );
                   }
                  }
                  dss.RunId = pLogin->databaseLogin.iRunId;
                  dss.CurrentServerCount = pDeliveryTransport->num_threads;
                  dss.CurrentClientCount = 1;
                 }
                }
    }

    if( (off_t) -1 == lseek( paramsFD, 0, SEEK_SET ) ||
                ParamsSize != write( paramsFD, &dss, ParamsSize ))
    {
                err = errno;
                status = ERR_DELIVERY_PARAMS_FILE;
    }
    close( paramsFD );
   }
   close( lock );
  }

  if( -1 == ( inputbuf = msgget( INPUT_QUEUE_ID, IPC_W )) ||
     -1 == ( outputbuf = msgget( OUTPUT_QUEUE_ID, IPC_R )))
  {
CRASH;
    status = ERR_DELIVERY_PIPE_OPEN;
  }

  return( status );
#else /* USE_PROCESSES */
  /* Let the caller decide on use of pipe. */
  gbUsePipe = pDeliveryTransport->use_pipe;
  if( gbUsePipe )
  {
   gbDisplayCompletions = FALSE;
  }
  else
  {
   gbDisplayCompletions = pDeliveryTransport->display_completions;
  }

  /* Open a log file if requested. */
  if( gbWriteDeliLog ) {
   fpLog = fopen(LogName, "wb");
   if ( NULL == fpLog )
    return ERR_CANNOT_CREATE_RESULTS_FILE;
  }

  if( gbUsePipe ) {
   /* start delivery pipe */
   inputbufsize = pDeliveryTransport->num_threads *
     pDeliveryTransport->num_queued_deliveries * sizeof( DeliveryData );
   if(0 == CreatePipe(&hPipeInputRead, &ghPipeInputWrite, NULL, inputbufsize)){
    iError = GetLastError( );
    return( ERR_DELIVERY_PIPE_CREATE );
   }
  }
  else {
   /* create delivery buffer */
   inputbufsize =
     pDeliveryTransport->num_queued_deliveries * sizeof( pDeliveryData );
```

```
    if( BUF_SUCCESS != bufopen( inputbufsize, &inputbuf ))
      return ERR_DELIVERY_PIPE_OPEN;

    if( gbDisplayCompletions ) {
      /* create response buffer */
      outputbufsize =
                pDeliveryTransport->num_queued_responses * sizeof( pDeliveryData );
      if( BUF_SUCCESS != bufopen( outputbufsize, &outputbuf ))
                return ERR_DELIVERY_PIPE_OPEN;
    }
  }

  /* prepare to start and verify starting of delivery threads */
  gDeliThreadStartEvent = CreateEvent( NULL, TRUE, FALSE, NULL );
  if ( gDeliThreadStartEvent == NULL )
    return ERR_CANT_CREATE_FORCE_THRED_STRT_EVENT;

  bDone = FALSE;

  if ( !bDone ){
    for( ii = 0; ii < pDeliveryTransport->num_threads; ii++ ) {
      ulhThread = _beginthreadex( NULL, 0, DELIThread, pLogin, 0, &tid );
      if( 0 == ulhThread )
                return ERR_CANT_START_DELIVERY_THREAD;
      hThread = ( HANDLE )ulhThread;
      CloseHandle( hThread );
      if( !pDeliveryTransport->use_transport )
                Sleep( loginDelay );
    }
  }

  WaitForSingleObject( gDeliThreadStartEvent, INFINITE );

  return giDeliInitStatus;
#endif
}

#ifdef USE_PROCESSES
void
DELISigchldHandler( int sig )
{
  int           wait_status;

  /* Make sure we get all of the children who may have status change info */
  while( wait3(&wait_status, WNOHANG, 0) > 0 ) {
    gDeliThreadStartCtr--;
  }
}
#endif

/* FUNCTION: void DELIThread( void *ptr )
 *
 * PURPOSE:             This function is executed by each delivery thread.
 *                      A thread will block on a read of the buffer from the client.
 *                      Return from the read will indicate either data is to be
 *                      processed, an error has occured, or the buffer has been
closed.
 *
 * ARGUMENTS:           void            *ptr            unused, passed by thread creation
routine.
 *
 * RETURNS:             None
 *
 * COMMENTS:            The registry key value NumberOfDeliveryThreads controls
how
 *                      many of these functions are running.
 *
 */

unsigned __stdcall
DELIThread( void *ptr )
{
  size_t                br;
  pDeliveryData                 pDelivery;
  int                           retcode;
  CallersContext        CC;
  pLoginData                    pCallersLogin;
  LoginData             login;
  char                          tmp[16];
  int                           status;
  DBContext             DBC;
#ifdef USE_PROCESSES
  BOOL                                  bDone = FALSE;
#endif /* USE_PROCESSES */

  /* Initialize calling parameters */
  pCallersLogin = ptr;
  CC = 0;
  DBC = INVALID_DB_CONTEXT;

  /* Should we connect directly ourselves, or should we use the transport to */
  /* perform the deliveries in the same manner as the other 4 transactions?  */
  if( gbDirectConnect ) {
    gethostname(tmp, sizeof(tmp));
    login.w_id = 0;
    login.ld_id = 0;
    login.CC = CC;
    strcpy( login.databaseLogin.szServer,
                pCallersLogin->databaseLogin.szServer );
    strcpy( login.databaseLogin.szDatabase,
                pCallersLogin->databaseLogin.szDatabase );
    strcpy( login.databaseLogin.szUser, pCallersLogin->databaseLogin.szUser );
    strcpy( login.databaseLogin.szPassword,
                pCallersLogin->databaseLogin.szPassword );
    sprintf( login.databaseLogin.szApplication, "%s:delisrv - %d",
                 tmp, GetCurrentThreadId());
    status = TPCCConnectDB( &DBC, &login );
    if( ERR_DB_SUCCESS != status && ERR_SUCCESS != giDeliInitStatus )
    {
      giDeliInitStatus = status;
    }
  }

#ifdef USE_PROCESSES
#else /* USE_PROCESSES */
  /* wait until all threads are started */
  if ( InterlockedDecrement( &gDeliThreadStartCtr ) == 0 )
    SetEvent( gDeliThreadStartEvent );

  WaitForSingleObject( gDeliThreadStartEvent, INFINITE );
#endif /* USE_PROCESSES */

  if( ERR_SUCCESS != giDeliInitStatus )
  {
    return( giDeliInitStatus );
  }

  /* while delisrv running i.e. user has not requested termination */
  while( !bDone ){

    if( gbUsePipe ) {
      pDelivery = ReserveTransactionStruct( DELIVERY_TRANS );
#ifdef USE_PROCESSES
      br = sizeof( DeliveryDataInput );
      status = DELIQueueRead( inputbuf, (void *)pDelivery, &br, TRUE );
      if( ERR_SUCCESS != status )
      {
                return( status );
      }
      else if( DELIVERY_SHUTDOWN == pDelivery->o_carrier_id )
      {
                bDone = TRUE;
                continue;
      }
#else /* USE_PROCESSES */
      if( 0 == ReadFile( hPipeInputRead, pDelivery, sizeof( DeliveryDataInput ),
                                        &br, NULL )) {
                status = GetLastError( );
                DELIErrorMessage( ERR_DELIVERY_PIPE_READ );
                continue;
      }
#endif /* USE_PROCESSES */
    }
#ifdef USE_PROCESSES
#else /* USE_PROCESSES */
    else {
      if(BUF_SUCCESS != bufread( &pDelivery, sizeof( pDeliveryData ), &br,
                                                INFINITE, inputbuf )) {
                DELIErrorMessage( ERR_DELIVERY_PIPE_READ );
                continue;
```

```
    }
   }
#endif  /* USE_PROCESSES */
   if( gbDirectConnect )
    retcode = TPCCDeliveryDB( DBC, pDelivery );
   else
    retcode = TPCCDeliveryDeferred( &pDelivery );

   retcode = TPCCDeliveryDeferredResponse( retcode, pDelivery );
  }

  return( status );
}

/* FUNCTION: int TPCCDeliveryDeferredResponse( DeliveryData *deliveryData )
 *
 * PURPOSE:              This function writes the completed delivery transaction
 *                       record to the log file and to the queue for reporting
 *                       to the browser.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK *pECB
 *                       DeliveryData                    *deliveryData pointer
to the delivery
 *                                                                       data
structure.
 *
 * RETURNS:              none
 *
 * COMMENTS:             none
 */

int
TPCCDeliveryDeferredResponse( int retcode, pDeliveryData pDelivery )
{
  size_t                          bw;
  int                                     status;

  if ( ERR_DB_PENDING == retcode )
  {
   return( retcode );
  }
  else
  {
   if( ERR_DB_SUCCESS != retcode)
   {
    /* send a flag to the reducer to mark an error on the delivery */
    pDelivery->queue_time = 1;
    DELIErrorMessage(retcode);
   }
   /* The delta time was overloaded by the client to be the local queue time */
   /* in milliseconds.  Here we get the tick count and subtract to      */
   /* calcuate the delta.
                */
   DeltaTime( pDelivery->delta_time );

   /* update log */
   if( gbWriteDeliLog )
    DELILog( pDelivery );

   if( gbDisplayCompletions ) {
    /* send a delivery completion record */
#ifdef USE_PROCESSES
    status = DELIQueueWrite( outputbuf, (void *)pDelivery,
                                        sizeof( *pDelivery ), TRUE );
    if( ERR_SUCCESS != status )
    {
            DELIErrorMessage( ERR_DELIVERY_OUTPUT_PIPE_WRITE );
            return( retcode );
    }
    /* The delivery transaction is at the end of its life */
    UnreserveTransactionStruct( DELIVERY_TRANS, &pDelivery );
#else /* USE_PROCESSES */
    status = bufwrite( &pDelivery, sizeof(pDeliveryData),
                                    &bw, INFINITE, outputbuf );
    if( BUF_SUCCESS != status )
    {
            DELIErrorMessage( ERR_DELIVERY_OUTPUT_PIPE_WRITE );
            return( retcode );
    }
#endif /* USE_PROCESSES */
   }
   else {
    /* The delivery transaction is at the end of its life */
    UnreserveTransactionStruct( DELIVERY_TRANS, &pDelivery );
   }
  }

  return( retcode );
}
```

deli_srv.h

```
#ifndef DELI_SRV_H
#define DELI_SRV_H
/*+**************************************************************************
*******
 *                                                      *
 *  COPYRIGHT (c) 1997, 2000 BY                         *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                *
 *                                                      *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                        *
 *                                                      *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                        *
 *                                                      *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                      *
 *                                                      *

**************************************************************************
******-*/

/*+
 * Abstract: This file contains definitions of the delivery server queue.
 *
 * Author:  W Carr
 * Creation Date: July 1997
 *
 * Modified history:
 *              21-Oct-1997 WCarr       Replaced NamedPipe with buffer code.
 *
*/

#ifdef _WIN32
# define StartTime(num) ((num) = GetTickCount( ))
# define DeltaTime(num) ((num) = GetTickCount( ) - num)
#else
# include <sys/timeb.h>
  typedef int BUFPTR;
# define StartTime(num) \
  {\
    struct timeb tb;\
    ftime( &tb );\
    /* get the time since midnight today UTC in milliseconds */\
    (num) = ((tb.time % (24*60*60)) * 1000) + tb.millitm;\
  }
# define DeltaTime(num) \
  {\
    struct timeb tb;\
```

```
    int now;\
    ftime( &tb );\
    /* get the time since midnight today UTC in milliseconds */\
    now = ((tb.time % (24*60*60)) * 1000) + tb.millitm;\
    if( now < (num) )\
    {\
      /* we wrapped past midnight UTC, add a day in milliseconds */\
      now += (24*60*60*1000);\
    }\
    (num) = now - (num);\
  }
#endif

#ifdef DELI_SRV_C
# define DELI_GLOBAL(thing,init) thing = init
#else
# define DELI_GLOBAL(thing,init) extern thing
#endif

#if defined DELI_CLI_C || defined DELI_SRV_C
# ifdef _WIN32
/* Used for OS pipe deliveries */
DELI_GLOBAL(HANDLE ghPipeInputWrite,INVALID_HANDLE_VALUE);
DELI_GLOBAL(BUFPTR inputbuf,NULL);                  /* submitted deliveries
*/
DELI_GLOBAL(BUFPTR outputbuf,NULL);                 /* completed
deliveries */
# else
DELI_GLOBAL(BUFPTR inputbuf,-1);                    /* submitted deliveries
*/
DELI_GLOBAL(BUFPTR outputbuf,-1);                   /* completed
deliveries */
# endif
#endif

int DELIServerStartup( pDeliveryTransportData pDeliveryTransport,
                       pLoginData pLogin, int loginDelay, char *Path );
int DELIServerShutdown( void );
int DELIQueueRead( int pipe, void *data, unsigned long *plen, BOOL block );
int DELIQueueWrite( int pipe, void *data, unsigned long len, BOOL block );

#endif /* DELI_SRV_H */
```

oracle_db8.c

```
/*+ file: oracle_db8.c based on Oracle file tpccpl.c
/*+=============================================================
======+
|      Copyright (c) 1994  Oracle Corp, Redwood Shores, CA      |
|                OPEN SYSTEMS PERFORMANCE GROUP                 |
|                    All Rights Reserved                        |

+=============================================================
=====+
| DESCRIPTION
|   TPC-C transactions in PL/SQL.

+=============================================================
====-*/
/*+*************************************************************
*******
 *                                                              *
 *  COPYRIGHT (c) 1997, 2000 BY                                 *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                        *
 *                                                              *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY  OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                *
 *                                                              *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                *
 *                                                              *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                              *
 *                                                              *

*************************************************************************
*****-*/

#ifdef _WIN32
# include <windows.h>
# include <process.h>
#endif

#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/timeb.h>

#include <oci.h>
#include <ocidfn.h>
#include <ociapr.h>

#define ORACLE_DB_C

#include <tpccerr.h>
#include <tpccstruct.h>
#include <tpccapi.h>
#include <tpcc.h>
#include <oracle_db8.h>

#include <config.h>

#ifdef _WIN32
# include <crtdbg.h>
#endif

#define DEADLOCKRETRIES 6

char szErrorLogName[256];
char szOraLogName[256];
char szOraErrorLogName[256];

/* anonymous block statement buffers */
text stmbuf_payz[SQL_BUF_SIZE] = { '\0' };
text stmbuf_paynz[SQL_BUF_SIZE] = { '\0' };
text stmbuf_tkvcpnew[SQL_BUF_SIZE] = { '\0' };

/* prototypes */
void vgetdate (unsigned char *oradt) ;
void cvtdmy (unsigned char *oradt, char *outdate);
void cvtdmyhms (unsigned char *oradt, char *outdate);


FILE  *vopen(char *fnam, char *mode)
{
FILE *fd;

#ifdef  DEBUG
  TPCCErr("tkvuopen() fnam: %s,   mode: %s\n", fnam, mode);
#endif

   fd = fopen((char *)fnam,(char *)mode);
   if (!fd){
      TPCCErr(" fopen on %s failed %d\n",fnam,fd);
            /*       exit(-1);  */
   }
   return(fd);
}

int sqlfile(char *fnam, text *linebuf)
```

```
{
FILE *fd;
int nulpt = 0;

#ifdef  DEBUG
  TPCCErr("sqlfile() fnam: %s,  linebuf: %#x\n", fnam, linebuf);
#endif

    fd = vopen(fnam,"r");
    if(NULLP == fd)
    {
      return(ERR_DB_ERROR);
    }
    while (fgets((char *)linebuf+nulpt, SQL_BUF_SIZE,fd))
    {
       nulpt = strlen((char *)linebuf);
    }

    fclose( fd );

    return(nulpt);
}

int getfile(char *filename, text *filebuf)
{
  char parsbuf[SQL_BUF_SIZE];
  int  pathlen = sizeof( parsbuf ) - 1;

  if( ERR_SUCCESS != GetConfigStr( PATH, &pathlen, parsbuf ))
  {
    return( ERR_DB_ERROR );
  }
  strcat(parsbuf, filename);
  return(sqlfile(parsbuf, filebuf));
}


int TPCCStartupDB( pConfigData pConfig )
{
#ifdef DEBUG_TPCCSTARTUPDB
  _ASSERT(FALSE);
#endif

  /* Tell Oracle this is multi-threaded code */
  OCIInitialize(OCI_THREADED | OCI_OBJECT ,(dvoid *)0,0,0,0);

  if(ERR_DB_ERROR == getfile("paynz.sql",stmbuf_paynz))
  {
    TPCCErr("Error opening the file paynz.sql");
    return ERR_DB_ERROR;
  }

  if(ERR_DB_ERROR == getfile("payz.sql",stmbuf_payz))
  {
    TPCCErr("Error opening the file payz.sql");
    return ERR_DB_ERROR;
  }

  if(ERR_DB_ERROR == getfile("tkvcpnew.sql",stmbuf_tkvcpnew))
  {
    TPCCErr("Error opening the file tkvcpnew.sql");
    return ERR_DB_ERROR;
  }

  return ERR_DB_SUCCESS;

}

int TPCCShutdownDB(void)
{

  /* Add Oracle specific code */

  return ERR_DB_SUCCESS;
}


int ocierror(char *fname, int lineno, OraContext *p, sword status)
{
  char errbuf[512];
  text tempbuf[512];
  sb4 errcode;
  OCIError *errhp;

  errhp = p->errhp;

  switch (status) {
  case OCI_SUCCESS:
    return RECOVERR;
    break;
  case OCI_SUCCESS_WITH_INFO:
    sprintf(errbuf, "Module %s Line %d\r\n", fname, lineno);
    strcat(errbuf, "Error - OCI_SUCCESS_WITH_INFO\r\n");
    break;
  case OCI_NEED_DATA:
    sprintf(errbuf, "Module %s Line %d\r\n", fname, lineno);
    strcat(errbuf, "Error - OCI_NEED_DATA\r\n");
    break;
  case OCI_NO_DATA:
    sprintf(errbuf, "Module %s Line %d\r\n", fname, lineno);
    sprintf(errbuf, "Error - OCI_NO_DATA\r\n");
    return IRRECERR;
    break;
  case OCI_ERROR:
    (void) OCIErrorGet (errhp, (ub4) 1,
                                          (text *) NULL, &errcode, tempbuf,
                                          (ub4) sizeof(errbuf), OCI_HTYPE_ERROR);

    switch(errcode){
    case NOT_SERIALIZABLE:
    /* if error is NOT_SERIALIZABLE return without writing anything */
      return errcode;

    case DEADLOCK:
      TPCCErr("Warning Deadlock, being retried");
      return RECOVERR;

    case SNAPSHOT_TOO_OLD:
      /* SNAPSHOT_TOO_OLD is considered recoverable */
      TPCCErr("Error snapshot too old: %s", tempbuf);
      return RECOVERR;

    default:
    /* else write a message */
      /* All else are irrecoverable */
      TPCCErr("Module %s Line %d\r\nError - %s\r\n",
                    fname, lineno, tempbuf);
      return errcode;
    }

/* vmm313    TPCCDisconnectDB(p); */
/* vmm313    exit(1); */
    break;
  case OCI_INVALID_HANDLE:
    sprintf(errbuf, "Module %s Line %d\r\n", fname, lineno);
    strcat(errbuf, "Error - OCI_INVALID_HANDLE\r\n");
    TPCCErr("%s", errbuf);
    TPCCDisconnectDB(p, NULL);
    return IRRECERR;
    /*      terminate(-1);  */
    /*      exit(-1);   */
    break;
  case OCI_STILL_EXECUTING:
    sprintf(errbuf, "Module %s Line %d\r\n", fname, lineno);
    strcat(errbuf, "Error - OCI_STILL_EXECUTE\r\n");
    break;
  case OCI_CONTINUE:
    sprintf(errbuf, "Module %s Line %d\r\n", fname, lineno);
    strcat(errbuf, "Error - OCI_CONTINUE\r\n");
    break;
  default:
    break;
  }
  TPCCErr("%s", errbuf);
  return RECOVERR;
}


/* FUNCTION: int TPCCConnectDB()
 *
```

```
 * PURPOSE:            This function opens the sql connection for use.
 *
 * ARGUMENTS:
 * RETURNS:                    int         0           if successful
 *                                         1           if an error occurs
 *
 * COMMENTS:           None
 *
 */


int
TPCCConnectDB( DBContext *pDBC, pLoginData pLogin )
{

#define SERIAL_TXT "alter session set isolation_level = serializable"
#ifdef SQL_TRACE
#define SQLTXT1 "alter session set sql_trace = true"
#endif

  /* Add Oracle specific code */

  text stmbuf[100];
  OraContext *p;

  p = (OraContext *) malloc(sizeof(OraContext));

  *pDBC = (DBContext) p;

  /* initialize flags to not initialized */
  p->new_init = 0;
  p->pay_init = 0;
  p->ord_init = 0;
  p->sto_init = 0;
  p->del_init = 0;

  OCIEnvInit(&(p->tpcenv), OCI_DEFAULT, 0, (dvoid **)0);
  OCIHandleAlloc((dvoid *)p->tpcenv, (dvoid **)&(p->tpcsrv),
OCI_HTYPE_SERVER,
                  0 , (dvoid **)0);
  OCIHandleAlloc((dvoid *)p->tpcenv, (dvoid **)&(p->errhp),
OCI_HTYPE_ERROR,
                  0 , (dvoid **)0);
  OCIHandleAlloc((dvoid *)p->tpcenv, (dvoid **)&(p->datecvterrhp),
OCI_HTYPE_ERROR,
                  0 , (dvoid **)0);
  OCIHandleAlloc((dvoid *)p->tpcenv, (dvoid **)&(p->tpcsvc),
OCI_HTYPE_SVCCTX,
                  0 , (dvoid **)0);
  if (RECOVERR != (OCIERROR(p, OCIServerAttach(p->tpcsrv, p->errhp,
                  (text *)pLogin->databaseLogin.szServer,
                  strlen( pLogin->databaseLogin.szServer ),
OCI_DEFAULT))))
    return ERR_DB_ERROR;

  OCIAttrSet((dvoid *)p->tpcsvc, OCI_HTYPE_SVCCTX, (dvoid *)p->tpcsrv,
             (ub4)0, OCI_ATTR_SERVER, p->errhp);
  OCIHandleAlloc((dvoid *)p->tpcenv, (dvoid **)&(p->tpcusr),
OCI_HTYPE_SESSION,
                  0 , (dvoid **)0);
  OCIAttrSet((dvoid *)p->tpcusr, OCI_HTYPE_SESSION,
             (dvoid *)pLogin->databaseLogin.szUser,
             (ub4)strlen(pLogin->databaseLogin.szUser),
             OCI_ATTR_USERNAME, p->errhp);
  OCIAttrSet((dvoid *)p->tpcusr, OCI_HTYPE_SESSION,
             (dvoid *)pLogin->databaseLogin.szPassword,
             (ub4)strlen(pLogin->databaseLogin.szPassword),
             OCI_ATTR_PASSWORD, p->errhp);
  if (RECOVERR != (OCIERROR(p, OCISessionBegin(p->tpcsvc, p->errhp, p-
>tpcusr,
                                               OCI_CRED_RDBMS,
OCI_DEFAULT))))
    return (ERR_DB_ERROR);

  OCIAttrSet(p->tpcsvc, OCI_HTYPE_SVCCTX, p->tpcusr, 0,
OCI_ATTR_SESSION,
             p->errhp);

  /* run all transaction in serializable mode */
  OCIHandleAlloc(p->tpcenv, (dvoid **)&(p->curi), OCI_HTYPE_STMT, 0,
(dvoid**)0);
  sprintf ((char *) stmbuf, SERIAL_TXT);
  OCIStmtPrepare(p->curi, p->errhp, stmbuf, strlen((char *)stmbuf),
                 OCI_NTV_SYNTAX, OCI_DEFAULT);
  if (RECOVERR != OCIERROR(p, OCIStmtExecute(p->tpcsvc, p->curi, p->errhp,
                                             1, 0, 0, 0,
OCI_DEFAULT)))
    return (ERR_DB_ERROR);
  OCIHandleFree(p->curi, OCI_HTYPE_STMT);


#ifdef SQL_TRACE
  /* Turn on the SQL_TRACE */
  OCIHandleAlloc(p->tpcenv, (dvoid **)&(p->curi), OCI_HTYPE_STMT, 0,
&xmem);
  sprintf ((char *) stmbuf, TRACE_TXT);
  OCIStmtPrepare(p->curi, p->errhp, stmbuf, strlen((char *)stmbuf),
                 OCI_NTV_SYNTAX, OCI_DEFAULT);
  if (RECOVERR != OCIERROR(p, OCIStmtExecute(p->tpcsvc, p->curi, p->errhp,
                                             1, 0, 0, 0,
OCI_DEFAULT)))
    return(ERR_DB_ERROR);
  OCIHandleFree((dvoid *)p->curi, OCI_HTYPE_STMT);
#endif /* End SQL_TRACE */

  /****  logon = 1;***/


  if (tkvcninit (&(p->bindvars.info.newOrder), p)) {
    TPCCDisconnectDB (p, NULL);
    return ERR_DB_ERROR;
  }
  else
    p->new_init = 1;

  if (tkvcpinit (&(p->bindvars.info.payment), p)) {
    TPCCDisconnectDB (p, NULL);
    return ERR_DB_ERROR;
  }
  else
    p->pay_init = 1;

  if (tkvcoinit (&(p->bindvars.info.orderStatus), p)) {
    TPCCDisconnectDB (p, NULL);
    return ERR_DB_ERROR;
  }
  else
    p->ord_init = 1;

  if (tkvcsinit (&(p->bindvars.info.stockLevel), p)) {
    TPCCDisconnectDB (p, NULL);
    return ERR_DB_ERROR;
  }
  else
    p->sto_init = 1;

  if (tkvcdinit (&(p->bindvars.info.delivery[0]), p)) {
    TPCCDisconnectDB (p, NULL);
    return ERR_DB_ERROR;
  }
  else
    p->del_init = 1;

  return ERR_DB_SUCCESS;
}


/* FUNCTION: int TPCCDisconnectDB(OraContext *dbproc)
 *
 * PURPOSE:            This function closes the sql connection.
 *
 * ARGUMENTS:
 *                     OraContext  *dbproc       pointer to OraContext
 *
 * RETURNS:            int         ERR_DB_SUCCESS      if successfull
 *                                 error value   if an error occurs
 *
 * COMMENTS:           None
 *
```

```
 */


int
TPCCDisconnectDB( DBContext DBC, pConnData pConn )
{
 OraContext *dbproc = (OraContext *)DBC;

  if (1 == dbproc->new_init) {
    tkvcndone(&(dbproc->nctx));
    dbproc->new_init = 0;
  }

  if (1 == dbproc->pay_init) {
    tkvcpdone(&(dbproc->pctx));
    dbproc->pay_init = 0;
  }
  if (1 == dbproc->ord_init) {
    tkvcodone(&(dbproc->octx));
    dbproc->ord_init = 0;
  }
  if (1 == dbproc->sto_init) {
    tkvcsdone(&(dbproc->sctx));
    dbproc->sto_init = 0;
  }
  if (1 == dbproc->del_init) {
    tkvcddone(&(dbproc->dctx));
    dbproc->del_init = 0;
  }

  OCIHandleFree((dvoid *)dbproc->tpcusr, OCI_HTYPE_SESSION);
  OCIHandleFree((dvoid *)dbproc->tpcsvc, OCI_HTYPE_SVCCTX);
  OCIHandleFree((dvoid *)dbproc->errhp, OCI_HTYPE_ERROR);
  OCIHandleFree((dvoid *)dbproc->datecvterrhp, OCI_HTYPE_ERROR);
  OCIHandleFree((dvoid *)dbproc->tpcsrv, OCI_HTYPE_SERVER);
  OCIHandleFree((dvoid *)dbproc->tpcenv, OCI_HTYPE_ENV);

#ifdef BATCH_DEL
  if (lfp) {
    fclose (lfp);
    lfp = NULL;
  }
#endif /* BATCH_DEL */


 return ERR_DB_SUCCESS;
}

/* FUNCTION: TPCCStockLevelDB(CallersContext               *pCC, int iTermId, int
iSyncId, OraContext *dbproc,  int deadlock_retry, StockLevelData *pStockLevel)
 *
 * PURPOSE:              This function handles the stock level transaction.
 *
 * ARGUMENTS:         CallersContext              *pCC
                passed in structure pointer from inetsrv.
 *                        int                         iTermId
                terminal id of browser
 *                        int                         iSyncId
                sync id of browser
 *                        OraContext  *dbproc
                connection db process id
 *                        StockLevelData              *pStockLevel
                stock level input / output data structure
 *                        int                         deadlock_retry
                retry count if deadlocked
 *
 * RETURNS:              int             ERR_DB_SUCCCESS    if successfull
 *                                       error value   if deadlocked
 *
 * COMMENTS:          None
 *
 */

int
TPCCStockLevelDB( DBContext DBC, pStockLevelData pStockLevel )
{
 OraContext *dbproc = (OraContext *)DBC;
 int tries,status;
 StockLevelData *pbindvars;
 pbindvars = &dbproc->bindvars.info.stockLevel;

 memcpy(pbindvars, pStockLevel, sizeof(StockLevelData));


 for ( tries = 0,status = RECOVERR;
                tries < DEADLOCKRETRIES && status == RECOVERR;  tries++) {

  status = tkvcs(dbproc);
 }

 pStockLevel->low_stock = dbproc->bindvars.info.stockLevel.low_stock;
 if (status == RECOVERR) return ERR_DB_DEADLOCK_LIMIT;
 else return (status);

}

/* FUNCTION: int TPCCNewOrderDB(CallersContext *pCC, int iTermId, int
iSyncId, int iTermId, int iSyncId, OraContext *dbproc, int deadlock_retry,
NewOrderData  *pNewOrder)
 *
 * PURPOSE:              This function handles the new order transaction.
 *
 * ARGUMENTS:         CallersContext              *pCC
                passed in structure pointer from inetsrv.
 *                        int                         iTermId
                terminal id of browser
 *                        int                         iSyncId
                sync id of browser
 *                        OraContext  *dbproc
                connection db process id
 *                        NewOrderData                *pNewOrder
                pointer to new order structure for input/output data
 *                        int                         deadlock_retry
                retry count if deadlocked
 *
 * RETURNS:              int             ERR_DB_SUCCESS
                transaction committed
 *                                       ERR_DB_NOT_COMMITTED    item number
is not valid
 *                                       ERR_DB_DEADLOCK_LIMIT        deadlock
max retry reached
 *                                       ERR_DB_ERROR
 *
 *
 * COMMENTS:          None
 *
 */

#pragma message ("FIXME: return code is overloaded.  How to report invalid item
number?")
int
TPCCNewOrderDB( DBContext DBC, pNewOrderData  pNewOrder )
{
 OraContext *dbproc = (OraContext *)DBC;
 int tries,status;
 int ii;
 int jj;
 int datebufsize;
 OCIError *datecvterrhp = dbproc->datecvterrhp;
 unsigned char localcr_date[7];
 int                                   temp;
 int                                   ol_i_id[MAX_OL];
 int                                   orig_pos[MAX_OL];

 NewOrderData *pbindvars = &(dbproc->bindvars.info.newOrder);
 newctx *nctx = &(dbproc->nctx);
 newtemp *ntemp = &(dbproc->tempvars.new);

/*  vgetdate(&ntemp->cr_date); */
 vgetdate(localcr_date);
 cvtdmyhms(localcr_date,ntemp->entry_date);
 OCIDateFromText(datecvterrhp,
                        (const unsigned char *)ntemp->entry_date,
                        strlen(ntemp->entry_date),
                        (const unsigned char *)"DD-MM-YYYY HH24:MI:SS",
                        21,(text *) 0,0,&ntemp->cr_date);

 ntemp->n_retry = 0;
```

```
 memcpy(pbindvars, pNewOrder, sizeof(NewOrderData));

 /* Oracle needs the items in assending order.  The item IDs are used */
 /* as a hierarchical locking scheme. Collect this order's list of item */
 /* IDs and record their original position. */
 for( jj = 0; jj < MAX_OL; jj++ )
 {
  ol_i_id[jj] = pNewOrder->o_ol[jj].ol_i_id;
  orig_pos[jj] = jj;
 }

 /* Swap-sort the arrays so that item IDs are in ascending order. */
 for( ii = 0; ii < pNewOrder->o_ol_cnt-1; ii++ )
 {
  for( jj = ii+1; jj < pNewOrder->o_ol_cnt; jj++ )
  {
   if( ol_i_id[ii] > ol_i_id[jj] )
   {
            /* If an item ID is to be swapped during the sort, perform a */
            /* corresponding swap in the original locations array. */
            temp = ol_i_id[ii];
            ol_i_id[ii] = ol_i_id[jj];
            ol_i_id[jj] = temp;
            temp = orig_pos[ii];
            orig_pos[ii] = orig_pos[jj];
            orig_pos[jj] = temp;
   }
  }
 }

 for(jj= 0; jj<MAX_OL; jj++)
  {
   /* Move the data from the input data structure into the bound */
   /* variables data structure using the ascending order of item IDs */
   /* generated above. */
   ntemp->nol_i_id[jj] = pbindvars->o_ol[orig_pos[jj]].ol_i_id;
   ntemp->nol_supply_w_id[jj] = pbindvars->o_ol[orig_pos[jj]].ol_supply_w_id;
   ntemp->nol_quantity[jj] = pbindvars->o_ol[orig_pos[jj]].ol_quantity;
  }

 for ( tries = 0,status = RECOVERR;
            tries < DEADLOCKRETRIES && status == RECOVERR; tries++)
  {
   status = tkvcn(&dbproc->bindvars.info.newOrder, dbproc);
  }

 memcpy(pNewOrder, pbindvars, sizeof(NewOrderData));

 /* convert and/or copy data to our structure format */
 pNewOrder->c_discount = ntemp->c_discount*100.0;
 pNewOrder->w_tax = (float)ntemp->w_tax*100.0;
 pNewOrder->d_tax = (float)ntemp->d_tax*100.0;


 for (ii = 0; ii < pNewOrder->o_ol_cnt; ii++)
  {
   /* return data from the bound and updated variables into the output */
   /* data structure using the original postions array generated above. */
   pNewOrder->o_ol[orig_pos[ii]].ol_i_id = ntemp->nol_i_id[ii];
   pNewOrder->o_ol[orig_pos[ii]].ol_supply_w_id = ntemp->nol_supply_w_id[ii];
   pNewOrder->o_ol[orig_pos[ii]].ol_quantity = ntemp->nol_quantity[ii];
   strncpy(pNewOrder->o_ol[orig_pos[ii]].i_name, ntemp->i_name[ii], 24);
   pNewOrder->o_ol[orig_pos[ii]].s_quantity = ntemp->s_quantity[ii];
   pNewOrder->o_ol[orig_pos[ii]].i_price = ntemp->i_price[ii]/100.0;
   pNewOrder->o_ol[orig_pos[ii]].ol_amount = ntemp->nol_amount[ii]/100.0;
   pNewOrder->o_ol[orig_pos[ii]].b_g[0]=ntemp->brand_generic[ii];
  }

 /* datebufsize = the size of entry_date in newtemp struct */
 datebufsize=21;
 /* datebufsize=sizeof(ntemp->entry_date); */
 /*  OCIDateToText(datecvterrhp, &ntemp->cr_date,(text *) "DD-MM-YYYY
 HH:MM:SS", 19, (text *) 0, 0, &datebufsize, &ntemp->entry_date); */
 /* cvtdmyhms(ntemp->cr_date, ntemp->entry_date);  */
 pNewOrder->o_entry_d.day = atoi(&(ntemp->entry_date[0]));
 pNewOrder->o_entry_d.month = atoi(&(ntemp->entry_date[3]));
 pNewOrder->o_entry_d.year = atoi(&(ntemp->entry_date[6]));
 pNewOrder->o_entry_d.hour = atoi(&(ntemp->entry_date[11]));
 pNewOrder->o_entry_d.minute = atoi(&(ntemp->entry_date[14]));
 pNewOrder->o_entry_d.second = atoi(&(ntemp->entry_date[17]));
 if (status == RECOVERR) return ERR_DB_DEADLOCK_LIMIT;
 else return (status);

}

/* FUNCTION: int TPCCPaymentDB(CallersContext *pCC, int iTermId, int iSyncId,
OraContext *dbproc, int deadlock_retry, PaymentData *pPayment)
 *
 * PURPOSE:          This function handles the payment transaction.
 *
 * ARGUMENTS:        CallersContext          *pCC
 *                   passed in structure pointer from inetsrv.
 *                   int                     iTermId
 *                   terminal id of browser
 *                   int                     iSyncId
 *                   sync id of browser
 *                   OraContext  *dbproc
 *                   connection db process id
 *                   PaymentData             *pPayment          pointer to
payment input/output data structure
 *                   int                     deadlock_retry
 *                   deadlock retry count
 *
 * RETURNS:          int         ERR_DB_SUCCESS              success
 *                               ERR_DB_DEADLOCK_LIMIT       max
deadlocked reached
 *                               ERR_DB_NOT_COMMITED         invalid
data entry
 *
 * COMMENTS:         None
 *
 */

int
TPCCPaymentDB( DBContext DBC, pPaymentData pPayment )
{
 OraContext *dbproc = (OraContext *)DBC;
 int tries;
 int status;
 unsigned int datebufsize;
 OCIError *datecvterrhp = dbproc->datecvterrhp;

 PaymentData *pbindvars = &(dbproc->bindvars.info.payment);
 payctx *pctx = &(dbproc->pctx);
 paytemp *ptemp = &(dbproc->tempvars.pay);

 ptemp->p_retry = 0;

 memcpy(pbindvars, pPayment, sizeof(PaymentData));

 /* the db is stored in pennies - convert input to cents. */
 ptemp->h_amount = (int)(pbindvars->h_amount*100);

 for ( tries = 0,status = RECOVERR;
            tries < DEADLOCKRETRIES && status == RECOVERR; tries++) {

  if ((pbindvars->c_id) == 0) {
   (pbindvars->byname) = TRUE;
  }
  else {
   (pbindvars->byname) = FALSE;
  }

  status = tkvcp(&dbproc->bindvars.info.payment, dbproc);

 }

 memcpy(pPayment, pbindvars, sizeof(PaymentData));
 /* datebufsize = the size of c_since_str in paytemp struct */
 datebufsize=sizeof(ptemp->c_since_str);
 /* convert date format */
 /* OCIDateToText(datecvterr, &ptemp->customer_sdate,(text *) 0, 10, (text *) 0, 0,
&datebufsize, &ptemp->c_since_str); */
 OCIDateToText(datecvterrhp, &ptemp->customer_sdate,(text *)
DATE_FORMAT_STR, 10, (text *) 0, 0, &datebufsize, (text *)&ptemp->c_since_str);
 /* cvtdmy(ptemp->customer_sdate, ptemp->c_since_str); */
 /* datebufsize = the size of h_date string in paytemp struct */
 /* datebufsize=DATE_TIME_SIZ; */
```

```
  /* OCIDateToText(datecvterrhp, &ptemp->cr_date,(text *) "DD-MM-
YYYY.HH24:MI:SS", 21, (text *) 0, 0, &datebufsize, &ptemp->h_date); */
  pPayment->c_credit_lim = ptemp->c_credit_lim/100.0;
  pPayment->c_discount = ptemp->c_discount*100.0;
  pPayment->c_balance = pPayment->c_balance/100.0;
  pPayment->h_amount = ptemp->h_amount/100.0;

  pPayment->c_since.day = atoi(&(ptemp->c_since_str[0]));
  pPayment->c_since.month = atoi(&(ptemp->c_since_str[3]));
  pPayment->c_since.year = atoi(&(ptemp->c_since_str[6]));
  pPayment->h_date.day = atoi(&(ptemp->h_date[0]));
  pPayment->h_date.month = atoi(&(ptemp->h_date[3]));
  pPayment->h_date.year = atoi(&(ptemp->h_date[6]));
  pPayment->h_date.hour = atoi(&(ptemp->h_date[11]));
  pPayment->h_date.minute = atoi(&(ptemp->h_date[14]));
  pPayment->h_date.second = atoi(&(ptemp->h_date[17]));

  if (status == RECOVERR) return ERR_DB_DEADLOCK_LIMIT;
  else return (status);

}


/* FUNCTION: int TPCCOrderStatusDB(CallersContext *pCC, int iTermId, int
iSyncId, OraContext *dbproc, int deadlock_retry, OrderStatusData *pOrderStatus)
 *
 * PURPOSE:              This function processes the Order Status transaction.
 *
 * ARGUMENTS:            CallersContext          *pCC
 *                       passed in structure pointer from inetsrv.
 *                       int                     iTermId
 *                       terminal id of browser
 *                       int                     iSyncId
 *                       sync id of browser
 *                       OraContext  *dbproc
 *                       connection db process id
 *                       OrderStatusData         *pOrderStatus
 *                       pointer to Order Status data input/output structure
 *                       int                     deadlock_retry
 *                       deadlock retry count
 *
 * RETURNS:              int           ERR_DB_DEADLOCK_LIMIT
 *                       max deadlock reached
 *                                     ERR_DB_NOT_COMMITED
 *                       No orders found for customer
 *                                     ERR_DB_SUCCESS
 *                       Transaction successfull
 *
 * COMMENTS:             None
 *
 */

int
TPCCOrderStatusDB( DBContext DBC, pOrderStatusData pOrderStatus )
{
  OraContext *dbproc = (OraContext *)DBC;
  int tries,status;
  int ii;
  OrderStatusData *pbindvars = &(dbproc->bindvars.info.orderStatus);
  ordtemp *otemp = &(dbproc->tempvars.ord);
  OCIError *datecvterrhp = dbproc->datecvterrhp;

  memcpy(pbindvars, pOrderStatus, sizeof(OrderStatusData));

  for ( tries = 0,status = RECOVERR;
     tries < DEADLOCKRETRIES && status == RECOVERR; tries++) {

    if ((pbindvars->c_id) == 0) {
      (pbindvars->byname) = TRUE;
    }
    else {
      (pbindvars->byname) = FALSE;
    }

    status = tkvco(&dbproc->bindvars.info.orderStatus, dbproc);

  }
  if (status == ERR_DB_ERROR) return status;
  memcpy(pOrderStatus,pbindvars, sizeof(OrderStatusData));

  for (ii=0; ii < pOrderStatus->o_ol_cnt; ii++)
   {
     pOrderStatus->s_ol[ii].ol_supply_w_id = otemp->loc_ol_supply_w_id[ii];
     pOrderStatus->s_ol[ii].ol_i_id = otemp->loc_ol_i_id[ii];
     pOrderStatus->s_ol[ii].ol_quantity = otemp->loc_ol_quantity[ii];
     pOrderStatus->s_ol[ii].ol_amount = otemp->loc_ol_amount[ii]/100.0;
     pOrderStatus->s_ol[ii].ol_delivery_d.day =
                atoi(&(otemp->ol_delivery_date_str[ii][0]));
     pOrderStatus->s_ol[ii].ol_delivery_d.month =
                atoi(&(otemp->ol_delivery_date_str[ii][3]));
     pOrderStatus->s_ol[ii].ol_delivery_d.year =
                atoi(&(otemp->ol_delivery_date_str[ii][6]));
   };


  pOrderStatus->c_balance = pOrderStatus->c_balance/100.0;
  pOrderStatus->o_entry_d.day = atoi(&(otemp->entry_date_str[0]));
  pOrderStatus->o_entry_d.month = atoi(&(otemp->entry_date_str[3]));
  pOrderStatus->o_entry_d.year = atoi(&(otemp->entry_date_str[6]));
  pOrderStatus->o_entry_d.hour = atoi(&(otemp->entry_date_str[11]));
  pOrderStatus->o_entry_d.minute = atoi(&(otemp->entry_date_str[14]));
  pOrderStatus->o_entry_d.second = atoi(&(otemp->entry_date_str[17]));

  if (status == RECOVERR) return ERR_DB_DEADLOCK_LIMIT;
  else return (status);

}



/* FUNCTION: int TPCCDeliveryDB( CallersContext *pCC, int iConnectionID,
   * int iSyncID, DBContext *pdbContext,
   * int deadlock_retry, pDeliveryData pDelivery )
   *
   * PURPOSE:            This function writes the delivery information to the
   *          delivery pipe. The information is sent as a long.
   *
   * ARGUMENTS:          CallersContext          *pCC                    passed
in structure
   *                          pointer from inetsrv.
   *                     int                     iTermId
terminal id of browser
   *                     int                     iSyncId                 sync id
of browser
   *                     OraContext  *dbproc                 connection db
process id
   *                     int                     deadlock_retry
deadlock retry count
   *                     DeliveryData            *pDelivery      pointer to Delivery
data
   *                          input/output structure
   *
   * RETURNS:            int           ERR_DB_SUCCESS                success
   *                                   ERR_DB_DEADLOCK_LIMIT         max
deadlocked reached
   *                                   ERR_DB_NOT_COMMITED           other
error
   *
   * COMMENTS:           The pipe is initially created with 16K buffer size this
   *          should allow for up to 4096 deliveries
   *                     to be queued before an overflow condition would occur.
   *                     The only reason that an overflow would occur is if the
delivery
   *                     application stopped listening while deliveries were being
   *          posted.
   *
   */

int
TPCCDeliveryDB( DBContext DBC, pDeliveryData pDeliveryData )
{
  OraContext *dbproc = (OraContext *)DBC;
  int retries = 0;
  int status;
  DeliveryData *pbindvars;

  pbindvars = &dbproc->bindvars.info.delivery[0];
  memcpy(pbindvars, pDeliveryData, sizeof(DeliveryData));

  for (retries = 0, status = RECOVERR;
```

```
    retries < DEADLOCKRETRIES &&status == RECOVERR; retries++){

  status = tkvcd(pDeliveryData, dbproc);
 }

 if(status == RECOVERR) return ERR_DB_DEADLOCK_LIMIT;
 else return (status);


}

/* FUNCTION: int TPCCCheckpointDB()
 *
 * PURPOSE:              This function does a checkpoinnt transaction.
 *
 * ARGUMENTS:
 * RETURNS:              int             ERR_DB_DEADLOCK_LIMIT       max
deadlock reached
 *                                       ERR_DB_NOT_COMMITED         No orders
found for customer
 *                                       ERR_DB_SUCCESS
            Transaction successfull
 *
 * COMMENTS:             None
 *
 */


#define CHECKPOINT_TXT "alter system checkpoint local"

int
TPCCCheckpointDB( DBContext DBC, pCheckpointData pCheckpoint )
{
 OraContext *dbproc = (OraContext *)DBC;
 text stmbuf[100];

 OCIHandleAlloc(dbproc->tpcenv, (dvoid **)&(dbproc->curi), OCI_HTYPE_STMT,
                0, (dvoid**)0);
 sprintf ((char *) stmbuf, CHECKPOINT_TXT);
 OCIERROR(dbproc, OCIStmtPrepare(dbproc->curi, dbproc->errhp, stmbuf,
                                 strlen((char *)stmbuf),
                                 OCI_NTV_SYNTAX,
OCI_DEFAULT));
 if (RECOVERR != OCIERROR(dbproc,
                          OCIStmtExecute(dbproc->tpcsvc, dbproc-
>curi,
                                         dbproc->errhp, 1, 0,
0, 0,
                                         OCI_DEFAULT)))
   return (ERR_DB_ERROR);
 OCIHandleFree(dbproc->curi, OCI_HTYPE_STMT);

 return ERR_DB_SUCCESS;

}
```

oracle_db8.h

```
/*+ file: oracle_db8.h based on Oracle file tpccpl.h
/*+=========================================================
=======+
|      Copyright (c) 1994  Oracle Corp, Redwood Shores, CA      |
|              OPEN SYSTEMS PERFORMANCE GROUP                   |
|                  All Rights Reserved                          |

+=========================================================
======+
| DESCRIPTION
|   header file for the TPC-C transactions.

+=========================================================
====-*/
/*+*****************************************************************
******-*/
/*+                                                    -*/
/*+  COPYRIGHT (c) 1998, 2000 BY                       -*/
/*+  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
-*/
/*+  ALL RIGHTS RESERVED.                                    -*/
/*+                                                    -*/
/*+  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE
USED AND COPIED  -*/
/*+  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND
WITH THE  -*/
/*+  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY  OTHER  -*/
/*+  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY  -*/
/*+  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY  -*/
/*+  TRANSFERRED.                                       -*/
/*+                                                    -*/
/*+  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE  -*/
/*+  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY
DIGITAL EQUIPMENT  -*/
/*+  CORPORATION.                                       -*/
/*+                                                    -*/
/*+  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS  -*/
/*+  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
-*/
/*+                                                    -*/
/*+                                                    -*/
/*****************************************************************
******-*/
#ifndef ORACLE_DB_H
#define ORACLE_DB_H



#ifndef DISCARD
# define DISCARD (void)
#endif

#ifndef sword
# define sword int
#endif

#define VER7          2

#define NA            -1     /* ANSI SQL NULL */
#define NLT           1      /* length for string null terminator */
#define DEADLOCK      60     /* ORA-00060: deadlock */
#define NO_DATA_FOUND  1403  /* ORA-01403: no data found */
#define NOT_SERIALIZABLE  8177  /* ORA-08177: transaction not serializable */
#define SNAPSHOT_TOO_OLD  1555  /* ORA-01555: snapshot too old */


#define RECOVERR -10
#define IRRECERR -20
#define NO_COMMIT -30
#define NOERR 111

#define DEADLOCKWAIT 10


#if (defined(__osf__) && defined(__alpha))
#define HDA_SIZ 512
#else
#define HDA_SIZ 256
#endif

#define DATE_FORMAT_STR "DD-MM-YYYY"                /* includes null
terminator */
#define DATE_SIZ sizeof(DATE_FORMAT_STR)
#define DATE_TIME_FORMAT_STR "DD-MM-YYYY HH24:MI:SS"     /*
includes null */
#define DATE_TIME_SIZ sizeof("DD-MM-YYYY HH:MI:SS")
#define MSG_SIZ 512
#define NITEMS 15
#define NDISTS 10
#define ROWIDLEN 20
#define OCIROWLEN 20
#define DEL_DATE_LEN 7
#define SQL_BUF_SIZE 16384
```

```
#ifndef NULLP
# define NULLP (void *)NULL
#endif /* NULLP */

/* global variables */
#ifdef ORACLE_DB_C
#else
#endif /* ORACLE_DB_C */


#define ADR(object) ((ub1 *)&(object))
#define SIZ(object) ((sword)sizeof(object))

typedef char date[24+NLT];
typedef char varchar2;


/* anonymous block statement buffers */
extern text stmbuf_payz[SQL_BUF_SIZE];
extern text stmbuf_paynz[SQL_BUF_SIZE];
extern text stmbuf_tkvcpnew[SQL_BUF_SIZE];

struct _delctx {
  sb2 cons_ind[NDISTS];
  sb2 w_id_ind[NDISTS];
  sb2 d_id_ind[NDISTS];
  sb2 c_id_ind[NDISTS];
  sb2 del_o_id_ind[NDISTS];
  sb2 del_date_ind[NDISTS];
  sb2 carrier_id_ind[NDISTS];
  sb2 amt_ind[NDISTS];
  sb2 no_rowid_ind[NDISTS];
  sb2 o_rowid_ind[NDISTS];
#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
  sb2 inum_ind;
#endif

  OCIBind *olamt_bp;

  ub2 cons_len[NDISTS];
  ub2 w_id_len[NDISTS];
  ub2 d_id_len[NDISTS];
  ub4 c_id_len[NDISTS];
  ub4 del_o_id_len[NDISTS];
  ub2 del_date_len[NDISTS];
  ub2 carrier_id_len[NDISTS];
  ub2 amt_len[NDISTS];
  ub2 no_rowid_len[NDISTS];
  ub2 no_rowid_ptr_len[NDISTS];
  ub2 o_rowid_len[NDISTS];
  ub2 o_rowid_ptr_len[NDISTS];
#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
  ub2 inum_len;
#endif

  ub2 cons_rcode[NDISTS];
  ub2 w_id_rcode[NDISTS];
  ub2 d_id_rcode[NDISTS];
  ub2 c_id_rcode[NDISTS];
  ub2 del_o_id_rcode[NDISTS];
  ub2 del_date_rcode[NDISTS];
  ub2 carrier_id_rcode[NDISTS];
  ub2 amt_rcode[NDISTS];
  ub2 no_rowid_rcode[NDISTS];
  ub2 o_rowid_rcode[NDISTS];
#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
  ub2 inum_rcode;
#endif

  int cons[NDISTS];
  int w_id[NDISTS];
  int d_id[NDISTS];
  int c_id[NDISTS];
  int del_o_id[NDISTS];
  int carrier_id[NDISTS];
  /* float amt[NDISTS]; Changed to int */
  int amt[NDISTS];
  ub4 del_o_id_rcnt;
  OCIRowid *no_rowid_ptr[NDISTS];
  OCIRowid *o_rowid_ptr[NDISTS];
  OCIDate del_date[NDISTS];
#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
  char inum[10];
#endif
  OCIStmt *curd0;
  OCIStmt *curd1;
  OCIStmt *curd2;
  OCIStmt *curd3;
  OCIStmt *curd4;
  OCIStmt *curd5;
  OCIStmt *curd6;
  OCIStmt *curdtest;

  OCIBind *w_id_bp;
  OCIBind *w_id_bp3;
  OCIBind *w_id_bp4;
  OCIBind *w_id_bp5;
  OCIBind *w_id_bp6;
  OCIBind *d_id_bp;
  OCIBind *d_id_bp3;
  OCIBind *d_id_bp4;
  OCIBind *d_id_bp6;
  OCIBind *o_id_bp;
  OCIBind *cr_date_bp;
  OCIBind *c_id_bp;
  OCIBind *c_id_bp3;
  OCIBind *no_rowid_bp;
  OCIBind *carrier_id_bp;
  OCIBind *o_rowid_bp;
  OCIBind *del_o_id_bp;
  OCIBind *del_o_id_bp3;
  OCIBind *amt_bp;
  OCIBind *bstr1_bp[10];
  OCIBind *bstr2_bp[10];
  OCIDefine *inum_dp;
  OCIDefine *d_id_dp;
  OCIDefine *del_o_id_dp;
  OCIDefine *no_rowid_dp;
  OCIDefine *c_id_dp;
  OCIDefine *o_rowid_dp;
  OCIDefine *cons_dp;
  OCIDefine *amt_dp;

  int norow;
};
typedef struct _delctx delctx;


struct _amtctx {
  int ol_amt[NDISTS][NITEMS];
  sb2 ol_amt_ind[NDISTS][NITEMS];
  ub4 ol_amt_len[NDISTS][NITEMS];
  ub2 ol_amt_rcode[NDISTS][NITEMS];
  int ol_cnt[NDISTS];
};
typedef struct _amtctx amtctx;



struct _newctx {
  sb2 nol_i_id_ind[NITEMS];
  sb2 nol_supply_w_id_ind[NITEMS];
  sb2 nol_quantity_ind[NITEMS];
  sb2 nol_amount_ind[NITEMS];
  sb2 i_name_ind[NITEMS];
  sb2 s_quantity_ind[NITEMS];
  sb2 i_price_ind[NITEMS];
  sb2 ol_w_id_ind[NITEMS];
  sb2 ol_d_id_ind[NITEMS];
  sb2 ol_o_id_ind[NITEMS];
  sb2 ol_number_ind[NITEMS];
  sb2 cons_ind[NITEMS];
  sb2 s_rowid_ind[NITEMS];
  sb2 s_remote_ind[NITEMS];
  sb2 s_quant_ind[NITEMS];
  sb2 i_data_ind[NITEMS];
  sb2 s_data_ind[NITEMS];
  sb2 s_dist_info_ind[NITEMS];
  sb2 ol_dist_info_ind[NITEMS];
```

```
  sb2 null_date_ind[NITEMS];
  sb2 s_bg_ind[NITEMS];

  ub2 nol_i_id_len[NITEMS];
  ub2 nol_supply_w_id_len[NITEMS];
  ub2 nol_quantity_len[NITEMS];
  ub2 nol_amount_len[NITEMS];
  ub2 i_name_len[NITEMS];
  ub2 s_quantity_len[NITEMS];
  ub2 i_price_len[NITEMS];
  ub2 ol_w_id_len[NITEMS];
  ub2 ol_d_id_len[NITEMS];
  ub2 ol_o_id_len[NITEMS];
  ub2 ol_number_len[NITEMS];
  ub2 cons_len[NITEMS];
  ub2 s_rowid_len[NITEMS];
  ub2 s_remote_len[NITEMS];
  ub2 s_quant_len[NITEMS];
  ub2 i_data_len[NITEMS];
  ub2 s_data_len[NITEMS];
  ub2 s_dist_info_len[NITEMS];
  ub2 ol_dist_info_len[NITEMS];
  ub2 null_date_len[NITEMS];
  sb2 s_bg_len[NITEMS];

  ub2 nol_i_id_rcode[NITEMS];
  ub2 nol_supply_w_id_rcode[NITEMS];
  ub2 nol_quantity_rcode[NITEMS];
  ub2 nol_amount_rcode[NITEMS];
  ub2 i_name_rcode[NITEMS];
  ub2 s_quantity_rcode[NITEMS];
  ub2 i_price_rcode[NITEMS];
  ub2 ol_w_id_rcode[NITEMS];
  ub2 ol_d_id_rcode[NITEMS];
  ub2 ol_o_id_rcode[NITEMS];
  ub2 ol_number_rcode[NITEMS];
  ub2 cons_rcode[NITEMS];
  ub2 s_rowid_rcode[NITEMS];
  ub2 s_remote_rcode[NITEMS];
  ub2 s_quant_rcode[NITEMS];
  ub2 i_data_rcode[NITEMS];
  ub2 s_data_rcode[NITEMS];
  ub2 s_dist_info_rcode[NITEMS];
  ub2 ol_dist_info_rcode[NITEMS];
  ub2 null_date_rcode[NITEMS];
  sb2 s_bg_rcode[NITEMS];

  int ol_w_id[NITEMS];
  int ol_d_id[NITEMS];
  int ol_o_id[NITEMS];
  int ol_number[NITEMS];
  int cons[NITEMS];

  OCIRowid  *s_rowid_ptr[NITEMS];

  int s_remote[NITEMS];
  char i_data[NITEMS][51];
  char s_data[NITEMS][51];
  char s_dist_info[NITEMS][25];

/* changed to OCIDate  unsigned char null_date[NITEMS][7]; */
  OCIDate null_date[NITEMS]; /* base date for null date entry */
/* not needed for Oracle 8.1.5   OCIStmt *curn; */
  OCIStmt *curn1;
  OCIStmt *curn2;
  OCIStmt *curn3[10];
  OCIStmt *curn4;
  OCIBind *i_price_bp;
  OCIBind *i_name_bp;
  OCIBind *s_bg_bp;
  OCIBind *s_data_bp;
  OCIBind *i_data_bp;
  ub4 nol_i_count;
  ub4 nol_s_count;
  ub4 nol_q_count;
  ub4 nol_item_count;
  ub4 nol_name_count;
  ub4 nol_qty_count;
  ub4 nol_bg_count;
  ub4 nol_am_count;
  ub4 s_remote_count;
  ub4 s_data_count;
  ub4 i_data_count;
  OCIBind *ol_i_id_bp4;
  OCIBind *ol_supply_w_id_bp4;
  OCIBind *ol_quantity_bp4;
  OCIBind *w_id_bp;
  OCIBind *d_id_bp;
  OCIBind *c_id_bp;
  OCIBind *o_all_local_bp;
  OCIBind *o_all_cnt_bp;
  OCIBind *w_tax_bp;
  OCIBind *d_tax_bp;
  OCIBind *o_id_bp;
  OCIBind *c_discount_bp;
  OCIBind *c_credit_bp;
  OCIBind *c_last_bp;
  OCIBind *retries_bp;
  OCIBind *cr_date_bp;
  OCIBind *ol_i_id_bp;
  OCIBind *ol_supply_w_id_bp;
  OCIBind *s_quantity_bp;
  OCIBind *s_rowid_bp;
  OCIBind *ol_quantity_bp;
  OCIBind *s_remote_bp;
  OCIBind *id_bp[10][15];
  OCIBind *sd_bp[10][15];
  OCIDefine *Dcons[10];
  OCIDefine *Ds_rowid[10];
  OCIDefine *Di_price[10];
  OCIDefine *Di_data[10];
  OCIDefine *Ds_dist_info[10];
  OCIDefine *Ds_data[10];
  OCIDefine *Ds_quantity[10];
  OCIDefine *Di_name[10];
  OCIBind *ol_o_id_bp;
  OCIBind *ol_d_id_bp;
  OCIBind *ol_w_id_bp;
  OCIBind *ol_number_bp;
  OCIBind *ol_amount_bp;
  OCIBind *ol_dist_info_bp;
  OCIBind *null_date_bp;
  sb2 w_id_ind;
  ub2 w_id_len;
  ub2 w_id_rc;

  sb2 d_id_ind;
  ub2 d_id_len;
  ub2 d_id_rc;

  sb2 c_id_ind;
  ub2 c_id_len;
  ub2 c_id_rc;

  sb2 o_all_local_ind;
  ub2 o_all_local_len;
  ub2 o_all_local_rc;

  sb2 o_ol_cnt_ind;
  ub2 o_ol_cnt_len;
  ub2 o_ol_cnt_rc;

  sb2 w_tax_ind;
  ub2 w_tax_len;
  ub2 w_tax_rc;

  sb2 d_tax_ind;
  ub2 d_tax_len;
  ub2 d_tax_rc;

  sb2 o_id_ind;
  ub2 o_id_len;
  ub2 o_id_rc;

  sb2 c_discount_ind;
  ub2 c_discount_len;
  ub2 c_discount_rc;

  sb2 c_credit_ind;
  ub2 c_credit_len;
```

```
  ub2 c_credit_rc;

  sb2 c_last_ind;
  ub2 c_last_len;
  ub2 c_last_rc;

  sb2 retries_ind;
  ub2 retries_len;
  ub2 retries_rc;

  sb2 cr_date_ind;
  ub2 cr_date_len;
  ub2 cr_date_rc;


  int cs;
  int norow;

/* context holders */
int i_name_ctx;
int i_data_ctx;
int i_price_ctx;
int s_data_ctx;
int s_dist_info_ctx;
int s_quantity_ctx;

};
typedef struct _newctx newctx;

struct _ordctx {
   sb2 c_rowid_ind[100];
   sb2 ol_supply_w_id_ind[NITEMS];
   sb2 ol_i_id_ind[NITEMS];
   sb2 ol_quantity_ind[NITEMS];
   sb2 ol_amount_ind[NITEMS];
   sb2 ol_delivery_d_ind[NITEMS];
   sb2 ol_w_id_ind;
   sb2 ol_d_id_ind;
   sb2 ol_o_id_ind;
   sb2 c_id_ind;
   sb2 c_first_ind;
   sb2 c_middle_ind;
   sb2 c_balance_ind;
   sb2 c_last_ind;
   sb2 o_id_ind;
   sb2 o_entry_d_ind;
   sb2 o_carrier_id_ind;
   sb2 o_ol_cnt_ind;

   ub4 c_rowid_len[100];
   ub2 ol_supply_w_id_len[NITEMS];
   ub2 ol_i_id_len[NITEMS];
   ub2 ol_quantity_len[NITEMS];
   ub2 ol_amount_len[NITEMS];
   ub2 ol_delivery_d_len[NITEMS];
   ub2 ol_w_id_len;
   ub2 ol_d_id_len;
   ub2 ol_o_id_len;

   ub2 c_rowid_rcode[100];
   ub2 ol_supply_w_id_rcode[NITEMS];
   ub2 ol_i_id_rcode[NITEMS];
   ub2 ol_quantity_rcode[NITEMS];
   ub2 ol_amount_rcode[NITEMS];
   ub2 ol_delivery_d_rcode[NITEMS];
   ub2 ol_w_id_rcode;
   ub2 ol_d_id_rcode;
   ub2 ol_o_id_rcode;

   ub4 ol_supply_w_id_csize;
   ub4 ol_i_id_csize;
   ub4 ol_quantity_csize;
   ub4 ol_amount_csize;
   ub4 ol_delivery_d_csize;
   ub4 ol_w_id_csize;
   ub4 ol_d_id_csize;
   ub4 ol_o_id_csize;

   OCIStmt *curo0;
   OCIStmt *curo1;
   OCIStmt *curo2;
   OCIStmt *curo3;
   OCIStmt *curo4;
   OCIBind *w_id_bp0;
   OCIBind *w_id_bp2;
   OCIBind *w_id_bp3;
   OCIBind *w_id_bp4;
   OCIBind *d_id_bp0;
   OCIBind *d_id_bp2;
   OCIBind *d_id_bp3;
   OCIBind *d_id_bp4;
   OCIBind *c_id_bp;
   OCIBind *byln_bp;
   OCIBind *c_last_bp;
   OCIBind *c_last_bp4;
   OCIBind *c_first_bp;
   OCIBind *c_middle_bp;
   OCIBind *c_balance_bp;
   OCIBind *o_id_bp;
   OCIBind *o_entry_d_bp;
   OCIBind *o_cr_id_bp;
   OCIBind *o_ol_cnt_bp;
   OCIBind *ol_s_w_id_bp;
   OCIBind *ol_i_id_bp;
   OCIBind *ol_quantity_bp;
   OCIBind *ol_amount_bp;
   OCIBind *ol_d_d_bp;
   OCIBind *c_rowid_bp;
   OCIDefine *c_rowid_dp;
   OCIDefine *c_last_dp;
   OCIDefine *c_last_dp1;
   OCIDefine *c_id_dp;
   OCIDefine *c_id_dp1;
   OCIDefine *c_first_dp1;
   OCIDefine *c_first_dp2;
   OCIDefine *c_middle_dp1;
   OCIDefine *c_middle_dp2;
   OCIDefine *c_balance_dp1;
   OCIDefine *c_balance_dp2;
   OCIDefine *o_id_dp1;
   OCIDefine *o_id_dp2;
   OCIDefine *o_entry_d_dp1;
   OCIDefine *o_entry_d_dp2;
   OCIDefine *o_cr_id_dp1;
   OCIDefine *o_cr_id_dp2;
   OCIDefine *o_ol_cnt_dp1;
   OCIDefine *o_ol_cnt_dp2;
   OCIDefine *ol_d_d_dp;
   OCIDefine *ol_i_id_dp;
   OCIDefine *ol_supply_w_id_dp;
   OCIDefine *ol_quantity_dp;
   OCIDefine *ol_amount_dp;
   OCIDefine *ol_d_base_dp;
   OCIDefine *c_count_dp;

   OCIRowid *c_rowid_ptr[100];
   OCIRowid *middle_cust;
   int cs;
   int cust_idx;
   int norow;
   int rcount;
   int somerows;
};
typedef struct _ordctx ordctx;


struct _defctx {
  boolean reexec;
  ub4 count;
};
typedef struct _defctx defctx;

struct _payctx {
  OCIStmt *curpi;
  OCIStmt *curp0;
  OCIStmt *curp1;
  OCIBind *w_id_bp;
  OCIBind *w_id_bp1;
  sb2 w_id_ind;
  ub2 w_id_len;
```

```
ub2 w_id_rc;

OCIBind *d_id_bp;
OCIBind *d_id_bp1;
sb2 d_id_ind;
ub2 d_id_len;
ub2 d_id_rc;

OCIBind *c_w_id_bp;
OCIBind *c_w_id_bp1;
sb2 c_w_id_ind;
ub2 c_w_id_len;
ub2 c_w_id_rc;

OCIBind *c_d_id_bp;
OCIBind *c_d_id_bp1;
sb2 c_d_id_ind;
ub2 c_d_id_len;
ub2 c_d_id_rc;

OCIBind *c_id_bp;
OCIBind *c_id_bp1;
sb2 c_id_ind;
ub2 c_id_len;
ub2 c_id_rc;

OCIBind *h_amount_bp;
OCIBind *h_amount_bp1;
sb2 h_amount_ind;
ub2 h_amount_len;
ub2 h_amount_rc;

OCIBind *c_last_bp;
OCIBind *c_last_bp1;
sb2 c_last_ind;
ub2 c_last_len;
ub2 c_last_rc;

OCIBind *w_street_1_bp;
OCIBind *w_street_1_bp1;
sb2 w_street_1_ind;
ub2 w_street_1_len;
ub2 w_street_1_rc;

OCIBind *w_street_2_bp;
OCIBind *w_street_2_bp1;
sb2 w_street_2_ind;
ub2 w_street_2_len;
ub2 w_street_2_rc;

OCIBind *w_city_bp;
OCIBind *w_city_bp1;
sb2 w_city_ind;
ub2 w_city_len;
ub2 w_city_rc;

OCIBind *w_state_bp;
OCIBind *w_state_bp1;
sb2 w_state_ind;
ub2 w_state_len;
ub2 w_state_rc;

OCIBind *w_zip_bp;
OCIBind *w_zip_bp1;
sb2 w_zip_ind;
ub2 w_zip_len;
ub2 w_zip_rc;

OCIBind *d_street_1_bp;
OCIBind *d_street_1_bp1;
sb2 d_street_1_ind;
ub2 d_street_1_len;
ub2 d_street_1_rc;

OCIBind *d_street_2_bp;
OCIBind *d_street_2_bp1;
sb2 d_street_2_ind;
ub2 d_street_2_len;
ub2 d_street_2_rc;

OCIBind *d_city_bp;
OCIBind *d_city_bp1;
sb2 d_city_ind;
ub2 d_city_len;
ub2 d_city_rc;

OCIBind *d_state_bp;
OCIBind *d_state_bp1;
sb2 d_state_ind;
ub2 d_state_len;
ub2 d_state_rc;

OCIBind *d_zip_bp;
OCIBind *d_zip_bp1;
sb2 d_zip_ind;
ub2 d_zip_len;
ub2 d_zip_rc;

OCIBind *c_first_bp;
OCIBind *c_first_bp1;
sb2 c_first_ind;
ub2 c_first_len;
ub2 c_first_rc;

OCIBind *c_middle_bp;
OCIBind *c_middle_bp1;
sb2 c_middle_ind;
ub2 c_middle_len;
ub2 c_middle_rc;

OCIBind *c_street_1_bp;
OCIBind *c_street_1_bp1;
sb2 c_street_1_ind;
ub2 c_street_1_len;
ub2 c_street_1_rc;

OCIBind *c_street_2_bp;
OCIBind *c_street_2_bp1;
sb2 c_street_2_ind;
ub2 c_street_2_len;
ub2 c_street_2_rc;

OCIBind *c_city_bp;
OCIBind *c_city_bp1;
sb2 c_city_ind;
ub2 c_city_len;
ub2 c_city_rc;

OCIBind *c_state_bp;
OCIBind *c_state_bp1;
sb2 c_state_ind;
ub2 c_state_len;
ub2 c_state_rc;

OCIBind *c_zip_bp;
OCIBind *c_zip_bp1;
sb2 c_zip_ind;
ub2 c_zip_len;
ub2 c_zip_rc;

OCIBind *c_phone_bp;
OCIBind *c_phone_bp1;
sb2 c_phone_ind;
ub2 c_phone_len;
ub2 c_phone_rc;

OCIBind *c_since_bp;
OCIBind *c_since_bp1;
sb2 c_since_ind;
ub2 c_since_len;
ub2 c_since_rc;

OCIBind *c_credit_bp;
OCIBind *c_credit_bp1;
sb2 c_credit_ind;
ub2 c_credit_len;
ub2 c_credit_rc;

OCIBind *c_credit_lim_bp;
OCIBind *c_credit_lim_bp1;
```

```
  sb2 c_credit_lim_ind;
  ub2 c_credit_lim_len;
  ub2 c_credit_lim_rc;

  OCIBind *c_discount_bp;
  OCIBind *c_discount_bp1;
  sb2 c_discount_ind;
  ub2 c_discount_len;
  ub2 c_discount_rc;

  OCIBind *c_balance_bp;
  OCIBind *c_balance_bp1;
  sb2 c_balance_ind;
  ub2 c_balance_len;
  ub2 c_balance_rc;

  OCIBind *c_data_bp;
  OCIBind *c_data_bp1;
  sb2 c_data_ind;
  ub2 c_data_len;
  ub2 c_data_rc;

  OCIBind *h_date_bp;
  OCIBind *h_date_bp1;
  sb2 h_date_ind;
  ub2 h_date_len;
  ub2 h_date_rc;

  OCIBind *retries_bp;
  OCIBind *retries_bp1;
  sb2 retries_ind;
  ub2 retries_len;
  ub2 retries_rc;

  OCIBind *cr_date_bp;
  OCIBind *cr_date_bp1;
  sb2 cr_date_ind;
  ub2 cr_date_len;
  ub2 cr_date_rc;

  OCIBind *byln_bp;
  sb2 byln_ind;
  ub2 byln_len;
  ub2 byln_rc;
};
typedef struct _payctx payctx;

struct _stoctx {
  OCIStmt *curs;
  OCIBind *w_id_bp;
  OCIBind *d_id_bp;
  OCIBind *threshold_bp;
  OCIDefine *low_stock_bp;
  int norow;
};
typedef struct _stoctx stoctx;

  /* temporary structures needed since oracle binds to some vars differently
     than we store in our tpcc structures from tpccstruct.h */

typedef struct _deltemp {
  char cvtcr_date[DATE_TIME_SIZ];
  OCIDate cr_date;
} deltemp;

typedef struct _newtemp {
  char entry_date[DATE_TIME_SIZ + 1];
  OCIDate cr_date;
  int nol_i_id[MAX_OL];
  int nol_supply_w_id[MAX_OL];
  int nol_quantity[MAX_OL];
  char i_name[MAX_OL][25];
  int s_quantity[MAX_OL];
  int i_price[MAX_OL];
  int nol_amount[MAX_OL];
  char brand_generic[MAX_OL];
  float c_discount;
  float w_tax;
  float d_tax;
  int n_retry;
} newtemp;

typedef struct _ordtemp {
  OCIDate entry_date;
  char entry_date_str[DATE_TIME_SIZ];
  int loc_ol_i_id[MAX_OL];
  int loc_ol_supply_w_id[MAX_OL];
  int loc_ol_quantity[MAX_OL];
  int loc_ol_amount[MAX_OL];
  OCIDate loc_ol_delivery_date[MAX_OL];
  char ol_delivery_date_str[MAX_OL][DATE_SIZ];
} ordtemp;

typedef struct _paytemp {
  char h_date[DATE_TIME_SIZ];
  OCIDate customer_sdate;
  char c_since_str[DATE_SIZ];
  OCIDate cr_date;
  float c_discount;
  int h_amount;
  int c_credit_lim;
  int p_retry;
} paytemp;


typedef struct _oracontext {
  /* V8 handles for talking to Oracle */
  OCIEnv *tpcenv;
  OCIServer *tpcsrv;
  OCIError *errhp;
  OCIError *datecvterrhp;
  OCISvcCtx *tpcsvc;
  OCISession *tpcusr;
  OCIStmt *curi;
  /* other V8 additions */
  dvoid *xmem;
  /* are these really needed since we do not malloc and therefore do not
     need to free in *txn*done  ???*/
  int del_init;
  int new_init;
  int pay_init;
  int ord_init;
  int sto_init;
  /* data areas where cursors will find data */
  GenericData                bindvars;
  /* oracle structures for bind data information during a transaction */
  ordctx octx;
  delctx dctx;
  delctx dctx2;
  newctx nctx;
  payctx pctx;
  stoctx sctx;
  defctx cbctx;
  amtctx actx;
  /* temporary data areas for cursor data - oracle stores/binds
     differently than tpcc */
  union {
    deltemp del;
    newtemp new;
    ordtemp ord;
    paytemp pay;
  } tempvars;
} OraContext;


#define OCIERROR(p,function)\
       ocierror(__FILE__,__LINE__,(p),(function))

#define OCIBND(stmp, bndp, p, sqlvar, progv, progvl, ftype)\
      ocierror(__FILE__,__LINE__, (p), \
           OCIBindByName((stmp), &(bndp), (p->errhp), \
                                 (text *)(sqlvar), strlen((sqlvar)),\
                                 (progv), (progvl), (ftype),0,0,0,0,0,OCI_DEFAULT))

#define OCIBNDRA(stmp,bndp,p,sqlvar,progv,progvl,ftype,indp,alen,arcode) \
      ocierror(__FILE__,__LINE__,(p), \
          OCIBindByName((stmp),&(bndp),(p->errhp),(text *)(sqlvar),strlen((sqlvar)),\

(progv),(progvl),(ftype),(indp),(alen),(arcode),0,0,OCI_DEFAULT))
```

```
#define
OCIBNDRAD(stmp,bndp,p,sqlvar,progvl,ftype,indp,ctxp,cbf_nodata,cbf_data) \
      ocierror(__FILE__,__LINE__,(p), \
          OCIBindByName((stmp),&(bndp),(p->errhp),(text *)(sqlvar), \
                    strlen((sqlvar)),0,(progvl),(ftype), \
                    indp,0,0,0,0,OCI_DATA_AT_EXEC)); \
      ocierror(__FILE__,__LINE__,(p), \
          OCIBindDynamic((bndp),(p->errhp),(ctxp),(cbf_nodata),(ctxp),(cbf_data)))



#define OCIBNDR(stmp,bndp,p,sqlvar,progv,progvl,ftype,indp,alen,arcode) \
      ocierror(__FILE__,__LINE__,(p), \
        OCIBindByName((stmp),&(bndp),(p->errhp),(text *)(sqlvar),strlen((sqlvar)),\

(progv),(progvl),(ftype),(indp),(alen),(arcode),0,0,OCI_DEFAULT))


#define OCIBNDRAA(stmp,bndp,p,sqlvar,progv,progvl,ftype,indp,alen,arcode,ms,cu)
\
      ocierror(__FILE__,__LINE__,(p), \
          OCIBindByName((stmp), &(bndp), (p->errhp), \
                              (text *)(sqlvar), strlen((sqlvar)),\
                              (progv), (progvl), (ftype),(indp),(alen),(arcode),\
                        (ms),(cu),OCI_DEFAULT))



#define OCIDEFINE(stmp,dfnp,errp,pos,progv,progvl,ftype)\
      OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progv),(progvl),(ftype),\
                                  0,0,0,OCI_DEFAULT)


#define OCIDEF(stmp,dfnp,errp,pos,progv,progvl,ftype) \
      OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progv),(progvl),\
                  (ftype),NULL,NULL,NULL,OCI_DEFAULT)


#define OCIDFNRA(stmp,dfnp,errp,pos,progv,progvl,ftype,indp,alen,arcode) \
      OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progv),\
                                                (progvl),(ftype),(indp),(alen),\
                                                (arcode),OCI_DEFAULT)


#define OCIDFNDYN(stmp,dfnp,errp,pos,progv,progvl,ftype,indp,ctxp,cbf_data) \
      ocierror(__FILE__,__LINE__,(errp), \
      OCIHandleAlloc((stmp),(dvoid**)&(dfnp),OCI_HTYPE_DEFINE,0,\
                                                (dvoid**)0));\
      ocierror(__FILE__,__LINE__,(errp), \
      OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progv), (progvl),(ftype),\
                          (indp),NULL,NULL, OCI_DYNAMIC_FETCH));\
      ocierror(__FILE__,__LINE__,(errp), \
      OCIDefineDynamic((dfnp),(errp),(ctxp),(cbf_data)));


 /* old defines for v7 */
                                                /******

#define OBNDRV(lda,cursor,sqlvar,progv,progvl,ftype)\
   if (obndrv((cursor),(text*)(sqlvar),NA,(ub1*)(progv),(progvl),(ftype),NA,\
     (sb2 *)0, (text *)0, NA, NA))\
     {ErrRpt(lda,cursor->rc);return(ERR_DB_ERROR);}\
   else\
     DISCARD 0

#define OBNDRA(lda,cursor,sqlvar,progv,progvl,ftype,indp,alen,arcode)\
   if (obndra((cursor),(text*)(sqlvar),NA,(ub1*)(progv),(progvl),(ftype),NA,\
     (indp),(alen),(arcode),(ub4)0,(ub4*)0,(text*)0,NA,NA))\
     {ErrRpt(lda,cursor->rc);return(ERR_DB_ERROR);}\
   else\
     DISCARD 0

#define OBNDRAA(lda,cursor,sqlvar,progv,progvl,ftype,indp,alen,arcode,ms,cs)\
   if (obndra((cursor),(text*)(sqlvar),NA,(ub1*)(progv),(progvl),(ftype),NA,\
     (indp),(alen),(arcode),(ub4)(ms),(ub4*)(cs),(text*)0,NA,NA))\
     {ErrRpt(lda,cursor->rc);return(ERR_DB_ERROR);}\
   else\
     DISCARD 0

#define ODEFIN(lda,cursor,pos,buf,bufl,ftype,scale,indp,fmt,fmtl,fmtt,rlen,rcode)\
   if (odefin((cursor),(pos),(ub1*)(buf),(bufl),(ftype),(scale),(indp),\
     (text*)(fmt),(fmtl),(fmtt),(rlen),(rcode)))\
     {ErrRpt(lda,cursor->rc);return(ERR_DB_ERROR);}\
   else\
     DISCARD 0

#define OEXFET(lda,cursor,nrows,cancel,exact)\
   if (oexfet((cursor),(nrows),(cancel),(exact)))\
     {if ((cursor)->rc == 1403) DISCARD 0; \
      else if (ErrRpt(lda,cursor->rc)==RECOVERR) \
        {orol(lda);return(RECOVERR);} \
      else{orol(lda);return(ERR_DB_ERROR);}}\
   else\
     DISCARD 0

#define OOPEN(lda,cursor)\
   if (oopen((cursor),(lda),(text*)0,NA,NA,(text*)0,NA))\
     {ErrRpt(lda,cursor->rc);return(ERR_DB_ERROR);}\
   else\
     DISCARD 0

#define OPARSE(lda,cursor,sqlstm,sqll,defflg,lngflg)\
   if (oparse((cursor),(sqlstm),(sb4)(sqll),(defflg),(ub4)(lngflg)))\
     {ErrRpt(lda,cursor->rc);return(ERR_DB_ERROR);}\
   else\
     DISCARD 0

#define OFEN(lda,cursor,nrows)\
   if (ofen((cursor),(nrows)))\
     {if (ErrRpt(lda,cursor->rc)==RECOVERR) \
        {orol(lda);return(RECOVERR);} \
      else{orol(lda);return(ERR_DB_ERROR);}}\
   else\
     DISCARD 0

#define OEXEC(lda,cursor)\
   if (oexec((cursor)))\
     {if (ErrRpt(lda,cursor->rc)==RECOVERR) \
        {orol(lda);return(RECOVERR);} \
      else{orol(lda);return(ERR_DB_ERROR);}}\
   else\
     DISCARD 0


#define OCOM(lda,cursor)\
   if (ocom((lda))) \
     {ErrRpt(lda,cursor->rc);orol(lda);return(-1);}\
   else\
     DISCARD 0


#define OEXN(lda,cursor,iters,rowoff)\
   if (oexn((cursor),(iters),(rowoff))) \
     {if (ErrRpt(lda,cursor->rc)==RECOVERR) \
        {orol(lda);return(RECOVERR);} \
      else{orol(lda);return(-1);}}\
   else\
     DISCARD 0

     ****************/


/* prototypes */
extern int tkvcninit (NewOrderData *pNew,
                                OraContext *p);

extern int tkvcn (NewOrderData *pNew, OraContext *p);

extern void tkvcndone (newctx *pnctx);

extern int tkvcpinit (PaymentData *pPay,
                                OraContext *p);

extern int tkvcp (PaymentData *pPay, OraContext *p);

extern void tkvcpdone (payctx *ppctx);

extern int tkvcoinit(OrderStatusData *pOrd,
                                OraContext *p);
```

```
extern int tkvco (OrderStatusData *pOrd, OraContext *p);

extern void tkvcodone (ordctx *poctx);

extern int tkvcsinit(StockLevelData *pOrd,
                                OraContext *p);

extern int tkvcs (OraContext *p);

extern void tkvcsdone (stoctx *psctx);


extern int tkvcdinit (DeliveryData *pDel,
                                OraContext *p);

extern int tkvcd (DeliveryData *pDel, OraContext *p);

extern void tkvcddone (delctx *pdctx);


int ocierror(char *fname, int lineno, OraContext *p, sword status);
extern int ErrRpt(Lda_Def *pLda, int rc);
void TPCCErr( char *fmt, ...);
void TPCCLog( char *fmt, ...);


#endif /* ORACLE_DB_H */
```

oracle_txns8.c

```
/*+ file: oracle_txns8.c  based on Oracle files - plpay.c plnew.c plord.c
                                pldel.c plsto.c        -*/
/*+=========================================================
======+
|       Copyright (c) 1995  Oracle Corp, Redwood Shores, CA       |
|                OPEN SYSTEMS PERFORMANCE GROUP                |
|                    All Rights Reserved                    |

+=========================================================
=====+
| DESCRIPTION
|   OCI version (using PL/SQL stored procedure) of
|   PAYMENT transaction in TPC-C benchmark.
|   OCI version (using PL/SQL stored procedure) of
|   NEW ORDER transaction in TPC-C benchmark.
|   OCI version (using PL/SQL anynomous block) of
|   ORDER STATUS transaction in TPC-C benchmark.
|   OCI version of DELIVERY transaction in TPC-C benchmark.
|   OCI version of STOCK LEVEL transaction in TPC-C benchmark.

+=========================================================
====-*/
/*+*********************************************************************
*******
 *                                                                  *
 *  COPYRIGHT (c) 1998, 2000 BY                                     *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                            *
 *                                                                  *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY  OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                    *
 *                                                                  *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                    *
 *                                                                  *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                  *
 *                                                                  *

*********************************************************************
*****-*/

/*+
 * Abstract: This file contains the transaction routines for connection
 *          to the oracle v8 database - for the tpcc benchmark.
 *
 * Modified history:
 *
 *
 */

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

#include <oci.h>
#include <ocidfn.h>
#include <ociapr.h>


#ifdef _WIN32
# include <windows.h>
#endif

#include <tpccerr.h>
#include <tpccstruct.h>
#include <oracle_db8.h>

#ifdef OL_CHECK
# include <httpext.h>
extern int iMaxWareHouses;
#endif

/* prototypes */


void vgetdate (unsigned char *oradt)
{
 struct tm *loctime;
 time_t  int_time;

 struct ORADATE {
 unsigned char              century;
 unsigned char              year;
 unsigned char              month;
 unsigned char              day;
 unsigned char              hour;
 unsigned char              minute;
 unsigned char              second;
 } Date;
int century;
int cnvrtOK;

/* assume convert is successful */
cnvrtOK = 1;

/* get the current date and time as an integer */
time( &int_time);

/* Convert the current date and time into local time */
loctime = localtime( &int_time);

century = (1900+loctime->tm_year) / 100;

Date.century = (unsigned char)(century + 100);
if (Date.century < 119 || Date.century > 120) cnvrtOK = 0;
Date.year  = (unsigned char)(loctime->tm_year+100);
if (Date.year > 99) Date.year -= 100;
if (Date.year < 100 || Date.year > 199) cnvrtOK = 0;
Date.month = (unsigned char)(loctime->tm_mon + 1);
if (Date.month < 1 || Date.month > 12) cnvrtOK = 0;
```

```
Date.day   = (unsigned char)loctime->tm_mday;
if (Date.day < 1 || Date.day > 31) cnvrtOK = 0;
Date.hour  = (unsigned char)(loctime->tm_hour + 1);
if (Date.hour < 1 || Date.hour > 24) cnvrtOK = 0;
Date.minute= (unsigned char)(loctime->tm_min + 1);
if (Date.minute < 1 || Date.minute > 60) cnvrtOK = 0;
Date.second= (unsigned char)(loctime->tm_sec + 1);
if (Date.second < 1 || Date.second > 60) cnvrtOK = 0;


if (cnvrtOK)
  memcpy(oradt,&Date,7);
  else
  *oradt = '\0';

 return;

}
void cvtdmy (unsigned char *oradt, char *outdate)
{

      struct ORADATE {
            unsigned char   century;
            unsigned char   year;
            unsigned char   month;
            unsigned char   day;
            unsigned char   hour;
            unsigned char   minute;
            unsigned char   second;
      } Date;


      int day,month,year;

      memcpy(&Date,oradt,7);

      year = (Date.century-100)*100 + Date.year-100;
      month = Date.month;
      day = Date.day;
      /* sprintf(outdate,"%02d-%02d-%4d\0",day,month,year); */
      sprintf(outdate,"%02d-%02d-%4d",day,month,year);

      return;
}

void cvtdmyhms (unsigned char *oradt, char *outdate)
{

      struct ORADATE {
            unsigned char   century;
            unsigned char   year;
            unsigned char   month;
            unsigned char   day;
            unsigned char   hour;
            unsigned char   minute;
            unsigned char   second;
      } Date;


      int day,month,year;
      int hour,min,sec;

      memcpy(&Date,oradt,7);

      year = (Date.century-100)*100 + Date.year-100;
      month = Date.month;
      day = Date.day;
      hour = Date.hour - 1;
      min = Date.minute - 1;
      sec = Date.second - 1;

      /*sprintf(outdate,"%02d-%02d-%4d %02d:%02d:%02d\0", */
      sprintf(outdate,"%02d-%02d-%4d %02d:%02d:%02d",
                  day,month,year,hour,min,sec);

      return;
}

swapitemstock (int i, int j, int *i_price, char i_name[][25],
                                 int *s_quantity, newctx *nctx)
{

  int tempi;
  int tempf;
  char tempstr[52];
  ub2 tempub2;
  sb2 tempsb2;
  OCIRowid  *tmprid;

  tempsb2 = nctx->cons_ind[i];
  nctx->cons_ind[i] = nctx->cons_ind[j];
  nctx->cons_ind[j] = tempsb2;
  tempub2 = nctx->cons_len[i];
  nctx->cons_len[i] = nctx->cons_len[j];
  nctx->cons_len[j] = tempub2;
  tempub2 = nctx->cons_rcode[i];
  nctx->cons_rcode[i] = nctx->cons_rcode[j];
  nctx->cons_rcode[j] = tempub2;
  tempi = nctx->cons[i];
  nctx->cons[i] = nctx->cons[j];
  nctx->cons[j] = tempi;

  tempsb2 = nctx->s_rowid_ind[i];
  nctx->s_rowid_ind[i] = nctx->s_rowid_ind[j];
  nctx->s_rowid_ind[j] = tempsb2;
  tempub2 = nctx->s_rowid_len[i];
  nctx->s_rowid_len[i] = nctx->s_rowid_len[j];
  nctx->s_rowid_len[j] = tempub2;
  tempub2 = nctx->s_rowid_rcode[i];
  nctx->s_rowid_rcode[i] = nctx->s_rowid_rcode[j];
  nctx->s_rowid_rcode[j] = tempub2;
  tmprid = nctx->s_rowid_ptr[i];
  nctx->s_rowid_ptr[i]= nctx->s_rowid_ptr[j];
  nctx->s_rowid_ptr[j]=tmprid;

  tempsb2 = nctx->i_price_ind[i];
  nctx->i_price_ind[i] = nctx->i_price_ind[j];
  nctx->i_price_ind[j] = tempsb2;
  tempub2 = nctx->i_price_len[i];
  nctx->i_price_len[i] = nctx->i_price_len[j];
  nctx->i_price_len[j] = tempub2;
  tempub2 = nctx->i_price_rcode[i];
  nctx->i_price_rcode[i] = nctx->i_price_rcode[j];
  nctx->i_price_rcode[j] = tempub2;
  tempf = i_price[i];
  i_price[i] = i_price[j];
  i_price[j] = tempf;

  tempsb2 = nctx->i_name_ind[i];
  nctx->i_name_ind[i] = nctx->i_name_ind[j];
  nctx->i_name_ind[j] = tempsb2;
  tempub2 = nctx->i_name_len[i];
  nctx->i_name_len[i] = nctx->i_name_len[j];
  nctx->i_name_len[j] = tempub2;
  tempub2 = nctx->i_name_rcode[i];
  nctx->i_name_rcode[i] = nctx->i_name_rcode[j];
  nctx->i_name_rcode[j] = tempub2;
  strncpy (tempstr, i_name[i], 25);
  strncpy (i_name[i], i_name[j], 25);
  strncpy (i_name[j], tempstr, 25);

  tempsb2 = nctx->i_data_ind[i];
  nctx->i_data_ind[i] = nctx->i_data_ind[j];
  nctx->i_data_ind[j] = tempsb2;
  tempub2 = nctx->i_data_len[i];
  nctx->i_data_len[i] = nctx->i_data_len[j];
  nctx->i_data_len[j] = tempub2;
  tempub2 = nctx->i_data_rcode[i];
  nctx->i_data_rcode[i] = nctx->i_data_rcode[j];
  nctx->i_data_rcode[j] = tempub2;
  strncpy (tempstr, nctx->i_data[i], 51);
  strncpy (nctx->i_data[i], nctx->i_data[j], 51);
  strncpy (nctx->i_data[j], tempstr, 51);

  tempsb2 = nctx->s_quantity_ind[i];
  nctx->s_quantity_ind[i] = nctx->s_quantity_ind[j];
  nctx->s_quantity_ind[j] = tempsb2;
  tempub2 = nctx->s_quantity_len[i];
  nctx->s_quantity_len[i] = nctx->s_quantity_len[j];
  nctx->s_quantity_len[j] = tempub2;
```

```
  tempub2 = nctx->s_quantity_rcode[i];
  nctx->s_quantity_rcode[i] = nctx->s_quantity_rcode[j];
  nctx->s_quantity_rcode[j] = tempub2;
  tempi = s_quantity[i];
  s_quantity[i] = s_quantity[j];
  s_quantity[j] = tempi;

  tempsb2 = nctx->s_dist_info_ind[i];
  nctx->s_dist_info_ind[i] = nctx->s_dist_info_ind[j];
  nctx->s_dist_info_ind[j] = tempsb2;
  tempub2 = nctx->s_dist_info_len[i];
  nctx->s_dist_info_len[i] = nctx->s_dist_info_len[j];
  nctx->s_dist_info_len[j] = tempub2;
  tempub2 = nctx->s_dist_info_rcode[i];
  nctx->s_dist_info_rcode[i] = nctx->s_dist_info_rcode[j];
  nctx->s_dist_info_rcode[j] = tempub2;
  strncpy (tempstr, nctx->s_dist_info[i], 25);
  strncpy (nctx->s_dist_info[i], nctx->s_dist_info[j], 25);
  strncpy (nctx->s_dist_info[j], tempstr, 25);

  tempsb2 = nctx->s_data_ind[i];
  nctx->s_data_ind[i] = nctx->s_data_ind[j];
  nctx->s_data_ind[j] = tempsb2;
  tempub2 = nctx->s_data_len[i];
  nctx->s_data_len[i] = nctx->s_data_len[j];
  nctx->s_data_len[j] = tempub2;
  tempub2 = nctx->s_data_rcode[i];
  nctx->s_data_rcode[i] = nctx->s_data_rcode[j];
  nctx->s_data_rcode[j] = tempub2;
  strncpy (tempstr, nctx->s_data[i], 51);
  strncpy (nctx->s_data[i], nctx->s_data[j], 51);
  strncpy (nctx->s_data[j], tempstr, 51);
  return 0;
}

/* the arrays are initialized based on a successful select from */
/* stock/item. We need to shift the values in the orderline array */
/* one position up to compensate when we have an invalid item */

shiftitemstock (int i, int j, newctx *nctx, OraContext *p)
{
  newtemp *ntemp = &(p->tempvars.new);

  /* shift up the values for the stock table */
  nctx->s_remote[i] =  nctx->s_remote[j];

  /* shift up the order_line values */

  nctx->nol_i_id_ind[i]=nctx->nol_i_id_ind[j];
  ntemp->nol_i_id[i] = ntemp->nol_i_id[j];

  nctx->nol_quantity_ind[i] =  nctx->nol_quantity_ind[j];
  ntemp->nol_quantity[i] = ntemp->nol_quantity[j];

  nctx->nol_supply_w_id_ind [i] =  nctx->nol_supply_w_id_ind[j];
  ntemp->nol_supply_w_id[i] = ntemp->nol_supply_w_id[j];
  return 0;
}

int SelItemStk (NewOrderData *pNew,
                                int *pstatus,
                                int retries, int proc_no, newctx *nctx, OraContext *p)
{

  int i, j, rpc3,rcount;
  int errcode;
  int execstatus;

#ifdef OL_CHECK
  newtemp *ntemp = &(p->tempvars.new);
  for (i = 0; i < MAX_OL; i++) {
    if((TRUE == nctx->nol_supply_w_id_ind[i]) &&
    ( ntemp->nol_supply_w_id[i] > iMaxWareHouses )) {
      TPCCErr( "Bad supply warehouse ol: %d, s_w_id: %d, query: %s",
                                i+1, ntemp->nol_supply_w_id[i],
                                ((EXTENSION_CONTROL_BLOCK *)pNew->pCC)-
>lpszQueryString );
    }
  }
#endif

 /* array select from item and stock tables */
execstatus=OCIStmtExecute(p->tpcsvc,(nctx->curn3)[pNew->d_id-1],p->errhp,
                                        pNew->o_ol_cnt,0,0,0,OCI_DEFAULT);
if((execstatus != OCI_SUCCESS) && (execstatus != OCI_NO_DATA)) {
  errcode = OCIERROR(p,execstatus);
  if((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR))
  {
  /* In case of NO_DATA this should NOT return, but simply fall through */
    OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
    return (RECOVERR);
  }
  else
  {
    OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
    return (IRRECERR);
  }
}
/* mark invalid items */
OCIAttrGet((nctx->curn3)[pNew->d_id-1], OCI_HTYPE_STMT,&rcount,NULL,
                    OCI_ATTR_ROW_COUNT, p->errhp);
rpc3 = rcount;

/* the result is in order, so we have to shift up to fill    */
/* the slot for the line with the invalid item.              */
/* If more than one item is wrong, this is not an simulated  */
/* error and we'll blow off */

if ((*pstatus = pNew->o_ol_cnt - rcount) >1)
{
   TPCCErr ("TPC-C server %d: more than 1 invalid item?\n", proc_no);
   return (rpc3);
}
if (*pstatus == 0) return (rpc3);

/* find the invalid item, transfer the rowid information */

for (i = 0; i < pNew->o_ol_cnt; i++) {
  if (nctx->cons[i] != i) break;   /* this item is invalid */
}
/***
TPCCErr("TPC-C server %d: reordering items and stocks\n",
                    proc_no);   ***/

/* not the last item - shift up */

for (j = i; j < pNew->o_ol_cnt-1; j++)
{
  shiftitemstock (j, j+1, nctx, p);
}
/* zero the last item */
  i = pNew->o_ol_cnt-1;
  nctx->nol_i_id_ind[i] = NA;
  nctx->nol_supply_w_id_ind[i] = NA;
  nctx->nol_quantity_ind[i] = NA;
  nctx->nol_amount_ind[i] = NA;
  nctx->ol_w_id_ind[i] = NA;
  nctx->ol_d_id_ind[i] = NA;
  nctx->ol_o_id_ind[i] = NA;
  nctx->null_date_ind[i]= NA;
  nctx->ol_number_ind[i] = NA;
  nctx->ol_dist_info_ind[i] = NA;
  nctx->s_remote_ind[i] = NA;
  nctx->s_quant_ind[i] = NA;

  nctx->nol_i_id_len[i] = 0;
  nctx->nol_supply_w_id_len[i] = 0;
  nctx->nol_quantity_len[i] = 0;
  nctx->nol_amount_len[i] = 0;
  nctx->ol_w_id_len[i] = 0;
  nctx->ol_d_id_len[i] = 0;
  nctx->ol_o_id_len[i] = 0;
  nctx->ol_number_len[i] = 0;
  nctx->ol_dist_info_len[i] = 0;
  nctx->null_date_ind[i] = 0;
  nctx->s_remote_len[i] = 0;
  nctx->s_quant_len[i] = 0;

 return (rpc3);
}
```

```
UpdStk (OraContext *p, NewOrderData *pNew, int proc_no)
{
  int rcount;
  int execstatus;
  int errcode;
  newctx *nctx = &(p->nctx);

#ifdef OL_CHECK
  int i;
   newtemp *ntemp = &(p->tempvars.new);
   for (i = 0; i < MAX_OL; i++) {
     if((TRUE == nctx->nol_supply_w_id_ind[i]) &&
     ( ntemp->nol_supply_w_id[i] > iMaxWareHouses )) {
       TPCCErr( "Bad supply warehouse ol in updstk: %d, s_w_id: %d, query: %s",
                                 i+1, ntemp->nol_supply_w_id[i],
                                 ((EXTENSION_CONTROL_BLOCK *)pNew->pCC)-
>lpszQueryString );
     }
   }
#endif
   /* array update of stock table */

  execstatus = OCIStmtExecute(p->tpcsvc,nctx->curn2,p->errhp,pNew->o_ol_cnt,
                        0,0,0,OCI_DEFAULT);
   if(execstatus != OCI_SUCCESS) {
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     errcode = OCIERROR(p, execstatus);
     if((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)) {
                 return (RECOVERR);
     } else {
       return (IRRECERR);
     }
   }
  OCIAttrGet(nctx->curn2,OCI_HTYPE_STMT,&rcount,NULL,
          OCI_ATTR_ROW_COUNT, p->errhp);
  if (rcount != (pNew->o_ol_cnt) ) {
     TPCCErr ("Error in TPC-C server %d: array update failed in UpdStk()\n",
            proc_no);
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     return (IRRECERR);
   }
   return (rcount);
}



UpdStk2 (OraContext *p, NewOrderData *pNew, int status, int proc_no)
{

  int rcount;
  int execstatus;
  int errcode;
  newctx *nctx = &(p->nctx);

#ifdef OL_CHECK
  int i;
   newtemp *ntemp = &(p->tempvars.new);
   for (i = 0; i < MAX_OL; i++) {
     if((TRUE == nctx->nol_supply_w_id_ind[i]) &&
     ( ntemp->nol_supply_w_id[i] > iMaxWareHouses )) {
       TPCCErr( "Bad supply warehouse ol in updstk2: %d, s_w_id: %d, query: %s",
                                 i+1, ntemp->nol_supply_w_id[i],
                                 ((EXTENSION_CONTROL_BLOCK *)pNew->pCC)-
>lpszQueryString );
     }
   }
#endif
   /* array update of stock table */

  execstatus = OCIStmtExecute(p->tpcsvc,nctx->curn2,p->errhp,
                                                pNew->o_ol_cnt-
status,0,0,0,OCI_DEFAULT);
   if(execstatus != OCI_SUCCESS) {
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     errcode = OCIERROR(p,execstatus);
     if(errcode == NOT_SERIALIZABLE) {
                 return (RECOVERR);
     } else if (errcode == RECOVERR) {
                 return (RECOVERR);
     } else {
       return (IRRECERR);
     }
   }
   OCIAttrGet(nctx->curn2,OCI_HTYPE_STMT,&rcount,NULL,
OCI_ATTR_ROW_COUNT,
                        p->errhp);

   if (rcount != (pNew->o_ol_cnt - status)) {
     TPCCErr("Error in TPC-C server %d: array update failed in UpdStk2()\n",
             proc_no);
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     return (NO_COMMIT);
   }

   return (rcount);

}


/* stock level transaction */

#define SLSQLTXT "SELECT count (DISTINCT s_i_id) \
          FROM ordl, stok, dist \
          WHERE d_id = :d_id AND d_w_id = :w_id AND \
               d_id = ol_d_id AND d_w_id = ol_w_id AND \
               ol_i_id = s_i_id AND ol_w_id = s_w_id AND \
               s_quantity < :threshold AND \
               ol_o_id BETWEEN (d_next_o_id - 20) AND (d_next_o_id - 1)"


#define SQLTXTTEST "BEGIN stocklevel.getstocklevel (:w_id, :d_id, \
:threshold); END;"

tkvcsinit (StockLevelData *pSL,
                  OraContext *p)

{
  stoctx *sctx = &(p->sctx);
  text stmbuf[SQL_BUF_SIZE];

  sctx->curs = NULL;

  memset(sctx,(char)0,sizeof(stoctx));
  sctx->norow=0;

  OCIERROR(p, OCIHandleAlloc(p->tpcenv,(dvoid**)&(sctx-
>curs),OCI_HTYPE_STMT,0,
                                                (dvoid**)0));
  sprintf ((char *) stmbuf, SLSQLTXT);
  OCIERROR(p,OCIStmtPrepare(sctx->curs,p->errhp,stmbuf,strlen((char *)stmbuf),
                                                OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(p, OCIAttrSet(sctx->curs,OCI_HTYPE_STMT,(dvoid*)&sctx-
>norow,0,
                                        OCI_ATTR_PREFETCH_ROWS,p->errhp));

  /* bind variables */

  OCIBND(sctx->curs,sctx->w_id_bp,p, ":w_id", ADR(pSL->w_id),sizeof(int),
                SQLT_INT);
  OCIBND(sctx->curs,sctx->d_id_bp,p, ":d_id", ADR(pSL->ld_id),sizeof(int),
                SQLT_INT);
  OCIBND(sctx->curs,sctx->threshold_bp,p, ":threshold", ADR(pSL->threshold),
                sizeof(int),SQLT_INT);
  OCIDEF(sctx->curs,sctx->low_stock_bp,p->errhp, 1, ADR(pSL->low_stock),
     sizeof(int), SQLT_INT);

  return (ERR_DB_SUCCESS);

}

tkvcs (OraContext *p)
{
  stoctx *sctx = &(p->sctx);

  int execstatus = 0;
  int errcode = 0;

  execstatus = OCIStmtExecute(p->tpcsvc,sctx->curs,p->errhp,1,0,0,0,
```

```
                                        OCI_COMMIT_ON_SUCCESS |
OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS) {
    OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
    errcode = OCIERROR(p,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      return (RECOVERR);
    } else if (errcode == RECOVERR) {
      return (RECOVERR);
    } else if (errcode == SNAPSHOT_TOO_OLD) {
      return (RECOVERR);
    } else {
      return (ERR_DB_ERROR);
    }
  }

  return (ERR_DB_SUCCESS);

}

void tkvcsdone (stoctx *psctx)

{
  stoctx sctx = *psctx;

  if(NULL != sctx.curs)
    OCIHandleFree((dvoid *)sctx.curs,OCI_HTYPE_STMT);

}


#define SQLTXT_PAY_ZERO "BEGIN apayment.adopayment(:w_id, :d_id, :c_w_id,
:c_d_id, \
   :c_id, 0, \
   :h_amount, :c_last, :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, :c_data, \
   :h_date, :retry, :cr_date); END;"

#define SQLTXT_PAY_NONZERO "BEGIN apayment.adopayment(:w_id, :d_id,
:c_w_id, :c_d_id, \
   :c_id, 1, \
   :h_amount, :c_last, :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, :c_data, \
   :h_date, :retry, :cr_date); END;"

#define SQLTXT_PAY_INIT "BEGIN initpay.pay_init; END;"


tkvcpinit (PaymentData *pPay,
                OraContext *p)

{
  payctx *pctx = &(p->pctx);
  paytemp *ptemp = &(p->tempvars.pay);

  text stmbuf[SQL_BUF_SIZE];

  pctx->curpi = NULL;
  pctx->curp0 = NULL;
  pctx->curp1 = NULL;

  memset(pctx,(char)0,sizeof(payctx));

/* cursor for init */
   OCIERROR(p,OCIHandleAlloc(p->tpcenv, (dvoid **)(&(pctx->curpi)),
         OCI_HTYPE_STMT,0,(dvoid**)0));

   OCIERROR(p,OCIHandleAlloc(p->tpcenv, (dvoid **)(&(pctx->curp0)),
         OCI_HTYPE_STMT,0,(dvoid**)0));
   OCIERROR(p,OCIHandleAlloc(p->tpcenv, (dvoid **)(&(pctx->curp1)),
         OCI_HTYPE_STMT,0,(dvoid**)0));

   /*  build the init statement  and execute it */

   sprintf ((char*)stmbuf, SQLTXT_PAY_INIT);
   OCIERROR(p,OCIStmtPrepare(pctx->curpi, p->errhp, stmbuf,
         strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));
  OCIERROR(p,
    OCIStmtExecute(p->tpcsvc,pctx->curpi,p->errhp,1,0,0,0,OCI_DEFAULT));

  /* customer id !=  0, go by customer id */
  OCIERROR(p,OCIStmtPrepare(pctx->curp0, p->errhp, stmbuf_paynz,
         strlen((char *)stmbuf_paynz), OCI_NTV_SYNTAX, OCI_DEFAULT));

  /* customer id ==  0, go by last name */
  OCIERROR(p,OCIStmtPrepare(pctx->curp1, p->errhp, stmbuf_payz,
         strlen((char *)stmbuf_payz), OCI_NTV_SYNTAX, OCI_DEFAULT));

 pctx->w_id_ind = TRUE;
 pctx->w_id_len = SIZ(pPay->w_id);
 pctx->d_id_ind = TRUE;
 pctx->d_id_len = SIZ(pPay->d_id);
 pctx->c_w_id_ind = TRUE;
 pctx->c_w_id_len = SIZ(pPay->c_w_id);
 pctx->c_d_id_ind = TRUE;
 pctx->c_d_id_len = SIZ(pPay->c_d_id);
 pctx->c_id_ind = TRUE;
 pctx->c_id_len = 0;
 pctx->h_amount_len = SIZ(ptemp->h_amount);
 pctx->h_amount_ind = TRUE;
 pctx->c_last_ind = TRUE;
 pctx->c_last_len = 0;
 pctx->w_street_1_ind = TRUE;
 pctx->w_street_1_len = 0;
 pctx->w_street_2_ind = TRUE;
 pctx->w_street_2_len = 0;
 pctx->w_city_ind = TRUE;
 pctx->w_city_len = 0;
 pctx->w_state_ind = TRUE;
 pctx->w_state_len = 0;
 pctx->w_zip_ind = TRUE;
 pctx->w_zip_len = 0;
 pctx->d_street_1_ind = TRUE;
 pctx->d_street_1_len = 0;
 pctx->d_street_2_ind = TRUE;
 pctx->d_street_2_len = 0;
 pctx->d_city_ind = TRUE;
 pctx->d_city_len = 0;
 pctx->d_state_ind = TRUE;
 pctx->d_state_len = 0;
 pctx->d_zip_ind = TRUE;
 pctx->d_zip_len = 0;
 pctx->c_first_ind = TRUE;
 pctx->c_first_len = 0;
 pctx->c_middle_ind = TRUE;
 pctx->c_middle_len = 0;
 pctx->c_street_1_ind = TRUE;
 pctx->c_street_1_len = 0;
 pctx->c_street_2_ind = TRUE;
 pctx->c_street_2_len = 0;
 pctx->c_city_ind = TRUE;
 pctx->c_city_len = 0;
 pctx->c_state_ind = TRUE;
 pctx->c_state_len = 0;
 pctx->c_zip_ind = TRUE;
 pctx->c_zip_len = 0;
 pctx->c_phone_ind = TRUE;
 pctx->c_phone_len = 0;
 pctx->c_since_ind = TRUE;
 pctx->c_since_len = 0;
 pctx->c_credit_ind = TRUE;
 pctx->c_credit_len = 0;
 pctx->c_credit_lim_ind = TRUE;
 pctx->c_credit_lim_len = 0;
 pctx->c_discount_ind = TRUE;
 pctx->c_discount_len = 0;
 pctx->c_balance_ind = TRUE;
 pctx->c_balance_len = sizeof(double);
 pctx->c_data_ind = TRUE;
 pctx->c_data_len = 0;
 pctx->h_date_ind = TRUE;
 pctx->h_date_len = 0;
 pctx->retries_ind = TRUE;
 pctx->retries_len = 0;
 pctx->cr_date_ind = TRUE;
 /* pctx->cr_date_len = 7; */
```

```
pctx->cr_date_len = sizeof(ptemp->cr_date);



 /* bind variables */


  OCIBNDR(pctx->curp0, pctx->w_id_bp, p,":w_id",ADR(pPay->w_id),
                SIZ(int), SQLT_INT, &pctx->w_id_ind, NULL, NULL);
  OCIBNDR(pctx->curp0, pctx->d_id_bp, p,":d_id",ADR(pPay->d_id),
                SIZ(int), SQLT_INT, &pctx->d_id_ind, NULL, NULL);
  OCIBND(pctx->curp0, pctx->c_w_id_bp, p,":c_w_id",ADR(pPay->c_w_id),
                SIZ(int), SQLT_INT);
  OCIBND(pctx->curp0, pctx->c_d_id_bp, p,":c_d_id",ADR(pPay->c_d_id),
                SIZ(int), SQLT_INT);
  OCIBND(pctx->curp0, pctx->c_id_bp, p,":c_id",ADR(pPay->c_id),
                SIZ(int),  SQLT_INT);
  OCIBNDR(pctx->curp0, pctx->h_amount_bp, p,":h_amount",
                ADR(ptemp->h_amount), SIZ(ptemp->h_amount),SQLT_INT,
                &pctx->h_amount_ind, &pctx->h_amount_len, &pctx->h_amount_rc);
  OCIBNDR(pctx->curp0, pctx->c_last_bp, p,":c_last",pPay->c_last,
                SIZ(pPay->c_last), SQLT_STR, &pctx->c_last_ind, &pctx->c_last_len,
                &pctx->c_last_rc);
  OCIBNDR(pctx->curp0, pctx->w_street_1_bp, p,":w_street_1",
                pPay->w_street_1, SIZ(pPay->w_street_1),SQLT_STR,
                &pctx->w_street_1_ind, &pctx->w_street_1_len, &pctx-
>w_street_1_rc);
  OCIBNDR(pctx->curp0, pctx->w_street_2_bp, p,":w_street_2",
                pPay->w_street_2, SIZ(pPay->w_street_2),SQLT_STR,
                &pctx->w_street_2_ind, &pctx->w_street_2_len, &pctx-
>w_street_2_rc);
  OCIBNDR(pctx->curp0, pctx->w_city_bp, p,":w_city",pPay->w_city,
                SIZ(pPay->w_city),
        SQLT_STR, &pctx->w_city_ind, &pctx->w_city_len, &pctx->w_city_rc);
  OCIBNDR(pctx->curp0, pctx->w_state_bp, p,":w_state",pPay->w_state,
                SIZ(pPay->w_state), SQLT_STR, &pctx->w_state_ind,
                &pctx->w_state_len, &pctx->w_state_rc);
  OCIBNDR(pctx->curp0, pctx->w_zip_bp, p,":w_zip",pPay->w_zip,
                SIZ(pPay->w_zip),
        SQLT_STR, &pctx->w_zip_ind, &pctx->w_zip_len, &pctx->w_zip_rc);
  OCIBNDR(pctx->curp0, pctx->d_street_1_bp, p,":d_street_1",
                pPay->d_street_1,
        SIZ(pPay->d_street_1),SQLT_STR, &pctx->d_street_1_ind,
        &pctx->d_street_1_len, &pctx->d_street_1_rc);
  OCIBNDR(pctx->curp0, pctx->d_street_2_bp, p,":d_street_2",
                pPay->d_street_2,
        SIZ(pPay->d_street_2),SQLT_STR, &pctx->d_street_2_ind,
        &pctx->d_street_2_len, &pctx->d_street_2_rc);
  OCIBNDR(pctx->curp0, pctx->d_city_bp, p,":d_city",pPay->d_city,
                SIZ(pPay->d_city),
        SQLT_STR, &pctx->d_city_ind, &pctx->d_city_len, &pctx->d_city_rc);
  OCIBNDR(pctx->curp0, pctx->d_state_bp, p,":d_state",pPay->d_state,
                SIZ(pPay->d_state), SQLT_STR,
        &pctx->d_state_ind, &pctx->d_state_len, &pctx->d_state_rc);
  OCIBNDR(pctx->curp0, pctx->d_zip_bp, p,":d_zip",pPay->d_zip,
                SIZ(pPay->d_zip),
        SQLT_STR, &pctx->d_zip_ind, &pctx->d_zip_len, &pctx->d_zip_rc);
  OCIBNDR(pctx->curp0, pctx->c_first_bp, p,":c_first",pPay->c_first,
                SIZ(pPay->c_first), SQLT_STR,
                &pctx->c_first_ind, &pctx->c_first_len, &pctx->c_first_rc);
  OCIBNDR(pctx->curp0, pctx->c_middle_bp, p,":c_middle",
                pPay->c_middle,2,
        SQLT_AFC, &pctx->c_middle_ind, &pctx->c_middle_len,
        &pctx->c_middle_rc);
  OCIBNDR(pctx->curp0, pctx->c_street_1_bp, p,":c_street_1",
                pPay->c_street_1,
        SIZ(pPay->c_street_1),SQLT_STR, &pctx->c_street_1_ind,
        &pctx->c_street_1_len, &pctx->c_street_1_rc);
  OCIBNDR(pctx->curp0, pctx->c_street_2_bp, p,":c_street_2",
                pPay->c_street_2,
        SIZ(pPay->c_street_2),SQLT_STR, &pctx->c_street_2_ind,
        &pctx->c_street_2_len, &pctx->c_street_2_rc);
  OCIBNDR(pctx->curp0, pctx->c_city_bp, p,":c_city",pPay->c_city,
                SIZ(pPay->c_city),
        SQLT_STR, &pctx->c_city_ind, &pctx->c_city_len, &pctx->c_city_rc);
  OCIBNDR(pctx->curp0, pctx->c_state_bp, p,":c_state",pPay->c_state,
                SIZ(pPay->c_state),
        SQLT_STR,&pctx->c_state_ind, &pctx->c_state_len, &pctx->c_state_rc);
  OCIBNDR(pctx->curp0, pctx->c_zip_bp, p,":c_zip",pPay->c_zip,
                SIZ(pPay->c_zip),
        SQLT_STR, &pctx->c_zip_ind, &pctx->c_zip_len, &pctx->c_zip_rc);
  OCIBNDR(pctx->curp0, pctx->c_phone_bp, p,":c_phone",pPay->c_phone,
                SIZ(pPay->c_phone), SQLT_STR,
                &pctx->c_phone_ind, &pctx->c_phone_len, &pctx->c_phone_rc);
  OCIBNDR(pctx->curp0, pctx->c_since_bp, p,":c_since",
                ADR(ptemp->customer_sdate),SIZ(ptemp->customer_sdate),
SQLT_ODT,
                &pctx->c_since_ind, &pctx->c_since_len, &pctx->c_since_rc);
  OCIBNDR(pctx->curp0, pctx->c_credit_bp, p,":c_credit",pPay->c_credit,
        SIZ(pPay->c_credit),SQLT_CHR,
                &pctx->c_credit_ind, &pctx->c_credit_len, &pctx->c_credit_rc);
  OCIBNDR(pctx->curp0, pctx->c_credit_lim_bp, p,":c_credit_lim",
        ADR(ptemp->c_credit_lim),SIZ(ptemp->c_credit_lim), SQLT_INT,
                &pctx->c_credit_lim_ind,
                &pctx->c_credit_lim_len, &pctx->c_credit_lim_rc);
  OCIBNDR(pctx->curp0, pctx->c_discount_bp, p,":c_discount",
        ADR(ptemp->c_discount),SIZ(ptemp->c_discount), SQLT_FLT,
                &pctx->c_discount_ind,
        &pctx->c_discount_len, &pctx->c_discount_rc);
  OCIBNDR(pctx->curp0, pctx->c_balance_bp, p,":c_balance",
                ADR(pPay->c_balance), SIZ(pPay->c_balance),SQLT_FLT,
                &pctx->c_balance_ind, &pctx->c_balance_len,
        &pctx->c_balance_rc);
  OCIBNDR(pctx->curp0, pctx->c_data_bp, p,":c_data",pPay->c_data,
                SIZ(pPay->c_data),
        SQLT_STR, &pctx->c_data_ind, &pctx->c_data_len, &pctx->c_data_rc);
/*   OCIBNDR(pctx->curp0, pctx->h_date_bp, p,":h_date",ptemp->h_date,
                SIZ(ptemp->h_date),
        SQLT_STR, &pctx->h_date_ind, &pctx->h_date_len, &pctx->h_date_rc);  */
  OCIBNDR(pctx->curp0, pctx->retries_bp, p,":retry",
                ADR(ptemp->p_retry),SIZ(ptemp->p_retry), SQLT_INT,
        &pctx->retries_ind, &pctx->retries_len, &pctx->retries_rc);
  OCIBNDR(pctx->curp0, pctx->cr_date_bp, p,":cr_date",
                ADR(ptemp->cr_date), SIZ(ptemp->cr_date),SQLT_ODT,
                &pctx->cr_date_ind, &pctx->cr_date_len,
        &pctx->cr_date_rc);


/* ---- Binds for  the second cursor   */


  OCIBNDR(pctx->curp1, pctx->w_id_bp1, p,":w_id",
                ADR(pPay->w_id),SIZ(pPay->w_id),
        SQLT_INT, &pctx->w_id_ind, &pctx->w_id_len, &pctx->w_id_rc);
  OCIBNDR(pctx->curp1, pctx->d_id_bp1, p,":d_id",ADR(pPay->d_id),
                SIZ(pPay->d_id),
        SQLT_INT, &pctx->d_id_ind, &pctx->d_id_len, &pctx->d_id_rc);
  OCIBND(pctx->curp1, pctx->c_w_id_bp1, p,":c_w_id",ADR(pPay->c_w_id),
                SIZ(pPay->c_w_id),
                SQLT_INT);
  OCIBND(pctx->curp1, pctx->c_d_id_bp1, p,":c_d_id",ADR(pPay->c_d_id),
                SIZ(pPay->c_d_id),
                SQLT_INT);
  OCIBNDR(pctx->curp1, pctx->c_id_bp1, p,":c_id",ADR(pPay->c_id),
                SIZ(int),
        SQLT_INT, &pctx->c_id_ind, &pctx->c_id_len, &pctx->c_id_rc);
  OCIBNDR(pctx->curp1, pctx->h_amount_bp1, p,":h_amount",
                ADR(ptemp->h_amount),
        SIZ(int),SQLT_INT, &pctx->h_amount_ind, &pctx->h_amount_len,
        &pctx->h_amount_rc);
  OCIBND(pctx->curp1, pctx->c_last_bp1, p,":c_last",pPay->c_last,
                SIZ(pPay->c_last),
                SQLT_STR);
  OCIBNDR(pctx->curp1, pctx->w_street_1_bp1, p,":w_street_1",
                pPay->w_street_1,
        SIZ(pPay->w_street_1),SQLT_STR, &pctx->w_street_1_ind,
        &pctx->w_street_1_len, &pctx->w_street_1_rc);
  OCIBNDR(pctx->curp1, pctx->w_street_2_bp1, p,":w_street_2",
                pPay->w_street_2,
        SIZ(pPay->w_street_2),SQLT_STR, &pctx->w_street_2_ind,
        &pctx->w_street_2_len, &pctx->w_street_2_rc);
  OCIBNDR(pctx->curp1, pctx->w_city_bp1, p,":w_city",pPay->w_city,
                SIZ(pPay->w_city),
        SQLT_STR, &pctx->w_city_ind, &pctx->w_city_len, &pctx->w_city_rc);
  OCIBNDR(pctx->curp1, pctx->w_state_bp1, p,":w_state",pPay->w_state,
                SIZ(pPay->w_state), SQLT_STR,
                &pctx->w_state_ind, &pctx->w_state_len, &pctx->w_state_rc);
  OCIBNDR(pctx->curp1, pctx->w_zip_bp1, p,":w_zip",pPay->w_zip,
                SIZ(pPay->w_zip),
```

```
        SQLT_STR, &pctx->w_zip_ind, &pctx->w_zip_len, &pctx->w_zip_rc);
  OCIBNDR(pctx->curp1, pctx->d_street_1_bp1, p,":d_street_1",
                pPay->d_street_1,
        SIZ(pPay->d_street_1),SQLT_STR, &pctx->d_street_1_ind,
        &pctx->d_street_1_len, &pctx->d_street_1_rc);
  OCIBNDR(pctx->curp1, pctx->d_street_2_bp1, p,":d_street_2",
                pPay->d_street_2,
        SIZ(pPay->d_street_2),SQLT_STR, &pctx->d_street_2_ind,
        &pctx->d_street_2_len, &pctx->d_street_2_rc);
  OCIBNDR(pctx->curp1, pctx->d_city_bp1, p,":d_city",
                pPay->d_city,SIZ(pPay->d_city),
        SQLT_STR, &pctx->d_city_ind, &pctx->d_city_len, &pctx->d_city_rc);
  OCIBNDR(pctx->curp1, pctx->d_state_bp1, p,":d_state",
                pPay->d_state, SIZ(pPay->d_state), SQLT_STR,
                &pctx->d_state_ind, &pctx->d_state_len,
        &pctx->d_state_rc);
  OCIBNDR(pctx->curp1, pctx->d_zip_bp1, p,":d_zip",pPay->d_zip,
                SIZ(pPay->d_zip),
        SQLT_STR, &pctx->d_zip_ind, &pctx->d_zip_len, &pctx->d_zip_rc);
  OCIBNDR(pctx->curp1, pctx->c_first_bp1, p,":c_first",pPay->c_first,
        SIZ(pPay->c_first), SQLT_STR,
                &pctx->c_first_ind, &pctx->c_first_len,
        &pctx->c_first_rc);
  OCIBNDR(pctx->curp1, pctx->c_middle_bp1, p,":c_middle",
                pPay->c_middle,2,
        SQLT_AFC, &pctx->c_middle_ind, &pctx->c_middle_len,
        &pctx->c_middle_rc);
  OCIBNDR(pctx->curp1, pctx->c_street_1_bp1, p,":c_street_1",
                pPay->c_street_1,
        SIZ(pPay->c_street_1),SQLT_STR, &pctx->c_street_1_ind,
        &pctx->c_street_1_len, &pctx->c_street_1_rc);
  OCIBNDR(pctx->curp1, pctx->c_street_2_bp1, p,":c_street_2",
                pPay->c_street_2,
        SIZ(pPay->c_street_2),SQLT_STR, &pctx->c_street_2_ind,
        &pctx->c_street_2_len, &pctx->c_street_2_rc);
  OCIBNDR(pctx->curp1, pctx->c_city_bp1, p,":c_city",pPay->c_city,
                SIZ(pPay->c_city),SQLT_STR,
                &pctx->c_city_ind, &pctx->c_city_len, &pctx->c_city_rc);
  OCIBNDR(pctx->curp1, pctx->c_state_bp1, p,":c_state",pPay->c_state,
                SIZ(pPay->c_state),SQLT_STR, &pctx->c_state_ind, &pctx-
>c_state_len,
                &pctx->c_state_rc);
  OCIBNDR(pctx->curp1, pctx->c_zip_bp1, p,":c_zip",pPay->c_zip,
                SIZ(pPay->c_zip),
        SQLT_STR, &pctx->c_zip_ind, &pctx->c_zip_len, &pctx->c_zip_rc);
  OCIBNDR(pctx->curp1, pctx->c_phone_bp1, p,":c_phone",pPay->c_phone,
                SIZ(pPay->c_phone), SQLT_STR,
                &pctx->c_phone_ind, &pctx->c_phone_len,
                &pctx->c_phone_rc);
  OCIBNDR(pctx->curp1, pctx->c_since_bp1, p,":c_since",
                ADR(ptemp->customer_sdate),SIZ(ptemp->customer_sdate),
SQLT_ODT,
                &pctx->c_since_ind, &pctx->c_since_len, &pctx->c_since_rc);
  OCIBNDR(pctx->curp1, pctx->c_credit_bp1, p,":c_credit",
                pPay->c_credit,
        SIZ(pPay->c_credit),SQLT_CHR,
                &pctx->c_credit_ind, &pctx->c_credit_len,
        &pctx->c_credit_rc);
  OCIBNDR(pctx->curp1, pctx->c_credit_lim_bp1, p,":c_credit_lim",
        ADR(ptemp->c_credit_lim),SIZ(int), SQLT_INT,
                &pctx->c_credit_lim_ind,
        &pctx->c_credit_lim_len, &pctx->c_credit_lim_rc);
  OCIBNDR(pctx->curp1, pctx->c_discount_bp1, p,":c_discount",
        ADR(ptemp->c_discount),SIZ(int), SQLT_FLT, &pctx->c_discount_ind,
        &pctx->c_discount_len, &pctx->c_discount_rc);
  OCIBNDR(pctx->curp1, pctx->c_balance_bp1, p,":c_balance",
                ADR(pPay->c_balance),
        SIZ(double),SQLT_FLT, &pctx->c_balance_ind, &pctx->c_balance_len,
        &pctx->c_balance_rc);
  OCIBNDR(pctx->curp1, pctx->c_data_bp1, p,":c_data",pPay->c_data,
                SIZ(pPay->c_data),
        SQLT_STR, &pctx->c_data_ind, &pctx->c_data_len, &pctx->c_data_rc);
/*   OCIBNDR(pctx->curp1, pctx->h_date_bp1, p,":h_date",
                ptemp->h_date,SIZ(ptemp->h_date),
        SQLT_STR, &pctx->h_date_ind, &pctx->h_date_len, &pctx->h_date_rc); */
  OCIBNDR(pctx->curp1, pctx->retries_bp1, p,":retry",
                ADR(ptemp->p_retry),SIZ(int), SQLT_INT,
                &pctx->retries_ind, &pctx->retries_len, &pctx->retries_rc);
  OCIBNDR(pctx->curp1, pctx->cr_date_bp1, p,":cr_date",
                ADR(ptemp->cr_date), SIZ(ptemp->cr_date), SQLT_ODT,
                        &pctx->cr_date_ind, &pctx->cr_date_len,
                &pctx->cr_date_rc);

 return (ERR_DB_SUCCESS);

}


tkvcp (PaymentData *pPay, OraContext *p)

{
  int execstatus;
  int errcode;
  payctx *pctx = &(p->pctx);
  paytemp *ptemp = &(p->tempvars.pay);
  unsigned char localcr_date[7];
  OCIError *datecvterrhp = p->datecvterrhp;

/*  vgetdate(ptemp->cr_date); */
  vgetdate(localcr_date);
  cvtdmyhms(localcr_date,ptemp->h_date);
  OCIDateFromText(datecvterrhp, (unsigned char *)ptemp->h_date,strlen(ptemp-
>h_date), (const unsigned char *)"DD-MM-YYYY HH24:MI:SS",21,(text *) 0,
0,&ptemp->cr_date);


  pctx->w_id_ind = TRUE;
  pctx->w_id_len = SIZ(pPay->w_id);
  pctx->d_id_ind = TRUE;
  pctx->d_id_len = SIZ(pPay->d_id);
  pctx->c_w_id_ind = TRUE;
  pctx->c_w_id_len = 0;
  pctx->c_d_id_ind = TRUE;
  pctx->c_d_id_len = 0;
  pctx->c_id_ind = TRUE;
  pctx->c_id_len = 0;
  pctx->h_amount_len = SIZ(ptemp->h_amount);
  pctx->h_amount_ind = TRUE;
  pctx->c_last_ind = TRUE;
  pctx->c_last_len = SIZ(pPay->c_last);
  pctx->w_street_1_ind = NA;
  pctx->w_street_1_len = 0;
  pctx->w_street_2_ind = NA;
  pctx->w_street_2_len = 0;
  pctx->w_city_ind = NA;
  pctx->w_city_len = 0;
  pctx->w_state_ind = NA;
  pctx->w_state_len = 0;
  pctx->w_zip_ind = NA;
  pctx->w_zip_len = 0;
  pctx->d_street_1_ind = NA;
  pctx->d_street_1_len = 0;
  pctx->d_street_2_ind = NA;
  pctx->d_street_2_len = 0;
  pctx->d_city_ind = NA;
  pctx->d_city_len = 0;
  pctx->d_state_ind = NA;
  pctx->d_state_len = 0;
  pctx->d_zip_ind = NA;
  pctx->d_zip_len = 0;
  pctx->c_first_ind = NA;
  pctx->c_first_len = 0;
  pctx->c_middle_ind = NA;
  pctx->c_middle_len = 0;
  pctx->c_street_1_ind = NA;
  pctx->c_street_1_len = 0;
  pctx->c_street_2_ind = NA;
  pctx->c_street_2_len = 0;
  pctx->c_city_ind = NA;
  pctx->c_city_len = 0;
  pctx->c_state_ind = NA;
  pctx->c_state_len = 0;
  pctx->c_zip_ind = NA;
  pctx->c_zip_len = 0;
  pctx->c_phone_ind = NA;
  pctx->c_phone_len = 0;
  pctx->c_since_ind = NA;
  pctx->c_since_len = 0;
  pctx->c_credit_ind = NA;
  pctx->c_credit_len = 0;
```

```
  pctx->c_credit_lim_ind = NA;
  pctx->c_credit_lim_len = 0;
  pctx->c_discount_ind = NA;
  pctx->c_discount_len = 0;
  pctx->c_balance_ind = NA;
  pctx->c_balance_len = sizeof(double);
  pctx->c_data_ind = NA;
  pctx->c_data_len = 0;
  pctx->h_date_ind = TRUE;
  pctx->h_date_len = 0;
  pctx->retries_ind = TRUE;
  pctx->retries_len = 0;
  pctx->cr_date_ind = TRUE;
/*  pctx->cr_date_len = 7; */
  pctx->cr_date_len = sizeof(ptemp->cr_date);



  if(pPay->byname)
  {

    pctx->c_id_ind = NA;
    execstatus=OCIStmtExecute(p->tpcsvc,pctx->curp1,p->errhp,

1,0,0,0,OCI_DEFAULT|OCI_COMMIT_ON_SUCCESS);
  }
  else
  {
    pctx->c_last_ind = NA;
    execstatus=OCIStmtExecute(p->tpcsvc,pctx->curp0,p->errhp,

1,0,0,0,OCI_DEFAULT|OCI_COMMIT_ON_SUCCESS);
  }


  if(execstatus != OCI_SUCCESS) {
    OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
    errcode = OCIERROR(p,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      return(RECOVERR);
    } else if (errcode == RECOVERR) {
      return(RECOVERR);
    } else if (errcode == SNAPSHOT_TOO_OLD) {
      return(RECOVERR);
    } else {
      return ERR_DB_ERROR;
    }
  }

  return (ERR_DB_SUCCESS);

}



void tkvcpdone (payctx *ppctx)
{
  payctx pctx = *ppctx;

  if(NULL != pctx.curpi)
    OCIHandleFree((dvoid *)pctx.curpi,OCI_HTYPE_STMT);
  if(NULL != pctx.curp0)
    OCIHandleFree((dvoid *)pctx.curp0,OCI_HTYPE_STMT);
  if(NULL != pctx.curp1)
    OCIHandleFree((dvoid *)pctx.curp1,OCI_HTYPE_STMT);

}


/*
--------------------------------------------------------------------------------
Orderstatus transaction
*/

#define SQL_ORD_CUR0 "SELECT rowid FROM cust \
              WHERE c_d_id = :d_id AND c_w_id = :w_id AND c_last = :c_last \
              ORDER BY c_w_id, c_d_id, c_last, c_first"

#define SQL_ORD_CUR1 "SELECT c_id, c_balance, c_first, c_middle, c_last, \
              o_id, o_entry_d, o_carrier_id, o_ol_cnt \
              FROM cust, ordr \
              WHERE cust.rowid = :cust_rowid \
              AND   o_d_id = c_d_id AND o_w_id = c_w_id AND o_c_id = c_id \
              ORDER BY o_w_id, o_d_id, o_c_id, o_id DESC"


#define SQL_ORD_CUR2 "SELECT c_balance, c_first, c_middle, c_last, \
              o_id, o_entry_d, o_carrier_id, o_ol_cnt, c_id \
              FROM cust, ordr \
              WHERE c_id = :c_id AND c_d_id = :d_id AND c_w_id = :w_id \
              AND o_d_id = c_d_id AND o_w_id = c_w_id AND o_c_id = c_id \
              ORDER BY o_c_id, o_d_id, o_w_id, o_id DESC"


#define SQL_ORD_CUR3 "SELECT ol_i_id,ol_supply_w_id,ol_quantity,ol_amount,
\
              ol_delivery_d\
              FROM ordl \
              WHERE ol_d_id = :d_id AND ol_w_id = :w_id AND ol_o_id = :o_id"

#define SQL_ORD_CUR4 "SELECT count (c_last) FROM cust \
              WHERE c_d_id = :d_id AND c_w_id = :w_id AND c_last = :c_last "



sb4 rid_data(dvoid *ctxp, OCIDefine *dp, ub4 iter, dvoid **bufpp, ub4 **alenp,
        ub1 *piecep, dvoid **indpp, ub2 **rcodepp)
{
ub4 i;

ordctx *octx = &(((OraContext *)ctxp)->octx);
defctx *cbctx = &(((OraContext *)ctxp)->cbctx);

if (cbctx->reexec)  /* if this is the second execute - use entry 0 */
{
  i=0;
  cbctx->count--;   /* count down */
}
else
  i=iter;

*bufpp = octx->c_rowid_ptr[i];
*indpp = &octx->c_rowid_ind[i] ;
*alenp = &octx->c_rowid_len[i] ;
*rcodepp = &octx->c_rowid_rcode[i] ;
*piecep = OCI_ONE_PIECE;

return (OCI_CONTINUE);
}

tkvcoinit (OrderStatusData *pOrd,
                    OraContext *p)
{

  int i;
  text stmbuf[8192];
  ordtemp *otemp = &(p->tempvars.ord);
  ordctx *octx = &(p->octx);

  memset(octx,(char)0,sizeof(ordctx));
  octx->cs = 1;
  octx->norow = 0;
  octx->somerows = 10;
  octx->curo0 = NULL;
  octx->curo1 = NULL;
  octx->curo2 = NULL;
  octx->curo3 = NULL;
  octx->curo4 = NULL;

/* get the rowid handles */
  for(i=0;i<100;i++) {
    OCIERROR(p, OCIDescriptorAlloc(p->tpcenv,(dvoid**)&octx->c_rowid_ptr[i],

OCI_DTYPE_ROWID,0,(dvoid**)0));
  }


   OCIERROR(p,OCIHandleAlloc(p->tpcenv,(dvoid**)&octx-
>curo0,OCI_HTYPE_STMT,
                              0,(dvoid**)0));
```

```
  OCIERROR(p,OCIHandleAlloc(p->tpcenv,(dvoid**)&octx->curo1,OCI_HTYPE_STMT,
                                        0,(dvoid**)0));
  OCIERROR(p,OCIHandleAlloc(p->tpcenv,(dvoid**)&octx->curo2,OCI_HTYPE_STMT,
                                        0,(dvoid**)0));
  OCIERROR(p,OCIHandleAlloc(p->tpcenv,(dvoid**)&octx->curo3,OCI_HTYPE_STMT,
                                        0,(dvoid**)0));
  OCIERROR(p,OCIHandleAlloc(p->tpcenv,(dvoid**)&octx->curo4,OCI_HTYPE_STMT,
                                        0,(dvoid**)0));

/*  c_id = 0, use find customer by lastname. Get an array or rowid's back*/
  sprintf((char *) stmbuf, SQL_ORD_CUR0);
  OCIERROR(p,OCIStmtPrepare(octx->curo0,p->errhp,stmbuf,
                                        strlen((char *)stmbuf),
                                        OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(p,OCIAttrSet(octx->curo0,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
                                    OCI_ATTR_PREFETCH_ROWS,p->errhp));

/* get order/customer info back based on rowid */
  sprintf((char *) stmbuf, SQL_ORD_CUR1);
  OCIERROR(p,OCIStmtPrepare(octx->curo1,p->errhp,stmbuf,
                                        strlen((char *)stmbuf),
                                        OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(p,OCIAttrSet(octx->curo1,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
                                    OCI_ATTR_PREFETCH_ROWS,p->errhp));

/*  c_id != 0, use id to find customer  */
  sprintf((char *) stmbuf, SQL_ORD_CUR2);
  OCIERROR(p,OCIStmtPrepare(octx->curo2,p->errhp,stmbuf,
                                        strlen((char *)stmbuf),
                                        OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(p,OCIAttrSet(octx->curo2,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
                                    OCI_ATTR_PREFETCH_ROWS,p->errhp));

  sprintf((char *) stmbuf, SQL_ORD_CUR3);
  OCIERROR(p,OCIStmtPrepare(octx->curo3,p->errhp,stmbuf,
                                        strlen((char *)stmbuf),
                                        OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(p,OCIAttrSet(octx->curo3,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
                                    OCI_ATTR_PREFETCH_ROWS,p->errhp));

  sprintf((char *) stmbuf, SQL_ORD_CUR4);
  OCIERROR(p,OCIStmtPrepare(octx->curo4,p->errhp,stmbuf,
                                        strlen((char *)stmbuf),
                                        OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(p,OCIAttrSet(octx->curo4,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
                                    OCI_ATTR_PREFETCH_ROWS,p->errhp));

  for (i = 0; i < NITEMS; i++) {
    octx->ol_supply_w_id_ind[i] = TRUE;
    octx->ol_i_id_ind[i] = TRUE;
    octx->ol_quantity_ind[i] = TRUE;
    octx->ol_amount_ind[i] = TRUE;
    octx->ol_delivery_d_ind[i] = TRUE;

    octx->ol_supply_w_id_len[i] = sizeof(int);
    octx->ol_i_id_len[i] = sizeof(int);
    octx->ol_quantity_len[i] = sizeof(int);
    octx->ol_amount_len[i] = sizeof(int);
    octx->ol_delivery_d_len[i] = sizeof(OCIDate);
    /* octx->ol_delivery_d_len[i] = sizeof(pOrd->s_ol->ol_delivery_d); */
  }
  octx->ol_supply_w_id_csize = NITEMS;
  octx->ol_i_id_csize = NITEMS;
  octx->ol_quantity_csize = NITEMS;
  octx->ol_amount_csize = NITEMS;
  octx->ol_delivery_d_csize = NITEMS;
  octx->ol_w_id_csize = NITEMS;
  octx->ol_o_id_csize = NITEMS;
  octx->ol_d_id_csize = NITEMS;
  octx->ol_w_id_ind = TRUE;
  octx->ol_d_id_ind = TRUE;
  octx->ol_o_id_ind = TRUE;
  octx->ol_w_id_len = sizeof(int);
  octx->ol_d_id_len = sizeof(int);
  octx->ol_o_id_len = sizeof(int);

  /* bind variables */

  /* c_id (customer id) is not known */
  /* cursor 0 */
  OCIBND(octx->curo0,octx->w_id_bp0,p,":w_id",ADR(pOrd->w_id),SIZ(int),
              SQLT_INT);
  OCIBND(octx->curo0,octx->d_id_bp0,p,":d_id",ADR(pOrd->d_id),SIZ(int),
              SQLT_INT);
  OCIBND(octx->curo0,octx->c_last_bp,p,":c_last",pOrd->c_last,
              SIZ(pOrd->c_last),SQLT_STR);

    ocierror(__FILE__,__LINE__,p, \
    OCIDefineByPos(octx->curo0,&octx->c_rowid_dp,p->errhp,1, \
              octx->c_rowid_ptr, sizeof(OCIRowid*),SQLT_RDD,octx->c_rowid_ind,\
              NULL,NULL, OCI_DYNAMIC_FETCH));
    ocierror(__FILE__,__LINE__,p, OCIDefineDynamic(octx->c_rowid_dp,\
              p->errhp,\
              p,rid_data));

/** sth   OCIDFNDYN(  octx->curo0,
                      octx->c_rowid_dp,
                      p,
                      1,
                      octx->c_rowid_ptr,
                      sizeof(OCIRowid*),
                      SQLT_RDD,
                      octx->c_rowid_ind,
              p,
                      rid_data);
**/


  OCIBND(octx->curo1,octx->c_rowid_bp,p,":cust_rowid",
       &octx->middle_cust,sizeof(octx->middle_cust),SQLT_RDD);

  OCIDEF(octx->curo1,octx->c_id_dp,p->errhp,1,ADR(pOrd->c_id),SIZ(int),
              SQLT_INT);
  OCIDEF(octx->curo1,octx->c_balance_dp1,p->errhp,2,ADR(pOrd->c_balance),
              SIZ(double),SQLT_FLT);
  OCIDEF(octx->curo1,octx->c_first_dp1,p->errhp,3,pOrd->c_first,
              SIZ(pOrd->c_first)-1,
              SQLT_CHR);
  OCIDEF(octx->curo1,octx->c_middle_dp1,p->errhp,4,pOrd->c_middle,
              SIZ(pOrd->c_middle)-1,SQLT_AFC);

  OCIDEF(octx->curo1,octx->c_last_dp1,p->errhp,5,pOrd->c_last,
              SIZ(pOrd->c_last)-1, SQLT_CHR);
  OCIDEF(octx->curo1,octx->o_id_dp1,p->errhp,6,ADR(pOrd->o_id),SIZ(int),
              SQLT_INT);
  OCIDEF(octx->curo1,octx->o_entry_d_dp1,p->errhp,7,
              &otemp->entry_date,SIZ(otemp->entry_date),SQLT_ODT);
  OCIDEF(octx->curo1,octx->o_cr_id_dp1,p->errhp,8,ADR(pOrd->o_carrier_id),
              SIZ(int),SQLT_INT);
  OCIDEF(octx->curo1,octx->o_ol_cnt_dp1,p->errhp,9,ADR(pOrd->o_ol_cnt),
              SIZ(int),SQLT_INT);



/* Bind for cursor 2 , no-zero customer id */
  OCIBND(octx->curo2,octx->w_id_bp2,p,":w_id",ADR(pOrd->w_id),SIZ(int),
              SQLT_INT);
  OCIBND(octx->curo2,octx->d_id_bp2,p,":d_id",ADR(pOrd->d_id),SIZ(int),
              SQLT_INT);
  OCIBND(octx->curo2,octx->c_id_bp,p,":c_id",ADR(pOrd->c_id),SIZ(int),
              SQLT_INT);
  OCIDEF(octx->curo2,octx->c_balance_dp2,p->errhp,1,ADR(pOrd->c_balance),
         SIZ(double),SQLT_FLT);
  OCIDEF(octx->curo2,octx->c_first_dp2,p->errhp,2,pOrd->c_first,
              SIZ(pOrd->c_first)-1,
              SQLT_CHR);
  OCIDEF(octx->curo2,octx->c_middle_dp2,p->errhp,3,pOrd->c_middle,
        SIZ(pOrd->c_middle)-1,SQLT_AFC);
  OCIDEF(octx->curo2,octx->c_last_dp,p->errhp,4,pOrd->c_last,
              SIZ(pOrd->c_last)-1,
              SQLT_CHR);
```

```
  OCIDEF(octx->curo2,octx->o_id_dp2,p->errhp,5,ADR(pOrd->o_id),SIZ(int),
                SQLT_INT);
  OCIDEF(octx->curo2,octx->o_entry_d_dp2,p->errhp,6,
        &otemp->entry_date,SIZ(otemp->entry_date),SQLT_ODT);
  OCIDEF(octx->curo2, octx->o_cr_id_dp2,p->errhp,7,ADR(pOrd->o_carrier_id),
        SIZ(int), SQLT_INT);
  OCIDEF(octx->curo2,octx->o_ol_cnt_dp2,p->errhp,8,ADR(pOrd->o_ol_cnt),
        SIZ(int),SQLT_INT);
  OCIDEF(octx->curo2,octx->c_id_dp1,p->errhp,9,ADR(pOrd->c_id),SIZ(int),
        SQLT_INT);

/* Bind for last cursor - 3 */

  OCIBND(octx->curo3,octx->w_id_bp3,p,":w_id",ADR(pOrd->w_id),SIZ(int),
                SQLT_INT);
  OCIBND(octx->curo3,octx->d_id_bp3,p,":d_id",ADR(pOrd->d_id),SIZ(int),
                SQLT_INT);
  OCIBND(octx->curo3,octx->o_id_bp,p,":o_id",ADR(pOrd->o_id),SIZ(int),
                SQLT_INT);

  OCIDFNRA(octx->curo3, octx->ol_i_id_dp, p->errhp, 1, otemp->loc_ol_i_id,
                SIZ(int), SQLT_INT,
                octx->ol_i_id_ind,octx->ol_i_id_len, octx->ol_i_id_rcode);
  OCIDFNRA(octx->curo3,octx->ol_supply_w_id_dp,p->errhp,2,
                otemp->loc_ol_supply_w_id,
        SIZ(int),SQLT_INT, octx->ol_supply_w_id_ind,
        octx->ol_supply_w_id_len, octx->ol_supply_w_id_rcode);
  OCIDFNRA(octx->curo3, octx->ol_quantity_dp,p->errhp,3,
                otemp->loc_ol_quantity, SIZ(int),
        SQLT_INT, octx->ol_quantity_ind,octx->ol_quantity_len,
                octx->ol_quantity_rcode);
  OCIDFNRA(octx->curo3,octx->ol_amount_dp,p->errhp,4,otemp->loc_ol_amount,
                SIZ(int),
        SQLT_INT,octx->ol_amount_ind, octx->ol_amount_len,
        octx->ol_amount_rcode);
  OCIDFNRA(octx->curo3,octx->ol_d_base_dp,p->errhp,5,
                otemp->loc_ol_delivery_date,SIZ(OCIDate), SQLT_ODT,
        octx->ol_delivery_d_ind,octx->ol_delivery_d_len,
                octx->ol_delivery_d_rcode);

  OCIBND(octx->curo4, octx->w_id_bp4, p, ":w_id", ADR(pOrd->w_id), SIZ(int),
                        SQLT_INT);
  OCIBND(octx->curo4,octx->d_id_bp4,p,":d_id",ADR(pOrd->d_id),SIZ(int),
                SQLT_INT);
  OCIBND(octx->curo4,octx->c_last_bp4,p,":c_last",ADR(pOrd->c_last),
                SIZ(pOrd->c_last), SQLT_STR);
  OCIDEF(octx->curo4,
                        octx->c_count_dp,
                        p->errhp,
                        1,
                        ADR(octx->rcount),
                        SIZ(int),
                SQLT_INT);

 return (ERR_DB_SUCCESS);

}


tkvco (OrderStatusData *pOrd, OraContext *p)
{
 ordctx *octx = &(p->octx);
 defctx *cbctx = &(p->cbctx);
 ordtemp *otemp = &(p->tempvars.ord);
 int i;
 int execstatus;
 int errcode;
 int entry_date_str_len = sizeof (otemp->entry_date_str);


 int rcount;

 for (i = 0; i < NITEMS; i++) {
    octx->ol_supply_w_id_ind[i] = TRUE;
    octx->ol_i_id_ind[i] = TRUE;
    octx->ol_quantity_ind[i] = TRUE;
    octx->ol_amount_ind[i] = TRUE;
    octx->ol_delivery_d_ind[i] = TRUE;
    octx->ol_supply_w_id_len[i] = sizeof(int);
    octx->ol_i_id_len[i] = sizeof(int);
    octx->ol_quantity_len[i] = sizeof(int);
    octx->ol_amount_len[i] = sizeof(int);
    octx->ol_delivery_d_len[i] = sizeof(otemp->loc_ol_delivery_date[i]);
  }
  octx->ol_supply_w_id_csize = NITEMS;
  octx->ol_i_id_csize = NITEMS;
  octx->ol_quantity_csize = NITEMS;
  octx->ol_amount_csize = NITEMS;
  octx->ol_delivery_d_csize = NITEMS;

  /* initialize bound output variables to null for oracle v8 */
  /*   octx->o_ol_cnt_ind = NA;*/
  /*   pOrd->o_ol_cnt = 0;*/
  if(pOrd->byname)
    {
    cbctx->reexec = FALSE;
    execstatus=OCIStmtExecute(p->tpcsvc,octx->curo0,p->errhp,
                                                100,0,0,0,OCI_DEFAULT);
    if ((execstatus != OCI_NO_DATA) && (execstatus != OCI_SUCCESS))
                /* will get OCI_NO_DATA if <100 found */
    {
       errcode = OCIERROR(p,execstatus);
       if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR) ||
         (errcode == SNAPSHOT_TOO_OLD))
       {
          OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
                return RECOVERR;
       } else {
        return ERR_DB_ERROR;
       }
    }
    if (execstatus == OCI_NO_DATA)  /* there are no more rows */
    {
     /* get rowcount, find middle one */
    OCIAttrGet(octx-
>curo0,OCI_HTYPE_STMT,&rcount,NULL,OCI_ATTR_ROW_COUNT,
                        p->errhp);
    if (rcount <1)
    {
    TPCCErr("No Data Found");
    return ERR_DB_ERROR;
    }
    octx->cust_idx=(rcount-1)/2 ;
    }
    else
    {
    /* count  the number of rows */
    execstatus = OCIStmtExecute(p->tpcsvc,octx->curo4,p->errhp,
                                                1,0,0,0,OCI_DEFAULT);
    if ((execstatus != OCI_NO_DATA) && (execstatus != OCI_SUCCESS))
    {
     errcode = OCIERROR(p,execstatus);
     if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
       || (errcode == SNAPSHOT_TOO_OLD))
      {
       OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
                return RECOVERR;
      } else {
       return ERR_DB_ERROR;
      }
    }
    if (octx->rcount+1 < 200)
       octx->cust_idx=(octx->rcount-1)/2;
    else
    {
    cbctx->reexec = TRUE;
    cbctx->count = (octx->rcount+1)/2;
    execstatus = OCIStmtExecute(p->tpcsvc,octx->curo0,p->errhp,
                                                cbctx-
>count,0,0,0,OCI_DEFAULT);
    /* will get OCI_NO_DATA if <100 found */
    if (cbctx->count>0)
    {
    TPCCErr("Did not get all rows.");
    return ERR_DB_ERROR;
    }
    if ((execstatus != OCI_NO_DATA) && (execstatus != OCI_SUCCESS))
    {
     errcode=OCIERROR(p,execstatus);
     if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
```

```
        || (errcode == SNAPSHOT_TOO_OLD))
    {
      OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
                return RECOVERR;
     } else {
      return ERR_DB_ERROR;
     }
    }
   octx->cust_idx=0;
  }
 }

 octx->middle_cust=octx->c_rowid_ptr[octx->cust_idx];
 execstatus = OCIStmtExecute(p->tpcsvc,octx->curo1,p->errhp,
                                                 1,0,0,0,OCI_DEFAULT);
 if (execstatus != OCI_SUCCESS)
  {
    errcode = OCIERROR(p,execstatus);
    OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
    if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR) ||
         (errcode == SNAPSHOT_TOO_OLD))
    {
     return RECOVERR;
    } else {
     return ERR_DB_ERROR;
    }
  }
}
else
{
/*    execstatus = OCIStmtExecute(p->tpcsvc,octx->curo2,p->errhp,
                                                 1,0,0,0,OCI_DEFAULT); */
   execstatus = OCIStmtExecute(p->tpcsvc,octx->curo2,p->errhp,
                                                 1,0,0,0,OCI_DEFAULT);
   if (execstatus != OCI_SUCCESS)
    {
      errcode = OCIERROR(p,execstatus);
      OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
      if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
          || (errcode == SNAPSHOT_TOO_OLD))
      {
                return RECOVERR;
      } else {
        return ERR_DB_ERROR;
      }
    }
  }
  octx->ol_w_id_ind = TRUE;
  octx->ol_d_id_ind = TRUE;
  octx->ol_o_id_ind = TRUE;
  octx->ol_w_id_len = sizeof(int);
  octx->ol_d_id_len = sizeof(int);
  octx->ol_o_id_len = sizeof(int);

  execstatus =   OCIStmtExecute(p->tpcsvc,octx->curo3,p->errhp,pOrd->o_ol_cnt,
                                                 0,0,0, OCI_DEFAULT |
OCI_COMMIT_ON_SUCCESS);
  if (execstatus != OCI_SUCCESS)
    {
      errcode = OCIERROR(p,execstatus);
      OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
      if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
          || (errcode == SNAPSHOT_TOO_OLD))
      {
                return RECOVERR;
      } else {
        return ERR_DB_ERROR;
      }
    }

/*
#ifdef NOTMORE
  OCIERROR(errhp,
    OCITransCommit(tpcsvc,errhp,OCI_DEFAULT));
#endif
*/

  /* clean up and convert the delivery dates */

 for (i = 0; i < pOrd->o_ol_cnt; i++) {
    if (octx->ol_delivery_d_ind[i] == -1) /* null date in field */
                strncpy(otemp->ol_delivery_date_str[i],"01-01-1811",10);
    else
/*  cvtdmy(otemp->loc_ol_delivery_date[i], otemp->ol_delivery_date_str[i]); */
    {
     octx->ol_delivery_d_len[i]=sizeof(otemp->ol_delivery_date_str[i]);
     OCIERROR(p, OCIDateToText(p->errhp,&otemp->loc_ol_delivery_date[i],
       (text*)"dd-mm-yyyy",strlen("dd-mm-yyyy"),(text*)0,0,
        &octx->ol_delivery_d_len[i],(text*)&otemp->ol_delivery_date_str[i]));
    }

  }

 /* convert the order entry date */
 /* cvtdmyhms(otemp->entry_date, otemp->entry_date_str);  */
   OCIERROR(p, OCIDateToText(p->errhp,&otemp->entry_date,
        (text*)"dd-mm-yyyy HH24:MI:SS",strlen("dd-mm-yyyy
HH:MI:SS"),(text*)0,0,
         &entry_date_str_len,(text *)&otemp->entry_date_str));

 return (ERR_DB_SUCCESS);
 }


void tkvcodone (ordctx *poctx)
{
 ordctx octx = *poctx;

 if(NULL != octx.curo0)
   OCIHandleFree((dvoid *)octx.curo0,OCI_HTYPE_STMT);
 if(NULL != octx.curo1)
   OCIHandleFree((dvoid *)octx.curo1,OCI_HTYPE_STMT);
 if(NULL != octx.curo2)
   OCIHandleFree((dvoid *)octx.curo2,OCI_HTYPE_STMT);
 if(NULL != octx.curo3)
   OCIHandleFree((dvoid *)octx.curo3,OCI_HTYPE_STMT);
 if(NULL != octx.curo4)
   OCIHandleFree((dvoid *)octx.curo4,OCI_HTYPE_STMT);

}


/**** delivery transaction  */

#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
#define SQLTXT0 "SELECT substr(value,1,5) FROM v$parameter \
  WHERE name = 'instance_number'"
#endif



#define SQLTXT1 "DELETE FROM nord WHERE no_d_id = :d_id \
       AND no_w_id=:w_id and rownum <=1 \
                RETURNING no_o_id into :o_id "

#define SQLTXT3 "UPDATE ordr SET o_carrier_id = :carrier_id \
  WHERE o_id=:o_id and o_d_id=:d_id and o_w_id=:w_id \
  returning o_c_id into :o_c_id"

#define SQLTXT4 "UPDATE /*+ buffer */ ordl SET ol_delivery_d = :cr_date \
  WHERE ol_w_id=:w_id and ol_d_id=:d_id and ol_o_id=:o_id \
  RETURNING ol_amount into :ol_amount "


#define SQLTXT6 "UPDATE cust SET c_balance = c_balance + :amt, \
  c_delivery_cnt = c_delivery_cnt + 1 WHERE c_w_id = :w_id AND \
  c_d_id = :d_id AND c_id = :c_id"



sb4 no_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
            dvoid **bufpp, ub4 *alenp, ub1 *piecep,
            dvoid **indpp)
{
 *bufpp = (dvoid*)0;
 *alenp =0;
 *indpp = (dvoid*)0;
 *piecep =OCI_ONE_PIECE;
 return (OCI_CONTINUE);
}
```

```
sb4 TPC_oid_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
            dvoid **bufpp, ub4 **alenp, ub1 *piecep,
            dvoid **indpp, ub2 **rcodepp)
{

  delctx *dctx;
  dctx=(delctx*)ctxp;

  *bufpp = &dctx->del_o_id[iter];
  *indpp= &dctx->del_o_id_ind[iter];
  dctx->del_o_id_len[iter]=sizeof(dctx->del_o_id[0]);
  *alenp= &dctx->del_o_id_len[iter];
  *rcodepp = &dctx->del_o_id_rcode[iter];
  *piecep =OCI_ONE_PIECE;
  return (OCI_CONTINUE);
}

sb4 cid_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
            dvoid **bufpp, ub4 **alenp, ub1 *piecep,
            dvoid **indpp, ub2 **rcodepp)
{

  delctx *dctx;
  dctx=(delctx*)ctxp;

  *bufpp = &dctx->c_id[iter];
  *indpp= &dctx->c_id_ind[iter];
  dctx->c_id_len[iter]=sizeof(dctx->c_id[0]);
  *alenp= &dctx->c_id_len[iter];
  *rcodepp = &dctx->c_id_rcode[iter];
  *piecep =OCI_ONE_PIECE;
  return (OCI_CONTINUE);
}


sb4 amt_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
            dvoid **bufpp, ub4 **alenp, ub1 *piecep,
            dvoid **indpp, ub2 **rcodepp)
{


  amtctx *actx;
  actx =(amtctx*)ctxp;
  actx->ol_cnt[iter]=actx->ol_cnt[iter]+1;
  *bufpp = &actx->ol_amt[iter][index];
  *indpp= &actx->ol_amt_ind[iter][index];
  actx->ol_amt_len[iter][index]=sizeof(actx->ol_amt[0][0]);
  *alenp= &actx->ol_amt_len[iter][index];
  *rcodepp = &actx->ol_amt_rcode[iter][index];
  *piecep =OCI_ONE_PIECE;
  return (OCI_CONTINUE);
}



void shiftdata(from,dctx)
int from;
delctx *dctx;
{
   int i;
   for (i=from;i<NDISTS-1; i++)
   {
     dctx->del_o_id_ind[i] = dctx->del_o_id_ind[i+1];
     dctx->del_o_id[i] = dctx->del_o_id[i+1];
     dctx->w_id[i] = dctx->w_id[i+1];
     dctx->d_id[i] = dctx->d_id[i+1];
     dctx->carrier_id[i] = dctx->carrier_id[i+1];
   }
}    /* end shiftdata */


tkvcdinit (DeliveryData *pDel,
                  OraContext *p)
{
   delctx *dctx = &(p->dctx);
   amtctx *actx = &(p->actx);
   text stmbuf[8192];
   memset(dctx,(char)0,sizeof(delctx));
   memset(actx,(char)0,sizeof(amtctx));

   dctx->norow = 0;
   dctx->curd1 = NULL;
   dctx->curd2 = NULL;
   dctx->curd3 = NULL;
   dctx->curd4 = NULL;
   dctx->curd5 = NULL;
   dctx->curd6 = NULL;


#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
   OCIHandleAlloc(tpcenv, (dvoid **)(&dctx->curd0), OCI_HTYPE_STMT, 0,
(dvoid**)0);
   sprintf ((char *) stmbuf, SQLTXT0);
   OCIStmtPrepare(dctx->curd0, p->errhp, stmbuf, strlen((char
*)stmbuf),OCI_NTV_SYNTAX, OCI_DEFAULT);

   OCIDFNRA(dctx->curd0, dctx->inum_dp,p->errhp,1,dctx->inum,SIZ(dctx-
>inum),SQLT_STR,
        dctx->inum_ind,dctx->inum_len,dctx->inum_rcode);
#endif


   /* open first cursor */

   OCIHandleAlloc(p->tpcenv, (dvoid **)(&dctx->curd1), OCI_HTYPE_STMT,
                        0, (dvoid**)0);
   sprintf ((char *) stmbuf, SQLTXT1);
   OCIStmtPrepare(dctx->curd1, p->errhp, stmbuf, strlen((char *)stmbuf),
                        OCI_NTV_SYNTAX, OCI_DEFAULT);



   /* bind variables */

   OCIBND(dctx->curd1, dctx->w_id_bp,p,":w_id",dctx->w_id,SIZ(int),
              SQLT_INT);
   OCIBNDRA(dctx->curd1, dctx->d_id_bp,p,":d_id",dctx->d_id,SIZ(int),
              SQLT_INT,NULL,NULL,NULL);
   OCIBNDRAD(dctx->curd1, dctx->del_o_id_bp,p,":o_id",SIZ(int),
              SQLT_INT,NULL,dctx,no_data,TPC_oid_data);

   /* open third cursor */

   OCIHandleAlloc(p->tpcenv, (dvoid **)(&dctx->curd3), OCI_HTYPE_STMT, 0,
                        (dvoid**)0);
   sprintf ((char *) stmbuf, SQLTXT3);
   OCIStmtPrepare(dctx->curd3, p->errhp, stmbuf, strlen((char *)stmbuf),
                        OCI_NTV_SYNTAX, OCI_DEFAULT);


   /* bind variables */

   OCIBNDRA(dctx->curd3, dctx->carrier_id_bp,p,":carrier_id",dctx->carrier_id,
        SIZ(int),SQLT_INT,dctx->carrier_id_ind,
        dctx->carrier_id_len,dctx->carrier_id_rcode);
   OCIBNDRA(dctx->curd3, dctx->w_id_bp3,p,":w_id",dctx->w_id,
              SIZ(int),SQLT_INT,NULL,NULL,NULL);
   OCIBNDRA(dctx->curd3, dctx->d_id_bp3,p,":d_id",dctx->d_id,
              SIZ(int),SQLT_INT,NULL,NULL,NULL);
   OCIBNDRA(dctx->curd3, dctx->del_o_id_bp3,p,":o_id",dctx->del_o_id,
              SIZ(int),SQLT_INT,NULL,NULL,NULL);

   OCIBNDRAD(dctx->curd3, dctx->c_id_bp3,p,":o_c_id",SIZ(int),
              SQLT_INT,NULL,dctx,no_data,cid_data);


   /* open fourth cursor */

   OCIHandleAlloc(p->tpcenv, (dvoid **)(&dctx->curd4), OCI_HTYPE_STMT, 0,
                        (dvoid**)0);
   sprintf ((char *) stmbuf, SQLTXT4);
   OCIStmtPrepare(dctx->curd4, p->errhp, stmbuf, strlen((char *)stmbuf),
                        OCI_NTV_SYNTAX, OCI_DEFAULT);

   /* bind variables */

   OCIBND(dctx->curd4, dctx->w_id_bp4,p,":w_id",dctx->w_id,SIZ(dctx->w_id[0]),
```

```
                SQLT_INT);
  OCIBND(dctx->curd4, dctx->d_id_bp4,p,":d_id",dctx->d_id,SIZ(dctx->d_id[0]),
                SQLT_INT);
  OCIBND(dctx->curd4, dctx->o_id_bp,p,":o_id",dctx->del_o_id,SIZ(int),
                SQLT_INT);
  OCIBND(dctx->curd4, dctx->cr_date_bp,p,":cr_date",dctx->del_date,
                SIZ(OCIDate), SQLT_ODT);

  OCIBNDRAD(dctx->curd4, dctx->olamt_bp,p,":ol_amount",SIZ(int),
               SQLT_INT,NULL,actx,no_data,amt_data);


  /* open sixth cursor */

  OCIHandleAlloc(p->tpcenv, (dvoid **)(&dctx->curd6), OCI_HTYPE_STMT, 0,
                           (dvoid**)0);
  sprintf ((char *) stmbuf, SQLTXT6);
  OCIStmtPrepare(dctx->curd6, p->errhp, stmbuf, strlen((char *)stmbuf),
                           OCI_NTV_SYNTAX, OCI_DEFAULT);

  /* bind variables */

  OCIBND(dctx->curd6, dctx->amt_bp,p,":amt",dctx->amt,SIZ(int),
                SQLT_INT);
  OCIBND(dctx->curd6, dctx->w_id_bp6,p,":w_id",dctx->w_id,SIZ(int),
                SQLT_INT);
  OCIBND(dctx->curd6, dctx->d_id_bp6,p,":d_id",dctx->d_id,SIZ(int),
                SQLT_INT);
  OCIBND(dctx->curd6, dctx->c_id_bp,p,":c_id",dctx->c_id,SIZ(int),
                SQLT_INT);


  return (ERR_DB_SUCCESS);

}


tkvcd (DeliveryData *pDel, OraContext *p)
{
 delctx *dctx = &(p->dctx);
 amtctx *actx = &(p->actx);
 deltemp *dtemp = &(p->tempvars.del);
 int i, j;
 int rpc,rcount,count;
 int invalid;
 unsigned char localcr_date[7];
 OCIError *datecvterrhp = p->datecvterrhp;
 int execstatus;
 int errcode;
 int proc_no = 0;

 invalid = 0;

 vgetdate(localcr_date);
 cvtdmyhms(localcr_date,dtemp->cvtcr_date);
 OCIDateFromText(datecvterrhp,(text *)dtemp->cvtcr_date,strlen(dtemp-
>cvtcr_date),(text *)"DD-MM-YYYY HH24:MI:SS",21,(text *) 0, 0,&dtemp-
>cr_date);



#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
  int hasno;
  int reread;
  char sdate[30];

  OCIStmtExecute(p->tpcsvc,dctx->curd0,p->errhp,1,0,0,0,OCI_DEFAULT);
  sysdate(sdate);
  printf ("Delivery started at %s on node %s\n",sdate,dctx->inum);
#endif


#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
 reread = 1;
#endif

#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
iso:
#endif

 /* initialization for array operations */

 for (i = 0; i < NDISTS; i++) {
   dctx->del_o_id_ind[i] = TRUE;
   dctx->cons_ind[i] = TRUE;
   dctx->w_id_ind[i] = TRUE;
   dctx->d_id_ind[i] = TRUE;
   dctx->c_id_ind[i] = TRUE;
   dctx->del_date_ind[i] = TRUE;
   dctx->carrier_id_ind[i] = TRUE;
   dctx->amt_ind[i] = TRUE;
   dctx->no_rowid_ind[i] = TRUE;
   dctx->o_rowid_ind[i] = TRUE;

   dctx->del_o_id_len[i] = SIZ(dctx->del_o_id[0]);
   dctx->cons_len[i] = SIZ(dctx->cons[0]);
   dctx->w_id_len[i] = SIZ(dctx->w_id[0]);
   dctx->d_id_len[i] = SIZ(dctx->d_id[0]);
   dctx->c_id_len[i] = SIZ(dctx->c_id[0]);
   dctx->del_date_len[i] = DEL_DATE_LEN;
   dctx->carrier_id_len[i] = SIZ(dctx->carrier_id[0]);
   dctx->amt_len[i] = SIZ(dctx->amt[0]);
   dctx->no_rowid_len[i] = ROWIDLEN;
   dctx->o_rowid_len[i] = ROWIDLEN;
   dctx->o_rowid_ptr_len[i] = SIZ(dctx->o_rowid_ptr[0]);
   dctx->no_rowid_ptr_len[i] = SIZ(dctx->no_rowid_ptr[0]);

   dctx->w_id[i] = pDel->w_id;
   dctx->d_id[i] = i+1;
   dctx->carrier_id[i] = pDel->o_carrier_id;
   memcpy(&dctx->del_date[i],&dtemp->cr_date,sizeof(OCIDate));

   actx->ol_cnt[i]=0;
 }


 /* array select from new_order and orders tables */

 execstatus=OCIStmtExecute(p->tpcsvc,dctx->curd1,p->errhp,NDISTS,0,0,0,
                                                 OCI_DEFAULT);
 if((execstatus != OCI_SUCCESS) && (execstatus != OCI_NO_DATA))
 {
   OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
   errcode = OCIERROR(p,execstatus);
   if(errcode == NOT_SERIALIZABLE)
   {
     return(RECOVERR);
   }
   else if (errcode == RECOVERR)
   {
     return(RECOVERR);
   }
   else if (errcode == SNAPSHOT_TOO_OLD)
   {
     return(RECOVERR);
   }
   else
   {
      return (ERR_DB_ERROR);
   }
 }


 /* mark districts with no new order */
 OCIAttrGet(dctx-
>curd1,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,p->errhp);
 rpc = rcount;

 if (rcount != NDISTS)
 {
   int j=0;
   for (i=0;i<NDISTS;i++)
   {
     if (dctx->del_o_id_ind[j] == 0)   /* there is data here  */
       j++;
     else
       shiftdata(j,dctx);
   }
 }
```

```
#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
  if (invalid)
  {
   sysdate(sdate);
   for (i=1;i<=NDISTS;i++)
   {
    hasno=0;
    for (j=0;j<rpc;j++)
    {
     if (dctx->d_id[j] == i)
     {
      hasno=1;
      break;
     }
    }
    if (!hasno)
      printf ("Delivery [dist %d] found no new order at %s\n",i,sdate);
   }
   if (reread)
   {
    sleep (60);
    sysdate(sdate);
    printf ("Delivery wake up at %s\n",sdate);
    reread=0;
    goto iso;
   }
  }   /* end if (invalid) */
#endif



   execstatus=OCIStmtExecute(p->tpcsvc,dctx->curd3,p->errhp,rpc,0,0,0,
                                                OCI_DEFAULT);
   if(execstatus != OCI_SUCCESS)
   {
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     errcode = OCIERROR(p,execstatus);
     if(errcode == NOT_SERIALIZABLE)
     {
       return(RECOVERR);
     }
     else if (errcode == RECOVERR)
     {
       return (RECOVERR);
     }
     else if (errcode == SNAPSHOT_TOO_OLD)
     {
       return (RECOVERR);
     }
     else
     {
       return (ERR_DB_ERROR);
     }
   }


   OCIAttrGet(dctx-
>curd3,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,p->errhp);

   if (rcount != rpc)
   {
     TPCCErr( "Error in TPC-C server %d: %d rows selected, %d ords updated\n",
            proc_no, rpc, rcount);
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     return (ERR_DB_ERROR);
   }

   /* array update of order_line table */
   execstatus=OCIStmtExecute(p->tpcsvc,dctx->curd4,p->errhp,rpc,0,0,0,
                                                OCI_DEFAULT);
   if(execstatus != OCI_SUCCESS)
   {
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     errcode = OCIERROR(p,execstatus);
     if(errcode == NOT_SERIALIZABLE)
     {
       return(RECOVERR);
     }
     else if (errcode == RECOVERR)
     {
       return(RECOVERR);
     }
     else if (errcode == SNAPSHOT_TOO_OLD)
     {
       return(RECOVERR);
     }
     else
     {
       return (ERR_DB_ERROR);
     }
   }

   OCIAttrGet(dctx-
>curd4,OCI_HTYPE_STMT,&rcount,NULL,OCI_ATTR_ROW_COUNT,p->errhp);
   /* add up amounts */
   count=0;
   for (i=0;i<rpc;i++)
   {
    dctx->amt[i]=0;
    for (j=0;j<actx->ol_cnt[i];j++)
      if (actx->ol_amt_rcode[i][j] == 0)
      {
        dctx->amt[i] = dctx->amt[i] + actx->ol_amt[i][j];
        count = count+1;
      }
   }
   if (rcount > rpc*NITEMS)
   {
     TPCCErr( "Error in TPC-C server %d: %d ordnrs updated, %d ordl updated\n",
            proc_no, rpc, rcount);
   }


#if defined(ISO5) || defined(ISO6)
   printf 9"d_id:amount\n";
   for (i=0;i<rpc;i++)
     printf ("%d:%.2f ", dctx->d_id[i], (float)dctx->amt[i]/100);
   printf ("\n");
#endif


   /* array update of customer table */

#if defined(ISO5) || defined(ISO6)
   execstatus=OCIStmtExecute(p->tpcsvc,dctx->curd6,p->errhp,rpc,0,0,0,
                                OCI_DEFAULT);
#else
   execstatus=OCIStmtExecute(p->tpcsvc,dctx->curd6,p->errhp,rpc,0,0,0,
                                OCI_COMMIT_ON_SUCCESS | OCI_DEFAULT);
#endif



   if(execstatus != OCI_SUCCESS)
   {
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     errcode = OCIERROR(p,execstatus);
     if(errcode == NOT_SERIALIZABLE)
     {
       return (RECOVERR);
     }
     else if (errcode == RECOVERR)
     {
       return (RECOVERR);
     }
     else if (errcode == SNAPSHOT_TOO_OLD)
     {
       return (RECOVERR);
     }
     else
     {
       return (ERR_DB_ERROR);
     }
   }

   OCIAttrGet(dctx-
>curd6,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,p->errhp);

   if (rcount != rpc) {
     TPCCErr ("Error in TPC-C server %d: %d rows selected, %d cust updated\n",
```

```
            proc_no, rpc, rcount);
    OCITransRollback(p->tpcsvc, p->errhp, OCI_DEFAULT);
    return (ERR_DB_ERROR);
  }


#if defined(ISO5) || defined(ISO6)
  sysdate;

#ifdef ISO5
  printf ("Delivery sleep before commit at %s\n"sdate);
#else
  printf ("Delivery sleep before abort at %s\n"sdate);
#endif
  sleep (60);
  sysdate (sdate);
  printf ("Delivery wake up at %s\n",sdate);
#endif

#ifdef ISO6
  printf ("Delivery ISO6 is rolling back.\n";
  OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
#endif

#ifdef ISO5
  OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
#endif

#if defined(ISO5) || defined(ISO6)
  sysdate(sdate);
  printf ("Delivery completed at: %s\n",sdate);
#endif


  /* return o_id's in district id order */

  for (i = 0; i < NDISTS; i++)
    pDel->o_id[i] = 0;
  for (i = 0; i < rpc; i++)
    pDel->o_id[dctx->d_id[i] - 1] = dctx->del_o_id[i];

  return (ERR_DB_SUCCESS);

}



void tkvcddone (delctx *pdctx)
{
 delctx dctx = *pdctx;

#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
 OCIHandleFree((dvoid *)dctx->curd0,OCI_HTYPE_STMT);
#endif

 if(NULL != dctx.curd1)
   OCIHandleFree((dvoid *)dctx.curd1,OCI_HTYPE_STMT);
 if(NULL != dctx.curd2)
   OCIHandleFree((dvoid *)dctx.curd2,OCI_HTYPE_STMT);
 if(NULL != dctx.curd3)
   OCIHandleFree((dvoid *)dctx.curd3,OCI_HTYPE_STMT);
 if(NULL != dctx.curd4)
   OCIHandleFree((dvoid *)dctx.curd4,OCI_HTYPE_STMT);
 if(NULL != dctx.curd5)
   OCIHandleFree((dvoid *)dctx.curd5,OCI_HTYPE_STMT);
 if(NULL != dctx.curd6)
   OCIHandleFree((dvoid *)dctx.curd6,OCI_HTYPE_STMT);
}

/*
--------------------------------------------------------------------------
NEW ORDER TRANSACTION
--------------------------------------------------------------------------
*/

#define NOSQLTXT2 "BEGIN initnew.new_init(:idx1arr); END;"

int tkvcninit (NewOrderData *pNew,
               OraContext *p)
{
 newctx *nctx = &(p->nctx);
 newtemp *ntemp = &(p->tempvars.new);
 int i;
 int execstatus;
 int errcode;
 text stmbuf[16384];

 memset(nctx,(char)0,sizeof(newctx));
 nctx->cs = 1;
 nctx->norow=0;
 nctx->curn1 = NULL;
 nctx->curn2 = NULL;
 for (i = 0; i < 100; i++)
  (nctx->curn3)[i] = NULL;
 nctx->curn4 = NULL;

 for(i=0;i<NITEMS;i++) {
  OCIERROR(p, OCIDescriptorAlloc(p->tpcenv,
                                 (dvoid**)&nctx->s_rowid_ptr[i],
OCI_DTYPE_ROWID,0,(dvoid**)0));
 }
 nctx->w_id_ind = TRUE;
 nctx->w_id_len = sizeof(pNew->w_id);
 nctx->d_id_ind = TRUE;
 nctx->d_id_len = sizeof(pNew->d_id);
 nctx->c_id_ind = TRUE;
 nctx->c_id_len = sizeof(pNew->c_id);
 nctx->o_all_local_ind = TRUE;
 nctx->o_all_local_len = sizeof(pNew->o_all_local);
 nctx->o_ol_cnt_ind = TRUE;
 nctx->o_ol_cnt_len = sizeof(pNew->o_ol_cnt);
 nctx->w_tax_ind = TRUE;
 nctx->w_tax_len = 0;
 nctx->d_tax_ind = TRUE;
 nctx->d_tax_len = 0;
 nctx->o_id_ind = TRUE;
 nctx->o_id_len = sizeof(pNew->o_id);
 nctx->c_discount_ind = TRUE;
 nctx->c_discount_len = 0;
 nctx->c_credit_ind = TRUE;
 nctx->c_credit_len = 0;
 nctx->c_last_ind = TRUE;
 nctx->c_last_len = 0;
 nctx->retries_ind = TRUE;
 nctx->retries_len = sizeof(ntemp->n_retry);
 nctx->cr_date_ind = TRUE;
 nctx->cr_date_len = sizeof(ntemp->cr_date);

 /* open first cursor */
 OCIERROR(p,OCIHandleAlloc(p->tpcenv,(dvoid **)(&nctx->curn1),
                           OCI_HTYPE_STMT, 0,
(dvoid**)0));
 OCIERROR(p,OCIStmtPrepare(nctx->curn1, p->errhp, stmbuf_tkvcpnew,
                           strlen((char *)stmbuf_tkvcpnew),
                           OCI_NTV_SYNTAX,
OCI_DEFAULT));

 /* bind variables */

 OCIBNDR(nctx->curn1, nctx->w_id_bp, p, ":w_id",ADR(pNew->w_id),
          SIZ(pNew->w_id),
      SQLT_INT, &nctx->w_id_ind, &nctx->w_id_len, &nctx->w_id_rc);
 OCIBNDR(nctx->curn1, nctx->d_id_bp, p, ":d_id",ADR(pNew->d_id),
          SIZ(pNew->d_id),
      SQLT_INT, &nctx->d_id_ind, &nctx->d_id_len, &nctx->d_id_rc);
 OCIBNDR(nctx->curn1, nctx->c_id_bp, p, ":c_id",ADR(pNew->c_id),
          SIZ(pNew->c_id),
      SQLT_INT, &nctx->c_id_ind, &nctx->c_id_len, &nctx->c_id_rc);
 OCIBNDR(nctx->curn1, nctx->o_all_local_bp, p, ":o_all_local",
      ADR(pNew->o_all_local), SIZ(pNew->o_all_local),SQLT_INT,
          &nctx->o_all_local_ind,
      &nctx->o_all_local_len, &nctx->o_all_local_rc);
 OCIBNDR(nctx->curn1, nctx->o_all_cnt_bp, p, ":o_ol_cnt",
      ADR(pNew->o_ol_cnt), SIZ(pNew->o_ol_cnt),SQLT_INT,
          &nctx->o_ol_cnt_ind, &nctx->o_ol_cnt_len, &nctx->o_ol_cnt_rc);
```

```
  OCIBNDR(nctx->curn1, nctx->w_tax_bp, p, ":w_tax",ADR(ntemp->w_tax),
                SIZ(ntemp->w_tax),
        SQLT_FLT, &nctx->w_tax_ind, &nctx->w_tax_len, &nctx->w_tax_rc);
  OCIBNDR(nctx->curn1, nctx->d_tax_bp, p, ":d_tax",ADR(ntemp->d_tax),
                SIZ(ntemp->d_tax),
        SQLT_FLT, &nctx->d_tax_ind, &nctx->d_tax_len, &nctx->d_tax_rc);
  OCIBNDR(nctx->curn1, nctx->o_id_bp, p, ":o_id",ADR(pNew->o_id),
                SIZ(pNew->o_id),
        SQLT_INT, &nctx->o_id_ind, &nctx->o_id_len, &nctx->o_id_rc);
  OCIBNDR(nctx->curn1, nctx->c_discount_bp, p, ":c_discount",
        ADR(ntemp->c_discount), SIZ(ntemp->c_discount),SQLT_FLT,
                &nctx->c_discount_ind, &nctx->c_discount_len, &nctx-
>c_discount_rc);
  OCIBNDR(nctx->curn1, nctx->c_credit_bp, p, ":c_credit",pNew->c_credit,
        SIZ(pNew->c_credit),SQLT_CHR,
                &nctx->c_credit_ind, &nctx->c_credit_len, &nctx->c_credit_rc);
  OCIBNDR(nctx->curn1, nctx->c_last_bp, p, ":c_last",pNew->c_last,
                SIZ(pNew->c_last),
        SQLT_STR, &nctx->c_last_ind, &nctx->c_last_len, &nctx->c_last_rc);
  OCIBNDR(nctx->curn1, nctx->retries_bp, p, ":retry",ADR(ntemp->n_retry),
        SIZ(ntemp->n_retry),SQLT_INT,
                &nctx->retries_ind, &nctx->retries_len, &nctx->retries_rc);
  OCIBNDR(nctx->curn1, nctx->cr_date_bp, p, ":cr_date",ADR(ntemp->cr_date),
        SIZ(ntemp->cr_date), SQLT_ODT,
                &nctx->cr_date_ind, &nctx->cr_date_len, &nctx->cr_date_rc);

  OCIBNDRAA(nctx->curn1, nctx->ol_i_id_bp, p, ":ol_i_id",ntemp->nol_i_id,
        SIZ(int), SQLT_INT, nctx->nol_i_id_ind, nctx->nol_i_id_len,
                nctx->nol_i_id_rcode, NITEMS,&nctx->nol_i_count);

  OCIBNDRAA(nctx->curn1, nctx->ol_supply_w_id_bp, p, ":ol_supply_w_id",
                ntemp->nol_supply_w_id, SIZ(int), SQLT_INT,
                nctx->nol_supply_w_id_ind, nctx->nol_supply_w_id_len,
                nctx->nol_supply_w_id_rcode, NITEMS,&nctx->nol_s_count);

  OCIBNDRAA(nctx->curn1, nctx->ol_quantity_bp, p, ":ol_quantity",
                ntemp->nol_quantity, SIZ(int), SQLT_INT, nctx->nol_quantity_ind,
                nctx->nol_quantity_len, nctx->nol_quantity_rcode, NITEMS,
                &nctx->nol_q_count);

  OCIBNDRAA(nctx->curn1, nctx->i_price_bp, p, ":i_price",
                ntemp->i_price, SIZ(int), SQLT_INT, nctx->i_price_ind,
                nctx->i_price_len, nctx->i_price_rcode, NITEMS,
                &nctx->nol_item_count);

  OCIBNDRAA(nctx->curn1, nctx->i_name_bp, p, ":i_name",
                ntemp->i_name, SIZ(pNew->o_ol[0].i_name), SQLT_STR,
                nctx->i_name_ind,
                nctx->i_name_len, nctx->i_name_rcode, NITEMS,
                &nctx->nol_name_count);

  OCIBNDRAA(nctx->curn1, nctx->s_quantity_bp, p, ":s_quantity",
                ntemp->s_quantity, SIZ(int), SQLT_INT, nctx->s_quant_ind,
                nctx->s_quant_len, nctx->s_quant_rcode, NITEMS,
                &nctx->nol_qty_count);

  OCIBNDRAA(nctx->curn1, nctx->s_bg_bp, p, ":brand_generic",
                ntemp->brand_generic, SIZ(char), SQLT_CHR, nctx->s_bg_ind,
                nctx->s_bg_len, nctx->s_bg_rcode, NITEMS,
                &nctx->nol_bg_count);

  OCIBNDRAA(nctx->curn1, nctx->ol_amount_bp, p, ":ol_amount",
                ntemp->nol_amount, SIZ(int), SQLT_INT, nctx->nol_amount_ind,
                nctx->nol_amount_len, nctx->nol_amount_rcode, NITEMS,
                &nctx->nol_am_count);

  OCIBNDRAA(nctx->curn1, nctx->s_remote_bp, p, ":s_remote",
                nctx->s_remote, SIZ(int), SQLT_INT, nctx->s_remote_ind,
                nctx->s_remote_len, nctx->s_remote_rcode, NITEMS,
                &nctx->s_remote_count);



  /* open second cursor */
  OCIERROR(p,OCIHandleAlloc(p->tpcenv, (dvoid **)(&nctx->curn2),
                                                OCI_HTYPE_STMT, 0,
(dvoid**)0));
  sprintf ((char *) stmbuf, NOSQLTXT2);
  OCIERROR(p,OCIStmtPrepare(nctx->curn2, p->errhp, stmbuf,
        strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));


/* execute second cursor to init newinit package */
{
  int idx1arr[NITEMS];
  OCIBind *idx1arr_bp;
  ub2 idx1arr_len[NITEMS];
  ub2 idx1arr_rcode[NITEMS];
  sb2 idx1arr_ind[NITEMS];
  ub4 idx1arr_count;
  ub2 idx;

  for (idx=0;idx<NITEMS;idx++)
  {
   idx1arr[idx] = idx + 1;
   idx1arr_ind[idx] = TRUE;
   idx1arr_len[idx] = sizeof(int);
  }
  idx1arr_count=NITEMS;
  pNew->o_ol_cnt=NITEMS;

/* Bind array */
  OCIBNDRAA(nctx->curn2,idx1arr_bp,p,":idx1arr",idx1arr,SIZ(int),SQLT_INT,
    idx1arr_ind,idx1arr_len,idx1arr_rcode,NITEMS,&idx1arr_count);

  execstatus = OCIStmtExecute(p->tpcsvc,nctx->curn2,p->errhp,1,0,0,0,
                                                OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS)
  {
   OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
   errcode = OCIERROR(p,execstatus);
   return ERR_DB_ERROR;
  }
}


return (ERR_DB_SUCCESS);

}



int tkvcn (NewOrderData *pNew, OraContext *p)
{
  int statusCnt;
  int execstatus;
  int errcode;
  newctx *nctx = &(p->nctx);
  newtemp *ntemp = &(p->tempvars.new);

  int proc_no = 0;
  int retries = 0;
  int i;
  int rcount;


  statusCnt = 0;                    /* number of invalid items */

   /* get number of order lines, and check if all are local */
  /*  this section of code is needed for cprime, but we have
      already calculated the number of order lines back in web_ui
      no need to recalculate. */
  /*  pNew->o_ol_cnt = NITEMS;
  pNew->o_all_local = 1;
  for (i = 0; i < NITEMS; i++) {
    if (ntemp->nol_i_id[i] == 0) {
      pNew->o_ol_cnt = i;
      break;
    }
    */

  for (i = 0; i < pNew->o_ol_cnt; i++) {
    if (ntemp->nol_supply_w_id[i] != pNew->w_id) {
      nctx->s_remote[i] = 1;
      pNew->o_all_local = 0;
    }
    else
      nctx->s_remote[i] = 0;
```

```
}

nctx->w_id_ind = TRUE;
nctx->w_id_len = sizeof(pNew->w_id);
nctx->d_id_ind = TRUE;
nctx->d_id_len = sizeof(pNew->d_id);
nctx->c_id_ind = TRUE;
nctx->c_id_len = sizeof(pNew->c_id);
nctx->o_all_local_ind = TRUE;
nctx->o_all_local_len = sizeof(pNew->o_all_local);
nctx->o_ol_cnt_ind = TRUE;
nctx->o_ol_cnt_len = sizeof(pNew->o_ol_cnt);
nctx->w_tax_ind = TRUE;
nctx->w_tax_len = 0;
nctx->d_tax_ind = TRUE;
nctx->d_tax_len = 0;
nctx->o_id_ind = TRUE;
nctx->o_id_len = sizeof(pNew->o_id);
nctx->c_discount_ind = TRUE;
nctx->c_discount_len = 0;
nctx->c_credit_ind = TRUE;
nctx->c_credit_len = 0;
nctx->c_last_ind = TRUE;
nctx->c_last_len = 0;
nctx->retries_ind = TRUE;
nctx->retries_len = sizeof(retries);
nctx->cr_date_ind = TRUE;
nctx->cr_date_len = sizeof(ntemp->cr_date);



/* this is the row count */
rcount = pNew->o_ol_cnt;
nctx->nol_i_count = pNew->o_ol_cnt;
nctx->nol_q_count = pNew->o_ol_cnt;
nctx->nol_s_count = pNew->o_ol_cnt;
nctx->s_remote_count = pNew->o_ol_cnt;

nctx->nol_qty_count  = 0;
nctx->nol_bg_count = 0;
nctx->nol_item_count = 0;
nctx->nol_name_count = 0;
nctx->nol_am_count   = 0;
/* following not relevant */
nctx->s_data_count = pNew->o_ol_cnt;
nctx->i_data_count = pNew->o_ol_cnt;

/* initialization for array operations */
for (i = 0; i < pNew->o_ol_cnt; i++) {
  nctx->ol_w_id[i] = pNew->w_id;
  nctx->ol_d_id[i] = pNew->d_id;
  nctx->ol_number[i] = i + 1;
  nctx->null_date_ind[i]= TRUE;
  nctx->nol_i_id_ind[i] = 0;
  nctx->nol_supply_w_id_ind[i] = TRUE;
  nctx->nol_quantity_ind[i] = TRUE;
  nctx->nol_amount_ind[i] = TRUE;
  nctx->ol_w_id_ind[i] = TRUE;
  nctx->ol_d_id_ind[i] = TRUE;
  nctx->ol_o_id_ind[i] = TRUE;
  nctx->ol_number_ind[i] = TRUE;
  nctx->ol_dist_info_ind[i] = TRUE;
  nctx->s_remote_ind[i] = TRUE;
  nctx->s_data_ind[i] = TRUE;
  nctx->i_data_ind[i] = TRUE;
  nctx->s_quant_ind[i] = TRUE;
  nctx->s_bg_ind[i] = TRUE;
  nctx->cons_ind[i] = TRUE;
  nctx->s_rowid_ind[i] = TRUE;
  nctx->nol_i_id_len[i] = sizeof(int);
  nctx->nol_supply_w_id_len[i] = sizeof(int);
  nctx->nol_quantity_len[i] = sizeof(int);
  nctx->nol_amount_len[i] = sizeof(int);
  nctx->ol_w_id_len[i] = sizeof(int);
  nctx->ol_d_id_len[i] = sizeof(int);
  nctx->ol_o_id_len[i] = sizeof(int);
  nctx->ol_number_len[i] = sizeof(int);
  nctx->ol_dist_info_len[i] = nctx->s_dist_info_len[i];
  nctx->null_date_len[i]=sizeof(OCIDate);
  nctx->s_remote_len[i] = sizeof(int);
  nctx->s_data_len[i] = sizeof(int);
  nctx->i_data_len[i] = sizeof(int);
  nctx->s_quant_len[i] = sizeof(int);
  nctx->s_rowid_len[i] = sizeof(nctx->s_rowid_ptr[0]);
  nctx->cons_len[i] = sizeof(int);
  nctx->i_name_len[i]=0;
  nctx->s_bg_len[i] = 0;
 }
 for (i = pNew->o_ol_cnt; i < NITEMS; i++) {
  nctx->nol_i_id_ind[i] = NA;
  nctx->nol_supply_w_id_ind[i] = NA;
  nctx->nol_quantity_ind[i] = NA;
  nctx->nol_amount_ind[i] = NA;
  nctx->ol_w_id_ind[i] = NA;
  nctx->ol_d_id_ind[i] = NA;
  nctx->ol_o_id_ind[i] = NA;
  nctx->ol_number_ind[i] = NA;
  nctx->ol_dist_info_ind[i] = NA;
  nctx->null_date_ind[i]= NA;
  nctx->s_remote_ind[i] = NA;
  nctx->s_data_ind[i] = NA;
  nctx->i_data_ind[i] = NA;
  nctx->s_quant_ind[i] = NA;
  nctx->s_bg_ind[i] = NA;
  nctx->cons_ind[i] = NA;
  nctx->s_rowid_ind[i] = NA;

  nctx->nol_i_id_len[i] = 0;
  nctx->nol_supply_w_id_len[i] = 0;
  nctx->nol_quantity_len[i] = 0;
  nctx->nol_amount_len[i] = 0;
  nctx->ol_w_id_len[i] = 0;
  nctx->ol_d_id_len[i] = 0;
  nctx->ol_o_id_len[i] = 0;
  nctx->ol_number_len[i] = 0;
  nctx->ol_dist_info_len[i] = 0;
  nctx->null_date_len[i]= 0;
  nctx->s_remote_len[i] = 0;
  nctx->i_data_len[i] = 0;
  nctx->s_data_len[i] = 0;
  nctx->s_quant_len[i] = 0;
  nctx->s_rowid_len[i] = 0;
  nctx->cons_len[i] = 0;
  nctx->i_name_len[i]=0;
  nctx->s_bg_len[i] = 0;
 }

  execstatus = OCIStmtExecute(p->tpcsvc,nctx->curn1,p->errhp,1,0,0,0,
                                                    OCI_DEFAULT |
OCI_COMMIT_ON_SUCCESS);

  if(execstatus != OCI_SUCCESS) {
   OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
   errcode = OCIERROR(p,execstatus);
   if(errcode == NOT_SERIALIZABLE)
   {
            retries++;
            return (RECOVERR);
   }
   else if (errcode == RECOVERR)
   {
            retries++;
            return (RECOVERR);
   }
   else if (errcode == SNAPSHOT_TOO_OLD)
   {
            retries++;
            return (RECOVERR);
   }
   else
   {
            return (ERR_DB_ERROR);
   }
  }


  /* did the txn succeed? */
  if (rcount != pNew->o_ol_cnt)
  {
   statusCnt = rcount - pNew->o_ol_cnt;
```

```
    pNew->o_ol_cnt = rcount;
    return (ERR_DB_NOT_COMMITED);
   }



#ifdef ISO1
   else {
     OCITransCommit(p->tpcsvc, p->errhp, OCI_DEFAULT);
   }
#endif

#if defined(ISO1) || defined(ISO7)
   sysdate (sdate);
   printf ("New Order completed at: %s\n", sdate);
#endif


/* calculate total amount */
  pNew->total_amount = 0.0;
  for (i=0;i<pNew->o_ol_cnt;i++)
  {
   if (nctx->nol_amount_ind[i] != NA)
   {
     pNew->total_amount += ntemp->nol_amount[i];
   }
  }
  pNew->total_amount *= ((float)(1-ntemp->c_discount)) * (float)(1.0 +
((float)(ntemp->d_tax))+((float)(ntemp->w_tax)));
  pNew->total_amount = pNew->total_amount/100;



  return (ERR_DB_SUCCESS);

}



void tkvcndone (newctx *pnctx)

{
   int i;
   newctx nctx = *pnctx;

   if(NULL != nctx.curn1)
     OCIHandleFree((dvoid *)nctx.curn1,OCI_HTYPE_STMT);
   if(NULL != nctx.curn2)
     OCIHandleFree((dvoid *)nctx.curn2,OCI_HTYPE_STMT);
   for (i = 0; i < 10; i++)
     if(NULL != ((nctx.curn3)[i]))
       OCIHandleFree((dvoid *)(nctx.curn3)[i],OCI_HTYPE_STMT);
   if(NULL != nctx.curn4)
     OCIHandleFree((dvoid *)nctx.curn4,OCI_HTYPE_STMT);

}
```

tpcc.c

```
/*+          FILE:       TPCC.C
 *                       Microsoft TPC-C Kit Ver. 3.00.000
 *                       Audited 08/23/96          By Francois Raab
 *
 *                       Copyright Microsoft, 1996
 *                       Copyright Digital Equipment Corp., 1997
 *
 *           PURPOSE:  Main module for TPCC.DLL which is an ISAPI service dll.
 *           Author:                Philip Durr
 *                                          philipdu@Microsoft.com
 *
 *           MODIFICATIONS:
 *
 *                       Routines substantially modified by:
 *                                  Anne Bradley              Digital Equipment
Corp.
 *                                  Bill Carr      Digital Equipment Corp.
 *
 */
/*+*****************************************************************
*******
 *                                                                  *
 *  COPYRIGHT (c) 1997, 2000 BY                                     *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
 *
 *  ALL RIGHTS RESERVED.                                            *
 *                                                                  *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS HEREBY   *
 *  TRANSFERRED.                                                    *
 *                                                                  *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                    *
 *                                                                  *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
 *
 *                                                                  *
 *                                                                  *

*******************************************************************
******-*/


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

#define TPCC_C

#include <tpccerr.h>
#include <tpccstruct.h>
#include <tpccapi.h>
#include <httpext.h>

#include <tpcc.h>
#include <web_ui.h>

/* FUNCTION: void FormatString(char *szDest, char *szPic, char *szSrc)
 *
 * PURPOSE:          This function formats a character string for inclusion in the
 *                   HTML formatted page being constructed.
 *
 * ARGUMENTS:        char        *szDest
                Destination buffer where
 *                                                                  formatted
string is to be
 *                                                                  placed
 *                   char        *szPic                             picture
string which describes
 *                                                                  how
character value is to be
 *
                formatted.
 *                   char        *szSrc                             character
string value.
 *
 * RETURNS:          None
 *
```

```
 * COMMENTS:          This functions is used to format TPC-C phone and zip value
 *                    strings.
 *
 */

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

  return;
}

/* FUNCTION: int ParseNewOrderQuery( char *pProcessedQuery[],
 *                                                  NewOrderData
*pNewOrderData )
 *
 * PURPOSE:           This function extracts and validates the new order query
 *                    from an http command string.
 *
 * ARGUMENTS:         char          *pProcessedQuery[]     array of char* that
points to
 *                                                         the value
of each name-value
 *                                                         pair.
 *                    NewOrderData  *pNewOrderData         pointer to new order
data
 *                                                         structure
 *
 * RETURNS:           int           ERR_SUCCESS            input data
successfully parsed
 *                                  error_code             reason for failure
 *
 * COMMENTS:          None
 *
 */

int ParseNewOrderQuery(char *pQueryString, NewOrderData *pNewOrderData)
{
  char    *ptr;
  int                i;
  short       items;
  char        *pProcessedQuery[MAXNEWORDERVALS];


  PARSE_QUERY_STRING(pQueryString, MAXNEWORDERVALS,
                     newOrderStrs, pProcessedQuery);

  if ( !GetValuePtr(pProcessedQuery, DID, &ptr) )
    return ERR_NEWORDER_FORM_MISSING_DID;

  GetNumeric(ptr, &pNewOrderData->d_id);
  if(0 == pNewOrderData->d_id)
    return ERR_NEWORDER_DISTRICT_INVALID;


  if ( !GetValuePtr(pProcessedQuery, CID, &ptr) )
    return ERR_NEWORDER_CUSTOMER_KEY;

  if( !GetNumeric(ptr, &pNewOrderData->c_id))
    return ERR_NEWORDER_CUSTOMER_INVALID;

  pNewOrderData->o_all_local = 1;


  for(i=0, items=0; i<MAX_OL; i++)
  {
    if( !GetValuePtr(pProcessedQuery, i*3+IID00, &ptr))
      return ERR_NEWORDER_MISSING_IID_KEY;
    if(*ptr != '&' && *ptr)
    {
      if(!GetNumeric(ptr, &pNewOrderData->o_ol[items].ol_i_id))
                return ERR_NEWORDER_ITEMID_INVALID;

      if(!GetValuePtr(pProcessedQuery, i*3+SP00, &ptr))
                return ERR_NEWORDER_MISSING_SUPPW_KEY;
      if(!GetNumeric(ptr, &pNewOrderData->o_ol[items].ol_supply_w_id))
                return ERR_NEWORDER_SUPPW_INVALID;
      if ( pNewOrderData->o_all_local &&
                  pNewOrderData->o_ol[items].ol_supply_w_id !=
                  pNewOrderData->w_id )
                pNewOrderData->o_all_local = 0;
      if(!GetValuePtr(pProcessedQuery, i*3+QTY00, &ptr))
                return ERR_NEWORDER_MISSING_QTY_KEY;
      if(!GetNumeric(ptr, &pNewOrderData->o_ol[items].ol_quantity))
                return ERR_NEWORDER_QTY_INVALID;
      if ( pNewOrderData->o_ol[items].ol_i_id >= 1000000 ||
                  pNewOrderData->o_ol[items].ol_i_id < 1 )
                return ERR_NEWORDER_ITEMID_RANGE;
      if ( pNewOrderData->o_ol[items].ol_quantity >= 100 ||
                  pNewOrderData->o_ol[items].ol_quantity < 1 )
                return ERR_NEWORDER_QTY_RANGE;
      items++;
    }
    else
    {
      if(!GetValuePtr(pProcessedQuery, i*3+SP00, &ptr))
                return ERR_NEWORDER_MISSING_SUPPW_KEY;
      if(*ptr != '&' && *ptr)
                return ERR_NEWORDER_SUPPW_WITHOUT_ITEMID;

      if(!GetValuePtr(pProcessedQuery, i*3+QTY00, &ptr))
                return ERR_NEWORDER_MISSING_QTY_KEY;
      if(*ptr != '&' && *ptr)
                return ERR_NEWORDER_QTY_WITHOUT_ITEMID;
    }
  }
  if ( items == 0 )
    return ERR_NEWORDER_NOITEMS_ENTERED;

  pNewOrderData->o_ol_cnt = items;

  return ERR_SUCCESS;
}

/* FUNCTION: int ParseOrderStatusQuery( char *pProcessedQuery[],
 *                                                  OrderStatusData
*pOrderStatusData )
 *
 * PURPOSE:           This function extracts and validates the order status query
 *                    from an http command string.
 *
 * ARGUMENTS:         char          *pProcessedQuery[]     array of char* that
points to
 *                                                         the value
of each name-value
 *                                                         pair.
 *                    OrderStatusData *pOrderStatusData pointer to new order
data
 *                                                         structure
 *
 * RETURNS:           int           ERR_SUCCESS            input data
successfully parsed
 *                                  error_code             reason for failure
 *
 * COMMENTS:          None
 *
 */
int ParseOrderStatusQuery(char *pQueryString,
                          OrderStatusData *pOrderStatusData)
{
  char        szTmp[26];
  char    *ptr;
  char        *pSzTmp;
  char        *pProcessedQuery[MAXORDERSTATUSVALS];


  PARSE_QUERY_STRING(pQueryString, MAXORDERSTATUSVALS,
```

```
                                orderStatusStrs, pProcessedQuery);

  if ( !GetValuePtr(pProcessedQuery, DID, &ptr) )
    return ERR_ORDERSTATUS_MISSING_DID_KEY;
  if ( !GetNumeric(ptr, &pOrderStatusData->d_id) )
    return ERR_ORDERSTATUS_DID_INVALID;

  if ( !GetValuePtr(pProcessedQuery, CID, &ptr) )
    return ERR_ORDERSTATUS_MISSING_CID_KEY;

  if ( *ptr == '&' || !(*ptr))
  {
    pSzTmp = szTmp;
    pOrderStatusData->c_id = INVALID_C_ID;
    if ( !GetValuePtr(pProcessedQuery, CLT_O, &ptr) )
      return ERR_ORDERSTATUS_MISSING_CLT_KEY;
    while(*ptr != '&' && *ptr)
    {
      *pSzTmp = *ptr;
      pSzTmp++;
      ptr++;
    }
    *pSzTmp = '\0';
    _strupr( szTmp );
    strcpy(pOrderStatusData->c_last, szTmp);
    if ( strlen(pOrderStatusData->c_last) > 16 )
      return ERR_ORDERSTATUS_CLT_RANGE;
  }
  else
  {
    if (!GetNumeric(ptr, &pOrderStatusData->c_id))
      return ERR_ORDERSTATUS_CID_INVALID;
    if ( !GetValuePtr(pProcessedQuery, CLT_O, &ptr) )
      return ERR_ORDERSTATUS_MISSING_CLT_KEY;
    if ( *ptr != '&' && *ptr)
      return ERR_ORDERSTATUS_CID_AND_CLT;
  }

  return ERR_SUCCESS;
}

/* FUNCTION: int ParsePaymentQuery( char *pProcessedQuery[],
 *                                                  PaymentData *pPaymentData )
 *
 * PURPOSE:             This function extracts and validates the payment query
 *                      from an http command string.
 *
 * ARGUMENTS:           char            *pProcessedQuery[]      array of char* that
points to
 *                                                                      the value
of each name-value
 *                                                                      pair.
 *                      PaymentData  *pPaymentData          pointer to payment
data
 *                                                                      structure
 *
 * RETURNS:             int             ERR_SUCCESS                     input data
successfully parsed
 *                                      error_code              reason for failure
 *
 * COMMENTS:            None
 *
 */

int ParsePaymentQuery(char *pQueryString, PaymentData *pPaymentData)
{
  char          szTmp[26];
  char          *ptr;
  char    *pPtr;
  char          *pSzTmp;
  char          *pProcessedQuery[MAXPAYMENTVALS];


  PARSE_QUERY_STRING(pQueryString, MAXPAYMENTVALS,
                                paymentStrs, pProcessedQuery);

  if ( !GetValuePtr(pProcessedQuery, DID, &ptr) )
    return ERR_PAYMENT_MISSING_DID_KEY;
  if ( !GetNumeric(ptr, &pPaymentData->d_id) )
    return ERR_PAYMENT_DISTRICT_INVALID;
  if ( !GetValuePtr(pProcessedQuery, CID, &ptr) )
    return ERR_PAYMENT_MISSING_CID_KEY;


  if(*ptr == '&' || !(*ptr))
  {
    pPaymentData->c_id = INVALID_C_ID;
    pSzTmp = szTmp;
    if ( !GetValuePtr(pProcessedQuery, CLT_P, &ptr) )
      return ERR_PAYMENT_MISSING_CLT;
    if (*ptr == '&' || !(*ptr))
      return ERR_PAYMENT_MISSING_CID_CLT;
    while(*ptr != '&' && *ptr)
    {
      *pSzTmp = *ptr;
      pSzTmp++;
      ptr++;
    }
    *pSzTmp = '\0';
    _strupr( szTmp );

    strcpy(pPaymentData->c_last, szTmp);
    if ( strlen(pPaymentData->c_last) > 16 )
      return ERR_PAYMENT_LAST_NAME_TO_LONG;
  }
  else
  {
    if (!GetNumeric(ptr, &pPaymentData->c_id))
      return ERR_PAYMENT_CUSTOMER_INVALID;
    if ( !GetValuePtr(pProcessedQuery, CLT_P, &ptr) )
      return ERR_PAYMENT_MISSING_CLT_KEY;
    if(*ptr != '&' && *ptr)
      return ERR_PAYMENT_CID_AND_CLT;
  }

  if ( !GetValuePtr(pProcessedQuery, CDI, &ptr) )
    return ERR_PAYMENT_MISSING_CDI_KEY;
  if ( !GetNumeric(ptr, &pPaymentData->c_d_id) )
    return ERR_PAYMENT_CDI_INVALID;

  if ( !GetValuePtr(pProcessedQuery, CWI, &ptr) )
    return ERR_PAYMENT_MISSING_CWI_KEY;

  if ( !GetNumeric(ptr, &pPaymentData->c_w_id) )
    return ERR_PAYMENT_CWI_INVALID;



  if ( !GetValuePtr(pProcessedQuery, HAM, &ptr) )
    return ERR_PAYMENT_MISSING_HAM_KEY;

  pPtr = ptr;
  while( *pPtr != '&' && *pPtr)
  {
    if ( *pPtr == '.' )
    {
      pPtr++;
      if ( !*pPtr )
              break;
      if ( *pPtr < '0' || *pPtr > '9' )
              return ERR_PAYMENT_HAM_INVALID;
      pPtr++;
      if ( !*pPtr )
              break;
      if ( *pPtr < '0' || *pPtr > '9' )
              return ERR_PAYMENT_HAM_INVALID;
      if ( !*pPtr )
              return ERR_PAYMENT_HAM_INVALID;
    }
    else if ( *pPtr < '0' || *pPtr > '9' )
      return ERR_PAYMENT_HAM_INVALID;
    pPtr++;
  }

  pPaymentData->h_amount = atof(ptr);
  if ( pPaymentData->h_amount >= 10000.00 || pPaymentData->h_amount < 0 )
    return ERR_PAYMENT_HAM_RANGE;

  return ERR_SUCCESS;
```

```
}
```

tpcc.h

```
#ifndef TPCC_H
#define TPCC_H

/*+*****************************************************************************
 *                                                                             *
 * COPYRIGHT (c) 1997, 2000 BY                                                 *
 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 * ALL RIGHTS RESERVED.                                                        *
 *                                                                             *
 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 * OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 * TRANSFERRED.                                                                *
 *                                                                             *
 * THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 * AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 * CORPORATION.                                                                *
 *                                                                             *
 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                             *
 *                                                                             *

*******************************************************************************
*****-*/

/*+
 * Abstract: This is the header file for web_ui.c.  it contains the
 *              function prototypes for the routines that are called outside web_ui.c
 *
 * Author: A Bradley
 * Creation Date: May 1997
 *
 * Modified history:
 *
 *
 */


#if defined WEB_UI_C || defined TPCC_C

void FormatString(char *szDest, char *szPic, char *szSrc);

int ParseNewOrderQuery(char *pQueryString, NewOrderData *pNewOrderData);
int ParsePaymentQuery(char *pQueryString, PaymentData *pPaymentData);
int ParseOrderStatusQuery(char *pQueryString,
                                        OrderStatusData *pOrderStatusData);
#endif /* defined WEB_UI_C || defined TPCC_C */

#endif /* TPCC_H */
```

tpcc_acmsxp.h

```
#ifndef TPCC_ACMSXP_H
#define TPCC_ACMSXP_H

/*+*****************************************************************************
 *                                                                             *
 * COPYRIGHT (c) 1997, 2000 BY                                                 *
 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 * ALL RIGHTS RESERVED.                                                        *
 *                                                                             *
 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 * OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 * TRANSFERRED.                                                                *
 *                                                                             *
 * THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 * AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 * CORPORATION.                                                                *
 *                                                                             *
 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                             *
 *                                                                             *

*******************************************************************************
*****-*/

int ACMSxpStartup( pConfigData pConfig, long totalGovValue );
int ACMSxpShutdown( void );

#endif /* TPCC_ACMSXP_H */
```

tpcc_acmsxp_pp.stdl

```
#Cinclude "tm_xp_workspaces"
!
! STDL DY Processing group definition for TPC-C
!
PROCESSING GROUP tpcc_dy
  UUID "11111111-8322-11cf-1111-08002be44c83";
  VERSION 1.0;
  SOURCE LANGUAGE IS C;

PROCEDURE dy_transaction_init
  ARGUMENT IS
                io_login_wksp PASSED AS INOUT;

PROCEDURE dy_transaction
  ARGUMENT IS
                io_dy_wksp PASSED AS INOUT;

END PROCESSING GROUP;

!
! STDL NO Processing group definition for TPC-C
!
PROCESSING GROUP tpcc_no
  UUID "2402d1dc-8322-11cf-ba68-08002be44c83";
  VERSION 1.0;
  SOURCE LANGUAGE IS C;

PROCEDURE no_transaction_init
  ARGUMENT IS
                io_login_wksp PASSED AS INOUT;

PROCEDURE no_transaction
  ARGUMENT IS
                io_no_wksp PASSED AS INOUT;
```

```
END PROCESSING GROUP;

!
! STDL OS Processing group definition for TPC-C
!
PROCESSING GROUP tpcc_os
  UUID "4d654a5f-8322-11cf-b420-08002be44c83";
  VERSION 1.0;
  SOURCE LANGUAGE IS C;

PROCEDURE os_transaction_init
  ARGUMENT IS
                io_login_wksp PASSED AS INOUT;

PROCEDURE os_transaction
  ARGUMENT IS
                io_os_wksp PASSED AS INOUT;

END PROCESSING GROUP;

!
! STDL PT Processing group definition for TPC-C
!
PROCESSING GROUP tpcc_pt
  UUID "70e38ff0-8322-11cf-99f7-08002be44c83";
  VERSION 1.0;
  SOURCE LANGUAGE IS C;

PROCEDURE pt_transaction_init
  ARGUMENT IS
                io_login_wksp PASSED AS INOUT;

PROCEDURE pt_transaction
  ARGUMENT IS
                io_pt_wksp PASSED AS INOUT;

END PROCESSING GROUP;

!
! STDL SL Processing group definition for TPC-C
!
PROCESSING GROUP tpcc_sl
  UUID "c02bdfc2-8322-11cf-b76e-08002be44c83";
  VERSION 1.0;
  SOURCE LANGUAGE IS C;

PROCEDURE sl_transaction_init
  ARGUMENT IS
                io_login_wksp PASSED AS INOUT;

PROCEDURE sl_transaction
  ARGUMENT IS
                io_sl_wksp PASSED AS INOUT;

END PROCESSING GROUP;

!
! STDL GC Processing group definition for TPC-C
!
PROCESSING GROUP tpcc_gc
  UUID "00000000-8322-11cf-0000-08002be44c83";
  VERSION 1.0;
  SOURCE LANGUAGE IS C;

PROCEDURE gc_transaction_init
  ARGUMENT IS
                io_login_wksp PASSED AS INOUT;

PROCEDURE gc_transaction
  ARGUMENT IS
                io_gc_wksp PASSED AS INOUT,
                int_gc_wksp PASSED AS INPUT;

END PROCESSING GROUP;
```

tpcc_fct.c

```
/*+*****************************************************************
*******
 *                                                                  *
 *  COPYRIGHT (c) 1997, 2000 BY                                     *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                            *
 *                                                                  *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS HEREBY   *
 *  TRANSFERRED.                                                    *
 *                                                                  *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND SHOULD NOT BE CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                    *
 *                                                                  *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                  *
 *                                                                  *

*****************************************************************
*******/

#ifdef _WIN32
# include <windows.h>
# include <process.h>
#else
#endif

#include <string.h>
#include <stdio.h>
#include <stdldefs.h>
#include <malloc.h>

#include <tpccstruct.h>
#include <tpccapi.h>
#include <tpccerr.h>

#include <transpool.h>

#include <tpcc_acmsxp.h>
#include <tpcc_acmsxp_pp.h>

#include <deli_cli.h>
#include <deli_srv.h>

#ifdef FFE_DEBUG
# include <crtdbg.h>
#endif

#define TOTAL_ADMIN_CONNECTIONS 1

#ifdef FORCE_CONNECT
static HANDLE gForceAllThreadsEvent;
static int    gForceAllThreadsCtr;
#endif /* FORCE_CONNECT */

static int    gInitRetStatus;
static BOOL   gbGeneric = FALSE;
static BOOL   gbSynchronous = FALSE;
```

```
typedef void ( _STDL_CALL_ * stdl_callback_t )( void * );
#define STDL_SYNCHRONOUS ((stdl_callback_t)-1)

void _STDL_CALL_ dy_callback( void *);
void _STDL_CALL_ no_callback( void *);
void _STDL_CALL_ os_callback( void *);
void _STDL_CALL_ pt_callback( void *);
void _STDL_CALL_ sl_callback( void *);

static pTPCCNewOrderResponseFuncpTPCCNewOrderResponse;
static pTPCCOrderStatusResponseFunc          pTPCCOrderStatusResponse;
static pTPCCPaymentResponseFunc              pTPCCPaymentResponse;
static pTPCCStockLevelResponseFunc           pTPCCStockLevelResponse;
static pTPCCResponseCompleteFunc pTPCCResponseComplete;

struct int_gc_wksp dy = { DELIVERY_TRANS };
struct int_gc_wksp no = { NEW_ORDER_TRANS };
struct int_gc_wksp os = { ORDER_STATUS_TRANS };
struct int_gc_wksp pt = { PAYMENT_TRANS };
struct int_gc_wksp sl = { STOCK_LEVEL_TRANS };
struct int_gc_wksp gc = { GENERIC_TRANS };

#ifdef FORCE_CONNECT
void __cdecl force_connect( void *arglist );
#endif /* FORCE_CONNECT */

BOOL setGovernorValue( char *group, long ctr );
#ifdef USE_MUX
stdl_call_dll_init( DWORD why_called );
#endif

#ifdef FFE_DEBUG
pDeliveryData gpDelivery = NULL;
pNewOrderData gpNewOrder = NULL;
pOrderStatusData gpOrderStatus = NULL;
pPaymentData gpPayment = NULL;
pStockLevelData gpStockLevel = NULL;
#endif

void _STDL_CALL_
dy_callback( void *data )
{
 pDeliveryData pDelivery = data;

#ifdef FFE_DEBUG
 _ASSERT( VALID_DB_ERR(pDelivery->status) );
#endif

 ( void )TPCCDeliveryDeferredResponse( pDelivery->status, pDelivery );

 /* batch transactions do not return to the UI, so no response complete */
 /* call is necessary.
                                */
}
void _STDL_CALL_
no_callback( void *data )
{
 pNewOrderData pNewOrder = data;
 CallersContext CC;

#ifdef FFE_DEBUG
 _ASSERT( VALID_DB_ERR(pNewOrder->status) );
#endif

 CC = pNewOrder->CC;

 ( void )( *pTPCCNewOrderResponse )( pNewOrder->status, pNewOrder );

 ( *pTPCCResponseComplete )( CC );
}
void _STDL_CALL_
os_callback( void *data )
{
 pOrderStatusData pOrderStatus = data;
 CallersContext CC;

#ifdef FFE_DEBUG
 _ASSERT( VALID_DB_ERR(pOrderStatus->status) );
#endif

 CC = pOrderStatus->CC;

 ( void )( *pTPCCOrderStatusResponse )( pOrderStatus->status, pOrderStatus );

 ( *pTPCCResponseComplete )( CC );
}
void _STDL_CALL_
pt_callback( void *data )
{
 pPaymentData pPayment = data;
 CallersContext CC;

#ifdef FFE_DEBUG
 _ASSERT( VALID_DB_ERR(pPayment->status) );
#endif

 CC = pPayment->CC;

 ( void )( *pTPCCPaymentResponse )( pPayment->status, pPayment );

 ( *pTPCCResponseComplete )( CC );
}
void _STDL_CALL_
sl_callback( void *data )
{
 pStockLevelData pStockLevel = data;
 CallersContext CC;

#ifdef FFE_DEBUG
 _ASSERT( VALID_DB_ERR(pStockLevel->status) );
#endif

 CC = pStockLevel->CC;

 ( void )( *pTPCCStockLevelResponse )( pStockLevel->status, pStockLevel );

 ( *pTPCCResponseComplete )( CC );
}

/*****************************************************/
int
TPCCDeliveryDeferred( pDeliveryData *ppDelivery )
{
 int retcode;
 stdl_callback_t callback;

 TRANSACTION_DEBUG_STAGE( *ppDelivery, IN_LH );
 TRANSACTION_DEBUG_TM_SYNC_SET( *ppDelivery, gbSynchronous );

 callback = ( gbSynchronous ) ? STDL_SYNCHRONOUS : dy_callback;

 /** Call ACMSxp for New Order **/
 if( gbGeneric )
  gc_transaction( callback, (void *)(*ppDelivery),
                  (struct io_gc_wksp *)(*ppDelivery), &dy );
 else
  dy_transaction( callback, (void *)(*ppDelivery),
                  (struct io_dy_wksp *)(*ppDelivery) );

 retcode = ( gbSynchronous ) ? (*ppDelivery)->status : ERR_DB_PENDING;

#ifdef FFE_DEBUG
 _ASSERT( VALID_DB_ERR(retcode));
#endif
 TRANSACTION_DEBUG_STAGE( *ppDelivery, LEAVING_LH );

 return retcode;
}

/*****************************************************/
int
TPCCNewOrder( pNewOrderData *ppNewOrder )
{
 int retcode;
 stdl_callback_t callback;

 TRANSACTION_DEBUG_STAGE( *ppNewOrder, IN_LH );
 TRANSACTION_DEBUG_TM_SYNC_SET( *ppNewOrder, gbSynchronous );

 callback = ( gbSynchronous ) ? STDL_SYNCHRONOUS : no_callback;
```

```
  /** Call ACMSxp for New Order **/
  if( gbGeneric )
    gc_transaction( callback, (void *)(*ppNewOrder),
                                (struct io_gc_wksp *)(*ppNewOrder), &no );
  else
    no_transaction( callback, (void *)(*ppNewOrder),
                                (struct io_no_wksp *)(*ppNewOrder) );

  retcode = ( gbSynchronous ) ? (*ppNewOrder)->status : ERR_DB_PENDING;

#ifdef FFE_DEBUG
  _ASSERT( VALID_DB_ERR(retcode));
#endif
  TRANSACTION_DEBUG_STAGE( *ppNewOrder, LEAVING_LH );

  return retcode;
}

/***********************************************/
int
TPCCOrderStatus( pOrderStatusData *ppOrderStatus )
{
  int retcode;
  stdl_callback_t callback;

  TRANSACTION_DEBUG_STAGE( *ppOrderStatus, IN_LH );
  TRANSACTION_DEBUG_TM_SYNC_SET( *ppOrderStatus, gbSynchronous );

  callback = ( gbSynchronous ) ? STDL_SYNCHRONOUS : os_callback;

  /** Call ACMSxp for order status **/
  if( gbGeneric )
    gc_transaction( callback, (void *)(*ppOrderStatus),
                                (struct io_gc_wksp *)(*ppOrderStatus), &os );
  else
    os_transaction( callback, (void *)(*ppOrderStatus),
                                (struct io_os_wksp *)(*ppOrderStatus) );

  retcode = ( gbSynchronous ) ? (*ppOrderStatus)->status : ERR_DB_PENDING;

#ifdef FFE_DEBUG
  _ASSERT( VALID_DB_ERR(retcode));
#endif
  TRANSACTION_DEBUG_STAGE( *ppOrderStatus, LEAVING_LH );

  return retcode;
}

/***********************************************/
int
TPCCPayment( pPaymentData *ppPayment )
{
  int retcode;
  stdl_callback_t callback;

  TRANSACTION_DEBUG_STAGE( *ppPayment, IN_LH );
  TRANSACTION_DEBUG_TM_SYNC_SET( *ppPayment, gbSynchronous );

  callback = ( gbSynchronous ) ? STDL_SYNCHRONOUS : pt_callback;

  /** Call ACMSxp for payment **/
  if( gbGeneric )
    gc_transaction( callback, (void *)(*ppPayment),
                                (struct io_gc_wksp *)(*ppPayment), &pt );
  else
    pt_transaction( callback, (void *)(*ppPayment),
                                (struct io_pt_wksp *)(*ppPayment) );

  retcode = ( gbSynchronous ) ? (*ppPayment)->status : ERR_DB_PENDING;

#ifdef FFE_DEBUG
  _ASSERT( VALID_DB_ERR(retcode));
#endif
  TRANSACTION_DEBUG_STAGE( *ppPayment, LEAVING_LH );

  return retcode;
}

/***********************************************/
int
TPCCStockLevel( pStockLevelData *ppStockLevel )
{
  int retcode;
  stdl_callback_t callback;

  TRANSACTION_DEBUG_STAGE( *ppStockLevel, IN_LH );
  TRANSACTION_DEBUG_TM_SYNC_SET( *ppStockLevel, gbSynchronous );

  callback = ( gbSynchronous ) ? STDL_SYNCHRONOUS : sl_callback;

  /** Call ACMSxp for stock level **/
  if( gbGeneric )
    gc_transaction( callback, (void *)(*ppStockLevel),
                                (struct io_gc_wksp *)(*ppStockLevel), &sl );
  else
    sl_transaction( callback, (void *)(*ppStockLevel),
                                (struct io_sl_wksp *)(*ppStockLevel) );

  retcode = ( gbSynchronous ) ? (*ppStockLevel)->status : ERR_DB_PENDING;

#ifdef FFE_DEBUG
  _ASSERT( VALID_DB_ERR(retcode));
#endif
  TRANSACTION_DEBUG_STAGE( *ppStockLevel, LEAVING_LH );

  return retcode;
}

/*
**++
**  FUNCTION NAME: setGovernorValue
**--
*/
int
setGovernorValue( char *group, long ctr )
{
  char          keyname[256];
  int           status;
#ifdef _WIN32
  HKEY      hKey;
  DWORD     size;
  DWORD     dwDisposition;
  LONG      lRet;

  sprintf( keyname,
             "SOFTWARE\\DigitalEquipmentCorporation\\TPware\\Group
Settings\\%s",
                group );

  lRet = RegCreateKeyEx( HKEY_LOCAL_MACHINE, keyname, 0, NULL,
                                          REG_OPTION_NON_VOLATILE,
KEY_ALL_ACCESS,
                                          NULL, &hKey, &dwDisposition );
  if( ERROR_SUCCESS != lRet ) {
    TPCCErr( "OS Error %d, group settings not created/found for %s\r\n",
                lRet, keyname );
    return( ERR_CANT_SET_GOVERNOR_VALUE );
  }

  size = sizeof( ctr );
  lRet = RegSetValueEx( hKey, "MaxThreads", 0, REG_DWORD, (LPBYTE)&ctr,
size );
  if ( ERROR_SUCCESS == lRet ) {
    status = ERR_SUCCESS;
  }
  else {
    TPCCErr( "OS Error %d, MaxThreads value not set for key %s\r\n",
                lRet, keyname );
    status = ERR_CANT_SET_GOVERNOR_VALUE;
  }

  RegCloseKey( hKey );

#else /* _WIN32 */
  sprintf( keyname,

"SOFTWARE_DigitalEquipmentCorporation_TPware_Group_Settings_%s=%d",
                group, ctr );
  putenv( keyname );
```

```
  status = ERR_SUCCESS;
#endif /* _WIN32 */
  return status;
}

/*
**++
**  FUNCTION NAME: TPCCGetTransportMode
**--
*/
int
TPCCGetTransportMode( int *Mode )
{
  *Mode = TRANS_ASYNC;

  return( ERR_SUCCESS );
}

/*
**++
**  FUNCTION NAME: TPCCStartup
**--
*/
int
TPCCStartup( pStartupData pStartup )
{
  int   status;
  unsigned long ulhThread;
  int counts[HIGHEST_TRANS] = { 0 };
  int count;
  int index;
  int trans;
  long total;
  int maxDBConnections;
  pLoginData pLoginArray;
  ConfigData Config;

#ifdef USE_MUX
  stdl_call_dll_init( DLL_PROCESS_ATTACH );
#endif

  /* Save response function pointers */
  pTPCCNewOrderResponse = pStartup->pTPCCNewOrderResponse;
  pTPCCOrderStatusResponse = pStartup->pTPCCOrderStatusResponse;
  pTPCCPaymentResponse = pStartup->pTPCCPaymentResponse;
  pTPCCStockLevelResponse = pStartup->pTPCCStockLevelResponse;
  pTPCCResponseComplete = pStartup->pTPCCResponseComplete;

  /* Save general mode of operation stuff */
  gbGeneric = pStartup->Transport.generic;
  gbSynchronous = !pStartup->Transport.asynchronous;

  /**** Set the ACMSxp Governor values in the registry. ****/
  if( gbGeneric ) {
    status = setGovernorValue( "tpcc_gc", pStartup->Transport.num_gc );
    if( ERR_SUCCESS != status ) return( status );
  }
  else {
    status = setGovernorValue( "tpcc_dy", pStartup->Transport.num_dy );
    if( ERR_SUCCESS != status ) return( status );
    status = setGovernorValue( "tpcc_no", pStartup->Transport.num_no );
    if( ERR_SUCCESS != status ) return( status );
    status = setGovernorValue( "tpcc_os", pStartup->Transport.num_os );
    if( ERR_SUCCESS != status ) return( status );
    status = setGovernorValue( "tpcc_pt", pStartup->Transport.num_pt );
    if( ERR_SUCCESS != status ) return( status );
    status = setGovernorValue( "tpcc_sl", pStartup->Transport.num_sl );
    if( ERR_SUCCESS != status ) return( status );
  }

  /**** start up server specific stuff ****/
  total = 0;
  if( gbGeneric ) {
    total = pStartup->Transport.num_gc;
  }
  else {
    /* zero out the array */
    for( trans = 0; trans < HIGHEST_TRANS; trans++ )
    {
      counts[trans] = 0;
    }
    counts[DELIVERY_TRANS] = pStartup->Transport.num_dy;
    total += pStartup->Transport.num_dy;
    counts[NEW_ORDER_TRANS] = pStartup->Transport.num_no;
    total += pStartup->Transport.num_no;
    counts[ORDER_STATUS_TRANS] = pStartup->Transport.num_os;
    total += pStartup->Transport.num_os;
    counts[PAYMENT_TRANS] = pStartup->Transport.num_pt;
    total += pStartup->Transport.num_pt;
    counts[STOCK_LEVEL_TRANS] = pStartup->Transport.num_sl;
    total += pStartup->Transport.num_sl;
  }

  maxDBConnections = total + TOTAL_ADMIN_CONNECTIONS;
  if( !pStartup->DeliveryTransport.use_transport )
    maxDBConnections += pStartup->DeliveryTransport.num_threads;

  Config.maxConnections = maxDBConnections;
  Config.maxDistricts = pStartup->Config.maxDistricts;
  status = ACMSxpStartup( &Config, total );
  if ( status != ERR_DB_SUCCESS )
    return status;

  /* start the delivery subsystem */
  status = DELIClientStartup( &pStartup->DeliveryTransport,
                              &pStartup->Login, pStartup->loginDelay,
                              pStartup->Path );
  if( ERR_SUCCESS != status )
    return status;

  gInitRetStatus = ERR_DB_SUCCESS;      /** preset to success */

#ifdef FORCE_CONNECT
  gForceAllThreadsCtr = total;
  gForceAllThreadsEvent = CreateEvent( 0, TRUE, FALSE, 0 );
  if ( NULL == gForceAllThreadsEvent ) return
ERR_CANT_CREATE_ALL_THREADS_EVENT;

  if( !gbGeneric ) {
    index = 0;
    pLoginArray = alloca( total * sizeof( LoginData ));
    for( trans = 0; trans < HIGHEST_TRANS; trans++ ) {
      for( count = 0; count < counts[trans]; count++ ) {
                /* copy login data and overload unused status value as the */
                /* type of server to start */
                memcpy( &pLoginArray[index], &pStartup->Login, sizeof( LoginData ));
                pLoginArray[index].status = trans;
                ulhThread = _beginthread( force_connect, 0, &pLoginArray[index] );
                if( -1 == ulhThread ) {
                  return ERR_CANT_START_FRCDINIT_THREAD;
                }
                Sleep( pStartup->loginDelay );
                index++;
      }
    }
  }
  else {
    for( count = 0; count < pStartup->Transport.num_gc; count++ ) {
      /* overload unused status value as the type of server to start */
      pStartup->Login.status = GENERIC_TRANS;
      ulhThread = _beginthread( force_connect, 0, &pStartup->Login );
      if( -1 == ulhThread ) {
                return ERR_CANT_START_FRCDINIT_THREAD;
      }
      Sleep( pStartup->loginDelay );
    }
  }

  WaitForSingleObject( gForceAllThreadsEvent, INFINITE );
#endif /* FORCE_CONNECT */

  pStartup->Login.status = gInitRetStatus;
  return gInitRetStatus;
}


/*
**++
**  FUNCTION NAME: TPCCConnect
```

```
**--
*/
int
TPCCConnect( pLoginData *ppLogin )
{
  return ERR_DB_SUCCESS;
}

/*
**++
**  FUNCTION NAME: TPCCDisconnect
**--
*/
int
TPCCDisconnect( pConnData *ppConn )
{
  return ERR_DB_SUCCESS;
}


/*
**++
**  FUNCTION NAME: TPCCShutdown
**--
*/
int
TPCCShutdown( void )
{
  int                                   retcode;

  retcode = ACMSxpShutdown( );
  if( ERR_SUCCESS != retcode )
    return retcode;

  retcode = DELIClientShutdown( );
  if( ERR_SUCCESS != retcode )
    return retcode;

  return retcode;
}


#ifdef FORCE_CONNECT
void __cdecl
force_connect( void *arglist )
{
  pLoginData                    pLogin;
  LoginData             loginLocal;
  int                                   txnType;

  pLogin = arglist;
  txnType = pLogin->status;


  memcpy( &loginLocal, pLogin, sizeof( LoginData ));
  loginLocal.w_id = 0;
  loginLocal.ld_id = 0;
  loginLocal.CC = 0;
  loginLocal.databaseLogin.szApplication[0] = '\0';

  switch ( txnType ) {
  case DELIVERY_TRANS:
    dy_transaction_init( STDL_SYNCHRONOUS, &loginLocal,
                                        (struct io_login_wksp *)&loginLocal );
    break;

  case NEW_ORDER_TRANS:
    no_transaction_init( STDL_SYNCHRONOUS, &loginLocal,
                                        (struct io_login_wksp *)&loginLocal );
    break;

  case ORDER_STATUS_TRANS:
    os_transaction_init( STDL_SYNCHRONOUS, &loginLocal,
                                        (struct io_login_wksp *)&loginLocal );
    break;

  case PAYMENT_TRANS:
    pt_transaction_init( STDL_SYNCHRONOUS, &loginLocal,
                                        (struct io_login_wksp *)&loginLocal );
    break;

  case STOCK_LEVEL_TRANS:
    sl_transaction_init( STDL_SYNCHRONOUS, &loginLocal,
                                        (struct io_login_wksp *)&loginLocal );
    break;

  case GENERIC_TRANS:
    gc_transaction_init( STDL_SYNCHRONOUS, &loginLocal,
                                        (struct io_login_wksp *)&loginLocal );
    break;
  }
  if ( loginLocal.status != ERR_DB_SUCCESS) {
    /** Only store the first failure **/
    if ( ERR_DB_SUCCESS == gInitRetStatus )
      gInitRetStatus = ERR_FORCE_CONNECT_THREAD_FAILED;

    TPCCErr( "Connect Transaction returned %8d\r\n", loginLocal.status );
  }
  if ( InterlockedDecrement( &gForceAllThreadsCtr ) == 0 )
    SetEvent( gForceAllThreadsEvent );
  return;
}
#endif /* FORCE_CONNECT */
```

tpcc_ps.c

```
/*+*****************************************************************
*******
 *                                                                  *
 *  COPYRIGHT (c) 1997, 2000 BY                                     *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                            *
 *                                                                  *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                    *
 *                                                                  *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                    *
 *                                                                  *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                  *
 *                                                                  *

*****************************************************************
******-*/
#ifdef _WIN32
# include <windows.h>
# include <process.h>
# include <winsock.h>
#else /* _WIN32 */
#endif /* _WIN32 */

#include <errno.h>
#include <string.h>
#include <stdio.h>
#include <sys/types.h>

#include <tpccstruct.h>
#include <tpccapi.h>
#include <tpccerr.h>
```

```
#include <transpool.h>

#include <tm_xp_workspaces.h>
#include <tpcc_acmsxp.h>

#ifdef FFE_DEBUG
# include <crtdbg.h>
#endif

#ifdef USE_PROCESSES
static int gTLSDBContext = 0;
static DBContext  gDBContext = INVALID_DB_CONTEXT;
int               TlsAlloc( void ) { return 1; }
DBContext TlsGetValue( int tls ) { return( gDBContext ); }
int               TlsSetValue( int tls, DBContext DBC ) { gDBContext = DBC; return 1;
}
void              TlsFree( int tls ) { return; }
#define GetCurrentThreadId( ) getpid( )
#else /* USE_PROCESSES */
static DWORD  gTLSDBContext = 0;
#endif /* USE_PROCESSES */

#ifdef FORCE_CONNECT
static LONG   gForceThreadStartCtr;
static HANDLE gForceThreadStartEvent;
#endif /* FORCE_CONNECT */


/*
**++
**  FUNCTION NAME: transaction_init
**--
*/
void
transaction_init( char *type, pLoginData pLogin )
{
  char                                  tmp[16];
  long                                  dwTID;

  DBContext DBC  = (DBContext) TlsGetValue(gTLSDBContext);

  if (DBC == INVALID_DB_CONTEXT) {
    /* make up app_name */
    gethostname(tmp, sizeof(tmp));
    dwTID = GetCurrentThreadId( );
    sprintf( pLogin->databaseLogin.szApplication, "%s:%s - %d",
                    tmp, type, dwTID );

    pLogin->status = TPCCConnectDB( &DBC, pLogin );

    TPCCLog( "%s, dbprocptr = %16X\r\n",
                    pLogin->databaseLogin.szApplication, DBC );

    if ( pLogin->status == ERR_DB_SUCCESS ) {
      if (TlsSetValue(gTLSDBContext, DBC) == 0)
              pLogin->status = ERR_CANT_SET_THREAD_LOCAL_STORAGE;
    }

#ifdef FORCE_CONNECT
    if ( InterlockedDecrement( &gForceThreadStartCtr ) == 0 )
      SetEvent( gForceThreadStartEvent );

    WaitForSingleObject( gForceThreadStartEvent, INFINITE );
#endif /* FORCE_CONNECT */
  }
  else {
    dwTID = GetCurrentThreadId( );
    TPCCLog( "Already logged in: %8X\n", dwTID );
    pLogin->status = ERR_ALREADY_LOGGED_IN;
  }
}

/*
**++
**  FUNCTION NAME: dy_transaction_init
**--
*/
void
dy_transaction_init( struct io_login_wksp *login_wksp )
{
  transaction_init( "DY", (pLoginData)login_wksp );
}

/*
**++
**  FUNCTION NAME: no_transaction_init
**--
*/
void
no_transaction_init( struct io_login_wksp *login_wksp )
{
  transaction_init( "NO", (pLoginData)login_wksp );
}

/*
**++
**  FUNCTION NAME: os_transaction_init
**--
*/
void
os_transaction_init( struct io_login_wksp *login_wksp )
{
  transaction_init( "OS", (pLoginData)login_wksp );
}

/*
**++
**  FUNCTION NAME: pt_transaction_init
**--
*/
void
pt_transaction_init( struct io_login_wksp *login_wksp )
{
  transaction_init( "PT", (pLoginData)login_wksp );
}

/*
**++
**  FUNCTION NAME: sl_transaction_init
**--
*/
void
sl_transaction_init( struct io_login_wksp *login_wksp )
{
  transaction_init( "SL", (pLoginData)login_wksp );
}

/*
**++
**  FUNCTION NAME: gc_transaction_init
**--
*/
void
gc_transaction_init( struct io_login_wksp *login_wksp )
{
  transaction_init( "GC", (pLoginData)login_wksp );
}

/*
**++
**  FUNCTION NAME: dy_transaction
**--
*/
void
dy_transaction( struct io_dy_wksp *dy_wksp )
{
  DBContext DBC  = (DBContext) TlsGetValue(gTLSDBContext);
  pDeliveryData ptr;

  ptr = (pDeliveryData)dy_wksp;

  TRANSACTION_DEBUG_STAGE( ptr, IN_RH );
  TRANSACTION_DEBUG_TM_THREAD_SET( ptr );

  if (DBC == INVALID_DB_CONTEXT) {
    ptr->status = ERR_DB_NOT_LOGGED_IN;
  }
  else {
    ptr->status = TPCCDeliveryDB( DBC, ptr );
  }
```

```
#ifdef FFE_DEBUG
    _ASSERT( VALID_DB_ERR(ptr->status) );
#endif
    TRANSACTION_DEBUG_STAGE( ptr, LEAVING_RH );
}

/*
**++
**  FUNCTION NAME: no_transaction
**--
*/
void
no_transaction( struct io_no_wksp *no_wksp )
{
  DBContext DBC  = (DBContext) TlsGetValue(gTLSDBContext);
  pNewOrderData ptr;

  ptr = (pNewOrderData)no_wksp;

  TRANSACTION_DEBUG_STAGE( ptr, IN_RH );
  TRANSACTION_DEBUG_TM_THREAD_SET( ptr );

  if (DBC == INVALID_DB_CONTEXT) {
    ptr->status = ERR_DB_NOT_LOGGED_IN;
  }
  else {
    ptr->status = TPCCNewOrderDB( DBC, ptr );
  }

#ifdef FFE_DEBUG
    _ASSERT( VALID_DB_ERR(ptr->status) );
#endif
    TRANSACTION_DEBUG_STAGE( ptr, LEAVING_RH );
}

/*
**++
**  FUNCTION NAME: os_transaction
**--
*/
void
os_transaction( struct io_os_wksp *os_wksp )
{
  DBContext DBC  = (DBContext) TlsGetValue(gTLSDBContext);
  pOrderStatusData ptr;

  ptr = (pOrderStatusData)os_wksp;

  TRANSACTION_DEBUG_STAGE( ptr, IN_RH );
  TRANSACTION_DEBUG_TM_THREAD_SET( ptr );

  if (DBC == INVALID_DB_CONTEXT) {
    ptr->status = ERR_DB_NOT_LOGGED_IN;
  }
  else {
    ptr->status = TPCCOrderStatusDB( DBC, ptr );
  }

#ifdef FFE_DEBUG
    _ASSERT( VALID_DB_ERR(ptr->status) );
#endif
    TRANSACTION_DEBUG_STAGE( ptr, LEAVING_RH );
}

/*
**++
**  FUNCTION NAME: pt_transaction
**--
*/
void
pt_transaction( struct io_pt_wksp *pt_wksp )
{
  DBContext DBC  = (DBContext) TlsGetValue(gTLSDBContext);
  pPaymentData ptr;

  ptr = (pPaymentData)pt_wksp;

  TRANSACTION_DEBUG_STAGE( ptr, IN_RH );
  TRANSACTION_DEBUG_TM_THREAD_SET( ptr );

  if (DBC == INVALID_DB_CONTEXT) {
    ptr->status = ERR_DB_NOT_LOGGED_IN;
  }
  else {
    ptr->status = TPCCPaymentDB( DBC, ptr );
  }

#ifdef FFE_DEBUG
    _ASSERT( VALID_DB_ERR(ptr->status) );
#endif
    TRANSACTION_DEBUG_STAGE( ptr, LEAVING_RH );
}

/*
**++
**  FUNCTION NAME: sl_transaction
**--
*/
void
sl_transaction( struct io_sl_wksp *sl_wksp )
{
  DBContext DBC  = (DBContext) TlsGetValue(gTLSDBContext);
  pStockLevelData ptr;

  ptr = (pStockLevelData)sl_wksp;

  TRANSACTION_DEBUG_STAGE( ptr, IN_RH );
  TRANSACTION_DEBUG_TM_THREAD_SET( ptr );

  if (DBC == INVALID_DB_CONTEXT) {
    ptr->status = ERR_DB_NOT_LOGGED_IN;
  }
  else {
    ptr->status = TPCCStockLevelDB( DBC, ptr );
  }

#ifdef FFE_DEBUG
    _ASSERT( VALID_DB_ERR(ptr->status) );
#endif
    TRANSACTION_DEBUG_STAGE( ptr, LEAVING_RH );
}

/*
**++
**  FUNCTION NAME: gc_transaction
**--
*/
void
gc_transaction( struct io_gc_wksp *gc_wksp, struct int_gc_wksp *ptype )
{
  int transaction_type;

  transaction_type = ptype->trans;
  switch( transaction_type ) {
  case DELIVERY_TRANS:
    dy_transaction( (struct io_dy_wksp *)gc_wksp );
    break;
  case NEW_ORDER_TRANS:
    no_transaction( (struct io_no_wksp *)gc_wksp );
    break;
  case ORDER_STATUS_TRANS:
    os_transaction( (struct io_os_wksp *)gc_wksp );
    break;
  case PAYMENT_TRANS:
    pt_transaction( (struct io_pt_wksp *)gc_wksp );
    break;
  case STOCK_LEVEL_TRANS:
    sl_transaction( (struct io_sl_wksp *)gc_wksp );
    break;
  default:
#ifdef FFE_DEBUG
    _ASSERT( FALSE );
#else
    ;
#endif
  }
}

/*
```

```
**++
**  FUNCTION NAME: ACMSxpStartup
**--
*/
int ACMSxpStartup( pConfigData pConfig, long totalGovValue )
{
 int ret_status;

 /***** Allocate an index into Thread Local Storage ****/
 if ((gTLSDBContext = TlsAlloc()) == 0xFFFFFFFF)
  return ERR_CANT_ALLOCATE_THREAD_LOCAL_STORAGE;

 /***  Initial value of a counter is set to the maximum number of threads which
      can be created by ACMSxp; i.e. the total of the governor values in the
      registry.  By blocking each of these threads until they have all
            started, we guarantee that they will all be created.  Waiting until
            demand requires them is the worst time to be connecting to a DB.
            They are unblocked by the counter reaching zero which causes the
            event to be set.  ******/

#ifdef FORCE_CONNECT
 gForceThreadStartCtr = totalGovValue;

 gForceThreadStartEvent = CreateEvent( NULL, TRUE, FALSE, NULL );
 if ( gForceThreadStartEvent == NULL )
  return ERR_CANT_CREATE_FORCE_THRED_STRT_EVENT;
#endif /* FORCE_CONNECT */

 ret_status = TPCCStartupDB( pConfig );
 return ret_status;
}

/*
**++
**  FUNCTION NAME: ACMSxpShutdown
**--
*/
int ACMSxpShutdown( void )
{
#ifdef FORCE_CONNECT
 CloseHandle( gForceThreadStartEvent );
#endif /* FORCE_CONNECT */
 TlsFree( gTLSDBContext );
 return( TPCCShutdownDB() );
}
```

tpccapi.h

```
#ifndef TPCCAPI_H
#define TPCCAPI_H
/*+*****************************************************************************
*******
 *                                                                *
 *  COPYRIGHT (c) 1997, 2000 BY                                   *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                          *
 *                                                                *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY  OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                  *
 *                                                                *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                  *
 *                                                                *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                *
 *                                                                *

*****************************************************************************
******-*/

/*+*****************************************************************************
*******
*********************************  tpccapi.h
************************************
*****************************************************************************
*********
*
**  tpccapi.h:  This header file declares function calls between TPCC
**          application and server
*
*
*  Authors: Tareef Kawaf and Bill Carr
**
**
**  02-05-97 FWM Added bQueueDelivery flag to startup call.
**  18-Feb-98 WCarr      Introduced TPCCAPI V2.0
**  08-Apr-99 WCarr      Introduced TPCCAPI V3.0
**
*/


#define FILENAMESIZE 256

#define TRANS_ASYNC             1               /* Transport async capable */
#define TRANS_SYNC  2           /* Transport not async capable */
#define TRANS_NONE  3           /* No transport connected */

#ifdef MAX
# undef MAX
#endif /* MAX */
#define MAX(a,b) (((a)>(b))?(a):(b))

int TPCCGetTransportModeUI( int *Mode );
typedef int (*TPCCGetTransportModeUIFunc)( int *Mode );
int TPCCGetTransportMode( int *Mode );
typedef int (*TPCCGetTransportModeFunc)( int *Mode );

int TPCCStartupUI( pStartupData pStartup );
typedef int (*TPCCStartupUIFunc)( pStartupData pStartup );
int TPCCStartup( pStartupData pStartup );
typedef int (*TPCCStartupFunc)( pStartupData pStartup );
int TPCCStartupDB( pConfigData pConfig );
typedef int (*TPCCStartupDBFunc)( pConfigData pConfig );

int TPCCConnectUI( pLoginData *ppLogin );
typedef int (*TPCCConnectUIFunc)( pLoginData *ppLogin );
int TPCCConnect( pLoginData *ppLogin );
typedef int (*TPCCConnectFunc)( pLoginData *ppLogin );
int TPCCConnectDB( DBContext *pDBC, pLoginData pLogin );
typedef int (*TPCCConnectDBFunc)( DBContext *pDBC, pLoginData pLogin );

int TPCCDeliveryUI( pDeliveryData *ppDelivery,
                    pDeliveryData
CompletedDeliveries[DELIVERY_RESPONSE_COUNT]);
typedef int (*TPCCDeliveryUIFunc)( pDeliveryData *ppDelivery,
                     pDeliveryData
CompletedDeliveries[DELIVERY_RESPONSE_COUNT]);
int TPCCDelivery( pDeliveryData *ppDelivery,
                    pDeliveryData
CompletedDeliveries[DELIVERY_RESPONSE_COUNT] );
typedef int (*TPCCDeliveryFunc)( pDeliveryData *ppDelivery,
                    pDeliveryData
CompletedDeliveries[DELIVERY_RESPONSE_COUNT] );
int TPCCDeliveryDeferred( pDeliveryData *ppDelivery );
int TPCCDeliveryDB( DBContext DBC, pDeliveryData pDelivery );
typedef int (*TPCCDeliveryDBFunc)( DBContext DBC, pDeliveryData pDelivery );

int TPCCNewOrderUI( pNewOrderData *ppNewOrder );
typedef int (*TPCCNewOrderUIFunc)( pNewOrderData *ppNewOrder );
int TPCCNewOrder( pNewOrderData *ppNewOrder );
```

```
typedef int (*TPCCNewOrderFunc)( pNewOrderData *ppNewOrder );
int TPCCNewOrderDB( DBContext DBC, pNewOrderData pNewOrder );
typedef int (*TPCCNewOrderDBFunc)( DBContext DBC, pNewOrderData
pNewOrder );

int TPCCOrderStatusUI( pOrderStatusData *ppOrderStatus );
typedef int (*TPCCOrderStatusUIFunc)( pOrderStatusData *ppOrderStatus );
int TPCCOrderStatus( pOrderStatusData *ppOrderStatus );
typedef int (*TPCCOrderStatusFunc)( pOrderStatusData *ppOrderStatus );
int TPCCOrderStatusDB( DBContext DBC, pOrderStatusData pOrderStatus );
typedef int (*TPCCOrderStatusDBFunc)( DBContext DBC, pOrderStatusData
pOrderStatus );

int TPCCPaymentUI( pPaymentData *ppPayment );
typedef int (*TPCCPaymentUIFunc)( pPaymentData *ppPayment );
int TPCCPayment( pPaymentData *ppPayment );
typedef int (*TPCCPaymentFunc)( pPaymentData *ppPayment );
int TPCCPaymentDB( DBContext DBC, pPaymentData pPayment );
typedef int (*TPCCPaymentDBFunc)( DBContext DBC, pPaymentData pPayment );

int TPCCStockLevelUI( pStockLevelData *ppStockLevel );
typedef int (*TPCCStockLevelUIFunc)( pStockLevelData *ppStockLevel );
int TPCCStockLevel( pStockLevelData *ppStockLevel );
typedef int (*TPCCStockLevelFunc)( pStockLevelData *ppStockLevel );
int TPCCStockLevelDB( DBContext DBC, pStockLevelData pStockLevel );
typedef int (*TPCCStockLevelDBFunc)( DBContext DBC, pStockLevelData
pStockLevel );

int TPCCCheckpointConnectUI( pLoginData *ppLogin );
typedef int (*TPCCCheckpointConnectUIFunc)( pLoginData *ppLogin );
int TPCCCheckpointConnect( pLoginData *ppLogin );
typedef int (*TPCCCheckpointConnectFunc)( pLoginData *ppLogin );

int TPCCCheckpointUI( pCheckpointData *ppCheckpoint );
typedef int (*TPCCCheckpointUIFunc)( pCheckpointData *ppCheckpoint );
int TPCCCheckpoint( pCheckpointData *ppCheckpoint );
typedef int (*TPCCCheckpointFunc)( pCheckpointData *ppCheckpoint );
int TPCCCheckpointDB( DBContext DBC, pCheckpointData pCheckpoint );
typedef int (*TPCCCheckpointDBFunc)( DBContext DBC, pCheckpointData
pCheckpoint );

int TPCCCheckpointDisconnectUI( pConnData *ppConn );
typedef int (*TPCCCheckpointDisconnectUIFunc)( pConnData *ppConn );
int TPCCCheckpointDisconnect( pConnData *ppConn );
typedef int (*TPCCCheckpointDisconnectFunc)( pConnData *ppConn );

int TPCCDisconnectUI( pConnData *ppConn );
typedef int (*TPCCDisconnectUIFunc)( pConnData *ppConn );
int TPCCDisconnect( pConnData *ppConn );
typedef int (*TPCCDisconnectFunc)( pConnData *ppConn );
int TPCCDisconnectDB( DBContext DBC, pConnData pConn );
typedef int (*TPCCDisconnectDBFunc)( DBContext DBC, pConnData pConn );

int TPCCShutdownUI( void );
typedef int (*TPCCShutdownUIFunc)( void );
int TPCCShutdown( void );
typedef int (*TPCCShutdownFunc)( void );
int TPCCShutdownDB( void );
typedef int (*TPCCShutdownDBFunc)( void );

int TPCCDeliveryResponse( int retcode, pDeliveryData pDelivery,
                          pDeliveryData
CompletedDeliveries[DELIVERY_RESPONSE_COUNT] );

int TPCCDeliveryDeferredResponse( int retcode, pDeliveryData pDelivery );

int TPCCNewOrderResponse( int retcode, pNewOrderData pNewOrder );

int TPCCOrderStatusResponse( int retcode, pOrderStatusData pOrderStatus );

int TPCCPaymentResponse( int retcode, pPaymentData pPayment );

int TPCCStockLevelResponse( int retcode, pStockLevelData pStockLevel );

int TPCCCheckpointResponse( int retcode, pCheckpointData pCheckpoint );

void TPCCResponseComplete( CallersContext CC );

void ErrorMessage( CallersContext CC, int iError, int iErrorType,
                   char *pszMesasge );

int TPCCGetTransportErrorString( int iErrorCode, int iBufSize, char *pBuffer );
int TPCCGetDBErrorString( int iErrorCode, int iBufSize, char *pBuffer );

BOOL TPCCOpenLogUI( char *Path );
typedef BOOL (*TPCCOpenLogUIFunc)( char *Path );
BOOL TPCCOpenLog( char *Path );
typedef BOOL (*TPCCOpenLogFunc)( char *Path );

BOOL TPCCCloseLogUI( void );
typedef BOOL (*TPCCCloseLogUIFunc)( void );
BOOL TPCCCloseLog( void );
typedef BOOL (*TPCCCloseLogFunc)( void );

void TPCCLog( char *fmt, ... );
typedef void (*TPCCLogFunc)( char *fmt, ... );

void TPCCErr( char *fmt, ... );
typedef void (*TPCCErrFunc)( char *fmt, ... );

char *TPCCErrString( int iError );
typedef char *(*TPCCErrStringFunc)( int iError );

void TPCCTransactionErr( pConnData pConn, char *fmt, ... );

int TPCCTMLoadUI( void );
typedef int (*TPCCTMLoadUIFunc)( void );
int TPCCTMLoad( void );

int TPCCTMUnloadUI( void );
typedef int (*TPCCTMUnloadUIFunc)( void );
int TPCCTMUnload( void );

#endif /* TPCCAPI_H */
```

tpccerr.h

```
#ifndef TPCCERR_H
#define TPCCERR_H

/*+*********************************************************************
*******
 *                                                                    *
 * COPYRIGHT (c) 1997, 2000 BY                                        *
 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 * ALL RIGHTS RESERVED.                                               *
 *                                                                    *
 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE
IS HEREBY   *
 * TRANSFERRED.                                                       *
 *                                                                    *
 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE
WITHOUT NOTICE   *
 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
EQUIPMENT   *
 * CORPORATION.                                                       *
 *                                                                    *
 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR
RELIABILITY OF ITS   *
 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                    *
 *                                                                    *

**********************************************************************
*****-*/
/*              FILE:                       TPCCERR.H
 *
```

```
*                               Copyright Microsoft, 1996
*                               Copyright Digital Equipment Corp., 1997
*
*       PURPOSE:  Header file for ISAPI TPCC.DLL, defines structures
*                               and error messages used by tpcc benchmark
code.
*       Author:         Philip Durr
*                               philipdu@Microsoft.com
*
*       Modified by:    William D. Carr
*                               carr@perfom.enet.dec.com
*
*/

#pragma message ("FIXME: the error types need to be made DB non-specific")

#define ERR_DB_SUCCESS                                  0
#define ERR_DB_ERROR                                    1
#define ERR_TRANSPORT_ERROR                     2
#define ERR_DB_INTERFACE                        3
#define ERR_DB_DEADLOCK_LIMIT                           4
#define ERR_DB_NOT_COMMITED                             5
#define ERR_DB_DEAD                                     6
#define ERR_DB_PENDING                                  7
#define ERR_DB_NOT_LOGGED_IN                            8
#define ERR_DB_LOGIN_FAILED                     9
#define ERR_DB_USE_FAILED                       10
#define ERR_DB_LOGOUT_FAILED                            11
#define ERR_DB_INVALID_CONTEXT                          12
#define ERR_DB_INVALID_W_ID                     13
#define ERR_DB_INVALID_D_ID                     14
#define ERR_DB_INVALID_NO_ITEM_ID               15
#define ERR_DB_MAX_CONN_EXCEEDED                16
/* NOTE: Be sure to update MAX_ERR if new error code is added. */
#define ERR_DB_MAX_ERR                                  16

#define VALID_DB_ERR(err) (((err) >= ERR_DB_SUCCESS)&&((err) <=
ERR_DB_MAX_ERR))

#define     ERR_SUCCESS                                 1000
#define     ERR_COMMAND_UNDEFINED                       1001
#define     ERR_NOT_IMPLEMENTED_YET                     1002
#define     ERR_CANNOT_INIT_TERMINAL            1003
#define     ERR_OUT_OF_MEMORY                           1004
#define     ERR_NEW_ORDER_NOT_PROCESSED                 1005
#define     ERR_PAYMENT_NOT_PROCESSED                   1006
#define     ERR_NO_SERVER_SPECIFIED                     1007
#define     ERR_ORDER_STATUS_NOT_PROCESSED              1008
#define     ERR_W_ID_INVALID                    1009
#define     ERR_CAN_NOT_SET_MAX_CONNECTIONS             1010
#define     ERR_NOSUCH_CUSTOMER                         1011
#define     ERR_D_ID_INVALID                    1012
#define     ERR_MAX_CONNECT_PARAM                       1013
#define     ERR_INVALID_SYNC_CONNECTION                 1014
#define     ERR_INVALID_TERMID                          1015
#define     ERR_PAYMENT_INVALID_CUSTOMER                1016
#define     ERR_SQL_OPEN_CONNECTION                     1017
#define     ERR_STOCKLEVEL_MISSING_THRESHOLD_KEY        1018
#define     ERR_STOCKLEVEL_THRESHOLD_INVALID    1019
#define     ERR_STOCKLEVEL_THRESHOLD_RANGE              1020
#define     ERR_STOCKLEVEL_NOT_PROCESSED                1021
#define     ERR_NEWORDER_FORM_MISSING_DID               1022
#define     ERR_NEWORDER_DISTRICT_INVALID               1023
#define     ERR_NEWORDER_DISTRICT_RANGE                 1024
#define     ERR_NEWORDER_CUSTOMER_KEY                   1025
#define     ERR_NEWORDER_CUSTOMER_INVALID               1026
#define     ERR_NEWORDER_CUSTOMER_RANGE                 1027
#define     ERR_NEWORDER_MISSING_IID_KEY                1028
#define     ERR_NEWORDER_ITEM_BLANK_LINES               1029
#define     ERR_NEWORDER_ITEMID_INVALID                 1030
#define     ERR_NEWORDER_MISSING_SUPPW_KEY              1031
#define     ERR_NEWORDER_SUPPW_INVALID                  1032
#define     ERR_NEWORDER_MISSING_QTY_KEY                1033
#define     ERR_NEWORDER_QTY_INVALID            1034
#define     ERR_NEWORDER_SUPPW_RANGE                    1035
#define     ERR_NEWORDER_ITEMID_RANGE                   1036
#define     ERR_NEWORDER_QTY_RANGE                      1037
#define     ERR_PAYMENT_DISTRICT_INVALID                1038
#define     ERR_NEWORDER_SUPPW_WITHOUT_ITEMID           1039
#define     ERR_NEWORDER_QTY_WITHOUT_ITEMID             1040
#define     ERR_NEWORDER_NOITEMS_ENTERED                1041
#define     ERR_PAYMENT_MISSING_DID_KEY                 1042
#define     ERR_PAYMENT_DISTRICT_RANGE                  1043
#define     ERR_PAYMENT_MISSING_CID_KEY                 1044
#define     ERR_PAYMENT_CUSTOMER_INVALID                1045
#define     ERR_PAYMENT_MISSING_CLT                     1046
#define     ERR_PAYMENT_LAST_NAME_TO_LONG               1047
#define     ERR_PAYMENT_CUSTOMER_RANGE                  1048
#define     ERR_PAYMENT_CID_AND_CLT                     1049
#define     ERR_PAYMENT_MISSING_CDI_KEY                 1050
#define     ERR_PAYMENT_CDI_INVALID                     1051
#define     ERR_PAYMENT_CDI_RANGE                       1052
#define     ERR_PAYMENT_MISSING_CWI_KEY                 1053
#define     ERR_PAYMENT_CWI_INVALID                     1054
#define     ERR_PAYMENT_CWI_RANGE                       1055
#define     ERR_PAYMENT_MISSING_HAM_KEY                 1056
#define     ERR_PAYMENT_HAM_INVALID                     1057
#define     ERR_PAYMENT_HAM_RANGE                       1058
#define     ERR_ORDERSTATUS_MISSING_DID_KEY             1059
#define     ERR_ORDERSTATUS_DID_INVALID                 1060
#define     ERR_ORDERSTATUS_DID_RANGE                   1061
#define     ERR_ORDERSTATUS_MISSING_CID_KEY             1062
#define     ERR_ORDERSTATUS_MISSING_CLT_KEY             1063
#define     ERR_ORDERSTATUS_CLT_RANGE                   1064
#define     ERR_ORDERSTATUS_CID_INVALID                 1065
#define     ERR_ORDERSTATUS_CID_RANGE                   1066
#define     ERR_ORDERSTATUS_CID_AND_CLT                 1067
#define     ERR_DELIVERY_MISSING_OCD_KEY                1068
#define     ERR_DELIVERY_CARRIER_INVALID                1069
#define     ERR_DELIVERY_CARRIER_ID_RANGE               1070
#define ERR_PAYMENT_MISSING_CLT_KEY             1071
#define ERR_CANT_FIND_TPCC_KEY                          1072
#define ERR_CANT_FIND_INETINFO_KEY              1073
#define ERR_CANT_FIND_POOLTHREADLIMIT                   1074
#define ERR_DB_DELIVERY_NOT_QUEUED              1075
#define ERR_DELIVERY_NOT_PROCESSED              1076
#define ERR_TERM_ALLOCATE_FAILED                1077
#define ERR_PENDING                             1078
#define ERR_CANT_START_FRCDINIT_THREAD                  1079
#define ERR_CANT_START_DELIVERY_THREAD                  1080
#define ERR_GOVERNOR_VALUE_NOT_FOUND        1081
#define ERR_SERVER_MISMATCH             1082
#define ERR_DATABASE_MISMATCH           1083
#define ERR_USER_MISMATCH               1084
#define ERR_PASSWORD_MISMATCH           1085
#define ERR_CANT_CREATE_ALL_THREADS_EVENT   1086
#define ERR_CANT_CREATE_FORCE_THRED_STRT_EVENT 1087
#define ERR_CANT_ALLOCATE_THREAD_LOCAL_STORAGE 1088
#define ERR_CANT_SET_THREAD_LOCAL_STORAGE   1089
#define ERR_FORCE_CONNECT_THREAD_FAILED                 1090
#define ERR_CANT_FIND_SERVER_VALUE              1091
#define ERR_NO_MESSAGE                                  1092
#define ERR_CANT_FIND_PATH_VALUE                1093
#define     ERR_CANNOT_CREATE_RESULTS_FILE              1094
#define ERR_DELIVERY_PIPE_SECURITY              1095
#define ERR_DELIVERY_PIPE_CREATE                1096
#define ERR_DELIVERY_PIPE_OPEN                          1097
#define ERR_DELIVERY_PIPE_READ                          1098
#define ERR_DELIVERY_PIPE_DESTROY               1099
#define ERR_CANT_FIND_DATABASE_VALUE                    1100
#define ERR_CANT_FIND_USER_VALUE                1101
#define ERR_CANT_FIND_PASSWORD_VALUE                    1102
#define ERR_DELIVERY_OUTPUT_PIPE_WRITE                  1103
#define ERR_DELIVERY_OUTPUT_PIPE_READ                   1104
#define ERR_DELIVERY_MISSING_QUEUETIME_KEY      1105
#define ERR_DELIVERY_QUEUETIME_INVALID                  1106
#define ERR_ALREADY_LOGGED_IN                           1107
#define ERR_INVALID_FORM                        1109
#define ERR_DELIVERY_MUST_CONNECTDB                     1110
#define ERR_INVALID_FORM_AND_CMD_NOT_BEGIN      1111
#define ERR_MAX_CONNECTIONS_EXCEEDED                    1112
#define ERR_CANNOT_FIND_CONNECTION              1113
#define ERR_CKPT_NOT_INITIALIZED                1114
#define ERR_CANT_FIND_CLASS                     1115
#define ERR_CANT_FIND_VALUE                     1116
#define ERR_LOGFILE_SUBSYSTEM_NOT_INITED        1117
#define ERR_CANT_FIND_TOTAL_DISTRICT_COUNT_VALUE        1118
#define ERR_CANT_FIND_MAXIMUM_CONNECTIONS_VALUE 1119
#define ERR_CANT_FIND_NUMBER_OF_DELIVERY_THREADS_VALUE
                1120
```

```
#define
ERR_CANT_FIND_NUMBER_OF_QUEUED_DELIVERY_TRANSACTIONS_VA
LUE          1121
#define ERR_CANT_FIND_USE_DELIVERY_PIPE_VALUE
             1122
#define ERR_CANT_FIND_DISPLAY_DELIVERY_COMPLETIONS_VALUE
                          1123
#define
ERR_CANT_FIND_DELIVERY_SERVER_USES_TRANSACTION_SERVER_VA
LUE          1124
#define ERR_CANT_FIND_FLUSH_DELIVERY_LOG_VALUE
                          1125
#define ERR_CANT_FIND_WRITE_DELIVERY_LOG_VALUE
                          1126
#define ERR_CANT_FIND_GENERIC_TRANSACTION_SERVERS_VALUE
                          1127
#define ERR_CANT_FIND_NUMBER_OF_GC_SERVERS_VALUE
             1128
#define ERR_CANT_FIND_NUMBER_OF_DY_SERVERS_VALUE
             1129
#define ERR_CANT_FIND_NUMBER_OF_NO_SERVERS_VALUE
             1130
#define ERR_CANT_FIND_NUMBER_OF_OS_SERVERS_VALUE
             1131
#define ERR_CANT_FIND_NUMBER_OF_PT_SERVERS_VALUE
             1132
#define ERR_CANT_FIND_NUMBER_OF_SL_SERVERS_VALUE
             1133
#define ERR_CANT_SET_GOVERNOR_VALUE
                          1134
#define ERR_VALUE_TOO_LONG
             1135
#define
ERR_CANT_FIND_ASYNCHRONOUS_TRANSACTION_SERVERS_VALUE
             1136
#define ERR_CANT_FIND_TM_VALUE
                          1137
#define ERR_CANT_LOAD_TM_MODULE
                          1138
#define ERR_CANT_FIND_TM_FUNC
                          1139
#define ERR_CANT_UNLOAD_TM_MODULE
             1140
#define ERR_CANT_FIND_RUN_ID_VALUE
             1141
#define ERR_DELIVERY_PARAMS_FILE
             1142
#define ERR_DELIVERY_SHUTDOWN
                          1143
#define      ERR_PAYMENT_MISSING_CID_CLT
                          1144
#define ERR_HOME_ENV_VAL_NOT_SET
             1145
#define ERR_CANT_OPEN_CONFIG_DATABASE
                          1146
#define ERR_CANT_CREATE_CONFIG_DATABASE
                          1147
#define ERR_CANT_SET_CONFIG_VALUE
             1148
#define ERR_TERM_IO_ERROR
             1149
#define ERR_OPERATION_CANCELLED
                          1150
#define ERR_CANT_DISPLAY_FORM
                          1151

/* 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;


#ifdef TPCCERR_C
SERRORMSG errorMsgs[] =
{
  { ERR_DB_SUCCESS, "Database completed operation successfully." },
  { ERR_DB_ERROR, "Database encountered an error in last operation." },
  { ERR_TRANSPORT_ERROR, "Transaction monitor encountered an error in last
operation." },
  { ERR_DB_INTERFACE, "Database interface module encountered an error in last
operation." },
  { ERR_DB_DEADLOCK_LIMIT, "Database deadlocked trying to complete last
operation." },
  { ERR_DB_NOT_COMMITED, "Database could not commit last transaction." },
  { ERR_DB_DEAD, "Database hase failed." },
  { ERR_DB_PENDING, "Transaction is continuing asynchronously." },
  { ERR_DB_NOT_LOGGED_IN, "Operation can not be completed without logging
in." },
  { ERR_DB_LOGIN_FAILED, "An error occured while logging into the database." },
  { ERR_DB_USE_FAILED, "Connecting to specified database segment failed." },
  { ERR_DB_LOGOUT_FAILED, "An error occured while logging out of the
database." },
  { ERR_DB_INVALID_CONTEXT, "The Database Context is invalid." },
  { ERR_DB_INVALID_W_ID, "The warehouse id is not valid." },
  { ERR_DB_INVALID_D_ID, "The district id is not valid." },
  { ERR_DB_INVALID_NO_ITEM_ID, "The item id specified in the new order is
invalid." },
  { ERR_DB_MAX_CONN_EXCEEDED, "The maximum number of database
connections has been exceeded." },
  { ERR_SUCCESS, "Success, no error." },
  { ERR_NO_MESSAGE, "No message string available for the specified error code."
},
  { ERR_COMMAND_UNDEFINED, "Command undefined." },
  { ERR_NOT_IMPLEMENTED_YET, "Not Implemented Yet." },
  { ERR_CANNOT_INIT_TERMINAL, "Cannot initialize client connection." },
  { ERR_OUT_OF_MEMORY, "Insufficient memory." },
  { ERR_NEW_ORDER_NOT_PROCESSED, "Cannot process new Order form." },
  { ERR_PAYMENT_NOT_PROCESSED, "Cannot process payment form." },
  { ERR_NO_SERVER_SPECIFIED, "No Server name specified." },
  { ERR_ORDER_STATUS_NOT_PROCESSED, "Cannot process order status form."
},
  { ERR_W_ID_INVALID, "Invalid Warehouse ID." },
  { ERR_CAN_NOT_SET_MAX_CONNECTIONS, "Insufficient memory to allocate
# connections." },
  { ERR_NOSUCH_CUSTOMER, "No such customer." },
  { ERR_D_ID_INVALID, "Invalid District ID Must be 1 to 10." },
  { ERR_MAX_CONNECT_PARAM, "Max client connections exceeded, run install
to increase." },
  { ERR_INVALID_SYNC_CONNECTION, "Invalid Terminal Sync ID." },
  { ERR_INVALID_TERMID, "Invalid Terminal ID." },
  { ERR_PAYMENT_INVALID_CUSTOMER, "Payment Form, No such Customer."
},
  { ERR_SQL_OPEN_CONNECTION, "SQLOpenConnection API Failed." },
  { ERR_STOCKLEVEL_MISSING_THRESHOLD_KEY, "Stock Level missing
Threshold key \"TT*\"." },
  { ERR_STOCKLEVEL_THRESHOLD_INVALID, "Stock Level Threshold invalid
data type range = 1 - 99." },
  { ERR_STOCKLEVEL_THRESHOLD_RANGE, "Stock Level Threshold out of
range, range must be 1 - 99." },
  { ERR_STOCKLEVEL_NOT_PROCESSED, "Stock Level not processed." },
  { ERR_NEWORDER_FORM_MISSING_DID, "New Order missing District key
\"DID*\"." },
  { ERR_NEWORDER_DISTRICT_INVALID, "New Order District ID Invalid range
1 - 10." },
  { ERR_NEWORDER_DISTRICT_RANGE, "New Order District ID out of Range.
Range = 1 - 10." },
  { ERR_NEWORDER_CUSTOMER_KEY, "New Order missing Customer key
\"CID*\"." },
  { ERR_NEWORDER_CUSTOMER_INVALID, "New Order customer id invalid
data type, range = 1 to 3000." },
  { ERR_NEWORDER_CUSTOMER_RANGE, "New Order customer id out of range,
range = 1 to 3000." },
  { ERR_NEWORDER_MISSING_IID_KEY, "New Order missing Item Id key
\"IID*\"." },
  { ERR_NEWORDER_ITEM_BLANK_LINES, "New Order blank order lines all
orders must be continuous." },
  { ERR_NEWORDER_ITEMID_INVALID, "New Order Item Id is wrong data type,
must be numeric." },
  { ERR_NEWORDER_MISSING_SUPPW_KEY, "New Order missing Supp_W key
\"SP##*\"." },
  { ERR_NEWORDER_SUPPW_INVALID, "New Order Supp_W invalid data type
must be numeric." },
  { ERR_NEWORDER_MISSING_QTY_KEY, "New Order Missing Qty key
\"Qty##*\"." },
  { ERR_NEWORDER_QTY_INVALID, "New Order Qty invalid must be numeric
range 1 - 99." },
```

```
  { ERR_NEWORDER_SUPPW_RANGE, "New Order Supp_W value out of range
range = 1 - Max Warehouses." },
  { ERR_NEWORDER_ITEMID_RANGE, "New Order Item Id is out of range. Range
= 1 to 999999." },
  { ERR_NEWORDER_QTY_RANGE, "New Order Qty is out of range. Range = 1 to
99." },
  { ERR_PAYMENT_DISTRICT_INVALID, "Payment District ID is invalid must be
1 - 10." },
  { ERR_NEWORDER_SUPPW_WITHOUT_ITEMID, "New Order Supp_W field
entered without a corrisponding Item_Id." },
  { ERR_NEWORDER_QTY_WITHOUT_ITEMID, "New Order Qty entered without
a corrisponding Item_Id." },
  { ERR_NEWORDER_NOITEMS_ENTERED, "New Order Blank Items between
items, items must be continuous." },
  { ERR_PAYMENT_MISSING_DID_KEY, "Payment missing District Key
\"DID*\"." },
  { ERR_PAYMENT_DISTRICT_RANGE, "Payment District Out of range, range = 1
- 10." },
  { ERR_PAYMENT_MISSING_CID_KEY, "Payment missing Customer Key
\"CID*\"." },
  { ERR_PAYMENT_CUSTOMER_INVALID, "Payment Customer data type invalid,
must be numeric." },
  { ERR_PAYMENT_MISSING_CLT, "Payment missing Customer Last Name Key
\"CLT*\"." },
  { ERR_PAYMENT_MISSING_CID_CLT, "Payment entered without Customer ID
or last Name. " },
  { ERR_PAYMENT_LAST_NAME_TO_LONG, "Payment Customer last name
longer than 16 characters." },
  { ERR_PAYMENT_CUSTOMER_RANGE, "Payment Customer ID out of range,
must be 1 to 3000." },
  { ERR_PAYMENT_CID_AND_CLT, "Payment Customer ID and Last Name
entered must be one or other." },
  { ERR_PAYMENT_MISSING_CDI_KEY, "Payment missing Customer district key
\"CDI*\"." },
  { ERR_PAYMENT_CDI_INVALID, "Payment Customer district invalid must be
numeric." },
  { ERR_PAYMENT_CDI_RANGE, "Payment Customer district out of range must be
1 - 10." },
  { ERR_PAYMENT_MISSING_CWI_KEY, "Payment missing Customer Warehouse
key \"CWI*\"." },
  { ERR_PAYMENT_CWI_INVALID, "Payment Customer Warehouse invalid must
be numeric." },
  { ERR_PAYMENT_CWI_RANGE, "Payment Customer Warehouse out of range, 1
to Max Warehouses." },
  { ERR_PAYMENT_MISSING_HAM_KEY, "Payment missing Amount key
\"HAM*\"." },
  { ERR_PAYMENT_HAM_INVALID, "Payment Amount invalid data type must be
numeric." },
  { ERR_PAYMENT_HAM_RANGE, "Payment Amount out of range, 0 - 9999.99."
},
  { ERR_ORDERSTATUS_MISSING_DID_KEY, "Order Status missing District key
\"DID*\"." },
  { ERR_ORDERSTATUS_DID_INVALID, "Order Status District invalid, value must
be numeric 1 - 10." },
  { ERR_ORDERSTATUS_DID_RANGE, "Order Status District out of range must be
1 - 10." },
  { ERR_ORDERSTATUS_MISSING_CID_KEY, "Order Status missing Customer
key \"CID*\"." },
  { ERR_ORDERSTATUS_MISSING_CLT_KEY, "Order Status missing Customer
Last Name key \"CLT*\"." },
  { ERR_ORDERSTATUS_CLT_RANGE, "Order Status Customer last name longer
than 16 characters." },
  { ERR_ORDERSTATUS_CID_INVALID, "Order Status Customer ID invalid, range
must be numeric 1 - 3000." },
  { ERR_ORDERSTATUS_CID_RANGE, "Order Status Customer ID out of range
must be 1 - 3000." },
  { ERR_ORDERSTATUS_CID_AND_CLT, "Order Status Customer ID and
LastName entered must be only one." },
  { ERR_DELIVERY_MISSING_OCD_KEY, "Delivery missing Carrier ID key
\"OCD*\"." },
  { ERR_DELIVERY_CARRIER_INVALID, "Delivery Carrier ID invalid must be
numeric 1 - 10." },
  { ERR_DELIVERY_CARRIER_ID_RANGE, "Delivery Carrier ID out of range
must be 1 - 10." },
  { ERR_PAYMENT_MISSING_CLT_KEY, "Payment missing Customer Last Name
key \"CLT*\"." },
  { ERR_DB_ERROR, "A Database error has occurred." },
  { ERR_DELIVERY_NOT_PROCESSED, "Delivery not processed." },
  { ERR_DB_DELIVERY_NOT_QUEUED, "Delivery not queued." },
  { ERR_CANT_FIND_TPCC_KEY, "TPCC key not found in registry." },
  { ERR_CANT_FIND_INETINFO_KEY, "inetinfo key not found in registry." },
  { ERR_CANT_FIND_POOLTHREADLIMIT, "PoolThreadLimit value not set in
inetinfo\\Parameters key." },
  { ERR_TERM_ALLOCATE_FAILED, "Failed to allocate terminal data structure." },
  { ERR_DELIVERY_PIPE_SECURITY, "Failed to initialize delivery pipe security."
},
  { ERR_DELIVERY_PIPE_CREATE, "Failed to create delivery pipe." },
  { ERR_DELIVERY_PIPE_OPEN, "Failed to open delivery pipe." },
  { ERR_DELIVERY_PIPE_READ, "Failed to read delivery pipe." },
  { ERR_DELIVERY_PIPE_DESTROY, "Failed to destroy delivery pipe." },
  { ERR_PENDING, "Transaction pending."},
  { ERR_CANT_START_FRCDINIT_THREAD, "Can't start Forced Initialization
thread." },
  { ERR_CANT_START_DELIVERY_THREAD, "Can't start delivery thread." },
  { ERR_GOVERNOR_VALUE_NOT_FOUND, "Governor value not found in
Registry." },
  { ERR_SERVER_MISMATCH, "Server does not match registry value." },
  { ERR_DATABASE_MISMATCH, "Database name does not match registry value."
},
  { ERR_USER_MISMATCH, "User name does not match registry value." },
  { ERR_PASSWORD_MISMATCH, "Password does not match registry value." },
  { ERR_CANT_CREATE_ALL_THREADS_EVENT, "Can't create All Threads
Event." },
  { ERR_CANT_CREATE_FORCE_THRED_STRT_EVENT, "Can't create Force
Thread Start Event." },
  { ERR_CANT_ALLOCATE_THREAD_LOCAL_STORAGE, "Can't allocate thread
local storage" },
  { ERR_CANT_SET_THREAD_LOCAL_STORAGE, "Can't set thread local
storage." },
  { ERR_FORCE_CONNECT_THREAD_FAILED, "At least one database connect
call failed, check log files for specific error." },
  { ERR_CANT_FIND_SERVER_VALUE, "Server value not set in TPCC key." },
  { ERR_CANT_FIND_PATH_VALUE, "PATH value not set in TPCC key." },
  { ERR_CANNOT_CREATE_RESULTS_FILE, "Cannot create results file." },
  { ERR_CANT_FIND_DATABASE_VALUE, "Database value not set in TPCC key."
},
  { ERR_CANT_FIND_USER_VALUE, "User value not set in TPCC key." },
  { ERR_CANT_FIND_PASSWORD_VALUE, "Password value not set in TPCC
key." },
  { ERR_DELIVERY_OUTPUT_PIPE_WRITE, "Failed to write output delivery
pipe." },
  { ERR_DELIVERY_OUTPUT_PIPE_READ, "Failed to read output delivery pipe."
},
  { ERR_DELIVERY_MISSING_QUEUETIME_KEY, "Delivery queue time missing
from query." },
  { ERR_DELIVERY_QUEUETIME_INVALID, "Delivery queue time is invalid." },
  { ERR_ALREADY_LOGGED_IN, "TPCCConnectDB has already been called." },
  { ERR_DB_NOT_LOGGED_IN, "TPCCConnectDB has not yet been called." },
  { ERR_INVALID_FORM, "The FORM field is missing or invalid." },
  { ERR_DELIVERY_MUST_CONNECTDB, "Specified TM does not allow Delivery
Servers to use Transaction Server queues." },
  { ERR_INVALID_FORM_AND_CMD_NOT_BEGIN, "The FORM field is missing
and CMD is not Begin." },
  { ERR_MAX_CONNECTIONS_EXCEEDED, "The maximum number of
connections has been exceeded." },
  { ERR_CANNOT_FIND_CONNECTION, "Transport layer unable to find a
DBContext coresponding to the CallersContext." },
  { ERR_CKPT_NOT_INITIALIZED, "The checkpoint subsystem has not been
started." },
  { ERR_CANT_FIND_CLASS, "Can't find the specified config variable class." },
  { ERR_CANT_FIND_VALUE, "Can't find the specified config variable." },
  { ERR_LOGFILE_SUBSYSTEM_NOT_INITED, "The Logfile Subsystem has not
been initialized." },
  { ERR_CANT_FIND_TOTAL_DISTRICT_COUNT_VALUE,
"TOTAL_DISTRICT_COUNT value not set in TPCC key." },
  { ERR_CANT_FIND_MAXIMUM_CONNECTIONS_VALUE, "MaxConnections
value not set in TPCC key." },
  { ERR_CANT_FIND_NUMBER_OF_DELIVERY_THREADS_VALUE,
"NumberOfDeliveryThreads value not set in TPCC key." },
  {
ERR_CANT_FIND_NUMBER_OF_QUEUED_DELIVERY_TRANSACTIONS_VA
LUE, "NumberOfQueuedDeliveryTransactions value not set in TPCC key." },
  { ERR_CANT_FIND_USE_DELIVERY_PIPE_VALUE, "UseDeliveryPipe value
not set in TPCC key." },
  { ERR_CANT_FIND_DISPLAY_DELIVERY_COMPLETIONS_VALUE,
"DisplayDeliveryCompletions value not set in TPCC key." },
  {
ERR_CANT_FIND_DELIVERY_SERVER_USES_TRANSACTION_SERVER_VA
LUE, "DeliveryServerUsesTransactionServer value not set in TPCC key." },
```

```
  { ERR_CANT_FIND_FLUSH_DELIVERY_LOG_VALUE, "FlushDeliveryLog
value not set in TPCC key." },
  { ERR_CANT_FIND_WRITE_DELIVERY_LOG_VALUE, "WriteDeliveryLog
value not set in TPCC key." },
  { ERR_CANT_FIND_GENERIC_TRANSACTION_SERVERS_VALUE,
"GenericTransactionServers value not set in TPCC key." },
  { ERR_CANT_FIND_NUMBER_OF_GC_SERVERS_VALUE,
"NumberOfGCServers value not set in TPCC key." },
  { ERR_CANT_FIND_NUMBER_OF_DY_SERVERS_VALUE,
"NumberOfDYServers value not set in TPCC key." },
  { ERR_CANT_FIND_NUMBER_OF_NO_SERVERS_VALUE,
"NumberOfNOServers value not set in TPCC key." },
  { ERR_CANT_FIND_NUMBER_OF_OS_SERVERS_VALUE,
"NumberOfOSServers value not set in TPCC key." },
  { ERR_CANT_FIND_NUMBER_OF_PT_SERVERS_VALUE,
"NumberOfPTServers value not set in TPCC key." },
  { ERR_CANT_FIND_NUMBER_OF_SL_SERVERS_VALUE,
"NumberOfSLServers value not set in TPCC key." },
  { ERR_CANT_SET_GOVERNOR_VALUE, "Can't set ACMSxp governor value in
Registry." },

  { ERR_VALUE_TOO_LONG, "Target string too short for value." },
  { ERR_CANT_FIND_ASYNCHRONOUS_TRANSACTION_SERVERS_VALUE,
"AsynchronousTransactionServers value not set in TPCC key." },
  { ERR_CANT_FIND_TM_VALUE, "TM module not set in TPCC key." },
  { ERR_CANT_LOAD_TM_MODULE, "Failed to load the TM module." },
  { ERR_CANT_FIND_TM_FUNC, "Failed to find TM routine in TM module." },
  { ERR_CANT_UNLOAD_TM_MODULE, "Failed to unload the TM module." },
  { ERR_CANT_FIND_RUN_ID_VALUE, "RUN_ID value not set in TPCC key." },
  { ERR_DELIVERY_PARAMS_FILE, "Error manipulating the delivery server
parameters file." },
  { ERR_DELIVERY_SHUTDOWN, "Error shutting down the delivery subsystem." },
  { ERR_HOME_ENV_VAL_NOT_SET, "FFE environment variable is not set." },
  { ERR_CANT_OPEN_CONFIG_DATABASE, "Can't open the configuration
database." },
  { ERR_CANT_CREATE_CONFIG_DATABASE, "Can't create the configuration
database." },
  { ERR_CANT_SET_CONFIG_VALUE, "Can't set the configuration database
value." },
  { ERR_TERM_IO_ERROR, "A terminal IO error has occured." },
  { ERR_OPERATION_CANCELLED, "The operation was cancelled." },
  { ERR_CANT_DISPLAY_FORM, "Can't display form." },
  { 0, "" }
};
#else /* TPCCERR_C */
extern SERRORMSG errorMsgs[];
#endif /* TPCCERR_C */

#endif /* TPCCERR_H */
```

tpccstruct.h

```
#ifndef TPCCSTRUCT_H
#define TPCCSTRUCT_H

/*+**********************************************************************
*******
 *                                                                      *
 *  COPYRIGHT (c) 1997, 2000 BY                                         *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                                *
 *                                                                      *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY  OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                        *
 *                                                                      *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                        *
 *                                                                      *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                      *
 *                                                                      *

**********************************************************************
******-*/

/*+**********************************************************************
********
****************************  tpccstruct.h
*********************************
**********************************************************************
*********/
/*
**  tpccstruct.h:  This header file declares data structures for use in
**          application and server
*/
/* Copyright 1996 Digital Equipment Corporation */
/*
**  Author: Bill Carr
**              (Majority of content from previous work by Ruth Morgenstein)
**
**
*/

#include <time.h>

#ifdef _WIN32
# ifndef BOOLEAN
#  define BOOLEAN BOOL
# endif
# define PATH_MAX MAX_PATH
#else
# include <sys/types.h>
# include <sys/time.h>
# define BOOL int
# define BOOLEAN BOOL
# define VMS     0
# define LINEMAX 256
# define Sleep(ms) \
  {struct timeval tv;\
   tv.tv_sec=(ms)/1000;tv.tv_usec=(((ms)%1000)*1000);\
   select(0,NULL,NULL,NULL,&tv);}
#endif

#include <confvar.h>

#define MIN_OL                          5
#define MAX_OL                          15
#define MAX_OL_QUANTITY                 10
#define INVALID_C_ID            0
#define DISTRICTS_PER_WAREHOUSE 10
#define MAX_I_ID_POPULATED      100000
#define MAX_I_ID                200000
#define MAX_C_ID                3000
#define MIN_SL_THRESHOLD        10
#define MAX_SL_THRESHOLD        20
#define MAX_CARRIER_ID                  10
#define MAX_PT_AMOUNT_CENTS             500000

#define ILLEGAL_ITEM 111111

#define DELIVERY_RESPONSE_COUNT 2


#ifdef TRANSACTION_DEBUG

# define CALLING_LH  0x0001
# define IN_LH                  0x0002
# define IN_RH                  0x0004
# define IN_DB                  0x0008
# define LEAVING_DB  0x0010
# define LEAVING_RH  0x0020
```

```
# define LEAVING_LH  0x0040
# define CALLING_RESP              0x0080
# define UNRESERVING               0x0100

# define TRANSACTION_DEBUG_STAGE(pTxn,Stage) \
                        (((pConnData)(pTxn))->iStage |= (Stage))

# define TRANSACTION_DEBUG_STAGE_CHECK(pTxn,Stages) \
                        (_ASSERT(((pConnData)(pTxn))->iStage == (Stages)))


# ifndef TRANSACTION_DEBUG_ALL_STAGES
#  define TRANSACTION_DEBUG_ALL_STAGES              0x01ff
# endif

# define TRANSACTION_DEBUG_INFO\
  int iStage;

#else /* TRANSACTION_DEBUG */

# define TRANSACTION_DEBUG_INFO
# define TRANSACTION_DEBUG_STAGE(pTxn,Stage)
# define TRANSACTION_DEBUG_STAGE_CHECK(pTxn,Stages)

#endif /* TRANSACTION_DEBUG */

_

#ifdef _WIN32
typedef LONGLONG CallersContext;
#else
typedef long CallersContext;
#endif
typedef void *DBContext;

#define INVALID_DB_CONTEXT NULL

/*
**  Function pointer prototypes;
**
*/

typedef struct _NewOrderData *pNewOrderData;
typedef struct _OrderStatusData *pOrderStatusData;
typedef struct _PaymentData *pPaymentData;
typedef struct _StockLevelData *pStockLevelData;

typedef int (*pTPCCNewOrderResponseFunc)( int retcode,
                                                        pNewOrderData
pNewOrder );
typedef int (*pTPCCOrderStatusResponseFunc)( int retcode,
                                                          pOrderStatusData
pOrderStatus );
typedef int (*pTPCCPaymentResponseFunc)( int retcode, pPaymentData pPayment );
typedef int (*pTPCCStockLevelResponseFunc)( int retcode,
                                                        pStockLevelData
pStockLevel );
typedef void (*pTPCCResponseCompleteFunc)( CallersContext CC );



/*
**  Data structures descriptions for IO data for each transaction type
**
*/

/* Holds constants for emalation/browser delays for each transaction */
typedef struct _emulated_display_delay_t {
  double  menu_delay;
  double  txn_delay;
} emulated_display_delay_t;

typedef struct _emulated_display_delay_data_t {
  emulated_display_delay_t dy;
  emulated_display_delay_t no;
  emulated_display_delay_t os;
  emulated_display_delay_t pt;
  emulated_display_delay_t sl;
} emulated_display_delay_data_t;

typedef struct _DBDate {
        int year;       /* 1900 - 2100 */
        int month;      /* 1 - 12 */
        int day;        /* 1 - 31 */
        int hour;       /* 0 - 23 */
        int minute;     /* 0 - 59 */
        int second;     /* 0 - 59 */
} DBDateData, *pDBDateData;

/* Data common to all transactions that represents the connection to the UI */
/* and the database are built as a macro to reduce duplication. */
#define CONN_DATA \
   int    iSize;\
   int    iType;\
   TRANSACTION_DEBUG_INFO\
   int    w_id;\
   int    ld_id;\
   CallersContext CC;\
   int    status;\
   int    dbstatus;


typedef struct _ConnData
{
   CONN_DATA
} ConnData, *pConnData;

/* DELIVERY is built as a macro so that i_delivery struct is consistant with */
/* the io_delivery struct.  Note also that the input portion of the delivery */
/* data can be simply memcpyed from the input to the input/output struct. */
#define I_DELIVERY \
   CONN_DATA\
   time_t  queue_time;\
   int    o_carrier_id; \
   unsigned delta_time;   /* in milliseconds */

typedef struct _DeliveryDataInput {
   I_DELIVERY
} DeliveryDataInput, *pDeliveryDataInput;

typedef struct _DeliveryData {
   I_DELIVERY                                   /* see comment above */
   int    o_id[10];
} DeliveryData, *pDeliveryData;

struct io_order_line {
   int    ol_i_id;
   int    ol_supply_w_id;
   int    ol_quantity;
   char   i_name[25];
   int    s_quantity;
   char   b_g[2];
   double  i_price;
   double  ol_amount;
};

typedef struct _NewOrderData {
   CONN_DATA
   int    d_id;
   int    c_id;
   int    o_ol_cnt;
   int    o_all_local;
   struct io_order_line o_ol[MAX_OL];
   DBDateData o_entry_d;
   char   c_last[17];
   char   c_credit[3];
   double  c_discount;
   double  w_tax;
   double  d_tax;
   int    o_id;
   double  tax_n_discount;
   double  total_amount;
} NewOrderData;

struct status_order_line {
   int    ol_supply_w_id;
   int    ol_i_id;
   int    ol_quantity;
   double  ol_amount;
   DBDateData   ol_delivery_d;
```

```
};

typedef struct _OrderStatusData {
   CONN_DATA
   BOOLEAN byname;
   int     d_id;
   int     c_id;
   char    c_last[17];
   char    c_first[17];
   char    c_middle[3];
   double  c_balance;
   int     o_id;
   DBDateData    o_entry_d;
   int     o_carrier_id;
   int     o_ol_cnt;
   struct status_order_line    s_ol[MAX_OL];
} OrderStatusData;

typedef struct _PaymentData {
   CONN_DATA
   BOOLEAN byname;
   int     d_id;
   int     c_id;
   char    c_last[17];
   int     c_w_id;
   int     c_d_id;
   double  h_amount;
   DBDateData h_date;
   char    w_street_1[21];
   char    w_street_2[21];
   char    w_city[21];
   char    w_state[3];
   char    w_zip[10];
   char    d_street_1[21];
   char    d_street_2[21];
   char    d_city[21];
   char    d_state[3];
   char    d_zip[10];
   char    c_first[17];
   char    c_middle[3];
   char    c_street_1[21];
   char    c_street_2[21];
   char    c_city[21];
   char    c_state[3];
   char    c_zip[10];
   char    c_phone[17];
   DBDateData    c_since;
   char    c_credit[3];
   double  c_credit_lim;
   double  c_discount;
   double  c_balance;
   char    c_data[201];
} PaymentData;

typedef struct _StockLevelData {
   CONN_DATA
   int     threshold;
   int     low_stock;
} StockLevelData;

typedef struct _CheckpointData {
   CONN_DATA
   int     how_many;
   int     interval;
} CheckpointData, *pCheckpointData;

/*
**  Data structure for input & output data
*/

typedef struct _LoginData {
  CONN_DATA
  DatabaseLoginData       databaseLogin;
} LoginData, *pLoginData;

typedef struct _GenericData {
  union {
    ConnData conn;
    LoginData login;
    DeliveryData delivery[1+DELIVERY_RESPONSE_COUNT];
    NewOrderData newOrder;
    OrderStatusData orderStatus;
    PaymentData payment;
    StockLevelData stockLevel;
    CheckpointData checkpoint;
  } info;
} GenericData, *pGenericData;


/* Data structure for passing startup and configuration data */
typedef struct _StartupData {
  ConfigData                    Config;
  DeliveryTransportData  DeliveryTransport;
  TransportData                 Transport;
  LoginData             Login;
  int                           loginDelay;
  char                          *Path;
  pTPCCNewOrderResponseFunc     pTPCCNewOrderResponse;
  pTPCCOrderStatusResponseFunc  pTPCCOrderStatusResponse;
  pTPCCPaymentResponseFunc      pTPCCPaymentResponse;
  pTPCCStockLevelResponseFunc   pTPCCStockLevelResponse;
  pTPCCResponseCompleteFunc     pTPCCResponseComplete;
} StartupData, *pStartupData;

#endif /* TPCCSTRUCT_H */
```

web_ui.c

```
/*+*****************************************************************
*******
 *                                                                 *
 *  COPYRIGHT (c) 1997, 2000 BY                                    *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                           *
 *                                                                 *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                   *
 *                                                                 *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                   *
 *                                                                 *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                 *
 *                                                                 *

*****************************************************************
******-*/

/*+
 * Abstract: This file contains the Digital created front end functions
 *                              for the tpcc benchmark.
 *
 * Author: A Bradley & W Carr
 * Creation Date: May 1997
 *
 * Modified history:
 *
 *
*/
#include <windows.h>
#include <winsock2.h>
```

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

#define WEB_UI_C
#include <tpccerr.h>
#include <tpccstruct.h>
#include <tpccapi.h>
#include <httpext.h>

#include <tpcc.h>
#include <web_ui.h>
#include <config.h>
#include <ckpt.h>
#include <transpool.h>
#include <tm_util.h>
#include <reg.h>

#ifdef FFE_DEBUG
# include <crtdbg.h>
static int tmpDbgFlag;
static _HFILE hMemFile;
#endif


#define PUT_STRING(szString, iLen, pStart, pStruct) \
pStruct.szStr=szString; pStruct.iIndex=pStart; pStruct.iFieldSize=iLen;

#define CONVERT_SPECIAL(pout,pin,iwid)\
{\
  char *out = pout;\
  char *in = pin;\
  int wid = iwid;\
  while( wid && '\0' != *in )\
  {\
    if( '>' == *in )\
    {*out++='&'; *out++='g'; *out++='t'; *out++=';';}\
    else if( '<' == *in )\
    {*out++='&'; *out++='l'; *out++='t'; *out++=';';}\
    else if( '&' == *in )\
    {*out++='&'; *out++='a'; *out++='m'; *out++='p'; *out++=';';}\
    else if( '\"' == *in )\
    {*out++='&'; *out++='q'; *out++='u'; *out++='o'; *out++='t'; *out++=';';}\
    else\
    {*out++=*in;}\
    in++;\
    wid--;\
  }\
  while( wid-- ) *out++ = ' ';\
}

/* define indexes for the building of the forms */
/* defines for new order */
#define NO_WDID 0
#define NO_WID NO_WDID + 1
#define NO_DID NO_WID + 1
#define NO_DATE NO_DID + 1
#define NO_CID NO_DATE + 1
#define NO_LAST NO_CID + 1
#define NO_CREDIT NO_LAST + 1
#define NO_DISC NO_CREDIT + 1
#define NO_OID NO_DISC + 1
#define NO_LINES NO_OID + 1
#define NO_W_TAX NO_LINES + 1
#define NO_D_TAX NO_W_TAX + 1
#define NO_S_WID NO_D_TAX + 1
#define NO_IID NO_S_WID + 1
#define NO_INAME NO_IID + 1
#define NO_QTY NO_INAME + 1
#define NO_STOCK NO_QTY + 1
#define NO_BG NO_STOCK + 1
#define NO_PRICE NO_BG + 1
#define NO_AMT NO_PRICE + 1
#define NO_STAT NO_AMT + (14*8) + 1
#define NO_TOTAL NO_STAT + 1

/* defines for payment input form */
#define PT_WDID_INPUT 0
#define PT_WID_INPUT PT_WDID_INPUT + 1

/* defines for payment output form */
#define PT_WDID 0
#define PT_LONG_DATE PT_WDID + 1
#define PT_WID PT_LONG_DATE + 1
#define PT_DID PT_WID + 1
#define PT_W_ST_1 PT_DID + 1
#define PT_D_ST_1 PT_W_ST_1 + 1
#define PT_W_ST_2 PT_D_ST_1 + 1
#define PT_D_ST_2 PT_W_ST_2 + 1
#define PT_W_CITY PT_D_ST_2 + 1
#define PT_W_ST PT_W_CITY + 1
#define PT_W_ZIP PT_W_ST + 1
#define PT_D_CITY PT_W_ZIP + 1
#define PT_D_ST PT_D_CITY + 1
#define PT_D_ZIP PT_D_ST + 1
#define PT_CID PT_D_ZIP + 1
#define PT_C_WID PT_CID + 1
#define PT_C_DID PT_C_WID + 1
#define PT_FIRST PT_C_DID + 1
#define PT_MIDDLE PT_FIRST + 1
#define PT_LAST PT_MIDDLE + 1
#define PT_SM_DATE PT_LAST + 1
#define PT_C_STR_1 PT_SM_DATE + 1
#define PT_CREDIT PT_C_STR_1 + 1
#define PT_C_STR_2 PT_CREDIT + 1
#define PT_DISC PT_C_STR_2 + 1
#define PT_C_CITY PT_DISC + 1
#define PT_C_ST PT_C_CITY + 1
#define PT_C_ZIP PT_C_ST + 1
#define PT_C_PHONE PT_C_ZIP + 1
#define PT_AMT PT_C_PHONE + 1
#define PT_BAL PT_AMT + 1
#define PT_LIM PT_BAL + 1
#define PT_CUST_DATA PT_LIM + 1

/* defines for order status */
#define OS_WDID 0
#define OS_WID OS_WDID + 1
#define OS_DID OS_WID + 1
#define OS_CID OS_DID + 1
#define OS_FIRST OS_CID + 1
#define OS_MIDDLE OS_FIRST + 1
#define OS_LAST OS_MIDDLE + 1
#define OS_BAL OS_LAST + 1
#define OS_OID OS_BAL + 1
#define OS_DATE OS_OID + 1
#define OS_CAR_ID OS_DATE + 1
#define OS_S_WID OS_CAR_ID + 1
#define OS_IID OS_S_WID + 1
#define OS_QTY OS_IID + 1
#define OS_AMT OS_QTY + 1
#define OS_SM_DATE OS_AMT + 1
/* defines for delivery form */
#define D_WDID 0
#define D_WID D_WDID + 1
#define D_CAR D_WID + 1
#define D_QUEUE1 D_CAR + 1
#define D_DELTA1 D_QUEUE1 + 1
#define D_WID1 D_DELTA1 + 1
#define D_CAR1 D_WID1 + 1
#define D_OID10 D_CAR1 + 1
#define D_OID11 D_OID10 + 1
#define D_OID12 D_OID11 + 1
#define D_OID13 D_OID12 + 1
#define D_OID14 D_OID13 + 1
#define D_OID15 D_OID14 + 1
#define D_OID16 D_OID15 + 1
#define D_OID17 D_OID16 + 1
#define D_OID18 D_OID17 + 1
#define D_OID19 D_OID18 + 1
#define D_QUEUE2 D_OID19 + 1
#define D_DELTA2 D_QUEUE2 + 1
#define D_WID2 D_DELTA2 + 1
```

```
#define D_CAR2 D_WID2 + 1
#define D_OID20 D_CAR2 + 1
#define D_OID21 D_OID20 + 1
#define D_OID22 D_OID21 + 1
#define D_OID23 D_OID22 + 1
#define D_OID24 D_OID23 + 1
#define D_OID25 D_OID24 + 1
#define D_OID26 D_OID25 + 1
#define D_OID27 D_OID26 + 1
#define D_OID28 D_OID27 + 1
#define D_OID29 D_OID28 + 1

/* defines for stock level form */
#define SL_WDID 0
#define SL_WID SL_WDID + 1
#define SL_DID SL_WID + 1
#define SL_TH SL_DID + 1
#define SL_LOW SL_TH + 1

#define NUMBER_POOL_FORM_TYPES 5
#define DELIVERY_FORM 0
#define NEW_ORDER_FORM 1
#define ORDER_STATUS_FORM 2
#define PAYMENT_FORM 3
#define STOCK_LEVEL_FORM 4

#define NUMBER_POOL_RESPONSE_TYPES 5
#define DELIVERY_RESPONSE 0
#define NEW_ORDER_RESPONSE 1
#define ORDER_STATUS_RESPONSE 2
#define PAYMENT_RESPONSE 3
#define STOCK_LEVEL_RESPONSE 4

#ifdef FFE_DEBUG
# define FFE_ASSERT(arg) _ASSERT(arg)
#else
# define FFE_ASSERT(arg)
#endif

#define RESERVE_FORM(type,szForm)\
 {\
  EnterCriticalSection( &gpForms->critSec[type] );\
  FFE_ASSERT( gpForms->iNextFreeForm[type] <= gpForms->iMaxIndex[type] );\
  szForm = gpForms->index[gpForms->iFirstFormIndex[type] +\
                                        gpForms->iNextFreeForm[type]++];\
  LeaveCriticalSection( &gpForms->critSec[type] );\
 }
#define UNRESERVE_FORM(type,szForm)\
 {\
  EnterCriticalSection( &gpForms->critSec[type] );\
  FFE_ASSERT( gpForms->iNextFreeForm[type] > 0 );\
  gpForms->index[gpForms->iFirstFormIndex[type] +\
                          --gpForms->iNextFreeForm[type]] = szForm;\
  LeaveCriticalSection( &gpForms->critSec[type] );\
 }

#define RESERVE_RESPONSE(type,szResponse)\
 {\
  EnterCriticalSection( &gpResponses->critSec[type] );\
  FFE_ASSERT(gpResponses->iNextFreeResponse[type]<=gpResponses-
>iMaxIndex[type]);\
  szResponse = gpResponses->index[gpResponses->iFirstResponseIndex[type] +\
                                        gpResponses-
>iNextFreeResponse[type]++];\
  LeaveCriticalSection( &gpResponses->critSec[type] );\
 }
#define UNRESERVE_RESPONSE(type,szResponse)\
 {\
  EnterCriticalSection( &gpResponses->critSec[type] );\
  FFE_ASSERT(gpResponses->iNextFreeResponse[type] > 0 );\
  gpResponses->index[gpResponses->iFirstResponseIndex[type] +\
                          --gpResponses->iNextFreeResponse[type]] = szResponse;\
  LeaveCriticalSection( &gpResponses->critSec[type] );\
 }

#define RESERVE_PANIC_FORM(szForm)\
 {\
  EnterCriticalSection( &gpPanicForms->critSec );\
  FFE_ASSERT( gpPanicForms->iNextFree <= gpPanicForms->iMaxIndex );\
  szForm = gpPanicForms->index[gpPanicForms->iNextFree++];\
  LeaveCriticalSection( &gpPanicForms->critSec );\
 }

#define UNRESERVE_PANIC_FORM(szForm)\
 {\
  EnterCriticalSection( &gpPanicForms->critSec );\
  FFE_ASSERT( gpPanicForms->iNextFree > 0 );\
  gpPanicForms->index[--gpPanicForms->iNextFree] = szForm;\
  LeaveCriticalSection( &gpPanicForms->critSec );\
 }

#if 0
CMD                 0
FORM ID             3
LOGIN WAREHOUSE     4
LOGIN DISTRICT      5
DELI QUEUE TIME     6
CARRIER ID          7
DISTRICT  8
CUSTOMER            9
NEWORDER FIELDS A-X,a-u
CUST LAST NAME      Y
CUST WAREHOUSE      Z
CUST DISTRICT       v
AMOUNT PAID         w
THRESHOLD           x
#endif

#define MENU_BAR \
"<HR>"\
"<INPUT TYPE=submit NAME=0 VALUE=NewOrder>"\
"<INPUT TYPE=submit NAME=0 VALUE=Payment>"\
"<INPUT TYPE=submit NAME=0 VALUE=Delivery>"\
"<INPUT TYPE=submit NAME=0 VALUE=OrderStatus>"\
"<INPUT TYPE=submit NAME=0 VALUE=StockLevel>"\
"<INPUT TYPE=submit NAME=0 VALUE=Exit>"

static char    szFormTemplate[] =
"<BODY><FORM ACTION=%s METHOD=GET>";

static char    szWelcomeFormTemplate[] =
"<BODY><FORM ACTION=%s METHOD=GET>"
"<INPUT TYPE=hidden "PAGE_STR/**/GREETING_PAGE_ID"00>"
"Please Identify your Warehouse and District for this session.<BR>"
"Warehouse ID <INPUT NAME=4 SIZE=5><BR>"
"District ID <INPUT NAME=5 SIZE=2><BR>"
"<HR>"
"<INPUT TYPE=submit NAME=0 VALUE=Submit>"
"</FORM>"END_BODY_STR;

static char szWelcomeForm[sizeof(szWelcomeFormTemplate)+FILENAMESIZE];
static int  iWelcomeFormLen;

static char    szMainMenuFormTemplate[] =
"<BODY><FORM ACTION=%s METHOD=GET>"
"<INPUT TYPE=hidden "PAGE_STR/**/MENU_BAR_PAGE_ID"%06d>"
"%55.55s<BR>"
"Select Desired Transaction.<BR>"
MENU_BAR
"</FORM>"END_BODY_STR;

static char szDeliveryFormTemp2i[] =
"<INPUT TYPE=hidden "PAGE_STR/**/DY_FORM_PAGE_ID"######>"
"<INPUT TYPE=hidden NAME=6 VALUE=0>"
"<PRE>                          Delivery<BR>"
W_ID_STR"#####<BR><BR>"
"Carrier Number: <INPUT NAME=7 SIZE=1><BR><BR>"
EXECUTION_STATUS"<BR></PRE>"
"<HR><INPUT TYPE=submit NAME=0 VALUE=Process>"
"<INPUT TYPE=submit NAME=0 VALUE=Menu>"
"</FORM>"END_BODY_STR;

static char szDeliveryFormTemp2p[] =
"<INPUT TYPE=hidden "PAGE_STR/**/DY_RESP_PAGE_ID"######>"
"<PRE>                          Delivery<BR>"
W_ID_STR"#####<BR><BR>"
"Carrier Number: ##<BR><BR>"
EXECUTION_STATUS"Delivery has been queued.<BR>"
DELI_DATA_STR_1
```

```
"########## ######## ##### ## #### #### #### #### #### #### #### #### ####
####"
DELI_DATA_STR_2
"########## ######## ##### ## #### #### #### #### #### #### #### #### ####
####"
DELI_DATA_STR_2
"</PRE>"
MENU_BAR
"</FORM>"END_BODY_STR;

static char szNewOrderFormTemp2i[] =
"<INPUT TYPE=hidden "PAGE_STR/**/NO_FORM_PAGE_ID"######>"
"<PRE>                    New Order<BR>"
W_ID_STR"#####"
"  "D_ID_STR"<INPUT NAME=8 SIZE=2>                Date:<BR>"
"Customer:  <INPUT NAME=9 SIZE=4>   "
"Name:              Credit:     %Disc:<BR>"
O_ID_STR"        Number of Lines:         "
"W_tax:       D_tax:<BR><BR>"
" Supp_W  Item_Id  Item Name              Qty  Stock  B/G  "
"Price    Amount<BR>"
" <INPUT NAME=A SIZE=4>  <INPUT NAME=B SIZE=6>"
"                  <INPUT NAME=C SIZE=1><BR>"
" <INPUT NAME=D SIZE=4>  <INPUT NAME=E SIZE=6>"
"                  <INPUT NAME=F SIZE=1><BR>"
" <INPUT NAME=G SIZE=4>  <INPUT NAME=H SIZE=6>"
"                  <INPUT NAME=I SIZE=1><BR>"
" <INPUT NAME=J SIZE=4>  <INPUT NAME=K SIZE=6>"
"                  <INPUT NAME=L SIZE=1><BR>"
" <INPUT NAME=M SIZE=4>  <INPUT NAME=N SIZE=6>"
"                  <INPUT NAME=O SIZE=1><BR>"
" <INPUT NAME=P SIZE=4>  <INPUT NAME=Q SIZE=6>"
"                  <INPUT NAME=R SIZE=1><BR>"
" <INPUT NAME=S SIZE=4>  <INPUT NAME=T SIZE=6>"
"                  <INPUT NAME=U SIZE=1><BR>"
" <INPUT NAME=V SIZE=4>  <INPUT NAME=W SIZE=6>"
"                  <INPUT NAME=X SIZE=1><BR>"
" <INPUT NAME=a SIZE=4>  <INPUT NAME=b SIZE=6>"
"                  <INPUT NAME=c SIZE=1><BR>"
" <INPUT NAME=d SIZE=4>  <INPUT NAME=e SIZE=6>"
"                  <INPUT NAME=f SIZE=1><BR>"
" <INPUT NAME=g SIZE=4>  <INPUT NAME=h SIZE=6>"
"                  <INPUT NAME=i SIZE=1><BR>"
" <INPUT NAME=j SIZE=4>  <INPUT NAME=k SIZE=6>"
"                  <INPUT NAME=l SIZE=1><BR>"
" <INPUT NAME=m SIZE=4>  <INPUT NAME=n SIZE=6>"
"                  <INPUT NAME=o SIZE=1><BR>"
" <INPUT NAME=p SIZE=4>  <INPUT NAME=q SIZE=6>"
"                  <INPUT NAME=r SIZE=1><BR>"
" <INPUT NAME=s SIZE=4>  <INPUT NAME=t SIZE=6>"
"                  <INPUT NAME=u SIZE=1><BR>"
EXECUTION_STATUS"                              Total:<BR><HR>"
"<INPUT TYPE=submit NAME=0 VALUE=Process>"
"<INPUT TYPE=submit NAME=0 VALUE=Menu>"
"</FORM>"END_BODY_STR;

static char szNewOrderFormTemp2p[] =
"<INPUT TYPE=hidden "PAGE_STR/**/NO_RESP_PAGE_ID"######>"
"<PRE>                    New Order<BR>"
W_ID_STR"#####"
"  "D_ID_STR"##                  Date: ################### <BR>"
"Customer:  ####   Name: ################   Credit: ##   "
"%Disc: #####          <BR>"
O_ID_STR"########  Number of Lines: ##       "
"W_tax: #####   D_tax: #####  <BR><BR>"
" Supp_W  Item_Id  Item Name              Qty  Stock  B/G  "
"Price    Amount<BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  ####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
EXECUTION_STATUS"########################             "
"Total:  $########  <BR>"
"</PRE>"
MENU_BAR
"</FORM>"END_BODY_STR;

static char szOrderStatusFormTemp2i[] =
"<INPUT TYPE=hidden "PAGE_STR/**/OS_FORM_PAGE_ID"######>"
"<PRE>                    Order-Status<BR>"
W_ID_STR"#####  "D_ID_STR"<INPUT NAME=8 SIZE=1><BR>"
"Customer: <INPUT NAME=9 SIZE=4>   Name:                 "
"<INPUT NAME=Y 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=0 VALUE=Process>"
"<INPUT TYPE=submit NAME=0 VALUE=Menu>"
"</FORM>"END_BODY_STR;

static char szOrderStatusFormTemp2p[] =
"<INPUT TYPE=hidden "PAGE_STR/**/OS_RESP_PAGE_ID"######>"
"<PRE>                    Order-Status<BR>"
W_ID_STR"#####  "D_ID_STR"##<BR>"
"Customer: ####   Name: ################ ## ################<BR>"
"Cust-Balance: $#########<BR><BR>"
"Order-Number: ########   Entry-Date: ###################   Carrier-Number:
##"
"<BR>"
"Supply-W     Item-Id    Qty     Amount      Delivery-Date<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"  ####       ######     ##     $########      ##########<BR>"
"</PRE>"
MENU_BAR
"</FORM>"END_BODY_STR;

static char szPaymentFormTemp2i[] =
"<INPUT TYPE=hidden "PAGE_STR/**/PT_FORM_PAGE_ID"######>"
"<PRE>                    Payment<BR>"
"Date:  <BR><BR>"
W_ID_STR"#####                "D_ID_STR"<INPUT NAME=8 SIZE=2><BR>"
"<BR><BR><BR><BR>"
"Customer: <INPUT NAME=9 SIZE=4>"
"Cust-Warehouse: <INPUT NAME=Z SIZE=5>  "
"Cust-District: <INPUT NAME=v SIZE=1><BR>"
"Name:                 <INPUT NAME=Y SIZE=16>              "
"Since:<BR>"
"                              Credit:<BR>"
"                              Disc:<BR>"
"                              Phone:<BR><BR>"
"Amount Paid:        $<INPUT NAME=w SIZE=7>     New Cust Balance:<BR>"
```

```
"Credit Limit:<BR><BR>Cust-Data: <BR><BR><BR><BR></PRE><HR>"
"<INPUT TYPE=submit NAME=0 VALUE=Process>"
"<INPUT TYPE=submit NAME=0 VALUE=Menu>"
"</FORM>"END_BODY_STR;

static char szPaymentFormTemp2p[] =
"<INPUT TYPE=hidden "PAGE_STR/**/PT_RESP_PAGE_ID"######>"
"<PRE>                          Payment<BR>"
"Date: ################## <BR><BR>"
W_ID_STR"#####                  "D_ID_STR"##<BR>"
"###################             ###################<BR>"
"###################             ###################<BR>"
"################## ## #########      ################## ##
#########"
"<BR><BR>"
"Customer: ####  Cust-Warehouse: ##### Cust-District: ##<BR>"
"Name:   ################ ## ################     Since:  ##########<BR>"
"       ###################             Credit: ##<BR>"
"       ###################             %Disc:  #####<BR>"
"       ################## ## #########       Phone:  ################"
"<BR><BR>"
"Amount Paid:       $#######      New Cust Balance: $##############<BR>"
"Credit Limit:   $############<BR><BR>"
"Cust-Data: ################################################<BR>"
"         ################################################<BR>"
"         ################################################<BR>"
"         ################################################<BR>"
"</PRE>"
MENU_BAR
"</FORM>"END_BODY_STR;

static char szStockLevelFormTemp2i[] =
"<INPUT TYPE=hidden "PAGE_STR/**/SL_FORM_PAGE_ID"######>"
"<PRE>                          Stock-Level<BR>"
W_ID_STR"#####  "D_ID_STR"##<BR><BR>"
"Stock Level Threshold: <INPUT NAME=x SIZE=2><BR><BR>"
"low stock:    <BR><HR>"
"<INPUT TYPE=submit NAME=0 VALUE=Process>"
"<INPUT TYPE=submit NAME=0 VALUE=Menu>"
"</FORM>"END_BODY_STR;

static char szStockLevelFormTemp2p[] =
"<INPUT TYPE=hidden "PAGE_STR/**/SL_RESP_PAGE_ID"######>"
"<PRE>                          Stock-Level<BR>"
W_ID_STR"#####  "D_ID_STR"##<BR><BR>"
"Stock Level Threshold: ##<BR><BR>"
"low stock: ###"
"</PRE>"
MENU_BAR
"</FORM>"END_BODY_STR;

static char szErrorFormTemplate[] =
"<BODY><FORM ACTION=%s METHOD=GET>"
"<INPUT TYPE=hidden "PAGE_STR/**/ERROR_PAGE_ID"%06d>"
ERROR_VALUE_STR"%d: %s"
MENU_BAR
"</FORM>"END_BODY_STR;

static char szResponseHeaderTemplate[] =
"Connection: keep-alive\r\n"
"Content-Type: text/html\r\n"
"Content-Length: ####\r\n"
"\r\n";
static char  szResponseHeader[sizeof(szResponseHeaderTemplate)];
FORM_INDEXES responseHeaderIndexes[1] = { 0 };
int                 responseHeaderLen = 0;


#define MATCHES_BEGIN(p) ('B'==p[0])
#define MATCHES_CHECKPOINT(p) \
(0==strncmp(p,"Checkpoint",strlen("Checkpoint")))
#define MATCHES_CHECKPOINT_STARTUP(p) \
(0==strncmp(p,"CheckpointStartup",strlen("CheckpointStartup")))
#define MATCHES_CHECKPOINT_SHUTDOWN(p) \
(0==strncmp(p,"CheckpointShutdown",strlen("CheckpointShutdown")))
#define MATCHES_CLEAR(p) ('C'==p[0]&&'l'==p[1])
#define MATCHES_DELIVERY(p) ('D'==p[0])
#define MATCHES_EXIT(p) ('E'==p[0])
#define MATCHES_MENU(p) ('M'==p[0])
#define MATCHES_NEWORDER(p) ('N'==p[0])
#define MATCHES_ORDERSTATUS(p) ('O'==p[0])
#define MATCHES_PAYMENT(p) ('P'==p[0]&&'a'==p[1])
#define MATCHES_PROCESS(p) ('P'==p[0]&&'r'==p[1])
#define MATCHES_STOCKLEVEL(p) ('S'==p[0]&&'t'==p[1])
#define MATCHES_SUBMIT(p) ('S'==p[0]&&'u'==p[1])
#ifdef FFE_DEBUG
# define MATCHES_MEMORYCHECK(p) ('!'==p[0]&&'M'==p[1])
#endif

/* function prototypes */
void BeginCmd( EXTENSION_CONTROL_BLOCK *pECB );
void CheckpointCmd( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id
);
void CheckpointStartupCmd( EXTENSION_CONTROL_BLOCK *pECB, int w_id,
int ld_id );
void CheckpointShutdownCmd( EXTENSION_CONTROL_BLOCK *pECB, int
w_id, int ld_id);
void ClearCmd( EXTENSION_CONTROL_BLOCK *pECB );
void ExitCmd( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id );
void MenuCmd( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id );
void SubmitCmd( EXTENSION_CONTROL_BLOCK *pECB, int *w_id, int *ld_id );
void MemoryCheckCmd( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int
ld_id );

BOOL GetKeyValuePtr( char *szIPtr, char *szKey, char **pszOPtr );
BOOL GetCharKeyValuePtr( char *szIPtr, char cKey, char **pszOPtr );
BOOL GetKeyValueString( char *szIPtr, char *szKey,
                                        char *szValue, int iSize );
BOOL GetWDID(char *ptr, int *lw_id, int *ld_id, char **optr);

int  GetCallingThreadLimit( int *pCallingThreadLimit );
void Log( char *szType, char *szStr );
void MakePanicPool( DWORD dwResponseSize );
void MakeResponseHeader( void );
void MakeTemplatePool( DWORD dwFormSize, DWORD dwResponseSize );
void DeletePanicPool( void );
void DeleteTemplatePool( void );

int ProcessDeliveryQuery( EXTENSION_CONTROL_BLOCK *pECB,
                                        char *lpszQueryString,
                                        int w_id, int ld_id );
int ProcessNewOrderQuery( EXTENSION_CONTROL_BLOCK *pECB,
                                        char *lpszQueryString,
                                        int w_id, int ld_id );
int ProcessOrderStatusQuery( EXTENSION_CONTROL_BLOCK *pECB,
                                          char *lpszQueryString,
                                          int w_id, int ld_id );
int ProcessPaymentQuery( EXTENSION_CONTROL_BLOCK *pECB,
                                       char *lpszQueryString,
                                       int w_id, int ld_id );
int ProcessStockLevelQuery( EXTENSION_CONTROL_BLOCK *pECB,
                                         char *lpszQueryString,
                                         int w_id, int ld_id );

DWORD ProcessQueryString(EXTENSION_CONTROL_BLOCK *pECB);

void PutNumeric( int iInt, int iFieldSize, char *pChar );
void SendErrorResponse( EXTENSION_CONTROL_BLOCK *pECB, int iError,
                                      char *szMsg, int w_id, int ld_id, pConnData
pConn );
void SendMainMenuForm( EXTENSION_CONTROL_BLOCK *pECB,
                                  int w_id, int ld_id, char *szStatus );
void SendResponse(EXTENSION_CONTROL_BLOCK *pECB, char *szStr, int
iStrLen);
void SendWelcomeForm(EXTENSION_CONTROL_BLOCK *pECB);
#ifdef FFE_DEBUG
unsigned __stdcall CheckMemory(void *param);
#endif

/* typedefs */
typedef struct
{
  char *szStr;
  int iIndex;
  int iFieldSize;
  int iNewIndex;
  int iNewFieldSize;
} PutStrStruct, *pPutStrStruct;

typedef struct
```

```
{
  CRITICAL_SECTION critSec;
#ifdef FFE_DEBUG
  int           iMaxIndex;
#endif
  int           iNextFree;
  char          *index[1];
  char          forms[PANIC_FORM_SIZE];
} PanicStruct, *pPanicStruct;

typedef struct
{
  CRITICAL_SECTION critSec[NUMBER_POOL_FORM_TYPES];
#ifdef FFE_DEBUG
  int           iMaxIndex[NUMBER_POOL_FORM_TYPES];
#endif
  int           iNextFreeForm[NUMBER_POOL_FORM_TYPES];
  int           iFirstFormIndex[NUMBER_POOL_FORM_TYPES];
  char          *index[1];
  char          forms[1];
} FormStruct, *pFormStruct;

typedef struct
{
  CRITICAL_SECTION critSec[NUMBER_POOL_RESPONSE_TYPES];
#ifdef FFE_DEBUG
  int           iMaxIndex[NUMBER_POOL_RESPONSE_TYPES];
#endif
  int           iNextFreeResponse[NUMBER_POOL_RESPONSE_TYPES];
  int           iFirstResponseIndex[NUMBER_POOL_RESPONSE_TYPES];
  char          *index[1];
  char          responses[1];
} ResponseStruct, *pResponseStruct;

/* global variables */
static int                  iInitStatus    = ERR_SUCCESS;
static CRITICAL_SECTION startupCriticalSection;
static BOOL           startupFlag    = FALSE;

static BOOL               bLog                        = FALSE;
static char               szModName[FILENAMESIZE] = { '\0' };
static char               szPath[FILENAMESIZE]              = { '\0' };

static pPanicStruct gpPanicForms      = NULL;
static int giPanic                              = 0;
static pFormStruct gpForms                      = 0;
static int giFormLen[NUMBER_POOL_FORM_TYPES] = { 0 };
static pResponseStruct gpResponses    = 0;
static int giResponseLen[NUMBER_POOL_RESPONSE_TYPES] = { 0 };

/* FUNCTION: BOOL APIENTRY DllMain(HANDLE hModule, DWORD
ul_reason_for_call,
 *                                                LPVOID lpReserved)
 *
 * PURPOSE:           This is the main entry point to an ISAPI dll.  All dll
 *                    global initializations should be done in this routine.
 *
 * ARGUMENTS:         HANDLE   hModule                              dll
module handle
 *                    DWORD    ul_reason_for_call       reason for call
 *                    LPVOID   lpReserved               reserved for future use
 *
 * RETURNS:           BOOL     Always TRUE                  Errors in
intiialization
 *                                                                  are
presented at the first
 *                                                                  screen to
the user.
 * COMMENTS:          None
 *
 */

BOOL APIENTRY
DllMain(HANDLE hModule, DWORD ul_reason_for_call, LPVOID lpRes)
{
  char                szTmpFileName[FILENAMESIZE];
  DWORD               dwFileNameLen;
  char                *pChr;

  switch( ul_reason_for_call )
  {
  case DLL_PROCESS_ATTACH:
#ifdef FFE_DEBUG
    tmpDbgFlag = _CrtSetDbgFlag(_CRTDBG_REPORT_FLAG);
    tmpDbgFlag |= _CRTDBG_CHECK_CRT_DF;
    tmpDbgFlag |= _CRTDBG_LEAK_CHECK_DF;
    tmpDbgFlag |= _CRTDBG_ALLOC_MEM_DF;
    tmpDbgFlag |= _CRTDBG_CHECK_ALWAYS_DF;
    _CrtSetDbgFlag(tmpDbgFlag);

    hMemFile = CreateFile( "MemErrors", GENERIC_WRITE, FILE_SHARE_READ,
NULL,
                                   OPEN_ALWAYS,
FILE_ATTRIBUTE_NORMAL, NULL );

    _CrtSetReportMode( _CRT_WARN, _CRTDBG_MODE_FILE );
    _CrtSetReportFile( _CRT_WARN, hMemFile );
    _CrtSetReportMode( _CRT_ERROR, _CRTDBG_MODE_FILE );
    _CrtSetReportFile( _CRT_ERROR, hMemFile );
    _CrtSetReportMode( _CRT_ASSERT, _CRTDBG_MODE_FILE );
    _CrtSetReportFile( _CRT_ASSERT, hMemFile );
#endif
#ifdef DEBUG_ENTRY
    *( int * )0 = 1;
#endif
    InitializeCriticalSection( &startupCriticalSection );

    dwFileNameLen = GetModuleFileName( hModule, szTmpFileName,
FILENAMESIZE-1);
    if( 0 == dwFileNameLen )
      return FALSE;

    pChr = strrchr( szTmpFileName, '\\' );
    if( NULL == pChr )
      return FALSE;

    pChr++;
    dwFileNameLen = strlen( pChr );
    if( 0 >= dwFileNameLen )
      return FALSE;

    CopyMemory( szModName, pChr, dwFileNameLen+1 );

    iWelcomeFormLen = sprintf(szWelcomeForm, szWelcomeFormTemplate,
szModName);

    if( ERR_SUCCESS != iInitStatus )
      return TRUE;

    break;
  case DLL_THREAD_ATTACH:
    break;
  case DLL_THREAD_DETACH:
    break;
  case DLL_PROCESS_DETACH:

    pTPCCShutdown( );

    pDeleteTransactionPool( );
    DeleteTemplatePool( );
    DeletePanicPool( );

    DeleteCriticalSection( &startupCriticalSection );

    pTPCCCloseLog( );

    break;
  }
  return TRUE;
}

/* FUNCTION: BOOL WINAPI GetExtensionVersion(HSE_VERSION_INFO
*pVersion)
 *
 * PURPOSE:           This function is called by the inet service when the DLL
 *                    is first loaded.
 *
 * ARGUMENTS:         HSE_VERSION_INFO  *pVersion     passed in structure in
 *                                                                  which to
place expected
```

```
 *                                                                    version
number.
 *
 * RETURNS:          TRUE                                             expected
return value.
 *
 * COMMENTS:         None
 *
 */

BOOL WINAPI
GetExtensionVersion(HSE_VERSION_INFO *pVersion)
{

  pVersion->dwExtensionVersion =
    MAKELONG(HSE_VERSION_MINOR, HSE_VERSION_MAJOR);
  strncpy(pVersion->lpszExtensionDesc,
                "Digital TPC-C Server.",
                HSE_MAX_EXT_DLL_NAME_LEN-1);
  *(pVersion->lpszExtensionDesc) = '\0';

  return TRUE;
}

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

DWORD WINAPI
HttpExtensionProc(EXTENSION_CONTROL_BLOCK *pECB)
{
#pragma message ("FIXME: If the server is set improperly, and the delivery servers
fail to log in in a min TM sybase config, the failure is not reported until the user tries
to log in.")
  int                           Len = sizeof( szPath ) - 1;
  StartupData                   Startup;
  pStartupData                  pStartup = &Startup;
  DWORD                         status;
  int                           dbstatus;
  int                           formPoolSize;
  int                           responsePoolSize;
  int                           panicPoolSize = 0;
  int                           transactionPoolSize;
  int                           transportThreadLimit;
  int                           callingThreadLimit;
  int                           transMode;
#ifdef FFE_DEBUG
  unsigned long                 ulHandle;
  unsigned          uTAddr;
#endif

  if ( ! startupFlag ) {
    EnterCriticalSection( &startupCriticalSection );
    if ( ! startupFlag ) {
      if( ERR_SUCCESS == iInitStatus ) {
            MakeResponseHeader( );
            pStartup->Path = szPath;
      }
      if( ERR_SUCCESS == iInitStatus ) {
            iInitStatus = GetCallingThreadLimit( &callingThreadLimit );
      }
      if( ERR_SUCCESS == iInitStatus ) {
            iInitStatus = TPCCGetConfig( &bLog,
                                         &Len, pStartup->Path,
                                         &pStartup->Login, &pStartup-
>loginDelay,
                                         &pStartup->Config,
                                         &pStartup->Transport,
                                         &pStartup->DeliveryTransport
);
      }
      if ( ERR_SUCCESS == iInitStatus ) {
            iInitStatus = TPCCTMLoad( );
      }
      if ( ERR_SUCCESS == iInitStatus ) {
            pTPCCOpenLog( szPath );
      }
      if ( ERR_SUCCESS == iInitStatus ) {
            iInitStatus = pTPCCGetTransportMode( &transMode );
      }
      if( ERR_SUCCESS == iInitStatus ) {
            /* The transaction pool size represents the maximum number of */
            /* transactions that can be carried by any part of the system. */
            /* Start out with each calling thread able to carry a transaction. */
            transactionPoolSize = callingThreadLimit;

            /* We need to see how we should run the transport. */
            pStartup->Transport.asynchronous &= ( TRANS_ASYNC == transMode
);

            if( pStartup->Transport.generic ) {
              transportThreadLimit = pStartup->Transport.num_gc;
            }
            else {
              transportThreadLimit = pStartup->Transport.num_dy +
                pStartup->Transport.num_no + pStartup->Transport.num_os +
                pStartup->Transport.num_pt + pStartup->Transport.num_sl;
            }

            /* Menu transactions are only carried by the web server */
            /* and hence are limited by the count of web server threads. */
            formPoolSize = callingThreadLimit;

            if( pStartup->Transport.asynchronous ) {
              /* Delivery transactions and their interactive response */
              /* are carried by the web server threads since they are */
              /* synchronous.  Database transactions and their responses */
              /* are limited by transport threads.  Therefore the maximum */
              /* number of concurrent responses is the greater of the */
              /* web server and transport thread counts. */
              responsePoolSize = MAX( callingThreadLimit, transportThreadLimit );

              /* Panic forms are used either to report errors or to */
              /* handle the case where a response required the quoting */
              /* of a character that is an HTML special character. */
              /* The maximum number of these forms is also the greater of the */
              /* web server and transport thread counts. */
              panicPoolSize = MAX( callingThreadLimit, transportThreadLimit );

              /* Asynchronous transport threads can carry a transaction. */
              transactionPoolSize += transportThreadLimit;
            }
            else {
              /* If the transport runs synchronously, only web server */
              /* threads can generate any form of response. */
              responsePoolSize = callingThreadLimit;
              panicPoolSize = callingThreadLimit;
            }

            /* Add the transactions carried by the delivery subsystem. */
            transactionPoolSize += pStartup->DeliveryTransport.num_threads +
              pStartup->DeliveryTransport.num_queued_deliveries +
              pStartup->DeliveryTransport.num_queued_responses;

            MakeTemplatePool( formPoolSize, responsePoolSize );
            MakePanicPool( panicPoolSize );
            pMakeTransactionPool( transactionPoolSize );
            pStartup->pTPCCNewOrderResponse = TPCCNewOrderResponse;
            pStartup->pTPCCOrderStatusResponse = TPCCOrderStatusResponse;
            pStartup->pTPCCPaymentResponse = TPCCPaymentResponse;
```

```
            pStartup->pTPCCStockLevelResponse = TPCCStockLevelResponse;
            pStartup->pTPCCResponseComplete = TPCCResponseComplete;
            dbstatus = pTPCCStartup( pStartup );
            if( ERR_DB_SUCCESS != dbstatus ) {
              iInitStatus = dbstatus;
            }
#ifdef FFE_DEBUG
            ulHandle = _beginthreadex( NULL, 0, CheckMemory, NULL, 0,
&uTAddr );
            _ASSERT( 0 != ulHandle );
#endif
     }
     startupFlag = TRUE;
   }
   LeaveCriticalSection( &startupCriticalSection );
  }

  if( ERR_SUCCESS != iInitStatus )
  {
   if( NULL == gpPanicForms ) {
     MakePanicPool( 50 );
   }
   SendErrorResponse(pECB, iInitStatus, NULL, -1, -1, NULL);
   return HSE_STATUS_SUCCESS;
  }

  /* if registry setting is for html logging then show http string passed in.*/
  if ( bLog )
  {
   pTPCCLog( "%s %s\r\n", "* QUERY    *", pECB->lpszQueryString );
  }

  /* process http query */
  status = ProcessQueryString(pECB);

  /* finish up with status returned by Processing functions */
  return status;
}

/* FUNCTION: void SendErrorResponse( EXTENSION_CONTROL_BLOCK
*pECB, int iError,
 *                                                  char *szMsg,
 *                                                  int w_id, int ld_id )
 *
 * PURPOSE:              This function displays an error form in the client browser.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB        IIS
context structure pointer
 *                                                                 unique to
this connection.
 *                       int            iError                  id of error
message
 *                       int            w_id                    Login
warehouse ID.
 *                       int            ld_id                   Login
district ID.
 *                       char           *szMsg                  optional
error message string
 *                                                               used with
ERR_TYPE_SQL and
 *
                ERR_TYPE_DBLIB
 *
 * RETURNS:              None
 *
 */

void
SendErrorResponse( EXTENSION_CONTROL_BLOCK *pECB, int iError,
                   char *szMsg, int w_id, int ld_id, pConnData pConn )
{
  char                  *szForm;
  int                   iStrLen;
  char                  *szDefaultErr = "Unable to look up this error
code.";
  char                  *szErrorMsg;
  static char           szNoMsg[] = "";

  if ( !szMsg  )
    szMsg = szNoMsg;

  RESERVE_PANIC_FORM( szForm );

  szErrorMsg = ( NULL == pTPCCErrString ) ?
    szDefaultErr : pTPCCErrString( iError );

  if( NULL != pTPCCErr )
  {
   if( NULL != pConn )
     pTPCCErr( "Transaction error. w_id: %d, ld_id: %d, CC: %I64x, "
                   "status: %d, dbstatus: %d, (%d): %s\r\n",
                   pConn->w_id, pConn->ld_id, pConn->CC,
                   pConn->status, pConn->dbstatus, iError, szErrorMsg );
   else
     pTPCCErr( "(%d): %s\r\n", iError, szErrorMsg );
  }

  iStrLen = sprintf( szForm, szErrorFormTemplate, szModName,
                     WDID(w_id,ld_id), iError, szErrorMsg );

  SendResponse(pECB, szForm, iStrLen);

  UNRESERVE_PANIC_FORM( szForm );
}

/* FUNCTION: void HandlePanic(pPutStrStruct pStruct,
 *                                                  char *szInput, int iInputSize,
 *                                                  char **szOutput, int *iOutputSize )
 *
 * PURPOSE:   This routine handles the case where the output string contains
 * at least one of the special characters double quote ("), ampersand (&),
 * less than (<), or greater than (>).  What it does is scan the strings
 * to be output checking for all special characters.  It then moves the
 * input string template sections further along in the output string
 * making enough room for the strings including their special quoted
 * charaters, then fills the new template with the output strings.
 *
 * ARGUMENTS:
 *
 * RETURNS:              void
 *
 * COMMENTS:
 */

void
HandlePanic( pPutStrStruct pStruct,
             char *szInput, int iInputSize,
             char **szOutput, int *iOutputSize )
{
  pPutStrStruct pStructTmp1;
  pPutStrStruct pStructTmp2;
  char *pIChar;
  int iExtra;
  int iTotalExtra;
  char *szTmp;

  RESERVE_PANIC_FORM( szTmp );

  /* first, save what we've done so far */
  *szOutput = szTmp;
  memcpy( szTmp, szInput, pStruct->iIndex );

  /* save the original values for string moving */
  pStructTmp1 = pStruct;
  while( NULL != pStructTmp1->szStr ) {
    pStructTmp1->iNewIndex = pStructTmp1->iIndex;
    pStructTmp1->iNewFieldSize = pStructTmp1->iFieldSize;
    pStructTmp1++;
  }

  /* parse all remaining strings for special characters and fix indicies */
  pStructTmp1 = pStruct;
  iTotalExtra = 0;
  while( NULL != pStructTmp1->szStr ) {
    pIChar = pStructTmp1->szStr;
    iExtra = 0;
    while( 0 != *pIChar )
    {
      if( "' == *pIChar )
            iExtra += 5;
```

```
    else if( '&' == *pIChar )
                iExtra += 4;
    else if( '<' == *pIChar )
                iExtra += 3;
    else if( '>' == *pIChar )
                iExtra += 3;
    pIChar++;
   }

   /* reset field width for this string */
   pStructTmp1->iNewFieldSize += iExtra;

   /* move all following indicies */
   for( pStructTmp2 = pStructTmp1+1;
                NULL != pStructTmp2->szStr;
                pStructTmp2++ )
    pStructTmp2->iNewIndex += iExtra;

   pStructTmp1++;
   iTotalExtra += iExtra;
  }

  /* update new string length */
  *iOutputSize = iInputSize + iTotalExtra;

  /* move end of string to new output string */
  --pStructTmp1;
  memcpy( &szTmp[pStructTmp1->iNewIndex + pStructTmp1->iNewFieldSize],
                &szInput[pStructTmp1->iIndex + pStructTmp1->iFieldSize],
                iInputSize - pStructTmp1->iIndex + pStructTmp1->iFieldSize);

  /* move input string pieces to new locations in output string */
  pStructTmp2 = pStructTmp1--;
  while( pStruct != pStructTmp2 )
  {
   memcpy( &szTmp[pStructTmp1->iNewIndex + pStructTmp1->iNewFieldSize],
                &szInput[pStructTmp1->iIndex + pStructTmp1->iFieldSize],
                pStructTmp2->iIndex -
                ( pStructTmp1->iIndex + pStructTmp1->iFieldSize ));
   pStructTmp2 = pStructTmp1--;
  }

  /* Now put in the strings */
  pStructTmp1 = pStruct;
  while( NULL != pStructTmp1->szStr ) {
   CONVERT_SPECIAL( &szTmp[pStructTmp1->iNewIndex], pStructTmp1->szStr,
                        pStructTmp1->iNewFieldSize );
   pStructTmp1++;
  }
}

/* FUNCTION: void SendResponse(EXTENSION_CONTROL_BLOCK *pECB, char
*szForm,
 *                                      int iStrLen)
 *
 * PURPOSE:
 *    This function takes the forms generated by each transaction routine
 *    and calls the server callback function to pass it on to the browser.
 *
 * ARGUMENTS:
 *    EXTENSION_CONTROL_BLOCK       *pECB           Server
context structure.
 *    char                          *szForm         form to pass to
browser.
 *    int                           iStrLen         length of form
excluding null.
 *
 * RETURNS:
 *    None
 *
 * COMMENTS:
 */

void
SendResponse(EXTENSION_CONTROL_BLOCK *pECB, char *szForm, int
iStrLen)
{
  char                  szHeader1[sizeof(szResponseHeader)];
  static char  szHeader[] = "200 Ok";
  static int     iSize = sizeof(szHeader)-1;
  int                   lpbSize;

  lpbSize = iStrLen + 1;

  if ( bLog )
    pTPCCLog( "%s %s\r\n", "* RESPONSE *", szForm );

  CopyMemory( szHeader1, szResponseHeader, sizeof(szResponseHeader) );
  PutNumeric(lpbSize, responseHeaderIndexes[0].iLen,
                &szHeader1[responseHeaderIndexes[0].iStartIndex]);

  (*pECB->ServerSupportFunction)(pECB->ConnID,
HSE_REQ_SEND_RESPONSE_HEADER,
                                        szHeader, &iSize,
(LPDWORD)szHeader1);
  (*pECB->WriteClient)(pECB->ConnID, szForm, &lpbSize, 0);
}

/* FUNCTION: ParseTemplateString(char *szForm, int *pcurLen,
 *                                      char *formTemplate,
FORM_INDEXES *indexes)
 *
 * PURPOSE:      This function parses the query string to find the ## signs
 *               that mark the positions for the values to be put, and
 *               stores these locations and lengths in the indexes structure.
 *
 * ARGUMENTS: char   *szForm       the resultant form
 *            int    *pcurLen the current length of szForm
 *            char   *formTemplate       the form's template
 *            FORM_INDEXES *indexes      ptr to the array of indexes for the
 *                                       tag values of the form
 *
 * RETURNS:      void
 *
 * COMMENTS:
 */

void
ParseTemplateString(char *szForm, int *pcurLen,
                        char *formTemplate, FORM_INDEXES *indexes)
{
  int   curIndex = 0;
  int           ii = 0;
  int           jj;
  int           curLen;

  curLen = *pcurLen;
  while ('\0' != formTemplate[ii])
  {
    if('#' != formTemplate[ii])
    {
      szForm[curLen] = formTemplate[ii];
      ii++;
      curLen++;
    }
    else
    {
      jj = 0;
      indexes[curIndex].iStartIndex = curLen;
      while('#' == formTemplate[ii])
      {
            jj++;
            szForm[curLen] = formTemplate[ii];
            curLen++;
            ii++;
      }
      indexes[curIndex].iLen = jj;
      curIndex++;
    }
  }
  szForm[curLen] = '\0';
   *pcurLen = curLen;
}

/* FUNCTION: void PutNumeric(int iInt, int iFieldSize, char *pChar )
 *
 * PURPOSE:      This function converts an integer to a char string.
 *
 * ARGUMENTS:    int     iInt            the integer to convert
to string
```

```
 *                      int         iFieldSize    max size of char string to return.
 *                      char        *pChar                the string to put the int
into.
 *
 * RETURNS:            None
 *
 * COMMENTS:           If the Integer value exceeds the max field size, then
 *                     the string will be filled with iFieldSize "*" to signal
 *                     an error.
 */

void
PutNumeric( int iInt, int iFieldSize, char *pChar )
{
  int iSaveSize = iFieldSize;
  char *pSaveStart = pChar;
  char pAsterisk[] = "**********";
  BOOL bSignFlag = TRUE;

  pChar += (iFieldSize - 1);
  if(0 > iInt)
  {
    bSignFlag = FALSE;
    iInt = abs(iInt);
  }

  do
  {
    *pChar = ( iInt % 10 ) + '0';
    iInt /= 10;
    iFieldSize--;
    if( iFieldSize )
      pChar--;
  } while( iFieldSize );

  if( !bSignFlag )
  {
    if('0' == *pChar)
      *pChar = '-';
    else
    {
      memcpy( pSaveStart, pAsterisk, iSaveSize );
      return;
    }
  }

  if( 0 != iInt )
  {
    /* put in string of ** to signal error */
    memcpy( pSaveStart, pAsterisk, iSaveSize );
  }
}

/* FUNCTION: void SendDeliveryForm( EXTENSION_CONTROL_BLOCK
*pECB,
 *                                                  int w_id, int ld_id )
 *
 * PURPOSE:            This function puts the data into the input form and then
 *                     returns the form to the browser.
 *
 * ARGUMENTS:          EXTENSION_CONTROL_BLOCK   *pECB       structure
pointer to passed in
 *                                                           internet
service information.
 *                     int         w_id                      Login
warehouse ID.
 *                     int         ld_id                     Login
district ID.
 &
 * RETURNS:                        None
 *
 * COMMENTS:           None
 *
 */

void
SendDeliveryForm( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id )
{
  char          *deliveryForm;

  RESERVE_FORM( DELIVERY_FORM, deliveryForm );

  PutNumeric(WDID(w_id,ld_id),
                deliveryFormIndexesI[D_WDID].iLen,
                &deliveryForm[deliveryFormIndexesI[D_WDID].iStartIndex]);
  PutNumeric(w_id,
                deliveryFormIndexesI[D_WID].iLen,
                &deliveryForm[deliveryFormIndexesI[D_WID].iStartIndex]);

  SendResponse(pECB, deliveryForm, giFormLen[DELIVERY_FORM]);

  UNRESERVE_FORM( DELIVERY_FORM, deliveryForm );
}

/* FUNCTION: void SendNewOrderForm( EXTENSION_CONTROL_BLOCK
*pECB,
 *                                                  int w_id, int ld_id )
 *
 * PURPOSE:            This function puts the data into the input form and then
 *                     returns the form to the browser.
 *
 * ARGUMENTS:          EXTENSION_CONTROL_BLOCK   *pECB       pointer to
the structure that
 *                                                           is passed
in the internet
 *                     int                       w_id        warehouse id
 *                     int                       ld_id       login district id
 *
 * RETURNS:            None
 *
 * COMMENTS:           None
 *
 */

void
SendNewOrderForm( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id
)
{
  char          *newOrderForm;

  RESERVE_FORM( NEW_ORDER_FORM, newOrderForm );

  PutNumeric(WDID(w_id,ld_id),
                newOrderFormIndexes[NO_WDID].iLen,
                &newOrderForm[newOrderFormIndexes[NO_WDID].iStartIndex]);
  PutNumeric(w_id,
                newOrderFormIndexes[NO_WID].iLen,
                &newOrderForm[newOrderFormIndexes[NO_WID].iStartIndex]);

  SendResponse(pECB, newOrderForm, giFormLen[NEW_ORDER_FORM]);

  UNRESERVE_FORM( NEW_ORDER_FORM, newOrderForm );
}

/* FUNCTION: void SendPaymentForm(EXTENSION_CONTROL_BLOCK *pECB,
 *                                                  int w_id, int ld_id, DBContext
*pdb)
 *
 * PURPOSE:            This function puts the data into the input form and then
 *                     returns the form to the browser.
 *
 * ARGUMENTS:
 *                     EXTENSION_CONTROL_BLOCK   *pECB       pointer to
structure passed in
 *                                                           the
internet
 *                     int                       w_id
          warehouse id
 *                     int                       ld_id       login
district id
 *
 * RETURNS:            None
 *
 * COMMENTS:           None
 *
 */

void
SendPaymentForm( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id )
{
```

```
 char        *paymentForm;

 RESERVE_FORM( PAYMENT_FORM, paymentForm );

 PutNumeric(WDID(w_id,ld_id),
                paymentFormIndexes[PT_WDID_INPUT].iLen,
&paymentForm[paymentFormIndexes[PT_WDID_INPUT].iStartIndex]);
  /* the date field is before wid for the response so use 2 here */
 PutNumeric(w_id,
                paymentFormIndexes[PT_WID_INPUT].iLen,
&paymentForm[paymentFormIndexes[PT_WID_INPUT].iStartIndex]);

 SendResponse(pECB, paymentForm, giFormLen[PAYMENT_FORM]);

 UNRESERVE_FORM( PAYMENT_FORM, paymentForm );
}

/* FUNCTION: void SendOrderStatusForm(EXTENSION_CONTROL_BLOCK
*pECB,
 *                                                  int w_id, int ld_id,
DBContext *pdb)
 *
 * PURPOSE:          This function fills in data and then sends the order status
 *                   input form back to the browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK *pECB       ptr to
structure passed in the
 *                                                              internet.
 *                   int                 w_id        warehouse id
 *                   int                 ld_id       login district id
 *
 * RETURNS:          None
 *
 * COMMENTS:         None
 *
 */

void
SendOrderStatusForm( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int
ld_id )
{
 char        *orderStatusForm;

 RESERVE_FORM( ORDER_STATUS_FORM, orderStatusForm );

 PutNumeric(WDID(w_id,ld_id),
                orderStatusFormIndexes[OS_WDID].iLen,
                &orderStatusForm[orderStatusFormIndexes[OS_WDID].iStartIndex]);
 PutNumeric(w_id,
                orderStatusFormIndexes[OS_WID].iLen,
                &orderStatusForm[orderStatusFormIndexes[OS_WID].iStartIndex]);
 SendResponse(pECB, orderStatusForm, giFormLen[ORDER_STATUS_FORM]);

 UNRESERVE_FORM( ORDER_STATUS_FORM, orderStatusForm );
}

/* FUNCTION: void SendStockLevelForm(EXTENSION_CONTROL_BLOCK
*pECB,
 *                                                  int w_id, int d_id,
DBContext *pdb)
 *
 * PURPOSE:          This function puts the data into the input form and then
 *                   returns the form to the browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK   *pECB       structure
pointer to passed
 *                                                              in internet
service information
 *                   int                 w_id
            warehouse id
 *                   int                 d_id                district id
 *                   DBContext  *pdb                 pointer to database
context.
 *
 * RETURNS:          None
 *
 * COMMENTS:         None
 *
 */

void
SendStockLevelForm( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int d_id
)
{
 char        *stockLevelForm;

 RESERVE_FORM( STOCK_LEVEL_FORM, stockLevelForm );

 PutNumeric(WDID(w_id,d_id),
                stockLevelFormIndexes[SL_WDID].iLen,
                &stockLevelForm[stockLevelFormIndexes[SL_WDID].iStartIndex]);
 PutNumeric(w_id,
                stockLevelFormIndexes[SL_WID].iLen,
                &stockLevelForm[stockLevelFormIndexes[SL_WID].iStartIndex]);
 PutNumeric(d_id,
                stockLevelFormIndexes[SL_DID].iLen,
                &stockLevelForm[stockLevelFormIndexes[SL_DID].iStartIndex]);

 SendResponse(pECB, stockLevelForm, giFormLen[STOCK_LEVEL_FORM]);

 UNRESERVE_FORM( STOCK_LEVEL_FORM, stockLevelForm );
}


/* FUNCTION: void SendMainMenuForm(EXTENSION_CONTROL_BLOCK
*pECB,
 *                                                      int w_id, int ld_id, char
*szStatus)
 *
 * PURPOSE:          This function sends the main menu form to the browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK   *pECB       IIS
context structure pointer
 *                                                              unique to
this connection.
 *
 *                   int         w_id
            warehouse id
 *                   int         ld_id                               login
district id
 *                   char        *szStatus           String to report
previous
 *                                                              operation
status.
 *
 * RETURNS:          None
 *
 * COMMENTS:
 */

void
SendMainMenuForm( EXTENSION_CONTROL_BLOCK *pECB,
                  int w_id, int ld_id, char *szStatus )
{
 char        *szForm;
 int         iStrLen;
 static char  *szNoStatus = "";
 char        *pszStatus;

 pszStatus = ( NULL == szStatus ) ? szNoStatus : szStatus;

 RESERVE_PANIC_FORM( szForm );

 iStrLen = sprintf( szForm, szMainMenuFormTemplate,
                    szModName, WDID(w_id,ld_id), pszStatus );

 SendResponse(pECB, szForm, iStrLen);

 UNRESERVE_PANIC_FORM( szForm );
}

/* FUNCTION: void SendWelcomeForm(EXTENSION_CONTROL_BLOCK
*pECB)
 *
 * PURPOSE:          This function sends the welcome form to the browser.
 *
 * ARGUMENTS:   None
 *
```

```
 * RETURNS:              None
 *
 * COMMENTS:             The welcome form is generated on initialization.
 */

void
SendWelcomeForm(EXTENSION_CONTROL_BLOCK *pECB)
{
  SendResponse( pECB, szWelcomeForm, iWelcomeFormLen );
}

/* FUNCTION: DWORD ProcessQueryString(EXTENSION_CONTROL_BLOCK
*pECB)
 *
 * PURPOSE:              This function extracts the relevent information out
 *                       of the http command passed in from the browser.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB        IIS
context structure pointer
 *                                                              unique to
this connection.
 *
 * RETURNS:              DWORD                                  server
connection status code
 *
 * COMMENTS:             If this is the initial connection i.e. client is at
 *                       welcome screen then there will not be a terminal id or
 *                       current form id if this is the case then the pTermid and
 *                       pFormid return values are undefined.
 */

DWORD
ProcessQueryString(EXTENSION_CONTROL_BLOCK *pECB)
{
  static char *beginptr = "Begin";
  char *ptr;
  char *cmdptr;
  int cFormID;
  int w_id;
  int ld_id;
  DWORD status;
  int retcode;

  w_id = 0;
  ld_id = 0;

  if ( GetCharKeyValuePtr( pECB->lpszQueryString, '3', &ptr ))
  {
    cFormID = *ptr++;
    if( !GetWDID( ptr, &w_id, &ld_id, &ptr )) {
      SendErrorResponse( pECB, ERR_W_ID_INVALID, NULL, w_id, ld_id, NULL );
      return HSE_STATUS_SUCCESS_AND_KEEP_CONN;
    }
  }
  else
    cFormID = '\0';

  /* now figure out what command we have and execute it */
  if ( !GetCharKeyValuePtr( ptr, '0', &cmdptr ))
  {
    if( 0 == strlen( pECB->lpszQueryString )) {
      cmdptr = beginptr;
    }
    else {
      SendErrorResponse( pECB, ERR_COMMAND_UNDEFINED, NULL, w_id,
ld_id, NULL );
      return HSE_STATUS_SUCCESS_AND_KEEP_CONN;
    }
  }

  if( '\0' == cFormID && !MATCHES_BEGIN( cmdptr )) {
    SendErrorResponse( pECB, ERR_INVALID_FORM_AND_CMD_NOT_BEGIN,
NULL,
                       w_id, ld_id, NULL );
    return HSE_STATUS_SUCCESS_AND_KEEP_CONN;
  }

  status = HSE_STATUS_SUCCESS_AND_KEEP_CONN;
  if( MATCHES_PROCESS( cmdptr ))
  {
    if( 'N' == cFormID )
      retcode = ProcessNewOrderQuery( pECB, ptr, w_id, ld_id );
    else if( 'P' == cFormID )
      retcode = ProcessPaymentQuery( pECB, ptr, w_id, ld_id );
    else if( 'D' == cFormID )
      retcode = ProcessDeliveryQuery( pECB, ptr, w_id, ld_id );
    else if( 'O' == cFormID )
      retcode = ProcessOrderStatusQuery( pECB, ptr, w_id, ld_id );
    else if( 'S' == cFormID )
      retcode = ProcessStockLevelQuery( pECB, ptr, w_id, ld_id );
    else {
      SendErrorResponse( pECB, ERR_INVALID_FORM, NULL, w_id, ld_id, NULL
);
      return HSE_STATUS_SUCCESS_AND_KEEP_CONN;
    }

    if( ERR_DB_PENDING == retcode )
      status = HSE_STATUS_PENDING;
    else if( ERR_DB_SUCCESS != retcode ) {
#pragma message ("FIXME: This is likely the second error report since the
Process*Query functions do their own error reporting!!  This is bad because we put
two error messages on the wire and this could confuse the next transaction.")
      SendErrorResponse( pECB, retcode, NULL, w_id, ld_id, NULL );
      return HSE_STATUS_SUCCESS_AND_KEEP_CONN;
    }
  }
  else if( MATCHES_PAYMENT( cmdptr ))
    SendPaymentForm( pECB, w_id, ld_id );
  else if( MATCHES_NEWORDER( cmdptr ))
    SendNewOrderForm( pECB, w_id, ld_id );
  else if( MATCHES_DELIVERY( cmdptr ))
    SendDeliveryForm( pECB, w_id, ld_id );
  else if( MATCHES_ORDERSTATUS( cmdptr ))
    SendOrderStatusForm( pECB, w_id, ld_id );
  else if( MATCHES_STOCKLEVEL( cmdptr ))
    SendStockLevelForm( pECB, w_id, ld_id );
  else if( MATCHES_EXIT( cmdptr ))
    ExitCmd( pECB, w_id, ld_id );
  else if( MATCHES_SUBMIT( cmdptr ))
    SubmitCmd( pECB, &w_id, &ld_id );
  else if( MATCHES_BEGIN( cmdptr ))
    BeginCmd( pECB );
  else if( MATCHES_MENU( cmdptr ))
    MenuCmd( pECB, w_id, ld_id );
  else if( MATCHES_CLEAR( cmdptr ))
    ClearCmd( pECB );
#pragma message ("FIXME: The ordering of the checkpoint commands is important,
because strncmp( 'CheckpointShutdown', 'Checkpoint', strlen( 'Checkpoint' )) is a
match.")
  else if( MATCHES_CHECKPOINT_STARTUP( cmdptr ))
    CheckpointStartupCmd( pECB, w_id, ld_id );
  else if( MATCHES_CHECKPOINT_SHUTDOWN( cmdptr ))
    CheckpointShutdownCmd( pECB, w_id, ld_id );
  else if( MATCHES_CHECKPOINT( cmdptr ))
    CheckpointCmd( pECB, w_id, ld_id );
#ifdef FFE_DEBUG
  else if( MATCHES_MEMORYCHECK( cmdptr ))
    MemoryCheckCmd( pECB, w_id, ld_id );
#endif
  else
    SendErrorResponse( pECB, ERR_COMMAND_UNDEFINED, NULL, w_id, ld_id,
NULL );

  return status;
}

/* FUNCTION: PutFloat2(double dVal, int iFieldSize, char *pChar )
 *
 * PURPOSE:              This function converts a double into a char string
 *                       in the format of xx.xx
 *
 * ARGUMENTS: double              dVal                    the value to convert to
char
 *            int         iFieldSize    max size of char string
 *            char        pChar                   string where to put value
 *
 * RETURNS:              void
 *
 * COMMENTS:             If the double exceeds the max field size entered,
 *                       the char string will be filled with iFieldSize *'s
```

```
 *                              to signal an error
 */

void
PutFloat2( double dVal, int iFieldSize, char *pChar )
{
  int iInt;
  int iDecimal;
  BOOL bSignFlag = TRUE;
  int iSaveSize = iFieldSize;
  char *pSaveStart = pChar;
  char pAsterisk[] = "**********************";

  pChar += (iFieldSize - 1);

  if(0 > dVal)
  {
    bSignFlag = FALSE;
    iInt = abs((int)( dVal * 100. ));
  }
  else
  {
    iInt = (int)( dVal * 100. );
  }
  iDecimal = 2;
  do
  {
    *pChar-- = ( iInt % 10 ) + '0';
    iInt /= 10;
    iFieldSize--;
  } while( --iDecimal );

  *pChar-- = '.';
  iFieldSize--;

  do
  {
    *pChar-- = ( iInt % 10 ) + '0';
    iInt /= 10;
    iFieldSize--;
  } while( iFieldSize && iInt != 0 );

  if( !iFieldSize && iInt != 0 )
  {
    /* put in string of ** to signal error */
    memcpy(pSaveStart, pAsterisk, iSaveSize);
    return;
  }
  if(!bSignFlag)
  {
    iFieldSize--;
    if( 0 >= iFieldSize )
    {
      /* put in string of  ** to signal error */
      memcpy(pSaveStart, pAsterisk, iSaveSize);
      return;
    }
    *pChar-- = '-';
  }

  /* Fill in the remaining spaces in the field with blanks. */
  while( iFieldSize-- )
    *pChar-- = ' ';
}
/* FUNCTION: void PutHTMLStrings( pPutStrStruct pStruct,
 *                                                      char *szInput, int
iInputSize,
 *                                                      char **szOutput, int
*iOutputSize )
 *
 * PURPOSE:  This routine takes a template output string and a data structure
 *           containing strings, positions, and field widths of strings to be
 *           compied into the template.  The routine scans all input strings to
 *           determine if any contain special charaters that need to be quoted
 *           in the output string.  If none exist, the template is filled with
 *           the desired strings.  If at least one special character exists in
 *           the output strings, a more expensive routine is called to build a
 *           new output string template containing the quoted strings.
 *
 * ARGUMENTS: pPutStrStruct    pStruct     pointer to structure containing the
 *                                                              strings, positions and
field lengths.
 *            char             *szInput    pointer to input form
 *            int              iInputSize  length of the input form
 *            char             **szOutput  pointer to the new input form
 *                                                              it may or
may not be different
 *                                                              than the
input form.
 *            int              iOutputSize length of the new input form.
 *
 * RETURNS:   none
 *
 * COMMENTS:          none
 */

void
PutHTMLStrings( pPutStrStruct pStruct,
                        char *szInput, int iInputSize,
                        char **szOutput, int *iOutputSize )
{
  char *pIChar;
  char *pOChar;
  int          iFieldSize;

  while( NULL != pStruct->szStr )
  {
    pIChar = pStruct->szStr;
    pOChar = szInput + pStruct->iIndex;
    iFieldSize = pStruct->iFieldSize;
    while( 0 != *pIChar && iFieldSize )
    {
      /* '>' is the highest ACSII value of the special characters. */
      /* If '>' is greater than the character is question, check further. */
      if( '>' > *pIChar )
      {
              if( '"' == *pIChar || '&' == *pIChar ||
                  '<' == *pIChar || '>' == *pIChar )
              {
                /* We have found at least one special character in the desired */
                /* output string, go the the more expensive routine to build */
                /* the desired output string. */
                InterlockedIncrement( &giPanic );
                HandlePanic( pStruct, szInput, iInputSize, szOutput, iOutputSize );
                return;
              }
              else
                *pOChar = *pIChar;
      }
      else
              *pOChar = *pIChar;

      pIChar++;
      pOChar++;
      iFieldSize--;
    }

    /* Fill in the remaining spaces in the field with blanks. */
    while( iFieldSize-- )
      *pOChar++ = ' ';

    pStruct++;
  }

  /* The output string is the template and the length is unchanged */
  *szOutput = szInput;
  *iOutputSize = iInputSize;

  return;
}

/* FUNCTION: int TPCCDeliveryResponse(  EXTENSION_CONTROL_BLOCK
*pECB,
 *                                                      int retcode,
 *                                                      DeliveryData
*deliveryData )
 *
 * PURPOSE:           This function fills in the values and returns the
 *                    response form to the browser.
 *
```

```
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK *pECB
 *                  int                     retcode  return code
from db
 *                  DeliveryData            *deliveryData pointer
to the delivery
 *                                                   data
structure.
 *
 * RETURNS:         none
 *
 * COMMENTS:        none
 */

int
TPCCDeliveryResponse( int retcode, pDeliveryData pDelivery,
                      pDeliveryData
pCompletedDeliveries[DELIVERY_RESPONSE_COUNT] )
{
  int ssCnt = 0;
  char *szOutput;
  int iOutputLen;
  PutStrStruct StrStruct[2];
  char *deliveryForm;
  int ii, jj, index;
  pDeliveryData pCompletedDelivery;
  static DeliveryData blankDelivery = { 0 };
  EXTENSION_CONTROL_BLOCK *pECB;

  pECB = (EXTENSION_CONTROL_BLOCK *)pDelivery->CC;

  if ( ERR_DB_PENDING == retcode )
  {
   return( retcode );
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {
    SendErrorResponse( pECB, ERR_DELIVERY_NOT_PROCESSED, NULL,
                       pDelivery->w_id, pDelivery->ld_id,
                       (pConnData)pDelivery );
    TRANSACTION_DEBUG_STAGE( pDelivery, UNRESERVING );
    pUnreserveTransactionStruct( DELIVERY_TRANS, &pDelivery );
    for( ii = 0; ii < DELIVERY_RESPONSE_COUNT; ii++ )
    {
      if( NULL != pCompletedDeliveries[ii] )
      {
                TRANSACTION_DEBUG_STAGE( pCompletedDeliveries[ii],
UNRESERVING );
                pUnreserveTransactionStruct( DELIVERY_TRANS,
&pCompletedDeliveries[ii]);
      }
    }
  }
  else          if ( ERR_DB_SUCCESS != retcode )
  {
    SendErrorResponse( pECB, ERR_DB_DELIVERY_NOT_QUEUED, NULL,
                       pDelivery->w_id, pDelivery->ld_id,
                       (pConnData)pDelivery );
    TRANSACTION_DEBUG_STAGE( pDelivery, UNRESERVING );
    pUnreserveTransactionStruct( DELIVERY_TRANS, &pDelivery );
    for( ii = 0; ii < DELIVERY_RESPONSE_COUNT; ii++ )
    {
      if( NULL != pCompletedDeliveries[ii] )
      {
                TRANSACTION_DEBUG_STAGE( pCompletedDeliveries[ii],
UNRESERVING );
                pUnreserveTransactionStruct( DELIVERY_TRANS,
&pCompletedDeliveries[ii]);
      }
    }
  }
  else
  {
    RESERVE_RESPONSE( DELIVERY_RESPONSE, deliveryForm );

    PutNumeric(WDID(pDelivery->w_id,pDelivery->ld_id),
                deliveryFormIndexesP[D_WDID].iLen,
                &deliveryForm[deliveryFormIndexesP[D_WDID].iStartIndex]);
    PutNumeric(pDelivery->w_id,
                deliveryFormIndexesP[D_WID].iLen,
                &deliveryForm[deliveryFormIndexesP[D_WID].iStartIndex]);
    PutNumeric(pDelivery->o_carrier_id,
                deliveryFormIndexesP[D_CAR].iLen,
                &deliveryForm[deliveryFormIndexesP[D_CAR].iStartIndex]);
    TRANSACTION_DEBUG_STAGE( pDelivery, UNRESERVING );
    pUnreserveTransactionStruct( DELIVERY_TRANS, &pDelivery );

    index = D_QUEUE1;
    for( ii = 0; ii < DELIVERY_RESPONSE_COUNT; ii++ ) {
     if( NULL == pCompletedDeliveries[ii] )
                pCompletedDelivery = &blankDelivery;
     else
                pCompletedDelivery = pCompletedDeliveries[ii];
     PutNumeric(pCompletedDelivery->queue_time,
                        deliveryFormIndexesP[index].iLen,
                        &deliveryForm[deliveryFormIndexesP[index].iStartIndex]);
     index++;
     PutNumeric(pCompletedDelivery->delta_time,
                        deliveryFormIndexesP[index].iLen,
                        &deliveryForm[deliveryFormIndexesP[index].iStartIndex]);
     index++;
     PutNumeric(pCompletedDelivery->w_id,
                        deliveryFormIndexesP[index].iLen,
                        &deliveryForm[deliveryFormIndexesP[index].iStartIndex]);
     index++;
     PutNumeric(pCompletedDelivery->o_carrier_id,
                        deliveryFormIndexesP[index].iLen,
                        &deliveryForm[deliveryFormIndexesP[index].iStartIndex]);
     index++;
     for( jj = 0; jj < DISTRICTS_PER_WAREHOUSE; jj++ ) {
                PutNumeric(pCompletedDelivery->o_id[jj],
                          deliveryFormIndexesP[index].iLen,

&deliveryForm[deliveryFormIndexesP[index].iStartIndex]);
                index++;
     }
     if( NULL != pCompletedDeliveries[ii] ) {
                TRANSACTION_DEBUG_STAGE( pCompletedDeliveries[ii],
UNRESERVING );
                pUnreserveTransactionStruct( DELIVERY_TRANS,
&pCompletedDeliveries[ii]);
     }
    }

    PUT_STRING(NULL, 0, 0, StrStruct[ssCnt]);
    PutHTMLStrings(StrStruct, deliveryForm,
giResponseLen[DELIVERY_RESPONSE],
                          &szOutput, &iOutputLen);

    SendResponse(pECB, szOutput, iOutputLen);

    UNRESERVE_RESPONSE( DELIVERY_RESPONSE, deliveryForm );

    if( szOutput != deliveryForm )
     UNRESERVE_PANIC_FORM( szOutput );
  }

  return( retcode );
}

/* FUNCTION: int TPCCNewOrderResponse(EXTENSION_CONTROL_BLOCK
*pECB,
 *                                                   int retcode,
 *                                                   NewOrderData
*newOrderData )
 *
 * PURPOSE:         This function fills in the values and returns the
 *                  response form to the browser.
 *
 * ARGUMENTS:       EXTENSION_CONTROL_BLOCK  *pECB       pointer to
the structure
 *                                                      that
contains the internet
 *                                                      service
information.
 *                  int                      retcode     return status from the
db.
 *                  NewOrderData             *newOrderData       pointer to
structure containing
 *                                                      data
about the current txn.
```

```
 *
 * RETURNS:              none
 *
 * COMMENTS:             none
 */

int
TPCCNewOrderResponse( int retcode, pNewOrderData pNewOrder )
{
  int             i;
  char  szDate[]   = "xx-xx-xxxx xx:xx:xx";
  char  szBlanks[] = "                    ";
  char  szDollar[] = "$";
  PutStrStruct StrStruct[133];
  int ssCnt = 0;
  int jj;
  int kk;
  int mm;
  char *newOrderForm;
  char *szOutput;
  int iOutputLen;
  BOOL bValid;
  char *execution_status;
  EXTENSION_CONTROL_BLOCK *pECB;

  pECB = (EXTENSION_CONTROL_BLOCK *)pNewOrder->CC;

  if ( ERR_DB_PENDING == retcode )
  {
   return( retcode );
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {
   SendErrorResponse( pECB, ERR_NEW_ORDER_NOT_PROCESSED, NULL,
                      pNewOrder->w_id, pNewOrder->ld_id,
                      (pConnData)pNewOrder );
   TRANSACTION_DEBUG_STAGE( pNewOrder, UNRESERVING );
   pUnreserveTransactionStruct( NEW_ORDER_TRANS, &pNewOrder );
   return( retcode );
  }
  else if( ERR_DB_SUCCESS != retcode && ERR_DB_NOT_COMMITED !=
retcode )
  {
   SendErrorResponse( pECB, retcode, NULL,
                      pNewOrder->w_id, pNewOrder->ld_id,
                      (pConnData)pNewOrder );
   TRANSACTION_DEBUG_STAGE( pNewOrder, UNRESERVING );
   pUnreserveTransactionStruct( NEW_ORDER_TRANS, &pNewOrder );
   return( retcode );
  }
  else if ( ERR_DB_SUCCESS == retcode )
  {
   bValid = TRUE;
   execution_status = TRANSACTION_COMMITED;
  }
  else if ( ERR_DB_NOT_COMMITED == retcode )
  {
   /* This is the expected error code for a transaction specifying an */
   /* item id that is in the valid range, but not assigned to an actual */
   /* item.  This is how the specification and therefore our test harness */
   /* validates the database is prepared to handle rollbacks.  So that we */
   /* do not return a valid page indicating the request specified an */
   /* invalid item and then further along in the execution, return an */
   /* error page that would incorrectly satisfy the next transaction, */
   /* we need to clear the error code. */
   retcode = ERR_DB_SUCCESS;
   bValid = FALSE;
   execution_status = ITEM_NUMBER_INVALID;
  }

  RESERVE_RESPONSE( NEW_ORDER_RESPONSE, newOrderForm );

  PutNumeric(WDID(pNewOrder->w_id,pNewOrder->ld_id),
                  newOrderResponseIndexes[NO_WDID].iLen,

&newOrderForm[newOrderResponseIndexes[NO_WDID].iStartIndex]);
  PutNumeric(pNewOrder->w_id,
                  newOrderResponseIndexes[NO_WID].iLen,
                  &newOrderForm[newOrderResponseIndexes[NO_WID].iStartIndex]);
  PutNumeric(pNewOrder->d_id,
                  newOrderResponseIndexes[NO_DID].iLen,
                  &newOrderForm[newOrderResponseIndexes[NO_DID].iStartIndex]);

  if(bValid)
  {
   /* put the date in if valid */
   PutNumeric(pNewOrder->o_entry_d.day, 2, &szDate[0]);
   PutNumeric(pNewOrder->o_entry_d.month, 2, &szDate[3]);
   PutNumeric(pNewOrder->o_entry_d.year, 4, &szDate[6]);
   PutNumeric(pNewOrder->o_entry_d.hour, 2, &szDate[11]);
   PutNumeric(pNewOrder->o_entry_d.minute, 2, &szDate[14]);
   PutNumeric(pNewOrder->o_entry_d.second, 2, &szDate[17]);
   memcpy(&newOrderForm[newOrderResponseIndexes[NO_DATE].iStartIndex],
               szDate, newOrderResponseIndexes[NO_DATE].iLen);
  }
  else
  {
   /* put in blanks for the date if not valid */
   memcpy(&newOrderForm[newOrderResponseIndexes[NO_DATE].iStartIndex],
               szBlanks, newOrderResponseIndexes[NO_DATE].iLen);
  }
  /* put in value for the customer id. */
  PutNumeric(pNewOrder->c_id,
                  newOrderResponseIndexes[NO_CID].iLen,
                  &newOrderForm[newOrderResponseIndexes[NO_CID].iStartIndex]);

  /* put in the values for the last name and  credit rating */
  PUT_STRING(pNewOrder->c_last,
                  newOrderResponseIndexes[NO_LAST].iLen,
                  newOrderResponseIndexes[NO_LAST].iStartIndex,
                  StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pNewOrder->c_credit,
                  newOrderResponseIndexes[NO_CREDIT].iLen,
                  newOrderResponseIndexes[NO_CREDIT].iStartIndex,
                  StrStruct[ssCnt]);
  ssCnt++;

  if(bValid)
  {
   /* put in the values */
   PutFloat2(pNewOrder->c_discount,
                   newOrderResponseIndexes[NO_DISC].iLen,

&newOrderForm[newOrderResponseIndexes[NO_DISC].iStartIndex]);
  }
  else
  {
   /* If the transaction is rolled back, we need to blank out fields. */
   /* spaces for discount */
   memcpy(&newOrderForm[newOrderResponseIndexes[NO_DISC].iStartIndex],
               szBlanks, newOrderResponseIndexes[NO_DISC].iLen);
  }

  /* Per 2.4.3.4 of the specification, provide the order id regardless of */
  /* whether the transaction was committed or rolled back. */
  PutNumeric(pNewOrder->o_id,
                  newOrderResponseIndexes[NO_OID].iLen,
                  &newOrderForm[newOrderResponseIndexes[NO_OID].iStartIndex]);

  if(bValid)
  {
   PutNumeric(pNewOrder->o_ol_cnt,
                   newOrderResponseIndexes[NO_LINES].iLen,

&newOrderForm[newOrderResponseIndexes[NO_LINES].iStartIndex]);
   PutFloat2(pNewOrder->w_tax,
                   newOrderResponseIndexes[NO_W_TAX].iLen,

&newOrderForm[newOrderResponseIndexes[NO_W_TAX].iStartIndex]);
   PutFloat2(pNewOrder->d_tax,
                   newOrderResponseIndexes[NO_D_TAX].iLen,

&newOrderForm[newOrderResponseIndexes[NO_D_TAX].iStartIndex]);


   for(i=0; i<pNewOrder->o_ol_cnt; i++)
   {
    PutNumeric(pNewOrder->o_ol[i].ol_supply_w_id,
                    newOrderResponseIndexes[NO_S_WID+(i*8)].iLen,
```

```
&newOrderForm[newOrderResponseIndexes[NO_S_WID+(i*8)].iStartIndex]);
    PutNumeric(pNewOrder->o_ol[i].ol_i_id,
                        newOrderResponseIndexes[NO_IID+(i*8)].iLen,

&newOrderForm[newOrderResponseIndexes[NO_IID+(i*8)].iStartIndex]);
    PUT_STRING(pNewOrder->o_ol[i].i_name,
                        newOrderResponseIndexes[NO_INAME+(i*8)].iLen,
                        newOrderResponseIndexes[NO_INAME+(i*8)].iStartIndex,
                        StrStruct[ssCnt]);
    ssCnt++;
    PutNumeric(pNewOrder->o_ol[i].ol_quantity,
                        newOrderResponseIndexes[NO_QTY+(i*8)].iLen,

&newOrderForm[newOrderResponseIndexes[NO_QTY+(i*8)].iStartIndex]);

    PutNumeric(pNewOrder->o_ol[i].s_quantity,
                        newOrderResponseIndexes[NO_STOCK+(i*8)].iLen,

&newOrderForm[newOrderResponseIndexes[NO_STOCK+(i*8)].iStartIndex]);
    PUT_STRING(pNewOrder->o_ol[i].b_g,
                        newOrderResponseIndexes[NO_BG+(i*8)].iLen,
                        newOrderResponseIndexes[NO_BG+(i*8)].iStartIndex,
                        StrStruct[ssCnt]);
    ssCnt++;

memcpy(&newOrderForm[newOrderResponseIndexes[NO_PRICE+(i*8)].iStartIndex
-1],
                szDollar, 1);
    PutFloat2(pNewOrder->o_ol[i].i_price,
                        newOrderResponseIndexes[NO_PRICE+(i*8)].iLen,

            &newOrderForm[newOrderResponseIndexes[NO_PRICE+(i*8)].iStartIn
dex]);

memcpy(&newOrderForm[newOrderResponseIndexes[NO_AMT+(i*8)].iStartIndex-
1],
                szDollar, 1);
    PutFloat2(pNewOrder->o_ol[i].ol_amount,
                        newOrderResponseIndexes[NO_AMT+(i*8)].iLen,

            &newOrderForm[newOrderResponseIndexes[NO_AMT+(i*8)].iStartInde
x]);

   }
   /* need to blank out the rest of the unused item rows */
   jj = NO_AMT + ((i-1)*8) + 1;
   for(kk=i; kk<MAX_OL; kk++)
   {
    /* there are 8 items per row - 6 plain and 2 with $*/
    for(mm=0; mm<6; mm++)
    {
            memcpy(&newOrderForm[newOrderResponseIndexes[jj].iStartIndex],
                szBlanks, newOrderResponseIndexes[jj].iLen);
            jj++;
    }
    /* blank out the '$' for the blank $values */
    for(mm=0; mm<2; mm++)
    {
            memcpy(&newOrderForm[newOrderResponseIndexes[jj].iStartIndex-1],
                szBlanks, newOrderResponseIndexes[jj].iLen+1);
            jj++;
    }
   }
  }
  else
  {
   /* spaces in the o_ol_cnt, w_tax, and d_tax fields */
   for(kk=0; kk<3; kk++)
   {

memcpy(&newOrderForm[newOrderResponseIndexes[NO_LINES+kk].iStartIndex],
                szBlanks, newOrderResponseIndexes[NO_LINES+kk].iLen);
   }
   /* spaces for each of the fields in the row items */
   jj = NO_S_WID;
   for(kk=0; kk<MAX_OL; kk++)
   {
    /* there are 8 items per row - 6 plain and 2 with $*/
    for(mm=0; mm<6; mm++)
    {
            memcpy(&newOrderForm[newOrderResponseIndexes[jj].iStartIndex],
                szBlanks, newOrderResponseIndexes[jj].iLen);
            jj++;
    }
    /* blank out the '$' for the blank $values */
    for(mm=0; mm<2; mm++)
    {
            memcpy(&newOrderForm[newOrderResponseIndexes[jj].iStartIndex-1],
                szBlanks, newOrderResponseIndexes[jj].iLen+1);
            jj++;
    }

   }
  }

  /* output the execution status */
  PUT_STRING(execution_status, newOrderResponseIndexes[NO_STAT].iLen,
                newOrderResponseIndexes[NO_STAT].iStartIndex,
                StrStruct[ssCnt]);
  ssCnt++;

  if(bValid)
  {
   /* total */
   PutFloat2(pNewOrder->total_amount,
                newOrderResponseIndexes[NO_TOTAL].iLen,

&newOrderForm[newOrderResponseIndexes[NO_TOTAL].iStartIndex]);
  }
  else
  {
   /* put blanks for total */
   memcpy(&newOrderForm[newOrderResponseIndexes[NO_TOTAL].iStartIndex],
                szBlanks, newOrderResponseIndexes[NO_TOTAL].iLen);
  }
  PUT_STRING(NULL, 0, 0, StrStruct[ssCnt]);
  PutHTMLStrings(StrStruct, newOrderForm,
giResponseLen[NEW_ORDER_RESPONSE],
                        &szOutput, &iOutputLen);

  TRANSACTION_DEBUG_STAGE( pNewOrder, UNRESERVING );

  pUnreserveTransactionStruct( NEW_ORDER_TRANS, &pNewOrder );

  SendResponse(pECB, szOutput, iOutputLen);

  UNRESERVE_RESPONSE( NEW_ORDER_RESPONSE, newOrderForm );

  if( szOutput != newOrderForm )
   UNRESERVE_PANIC_FORM( szOutput );

  return( retcode );
}

/* FUNCTION: int TPCCPaymentResponse(EXTENSION_CONTROL_BLOCK
*pECB,
 *                                                      int retcode,
 *                                                      PaymentData
*paymentData)
 *
 * PURPOSE:             This function fills in the values and returns the
 *                      response form to the browser.
 *
 * ARGUMENTS:           EXTENSION_CONTROL_BLOCK  *pECB          pointer to
structure that
 *                                                              contains
internet service
 *
                information.
 *                      int                     retcode         return status from the
db call
 *                      PaymentData             *paymentData            pointer to
structure containing
 *                                                              the data
for this transaction.
 *
 * RETURNS:             none
 *
 * COMMENTS:            none
```

```
 */

int
TPCCPaymentResponse( int retcode, pPaymentData pPayment )
{
  char          *ptr;
  char          szcdata[4][64];
  char          szW_Zip[26];
  char          szD_Zip[26];
  char          szC_Zip[26];
  char          szC_Phone[26];
  int           i;
  int           l;
  char          *szZipPic = "XXXXX-XXXX";
  char  szLongDate[] = "XX-XX-XXXX XX:XX:XX";
  char  szDate[]   = "xx-xx-xxxx";
  char  szBlanks[] = "                                                  ";
  PutStrStruct StrStruct[34];
  int ssCnt = 0;
  char *paymentForm;
  char *szOutput;
  int iOutputLen;
  EXTENSION_CONTROL_BLOCK *pECB;

  pECB = (EXTENSION_CONTROL_BLOCK *)pPayment->CC;

  if ( ERR_DB_PENDING == retcode )
  {
   return( retcode );
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {
   SendErrorResponse( pECB, ERR_PAYMENT_NOT_PROCESSED, NULL,
                      pPayment->w_id, pPayment->ld_id,
                      (pConnData)pPayment );
   TRANSACTION_DEBUG_STAGE( pPayment, UNRESERVING );
   pUnreserveTransactionStruct( PAYMENT_TRANS, &pPayment );
   return( retcode );
  }
  else if ( ERR_DB_NOT_COMMITED == retcode )
  {
   SendErrorResponse( pECB, ERR_PAYMENT_INVALID_CUSTOMER, NULL,
                      pPayment->w_id, pPayment->ld_id,
                      (pConnData)pPayment );
   TRANSACTION_DEBUG_STAGE( pPayment, UNRESERVING );
   pUnreserveTransactionStruct( PAYMENT_TRANS, &pPayment );
   return( retcode );
  }
  else if ( ERR_DB_SUCCESS != retcode )
  {
   SendErrorResponse( pECB, retcode, NULL,
                      pPayment->w_id, pPayment->ld_id,
                      (pConnData)pPayment );
   TRANSACTION_DEBUG_STAGE( pPayment, UNRESERVING );
   pUnreserveTransactionStruct( PAYMENT_TRANS, &pPayment );
   return( retcode );
  }

  RESERVE_RESPONSE( PAYMENT_RESPONSE,  paymentForm );

  PutNumeric(WDID(pPayment->w_id,pPayment->ld_id),
             paymentResponseIndexes[PT_WDID].iLen,
             &paymentForm[paymentResponseIndexes[PT_WDID].iStartIndex]);
  PutNumeric(pPayment->h_date.day, 2,
             &szLongDate[0]);
  PutNumeric(pPayment->h_date.month, 2,
             &szLongDate[3]);
  PutNumeric(pPayment->h_date.year, 4,
             &szLongDate[6]);
  PutNumeric(pPayment->h_date.hour, 2,
             &szLongDate[11]);
  PutNumeric(pPayment->h_date.minute, 2,
             &szLongDate[14]);
  PutNumeric(pPayment->h_date.second, 2,
             &szLongDate[17]);
  memcpy(&paymentForm[paymentResponseIndexes[PT_LONG_DATE].iStartIndex],
         szLongDate, paymentResponseIndexes[PT_LONG_DATE].iLen);

  PutNumeric(pPayment->w_id,
             paymentResponseIndexes[PT_WID].iLen,
             &paymentForm[paymentResponseIndexes[PT_WID].iStartIndex]);
  PutNumeric(pPayment->d_id,
             paymentResponseIndexes[PT_DID].iLen,
             &paymentForm[paymentResponseIndexes[PT_DID].iStartIndex]);

  PUT_STRING(pPayment->w_street_1,
             paymentResponseIndexes[PT_W_ST_1].iLen,
             paymentResponseIndexes[PT_W_ST_1].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pPayment->d_street_1,
             paymentResponseIndexes[PT_D_ST_1].iLen,
             paymentResponseIndexes[PT_D_ST_1].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pPayment->w_street_2,
             paymentResponseIndexes[PT_W_ST_2].iLen,
             paymentResponseIndexes[PT_W_ST_2].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pPayment->d_street_2,
             paymentResponseIndexes[PT_D_ST_2].iLen,
             paymentResponseIndexes[PT_D_ST_2].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pPayment->w_city,
             paymentResponseIndexes[PT_W_CITY].iLen,
             paymentResponseIndexes[PT_W_CITY].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pPayment->w_state,
             paymentResponseIndexes[PT_W_ST].iLen,
             paymentResponseIndexes[PT_W_ST].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  FormatString(szW_Zip, szZipPic, pPayment->w_zip);
  memcpy(&paymentForm[paymentResponseIndexes[PT_W_ZIP].iStartIndex],
         szW_Zip, paymentResponseIndexes[PT_W_ZIP].iLen);
  PUT_STRING(pPayment->d_city,
             paymentResponseIndexes[PT_D_CITY].iLen,
             paymentResponseIndexes[PT_D_CITY].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pPayment->d_state,
             paymentResponseIndexes[PT_D_ST].iLen,
             paymentResponseIndexes[PT_D_ST].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  FormatString(szD_Zip, szZipPic, pPayment->d_zip);
  memcpy(&paymentForm[paymentResponseIndexes[PT_D_ZIP].iStartIndex],
         szD_Zip, paymentResponseIndexes[PT_D_ZIP].iLen);
  PutNumeric(pPayment->c_id,
             paymentResponseIndexes[PT_CID].iLen,
             &paymentForm[paymentResponseIndexes[PT_CID].iStartIndex]);
  PutNumeric(pPayment->c_w_id,
             paymentResponseIndexes[PT_C_WID].iLen,
             &paymentForm[paymentResponseIndexes[PT_C_WID].iStartIndex]);
  PutNumeric(pPayment->c_d_id,
             paymentResponseIndexes[PT_C_DID].iLen,
             &paymentForm[paymentResponseIndexes[PT_C_DID].iStartIndex]);

  PUT_STRING(pPayment->c_first,
             paymentResponseIndexes[PT_FIRST].iLen,
             paymentResponseIndexes[PT_FIRST].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pPayment->c_middle,
             paymentResponseIndexes[PT_MIDDLE].iLen,
             paymentResponseIndexes[PT_MIDDLE].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pPayment->c_last,
             paymentResponseIndexes[PT_LAST].iLen,
             paymentResponseIndexes[PT_LAST].iStartIndex,
             StrStruct[ssCnt]);
  ssCnt++;

  PutNumeric(pPayment->c_since.day, 2, &szDate[0]);
  PutNumeric(pPayment->c_since.month, 2, &szDate[3]);
  PutNumeric(pPayment->c_since.year, 4, &szDate[6]);
```

```
  memcpy(&paymentForm[paymentResponseIndexes[PT_SM_DATE].iStartIndex],
szDate,
                paymentResponseIndexes[PT_SM_DATE].iLen);

  PUT_STRING(pPayment->c_street_1,
                    paymentResponseIndexes[PT_C_STR_1].iLen,
                    paymentResponseIndexes[PT_C_STR_1].iStartIndex,
                    StrStruct[ssCnt]);
  ssCnt++;
  PUT_STRING(pPayment->c_credit,
                    paymentResponseIndexes[PT_CREDIT].iLen,
                    paymentResponseIndexes[PT_CREDIT].iStartIndex,
                    StrStruct[ssCnt]);
  ssCnt++;

  PUT_STRING(pPayment->c_street_2,
                    paymentResponseIndexes[PT_C_STR_2].iLen,
                    paymentResponseIndexes[PT_C_STR_2].iStartIndex,
                    StrStruct[ssCnt]);
  ssCnt++;

  PutFloat2(pPayment->c_discount,
                   paymentResponseIndexes[PT_DISC].iLen,
                   &paymentForm[paymentResponseIndexes[PT_DISC].iStartIndex]);

  PUT_STRING(pPayment->c_city,
                    paymentResponseIndexes[PT_C_CITY].iLen,
                    paymentResponseIndexes[PT_C_CITY].iStartIndex,
                    StrStruct[ssCnt]);
  ssCnt++;

  PUT_STRING(pPayment->c_state,
                    paymentResponseIndexes[PT_C_ST].iLen,
                    paymentResponseIndexes[PT_C_ST].iStartIndex,
                    StrStruct[ssCnt]);
  ssCnt++;

  FormatString(szC_Zip, szZipPic, pPayment->c_zip);
  memcpy(&paymentForm[paymentResponseIndexes[PT_C_ZIP].iStartIndex],
szC_Zip,
                paymentResponseIndexes[PT_C_ZIP].iLen);
  FormatString(szC_Phone, "XXXXXX-XXX-XXX-XXXX",
                     pPayment->c_phone);
  memcpy(&paymentForm[paymentResponseIndexes[PT_C_PHONE].iStartIndex],
                szC_Phone, paymentResponseIndexes[PT_C_PHONE].iLen);

  PutFloat2(pPayment->h_amount,
                    paymentResponseIndexes[PT_AMT].iLen,
                    &paymentForm[paymentResponseIndexes[PT_AMT].iStartIndex]);
  PutFloat2(pPayment->c_balance,
                    paymentResponseIndexes[PT_BAL].iLen,
                    &paymentForm[paymentResponseIndexes[PT_BAL].iStartIndex]);

  PutFloat2(pPayment->c_credit_lim,
                    paymentResponseIndexes[PT_LIM].iLen,
                    &paymentForm[paymentResponseIndexes[PT_LIM].iStartIndex]);

  ptr = pPayment->c_credit;
  if ( *ptr == 'B' && *(ptr+1) == 'C' )
  {
    ptr = pPayment->c_data;
    l = strlen( ptr ) / 50;
    for(i=0; i<4; i++, ptr += 50)
    {
      if ( i <= l )
      {
                strncpy(szcdata[i], ptr, 50);
                szcdata[i][50] = '\0';
      }
      else
                szcdata[i][0] = 0;

      PUT_STRING(szcdata[i],
                              paymentResponseIndexes[PT_CUST_DATA+i].iLen,
                              paymentResponseIndexes[PT_CUST_DATA+i].iStartIndex,
                              StrStruct[ssCnt]);
      ssCnt++;
    }
  }
  else
  {
    for(i=0; i<4; i++)
    {

memcpy(&paymentForm[paymentResponseIndexes[PT_CUST_DATA+i].iStartIndex]
,
                        szBlanks, paymentResponseIndexes[PT_CUST_DATA+i].iLen);
    }
  }

  PUT_STRING(NULL, 0, 0, StrStruct[ssCnt]);

  PutHTMLStrings(StrStruct, paymentForm,
giResponseLen[PAYMENT_RESPONSE],
                              &szOutput, &iOutputLen);

  TRANSACTION_DEBUG_STAGE( pPayment, UNRESERVING );
  pUnreserveTransactionStruct( PAYMENT_TRANS, &pPayment );

  SendResponse(pECB, szOutput, iOutputLen);

  UNRESERVE_RESPONSE( PAYMENT_RESPONSE, paymentForm );

  if( szOutput != paymentForm )
    UNRESERVE_PANIC_FORM( szOutput );

  return( retcode );
}

/* FUNCTION: int TPCCOrderStatusResponse( int retcode,
 *                                                              OrderStatusData
*orderStatusData)
 *
 * PURPOSE:             This function fills in the values and returns the
 *                      response form to the browser.
 *
 * ARGUMENTS:           EXTENSION_CONTROL_BLOCK   *pECB        pointer to
structure containing
 *                                                                      internet
service information.
 *                      int                             retcode return status from db call
 *                      OrderStatusData *orderStatusData      pointer to structure
 *
                of data for this txn.
 *
 * RETURNS:             none
 *
 * COMMENTS:            none
 */

int
TPCCOrderStatusResponse( int retcode, pOrderStatusData pOrderStatus )
{
  int           i;
  int           jj;
  int           kk;
  int           mm;
  char  szLongDate[] = "XX-XX-XXXX XX:XX:XX";
  char  szDate[]   = "XX-XX-XXXX";
  char  szBlanks[] = "                    ";
  char  szDollar[] = "$";
  PutStrStruct StrStruct[4];
  int   ssCnt = 0;
  char          *orderStatusForm;
  char          *szOutput;
  int           iOutputLen;
  EXTENSION_CONTROL_BLOCK *pECB;

  pECB = (EXTENSION_CONTROL_BLOCK *)pOrderStatus->CC;

  if ( ERR_DB_PENDING == retcode )
  {
    return( retcode );
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {
    SendErrorResponse( pECB, ERR_ORDER_STATUS_NOT_PROCESSED, NULL,
                                pOrderStatus->w_id, pOrderStatus->ld_id,
                                (pConnData)pOrderStatus );
    TRANSACTION_DEBUG_STAGE( pOrderStatus, UNRESERVING );
```

```
  pUnreserveTransactionStruct( ORDER_STATUS_TRANS, &pOrderStatus );
  return( retcode );
 }
 else if ( ERR_DB_NOT_COMMITED == retcode )
 {
  SendErrorResponse( pECB, ERR_NOSUCH_CUSTOMER, NULL,
                     pOrderStatus->w_id, pOrderStatus->ld_id,
                     (pConnData)pOrderStatus );
  TRANSACTION_DEBUG_STAGE( pOrderStatus, UNRESERVING );
  pUnreserveTransactionStruct( ORDER_STATUS_TRANS, &pOrderStatus );
  return( retcode );
 }
 else if ( ERR_DB_SUCCESS != retcode )
 {
  SendErrorResponse( pECB, retcode, NULL,
                     pOrderStatus->w_id, pOrderStatus->ld_id,
                     (pConnData)pOrderStatus );
  TRANSACTION_DEBUG_STAGE( pOrderStatus, UNRESERVING );
  pUnreserveTransactionStruct( ORDER_STATUS_TRANS, &pOrderStatus );
  return( retcode );
 }

 RESERVE_RESPONSE( ORDER_STATUS_RESPONSE, orderStatusForm );

 PutNumeric(WDID(pOrderStatus->w_id,pOrderStatus->ld_id),
            orderStatusResponseIndexes[OS_WDID].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_WDID].iStartIndex]);
 PutNumeric(pOrderStatus->w_id,
            orderStatusResponseIndexes[OS_WID].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_WID].iStartIndex]);
 PutNumeric(pOrderStatus->d_id,
            orderStatusResponseIndexes[OS_DID].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_DID].iStartIndex]);
 PutNumeric(pOrderStatus->c_id,
            orderStatusResponseIndexes[OS_CID].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_CID].iStartIndex]);
 PUT_STRING(pOrderStatus->c_first,
            orderStatusResponseIndexes[OS_FIRST].iLen,
            orderStatusResponseIndexes[OS_FIRST].iStartIndex,
StrStruct[ssCnt]);
 ssCnt++;
 PUT_STRING(pOrderStatus->c_middle,
            orderStatusResponseIndexes[OS_MIDDLE].iLen,
            orderStatusResponseIndexes[OS_MIDDLE].iStartIndex,
            StrStruct[ssCnt]);
 ssCnt++;
 PUT_STRING(pOrderStatus->c_last,
            orderStatusResponseIndexes[OS_LAST].iLen,
            orderStatusResponseIndexes[OS_LAST].iStartIndex,
StrStruct[ssCnt]);
 ssCnt++;
 PutFloat2(pOrderStatus->c_balance,
           orderStatusResponseIndexes[OS_BAL].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_BAL].iStartIndex]);
 PutNumeric(pOrderStatus->o_id,
            orderStatusResponseIndexes[OS_OID].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_OID].iStartIndex]);

 PutNumeric(pOrderStatus->o_entry_d.day, 2, &szLongDate[0]);
 PutNumeric(pOrderStatus->o_entry_d.month, 2, &szLongDate[3]);
 PutNumeric(pOrderStatus->o_entry_d.year, 4, &szLongDate[6]);
 PutNumeric(pOrderStatus->o_entry_d.hour, 2, &szLongDate[11]);
 PutNumeric(pOrderStatus->o_entry_d.minute, 2, &szLongDate[14]);
 PutNumeric(pOrderStatus->o_entry_d.second, 2, &szLongDate[17]);
 memcpy(&orderStatusForm[orderStatusResponseIndexes[OS_DATE].iStartIndex],
        szLongDate, orderStatusResponseIndexes[OS_DATE].iLen);
 PutNumeric(pOrderStatus->o_carrier_id,
            orderStatusResponseIndexes[OS_CAR_ID].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_CAR_ID].iStartIndex]);

 for(i=0; i<pOrderStatus->o_ol_cnt; i++)
 {
  PutNumeric(pOrderStatus->s_ol[i].ol_supply_w_id,
             orderStatusResponseIndexes[OS_S_WID+(i*5)].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_S_WID+(i*5)].iStartIndex]);
  PutNumeric(pOrderStatus->s_ol[i].ol_i_id,
             orderStatusResponseIndexes[OS_IID+(i*5)].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_IID+(i*5)].iStartIndex]);
  PutNumeric(pOrderStatus->s_ol[i].ol_quantity,
             orderStatusResponseIndexes[OS_QTY+(i*5)].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_QTY+(i*5)].iStartIndex]);

memcpy(&orderStatusForm[orderStatusResponseIndexes[OS_AMT+(i*5)].iStartInde
x-1],
                szDollar, 1);
  PutFloat2(pOrderStatus->s_ol[i].ol_amount,
            orderStatusResponseIndexes[OS_AMT+(i*5)].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_AMT+(i*5)].iStartIndex]);
  PutNumeric(pOrderStatus->s_ol[i].ol_delivery_d.day,
             2, &szDate[0]);
  PutNumeric(pOrderStatus->s_ol[i].ol_delivery_d.month,
             2, &szDate[3]);
  PutNumeric(pOrderStatus->s_ol[i].ol_delivery_d.year,
             4, &szDate[6]);

memcpy(&orderStatusForm[orderStatusResponseIndexes[OS_SM_DATE+(i*5)].iStar
tIndex],
             szDate, orderStatusResponseIndexes[OS_SM_DATE+(i*5)].iLen);
 }
 /* need to blank out the rest of the unused item rows */
 jj = OS_SM_DATE + ((i-1)*5) + 1;
 for(kk=i; kk<MAX_OL; kk++)
 {
  /* there are 5 items per row - 4 plain and 1 with $*/
  for(mm=0; mm<3; mm++)
  {
   memcpy(&orderStatusForm[orderStatusResponseIndexes[jj].iStartIndex],
          szBlanks, orderStatusResponseIndexes[jj].iLen);
   jj++;
  }
  /* blank out the '$' for the blank $values */
  memcpy(&orderStatusForm[orderStatusResponseIndexes[jj].iStartIndex-1],
         szBlanks, orderStatusResponseIndexes[jj].iLen+1);
  jj++;
  memcpy(&orderStatusForm[orderStatusResponseIndexes[jj].iStartIndex],
         szBlanks, orderStatusResponseIndexes[jj].iLen);
  jj++;
 }

 PUT_STRING(NULL, 0, 0, StrStruct[ssCnt]);
 PutHTMLStrings(StrStruct, orderStatusForm,
                giResponseLen[ORDER_STATUS_RESPONSE],
                &szOutput, &iOutputLen);

 TRANSACTION_DEBUG_STAGE( pOrderStatus, UNRESERVING );
 pUnreserveTransactionStruct( ORDER_STATUS_TRANS, &pOrderStatus );

 SendResponse(pECB, szOutput, iOutputLen);

 UNRESERVE_RESPONSE( ORDER_STATUS_RESPONSE, orderStatusForm );

 if( szOutput != orderStatusForm )
  UNRESERVE_PANIC_FORM( szOutput );

 return( retcode );
}

/* FUNCTION: int TPCCStockLevelResponse(int retcode,
 *                                      StockLevelData
*stockLevelData)
 *
 * PURPOSE:          This function puts the response data for the transaction
 *                   into the form and sends the form back to the browser.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK  *pECB       pointer to
structure containing
 *                                                        internet
service information.
 *                   int                      retcode return status from db call
```

```
 *                          StockLevelData          *stockLevelData         pointer to
structure containing
 *                                                                          data for
this transaction.
 *
 * RETURNS:          none
 *
 * COMMENTS:         none
 */

int
TPCCStockLevelResponse( int retcode, StockLevelData *pStockLevel )
{
  char *stockLevelForm;
  EXTENSION_CONTROL_BLOCK *pECB;

  pECB = (EXTENSION_CONTROL_BLOCK *)pStockLevel->CC;

  if ( ERR_DB_PENDING == retcode )
  {
    return( retcode );
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {
    SendErrorResponse( pECB, ERR_STOCKLEVEL_NOT_PROCESSED, NULL,
                       pStockLevel->w_id, pStockLevel->ld_id,
                       (pConnData)pStockLevel );
    TRANSACTION_DEBUG_STAGE( pStockLevel, UNRESERVING );
    pUnreserveTransactionStruct( STOCK_LEVEL_TRANS, &pStockLevel );
    return( retcode );
  }
  else if ( ERR_DB_SUCCESS != retcode )
  {
    SendErrorResponse( pECB, retcode, NULL,
                       pStockLevel->w_id, pStockLevel->ld_id,
                       (pConnData)pStockLevel );
    TRANSACTION_DEBUG_STAGE( pStockLevel, UNRESERVING );
    pUnreserveTransactionStruct( STOCK_LEVEL_TRANS, &pStockLevel );
    return( retcode );
  }

  RESERVE_RESPONSE( STOCK_LEVEL_RESPONSE, stockLevelForm );

  PutNumeric(WDID(pStockLevel->w_id,pStockLevel->ld_id),
             stockLevelResponseIndexes[SL_WDID].iLen,

&stockLevelForm[stockLevelResponseIndexes[SL_WDID].iStartIndex]);
  PutNumeric(pStockLevel->w_id,
             stockLevelResponseIndexes[SL_WID].iLen,

&stockLevelForm[stockLevelResponseIndexes[SL_WID].iStartIndex]);
  PutNumeric(pStockLevel->ld_id,
             stockLevelResponseIndexes[SL_DID].iLen,

&stockLevelForm[stockLevelResponseIndexes[SL_DID].iStartIndex]);
  PutNumeric(pStockLevel->threshold,
             stockLevelResponseIndexes[SL_TH].iLen,
             &stockLevelForm[stockLevelResponseIndexes[SL_TH].iStartIndex]);
  PutNumeric(pStockLevel->low_stock,
             stockLevelResponseIndexes[SL_LOW].iLen,

&stockLevelForm[stockLevelResponseIndexes[SL_LOW].iStartIndex]);

  TRANSACTION_DEBUG_STAGE( pStockLevel, UNRESERVING );
  pUnreserveTransactionStruct( STOCK_LEVEL_TRANS, &pStockLevel );

  SendResponse(pECB, stockLevelForm,
               giResponseLen[STOCK_LEVEL_RESPONSE]);

  UNRESERVE_RESPONSE( STOCK_LEVEL_RESPONSE, stockLevelForm );

  return( retcode );
}


/* FUNCTION: void TPCCResponseComplete( EXTENSION_CONTROL_BLOCK
*pECB )
 *
 * PURPOSE:
 *    This function completes the asynchronous web transaction.
 *
 * ARGUMENTS:
 *    EXTENSION_CONTROL_BLOCK          *pECB                   Server
context structure.
 *
 * RETURNS:
 *    None
 *
 * COMMENTS:
 */

void
TPCCResponseComplete( CallersContext CC )
{
  DWORD    status;
  EXTENSION_CONTROL_BLOCK *pECB;

  pECB = (EXTENSION_CONTROL_BLOCK *)CC;
  status = HSE_STATUS_SUCCESS_AND_KEEP_CONN;
  (pECB->ServerSupportFunction)(pECB->ConnID,
HSE_REQ_DONE_WITH_SESSION,
                                &status, NULL, NULL);
}

int
ParseDeliveryQuery( char *lpszQueryString, pDeliveryData pDelivery )
{
  char                              *ptr;
  char                              *deliveryVals[MAXDELIVERYVALS];

  PARSE_QUERY_STRING(lpszQueryString, MAXDELIVERYVALS,
                     deliveryStrs, deliveryVals);

  if ( !GetValuePtr(deliveryVals, QUEUETIME, &ptr) )
    return ERR_DELIVERY_MISSING_QUEUETIME_KEY;

  if ( !GetNumeric(ptr, &pDelivery->queue_time) )
    return ERR_DELIVERY_QUEUETIME_INVALID;

  if ( !GetValuePtr(deliveryVals, OCD, &ptr) )
    return ERR_DELIVERY_MISSING_OCD_KEY;

  if ( !GetNumeric(ptr, &pDelivery->o_carrier_id) )
    return ERR_DELIVERY_CARRIER_INVALID;

  if ( pDelivery->o_carrier_id < 1 ||
       pDelivery->o_carrier_id > DISTRICTS_PER_WAREHOUSE )
    return ERR_DELIVERY_CARRIER_ID_RANGE;

  return( ERR_SUCCESS );
}

int
ParseStockLevelQuery( char *lpszQueryString, pStockLevelData pStockLevel )
{
  char                              *ptr;
  char                              *stockLevelVals[MAXSTOCKLEVELVALS];

  PARSE_QUERY_STRING(lpszQueryString, MAXSTOCKLEVELVALS,
                     stockLevelStrs, stockLevelVals);

  if ( !GetValuePtr(stockLevelVals, TT, &ptr))
    return ERR_STOCKLEVEL_MISSING_THRESHOLD_KEY;

  if ( !GetNumeric(ptr, &pStockLevel->threshold) )
    return ERR_STOCKLEVEL_THRESHOLD_INVALID;

  if ( pStockLevel->threshold >= 100 || pStockLevel->threshold < 0 )
    return ERR_STOCKLEVEL_THRESHOLD_RANGE;

  return( ERR_SUCCESS );
}

/* FUNCTION: int ProcessDeliveryQuery( EXTENSION_CONTROL_BLOCK
*pECB,
 *
 * PURPOSE:          This function parses the query string, validates the data,
 *                   and sends the request to the db/transport and returns
 *                   a response to the browser.
 *
```

```
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK   *pECB        ptr to the
structure
 *
                containing the internet server
 *
                information.
 *
 * RETURNS:         int                       status
 *
 * COMMENTS:        None
 *
 */

int
ProcessDeliveryQuery( EXTENSION_CONTROL_BLOCK *pECB, char
*lpszQueryString,
                        int w_id, int ld_id )
{
  int                       retcode;
  pDeliveryData             pDelivery;
  pDeliveryData
                pCompletedDeliveries[DELIVERY_RESPONSE_COUNT];

  pDelivery = pReserveTransactionStruct( DELIVERY_TRANS );

  pDelivery->w_id = w_id;
  pDelivery->ld_id = ld_id;
  pDelivery->CC = (CallersContext)pECB;

  if ( ERR_SUCCESS != ( retcode = ParseDeliveryQuery( lpszQueryString,

pDelivery )))
   {
    pUnreserveTransactionStruct( DELIVERY_TRANS, &pDelivery );
    return( retcode );
   }

  TRANSACTION_DEBUG_STAGE( pDelivery, CALLING_LH );

  retcode = pTPCCDelivery( &pDelivery, pCompletedDeliveries );

#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));
#endif

  TRANSACTION_DEBUG_STAGE( pDelivery, CALLING_RESP );

  retcode = TPCCDeliveryResponse( retcode, pDelivery, pCompletedDeliveries );

  return( retcode );
}

/* FUNCTION: int ProcessNewOrderQuery( EXTENSION_CONTROL_BLOCK
*pECB,
 *
 * PURPOSE:          This function parses the query string, validates the data,
 *                   and sends the request to the db/transport and returns
 *                   a response to the browser.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK   *pECB        ptr to
structure containing
 *                                                          internet
server info
 *
 * RETURNS:         int                       status
 *
 * COMMENTS:        None
 *
 */

int
ProcessNewOrderQuery( EXTENSION_CONTROL_BLOCK *pECB, char
*lpszQueryString,
                        int w_id, int ld_id )
{
  int                       retcode;
  NewOrderData              *pNewOrder;

  pNewOrder = pReserveTransactionStruct( NEW_ORDER_TRANS );

  pNewOrder->w_id = w_id;
  pNewOrder->ld_id = ld_id;
  pNewOrder->CC = (CallersContext)pECB;

  if ( ERR_SUCCESS != ( retcode = ParseNewOrderQuery( lpszQueryString,

pNewOrder )))
   {
    pUnreserveTransactionStruct( NEW_ORDER_TRANS, &pNewOrder );
    return( retcode );
   }

  TRANSACTION_DEBUG_STAGE( pNewOrder, CALLING_LH );

  retcode = pTPCCNewOrder( &pNewOrder );

#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));
#endif

  TRANSACTION_DEBUG_STAGE( pNewOrder, CALLING_RESP );

  retcode = TPCCNewOrderResponse( retcode, pNewOrder );

  return( retcode );
}

/* FUNCTION: int ProcessOrderStatusQuery( EXTENSION_CONTROL_BLOCK
*pECB,
 *
 * PURPOSE:          This function parses the query string, validates the data,
 *                   and sends the request to the db/transport and returns
 *                   a response to the browser.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK   *pECB        ptr to
structure that contains
 *                                                          the
internet server info.
 *
 * RETURNS:         int                       status
 *
 * COMMENTS:        None
 *
 */

int
ProcessOrderStatusQuery( EXTENSION_CONTROL_BLOCK *pECB, char
*lpszQueryString,
                        int w_id, int ld_id )
{
  int                       retcode;
  OrderStatusData           *pOrderStatus;

  pOrderStatus = pReserveTransactionStruct( ORDER_STATUS_TRANS );

  pOrderStatus->w_id = w_id;
  pOrderStatus->ld_id = ld_id;
  pOrderStatus->CC = (CallersContext)pECB;

  if( ERR_SUCCESS != ( retcode = ParseOrderStatusQuery( lpszQueryString,

                pOrderStatus )))
  {
   pUnreserveTransactionStruct( ORDER_STATUS_TRANS, &pOrderStatus );
   return( retcode );
  }

  TRANSACTION_DEBUG_STAGE( pOrderStatus, CALLING_LH );

  retcode = pTPCCOrderStatus( &pOrderStatus );

#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));
#endif

  TRANSACTION_DEBUG_STAGE( pOrderStatus, CALLING_RESP );

  retcode = TPCCOrderStatusResponse( retcode, pOrderStatus );

  return( retcode );
```

```
}

/* FUNCTION: int ProcessPaymentQuery( EXTENSION_CONTROL_BLOCK
*pECB,
 *
 * PURPOSE:              This function gets and validates the input data from the
 *                       payment form filling in the required input variables.
 *                       It then calls the SQLPayment transaction, constructs the
 *                       output form and writes it back to client browser.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB        ptr to
structure that contains
 *                                                               the
internet server info.
 *
 * RETURNS:              int                       status
 *
 * COMMENTS:             None
 *
 */

int
ProcessPaymentQuery( EXTENSION_CONTROL_BLOCK *pECB, char
*lpszQueryString,
                     int w_id, int ld_id )
{
  int                       retcode;
  PaymentData               *pPayment;

  pPayment = pReserveTransactionStruct( PAYMENT_TRANS );

  pPayment->w_id = w_id;
  pPayment->ld_id = ld_id;
  pPayment->CC = (CallersContext)pECB;

  if( ERR_SUCCESS != ( retcode = ParsePaymentQuery( lpszQueryString,

pPayment )))
  {
    pUnreserveTransactionStruct( PAYMENT_TRANS, &pPayment );
    return( retcode );
  }

  TRANSACTION_DEBUG_STAGE( pPayment, CALLING_LH );

  retcode = pTPCCPayment( &pPayment );

#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));
#endif

  TRANSACTION_DEBUG_STAGE( pPayment, CALLING_RESP );

  retcode = TPCCPaymentResponse( retcode, pPayment );

  return( retcode );
}

/* FUNCTION: int ProcessStockLevelQuery( EXTENSION_CONTROL_BLOCK
*pECB,
 *
 * PURPOSE:              This function gets and validates the input data from the
 *                       Stock Level form filling in the required input variables.
 *                       It then calls the SQLStockLevel transaction, constructs
 *                       the output form and writes it back to client browser.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB        ptr to
structure that contains
 *                                                               the
internet server info.
 *                       int                       iSyncId      client
browser sync id
 *
 * RETURNS:              int                       status
 *
 * COMMENTS:             None
 *
 */

int
ProcessStockLevelQuery( EXTENSION_CONTROL_BLOCK *pECB, char
*lpszQueryString,
                        int w_id, int ld_id )
{
  int                       retcode;
  StockLevelData            *pStockLevel;

  pStockLevel = pReserveTransactionStruct( STOCK_LEVEL_TRANS );

  pStockLevel->w_id = w_id;
  pStockLevel->ld_id = ld_id;
  pStockLevel->CC = (CallersContext)pECB;

  if ( ERR_SUCCESS != ( retcode = ParseStockLevelQuery( lpszQueryString,

                pStockLevel )))
  {
    pUnreserveTransactionStruct( STOCK_LEVEL_TRANS, &pStockLevel );
    return( retcode );
  }

  TRANSACTION_DEBUG_STAGE( pStockLevel, CALLING_LH );

  retcode = pTPCCStockLevel( &pStockLevel );

#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));
#endif

  TRANSACTION_DEBUG_STAGE( pStockLevel, CALLING_RESP );

  retcode = TPCCStockLevelResponse( retcode, pStockLevel );

  return( retcode );
}

/* FUNCTION: BOOL GetValuePtr(char *pProcessedQuery[], int iIndex,
 *                                                    char **pValue)
 *
 * PURPOSE:              This function passes back a pointer to the char ptr to the
 *                       value requested.
 *
 * ARGUMENTS:            char  *pProcessedQuery[]   char* array of query string values
 *             int   iIndex    index into the ProcessedQuery array
 *                     char *pValue          character ptr into to the key's value
 *
 * RETURNS:              BOOL        FALSE       there is no valid ptr for this value
 *                               TRUE  the ptr returned is valid
 *
 *
 * COMMENTS:             none.
 */

BOOL
GetValuePtr(char *pProcessedQuery[], int iIndex, char **pValue)
{
  *pValue = pProcessedQuery[iIndex];

  if(NULL == *pValue)return FALSE;

  return TRUE;
}

/* FUNCTION: void MakeDeliveryTemplates( char *deliveryForm,
 *                                                                  char
*deliveryResponse )
 *
 * PURPOSE:              This function constructs the templates for the
 *                       Delivery input and response HTML forms.
 *
 * ARGUMENTS:            char *deliveryForm        pointer to the HTML input form.
 *                       char *deliveryResponse   pointer to the HTML response
form.
 *
 * RETURNS:              None
 *
 * COMMENTS:             None
 */

void
```

```
MakeDeliveryTemplates( char *deliveryForm, char *deliveryResponse )
{
  int           curLen;

  /* first make the input form template */
  curLen = sprintf(deliveryForm, szFormTemplate, szModName);
  ParseTemplateString(deliveryForm, &curLen, szDeliveryFormTemp2i,
                                deliveryFormIndexesI);
  giFormLen[DELIVERY_FORM] = curLen;

  /* now make the process form template */
  curLen = sprintf(deliveryResponse, szFormTemplate, szModName);
  ParseTemplateString(deliveryResponse, &curLen, szDeliveryFormTemp2p,
                                deliveryFormIndexesP);
  giResponseLen[DELIVERY_RESPONSE] = curLen;
}

/* FUNCTION: void MakeNewOrderTemplates(char *newOrderForm,
                                                                        char
*newOrderResponse )
 *
 * PURPOSE:             This function constructs the templates for both the input
 *                      and the response HTML forms for NewOrder function.
 *
 * ARGUMENTS:           char            *newOrderForm           pointer to the input
HTML form.
 *                      char            *newOrderResponse pointer to the response
HTML form.
 *
 * RETURNS:             none
 *
 * COMMENTS:            none.
 */

void
MakeNewOrderTemplates( char *newOrderForm, char *newOrderResponse )
{
  int           curLen;

  /* first make the input template */
  curLen = sprintf(newOrderForm, szFormTemplate, szModName);
  ParseTemplateString(newOrderForm, &curLen, szNewOrderFormTemp2i,
                                newOrderFormIndexes);
  giFormLen[NEW_ORDER_FORM] = curLen;

  /* now make the process template */
  curLen = sprintf(newOrderResponse, szFormTemplate, szModName);
  ParseTemplateString(newOrderResponse, &curLen, szNewOrderFormTemp2p,
                                newOrderResponseIndexes);
  giResponseLen[NEW_ORDER_RESPONSE] = curLen;
}

/* FUNCTION: void MakeOrderStatusTemplates(char *orderStatusForm,
 *                                                      char *orderStatusResponse)
 *
 * PURPOSE:             This function constructs the template HTML forms
 *                      for Order Status.
 *
 * ARGUMENTS:           char *orderStatusForm         pointer to the input HTML form
 *                      char *orderStatusResponse     pointer to the response HTML
form
 *
 * RETURNS:             none
 *
 * COMMENTS:    none
 */

void
MakeOrderStatusTemplates(char *orderStatusForm, char *orderStatusResponse)
{
  int           curLen;

  /* first make the input form template */
  curLen = sprintf(orderStatusForm, szFormTemplate, szModName);
  ParseTemplateString(orderStatusForm, &curLen, szOrderStatusFormTemp2i,
                                orderStatusFormIndexes);
  giFormLen[ORDER_STATUS_FORM] = curLen;

  /* now make the process template */
  curLen = sprintf(orderStatusResponse, szFormTemplate, szModName);
  ParseTemplateString(orderStatusResponse, &curLen, szOrderStatusFormTemp2p,
                                orderStatusResponseIndexes);
  giResponseLen[ORDER_STATUS_RESPONSE] = curLen;
}

/* FUNCTION: void MakePaymentTemplates(char *paymentForm,
 *                                                                      char
*paymentResponse)
 *
 * PURPOSE:             This function constructs the templates for the
 *                      Payment input and response HTML forms.
 *
 * ARGUMENTS:           char            *paymentForm            pointer to the input
HTML form.
 *                      char            *paymentResponse pointer to the response
HTML form.
 *
 * RETURNS:             none
 *
 * COMMENTS:            none
 */

void
MakePaymentTemplates(char *paymentForm, char *paymentResponse)
{
  int           curLen;

  /* first make the input form template */
  curLen = sprintf(paymentForm, szFormTemplate, szModName);
  ParseTemplateString(paymentForm, &curLen, szPaymentFormTemp2i,
                                paymentFormIndexes);
  giFormLen[PAYMENT_FORM] = curLen;

  /* now make the process form template */
  curLen = sprintf(paymentResponse, szFormTemplate, szModName);
  ParseTemplateString(paymentResponse, &curLen, szPaymentFormTemp2p,
                                paymentResponseIndexes);
  giResponseLen[PAYMENT_RESPONSE] = curLen;
}

/* FUNCTION: void MakeStockLevelTemplates(char *stockLevelForm,
 *                                                      char *stockLevelResponse)
 *
 * PURPOSE:             This function constructs the templates for the
 *                      input and response Stock Level HTML pages.
 *
 * ARGUMENTS:           char            *stockLevelForm         pointer to the input
HTML form
 *                      char            *stockLevelResponse   pointer to the response
HTML form
 *
 * RETURNS:             none
 *
 * COMMENTS:            none
 */
void
MakeStockLevelTemplates(char *stockLevelForm, char *stockLevelResponse)
{
  int           curLen;

  /* first make the input template */
  curLen = sprintf(stockLevelForm, szFormTemplate, szModName);
  ParseTemplateString(stockLevelForm, &curLen, szStockLevelFormTemp2i,
                                stockLevelFormIndexes);
  giFormLen[STOCK_LEVEL_FORM] = curLen;

  /* now make the process template */
  curLen = sprintf(stockLevelResponse, szFormTemplate, szModName);
  ParseTemplateString(stockLevelResponse, &curLen, szStockLevelFormTemp2p,
                                stockLevelResponseIndexes);
  giResponseLen[STOCK_LEVEL_RESPONSE] = curLen;
}

/* FUNCTION: void MakeResponseHeader(void)
 *
 * PURPOSE:             This function constructs the HTML response header.
 *
 * ARGUMENTS:           char            *responseString         pointer to the
header string
 *
```

```
 * RETURNS:              none
 *
 * COMMENTS:             none
 */
void
MakeResponseHeader(void)
{
  ParseTemplateString(szResponseHeader, &responseHeaderLen,
                                  szResponseHeaderTemplate, responseHeaderIndexes);
}

/* FUNCTION: void MakePanicPool( DWORD dwResponseSize )
 *
 * PURPOSE:              This function builds the array of panic forms to be used
 *                       by the threads as they need an oversize form, or to report
 *                       an error.
 *
 * ARGUMENTS:            none
 *
 * RETURNS:              none
 *
 * COMMENTS:             none
 */

void
MakePanicPool( DWORD dwResponseSize )
{
  int iMallocSize;
  char *pForm;
  DWORD ii;

  /* set up area for forms (including errors) that are built on the fly. */
  iMallocSize = (((char *)&gpPanicForms->index - (char *)gpPanicForms) +
                                  (((char *)gpPanicForms->forms - (char *)gpPanicForms-
>index)
                                   * dwResponseSize) +
                                  (((char *)&gpPanicForms->forms[PANIC_FORM_SIZE] -
                                    (char *)&gpPanicForms->forms[0]) * dwResponseSize));
  gpPanicForms = malloc( iMallocSize );
  InitializeCriticalSection( &gpPanicForms->critSec );
#ifdef FFE_DEBUG
  gpPanicForms->iMaxIndex = dwResponseSize - 1;
#endif
  gpPanicForms->iNextFree = 0;
  pForm =
    ((char *)&gpPanicForms->index[0] +
     (((char *)&gpPanicForms->forms[0] - (char *)&gpPanicForms->index[0]) *
      dwResponseSize));

  for( ii = 0; ii < dwResponseSize; ii++ )
  {
    gpPanicForms->index[ii] = pForm;
    pForm += PANIC_FORM_SIZE;
  }
}

/* FUNCTION: void DeletePanicPool( void )
 *
 * PURPOSE:              This function destroys the array of panic forms to be used
 *                       by the threads as they need an oversize or error form.
 *
 * ARGUMENTS:            none
 *
 * RETURNS:              none
 *
 * COMMENTS:             none
 */
void
DeletePanicPool( void )
{
  DeleteCriticalSection( &gpPanicForms->critSec );
  free( gpPanicForms );
}

/* FUNCTION: void MakeTemplatePool( DWORD dwFormSize, DWORD
dwResponseSize )
 *
 * PURPOSE:              This function builds the array of forms to be used
 *                       by the threads as they need a form.  The forms are
 *                       reserved and released by each thread as needed.
 *
 * ARGUMENTS:            none
 *
 * RETURNS:              none
 *
 * COMMENTS:             none
 */

void
MakeTemplatePool( DWORD dwFormSize, DWORD dwResponseSize )
{
  char szDeliveryForm[sizeof(szFormTemplate)+FILENAMESIZE+
                                        sizeof(szDeliveryFormTemp2i)];
  char szNewOrderForm[sizeof(szFormTemplate)+FILENAMESIZE+
                                        sizeof(szNewOrderFormTemp2i)];
  char szOrderStatusForm[sizeof(szFormTemplate)+FILENAMESIZE+
                                          sizeof(szOrderStatusFormTemp2i)];
  char szPaymentForm[sizeof(szFormTemplate)+FILENAMESIZE+
                                       sizeof(szPaymentFormTemp2i)];
  char szStockLevelForm[sizeof(szFormTemplate)+FILENAMESIZE+
                                         sizeof(szStockLevelFormTemp2i)];
  char szDeliveryResponse[sizeof(szFormTemplate)+FILENAMESIZE+
                                          sizeof(szDeliveryFormTemp2p)];
  char szNewOrderResponse[sizeof(szFormTemplate)+FILENAMESIZE+
                                          sizeof(szNewOrderFormTemp2p)];
  char szOrderStatusResponse[sizeof(szFormTemplate)+FILENAMESIZE+
                                           sizeof(szOrderStatusFormTemp2p)];
  char szPaymentResponse[sizeof(szFormTemplate)+FILENAMESIZE+
                                         sizeof(szPaymentFormTemp2p)];
  char szStockLevelResponse[sizeof(szFormTemplate)+FILENAMESIZE+
                                           sizeof(szStockLevelFormTemp2p)];
  int           iFormLen[NUMBER_POOL_FORM_TYPES];
  int           iResponseLen[NUMBER_POOL_RESPONSE_TYPES];
  int iMallocSize;
  int iRowSize;
  DWORD ii;
  int jj;
  char *pForm;
  char *pResponse;

  /* now build the forms that are static */
  MakeDeliveryTemplates( szDeliveryForm, szDeliveryResponse );
  MakeNewOrderTemplates( szNewOrderForm, szNewOrderResponse );
  MakeOrderStatusTemplates( szOrderStatusForm, szOrderStatusResponse );
  MakePaymentTemplates( szPaymentForm, szPaymentResponse );
  MakeStockLevelTemplates( szStockLevelForm, szStockLevelResponse );

  /* calculate the size of one row of forms */
  iRowSize = 0;
  for( jj = 0; jj < NUMBER_POOL_FORM_TYPES; jj++ )
  {
    iFormLen[jj] = ( giFormLen[jj] + 8 ) & ( ~(int)7 );
    iRowSize += iFormLen[jj];
  }

  iMallocSize = (((char *)&gpForms->index - (char *)gpForms) +
                                  (((char *)gpForms->forms - (char *)gpForms->index)
                                   * dwFormSize * NUMBER_POOL_FORM_TYPES ) +
                                  (((char *)&gpForms->forms[iRowSize * dwFormSize] -
                                    (char *)&gpForms->forms[0])));
  gpForms = malloc( iMallocSize );

  for( jj = 0; jj < NUMBER_POOL_FORM_TYPES; jj++ )
  {
    InitializeCriticalSection( &gpForms->critSec[jj] );
    gpForms->iNextFreeForm[jj] = 0;
    gpForms->iFirstFormIndex[jj] = jj * dwFormSize;
#ifdef FFE_DEBUG
    gpForms->iMaxIndex[jj] = dwFormSize - 1;
#endif
  }

  pForm = ((char *)&gpForms->index[0] +
                  (((char *)&gpForms->forms[0] - (char *)&gpForms->index[0]) *
                   NUMBER_POOL_FORM_TYPES * dwFormSize));
  for( ii = 0; ii < dwFormSize; ii++ )
  {
    for( jj = 0; jj < NUMBER_POOL_FORM_TYPES; jj++ )
    {
      gpForms->index[jj*dwFormSize+ii] = pForm;
```

```
    pForm += iFormLen[jj];
   }
  }

  /* load the first row with the templates */
  pForm = gpForms->index[0];

  memcpy( pForm, szDeliveryForm, iFormLen[DELIVERY_FORM] );
  pForm += iFormLen[DELIVERY_FORM];

  memcpy( pForm, szNewOrderForm, iFormLen[NEW_ORDER_FORM] );
  pForm += iFormLen[NEW_ORDER_FORM];

  memcpy( pForm, szOrderStatusForm, iFormLen[ORDER_STATUS_FORM] );
  pForm += iFormLen[ORDER_STATUS_FORM];

  memcpy( pForm, szPaymentForm, iFormLen[PAYMENT_FORM] );
  pForm += iFormLen[PAYMENT_FORM];

  memcpy( pForm, szStockLevelForm, iFormLen[STOCK_LEVEL_FORM] );
  pForm += iFormLen[STOCK_LEVEL_FORM];

  /* copy the first row to all the other rows */
  pForm = gpForms->index[0];
  for( ii = 1; ii < dwFormSize; ii++ )
  {
   memcpy( gpForms->index[ii], pForm, iRowSize );
  }

  /* calculate the size of one row of responses */
  iRowSize = 0;
  for( jj = 0; jj < NUMBER_POOL_RESPONSE_TYPES; jj++ )
  {
   iResponseLen[jj] = ( giResponseLen[jj] + 8 ) & ( ~(int)7 );
   iRowSize += iResponseLen[jj];
  }

  iMallocSize = (((char *)&gpResponses->index - (char *)gpResponses) +
                 (((char *)gpResponses->responses - (char *)gpResponses-
>index)
                  * dwResponseSize *
NUMBER_POOL_RESPONSE_TYPES ) +
                 (((char *)&gpResponses->responses[iRowSize *
dwResponseSize] -
                   (char *)&gpResponses->responses[0])));
  gpResponses = malloc( iMallocSize );

  for( jj = 0; jj < NUMBER_POOL_RESPONSE_TYPES; jj++ )
  {
   InitializeCriticalSection( &gpResponses->critSec[jj] );
#ifdef FFE_DEBUG
   gpResponses->iMaxIndex[jj] = dwResponseSize - 1;
#endif
   gpResponses->iNextFreeResponse[jj] = 0;
   gpResponses->iFirstResponseIndex[jj] = jj * dwResponseSize;
  }

  pResponse = ((char *)&gpResponses->index[0] +
               (((char *)&gpResponses->responses[0] -
                 (char *)&gpResponses->index[0]) *
                NUMBER_POOL_RESPONSE_TYPES * dwResponseSize));
  for( ii = 0; ii < dwResponseSize; ii++ )
  {
   for( jj = 0; jj < NUMBER_POOL_RESPONSE_TYPES; jj++ )
   {
    gpResponses->index[jj*dwResponseSize+ii] = pResponse;
    pResponse += iResponseLen[jj];
   }
  }

  /* load the first row with the templates */
  pResponse = gpResponses->index[0];

  memcpy( pResponse, szDeliveryResponse, iResponseLen[DELIVERY_RESPONSE]
);
  pResponse += iResponseLen[DELIVERY_RESPONSE];

  memcpy( pResponse, szNewOrderResponse,
iResponseLen[NEW_ORDER_RESPONSE] );
  pResponse += iResponseLen[NEW_ORDER_RESPONSE];

  memcpy(pResponse, szOrderStatusResponse,
iResponseLen[ORDER_STATUS_RESPONSE]);
  pResponse += iResponseLen[ORDER_STATUS_RESPONSE];

  memcpy( pResponse, szPaymentResponse, iResponseLen[PAYMENT_RESPONSE]
);
  pResponse += iResponseLen[PAYMENT_RESPONSE];

  memcpy( pResponse, szStockLevelResponse,
iResponseLen[STOCK_LEVEL_RESPONSE] );
  pResponse += iResponseLen[STOCK_LEVEL_RESPONSE];

  /* copy the first row to all the other rows */
  pResponse = gpResponses->index[0];
  for( ii = 1; ii < dwResponseSize; ii++ )
  {
   memcpy( gpResponses->index[ii], pResponse, iRowSize );
  }
}

/* FUNCTION: void DeleteTemplatePool( void )
 *
 * PURPOSE:          This function destroys the array of forms to be used
 *                   by the threads as they need a form.
 *
 * ARGUMENTS:        none
 *
 * RETURNS:          none
 *
 * COMMENTS:         none
 */
void
DeleteTemplatePool( void )
{
  int                                   jj;

  for( jj = 0; jj < NUMBER_POOL_RESPONSE_TYPES; jj++ )
  {
   DeleteCriticalSection( &gpResponses->critSec[jj] );
  }
  free( gpResponses );

  for( jj = 0; jj < NUMBER_POOL_FORM_TYPES; jj++ )
  {
   DeleteCriticalSection( &gpForms->critSec[jj] );
  }
  free( gpForms );

  DeleteCriticalSection( &gpPanicForms->critSec );
  free( gpPanicForms );
}

/* FUNCTION: void BeginCmd( EXTENSION_CONTROL_BLOCK *pECB )
 *
 * PURPOSE:          This routine is executed in response to the browser query
 *                   'CMD=Begin&Server=??????'.
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK   *pECB       IIS
context structure pointer
 *                                                                  unique to
this connection.
 *                                                                  at login.
 * RETURNS:          None
 *
 * COMMENTS:         Specification of a server machine is required.
 */

void
BeginCmd( EXTENSION_CONTROL_BLOCK *pECB )
{
 SendWelcomeForm(pECB);
}

/* FUNCTION: void CheckpointCmd(EXTENSION_CONTROL_BLOCK *pECB,
 *
 * PURPOSE:          This function causes a checkpoint to occur in the database
 *
 * ARGUMENTS:        EXTENSION_CONTROL_BLOCK   *pECB       IIS
context structure pointer
```

```
 *                                                                    unique to
this connection.
 *
 * RETURNS:              None
 *
 * COMMENTS:             none
 */

void
CheckpointCmd(EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id )
{
  int                                 retcode;
  CheckpointData            *pCheckpoint;

  pCheckpoint = pReserveTransactionStruct( CHECKPOINT_TRANS );

  pCheckpoint->w_id = w_id;
  pCheckpoint->ld_id = ld_id;
  pCheckpoint->CC = (CallersContext)pECB;

  retcode = pTPCCCheckpoint( &pCheckpoint );
  if ( ERR_DB_SUCCESS == retcode ) {
    SendMainMenuForm(pECB, w_id, ld_id,
                                  "Checkpoint issued (non-blocking), completed
(blocking).");
  }
  else {
    SendErrorResponse( pECB, retcode, NULL, w_id, ld_id, NULL );
  }

  pUnreserveTransactionStruct( CHECKPOINT_TRANS, &pCheckpoint );
}

/* FUNCTION: void CheckpointShutdownCmd(EXTENSION_CONTROL_BLOCK
*pECB,
 *
 * PURPOSE:              This function causes initialization of the checkpoint
command.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB       IIS
context structure pointer
 *                                                                    unique to
this connection.
 *
 * RETURNS:              None
 *
 * COMMENTS:             none
 */

void
CheckpointShutdownCmd(EXTENSION_CONTROL_BLOCK *pECB, int w_id, int
ld_id )
{
  int                                 retcode;
  pConnData                 pConn;

  pConn = pReserveTransactionStruct( DISCONNECT_TRANS );

  pConn->w_id = w_id;
  pConn->ld_id = ld_id;
  pConn->CC = (CallersContext)pECB;

  if( ERR_DB_SUCCESS != ( retcode = pTPCCCheckpointDisconnect( &pConn )))
  {
    SendErrorResponse( pECB, retcode, NULL, w_id, ld_id, NULL );
  }
  else
  {
    SendMainMenuForm(pECB, w_id, ld_id, "Checkpoint Shutdown Succeded.");
  }

  pUnreserveTransactionStruct( DISCONNECT_TRANS, &pConn );
}

/* FUNCTION: void CheckpointStartupCmd(EXTENSION_CONTROL_BLOCK
*pECB,
 *
 * PURPOSE:              This function causes initialization of the checkpoint
command.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB       IIS
context structure pointer
 *                                                                    unique to
this connection.
 *
 * RETURNS:              None
 *
 * COMMENTS:             none
 */

void
CheckpointStartupCmd(EXTENSION_CONTROL_BLOCK *pECB, int w_id, int
ld_id )
{
  int                                 retcode;
  pLoginData                  pLogin;

  pLogin = pReserveTransactionStruct( CONNECT_TRANS );


  pLogin->w_id = w_id;
  pLogin->ld_id = ld_id;
  pLogin->CC = (CallersContext)pECB;

  if( ERR_SUCCESS != ( retcode = GetLoginDataCheckpoint( pLogin )) ||
      ERR_DB_SUCCESS != ( retcode = pTPCCCheckpointConnect( &pLogin )))
  {
    SendErrorResponse( pECB, retcode, NULL, w_id, ld_id, NULL );
  }
  else
  {
    SendMainMenuForm(pECB, w_id, ld_id, "Checkpoint Startup Succeded.");
  }

  pUnreserveTransactionStruct( CONNECT_TRANS, &pLogin );
}

/* FUNCTION: void ClearCmd(EXTENSION_CONTROL_BLOCK *pECB)
 *
 * PURPOSE:              This resets all terminals and resets the log file.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB       IIS
context structure pointer
 *                                                                    unique to
this connection.
 *
 * RETURNS:                        None
 *
 * COMMENTS:             This function resets the connection information for the
 *                       dll.  Any "users" with current connections will be given
 *                       an error message on their next transaction.
 */

void
ClearCmd(EXTENSION_CONTROL_BLOCK *pECB)
{
  pTPCCCloseLog( );
  pTPCCOpenLog( szPath );

  SendWelcomeForm(pECB);
}

/* FUNCTION: void ExitCmd(EXTENSION_CONTROL_BLOCK *pECB,
 *
 * PURPOSE:              This function deallocates the terminal associated with
 *                       the browser and presents the login screen.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB       IIS
context structure pointer
 *                                                                    unique to
this connection.
 * RETURNS:              None
 *
 * COMMENTS:             None
 *
 */

void
ExitCmd( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id )
{
```

```
  ConnData                *pConn;

  pConn = pReserveTransactionStruct( DISCONNECT_TRANS );

  pConn->w_id = w_id;
  pConn->ld_id = ld_id;
  pConn->CC = (CallersContext)pECB;

  pTPCCDisconnect( &pConn );

  pUnreserveTransactionStruct( DISCONNECT_TRANS, &pConn );

  SendWelcomeForm( pECB );
}

/* FUNCTION: void MenuCmd( EXTENSION_CONTROL_BLOCK *pECB,
 *
 * PURPOSE:              This function displays the main menu.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB        IIS
context structure pointer
 *                                                              unique to
this connection.
 * RETURNS:              None
 *
 * COMMENTS:             None
 *
 */

void
MenuCmd( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id )
{
  SendMainMenuForm(pECB, w_id, ld_id, NULL);
}

/* FUNCTION: void SubmitCmd( EXTENSION_CONTROL_BLOCK *pECB )
 *
 * PURPOSE:              This function assigns a unique terminal id to the calling
 *                       browser.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB        IIS
context structure pointer
 *                                                              unique to
this connection.
 * RETURNS:              None
 *
 * COMMENTS:             A terminal id can be allocated but still be invalid if the
 *                       requested warehouse number is outside the range specified
 *                       in the registry. This then will force the client id
 *                       to be invalid and an error message sent to the users browser.
 */

void
SubmitCmd( EXTENSION_CONTROL_BLOCK *pECB, int *w_id, int *ld_id )
{
  int           iStatus;
  LoginData *pLogin;
  char          *ptr;

  if ( !GetCharKeyValuePtr( pECB->lpszQueryString, '4', &ptr ) ||
       ( 0 == ( *w_id = atoi( ptr ))) ||
       ( *w_id < 0 ))
  {
    SendErrorResponse( pECB, ERR_W_ID_INVALID, NULL, *w_id, -1, NULL );
    goto SubmitDone;
  }

  if ( !GetCharKeyValuePtr( pECB->lpszQueryString, '5', &ptr ) ||
       ( 0 == ( *ld_id = atoi( ptr ))) ||
       ( *ld_id > 10 ) ||
       ( *ld_id < 0 ))
  {
    SendErrorResponse( pECB, ERR_D_ID_INVALID, NULL, *w_id, *ld_id, NULL
);
    goto SubmitDone;
  }

  pLogin = pReserveTransactionStruct( CONNECT_TRANS );

  pLogin->w_id = *w_id;
  pLogin->ld_id = *ld_id;
  pLogin->CC = (CallersContext)pECB;

  if( ERR_SUCCESS != ( iStatus = GetLoginData( pLogin )) ||
      ERR_DB_SUCCESS != ( iStatus = pTPCCConnect( &pLogin ) ))
  {
    SendErrorResponse( pECB, iStatus, NULL, *w_id, *ld_id, NULL );
  }
  else
  {
    SendMainMenuForm(pECB, *w_id, *ld_id, NULL);
  }

  pUnreserveTransactionStruct( CONNECT_TRANS, &pLogin );

SubmitDone:
  return;
}

/* FUNCTION: void MemoryCheckCmd( EXTENSION_CONTROL_BLOCK
*pECB,
 *
 * PURPOSE:              This function displays the main menu.
 *
 * ARGUMENTS:            EXTENSION_CONTROL_BLOCK   *pECB        IIS
context structure pointer
 *                                                              unique to
this connection.
 * RETURNS:              None
 *
 * COMMENTS:             None
 *
 */

void
MemoryCheckCmd( EXTENSION_CONTROL_BLOCK *pECB, int w_id, int ld_id )
{
  _ASSERTE( _CrtCheckMemory( ));

  SendErrorResponse( pECB, ERR_SUCCESS, NULL, w_id, ld_id, NULL );
}

/* FUNCTION: BOOL GetKeyValuePtr( char *szIPtr, char *szKey, char **pszOPtr )
 *
 * PURPOSE:              This function searches the input string for the key
 *                       specified.  If found, it returns a pointer to the value.
 *
 * ARGUMENTS:            char    *szIPtr                 pointer to string to
check.
 *                       char            *szKey                  pointer to key to find.
 *                       char            **pszOPtr   pointer to value.
 *
 * RETURNS:              BOOL    FALSE                   if key is not found.
 *                               TRUE                    if key is found.
 *
 * COMMENTS:             A side affect of this routine is that the output string
 *                       pointer will either point at the start of the value being
 *                       searched or at the *start* point where ptr originated.
 */
BOOL
GetKeyValuePtr( char *szIPtr, char *szKey, char **pszOPtr )
{
  char *szPtr1, *szPtr2;

  *pszOPtr = szIPtr;
  while (*szIPtr)
  {
    szPtr1 = szIPtr;
    szPtr2 = szKey;

    while ( *szPtr1 && *szPtr2 && 0 == ( *szPtr1 - *szPtr2 ))
      szPtr1++, szPtr2++;

    if ( '=' == *szPtr1 && '\0' == *szPtr2 )
    {
      *pszOPtr = ++szPtr1;
      return TRUE;
    }

    szIPtr++;
```

```
  }

  return FALSE;
}

/* FUNCTION: BOOL GetKeyValueCharPtr( char *szIPtr, char cKey, char **pszOPtr
)
 *
 * PURPOSE:             This function searches the input string for the single char key
 *                      specified.  If found, it returns a pointer to the value.
 *
 * ARGUMENTS:           char    *szIPtr                         pointer to string to
check.
 *                      char            cKey                    pointer to key to find.
 *                      char            **pszOPtr   pointer to value.
 *
 * RETURNS:             BOOL    FALSE                   if key is not found.
 *                              TRUE                    if key is found.
 *
 * COMMENTS:            A side affect of this routine is that the output string
 *                      pointer will either point at the start of the value being
 *                      searched or at the *start* point where ptr originated.
 */
BOOL
GetCharKeyValuePtr( char *szIPtr, char cKey, char **pszOPtr )
{
  BOOL      bGotStart;

  *pszOPtr = szIPtr;
  bGotStart = FALSE;
  while( *szIPtr )
  {
    if( cKey == *szIPtr && '=' == *++szIPtr )
    {
      *pszOPtr = ++szIPtr;
      return TRUE;
    }
    while( *szIPtr )
    {
      if( '&' == *szIPtr )
      {
                szIPtr++;
                break;
      }
      szIPtr++;
    }
  }

  return FALSE;
}

/* FUNCTION: BOOL GetNumeric(char *ptr, int *iValue)
 *
 * PURPOSE:             This function converts the string value to integer, and
 *                      determines if the string is terminated properly.  If it
 *                      contains non-numeric characters or if any characters
 *                      other than '&' or '\0' terminate the integer portion
 *                      of the string, this function fails.
 *
 * ARGUMENTS:           char    *ptr   pointer to string to check.
 *
 * RETURNS:             BOOL    FALSE       if string is not all numeric and
properly
 *                                          terminated.
 *                              TRUE        if string contains only numeric
characters
 *                                          i.e. '0' - '9' and is properly
terminated.
 *
 * COMMENTS:            None
 *
 */
BOOL
GetNumeric(char *ptr, int *iValue)
{
  int c;            /* current char */
  int total;        /* current total */
  BOOL bGotSomething = FALSE;

  c = (int)(unsigned char)*ptr++;
  total = 0;

  while ((c >= '0') && (c <= '9'))
  {
    total = 10 * total + (c - '0');     /* accumulate digit */
    c = (int)(unsigned char)*ptr++;     /* get next char */
    bGotSomething = TRUE;
  }
  if(('\0' == c) || ('&' == c) && bGotSomething)
  {
    *iValue = total;
    return (TRUE);   /* return result */
  }
  else
  {
    *iValue = 0;
    return(FALSE);
  }
}

/* FUNCTION: BOOL GetWDID(char *ptr, int *lw_id, int *ld_id, char **optr)
 *
 * PURPOSE:             This function converts the string value to a pair of integers
 *                      where the ascii numeric field represents an encoded
warehouse
 *                      and district id.  The least significant digit is one less than
 *                      the actual local district id, and the remaining high order
 *                      digits are 10 times the actual local warehouse id.
 *
 * ARGUMENTS:           char    *ptr   pointer to string to check.
 *
 * RETURNS:             BOOL    FALSE       if string is not all numeric and
properly
 *                                          terminated.
 *                              TRUE        if string contains only numeric
characters
 *                                          i.e. '0' - '9' and is properly
terminated.
 *
 * COMMENTS:            A side affect of this routine is that the output string
 *                      pointer will either point at the end of the values being
 *                      searched or at the *start* point where ptr originated.
 *
 */
BOOL
GetWDID(char *ptr, int *lw_id, int *ld_id, char **optr)
{
  int c;            /* current char */
  int pc;             /* previous character */
  int total;        /* current total */
  BOOL bGotSomething = FALSE;

  *lw_id = 0;
  *ld_id = 0;
  total = 0;

  *optr = ptr;
  pc = (int)(unsigned char)*ptr++;
  if((pc < '0') || (pc > '9'))
    return FALSE;

  c = (int)(unsigned char)*ptr++;

  while ((c >= '0') && (c <= '9'))
  {
    total = 10 * total + (pc - '0');     /* accumulate digit */
    pc = c;
    c = (int)(unsigned char)*ptr++;     /* get next char */
    bGotSomething = TRUE;
  }
  if(('\0' == c) || ('&' == c) && bGotSomething)
  {
    *lw_id = total;
    *ld_id = (pc - '0') + 1;
    *optr = ptr;
    return TRUE;   /* return result */
  }
  else
    return FALSE;
```

```
}

/* FUNCTION: BOOL GetKeyValueString(char *szIPtr, char *szKey,
 *                                  char *szValue, int iSize)
 *
 * PURPOSE:          This function searches for the key specified and returns
 *                   the string value associated with it.
 *
 * ARGUMENTS:        char     *szIPtr                 string to
search
 *                   char     *szKey                  key to
search for
 *                   char     *szValue                location to store value
 *                   int      iSize                   size of
output array.
 *
 * RETURNS:          BOOL     FALSE                   key not
found
 *                            TRUE                    key
found, value stored
 *
 *
 * COMMENTS:         http keys are formatted either KEY=value& or
KEY=value\0.
 *                   This DLL formats TPC-C input fields in such a manner that
 *                   the keys can be extracted in the above manner.
 */

BOOL
GetKeyValueString(char *szIPtr, char *szKey,
                  char *szValue, int iSize)
{
  char *ptr;

  if( !GetKeyValuePtr( szIPtr, szKey, &ptr ))
    return FALSE;

  /* force zero termination of output string */
  iSize--;

  while( '\0' != *ptr && '&' != *ptr && iSize)
  {
    *szValue++ = *ptr++;
    iSize--;
  }
  *szValue = 0;
  return TRUE;
}

int
GetCallingThreadLimit( int *pCallingThreadLimit )
{
  int                   Status;
  int                   size;

  size = sizeof( *pCallingThreadLimit );
  Status = GetRegValue( INETINFO_CLASS, "PoolThreadLimit",
                        &size, (char *)pCallingThreadLimit );
  if( ERR_SUCCESS != Status )
  {
    return ERR_CANT_FIND_POOLTHREADLIMIT;
  }
}

#ifdef FFE_DEBUG
/* FUNCTION: void CheckMemory(void *param)
 *
 * PURPOSE:          This function loops calling _CrtCheckMemory()
 *
 * ARGUMENTS:
 *                   void     *param                  not used
 *
 * RETURNS:          nothing
 *
 * COMMENTS:
 */

unsigned __stdcall
CheckMemory(void *param)
{
  while (TRUE)
  {
    _ASSERTE( _CrtCheckMemory( ));
    Sleep( 1000 );
  }
  return 0;
}

#endif
```

Appendix B

addfile.sh

```
#!sh

$SEQLPLUS tpcc/tpcc <<!
  spool step5addfile_$1.log
  set echo on
  alter tablespace $1 add datafile '$2' size $3 reuse;
  set echo off
  spool off
  exit sql.sqlcode;
!
```

addtempfile.sh

```
#!sh

$SEQLPLUS tpcc/tpcc <<!
  spool step5addfile_$1.log
  set echo on
  alter tablespace $1 add tempfile '$2' size $3 reuse;
  set echo off
  spool off
  exit sql.sqlcode;
!
```

addtempts.sh

```
#!/bin/sh

$SEQLPLUS tpcc/tpcc <<!
  spool step5createts_$1.log
  set echo on
  drop tablespace $1 including contents;
  create temporary tablespace $1 tempfile '$2' size $3 reuse extent management local
uniform size $4;
  set echo off
  spool off
  exit sql.sqlcode;
!
```

addts.sh

```
#!/bin/sh

$SEQLPLUS tpcc/tpcc <<!
  spool step5createts_$1.log
  set echo on
  drop tablespace $1 including contents;
  create tablespace $1 datafile '$2' size $3 reuse extent management local uniform size
$4 nologging ;
  set echo off
  spool off
  exit sql.sqlcode;
!
```

addts_dict.sh

```
#!/bin/sh

$SEQLPLUS tpcc/tpcc <<!
  spool step5createts_$1.log
  set echo on
  drop tablespace $1 including contents;
  create tablespace $1 datafile '$2' size $3 reuse;
  set echo off
  spool off
  exit sql.sqlcode;
!
```

buffer_pool_off.sh

```
#!/bin/sh
./undml.sh
svrmgrl <<!
  connect tpcc/tpcc;
  alter cluster custcluster storage (buffer_pool default);
  alter table hist storage (buffer_pool default);
  alter cluster itemcluster storage (buffer_pool default);
  alter cluster stokcluster storage (buffer_pool default);
exit;
!
./dml.sh
```

c_stat.sql

```
rem
rem
rem=========================================================
=======+
rem       Copyright (c) 1997  Oracle Corp, Redwood Shores, CA       |
rem                         All Rights Reserved                     |
rem=========================================================
=======+
rem FILENAME
rem      cs_tpcc.sq
rem DESCRIPTION
rem      Create tables for saving TPC-C results.
rem=========================================================
=======
rem  Usage: sqlplus user/password @cs_tpcc.sql
rem spool cs_tpcc.log

connect tpcc/tpcc;
set echo on
alter user tpcc default tablespace system;
DROP TABLE tpcc_run_desc;
DROP TABLE tpcc_run_int;
DROP TABLE bench_run_int;
DROP TABLE tpcc_back_res;
DROP TABLE tpcc_user_res;
DROP TABLE bench_user_res;
DROP TABLE tpcc_tpm;
DROP TABLE tpcc_new_res;
DROP TABLE bench_new_res;
DROP TABLE tpcc_pay_res;
DROP TABLE bench_pay_res;
DROP TABLE tpcc_ord_res;
DROP TABLE bench_ord_res;
DROP TABLE tpcc_del_res;
DROP TABLE bench_del_res;
DROP TABLE tpcc_sto_res;
DROP TABLE bench_sto_res;

rem
```

```
rem  description of a run
rem
            CREATE TABLE tpcc_run_desc
            (
              run_name       VARCHAR2(20),
              rundate        DATE,
              time           NUMBER,
              rampup         NUMBER,
              rampdown       NUMBER,
              warehouses     NUMBER,
              customers      NUMBER,
              users          NUMBER,
              driver         VARCHAR2(40),
              commnt         VARCHAR2(80)
            );

rem
rem  throughput of new order transactions
rem
            CREATE TABLE tpcc_run_int
            (
              run_name       VARCHAR2(20),
              interval       NUMBER,
              interval_count  NUMBER,
              response_time   NUMBER,
              think_time      NUMBER
            );

rem
rem  throughput of new order transactions
rem
            CREATE TABLE bench_run_int
            (
              run_name       VARCHAR2(20),
              proc_no        NUMBER,
              interval       NUMBER,
              interval_count  NUMBER,
              response_time   NUMBER,
              think_time      NUMBER
            );

rem
rem  Results from delivery servers
rem
            CREATE TABLE tpcc_back_res
            (
              run_name       VARCHAR2(20),
              in_timing_int   NUMBER,
              fast           NUMBER,
              resp_time      NUMBER,
        retries        NUMBER
            );

rem
rem  Aggregate results for all generators.
rem  These results are from the measurement interval only.
rem  These results are used to calculate the TPS rate over
rem  the measurement interval.
rem
            CREATE TABLE tpcc_user_res
            (
              run_name       VARCHAR2(20),
              no_men         NUMBER,
              fast_men       NUMBER,
              in_flight_men   NUMBER,
        retry_men      NUMBER,
              min_time_men    NUMBER,
              max_time_men    NUMBER,
              sum_time_men    NUMBER,
        ninety_per_men  NUMBER,
        think_min_men   NUMBER,
        think_max_men   NUMBER,
              think_sum_men   NUMBER,
        key_min_men     NUMBER,
        key_max_men     NUMBER,
              key_sum_men     NUMBER,
              no_new          NUMBER,
              fast_new        NUMBER,
              in_flight_new   NUMBER,
        retry_new       NUMBER,
              min_time_new    NUMBER,
              max_time_new    NUMBER,
              sum_time_new    NUMBER,
        ninety_per_new  NUMBER,
        think_min_new   NUMBER,
        think_max_new   NUMBER,
              think_sum_new   NUMBER,
        key_min_new     NUMBER,
        key_max_new     NUMBER,
              key_sum_new     NUMBER,
              remote_new      NUMBER,
              rollback_new    NUMBER,
              sum_ol_new      NUMBER,
              remote_ol_new   NUMBER,
        allrollback_new NUMBER,
              no_pay          NUMBER,
              fast_pay        NUMBER,
              in_flight_pay   NUMBER,
        retry_pay       NUMBER,
              min_time_pay    NUMBER,
              max_time_pay    NUMBER,
              sum_time_pay    NUMBER,
        ninety_per_pay  NUMBER,
        think_min_pay   NUMBER,
        think_max_pay   NUMBER,
              think_sum_pay   NUMBER,
        key_min_pay     NUMBER,
        key_max_pay     NUMBER,
              key_sum_pay     NUMBER,
              remote_pay      NUMBER,
              bylast_pay      NUMBER,
              no_ord          NUMBER,
              fast_ord        NUMBER,
              in_flight_ord   NUMBER,
        retry_ord       NUMBER,
              min_time_ord    NUMBER,
              max_time_ord    NUMBER,
              sum_time_ord    NUMBER,
        ninety_per_ord  NUMBER,
        think_min_ord   NUMBER,
        think_max_ord   NUMBER,
              think_sum_ord   NUMBER,
        key_min_ord     NUMBER,
        key_max_ord     NUMBER,
              key_sum_ord     NUMBER,
              bylast_ord      NUMBER,
              no_del          NUMBER,
              fast_del        NUMBER,
              in_flight_del   NUMBER,
        retry_del       NUMBER,
              min_time_del    NUMBER,
              max_time_del    NUMBER,
              sum_time_del    NUMBER,
        ninety_per_del  NUMBER,
        think_min_del   NUMBER,
        think_max_del   NUMBER,
              think_sum_del   NUMBER,
        key_min_del     NUMBER,
        key_max_del     NUMBER,
              key_sum_del     NUMBER,
              no_sto          NUMBER,
              fast_sto        NUMBER,
              in_flight_sto   NUMBER,
        retry_sto       NUMBER,
              min_time_sto    NUMBER,
              max_time_sto    NUMBER,
              sum_time_sto    NUMBER,
        ninety_per_sto  NUMBER,
        think_min_sto   NUMBER,
        think_max_sto   NUMBER,
              think_sum_sto   NUMBER,
        key_min_sto     NUMBER,
        key_max_sto     NUMBER,
              key_sum_sto     NUMBER,
              cpu_time        NUMBER,
        deadlocks       NUMBER
            );

rem
rem  Results from individual generators.
```

```
rem  These results are from the measurement interval only.
rem  These results are used to calculate the TPS rate over
rem  the measurement interval.
rem
                CREATE TABLE bench_user_res
                (
                   run_name        VARCHAR2(20),
                   audit_str       VARCHAR2(10),
                   proc_no         NUMBER,
                   hid             NUMBER,
                   no_men          NUMBER,
                   fast_men        NUMBER,
                   in_flight_men   NUMBER,
        retry_men       NUMBER,
                   min_time_men    NUMBER,
                   max_time_men    NUMBER,
                   sum_time_men    NUMBER,
        ninety_per_men  NUMBER,
        think_min_men   NUMBER,
        think_max_men   NUMBER,
                   think_sum_men   NUMBER,
        key_min_men     NUMBER,
        key_max_men     NUMBER,
                   key_sum_men     NUMBER,
                   no_new          NUMBER,
                   fast_new        NUMBER,
                   in_flight_new   NUMBER,
        retry_new       NUMBER,
                   min_time_new    NUMBER,
                   max_time_new    NUMBER,
                   sum_time_new    NUMBER,
        ninety_per_new  NUMBER,
        think_min_new   NUMBER,
        think_max_new   NUMBER,
                   think_sum_new   NUMBER,
        key_min_new     NUMBER,
        key_max_new     NUMBER,
                   key_sum_new     NUMBER,
                   remote_new      NUMBER,
                   rollback_new    NUMBER,
                   sum_ol_new      NUMBER,
                   remote_ol_new   NUMBER,
        allrollback_new NUMBER,
                   no_pay          NUMBER,
                   fast_pay        NUMBER,
                   in_flight_pay   NUMBER,
        retry_pay       NUMBER,
                   min_time_pay    NUMBER,
                   max_time_pay    NUMBER,
                   sum_time_pay    NUMBER,
        ninety_per_pay  NUMBER,
        think_min_pay   NUMBER,
        think_max_pay   NUMBER,
                   think_sum_pay   NUMBER,
        key_min_pay     NUMBER,
        key_max_pay     NUMBER,
                   key_sum_pay     NUMBER,
                   remote_pay      NUMBER,
                   bylast_pay      NUMBER,
                   no_ord          NUMBER,
                   fast_ord        NUMBER,
                   in_flight_ord   NUMBER,
        retry_ord       NUMBER,
                   min_time_ord    NUMBER,
                   max_time_ord    NUMBER,
                   sum_time_ord    NUMBER,
        ninety_per_ord  NUMBER,
        think_min_ord   NUMBER,
        think_max_ord   NUMBER,
                   think_sum_ord   NUMBER,
        key_min_ord     NUMBER,
        key_max_ord     NUMBER,
                   key_sum_ord     NUMBER,
                   bylast_ord      NUMBER,
                   no_del          NUMBER,
                   fast_del        NUMBER,
                   in_flight_del   NUMBER,
        retry_del       NUMBER,
                   min_time_del    NUMBER,
                   max_time_del    NUMBER,
                   sum_time_del    NUMBER,
        ninety_per_del  NUMBER,
        think_min_del   NUMBER,
        think_max_del   NUMBER,
                   think_sum_del   NUMBER,
        key_min_del     NUMBER,
        key_max_del     NUMBER,
                   key_sum_del     NUMBER,
                   no_sto          NUMBER,
                   fast_sto        NUMBER,
                   in_flight_sto   NUMBER,
        retry_sto       NUMBER,
                   min_time_sto    NUMBER,
                   max_time_sto    NUMBER,
                   sum_time_sto    NUMBER,
        ninety_per_sto  NUMBER,
        think_min_sto   NUMBER,
        think_max_sto   NUMBER,
                   think_sum_sto   NUMBER,
        key_min_sto     NUMBER,
        key_max_sto     NUMBER,
                   key_sum_sto     NUMBER,
                   cpu_time        NUMBER,
        deadlocks       NUMBER
                );

rem
rem  Aggregate results for generators on each host.
rem  These results are from the measurement interval only.
rem  These results are used to calculate the TPM rate over
rem  the measurement interval.
rem
                CREATE TABLE tpcc_tpm
                (
                   run_name        VARCHAR2(20),
                   hid             NUMBER,
                   no_new          NUMBER
                );

rem
rem  Aggregate results for new order transactions.
rem  These results are from the measurement interval only.
rem
                CREATE TABLE tpcc_new_res
                (
                   run_name        VARCHAR2(20),
                   rep1            NUMBER,
                   rep2            NUMBER,
                   rep3            NUMBER,
                   rep4            NUMBER,
                   rep5            NUMBER,
                   rep6            NUMBER,
                   rep7            NUMBER,
                   rep8            NUMBER,
                   rep9            NUMBER,
                   rep10           NUMBER,
                   rep11           NUMBER,
                   rep12           NUMBER,
                   rep13           NUMBER,
                   rep14           NUMBER,
                   rep15           NUMBER,
                   rep16           NUMBER,
                   rep17           NUMBER,
                   rep18           NUMBER,
                   rep19           NUMBER,
                   rep20           NUMBER,
                   rep21           NUMBER,
                   rep22           NUMBER,
                   rep23           NUMBER,
                   rep24           NUMBER,
                   rep25           NUMBER,
                   rep26           NUMBER,
                   rep27           NUMBER,
                   rep28           NUMBER,
                   rep29           NUMBER,
                   rep30           NUMBER,
                   rep31           NUMBER,
                   rep32           NUMBER,
                   rep33           NUMBER,
                   rep34           NUMBER,
```

```
    rep35       NUMBER,
    rep36       NUMBER,
    rep37       NUMBER,
    rep38       NUMBER,
    rep39       NUMBER,
    rep40       NUMBER,
    rep41       NUMBER,
    rep42       NUMBER,
    rep43       NUMBER,
    rep44       NUMBER,
    rep45       NUMBER,
    rep46       NUMBER,
    rep47       NUMBER,
    rep48       NUMBER,
    rep49       NUMBER,
    rep50       NUMBER,
    rep51       NUMBER,
    rep52       NUMBER,
    rep53       NUMBER,
    rep54       NUMBER,
    rep55       NUMBER,
    rep56       NUMBER,
    rep57       NUMBER,
    rep58       NUMBER,
    rep59       NUMBER,
    rep60       NUMBER,
    rep61       NUMBER,
    rep62       NUMBER,
    rep63       NUMBER,
    rep64       NUMBER,
    rep65       NUMBER,
    rep66       NUMBER,
    rep67       NUMBER,
    rep68       NUMBER,
    rep69       NUMBER,
    rep70       NUMBER,
    rep71       NUMBER,
    rep72       NUMBER,
    rep73       NUMBER,
    rep74       NUMBER,
    rep75       NUMBER,
    rep76       NUMBER,
    rep77       NUMBER,
    rep78       NUMBER,
    rep79       NUMBER,
    rep80       NUMBER,
    rep81       NUMBER,
    rep82       NUMBER,
    rep83       NUMBER,
    rep84       NUMBER,
    rep85       NUMBER,
    rep86       NUMBER,
    rep87       NUMBER,
    rep88       NUMBER,
    rep89       NUMBER,
    rep90       NUMBER,
    rep91       NUMBER,
    rep92       NUMBER,
    rep93       NUMBER,
    rep94       NUMBER,
    rep95       NUMBER,
    rep96       NUMBER,
    rep97       NUMBER,
    rep98       NUMBER,
    rep99       NUMBER,
    rep100      NUMBER,
    thk1        NUMBER,
    thk2        NUMBER,
    thk3        NUMBER,
    thk4        NUMBER,
    thk5        NUMBER,
    thk6        NUMBER,
    thk7        NUMBER,
    thk8        NUMBER,
    thk9        NUMBER,
    thk10       NUMBER,
    thk11       NUMBER,
    thk12       NUMBER,
    thk13       NUMBER,
    thk14       NUMBER,
    thk15       NUMBER,
    thk16       NUMBER,
    thk17       NUMBER,
    thk18       NUMBER,
    thk19       NUMBER,
    thk20       NUMBER,
    thk21       NUMBER,
    thk22       NUMBER,
    thk23       NUMBER,
    thk24       NUMBER,
    thk25       NUMBER,
    key1        NUMBER,
    key2        NUMBER,
    key3        NUMBER,
    key4        NUMBER,
    key5        NUMBER,
    key6        NUMBER,
    key7        NUMBER,
    key8        NUMBER,
    key9        NUMBER,
    key10       NUMBER
);

rem
rem  Results for new order transactions.
rem  These results are from the measurement interval only.
rem
CREATE TABLE bench_new_res
(
    run_name    VARCHAR2(20),
    audit_str   VARCHAR2(10),
    proc_no     NUMBER,
    rep1        NUMBER,
    rep2        NUMBER,
    rep3        NUMBER,
    rep4        NUMBER,
    rep5        NUMBER,
    rep6        NUMBER,
    rep7        NUMBER,
    rep8        NUMBER,
    rep9        NUMBER,
    rep10       NUMBER,
    rep11       NUMBER,
    rep12       NUMBER,
    rep13       NUMBER,
    rep14       NUMBER,
    rep15       NUMBER,
    rep16       NUMBER,
    rep17       NUMBER,
    rep18       NUMBER,
    rep19       NUMBER,
    rep20       NUMBER,
    rep21       NUMBER,
    rep22       NUMBER,
    rep23       NUMBER,
    rep24       NUMBER,
    rep25       NUMBER,
    rep26       NUMBER,
    rep27       NUMBER,
    rep28       NUMBER,
    rep29       NUMBER,
    rep30       NUMBER,
    rep31       NUMBER,
    rep32       NUMBER,
    rep33       NUMBER,
    rep34       NUMBER,
    rep35       NUMBER,
    rep36       NUMBER,
    rep37       NUMBER,
    rep38       NUMBER,
    rep39       NUMBER,
    rep40       NUMBER,
    rep41       NUMBER,
    rep42       NUMBER,
    rep43       NUMBER,
    rep44       NUMBER,
    rep45       NUMBER,
    rep46       NUMBER,
    rep47       NUMBER,
    rep48       NUMBER,
```

```
    rep49       NUMBER,
    rep50       NUMBER,
    rep51       NUMBER,
    rep52       NUMBER,
    rep53       NUMBER,
    rep54       NUMBER,
    rep55       NUMBER,
    rep56       NUMBER,
    rep57       NUMBER,
    rep58       NUMBER,
    rep59       NUMBER,
    rep60       NUMBER,
    rep61       NUMBER,
    rep62       NUMBER,
    rep63       NUMBER,
    rep64       NUMBER,
    rep65       NUMBER,
    rep66       NUMBER,
    rep67       NUMBER,
    rep68       NUMBER,
    rep69       NUMBER,
    rep70       NUMBER,
    rep71       NUMBER,
    rep72       NUMBER,
    rep73       NUMBER,
    rep74       NUMBER,
    rep75       NUMBER,
    rep76       NUMBER,
    rep77       NUMBER,
    rep78       NUMBER,
    rep79       NUMBER,
    rep80       NUMBER,
    rep81       NUMBER,
    rep82       NUMBER,
    rep83       NUMBER,
    rep84       NUMBER,
    rep85       NUMBER,
    rep86       NUMBER,
    rep87       NUMBER,
    rep88       NUMBER,
    rep89       NUMBER,
    rep90       NUMBER,
    rep91       NUMBER,
    rep92       NUMBER,
    rep93       NUMBER,
    rep94       NUMBER,
    rep95       NUMBER,
    rep96       NUMBER,
    rep97       NUMBER,
    rep98       NUMBER,
    rep99       NUMBER,
    rep100      NUMBER,
    thk1        NUMBER,
    thk2        NUMBER,
    thk3        NUMBER,
    thk4        NUMBER,
    thk5        NUMBER,
    thk6        NUMBER,
    thk7        NUMBER,
    thk8        NUMBER,
    thk9        NUMBER,
    thk10       NUMBER,
    thk11       NUMBER,
    thk12       NUMBER,
    thk13       NUMBER,
    thk14       NUMBER,
    thk15       NUMBER,
    thk16       NUMBER,
    thk17       NUMBER,
    thk18       NUMBER,
    thk19       NUMBER,
    thk20       NUMBER,
    thk21       NUMBER,
    thk22       NUMBER,
    thk23       NUMBER,
    thk24       NUMBER,
    thk25       NUMBER,
    key1        NUMBER,
    key2        NUMBER,
    key3        NUMBER,
    key4        NUMBER,
    key5        NUMBER,
    key6        NUMBER,
    key7        NUMBER,
    key8        NUMBER,
    key9        NUMBER,
    key10       NUMBER
);

rem
rem  Aggregate results for payment transactions.
rem  These results are from the measurement interval only.
rem
    CREATE TABLE tpcc_pay_res
    (
    run_name    VARCHAR2(20),
    rep1        NUMBER,
    rep2        NUMBER,
    rep3        NUMBER,
    rep4        NUMBER,
    rep5        NUMBER,
    rep6        NUMBER,
    rep7        NUMBER,
    rep8        NUMBER,
    rep9        NUMBER,
    rep10       NUMBER,
    rep11       NUMBER,
    rep12       NUMBER,
    rep13       NUMBER,
    rep14       NUMBER,
    rep15       NUMBER,
    rep16       NUMBER,
    rep17       NUMBER,
    rep18       NUMBER,
    rep19       NUMBER,
    rep20       NUMBER,
    rep21       NUMBER,
    rep22       NUMBER,
    rep23       NUMBER,
    rep24       NUMBER,
    rep25       NUMBER,
    rep26       NUMBER,
    rep27       NUMBER,
    rep28       NUMBER,
    rep29       NUMBER,
    rep30       NUMBER,
    rep31       NUMBER,
    rep32       NUMBER,
    rep33       NUMBER,
    rep34       NUMBER,
    rep35       NUMBER,
    rep36       NUMBER,
    rep37       NUMBER,
    rep38       NUMBER,
    rep39       NUMBER,
    rep40       NUMBER,
    rep41       NUMBER,
    rep42       NUMBER,
    rep43       NUMBER,
    rep44       NUMBER,
    rep45       NUMBER,
    rep46       NUMBER,
    rep47       NUMBER,
    rep48       NUMBER,
    rep49       NUMBER,
    rep50       NUMBER,
    rep51       NUMBER,
    rep52       NUMBER,
    rep53       NUMBER,
    rep54       NUMBER,
    rep55       NUMBER,
    rep56       NUMBER,
    rep57       NUMBER,
    rep58       NUMBER,
    rep59       NUMBER,
    rep60       NUMBER,
    rep61       NUMBER,
    rep62       NUMBER,
    rep63       NUMBER,
    rep64       NUMBER,
```

```
    rep65       NUMBER,
    rep66       NUMBER,
    rep67       NUMBER,
    rep68       NUMBER,
    rep69       NUMBER,
    rep70       NUMBER,
    rep71       NUMBER,
    rep72       NUMBER,
    rep73       NUMBER,
    rep74       NUMBER,
    rep75       NUMBER,
    rep76       NUMBER,
    rep77       NUMBER,
    rep78       NUMBER,
    rep79       NUMBER,
    rep80       NUMBER,
    rep81       NUMBER,
    rep82       NUMBER,
    rep83       NUMBER,
    rep84       NUMBER,
    rep85       NUMBER,
    rep86       NUMBER,
    rep87       NUMBER,
    rep88       NUMBER,
    rep89       NUMBER,
    rep90       NUMBER,
    rep91       NUMBER,
    rep92       NUMBER,
    rep93       NUMBER,
    rep94       NUMBER,
    rep95       NUMBER,
    rep96       NUMBER,
    rep97       NUMBER,
    rep98       NUMBER,
    rep99       NUMBER,
    rep100      NUMBER,
    thk1        NUMBER,
    thk2        NUMBER,
    thk3        NUMBER,
    thk4        NUMBER,
    thk5        NUMBER,
    thk6        NUMBER,
    thk7        NUMBER,
    thk8        NUMBER,
    thk9        NUMBER,
    thk10       NUMBER,
    thk11       NUMBER,
    thk12       NUMBER,
    thk13       NUMBER,
    thk14       NUMBER,
    thk15       NUMBER,
    thk16       NUMBER,
    thk17       NUMBER,
    thk18       NUMBER,
    thk19       NUMBER,
    thk20       NUMBER,
    thk21       NUMBER,
    thk22       NUMBER,
    thk23       NUMBER,
    thk24       NUMBER,
    thk25       NUMBER,
    key1        NUMBER,
    key2        NUMBER,
    key3        NUMBER,
    key4        NUMBER,
    key5        NUMBER,
    key6        NUMBER,
    key7        NUMBER,
    key8        NUMBER,
    key9        NUMBER,
    key10       NUMBER
  );

rem
rem  Results for payment transactions.
rem  These results are from the measurement interval only.
rem
  CREATE TABLE bench_pay_res
  (
    run_name    VARCHAR2(20),
    audit_str   VARCHAR2(10),
    proc_no     NUMBER,
    rep1        NUMBER,
    rep2        NUMBER,
    rep3        NUMBER,
    rep4        NUMBER,
    rep5        NUMBER,
    rep6        NUMBER,
    rep7        NUMBER,
    rep8        NUMBER,
    rep9        NUMBER,
    rep10       NUMBER,
    rep11       NUMBER,
    rep12       NUMBER,
    rep13       NUMBER,
    rep14       NUMBER,
    rep15       NUMBER,
    rep16       NUMBER,
    rep17       NUMBER,
    rep18       NUMBER,
    rep19       NUMBER,
    rep20       NUMBER,
    rep21       NUMBER,
    rep22       NUMBER,
    rep23       NUMBER,
    rep24       NUMBER,
    rep25       NUMBER,
    rep26       NUMBER,
    rep27       NUMBER,
    rep28       NUMBER,
    rep29       NUMBER,
    rep30       NUMBER,
    rep31       NUMBER,
    rep32       NUMBER,
    rep33       NUMBER,
    rep34       NUMBER,
    rep35       NUMBER,
    rep36       NUMBER,
    rep37       NUMBER,
    rep38       NUMBER,
    rep39       NUMBER,
    rep40       NUMBER,
    rep41       NUMBER,
    rep42       NUMBER,
    rep43       NUMBER,
    rep44       NUMBER,
    rep45       NUMBER,
    rep46       NUMBER,
    rep47       NUMBER,
    rep48       NUMBER,
    rep49       NUMBER,
    rep50       NUMBER,
    rep51       NUMBER,
    rep52       NUMBER,
    rep53       NUMBER,
    rep54       NUMBER,
    rep55       NUMBER,
    rep56       NUMBER,
    rep57       NUMBER,
    rep58       NUMBER,
    rep59       NUMBER,
    rep60       NUMBER,
    rep61       NUMBER,
    rep62       NUMBER,
    rep63       NUMBER,
    rep64       NUMBER,
    rep65       NUMBER,
    rep66       NUMBER,
    rep67       NUMBER,
    rep68       NUMBER,
    rep69       NUMBER,
    rep70       NUMBER,
    rep71       NUMBER,
    rep72       NUMBER,
    rep73       NUMBER,
    rep74       NUMBER,
    rep75       NUMBER,
    rep76       NUMBER,
    rep77       NUMBER,
    rep78       NUMBER,
```

```
    rep79       NUMBER,
    rep80       NUMBER,
    rep81       NUMBER,
    rep82       NUMBER,
    rep83       NUMBER,
    rep84       NUMBER,
    rep85       NUMBER,
    rep86       NUMBER,
    rep87       NUMBER,
    rep88       NUMBER,
    rep89       NUMBER,
    rep90       NUMBER,
    rep91       NUMBER,
    rep92       NUMBER,
    rep93       NUMBER,
    rep94       NUMBER,
    rep95       NUMBER,
    rep96       NUMBER,
    rep97       NUMBER,
    rep98       NUMBER,
    rep99       NUMBER,
    rep100      NUMBER,
    thk1        NUMBER,
    thk2        NUMBER,
    thk3        NUMBER,
    thk4        NUMBER,
    thk5        NUMBER,
    thk6        NUMBER,
    thk7        NUMBER,
    thk8        NUMBER,
    thk9        NUMBER,
    thk10       NUMBER,
    thk11       NUMBER,
    thk12       NUMBER,
    thk13       NUMBER,
    thk14       NUMBER,
    thk15       NUMBER,
    thk16       NUMBER,
    thk17       NUMBER,
    thk18       NUMBER,
    thk19       NUMBER,
    thk20       NUMBER,
    thk21       NUMBER,
    thk22       NUMBER,
    thk23       NUMBER,
    thk24       NUMBER,
    thk25       NUMBER,
    key1        NUMBER,
    key2        NUMBER,
    key3        NUMBER,
    key4        NUMBER,
    key5        NUMBER,
    key6        NUMBER,
    key7        NUMBER,
    key8        NUMBER,
    key9        NUMBER,
    key10       NUMBER
);

rem
rem Aggregate results for order status transactions.
rem These results are from the measurement interval only.
rem
CREATE TABLE tpcc_ord_res
(
    run_name    VARCHAR2(20),
    rep1        NUMBER,
    rep2        NUMBER,
    rep3        NUMBER,
    rep4        NUMBER,
    rep5        NUMBER,
    rep6        NUMBER,
    rep7        NUMBER,
    rep8        NUMBER,
    rep9        NUMBER,
    rep10       NUMBER,
    rep11       NUMBER,
    rep12       NUMBER,
    rep13       NUMBER,
    rep14       NUMBER,
    rep15       NUMBER,
    rep16       NUMBER,
    rep17       NUMBER,
    rep18       NUMBER,
    rep19       NUMBER,
    rep20       NUMBER,
    rep21       NUMBER,
    rep22       NUMBER,
    rep23       NUMBER,
    rep24       NUMBER,
    rep25       NUMBER,
    rep26       NUMBER,
    rep27       NUMBER,
    rep28       NUMBER,
    rep29       NUMBER,
    rep30       NUMBER,
    rep31       NUMBER,
    rep32       NUMBER,
    rep33       NUMBER,
    rep34       NUMBER,
    rep35       NUMBER,
    rep36       NUMBER,
    rep37       NUMBER,
    rep38       NUMBER,
    rep39       NUMBER,
    rep40       NUMBER,
    rep41       NUMBER,
    rep42       NUMBER,
    rep43       NUMBER,
    rep44       NUMBER,
    rep45       NUMBER,
    rep46       NUMBER,
    rep47       NUMBER,
    rep48       NUMBER,
    rep49       NUMBER,
    rep50       NUMBER,
    rep51       NUMBER,
    rep52       NUMBER,
    rep53       NUMBER,
    rep54       NUMBER,
    rep55       NUMBER,
    rep56       NUMBER,
    rep57       NUMBER,
    rep58       NUMBER,
    rep59       NUMBER,
    rep60       NUMBER,
    rep61       NUMBER,
    rep62       NUMBER,
    rep63       NUMBER,
    rep64       NUMBER,
    rep65       NUMBER,
    rep66       NUMBER,
    rep67       NUMBER,
    rep68       NUMBER,
    rep69       NUMBER,
    rep70       NUMBER,
    rep71       NUMBER,
    rep72       NUMBER,
    rep73       NUMBER,
    rep74       NUMBER,
    rep75       NUMBER,
    rep76       NUMBER,
    rep77       NUMBER,
    rep78       NUMBER,
    rep79       NUMBER,
    rep80       NUMBER,
    rep81       NUMBER,
    rep82       NUMBER,
    rep83       NUMBER,
    rep84       NUMBER,
    rep85       NUMBER,
    rep86       NUMBER,
    rep87       NUMBER,
    rep88       NUMBER,
    rep89       NUMBER,
    rep90       NUMBER,
    rep91       NUMBER,
    rep92       NUMBER,
    rep93       NUMBER,
    rep94       NUMBER,
```

```
        rep95       NUMBER,
        rep96       NUMBER,
        rep97       NUMBER,
        rep98       NUMBER,
        rep99       NUMBER,
        rep100      NUMBER,
        thk1        NUMBER,
        thk2        NUMBER,
        thk3        NUMBER,
        thk4        NUMBER,
        thk5        NUMBER,
        thk6        NUMBER,
        thk7        NUMBER,
        thk8        NUMBER,
        thk9        NUMBER,
        thk10       NUMBER,
        thk11       NUMBER,
        thk12       NUMBER,
        thk13       NUMBER,
        thk14       NUMBER,
        thk15       NUMBER,
        thk16       NUMBER,
        thk17       NUMBER,
        thk18       NUMBER,
        thk19       NUMBER,
        thk20       NUMBER,
        thk21       NUMBER,
        thk22       NUMBER,
        thk23       NUMBER,
        thk24       NUMBER,
        thk25       NUMBER,
        key1        NUMBER,
        key2        NUMBER,
        key3        NUMBER,
        key4        NUMBER,
        key5        NUMBER,
        key6        NUMBER,
        key7        NUMBER,
        key8        NUMBER,
        key9        NUMBER,
        key10       NUMBER
    );

rem
rem  Results for order status transactions.
rem  These results are from the measurement interval only.
rem
    CREATE TABLE bench_ord_res
    (
      run_name      VARCHAR2(20),
      audit_str     VARCHAR2(10),
      proc_no       NUMBER,
      rep1          NUMBER,
      rep2          NUMBER,
      rep3          NUMBER,
      rep4          NUMBER,
      rep5          NUMBER,
      rep6          NUMBER,
      rep7          NUMBER,
      rep8          NUMBER,
      rep9          NUMBER,
      rep10         NUMBER,
      rep11         NUMBER,
      rep12         NUMBER,
      rep13         NUMBER,
      rep14         NUMBER,
      rep15         NUMBER,
      rep16         NUMBER,
      rep17         NUMBER,
      rep18         NUMBER,
      rep19         NUMBER,
      rep20         NUMBER,
      rep21         NUMBER,
      rep22         NUMBER,
      rep23         NUMBER,
      rep24         NUMBER,
      rep25         NUMBER,
      rep26         NUMBER,
      rep27         NUMBER,
      rep28         NUMBER,
      rep29         NUMBER,
      rep30         NUMBER,
      rep31         NUMBER,
      rep32         NUMBER,
      rep33         NUMBER,
      rep34         NUMBER,
      rep35         NUMBER,
      rep36         NUMBER,
      rep37         NUMBER,
      rep38         NUMBER,
      rep39         NUMBER,
      rep40         NUMBER,
      rep41         NUMBER,
      rep42         NUMBER,
      rep43         NUMBER,
      rep44         NUMBER,
      rep45         NUMBER,
      rep46         NUMBER,
      rep47         NUMBER,
      rep48         NUMBER,
      rep49         NUMBER,
      rep50         NUMBER,
      rep51         NUMBER,
      rep52         NUMBER,
      rep53         NUMBER,
      rep54         NUMBER,
      rep55         NUMBER,
      rep56         NUMBER,
      rep57         NUMBER,
      rep58         NUMBER,
      rep59         NUMBER,
      rep60         NUMBER,
      rep61         NUMBER,
      rep62         NUMBER,
      rep63         NUMBER,
      rep64         NUMBER,
      rep65         NUMBER,
      rep66         NUMBER,
      rep67         NUMBER,
      rep68         NUMBER,
      rep69         NUMBER,
      rep70         NUMBER,
      rep71         NUMBER,
      rep72         NUMBER,
      rep73         NUMBER,
      rep74         NUMBER,
      rep75         NUMBER,
      rep76         NUMBER,
      rep77         NUMBER,
      rep78         NUMBER,
      rep79         NUMBER,
      rep80         NUMBER,
      rep81         NUMBER,
      rep82         NUMBER,
      rep83         NUMBER,
      rep84         NUMBER,
      rep85         NUMBER,
      rep86         NUMBER,
      rep87         NUMBER,
      rep88         NUMBER,
      rep89         NUMBER,
      rep90         NUMBER,
      rep91         NUMBER,
      rep92         NUMBER,
      rep93         NUMBER,
      rep94         NUMBER,
      rep95         NUMBER,
      rep96         NUMBER,
      rep97         NUMBER,
      rep98         NUMBER,
      rep99         NUMBER,
      rep100        NUMBER,
      thk1          NUMBER,
      thk2          NUMBER,
      thk3          NUMBER,
      thk4          NUMBER,
      thk5          NUMBER,
      thk6          NUMBER,
      thk7          NUMBER,
      thk8          NUMBER,
```

```
    thk9        NUMBER,
    thk10       NUMBER,
    thk11       NUMBER,
    thk12       NUMBER,
    thk13       NUMBER,
    thk14       NUMBER,
    thk15       NUMBER,
    thk16       NUMBER,
    thk17       NUMBER,
    thk18       NUMBER,
    thk19       NUMBER,
    thk20       NUMBER,
    thk21       NUMBER,
    thk22       NUMBER,
    thk23       NUMBER,
    thk24       NUMBER,
    thk25       NUMBER,
    key1        NUMBER,
    key2        NUMBER,
    key3        NUMBER,
    key4        NUMBER,
    key5        NUMBER,
    key6        NUMBER,
    key7        NUMBER,
    key8        NUMBER,
    key9        NUMBER,
    key10       NUMBER
  );

rem
rem  Aggregate results for delivery transactions.
rem  These results are from the measurement interval only.
rem
  CREATE TABLE tpcc_del_res
  (
    run_name    VARCHAR2(20),
    rep1        NUMBER,
    rep2        NUMBER,
    rep3        NUMBER,
    rep4        NUMBER,
    rep5        NUMBER,
    rep6        NUMBER,
    rep7        NUMBER,
    rep8        NUMBER,
    rep9        NUMBER,
    rep10       NUMBER,
    rep11       NUMBER,
    rep12       NUMBER,
    rep13       NUMBER,
    rep14       NUMBER,
    rep15       NUMBER,
    rep16       NUMBER,
    rep17       NUMBER,
    rep18       NUMBER,
    rep19       NUMBER,
    rep20       NUMBER,
    rep21       NUMBER,
    rep22       NUMBER,
    rep23       NUMBER,
    rep24       NUMBER,
    rep25       NUMBER,
    rep26       NUMBER,
    rep27       NUMBER,
    rep28       NUMBER,
    rep29       NUMBER,
    rep30       NUMBER,
    rep31       NUMBER,
    rep32       NUMBER,
    rep33       NUMBER,
    rep34       NUMBER,
    rep35       NUMBER,
    rep36       NUMBER,
    rep37       NUMBER,
    rep38       NUMBER,
    rep39       NUMBER,
    rep40       NUMBER,
    rep41       NUMBER,
    rep42       NUMBER,
    rep43       NUMBER,
    rep44       NUMBER,
    rep45       NUMBER,
    rep46       NUMBER,
    rep47       NUMBER,
    rep48       NUMBER,
    rep49       NUMBER,
    rep50       NUMBER,
    rep51       NUMBER,
    rep52       NUMBER,
    rep53       NUMBER,
    rep54       NUMBER,
    rep55       NUMBER,
    rep56       NUMBER,
    rep57       NUMBER,
    rep58       NUMBER,
    rep59       NUMBER,
    rep60       NUMBER,
    rep61       NUMBER,
    rep62       NUMBER,
    rep63       NUMBER,
    rep64       NUMBER,
    rep65       NUMBER,
    rep66       NUMBER,
    rep67       NUMBER,
    rep68       NUMBER,
    rep69       NUMBER,
    rep70       NUMBER,
    rep71       NUMBER,
    rep72       NUMBER,
    rep73       NUMBER,
    rep74       NUMBER,
    rep75       NUMBER,
    rep76       NUMBER,
    rep77       NUMBER,
    rep78       NUMBER,
    rep79       NUMBER,
    rep80       NUMBER,
    rep81       NUMBER,
    rep82       NUMBER,
    rep83       NUMBER,
    rep84       NUMBER,
    rep85       NUMBER,
    rep86       NUMBER,
    rep87       NUMBER,
    rep88       NUMBER,
    rep89       NUMBER,
    rep90       NUMBER,
    rep91       NUMBER,
    rep92       NUMBER,
    rep93       NUMBER,
    rep94       NUMBER,
    rep95       NUMBER,
    rep96       NUMBER,
    rep97       NUMBER,
    rep98       NUMBER,
    rep99       NUMBER,
    rep100      NUMBER,
    thk1        NUMBER,
    thk2        NUMBER,
    thk3        NUMBER,
    thk4        NUMBER,
    thk5        NUMBER,
    thk6        NUMBER,
    thk7        NUMBER,
    thk8        NUMBER,
    thk9        NUMBER,
    thk10       NUMBER,
    thk11       NUMBER,
    thk12       NUMBER,
    thk13       NUMBER,
    thk14       NUMBER,
    thk15       NUMBER,
    thk16       NUMBER,
    thk17       NUMBER,
    thk18       NUMBER,
    thk19       NUMBER,
    thk20       NUMBER,
    thk21       NUMBER,
    thk22       NUMBER,
    thk23       NUMBER,
    thk24       NUMBER,
```

```
    thk25       NUMBER,
    key1        NUMBER,
    key2        NUMBER,
    key3        NUMBER,
    key4        NUMBER,
    key5        NUMBER,
    key6        NUMBER,
    key7        NUMBER,
    key8        NUMBER,
    key9        NUMBER,
    key10       NUMBER
  );

rem
rem  Results for delivery transactions.
rem  These results are from the measurement interval only.
rem
  CREATE TABLE bench_del_res
  (
    run_name      VARCHAR2(20),
    audit_str     VARCHAR2(10),
    proc_no       NUMBER,
    rep1          NUMBER,
    rep2          NUMBER,
    rep3          NUMBER,
    rep4          NUMBER,
    rep5          NUMBER,
    rep6          NUMBER,
    rep7          NUMBER,
    rep8          NUMBER,
    rep9          NUMBER,
    rep10         NUMBER,
    rep11         NUMBER,
    rep12         NUMBER,
    rep13         NUMBER,
    rep14         NUMBER,
    rep15         NUMBER,
    rep16         NUMBER,
    rep17         NUMBER,
    rep18         NUMBER,
    rep19         NUMBER,
    rep20         NUMBER,
    rep21         NUMBER,
    rep22         NUMBER,
    rep23         NUMBER,
    rep24         NUMBER,
    rep25         NUMBER,
    rep26         NUMBER,
    rep27         NUMBER,
    rep28         NUMBER,
    rep29         NUMBER,
    rep30         NUMBER,
    rep31         NUMBER,
    rep32         NUMBER,
    rep33         NUMBER,
    rep34         NUMBER,
    rep35         NUMBER,
    rep36         NUMBER,
    rep37         NUMBER,
    rep38         NUMBER,
    rep39         NUMBER,
    rep40         NUMBER,
    rep41         NUMBER,
    rep42         NUMBER,
    rep43         NUMBER,
    rep44         NUMBER,
    rep45         NUMBER,
    rep46         NUMBER,
    rep47         NUMBER,
    rep48         NUMBER,
    rep49         NUMBER,
    rep50         NUMBER,
    rep51         NUMBER,
    rep52         NUMBER,
    rep53         NUMBER,
    rep54         NUMBER,
    rep55         NUMBER,
    rep56         NUMBER,
    rep57         NUMBER,
    rep58         NUMBER,
    rep59         NUMBER,
    rep60         NUMBER,
    rep61         NUMBER,
    rep62         NUMBER,
    rep63         NUMBER,
    rep64         NUMBER,
    rep65         NUMBER,
    rep66         NUMBER,
    rep67         NUMBER,
    rep68         NUMBER,
    rep69         NUMBER,
    rep70         NUMBER,
    rep71         NUMBER,
    rep72         NUMBER,
    rep73         NUMBER,
    rep74         NUMBER,
    rep75         NUMBER,
    rep76         NUMBER,
    rep77         NUMBER,
    rep78         NUMBER,
    rep79         NUMBER,
    rep80         NUMBER,
    rep81         NUMBER,
    rep82         NUMBER,
    rep83         NUMBER,
    rep84         NUMBER,
    rep85         NUMBER,
    rep86         NUMBER,
    rep87         NUMBER,
    rep88         NUMBER,
    rep89         NUMBER,
    rep90         NUMBER,
    rep91         NUMBER,
    rep92         NUMBER,
    rep93         NUMBER,
    rep94         NUMBER,
    rep95         NUMBER,
    rep96         NUMBER,
    rep97         NUMBER,
    rep98         NUMBER,
    rep99         NUMBER,
    rep100        NUMBER,
    thk1          NUMBER,
    thk2          NUMBER,
    thk3          NUMBER,
    thk4          NUMBER,
    thk5          NUMBER,
    thk6          NUMBER,
    thk7          NUMBER,
    thk8          NUMBER,
    thk9          NUMBER,
    thk10         NUMBER,
    thk11         NUMBER,
    thk12         NUMBER,
    thk13         NUMBER,
    thk14         NUMBER,
    thk15         NUMBER,
    thk16         NUMBER,
    thk17         NUMBER,
    thk18         NUMBER,
    thk19         NUMBER,
    thk20         NUMBER,
    thk21         NUMBER,
    thk22         NUMBER,
    thk23         NUMBER,
    thk24         NUMBER,
    thk25         NUMBER,
    key1          NUMBER,
    key2          NUMBER,
    key3          NUMBER,
    key4          NUMBER,
    key5          NUMBER,
    key6          NUMBER,
    key7          NUMBER,
    key8          NUMBER,
    key9          NUMBER,
    key10         NUMBER
  );

rem
```

```
rem  Aggregate results for stock level transactions.
rem  These results are from the measurement interval only.
rem
        CREATE TABLE tpcc_sto_res
        (
          run_name      VARCHAR2(20),
          rep1          NUMBER,
          rep2          NUMBER,
          rep3          NUMBER,
          rep4          NUMBER,
          rep5          NUMBER,
          rep6          NUMBER,
          rep7          NUMBER,
          rep8          NUMBER,
          rep9          NUMBER,
          rep10         NUMBER,
          rep11         NUMBER,
          rep12         NUMBER,
          rep13         NUMBER,
          rep14         NUMBER,
          rep15         NUMBER,
          rep16         NUMBER,
          rep17         NUMBER,
          rep18         NUMBER,
          rep19         NUMBER,
          rep20         NUMBER,
          rep21         NUMBER,
          rep22         NUMBER,
          rep23         NUMBER,
          rep24         NUMBER,
          rep25         NUMBER,
          rep26         NUMBER,
          rep27         NUMBER,
          rep28         NUMBER,
          rep29         NUMBER,
          rep30         NUMBER,
          rep31         NUMBER,
          rep32         NUMBER,
          rep33         NUMBER,
          rep34         NUMBER,
          rep35         NUMBER,
          rep36         NUMBER,
          rep37         NUMBER,
          rep38         NUMBER,
          rep39         NUMBER,
          rep40         NUMBER,
          rep41         NUMBER,
          rep42         NUMBER,
          rep43         NUMBER,
          rep44         NUMBER,
          rep45         NUMBER,
          rep46         NUMBER,
          rep47         NUMBER,
          rep48         NUMBER,
          rep49         NUMBER,
          rep50         NUMBER,
          rep51         NUMBER,
          rep52         NUMBER,
          rep53         NUMBER,
          rep54         NUMBER,
          rep55         NUMBER,
          rep56         NUMBER,
          rep57         NUMBER,
          rep58         NUMBER,
          rep59         NUMBER,
          rep60         NUMBER,
          rep61         NUMBER,
          rep62         NUMBER,
          rep63         NUMBER,
          rep64         NUMBER,
          rep65         NUMBER,
          rep66         NUMBER,
          rep67         NUMBER,
          rep68         NUMBER,
          rep69         NUMBER,
          rep70         NUMBER,
          rep71         NUMBER,
          rep72         NUMBER,
          rep73         NUMBER,
          rep74         NUMBER,
          rep75         NUMBER,
          rep76         NUMBER,
          rep77         NUMBER,
          rep78         NUMBER,
          rep79         NUMBER,
          rep80         NUMBER,
          rep81         NUMBER,
          rep82         NUMBER,
          rep83         NUMBER,
          rep84         NUMBER,
          rep85         NUMBER,
          rep86         NUMBER,
          rep87         NUMBER,
          rep88         NUMBER,
          rep89         NUMBER,
          rep90         NUMBER,
          rep91         NUMBER,
          rep92         NUMBER,
          rep93         NUMBER,
          rep94         NUMBER,
          rep95         NUMBER,
          rep96         NUMBER,
          rep97         NUMBER,
          rep98         NUMBER,
          rep99         NUMBER,
          rep100        NUMBER,
          thk1          NUMBER,
          thk2          NUMBER,
          thk3          NUMBER,
          thk4          NUMBER,
          thk5          NUMBER,
          thk6          NUMBER,
          thk7          NUMBER,
          thk8          NUMBER,
          thk9          NUMBER,
          thk10         NUMBER,
          thk11         NUMBER,
          thk12         NUMBER,
          thk13         NUMBER,
          thk14         NUMBER,
          thk15         NUMBER,
          thk16         NUMBER,
          thk17         NUMBER,
          thk18         NUMBER,
          thk19         NUMBER,
          thk20         NUMBER,
          thk21         NUMBER,
          thk22         NUMBER,
          thk23         NUMBER,
          thk24         NUMBER,
          thk25         NUMBER,
          key1          NUMBER,
          key2          NUMBER,
          key3          NUMBER,
          key4          NUMBER,
          key5          NUMBER,
          key6          NUMBER,
          key7          NUMBER,
          key8          NUMBER,
          key9          NUMBER,
          key10         NUMBER
        );

rem
rem  Results for stock level transactions.
rem  These results are from the measurement interval only.
rem
        CREATE TABLE bench_sto_res
        (
          run_name      VARCHAR2(20),
          audit_str     VARCHAR2(10),
          proc_no       NUMBER,
          rep1          NUMBER,
          rep2          NUMBER,
          rep3          NUMBER,
          rep4          NUMBER,
          rep5          NUMBER,
          rep6          NUMBER,
          rep7          NUMBER,
          rep8          NUMBER,
```

```
rep9        NUMBER,
rep10       NUMBER,
rep11       NUMBER,
rep12       NUMBER,
rep13       NUMBER,
rep14       NUMBER,
rep15       NUMBER,
rep16       NUMBER,
rep17       NUMBER,
rep18       NUMBER,
rep19       NUMBER,
rep20       NUMBER,
rep21       NUMBER,
rep22       NUMBER,
rep23       NUMBER,
rep24       NUMBER,
rep25       NUMBER,
rep26       NUMBER,
rep27       NUMBER,
rep28       NUMBER,
rep29       NUMBER,
rep30       NUMBER,
rep31       NUMBER,
rep32       NUMBER,
rep33       NUMBER,
rep34       NUMBER,
rep35       NUMBER,
rep36       NUMBER,
rep37       NUMBER,
rep38       NUMBER,
rep39       NUMBER,
rep40       NUMBER,
rep41       NUMBER,
rep42       NUMBER,
rep43       NUMBER,
rep44       NUMBER,
rep45       NUMBER,
rep46       NUMBER,
rep47       NUMBER,
rep48       NUMBER,
rep49       NUMBER,
rep50       NUMBER,
rep51       NUMBER,
rep52       NUMBER,
rep53       NUMBER,
rep54       NUMBER,
rep55       NUMBER,
rep56       NUMBER,
rep57       NUMBER,
rep58       NUMBER,
rep59       NUMBER,
rep60       NUMBER,
rep61       NUMBER,
rep62       NUMBER,
rep63       NUMBER,
rep64       NUMBER,
rep65       NUMBER,
rep66       NUMBER,
rep67       NUMBER,
rep68       NUMBER,
rep69       NUMBER,
rep70       NUMBER,
rep71       NUMBER,
rep72       NUMBER,
rep73       NUMBER,
rep74       NUMBER,
rep75       NUMBER,
rep76       NUMBER,
rep77       NUMBER,
rep78       NUMBER,
rep79       NUMBER,
rep80       NUMBER,
rep81       NUMBER,
rep82       NUMBER,
rep83       NUMBER,
rep84       NUMBER,
rep85       NUMBER,
rep86       NUMBER,
rep87       NUMBER,
rep88       NUMBER,
rep89       NUMBER,
rep90       NUMBER,
rep91       NUMBER,
rep92       NUMBER,
rep93       NUMBER,
rep94       NUMBER,
rep95       NUMBER,
rep96       NUMBER,
rep97       NUMBER,
rep98       NUMBER,
rep99       NUMBER,
rep100      NUMBER,
thk1        NUMBER,
thk2        NUMBER,
thk3        NUMBER,
thk4        NUMBER,
thk5        NUMBER,
thk6        NUMBER,
thk7        NUMBER,
thk8        NUMBER,
thk9        NUMBER,
thk10       NUMBER,
thk11       NUMBER,
thk12       NUMBER,
thk13       NUMBER,
thk14       NUMBER,
thk15       NUMBER,
thk16       NUMBER,
thk17       NUMBER,
thk18       NUMBER,
thk19       NUMBER,
thk20       NUMBER,
thk21       NUMBER,
thk22       NUMBER,
thk23       NUMBER,
thk24       NUMBER,
thk25       NUMBER,
key1        NUMBER,
key2        NUMBER,
key3        NUMBER,
key4        NUMBER,
key5        NUMBER,
key6        NUMBER,
key7        NUMBER,
key8        NUMBER,
key9        NUMBER,
key10       NUMBER
            );
commit;
set echo off;
rem spool off;
rem exit;
```

cre_tab.sql

```
rem
rem
================================================================
====+
rem        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA      |
rem                  OPEN SYSTEMS PERFORMANCE GROUP                 |
rem                       All Rights Reserved                       |
rem
================================================================
====+
rem  FILENAME
rem       cre_tab.sql
rem  DESCRIPTION
rem       Create temporary tables for consistency tests.
rem
================================================================
====
rem
rem Usage:   sqlplus tpcc/tpcc @cre_tab
rem

connect tpcc/tpcc;
```

```
set echo on;

drop table temp_o1;
drop table temp_no;
drop table temp_o2;
drop table temp_ol;
drop table tpcc_audit_tab;

create table temp_o1 (
  o_w_id  integer,
  o_d_id  integer,
  o_o_id  integer);

create table temp_no (
  no_w_id integer,
  no_d_id integer,
  no_o_id integer);

create table temp_o2 (
  o_w_id  integer,
  o_d_id  integer,
  o_count integer);

create table temp_ol (
  ol_w_id  integer,
  ol_d_id  integer,
  ol_count integer);

create table tpcc_audit_tab (starttime date);

delete from tpcc_audit_tab;

set echo off;
```

create_cache_views.sql

```
rem This script creates four views that when queried will return
rem the total number of buffers in the buffer cache and the total
rem number of cloned buffers from each of the database's tablespaces.
rem
rem This assumes that each table and index is in its own tablespace.
rem If this is not the case, another query can be used which uses the
rem database's object tables to decipher the different objects.  However,
rem this query is slower.
rem
rem This script assumes 7.3.x.  If you are using V7.2.x or below, please
rem replace svrmgrl with sqldba lmode=y.
rem
rem Modification History:
rem
rem wbattist      16-Jun-1996     Create two additional views to keep
rem                               track of the number of clones in each
rem                               tablespace.
rem
rem wbattist      24-May-1995     Add the state check for the cbf view
rem                               to ensure that cloned blocks are not
rem                               counted.
rem

connect internal/internal;
set echo on;
drop view cbf;
create view cbf as
 select distinct(dbarfil) file#, count(1) blocks
 from x$bh
   where dbarfil > 0 and state <> 3
 group by dbarfil;
drop view cbt;
create view cbt as
  select ts$.name name,sum(cbf.blocks) buffers
  from cbf, file$, ts$
    where cbf.file#=file$.file# and file$.ts#=ts$.ts#
  group by file$.ts#, ts$.name;
drop view cbfcln;
create view cbfcln as
 select distinct(dbarfil) file#, count(1) blocks
 from x$bh
   where dbarfil > 0
 group by dbarfil;
drop view cbtcln;
create view cbtcln as
  select ts$.name name,sum(cbfcln.blocks) buffers
  from cbfcln, file$, ts$
    where cbfcln.file#=file$.file# and file$.ts#=ts$.ts#
  group by file$.ts#, ts$.name;

set echo off;
```

createbig_rbs.sh

```
#!/bin/sh
$SEQLPLUS internal/internal @createbig_rbs > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

createbig_rbs.sql

```
spool createbig_rbs.log
set echo on

create tablespace temperm2 datafile '?/dbs/tpcc_disks/temp_perm'
size 6000M reuse extent management local uniform size 100k nologging;
create tablespace temperm3 datafile '?/dbs/tpcc_disks/temp_perm_2'
size 6000M reuse extent management local uniform size 500K nologging;
create tablespace temperm4 datafile '?/dbs/tpcc_disks/temp_perm_3'
size 4000M reuse extent management local uniform size 100K nologging;


alter rollback segment big_rbs offline;
drop  rollback segment big_rbs ;
create rollback segment big_rbs
tablespace temperm2;
rem storage (minextents 2);

alter rollback segment big_rbs2 offline;
drop  rollback segment big_rbs2;
create rollback segment big_rbs2
tablespace temperm2;
rem storage (minextents 2);
drop  rollback segment big_rbs3 ;
create rollback segment big_rbs3
tablespace temperm3
storage (minextents 2);
drop  rollback segment big_rbs4 ;
create rollback segment big_rbs4
tablespace temperm3
storage (minextents 2);
drop  rollback segment big_rbs5 ;
create rollback segment big_rbs5
tablespace temperm3
storage (minextents 2);
drop  rollback segment big_rbs6 ;
create rollback segment big_rbs6
tablespace temperm3
storage (minextents 2);
drop  rollback segment big_rbs7 ;
create rollback segment big_rbs7
tablespace temperm3
storage (minextents 2);
drop  rollback segment big_rbs8 ;
create rollback segment big_rbs8
tablespace temperm3
storage (minextents 2);
spool off
```

```
set echo off
exit
```

dml.sh

```
#!/bin/sh
svrmgrl <<!
connect tpcc/tpcc;

set echo on;
  alter table ware disable table lock;
  alter table dist disable table lock;
  alter table cust disable table lock;
  alter table hist disable table lock;
  alter table item disable table lock;
  alter table stok disable table lock;
  alter table ordr disable table lock;
  alter table nord disable table lock;
  alter table ordl disable table lock;
  alter table tpcc_audit_tab disable table lock;
  alter table tpcc_back_res  disable table lock;
  alter table tpcc_del_res   disable table lock;
  alter table tpcc_new_res   disable table lock;
  alter table tpcc_ord_res   disable table lock;
  alter table tpcc_pay_res   disable table lock;
  alter table tpcc_run_desc  disable table lock;
  alter table tpcc_run_int   disable table lock;
  alter table tpcc_sto_res   disable table lock;
  alter table tpcc_tpm       disable table lock;
  alter table tpcc_user_res  disable table lock;
  alter table bench_del_res  disable table lock;
  alter table bench_new_res  disable table lock;
  alter table bench_ord_res  disable table lock;
  alter table bench_pay_res  disable table lock;
  alter table bench_run_int  disable table lock;
  alter table bench_sto_res  disable table lock;
  alter table bench_user_res disable table lock;
set echo off;
exit;
!
```

driver.sh

```
#!sh

STEP=1
START=0
END=0
CONTINUE=1
PROGNAME="driver.sh"
STOPFILE="stop"
TRACEFILE="log/trace.log"
JUNKFILE="junk"

usage()
{
  echo ""
  echo "Usage: $PROGNAME [<startstepno> <stopstepno>] [<startstepno>] [-step
<stepno>]"
  echo "      [<startstepno> <stopstepno>] - allows user to run a specified"
  echo "                   range of steps."
  echo "      [<startstepno>]              - runs from step number <start> till"
  echo "                   the end of the script."
  echo "      [-step <stepno>]             - runs only step number <stepno> and"
  echo "                   then stops."
  echo ""
  echo "            STEP    FUNCTION"
  echo "-------------------------------------------------------------"
  echo "             1      Create Database."
  echo "             2      Create Rollback Segments for Build."
  echo "             3      Shutdown Database."
  echo "             4      Startup Database for Build"
  echo "             5      Create User TPCC."
  echo "             6      Create Tablespaces."
  echo "             7      Assign Temporary Tablespace to user TPCC."
  echo "             8      Create Data Dictonary Views."
  echo "             9      Create Warehouse."
  echo "             10     Create District."
  echo "             11     Create Customer."
  echo "             12     Create History."
  echo "             13     Create Order."
  echo "             14     Create Neworder."
  echo "             15     Create Orderline."
  echo "             16     Create Stock."
  echo "             17     Create Item."
  echo "             18     Load Warehouse."
  echo "             19     Load District."
  echo "             20     Load Item."
  echo "             21     Load History"
  echo "             22     Load Neworder"
  echo "             23     Load Order/Orderline"
  echo "             24     Load Customer"
  echo "             25     Load Stock"
  echo "             26     Create Warehouse Index."
  echo "             27     Create District Index."
  echo "             28     Create Item Index."
  echo "             29     Create Customer1 Index."
  echo "             30     Create Customer2 Index."
  echo "             31     Create Stock Index."
  echo "             32     Create Orders1 Index."
  echo "             33     Create Orders2 Index."
  echo "             34     Create Neworder Index."
  echo "             35     Create Orderline Index."
  echo "             36     Analyze Tables/Clusters/Indexes."
  echo "             37     Create Statistics Tables."
  echo "             38     Load Stored Procedures."
  echo "             39     Create Rollback Segments for Runs."
  echo "             40     Generate Space Report."
  echo "             41     Misc."
  echo "             42     Offline Rollback segments for Build."
  echo "-------------------------------------------------------------"

  exit 1;
}

torun()
{
  mv -f *.log* log > junk 2>&1
  rm -f $JUNKFILE*

  if test -f $STOPFILE
  then
    echo "An error has occured, please look into the $TRACEFILE file."
    echo "OR you need to remove the 'stop' file found in the current diretory."
    echo "The 'stop' file need to be removed after an error occurs and before resuming."
    exit 1;
  fi
  if test $STEP -ge $START
  then
    if test $STEP -le $END
    then
      STEP=`expr $STEP + 1`
      return 0
    else
      if test $CONTINUE -eq 0
      then
        STEP=`expr $STEP + 1`
        return 0
      fi
    fi
  fi
  STEP=`expr $STEP + 1`
  return 1
}

case $# in
      0)   usage;
           ;;
      1)   case $1 in
                -h)  usage
                     ;;
           [0-9]*)  START=$1
                    CONTINUE=0
```

```
                ;;
            *)  usage
                ;;
        esac
        ;;
    2)  case $1 in
            -step) shift
                case $1 in
                  [0-9]*) ;;
                       *) usage
                          ;;
                esac
                START=$1
                END=$1
                CONTINUE=1
                ;;
          [0-9]*) START=$1
                shift
                case $1 in
                  [0-9]*) ;;
                       *) usage
                          ;;
                esac
                END=$1
                CONTINUE=1
                ;;
              *) usage
                ;;
        esac
        ;;
    *)  usage
        ;;
esac

if torun
then
  echo "Creating the Database ..."
  echo "Creating the Database ..." `date` "\n" >> $TRACEFILE
  ./step2createdb.sh
  if test $? -ne 0
  then
    echo "Creating the Database failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step2createdb.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Creating the Database failed ..."
  else
    echo "Creating the Database done." `date` "\n" >> $TRACEFILE
    echo "Creating the Database done ..."
  fi
fi

if torun
then
  echo "Creating the Rollback Segments ..."
  echo "Creating the Rollback Segments ..." `date` "\n" >> $TRACEFILE
  ./step3createrollback.sh
  if test $? -ne 0
  then
    echo "Creating the Rollback Segments failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step3createrollback.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Creating the Rollback Segments failed ..."
  else
    echo "Creating the Rollback Segments done." `date` "\n" >> $TRACEFILE
    echo "Creating the Rollback Segments done ..."
  fi
fi

if torun
then
  echo "Shutting down the Database ..."
  echo "Shutting down the Database ..." `date` "\n" >> $TRACEFILE
  ./stepshut.sh
  if test $? -ne 0
  then
    echo "Shutting down the Database failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/stepshut.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Shutting down the Database failed ..."
  else
    echo "Shutting down the Database done." `date` "\n" >> $TRACEFILE
    echo "Shutting down the Database done ..."
  fi
fi

if torun
then
  echo "Start up Database for Build ..."
  echo "Start up Database for Build ..." `date` "\n" >> $TRACEFILE
  ./stepstartb.sh
  if test $? -ne 0
  then
    echo "Start up Database for Build failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/stepstartb.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Start up Database for Build failed ..."
  else
    echo "Start up Database for Build done." `date` "\n" >> $TRACEFILE
    echo "Start up Database for Build done ..."
  fi
fi

if torun
then
  echo "Create User TPCC ..."
  echo "Create User TPCC ..." `date` "\n" >> $TRACEFILE
  ./stepcreateuser.sh
  if test $? -ne 0
  then
    echo "Create User TPCC failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/stepcreateuser.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create User TPCC failed ..."
  else
    echo "Create User TPCC  done." `date` "\n" >> $TRACEFILE
    echo "Create User TPCC done ..."
  fi
fi

if torun
then
  echo "Create Tablespaces ..."
  echo "Create Tablespaces ..." `date` "\n" >> $TRACEFILE
  ./step5createts.sh
  if test $? -ne 0
  then
    echo "Create Tablespaces failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step5createts*.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Tablespaces failed ..."
  else
    echo "Create Tablespaces done." `date` "\n" >> $TRACEFILE
    echo "Create Tablespaces done ..."
  fi
fi

if torun
then
  echo "Assign Temporary Tablespace to user TPCC ..."
  echo "Assign Temporary Tablespace to user TPCC ..." `date` "\n" >> $TRACEFILE
  ./stepusertemp.sh
  if test $? -ne 0
  then
    echo "Assign Temporary Tablespace to user TPCC failed." `date` "\n" >>
$TRACEFILE
    echo "Look at log/stepusertemp.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Assign Temporary Tablespace to user TPCC failed ..."
  else
    echo "Assign Temporary Tablespace to user TPCC done." `date` "\n" >>
$TRACEFILE
    echo "Assign Temporary Tablespace to user TPCC done ..."
  fi
fi

if torun
then
  echo "Create Data Dictionary views ..."
  echo "Create Data Dictionary views ..." `date` "\n" >> $TRACEFILE
```

```
  ./step6createddviews.sh
  if test $? -ne 0
  then
    echo "Create Data Dictionary views failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step6createddviews.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Data Dictionary views failed ..."
  else
    echo "Create Data Dictionary views done." `date` "\n" >> $TRACEFILE
    echo "Create Data Dictionary views done ..."
  fi
fi

if torun
then
  echo "Create Warehouse ..."
  echo "Create Warehouse ..." `date` "\n" >> $TRACEFILE
  ./step9createware.sh
  if test $? -ne 0
  then
    echo "Create Warehouse failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step9createware.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Warehouse failed ..."
  else
    echo "Create Warehouse  done." `date` "\n" >> $TRACEFILE
    echo "Create Warehouse  done ..."
  fi
fi

if torun
then
  echo "Create District ..."
  echo "Create District ..." `date` "\n" >> $TRACEFILE
  ./step10createdist.sh
  if test $? -ne 0
  then
    echo "Create District failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step10createdist.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create District failed ..."
  else
    echo "Create District done." `date` "\n" >> $TRACEFILE
    echo "Create District done ..."
  fi
fi

if torun
then
  echo "Create Customer ..."
  echo "Create Customer ..." `date` "\n" >> $TRACEFILE
  ./step11createcust.sh
  if test $? -ne 0
  then
    echo "Create Customer failed.\n" `date` "\n" >> $TRACEFILE
    echo "Look at log/step11createcust.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Customer failed ..."
  else
    echo "Create Customer done." `date` "\n" >> $TRACEFILE
    echo "Create Customer done ..."
  fi
fi

if torun
then
  echo "Create History ..."
  echo "Create History ..." `date` "\n" >> $TRACEFILE
  ./step12createhist.sh
  if test $? -ne 0
  then
    echo "Create History failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step12createhist.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create History failed ..."
  else
    echo "Create History done." `date` "\n" >> $TRACEFILE
    echo "Create History done ..."
  fi
fi

if torun
then
  echo "Create Order ..."
  echo "Create Order ..." `date` "\n" >> $TRACEFILE
  ./step13createordr.sh
  if test $? -ne 0
  then
    echo "Create Order failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step13createordr.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Order failed ..."
  else
    echo "Create Order done." `date` "\n" >> $TRACEFILE
    echo "Create Order done ..."
  fi
fi

if torun
then
  echo "Create Neworder ..."
  echo "Create Neworder ..." `date` "\n" >> $TRACEFILE
  ./step14createnord.sh
  if test $? -ne 0
  then
    echo "Create Neworder failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step14createnord.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Neworder failed ..."
  else
    echo "Create Neworder done." `date` "\n" >> $TRACEFILE
    echo "Create Neworder done ..."
  fi
fi

if torun
then
  echo "Create Orderline ..."
  echo "Create Orderline ..." `date` "\n" >> $TRACEFILE
  ./step15createordl.sh
  if test $? -ne 0
  then
    echo "Create Orderline failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step15createordl.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Orderline failed ..."
  else
    echo "Create Orderline done." `date` "\n" >> $TRACEFILE
    echo "Create Orderline done ..."
  fi
fi

if torun
then
  echo "Create Stock ..."
  echo "Create Stock ..." `date` "\n" >> $TRACEFILE
  ./step16createstok.sh
  if test $? -ne 0
  then
    echo "Create Stock failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step16createstok.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Stock failed ..."
  else
    echo "Create Stock done." `date` "\n" >> $TRACEFILE
    echo "Create Stock done ..."
  fi
fi

if torun
then
```

```
  echo "Create Item ..."
  echo "Create Item ..." `date` "\n" >> $TRACEFILE
  ./step17createitem.sh
  if test $? -ne 0
  then
    echo "Create Item failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step17createitem.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Item failed ..."
  else
    echo "Create Item done." `date` "\n" >> $TRACEFILE
    echo "Create Item done ..."
  fi
fi

if torun
then
  echo "Load Warehouse ..."
  echo "Load Warehouse ..." `date` "\n" >> $TRACEFILE
  ./step20loadware.sh
  if test $? -ne 0
  then
    echo "Load Warehouse failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step20loadware.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Warehouse failed ..."
  else
    echo "Load Warehouse done." `date` "\n" >> $TRACEFILE
    echo "Load Warehouse done ..."
  fi
fi

if torun
then
  echo "Load District ..."
  echo "Load District ..." `date` "\n" >> $TRACEFILE
  ./step21loaddist.sh
  if test $? -ne 0
  then
    echo "Load District failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step21loaddist.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load District failed ..."
  else
    echo "Load District done." `date` "\n" >> $TRACEFILE
    echo "Load District done ..."
  fi
fi

if torun
then
  echo "Load Item ..."
  echo "Load Item ..." `date` "\n" >> $TRACEFILE
  ./step22loaditem.sh
  if test $? -ne 0
  then
    echo "Load Item failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step22loaditem.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Item failed ..."
  else
    echo "Load Item done." `date` "\n" >> $TRACEFILE
    echo "Load Item done ..."
  fi
fi

if torun
then
  echo "Load History ..."
  echo "Load History ..." `date` "\n" >> $TRACEFILE
  ./step23loadhist.sh
  if test $? -ne 0
  then
    echo "Load History failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step23loadhist*.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load History failed ..."
  else
    echo "Load History done." `date` "\n" >> $TRACEFILE
    echo "Load History done ..."
  fi
fi

if torun
then
  echo "Load Neworder ..."
  echo "Load Neworder ..." `date` "\n" >> $TRACEFILE
  ./step24loadnord.sh
  if test $? -ne 0
  then
    echo "Load Neworder failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step24loadnord*.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Neworder failed ..."
  else
    echo "Load Neworder done." `date` "\n" >> $TRACEFILE
    echo "Load Neworder done ..."
  fi
fi

if torun
then
  echo "Load Order/Orderline ..."
  echo "Load Order/Orderline ..." `date` "\n" >> $TRACEFILE
  ./step25loadordrordl.sh
  if test $? -ne 0
  then
    echo "Load Order/Orderline failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step25loadordrordl*.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Order/Orderline failed ..."
  else
    echo "Load Order/Orderline done." `date` "\n" >> $TRACEFILE
    echo "Load Order/Orderline done ..."
  fi
fi

if torun
then
  echo "Load Customer ..."
  echo "Load Customer ..." `date` "\n" >> $TRACEFILE
  ./step26loadcust.sh
  if test $? -ne 0
  then
    echo "Load Customer failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step26loadcust*.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Customer failed ..."
  else
    echo "Load Customer done." `date` "\n" >> $TRACEFILE
    echo "Load Customer done ..."
  fi
fi

if torun
then
  echo "Load Stock ..."
  echo "Load Stock ..." `date` "\n" >> $TRACEFILE
  ./step27loadstok.sh
  if test $? -ne 0
  then
    echo "Load Stock failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step27loadstok*.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Stock failed ..."
  else
    echo "Load Stock done." `date` "\n" >> $TRACEFILE
    echo "Load Stock done ..."
  fi
fi

if torun
```

```
then
  echo "Create Warehouse Index ..."
  echo "Create Warehouse Index ..." `date` "\n" >> $TRACEFILE
  ./step29createiware.sh
  if test $? -ne 0
  then
    echo "Create Warehouse Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step29createiware.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Warehouse Index failed ..."
  else
    echo "Create Warehouse Index done." `date` "\n" >> $TRACEFILE
    echo "Create Warehouse Index done ..."
  fi
fi

if torun
then
  echo "Create District Index ..."
  echo "Create District Index ..." `date` "\n" >> $TRACEFILE
  ./step30createidist.sh
  if test $? -ne 0
  then
    echo "Create District Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step30createidist.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create District Index failed ..."
  else
    echo "Create District Index done." `date` "\n" >> $TRACEFILE
    echo "Create District Index done ..."
  fi
fi

if torun
then
  echo "Create Item Index ..."
  echo "Create Item Index ..." `date` "\n" >> $TRACEFILE
  ./step31createiitem.sh
  if test $? -ne 0
  then
    echo "Create Item Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step31createiitem.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Item Index failed ..."
  else
    echo "Create Item Index done." `date` "\n" >> $TRACEFILE
    echo "Create Item Index done ..."
  fi
fi

if torun
then
  echo "Create Customer1 Index ..."
  echo "Create Customer1 Index ..." `date` "\n" >> $TRACEFILE
  ./step32createicust1.sh
  if test $? -ne 0
  then
    echo "Create Customer1 Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step32createicust1.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Customer1 Index failed ..."
  else
    echo "Create Customer1 Index done." `date` "\n" >> $TRACEFILE
    echo "Create Customer1 Index done ..."
  fi
fi

if torun
then
  echo "Create Customer2 Index ..."
  echo "Create Customer2 Index ..." `date` "\n" >> $TRACEFILE
  ./step33createicust2.sh
  if test $? -ne 0
  then
    echo "Create Customer2 Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step33createicust2.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Customer2 Index failed ..."
  else
    echo "Create Customer2 Index done." `date` "\n" >> $TRACEFILE
    echo "Create Customer2 Index done ..."
  fi
fi

if torun
then
  echo "Create Stock Index ..."
  echo "Create Stock Index ..." `date` "\n" >> $TRACEFILE
  ./step34createistok.sh
  if test $? -ne 0
  then
    echo "Create Stock Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step34createistok.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Stock Index failed ..."
  else
    echo "Create Stock Index done." `date` "\n" >> $TRACEFILE
    echo "Create Stock Index done ..."
  fi
fi

if torun
then
  echo "Create Order1 Index ..."
  echo "Create Order1 Index ..." `date` "\n" >> $TRACEFILE
  ./step35createiordr1.sh
  if test $? -ne 0
  then
    echo "Create Order1 Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step35createiordr1.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Order1 Index failed ..."
  else
    echo "Create Order1 Index done." `date` "\n" >> $TRACEFILE
    echo "Create Order1 Index done ..."
  fi
fi

if torun
then
  echo "Create Order2 Index ..."
  echo "Create Order2 Index ..." `date` "\n" >> $TRACEFILE
  ./step36createiordr2.sh
  if test $? -ne 0
  then
    echo "Create Order2 Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step36createiordr2.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Order2 Index failed ..."
  else
    echo "Create Order2 Index done." `date` "\n" >> $TRACEFILE
    echo "Create Order2 Index done ..."
  fi
fi

if torun
then
  echo "Create Neworder Index ..."
  echo "Create Neworder Index ..." `date` "\n" >> $TRACEFILE
  ./step37createinord.sh
  if test $? -ne 0
  then
    echo "Create Neworder Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step37createinord.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Neworder Index failed ..."
  else
    echo "Create Neworder Index done." `date` "\n" >> $TRACEFILE
    echo "Create Neworder Index done ..."
  fi
```

```
fi

if torun
then
  echo "Create Orderline Index ..."
  echo "Create Orderline Index ..." `date` "\n" >> $TRACEFILE
  ./step38createiordl.sh
  if test $? -ne 0
  then
    echo "Create Orderline Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step38createiordl.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Orderline Index failed ..."
  else
    echo "Create Orderline Index done." `date` "\n" >> $TRACEFILE
    echo "Create Orderline Index done ..."
  fi
fi

if torun
then
  echo "Analyze Tables/Clusters/Indexes ..."
  echo "Analyze Tables/Clusters/Indexes ..." `date` "\n" >> $TRACEFILE
  ./step39analyze.sh
  if test $? -ne 0
  then
    echo "Analyze Tables/Clusters/Indexes failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step39analyze.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Analyze Tables/Clusters/Indexes failed ..."
  else
    echo "Analyze Tables/Clusters/Indexes done." `date` "\n" >> $TRACEFILE
    echo "Analyze Tables/Clusters/Indexes done ..."
  fi
fi

if torun
then
  echo "Create Statistics Tables ..."
  echo "Create Statistics Tables ..." `date` "\n" >> $TRACEFILE
  ./step40createstats.sh
  if test $? -ne 0
  then
    echo "Create Statistics Tables failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step40createstats.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Statistics Tables failed ..."
  else
    echo "Create Statistics Tables done." `date` "\n" >> $TRACEFILE
    echo "Create Statistics Tables done ..."
  fi
fi

if torun
then
  echo "Load Stored Procedures ..."
  echo "Load Stored Procedures ..." `date` "\n" >> $TRACEFILE
  ./step41createstoredprocs.sh
  if test $? -ne 0
  then
    echo "Load Stored Procedures failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step41createstoredprocs.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Stored Procedures failed ..."
  else
    echo "Load Stored Procedures done." `date` "\n" >> $TRACEFILE
    echo "Load Stored Procedures done ..."
  fi
fi

if torun
then
  echo "Create Rollback Segments for Runs ..."
  echo "Create Rollback Segments for Runs ..." `date` "\n" >> $TRACEFILE
  ./step18createrollsegs.sh
  if test $? -ne 0
  then
    echo "Create Rollback Segments for Runs failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step18createrollsegs.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Rollback Segments for Runs failed ..."
  else
    echo "Create Rollback Segments for Runs done." `date` "\n" >> $TRACEFILE
    echo "Create Rollback Segments for Runs done ..."
  fi
fi

if torun
then
  echo "Generate Space Reports ..."
  echo "Generate Space Reports ..." `date` "\n" >> $TRACEFILE
  ./step42createspacestats.sh
  if test $? -ne 0
  then
    echo "Generate Space Reports failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step42createspacestats.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Generate Space Reports failed ..."
  else
    echo "Generate Space Reports done." `date` "\n" >> $TRACEFILE
    echo "Generate Space Reports done ..."
  fi
fi

if torun
then
  echo "Misc ..."
  echo "Misc ..." `date` "\n" >> $TRACEFILE
  ./step43createmisc.sh
  if test $? -ne 0
  then
    echo "Misc failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step43createmisc.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Misc failed ..."
  else
    echo "Misc done." `date` "\n" >> $TRACEFILE
    echo "Misc done ..."
  fi
fi

if torun
then
  echo "Offline Rollback Segments for Build ..."
  echo "Offline Rollback Segments for Build ..." `date` "\n" >> $TRACEFILE
  ./step47offlinerollsegs.sh
  if test $? -ne 0
  then
    echo "Offline Rollback Segments for Build failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step47offlinerollsegs.log for more details." `date` "\n" >>
$TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Offline Rollback Segments for Build failed ..."
  else
    echo "Offline Rollback Segments for Build done." `date` "\n" >> $TRACEFILE
    echo "Offline Rollback Segments for Build done ..."
  fi
fi

torun
```

droptemp.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @droptemp > droptemp 2>&1

  exit ;
```

droptemp.sql

```
spool droptemp.log;
set echo on;
drop tablespace temp_0;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_1' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_2' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_3' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_4' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_5' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_6' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_7' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_8' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_9' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_10' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_11' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_12' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_13' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_14' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_15' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_16' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_17' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_18' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_19' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_20' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_21' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_22' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_23' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_24' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_25' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_26' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_27' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_28' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_29' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_30' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_31' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_32' OFFLINE DROP;
alter database datafile '/ora815ufs/dbs/tpcc_disks/temp_0_33' OFFLINE DROP;
set timing on;

spool off;
set echo off;
exit ;
```

extent.sql

```
REM=============================================================
========+
REM          Copyright (c) 1994  Oracle Corp, Belmont, CA         |
REM               OPEN SYSTEMS PERFORMANCE GROUP                  |
REM                   All Rights Reserved                         |
REM=============================================================
========+
REM FILENAME
REM      extent.sql
REM DESCRIPTION
REM      List all extents in all the TPCC tablespaces.
REM
REM  Usage: sqlplus internal/internal @extent [<block size in bytes>]
REM=============================================================
========*/

    set space 2
    set pagesize 2000
    set echo off
    set termout off
    set verify off
    set feedback off
    spool extent.rpt

    select substr(tablespace_name,1,8) tspace,
      substr(segment_name,1,11) segment, substr(segment_type,1,15) type,
      substr(extent_id,1,5) eid, substr(file_id,1,5) fid, blocks, blocks * &&1 / 1048576
size_MB
    from   dba_extents
    where  owner = 'TPCC' AND ( segment_type = 'INDEX' OR
      segment_type = 'INDEX PARTITION' OR segment_type = 'CLUSTER'
      OR segment_type = 'TABLE' OR segment_type = 'TABLE PARTITION')
      AND tablespace_name <> 'SYSTEM'
    order by tablespace_name, segment_name, extent_id, file_id;

    select substr(tablespace_name,1,8) tspace,
      substr(segment_name,1,11) segment,
      sum(blocks) tot_blk, sum(blocks) * &&1 / 1048576 size_MB
    from   dba_extents
    where  owner = 'TPCC' AND ( segment_type = 'INDEX' OR
      segment_type = 'INDEX PARTITION' OR segment_type = 'CLUSTER'
      OR segment_type = 'TABLE' OR segment_type = 'TABLE PARTITION')
      AND tablespace_name <> 'SYSTEM'
    group by tablespace_name, segment_name
    order by tablespace_name, segment_name;

    spool off;
```

freeext.sql

```
REM=============================================================
========+
REM          Copyright (c) 1994  Oracle Corp, Belmont, CA         |
REM               OPEN SYSTEMS PERFORMANCE GROUP                  |
REM                   All Rights Reserved                         |
REM=============================================================
========+
REM FILENAME
REM      freeext.sql
REM DESCRIPTION
REM      List all free extents in all the TPCC tablespace
REM
REM  Usage: sqlplus internal/internal @freeext [<block size in bytes>]
REM=============================================================
========*/

    set space 2
    set pagesize 2000
    set echo off
    set termout off
    set verify off
    set feedback off
    spool freeextent.rpt

    select substr(tablespace_name,1,8) tspace, file_id, block_id, blocks,
        blocks * &&1 / 1048576 size_MB
    from dba_free_space
    order by tablespace_name, file_id, block_id;

    select substr(tablespace_name,1,8) tspace, sum(blocks) tot_blk,
        sum(blocks) * &&1 / 1048576 size_MB
    from   dba_free_space
    group by tablespace_name
    order by tablespace_name;
```

initnew.sql

```
-- The initnew package for storing variables used in the
-- New Order anonymous block

CREATE OR REPLACE PACKAGE initnew
AS
 TYPE intarray IS TABLE OF INTEGER index by binary_integer;
 TYPE distarray IS TABLE OF VARCHAR(24) index by binary_integer;
 nulldate       DATE;
 s_dist                                  distarray;
 idx1arr                    intarray;
 s_remote                   intarray;
 PROCEDURE new_init(idxarr intarray);
```

```
END initnew;
/
show errors;

CREATE OR REPLACE PACKAGE BODY initnew AS
  PROCEDURE new_init (idxarr  intarray)
  IS
  BEGIN
       -- initialize null date
   nulldate := TO_DATE('01-01-1811', 'MM-DD-YYYY');
    idx1arr := idxarr;
  END new_init;
END initnew;
/
show errors
```

initpay.sql

```
CREATE OR REPLACE PACKAGE initpay
AS
 TYPE rowidarray IS TABLE OF ROWID INDEX BY BINARY_INTEGER;
 row_id                rowidarray;
 cust_rowid              ROWID;
 dist_name             VARCHAR2(11);
 ware_name               VARCHAR2(11);
 c_num                 BINARY_INTEGER;
 PROCEDURE pay_init;
END initpay;
/
show errors;

CREATE OR REPLACE PACKAGE BODY initpay AS
  PROCEDURE pay_init IS
  BEGIN
    NULL;
  END pay_init;
END initpay;
/
show errors;
```

noparallel.sh

```
#!/bin/sh
./undml.sh
  svrmgrl <<!
  connect tpcc/tpcc;
  alter cluster stokcluster deallocate unused;
  alter cluster custcluster deallocate unused;
  alter table ware deallocate unused;
  alter table dist deallocate unused;
  alter cluster dcluster deallocate unused;
  alter cluster wcluster deallocate unused;
  alter cluster itemcluster deallocate unused;
  alter table hist deallocate unused;
  alter table ordr deallocate unused;
  alter table ordl deallocate unused;
  alter table nord deallocate unused;
  alter index icust1 deallocate unused;
  alter index icust2 deallocate unused;
  alter index istok deallocate unused;
  alter index iordr1 deallocate unused;
  alter index iordr2 deallocate unused;
  alter index iitem deallocate unused;
  alter index istok deallocate unused;
  alter index iware deallocate unused;
  alter index idist deallocate unused;
  alter cluster stokcluster noparallel;
  alter cluster custcluster noparallel;
  alter table ware noparallel;
  alter table dist noparallel;
  alter table wcluster noparallel;
  alter table dcluster noparallel;
  alter cluster itemcluster noparallel;
  alter table hist noparallel;
  alter table ordr noparallel;
  alter table ordl noparallel;
  alter table nord noparallel;
  alter index icust1 noparallel;
  alter index icust2 noparallel;
  alter index iordr1 noparallel;
  alter index iordr2 noparallel;
  alter index iitem noparallel;
  alter index istok noparallel;
  alter index iware noparallel;
  alter index idist noparallel;
  alter tablespace SYSTEM coalesce;
  alter tablespace stok_0  coalesce;
  alter tablespace istok_0 coalesce;
  alter tablespace ordr_0 coalesce;
  alter tablespace item_0 coalesce;
  alter tablespace iitem_0 coalesce;
  alter tablespace ware_0 coalesce;
  alter tablespace dist_0 coalesce;
  alter tablespace ordl_0 coalesce;
  alter tablespace nord_0 coalesce;
  alter tablespace temperm2 coalesce;
  alter tablespace hist_0 coalesce;
  alter tablespace IORDR1_0 coalesce;
  alter tablespace  IWARE_0 coalesce;
  alter tablespace icust1_0 coalesce;
  alter tablespace icust2_0 coalesce;
  alter tablespace idist_0 coalesce;
  alter tablespace iordr2_0 coalesce;
  alter tablespace cust_0 coalesce;
  alter tablespace roll_0 coalesce;
  alter cluster custcluster storage (buffer_pool default);
  alter table hist storage (buffer_pool default);
  alter cluster itemcluster storage (buffer_pool default);
  alter cluster stokcluster storage (buffer_pool default);
exit;
!
./dml.sh
```

params.dat

The parameters that generated this kit

```
ops = 0
os = 1
last_name = battistella
node = 8
ldrive = 1
company_name = oracle
cpu = 4
email = wbattist@us.oracle.com
first_name = walter
kver = 0
scale = 13504
memsize = 16384
view = 0
ver = 0
```

plsql_mon.sql

```
rem
rem
============================================================
====+
rem        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA       |
rem                OPEN SYSTEMS PERFORMANCE GROUP                |
rem                     All Rights Reserved                      |
rem
============================================================
====+
rem  FILENAME
rem       plsql_mon.sql
```

```
rem  DESCRIPTION
rem       SQL script to create a stored package for PL/SQL stored
rem       procedures to dump messages.
rem
============================================================
====
rem
rem Usage:   sqlplus tpcc/tpcc @plsql_mon
rem

connect tpcc/tpcc;
set echo on;
CREATE OR REPLACE PACKAGE plsql_mon_pack
IS
  PROCEDURE print
  (
    info        VARCHAR2
  );
END;
/
show errors;

CREATE OR REPLACE PACKAGE BODY plsql_mon_pack
IS
  PROCEDURE print
  (
    info        VARCHAR2
  )
  IS
    s           NUMBER;
  BEGIN
    dbms_pipe.pack_message (info);
    s := dbms_pipe.send_message ('plsql_mon');
    IF (s <> 0) THEN
      raise_application_error (-20000, 'Error:' || to_char(s) ||
                        ' sending on pipe');
    END IF;
  END;
END;
/
show errors;

set echo off;
```

post_restore.sh

```
#!/bin/sh
svrmgrl <<!
connect internal;
startup pfile=p_recov.ora;
exit;
!
./undml.sh
sqlplus  sys/change_on_install @step40post
sqlplus tpcc/tpcc @step43post
svrmgrl <<!
  connect tpcc/tpcc;
  alter cluster stokcluster deallocate unused;
  alter cluster custcluster deallocate unused;
  alter table ware deallocate unused;
  alter table dist deallocate unused;
  alter cluster itemcluster deallocate unused;
  alter table hist deallocate unused;
  alter table ordr deallocate unused;
  alter table ordl deallocate unused;
  alter table nord deallocate unused;
  alter index icust1 deallocate unused;
  alter index icust2 deallocate unused;
  alter index istok deallocate unused;
  alter index iordr1 deallocate unused;
  alter index iordr2 deallocate unused;
  alter index iitem deallocate unused;
  alter index istok deallocate unused;
  alter index iware deallocate unused;
  alter index idist deallocate unused;
  alter cluster stokcluster noparallel;
  alter cluster custcluster noparallel;
  alter table ware noparallel;
  alter table dist noparallel;
  alter cluster itemcluster noparallel;
  alter table hist noparallel;
  alter table ordr noparallel;
  alter table ordl noparallel;
  alter table nord noparallel;
  alter index icust1 noparallel;
  alter index icust2 noparallel;
  alter index iordr1 noparallel;
  alter index iordr2 noparallel;
  alter index iitem noparallel;
  alter index istok noparallel;
  alter index iware noparallel;
  alter index idist noparallel;
  alter tablespace SYSTEM coalesce;
  alter tablespace stok_0  coalesce;
  alter tablespace istok_0 coalesce;
  alter tablespace ordr_0 coalesce;
  alter tablespace item_0 coalesce;
  alter tablespace iitem_0 coalesce;
  alter tablespace ware_0 coalesce;
  alter tablespace dist_0 coalesce;
  alter tablespace ordl_0 coalesce;
  alter tablespace nord_0 coalesce;
  alter tablespace temperm2 coalesce;
  alter tablespace hist_0 coalesce;
  alter tablespace IORDR1_0 coalesce;
  alter tablespace  IWARE_0 coalesce;
  alter tablespace icust1_0 coalesce;
  alter tablespace icust2_0 coalesce;
  alter tablespace idist_0 coalesce;
  alter tablespace iordr2_0 coalesce;
  alter tablespace cust_0 coalesce;
  alter tablespace roll_0 coalesce;
  alter table ware enable row movement;
  alter table dist enable row movement;
  alter table item enable row movement;
  alter table hist enable row movement;
  alter table ordr enable row movement;
  alter table ordl enable row movement;
  alter table nord enable row movement;
  alter table stok enable row movement;
  alter table cust enable row movement;
  connect internal/internal
  alter sequence sys.audses$ cache 200;
exit
!
./dml.sh
./buffer_pool_off.sh
svrmgrl <<!
connect internal;
shutdown;
exit;
!
```

pst_c.sql

```
rem
rem
rem
============================================================
====+
rem          Copyright (c) 1992  Oracle Corp, Belmont, CA          |
rem               OPEN SYSTEMS PERFORMANCE GROUP                |
rem                    All Rights Reserved                      |
rem
============================================================
====+
rem  FILENAME
rem       pst_c.sql
rem  DESCRIPTION
rem       Create Table for OS Specific Process Stats
rem
============================================================
====*/
rem
```

```
rem  Tables for Unix-specific process statistics
rem
rem  Usage: sqlplus internal/internal @pst_c
rem

      connect sys/change_on_install;
      set echo on;
      DROP TABLE proc_resource;
      DROP TABLE os_stat;

rem
rem  Resource usage for a process.
rem

      CREATE TABLE proc_resource
      (
         config       VARCHAR2(10),
         run          NUMBER,
         proc         NUMBER,
         child        NUMBER,
         user_cpu_ms   NUMBER,
         system_cpu_ms NUMBER,
         maxrss       NUMBER,
         pagein       NUMBER,
         reclaim      NUMBER,
         zerofill     NUMBER,
         pffincr      NUMBER,
         pffdecr      NUMBER,
         swap         NUMBER,
         syscall      NUMBER,
         volcsw       NUMBER,
         involcsw     NUMBER,
         signal       NUMBER,
         lread        NUMBER,
         lwrite       NUMBER,
         bread        NUMBER,
         bwrite       NUMBER,
         phread       NUMBER,
         phwrite      NUMBER
      );

rem
rem  OS statistics.
rem  These results are from the measurement interval only.
rem

      CREATE TABLE os_stat
      (
         config        VARCHAR2(10),
         run           NUMBER,
         hid           NUMBER,
         syscall       NUMBER,
         intr          NUMBER,
         cswitch       NUMBER,
         pagefault     NUMBER,
         usr           NUMBER,
         sys           NUMBER,
         idl           NUMBER,
         wio           NUMBER
      );

      set echo off;
```

space_get.sql

```
REM=============================================================
========+
REM       Copyright (c) 1995  Oracle Corp, Redwood Shores, CA     |
REM                 OPEN SYSTEMS PERFORMANCE GROUP                |
REM                     All Rights Reserved                       |
REM=============================================================
========+
REM FILENAME
REM      space_get.sql
REM DESCRIPTION
REM      Get sizes of tables, indexes and tablespaces.
REM  Usage: sqlplus internal/internal @space_get [<tpm> <# of warehouses>]
REM=============================================================
========*/

  set echo on;
  delete from tpcc_data;
  delete from tpcc_space;
  delete from tpcc_totspace;

  insert into tpcc_data
  select substr(segment_name,1,11), substr(segment_type,1,15),
        sum(blocks),
        round(sum(blocks) * 0.05), 0,
        sum(blocks) + round(sum(blocks) * 0.05)
  from   dba_extents
  where  owner = 'TPCC' AND ( segment_type = 'INDEX' OR
        segment_type = 'INDEX PARTITION' OR segment_type = 'CLUSTER'
        OR segment_type = 'TABLE' OR segment_type = 'TABLE PARTITION')
        AND tablespace_name <> 'SYSTEM'
  group by segment_name, segment_type;

  insert into tpcc_data
    select 'SYSTEM', 'SYS', sum(blocks), 0, 0, sum(blocks)
    from   dba_data_files
    where  tablespace_name = 'SYSTEM';

  insert into tpcc_data
    select 'ROLL_SEG', 'SYS',
          sum(blocks), 0, 0, sum(blocks)
    from   dba_data_files
    where  tablespace_name like 'ROLL%'
    group by tablespace_name;

  update tpcc_data
    set five_pct = 0,
        daily_grow = round(blocks * &&1 / 62.5 / &&2),
        total = blocks + round(blocks * &&1 / 62.5 / &&2)
    where segment = 'HIST' OR segment = 'ORDR' OR
          segment = 'IORDL';

  insert into tpcc_space
  select substr(ex$.name,1,11), sum(sp$.sz_blocks), 0, 0, 0, 0
  from
    (select tablespace_name , sum(blocks) sz_blocks
     from dba_data_files
     where tablespace_name <> 'SYSTEM'
     group by tablespace_name
    ) sp$,
    (select distinct tablespace_name, segment_name name
     from dba_extents
     where owner = 'TPCC'
       and (segment_type = 'CLUSTER' or segment_type = 'TABLE'
         or segment_type = 'TABLE PARTITION' or segment_type = 'INDEX'
         or segment_type = 'INDEX PARTITION')
       and tablespace_name <> 'SYSTEM'
    ) ex$
  where sp$.tablespace_name = ex$.tablespace_name
  group by ex$.name;

  insert into tpcc_space
  select substr(tablespace_name,1,11), sum(blocks), 0, 0, 0, 0
  from dba_data_files
  where tablespace_name = 'SYSTEM'
  group by tablespace_name;

  update tpcc_space
    set required =
    (
      select sum(total)
      from   tpcc_data
      where  tpcc_data.segment = tpcc_space.segment
    )
    where segment in
    (
      select segment from tpcc_data
    );

  update tpcc_space
    set static =
    (
      select sum(total)
```

```
    from   tpcc_data
    where  tpcc_data.segment = tpcc_space.segment
  )
  where segment in
  (
    select segment from tpcc_data
  );

update tpcc_space
  set static = 0,
     dynamic =
  (
    select sum(blocks)
    from   tpcc_data
    where  tpcc_data.segment = tpcc_space.segment
  )
  where segment in ('HIST', 'ORDR', 'IORDL');

update tpcc_space
  set oversize = blocks - required;

insert into tpcc_totspace
  select &&1, &&2, sum(static), sum(dynamic), sum(oversize), 0, 0, 0
  from   tpcc_space;

update tpcc_totspace
  set daily_grow =
  (
    select sum(daily_grow)
    from   tpcc_data
  );

update tpcc_totspace
  set space180 = static + 180 * daily_grow;

set echo off;
```

space_init.sql

```
REM=============================================================
========+
REM FILENAME
REM       space_init.sql
REM DESCRIPTION
REM       Create tables for space calculations.
REM  Usage: sqlplus internal/internal @space_init.sql
REM=============================================================
========*/

  set echo on;
  drop table tpcc_data;
  drop table tpcc_space;
  drop table tpcc_totspace;

  create table tpcc_data (
    segment      varchar2(11),
    type         varchar2(15),
    blocks       number,
    five_pct     number,
    daily_grow   number,
    total        number
  );

  create table tpcc_space (
    segment       varchar2(11),
    blocks        number,
    required      number,
    static        number,
    dynamic       number,
    oversize      number
  );

  create table tpcc_totspace (
    tpm           number,
    nware         number,
    static        number,
    dynamic       number,
    oversize      number,
    daily_grow    number,
    daily_spre    number,
    space180      number
  );

  create unique index itpcc_data on tpcc_data (segment);
  create unique index itpcc_space on tpcc_space (segment);
  set echo off;
```

space_rpt.sql

```
REM=============================================================
========+
REM         Copyright (c) 1995  Oracle Corp, Redwood Shores, CA      |
REM                  OPEN SYSTEMS PERFORMANCE GROUP                  |
REM                       All Rights Reserved                        |
REM=============================================================
========+
REM FILENAME
REM       space_rpt.sql
REM DESCRIPTION
REM       Generate space report and save it in space.rpt
REM  Usage: sqlplus internal/internal @space_rpt.sql
REM=============================================================
========*/

  set space 2
  set pagesize 2000
  set echo off
  set termout off
  set verify off
  set feedback off
  spool space.rpt

  select tpm, nware from tpcc_totspace;

  select * from tpcc_data order by segment;

  select * from tpcc_space order by segment;

  select static, dynamic, oversize, daily_grow, daily_spre, space180
    from tpcc_totspace;

  spool off;
```

step10createdist.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step10createdist > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step10createdist.sql

```
spool step10createdist.log;
set echo on;
drop table dist;

set timing on;

create table dist (
        d_id            number(2,0),
                d_w_id          number(5,0),
        d_ytd           number,
```

```
        d_tax           number,
        d_next_o_id     number,
        d_name          varchar2(10),
        d_street_1      varchar2(20),
        d_street_2      varchar2(20),
        d_city          varchar2(20),
        d_state         char(2),
        d_zip           char(9)
)
  partition by range(d_w_id)
  (
    partition dist_0 values less than (1689)
      tablespace dist_0,
    partition dist_1 values less than (3377)
      tablespace dist_1,
    partition dist_2 values less than (5065)
      tablespace dist_2,
    partition dist_3 values less than (6753)
      tablespace dist_3,
    partition dist_4 values less than (8441)
      tablespace dist_4,
    partition dist_5 values less than (10129)
      tablespace dist_5,
    partition dist_6 values less than (11817)
      tablespace dist_6,
    partition dist_7 values less than (maxvalue)
      tablespace dist_7
  )
  initrans    4
  pctfree    99
  pctused    0
  storage (   freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

step11createcust.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step11createcust > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step11createcust.sql

```
spool step11createcust.log;
set echo on;
drop table cust;
drop cluster custcluster including tables;

set timing on;

create cluster custcluster (
c_id    number(5,0)
,c_d_id    number(2,0)
,c_w_id    number(5,0)
  )
  single    table
  hashkeys    405120000
  hash is   (c_w_id * 30000 + c_id * 10 + c_d_id - 30011)
  size    850
  initrans    3
  pctfree    0
storage (initial 1582510K next 1582500K pctincrease 0 minextents 257 buffer_pool
recycle freelists 22 freelist groups 4)
  tablespace    cust_0;

create table cust (
        c_id            number(5,0),
        c_d_id          number(2,0),
        c_w_id          number(5,0),
        c_discount      number,
        c_credit        char(2),
        c_last          varchar2(16),
        c_first         varchar2(16),
        c_credit_lim    number,
        c_balance       number,
        c_ytd_payment   number,
        c_payment_cnt   number,
        c_delivery_cnt  number,
        c_street_1      varchar2(20),
        c_street_2      varchar2(20),
        c_city          varchar2(20),
        c_state         char(2),
        c_zip           char(9),
        c_phone         char(16),
        c_since         date,
        c_middle        char(2),
        c_data          varchar2(500)
)
cluster custcluster (c_id
,c_d_id
,c_w_id
);
spool off;
set echo off;
exit sql.sqlcode;
```

step12createhist.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step12createhist > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step12createhist.sql

```
spool step12createhist.log;
set echo on;
drop table hist;

set timing on;

create table hist (
        h_c_id          number,
        h_c_d_id        number,
        h_c_w_id        number,
        h_d_id          number,
        h_w_id          number,
        h_date          date,
        h_amount        number,
        h_data          varchar2(24)
)
  partition by range(h_w_id)
  (
    partition hist_0 values less than (1689)
      tablespace hist_0,
    partition hist_1 values less than (3377)
      tablespace hist_1,
    partition hist_2 values less than (5065)
      tablespace hist_2,
    partition hist_3 values less than (6753)
      tablespace hist_3,
    partition hist_4 values less than (8441)
      tablespace hist_4,
    partition hist_5 values less than (10129)
```

```
    tablespace hist_5,
  partition hist_6 values less than (11817)
    tablespace hist_6,
  partition hist_7 values less than (maxvalue)
    tablespace hist_7
 )
 initrans   4
 pctfree   5
 storage ( freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

step13createordr.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step13createordr > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step13createordr.sql

```
spool step13createordr.log;
set echo on;
drop table ordr;

set timing on;

create table ordr (
       o_id            number,
       o_w_id          number,
       o_d_id          number,
       o_c_id          number,
       o_carrier_id    number,
       o_ol_cnt        number,
       o_all_local     number,
       o_entry_d       date
)
  partition by range(o_w_id)
  (
    partition ordr_0 values less than (1689)
      tablespace ordr_0,
    partition ordr_1 values less than (3377)
      tablespace ordr_1,
    partition ordr_2 values less than (5065)
      tablespace ordr_2,
    partition ordr_3 values less than (6753)
      tablespace ordr_3,
    partition ordr_4 values less than (8441)
      tablespace ordr_4,
    partition ordr_5 values less than (10129)
      tablespace ordr_5,
    partition ordr_6 values less than (11817)
      tablespace ordr_6,
    partition ordr_7 values less than (maxvalue)
      tablespace ordr_7
  )
  initrans   4
  pctfree   5
  storage ( freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

step14createnord.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step14createnord > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step14createnord.sql

```
spool step14createnord.log;
set echo on;
drop table nord;

set timing on;

create table nord (
       no_w_id         number,
       no_d_id         number,
       no_o_id         number,
                constraint inord primary key (no_w_id, no_d_id, no_o_id)
)
  organization index
  partition by range(no_w_id)
  (
    partition nord_0 values less than (1689)
      tablespace nord_0,
    partition nord_1 values less than (3377)
      tablespace nord_1,
    partition nord_2 values less than (5065)
      tablespace nord_2,
    partition nord_3 values less than (6753)
      tablespace nord_3,
    partition nord_4 values less than (8441)
      tablespace nord_4,
    partition nord_5 values less than (10129)
      tablespace nord_5,
    partition nord_6 values less than (11817)
      tablespace nord_6,
    partition nord_7 values less than (maxvalue)
      tablespace nord_7
  )
  initrans   4
  pctfree   5
  storage ( freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

step15createordl.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step15createordl > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step15createordl.sql

```
spool step15createordl.log;
set echo on;
drop table ordl;

set timing on;

create table ordl (
       ol_w_id         number,
       ol_d_id         number,
       ol_o_id         number,
       ol_number       number,
       ol_i_id         number,
       ol_delivery_d   date,
       ol_amount       number,
       ol_supply_w_id  number,
       ol_quantity     number,
       ol_dist_info    char(24),
                constraint iordl primary key (ol_w_id, ol_d_id, ol_o_id, ol_number)
)
  organization index
  partition by range(ol_w_id)
  (
    partition ordl_0 values less than (1689)
      tablespace ordl_0,
    partition ordl_1 values less than (3377)
      tablespace ordl_1,
    partition ordl_2 values less than (5065)
      tablespace ordl_2,
    partition ordl_3 values less than (6753)
      tablespace ordl_3,
    partition ordl_4 values less than (8441)
      tablespace ordl_4,
    partition ordl_5 values less than (10129)
      tablespace ordl_5,
    partition ordl_6 values less than (11817)
      tablespace ordl_6,
    partition ordl_7 values less than (maxvalue)
      tablespace ordl_7
  )
  initrans    4
  pctfree    5
  storage ( freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

step16createstok.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step16createstok > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step16createstok.sql

```
spool step16createstok.log;
set echo on;
drop table stok;
drop cluster stokcluster including tables;

set timing on;

create cluster stokcluster (
s_i_id    number(6,0)
,s_w_id    number(5,0)
  )
  single    table
  hashkeys    1350400000
  hash is   (abs(s_i_id - 1) * 1688  + mod((s_w_id - 1), 1688) + trunc ((s_w_id - 1) /
1688) * 168800000)
  size    350
  initrans    3
  pctfree    0
  storage ( initial 1688010K next 1688000K pctincrease 0 maxextents unlimited
          minextents 321 buffer_pool keep freelists 22 freelist groups 4 )
  tablespace    stok_0;

create table stok (
       s_i_id          number(6,0),
       s_w_id          number(5,0),
       s_quantity      number,
       s_ytd           number,
       s_order_cnt     number,
       s_remote_cnt    number,
       s_data          varchar2(50),
       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)
)
cluster stokcluster (s_i_id
,s_w_id
);
spool off;
set echo off;
exit sql.sqlcode;
```

step17createitem.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step17createitem > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step17createitem.sql

```
spool step17createitem.log;
set echo on;
drop table item;
drop cluster itemcluster including tables;

set timing on;

create cluster itemcluster (
i_id    number(6,0)
  )
  single    table
  hashkeys    100000
  hash is    (i_id + 1)
  size    120
  initrans    3
  pctfree    0
  storage ( buffer_pool keep freelists 22 freelist groups 43 )
  tablespace    item_0;
```

```
create table item (
      i_id          number(6,0),
      i_name        varchar2(24),
      i_price       number,
      i_data        varchar2(50),
      i_im_id       number
)
cluster itemcluster (i_id
);
spool off;
set echo off;
exit sql.sqlcode;
```

step18createrollsegs.sh

```
#!sh
./addts.sh    roll_0 \?/dbs/tpcc_disks/roll_0_1 1024M 40k
./addfile.sh  roll_0 \?/dbs/tpcc_disks/roll_0_2 1024M &
./addfile.sh  roll_0 \?/dbs/tpcc_disks/roll_0_3 1024M &
./addfile.sh  roll_0 \?/dbs/tpcc_disks/roll_0_4 1024M &
wait
./addts.sh    roll_1 \?/dbs/tpcc_disks/roll_1_1 1024M 40k
./addfile.sh  roll_1 \?/dbs/tpcc_disks/roll_1_2 1024M &
./addfile.sh  roll_1 \?/dbs/tpcc_disks/roll_1_3 1024M &
./addfile.sh  roll_1 \?/dbs/tpcc_disks/roll_1_4 1024M &
wait
./addts.sh    roll_2 \?/dbs/tpcc_disks/roll_2_1 1024M 40k
./addfile.sh  roll_2 \?/dbs/tpcc_disks/roll_2_2 1024M &
./addfile.sh  roll_2 \?/dbs/tpcc_disks/roll_2_3 1024M &
./addfile.sh  roll_2 \?/dbs/tpcc_disks/roll_2_4 1024M &
wait
./addts.sh    roll_3 \?/dbs/tpcc_disks/roll_3_1 1024M 40k
./addfile.sh  roll_3 \?/dbs/tpcc_disks/roll_3_2 1024M &
./addfile.sh  roll_3 \?/dbs/tpcc_disks/roll_3_3 1024M &
./addfile.sh  roll_3 \?/dbs/tpcc_disks/roll_3_4 1024M &
wait
./addts.sh    roll_4 \?/dbs/tpcc_disks/roll_4_1 1024M 40k
./addfile.sh  roll_4 \?/dbs/tpcc_disks/roll_4_2 1024M &
./addfile.sh  roll_4 \?/dbs/tpcc_disks/roll_4_3 1024M &
./addfile.sh  roll_4 \?/dbs/tpcc_disks/roll_4_4 1024M &
wait
./addts.sh    roll_5 \?/dbs/tpcc_disks/roll_5_1 1024M 40k
./addfile.sh  roll_5 \?/dbs/tpcc_disks/roll_5_2 1024M &
./addfile.sh  roll_5 \?/dbs/tpcc_disks/roll_5_3 1024M &
./addfile.sh  roll_5 \?/dbs/tpcc_disks/roll_5_4 1024M &
wait
./addts.sh    roll_6 \?/dbs/tpcc_disks/roll_6_1 1024M 40k
./addfile.sh  roll_6 \?/dbs/tpcc_disks/roll_6_2 1024M &
./addfile.sh  roll_6 \?/dbs/tpcc_disks/roll_6_3 1024M &
./addfile.sh  roll_6 \?/dbs/tpcc_disks/roll_6_4 1024M &
wait
./addts.sh    roll_7 \?/dbs/tpcc_disks/roll_7_1 1024M 40k
./addfile.sh  roll_7 \?/dbs/tpcc_disks/roll_7_2 1024M &
./addfile.sh  roll_7 \?/dbs/tpcc_disks/roll_7_3 1024M &
./addfile.sh  roll_7 \?/dbs/tpcc_disks/roll_7_4 1024M &
wait
echo "Done with adding RBS files at ‘date‘"

sqlplus system/manager @tpcc_roll_0_1 &
sqlplus system/manager @tpcc_roll_0_2 &
sqlplus system/manager @tpcc_roll_0_3 &
sqlplus system/manager @tpcc_roll_0_4 &
sqlplus system/manager @tpcc_roll_1_1 &
sqlplus system/manager @tpcc_roll_1_2 &
sqlplus system/manager @tpcc_roll_1_3 &
sqlplus system/manager @tpcc_roll_1_4 &
sqlplus system/manager @tpcc_roll_2_1 &
sqlplus system/manager @tpcc_roll_2_2 &
sqlplus system/manager @tpcc_roll_2_3 &
sqlplus system/manager @tpcc_roll_2_4 &
sqlplus system/manager @tpcc_roll_3_1 &
sqlplus system/manager @tpcc_roll_3_2 &
sqlplus system/manager @tpcc_roll_3_3 &
sqlplus system/manager @tpcc_roll_3_4 &
wait
sqlplus system/manager @tpcc_roll_4_1 &
sqlplus system/manager @tpcc_roll_4_2 &
sqlplus system/manager @tpcc_roll_4_3 &
sqlplus system/manager @tpcc_roll_4_4 &
wait
sqlplus system/manager @tpcc_roll_5_1 &
sqlplus system/manager @tpcc_roll_5_2 &
sqlplus system/manager @tpcc_roll_5_3 &
sqlplus system/manager @tpcc_roll_5_4 &
wait
sqlplus system/manager @tpcc_roll_6_1 &
sqlplus system/manager @tpcc_roll_6_2 &
sqlplus system/manager @tpcc_roll_6_3 &
sqlplus system/manager @tpcc_roll_6_4 &
wait
sqlplus system/manager @tpcc_roll_7_1 &
sqlplus system/manager @tpcc_roll_7_2 &
sqlplus system/manager @tpcc_roll_7_3 &
sqlplus system/manager @tpcc_roll_7_4 &
wait
echo "Done with adding RBS at ‘date‘"
```

step1tpccenv.sh

```
#!sh
SQLDBA=svrmgrl
export SQLDBA
SEQLPLUS=sqlplus
export SEQLPLUS
TPCCLOAD=tpccload.exe
export TPCCLOAD
ORACLE_SID=tpcc
export ORACLE_SID
```

step20loadware.sh

```
#!sh
$TPCCLOAD -M 13504 -w > step20loadware.log 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step21loaddist.sh

```
#!sh
$TPCCLOAD -M 13504 -d > step21loaddist.log 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step22loaditem.sh

```
#!sh
$TPCCLOAD -M 13504 -i > step22loaditem.log 2>&1

if test $? -ne 0
then
  exit 1;
else
```

```
  exit 0;
fi
```

step23loadhist.sh

```
#!sh
$TPCCLOAD -M 13504 -h -b 1 -e 1688 > step23loadhist.log0 2>&1 &
proc0=$!
$TPCCLOAD -M 13504 -h -b 1689 -e 3376 > step23loadhist.log1 2>&1 &
proc1=$!
$TPCCLOAD -M 13504 -h -b 3377 -e 5064 > step23loadhist.log2 2>&1 &
proc2=$!
$TPCCLOAD -M 13504 -h -b 5065 -e 6752 > step23loadhist.log3 2>&1 &
proc3=$!
$TPCCLOAD -M 13504 -h -b 6753 -e 8440 > step23loadhist.log4 2>&1 &
proc4=$!
$TPCCLOAD -M 13504 -h -b 8441 -e 10128 > step23loadhist.log5 2>&1 &
proc5=$!
$TPCCLOAD -M 13504 -h -b 10129 -e 11816 > step23loadhist.log6 2>&1 &
proc6=$!
$TPCCLOAD -M 13504 -h -b 11817 -e 13504 > step23loadhist.log7 2>&1 &
proc7=$!
wait $proc0
proc0=$?
wait $proc1
proc1=$?
wait $proc2
proc2=$?
wait $proc3
proc3=$?
wait $proc4
proc4=$?
wait $proc5
proc5=$?
wait $proc6
proc6=$?
wait $proc7
proc7=$?
proc=‘expr $proc0 + $proc1 + $proc2 + $proc3 + $proc4 + $proc5 + $proc6 + $proc7
+ 0‘
proc=‘expr $proc % 127‘

if test $proc -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step24loadnord.sh

```
#!sh
$TPCCLOAD -M 13504 -n -b 1 -e 1688 >step24loadnord.log0 2>&1 &
proc0=$!
$TPCCLOAD -M 13504 -n -b 1689 -e 3376 >step24loadnord.log1 2>&1 &
proc1=$!
$TPCCLOAD -M 13504 -n -b 3377 -e 5064 >step24loadnord.log2 2>&1 &
proc2=$!
$TPCCLOAD -M 13504 -n -b 5065 -e 6752 >step24loadnord.log3 2>&1 &
proc3=$!
$TPCCLOAD -M 13504 -n -b 6753 -e 8440 >step24loadnord.log4 2>&1 &
proc4=$!
$TPCCLOAD -M 13504 -n -b 8441 -e 10128 >step24loadnord.log5 2>&1 &
proc5=$!
$TPCCLOAD -M 13504 -n -b 10129 -e 11816 >step24loadnord.log6 2>&1 &
proc6=$!
$TPCCLOAD -M 13504 -n -b 11817 -e 13504 >step24loadnord.log7 2>&1 &
proc7=$!
wait $proc0
proc0=$?
wait $proc1
proc1=$?
wait $proc2
proc2=$?
wait $proc3
proc3=$?
wait $proc4
proc4=$?
wait $proc5
proc5=$?
wait $proc6
proc6=$?
wait $proc7
proc7=$?
proc=‘expr $proc0 + $proc1 + $proc2 + $proc3 + $proc4 + $proc5 + $proc6 + $proc7
+ 0‘
proc=‘expr $proc % 127‘

if test $proc -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step25loadordrordl.sh

```
#!sh
$TPCCLOAD -M 13504 -o ?/dbs/tpcc_disks/dummy0.dat -b 1 -e 1688
>step25loadordrordl.log0 2>&1 &
proc0=$!
$TPCCLOAD -M 13504 -o ?/dbs/tpcc_disks/dummy1.dat -b 1689 -e 3376
>step25loadordrordl.log1 2>&1 &
proc1=$!
$TPCCLOAD -M 13504 -o ?/dbs/tpcc_disks/dummy2.dat -b 3377 -e 5064
>step25loadordrordl.log2 2>&1 &
proc2=$!
$TPCCLOAD -M 13504 -o ?/dbs/tpcc_disks/dummy3.dat -b 5065 -e 6752
>step25loadordrordl.log3 2>&1 &
proc3=$!
$TPCCLOAD -M 13504 -o ?/dbs/tpcc_disks/dummy4.dat -b 6753 -e 8440
>step25loadordrordl.log4 2>&1 &
proc4=$!
$TPCCLOAD -M 13504 -o ?/dbs/tpcc_disks/dummy5.dat -b 8441 -e 10128
>step25loadordrordl.log5 2>&1 &
proc5=$!
$TPCCLOAD -M 13504 -o ?/dbs/tpcc_disks/dummy6.dat -b 10129 -e 11816
>step25loadordrordl.log6 2>&1 &
proc6=$!
$TPCCLOAD -M 13504 -o ?/dbs/tpcc_disks/dummy7.dat -b 11817 -e 13504
>step25loadordrordl.log7 2>&1 &
proc7=$!
wait $proc0
proc0=$?
wait $proc1
proc1=$?
wait $proc2
proc2=$?
wait $proc3
proc3=$?
wait $proc4
proc4=$?
wait $proc5
proc5=$?
wait $proc6
proc6=$?
wait $proc7
proc7=$?
proc=‘expr $proc0 + $proc1 + $proc2 + $proc3 + $proc4 + $proc5 + $proc6 + $proc7
+ 0‘
proc=‘expr $proc % 127‘

if test $proc -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step26loadcust.sh

```
#!sh
$TPCCLOAD -M 13504 -c -b 1 -e 1688 >step26loadcust.log0 2>&1 &
proc0=$!
$TPCCLOAD -M 13504 -c -b 1689 -e 3376 >step26loadcust.log1 2>&1 &
proc1=$!
$TPCCLOAD -M 13504 -c -b 3377 -e 5064 >step26loadcust.log2 2>&1 &
proc2=$!
$TPCCLOAD -M 13504 -c -b 5065 -e 6752 >step26loadcust.log3 2>&1 &
proc3=$!
$TPCCLOAD -M 13504 -c -b 6753 -e 8440 >step26loadcust.log4 2>&1 &
proc4=$!
$TPCCLOAD -M 13504 -c -b 8441 -e 10128 >step26loadcust.log5 2>&1 &
proc5=$!
$TPCCLOAD -M 13504 -c -b 10129 -e 11816 >step26loadcust.log6 2>&1 &
proc6=$!
$TPCCLOAD -M 13504 -c -b 11817 -e 13504 >step26loadcust.log7 2>&1 &
proc7=$!
wait $proc0
proc0=$?
wait $proc1
proc1=$?
wait $proc2
proc2=$?
wait $proc3
proc3=$?
wait $proc4
proc4=$?
wait $proc5
proc5=$?
wait $proc6
proc6=$?
wait $proc7
proc7=$?
proc=‘expr $proc0 + $proc1 + $proc2 + $proc3 + $proc4 + $proc5 + $proc6 + $proc7
+ 0‘
proc=‘expr $proc % 127‘

if test $proc -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step27loadstok.sh

```
#!sh
$TPCCLOAD -M 13504 -S -j 1 -k 12500 >step27loadstok.log0 2>&1 &
proc0=$!
$TPCCLOAD -M 13504 -S -j 12501 -k 25000 >step27loadstok.log1 2>&1 &
proc1=$!
$TPCCLOAD -M 13504 -S -j 25001 -k 37500 >step27loadstok.log2 2>&1 &
proc2=$!
$TPCCLOAD -M 13504 -S -j 37501 -k 50000 >step27loadstok.log3 2>&1 &
proc3=$!
$TPCCLOAD -M 13504 -S -j 50001 -k 62500 >step27loadstok.log4 2>&1 &
proc4=$!
$TPCCLOAD -M 13504 -S -j 62501 -k 75000 >step27loadstok.log5 2>&1 &
proc5=$!
$TPCCLOAD -M 13504 -S -j 75001 -k 87500 >step27loadstok.log6 2>&1 &
proc6=$!
$TPCCLOAD -M 13504 -S -j 87501 -k 100000 >step27loadstok.log7 2>&1 &
proc7=$!
wait $proc0
proc0=$?
wait $proc1
proc1=$?
wait $proc2
proc2=$?
wait $proc3
proc3=$?
wait $proc4
proc4=$?
wait $proc5
proc5=$?
wait $proc6
proc6=$?
wait $proc7
proc7=$?
proc=‘expr $proc0 + $proc1 + $proc2 + $proc3 + $proc4 + $proc5 + $proc6 + $proc7
+ 0‘
proc=‘expr $proc % 127‘

if test $proc -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step29createiware.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step29createiware > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step29createiware.sql

```
spool step29createiware.log;
set echo on;
drop index iware;

set timing on;

create unique index iware on ware (w_id)
  local
  (
    partition iware_0 tablespace iware_0,
    partition iware_1 tablespace iware_1,
    partition iware_2 tablespace iware_2,
    partition iware_3 tablespace iware_3,
    partition iware_4 tablespace iware_4,
    partition iware_5 tablespace iware_5,
    partition iware_6 tablespace iware_6,
    partition iware_7 tablespace iware_7
  )
  initrans    3
  parallel    8
  pctfree    1
  storage ( freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

step2createdb.new

```
spool step2createdb.log

set echo on

startup pfile=p_create.ora nomount
create database tpcc
  controlfile reuse
  maxdatafiles 500
  maxinstances 8
```

```
  datafile '?/dbs/tpcc_disks/system_001' size 1000M reuse
  logfile group 1 ('?/dbs/tpcc_disks/log1') size 2000M reuse,
    group 2 ('?/dbs/tpcc_disks/log2') size 2000M reuse;
spool off
set echo off
exit sql.sqlcode
```

step2createdb.sh

```
#!sh
$SEQLPLUS internal/internal @step2createdb > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step2createdb.sql

```
spool step2createdb.log

set echo on

startup pfile=p_create.ora nomount
create database tpcc
  controlfile reuse
  maxdatafiles 1254
  maxinstances 8
  datafile '?/dbs/tpcc_disks/system_001' size 1000M reuse,
  logfile thread 1 group 1 ('?/dbs/tpcc_disks/log0_1') size 8395M reuse,
                group 2 ('?/dbs/tpcc_disks/log0_2') size 8395M reuse;
  alter database add logfile thread 2 group 3 ('?/dbs/tpcc_disks/log1_1') size 8395M
reuse,
                                                group 4
('?/dbs/tpcc_disks/log1_2') size 8395M reuse;
  alter database enable public thread 2;
  alter database add logfile thread 3 group 5 ('?/dbs/tpcc_disks/log2_1') size 8395M
reuse,
                                                group 6
('?/dbs/tpcc_disks/log2_2') size 8395M reuse;
  alter database enable public thread 3;
  alter database add logfile thread 4 group 7 ('?/dbs/tpcc_disks/log3_1') size 8395M
reuse,
                                                group 8
('?/dbs/tpcc_disks/log3_2') size 8395M reuse;
  alter database enable public thread 4;
  alter database add logfile thread 5 group 9 ('?/dbs/tpcc_disks/log4_1') size 8395M
reuse,
                                                group 10
('?/dbs/tpcc_disks/log4_2') size 8395M reuse;
  alter database enable public thread 5;
  alter database add logfile thread 6 group 11 ('?/dbs/tpcc_disks/log5_1') size 8395M
reuse,
                                                group 12
('?/dbs/tpcc_disks/log5_2') size 8395M reuse;
  alter database enable public thread 6;
  alter database add logfile thread 7 group 13 ('?/dbs/tpcc_disks/log6_1') size 8395M
reuse,
                                                group 14
('?/dbs/tpcc_disks/log6_2') size 8395M reuse;
  alter database enable public thread 7;
  alter database add logfile thread 8 group 15 ('?/dbs/tpcc_disks/log7_1') size 8395M
reuse,
                                                group 16
('?/dbs/tpcc_disks/log7_2') size 8395M reuse;
  alter database enable public thread 8;
  exit;
!
spool off
set echo off
exit sql.sqlcode
```

step30createidist.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step30createidist > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step30createidist.sql

```
spool step30createidist.log;
set echo on;
drop index idist;

set timing on;

create unique index idist on dist (d_w_id, d_id)
  local
  (
    partition idist_0 tablespace idist_0,
    partition idist_1 tablespace idist_1,
    partition idist_2 tablespace idist_2,
    partition idist_3 tablespace idist_3,
    partition idist_4 tablespace idist_4,
    partition idist_5 tablespace idist_5,
    partition idist_6 tablespace idist_6,
    partition idist_7 tablespace idist_7
  )
  initrans    3
  parallel    8
  pctfree     5
  storage ( freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

step31createiitem.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step31createiitem > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step31createiitem.sql

```
spool step31createiitem.log;
set echo on;
drop index iitem;

set timing on;

create unique index iitem on item (i_id)
  initrans    4
  parallel    8
  pctfree     5
  storage ( freelists 22 freelist groups 43 )
  tablespace    iitem_0;
```

```
spool off;
set echo off;
exit sql.sqlcode;
```

step32createicust1.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step32createicust1 > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step32createicust1.sql

```
spool step32createicust1.log;
set echo on;
drop index icust1;

set timing on;

create unique index icust1 on cust (c_w_id, c_d_id, c_id)
  initrans    3
  parallel    8
  pctfree    1
  storage ( freelists 22 freelist groups 43 )
  tablespace    icust1_0;
spool off;
set echo off;
exit sql.sqlcode;
```

step33createicust2.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step33createicust2 > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step33createicust2.sql

```
spool step33createicust2.log;
set echo on;
drop index icust2;

set timing on;

create unique index icust2 on cust (c_last, c_w_id, c_d_id, c_first, c_id)
  initrans    3
  parallel    8
  pctfree    1
  storage ( freelists 22 freelist groups 43 )
  tablespace    icust2_0;
spool off;
set echo off;
exit sql.sqlcode;
```

step34createistok.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step34createistok > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step34createistok.sql

```
spool step34createistok.log;
set echo on;
drop index istok;

set timing on;

create unique index istok on stok (s_i_id, s_w_id)
  initrans    3
  parallel    8
  pctfree    1
  storage ( freelists 19 freelist groups 4 )
  tablespace    istok_0;

analyze index istok estimate statistics;
spool off;
set echo off;
exit sql.sqlcode;
```

step35createiordr1.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step35createiordr1 > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step35createiordr1.sql

```
spool step35createiordr1.log;
set echo on;
drop index iordr1;

set timing on;

create unique index iordr1 on ordr (o_w_id, o_d_id, o_id)
  local
  (
    partition iordr1_0 tablespace iordr1_0,
    partition iordr1_1 tablespace iordr1_1,
    partition iordr1_2 tablespace iordr1_2,
    partition iordr1_3 tablespace iordr1_3,
    partition iordr1_4 tablespace iordr1_4,
    partition iordr1_5 tablespace iordr1_5,
    partition iordr1_6 tablespace iordr1_6,
    partition iordr1_7 tablespace iordr1_7
  )
  initrans    3
  parallel    8
```

```
  pctfree    1
  storage ( freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

step36createiordr2.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step36createiordr2 > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step36createiordr2.sql

```
spool step36createiordr2.log;
set echo on;
drop index iordr2;

set timing on;

create unique index iordr2 on ordr (o_w_id, o_d_id, o_c_id, o_id)
  local
  (
    partition iordr2_0 tablespace iordr2_0,
    partition iordr2_1 tablespace iordr2_1,
    partition iordr2_2 tablespace iordr2_2,
    partition iordr2_3 tablespace iordr2_3,
    partition iordr2_4 tablespace iordr2_4,
    partition iordr2_5 tablespace iordr2_5,
    partition iordr2_6 tablespace iordr2_6,
    partition iordr2_7 tablespace iordr2_7
  )
  initrans    4
  parallel    8
  pctfree    25
  storage ( freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

step37createinord.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step37createinord > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step37createinord.sql

```
exit sql.sqlcode;
```

step38createiordl.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step38createiordl > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step38createiordl.sql

```
exit sql.sqlcode;
```

step39analyze.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step39analyze > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step39analyze.sql

```
spool step39analyze.log;
set echo on;
analyze cluster stokcluster estimate statistics;
analyze index istok estimate statistics;
analyze table ware compute statistics;
analyze cluster itemcluster estimate statistics;
analyze index iordl estimate statistics;
analyze index iordr1 estimate statistics;
analyze index iordr2 estimate statistics;
analyze index inord estimate statistics;
analyze table dist compute statistics;
analyze table ordr estimate statistics;
analyze table hist estimate statistics;
analyze index iitem estimate statistics;
analyze index iware compute statistics;
analyze index icust1 estimate statistics;
analyze index icust2 estimate statistics;
analyze index idist compute statistics;
analyze cluster custcluster estimate statistics;

set echo off;
spool off;

exit sql.sqlcode;
```

step3createrollback.sh

```
#!sh
$SEQLPLUS internal/internal @step3createrollback > junk 2>&1

if test $? -ne 0
then
  exit 1;
```

```
else
  exit 0;
fi
```

step3createrollback.sql

```
spool step3createrollback.log
set echo on

alter rollback segment t1 offline;
drop public rollback segment t1;
create public rollback segment t1
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t1 online;
alter rollback segment t2 offline;
drop public rollback segment t2;
create public rollback segment t2
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t2 online;
alter rollback segment t3 offline;
drop public rollback segment t3;
create public rollback segment t3
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t3 online;
alter rollback segment t4 offline;
drop public rollback segment t4;
create public rollback segment t4
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t4 online;
alter rollback segment t5 offline;
drop public rollback segment t5;
create public rollback segment t5
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t5 online;
alter rollback segment t6 offline;
drop public rollback segment t6;
create public rollback segment t6
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t6 online;
alter rollback segment t7 offline;
drop public rollback segment t7;
create public rollback segment t7
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t7 online;
alter rollback segment t8 offline;
drop public rollback segment t8;
create public rollback segment t8
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t8 online;
alter rollback segment t9 offline;
drop public rollback segment t9;
create public rollback segment t9
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t9 online;
alter rollback segment t10 offline;
drop public rollback segment t10;
create public rollback segment t10
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t10 online;
alter rollback segment t11 offline;
drop public rollback segment t11;
create public rollback segment t11
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t11 online;
alter rollback segment t12 offline;
drop public rollback segment t12;
create public rollback segment t12
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t12 online;
alter rollback segment t13 offline;
drop public rollback segment t13;
create public rollback segment t13
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t13 online;
alter rollback segment t14 offline;
drop public rollback segment t14;
create public rollback segment t14
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t14 online;
alter rollback segment t15 offline;
drop public rollback segment t15;
create public rollback segment t15
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t15 online;
alter rollback segment t16 offline;
drop public rollback segment t16;
create public rollback segment t16
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t16 online;
alter rollback segment t17 offline;
drop public rollback segment t17;
create public rollback segment t17
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t17 online;
alter rollback segment t18 offline;
drop public rollback segment t18;
create public rollback segment t18
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t18 online;
alter rollback segment t19 offline;
drop public rollback segment t19;
create public rollback segment t19
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t19 online;
alter rollback segment t20 offline;
drop public rollback segment t20;
create public rollback segment t20
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t20 online;
alter rollback segment t21 offline;
drop public rollback segment t21;
create public rollback segment t21
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t21 online;
alter rollback segment t22 offline;
drop public rollback segment t22;
create public rollback segment t22
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t22 online;
alter rollback segment t23 offline;
drop public rollback segment t23;
create public rollback segment t23
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t23 online;
alter rollback segment t24 offline;
drop public rollback segment t24;
create public rollback segment t24
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t24 online;
alter rollback segment t25 offline;
drop public rollback segment t25;
create public rollback segment t25
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t25 online;
alter rollback segment t26 offline;
drop public rollback segment t26;
create public rollback segment t26
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t26 online;
alter rollback segment t27 offline;
drop public rollback segment t27;
create public rollback segment t27
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t27 online;
alter rollback segment t28 offline;
drop public rollback segment t28;
create public rollback segment t28
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t28 online;
alter rollback segment t29 offline;
drop public rollback segment t29;
create public rollback segment t29
      storage (initial 200K minextents 2 next 200K);
alter rollback segment t29 online;
alter rollback segment t30 offline;
drop public rollback segment t30;
create public rollback segment t30
```

```
    storage (initial 200K minextents 2 next 200K);
alter rollback segment t30 online;

spool off
set echo off
exit sql.sqlcode
```

step40createstats.sh

```
#!sh
$SEQLPLUS internal/internal @step40createstats > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step40createstats.sql

```
spool step40createstats.log
@orst_cre
@c_stat
@pst_c
spool off
exit sql.sqlcode;
```

step40post.sql

```
spool step40createstats.log
set echo on
alter user tpcc default tablespace system;
alter user sys default tablespace system;
alter user sys temporary tablespace system;
@orst_cre.sql.ping
@c_stat
@pst_c
spool off
set echo off
exit
```

step41createstoredprocs.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step41createstoredprocs > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step41createstoredprocs.sql

```
spool step41createstoredprocs.log
@initpay
@initnew
spool off
exit sql.sqlcode;
```

step42createspacestats.sh

```
#!sh
$SEQLPLUS internal/internal @step42createspacestats > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step42createspacestats.sql

```
spool step42createspacestats.log
@space_init
@space_get 12 13504
@space_rpt
spool off
exit;
```

step43createmisc.sh

```
#!/bin/sh
$SEQLPLUS internal/internal @step43createmisc > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step43createmisc.sql

```
spool step43createmisc.log
set echo on;
alter user tpcc temporary tablespace system;
grant execute on dbms_lock to public;
grant execute on dbms_pipe to public;
grant select on v_$parameter to public;
@createbig_rbs.sql
@plsql_mon
@cre_tab
@create_cache_views
@views
@dml
@$ORACLE_HOME/rdbms/admin/catparr
@extent 2048
@freeext 2048
spool off
exit sql.sqlcode;
```

step43post.sql

```
spool step43createmisc.log
set echo on;
rem alter user tpcc default tablespace temperm2;
@plsql_mon
@cre_tab
@create_cache_views
@views
spool off
```

```
set echo off
exit
```

step47offlinerollsegs.sh

```
#!sh
$SEQLPLUS internal/internal @step47offlinerollsegs > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step47offlinerollsegs.sql

```
spool step47offlinerollsegs.log
set echo on

alter rollback segment t1 offline;
alter rollback segment t2 offline;
alter rollback segment t3 offline;
alter rollback segment t4 offline;
alter rollback segment t5 offline;
alter rollback segment t6 offline;
alter rollback segment t7 offline;
alter rollback segment t8 offline;
alter rollback segment t9 offline;
alter rollback segment t10 offline;
alter rollback segment t11 offline;
alter rollback segment t12 offline;
alter rollback segment t13 offline;
alter rollback segment t14 offline;
alter rollback segment t15 offline;
alter rollback segment t16 offline;
alter rollback segment t17 offline;
alter rollback segment t18 offline;
alter rollback segment t19 offline;
alter rollback segment t20 offline;
alter rollback segment t21 offline;
alter rollback segment t22 offline;
alter rollback segment t23 offline;
alter rollback segment t24 offline;
alter rollback segment t25 offline;
alter rollback segment t26 offline;
alter rollback segment t27 offline;
alter rollback segment t28 offline;
alter rollback segment t29 offline;
alter rollback segment t30 offline;

spool off
set echo off
exit sql.sqlcode
```

step5createts.sh

```
#!sh
addts_dict.sh stok_0 \?/dbs/tpcc_disks/stok_0_0 7200M > junk1 2>&1 &
addts_dict.sh cust_0 \?/dbs/tpcc_disks/cust_0_0 7200M > junk88 2>&1 &
addts.sh istok_0 \?/dbs/tpcc_disks/istok_0_0 3900M 100M > junk2 2>&1 &
addts.sh ware_0 \?/dbs/tpcc_disks/ware_0_0 5M 1M > junk3 2>&1 &
addts.sh ware_1 \?/dbs/tpcc_disks/ware_1_0 5M 1M > junk4 2>&1 &
addts.sh ware_2 \?/dbs/tpcc_disks/ware_2_0 5M 1M > junk5 2>&1 &
addts.sh ware_3 \?/dbs/tpcc_disks/ware_3_0 5M 1M > junk6 2>&1 &
addts.sh ware_4 \?/dbs/tpcc_disks/ware_4_0 5M 1M > junk7 2>&1 &
addts.sh ware_5 \?/dbs/tpcc_disks/ware_5_0 5M 1M > junk8 2>&1 &
addts.sh ware_6 \?/dbs/tpcc_disks/ware_6_0 5M 1M > junk9 2>&1 &
addts.sh ware_7 \?/dbs/tpcc_disks/ware_7_0 5M 1M > junk10 2>&1 &
addts.sh item_0 \?/dbs/tpcc_disks/item_0_0 16M 2M > junk11 2>&1 &
addts.sh ordl_0 \?/dbs/tpcc_disks/ordl_0_0 7500M 100M > junk12 2>&1 &
addts.sh ordl_1 \?/dbs/tpcc_disks/ordl_1_0 7500M 100M > junk13 2>&1 &
addts.sh ordl_2 \?/dbs/tpcc_disks/ordl_2_0 7500M 100M > junk14 2>&1 &
addts.sh ordl_3 \?/dbs/tpcc_disks/ordl_3_0 7500M 100M > junk15 2>&1 &
addts.sh ordl_4 \?/dbs/tpcc_disks/ordl_4_0 7500M 100M > junk16 2>&1 &
addts.sh ordl_5 \?/dbs/tpcc_disks/ordl_5_0 7500M 100M > junk17 2>&1 &
addts.sh ordl_6 \?/dbs/tpcc_disks/ordl_6_0 7500M 100M > junk18 2>&1 &
addts.sh ordl_7 \?/dbs/tpcc_disks/ordl_7_0 7500M 100M > junk19 2>&1 &
addts.sh iordr1_0 \?/dbs/tpcc_disks/iordr1_0_0 2000M 100M > junk20 2>&1 &
addts.sh iordr1_1 \?/dbs/tpcc_disks/iordr1_1_0 2000M 100M > junk21 2>&1 &
addts.sh iordr1_2 \?/dbs/tpcc_disks/iordr1_2_0 2000M 100M > junk22 2>&1 &
addts.sh iordr1_3 \?/dbs/tpcc_disks/iordr1_3_0 2000M 100M > junk23 2>&1 &
addts.sh iordr1_4 \?/dbs/tpcc_disks/iordr1_4_0 2000M 100M > junk24 2>&1 &
addts.sh iordr1_5 \?/dbs/tpcc_disks/iordr1_5_0 2000M 100M > junk25 2>&1 &
addts.sh iordr1_6 \?/dbs/tpcc_disks/iordr1_6_0 2000M 100M > junk26 2>&1 &
addts.sh iordr1_7 \?/dbs/tpcc_disks/iordr1_7_0 2000M 100M > junk27 2>&1 &
addts.sh iordr2_0 \?/dbs/tpcc_disks/iordr2_0_0 4000M 100M > junk28 2>&1 &
addts.sh iordr2_1 \?/dbs/tpcc_disks/iordr2_1_0 4000M 100M > junk29 2>&1 &
addts.sh iordr2_2 \?/dbs/tpcc_disks/iordr2_2_0 4000M 100M > junk30 2>&1 &
addts.sh iordr2_3 \?/dbs/tpcc_disks/iordr2_3_0 4000M 100M > junk31 2>&1 &
addts.sh iordr2_4 \?/dbs/tpcc_disks/iordr2_4_0 4000M 100M > junk32 2>&1 &
addts.sh iordr2_5 \?/dbs/tpcc_disks/iordr2_5_0 4000M 100M > junk33 2>&1 &
addts.sh iordr2_6 \?/dbs/tpcc_disks/iordr2_6_0 4000M 100M > junk34 2>&1 &
addts.sh iordr2_7 \?/dbs/tpcc_disks/iordr2_7_0 4000M 100M > junk35 2>&1 &
addts.sh nord_0 \?/dbs/tpcc_disks/nord_0_0 447M 10M > junk36 2>&1 &
addts.sh nord_1 \?/dbs/tpcc_disks/nord_1_0 447M 10M > junk37 2>&1 &
addts.sh nord_2 \?/dbs/tpcc_disks/nord_2_0 447M 10M > junk38 2>&1 &
addts.sh nord_3 \?/dbs/tpcc_disks/nord_3_0 447M 10M > junk39 2>&1 &
addts.sh nord_4 \?/dbs/tpcc_disks/nord_4_0 447M 10M > junk40 2>&1 &
addts.sh nord_5 \?/dbs/tpcc_disks/nord_5_0 447M 10M > junk41 2>&1 &
addts.sh nord_6 \?/dbs/tpcc_disks/nord_6_0 447M 10M > junk42 2>&1 &
addts.sh nord_7 \?/dbs/tpcc_disks/nord_7_0 447M 10M > junk43 2>&1 &
addts.sh dist_0 \?/dbs/tpcc_disks/dist_0_0 36M 1M > junk44 2>&1 &
addts.sh dist_1 \?/dbs/tpcc_disks/dist_1_0 36M 1M > junk45 2>&1 &
addts.sh dist_2 \?/dbs/tpcc_disks/dist_2_0 36M 1M > junk46 2>&1 &
addts.sh dist_3 \?/dbs/tpcc_disks/dist_3_0 36M 1M > junk47 2>&1 &
addts.sh dist_4 \?/dbs/tpcc_disks/dist_4_0 36M 1M > junk48 2>&1 &
addts.sh dist_5 \?/dbs/tpcc_disks/dist_5_0 36M 1M > junk49 2>&1 &
addts.sh dist_6 \?/dbs/tpcc_disks/dist_6_0 36M 1M > junk50 2>&1 &
addts.sh dist_7 \?/dbs/tpcc_disks/dist_7_0 36M 1M > junk51 2>&1 &
addtempts.sh temp_0 \?/dbs/tpcc_disks/temp_0_0 7500M 100M > junk52 2>&1 &
addts.sh temperm2 \?/dbs/tpcc_disks/temp_perm 7500M 100K > junk716 2>&1 &
addts.sh temperm3 \?/dbs/tpcc_disks/temp_perm_2 6000M 100K > junk716 2>&1 &
addts.sh temperm4 \?/dbs/tpcc_disks/temp_perm_3 4000M 100K > junk716 2>&1 &
addts.sh ordr_0 \?/dbs/tpcc_disks/ordr_0_0 3800M 100M > junk53 2>&1 &
addts.sh ordr_1 \?/dbs/tpcc_disks/ordr_1_0 3800M 100M > junk54 2>&1 &
addts.sh ordr_2 \?/dbs/tpcc_disks/ordr_2_0 3800M 100M > junk55 2>&1 &
addts.sh ordr_3 \?/dbs/tpcc_disks/ordr_3_0 3800M 100M > junk56 2>&1 &
addts.sh ordr_4 \?/dbs/tpcc_disks/ordr_4_0 3800M 100M > junk57 2>&1 &
addts.sh ordr_5 \?/dbs/tpcc_disks/ordr_5_0 3800M 100M > junk58 2>&1 &
addts.sh ordr_6 \?/dbs/tpcc_disks/ordr_6_0 3800M 100M > junk59 2>&1 &
addts.sh ordr_7 \?/dbs/tpcc_disks/ordr_7_0 3800M 100M > junk60 2>&1 &
addts.sh hist_0 \?/dbs/tpcc_disks/hist_0_0 5006M 100M > junk61 2>&1 &
addts.sh hist_1 \?/dbs/tpcc_disks/hist_1_0 5006M 100M > junk62 2>&1 &
addts.sh hist_2 \?/dbs/tpcc_disks/hist_2_0 5006M 100M > junk63 2>&1 &
addts.sh hist_3 \?/dbs/tpcc_disks/hist_3_0 5006M 100M > junk64 2>&1 &
addts.sh hist_4 \?/dbs/tpcc_disks/hist_4_0 5006M 100M > junk65 2>&1 &
addts.sh hist_5 \?/dbs/tpcc_disks/hist_5_0 5006M 100M > junk66 2>&1 &
addts.sh hist_6 \?/dbs/tpcc_disks/hist_6_0 5006M 100M > junk67 2>&1 &
addts.sh hist_7 \?/dbs/tpcc_disks/hist_7_0 5006M 100M > junk68 2>&1 &
addts.sh iitem_0 \?/dbs/tpcc_disks/iitem_0_0 3M 355K > junk69 2>&1 &
addts.sh iware_0 \?/dbs/tpcc_disks/iware_0_0 2M 865K > junk70 2>&1 &
addts.sh iware_1 \?/dbs/tpcc_disks/iware_1_0 2M 865K > junk71 2>&1 &
addts.sh iware_2 \?/dbs/tpcc_disks/iware_2_0 2M 865K > junk72 2>&1 &
addts.sh iware_3 \?/dbs/tpcc_disks/iware_3_0 2M 865K > junk73 2>&1 &
addts.sh iware_4 \?/dbs/tpcc_disks/iware_4_0 2M 865K > junk74 2>&1 &
addts.sh iware_5 \?/dbs/tpcc_disks/iware_5_0 2M 865K > junk75 2>&1 &
addts.sh iware_6 \?/dbs/tpcc_disks/iware_6_0 2M 865K > junk76 2>&1 &
addts.sh iware_7 \?/dbs/tpcc_disks/iware_7_0 2M 865K > junk77 2>&1 &
addts.sh icust1_0 \?/dbs/tpcc_disks/icust1_0_0 1400M 100M > junk78 2>&1 &
addts.sh icust2_0 \?/dbs/tpcc_disks/icust2_0_0 4500M 100M > junk79 2>&1 &
addts.sh idist_0 \?/dbs/tpcc_disks/idist_0_0 12M 1M > junk80 2>&1 &
addts.sh idist_1 \?/dbs/tpcc_disks/idist_1_0 12M 1M > junk81 2>&1 &
addts.sh idist_2 \?/dbs/tpcc_disks/idist_2_0 12M 1M > junk82 2>&1 &
addts.sh idist_3 \?/dbs/tpcc_disks/idist_3_0 12M 1M > junk83 2>&1 &
addts.sh idist_4 \?/dbs/tpcc_disks/idist_4_0 12M 1M > junk84 2>&1 &
addts.sh idist_5 \?/dbs/tpcc_disks/idist_5_0 12M 1M > junk85 2>&1 &
addts.sh idist_6 \?/dbs/tpcc_disks/idist_6_0 12M 1M > junk86 2>&1 &
addts.sh idist_7 \?/dbs/tpcc_disks/idist_7_0 12M 1M > junk87 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_1 7200M > junk89 2>&1 &
```

```
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_2 7200M > junk90 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_3 7200M > junk91 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_4 7200M > junk92 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_5 7200M > junk93 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_6 7200M > junk94 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_7 7200M > junk95 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_8 7200M > junk96 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_9 7200M > junk97 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_10 7200M > junk98 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_11 7200M > junk99 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_12 7200M > junk100 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_13 7200M > junk101 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_14 7200M > junk102 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_15 7200M > junk103 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_16 7200M > junk104 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_17 7200M > junk105 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_18 7200M > junk106 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_19 7200M > junk107 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_20 7200M > junk108 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_21 7200M > junk109 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_22 7200M > junk110 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_23 7200M > junk111 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_24 7200M > junk112 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_25 7200M > junk113 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_26 7200M > junk114 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_27 7200M > junk115 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_28 7200M > junk116 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_29 7200M > junk117 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_30 7200M > junk118 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_31 7200M > junk119 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_32 7200M > junk120 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_33 7200M > junk121 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_34 7200M > junk122 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_35 7200M > junk123 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_36 7200M > junk124 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_37 7200M > junk125 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_38 7200M > junk126 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_39 7200M > junk127 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_40 7200M > junk128 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_41 7200M > junk129 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_42 7200M > junk130 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_43 7200M > junk131 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_44 7200M > junk132 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_45 7200M > junk133 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_46 7200M > junk134 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_47 7200M > junk135 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_48 7200M > junk136 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_49 7200M > junk137 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_50 7200M > junk138 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_51 7200M > junk139 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_52 7200M > junk140 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_53 7200M > junk141 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_54 7200M > junk142 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_55 7200M > junk143 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_56 7200M > junk144 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_57 7200M > junk145 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_58 7200M > junk146 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_59 7200M > junk147 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_60 7200M > junk148 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_61 7200M > junk149 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_62 7200M > junk150 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_63 7200M > junk151 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_64 7200M > junk152 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_65 7200M > junk153 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_66 7200M > junk154 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_67 7200M > junk155 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_68 7200M > junk156 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_69 7200M > junk157 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_70 7200M > junk158 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_71 7200M > junk159 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_72 7200M >>  junk159 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_73 7200M >> junk159 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_74 7200M >> junk159 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_75 7200M >> junk159 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_76 7200M >> junk159 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_77 7200M >> junk159 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_78 7200M >> junk159 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_79 7200M >> junk159 2>&1 &
addfile.sh stok_0 \?/dbs/tpcc_disks/stok_0_80 7200M >> junk159 2>&1 &
addfile.sh istok_0 \?/dbs/tpcc_disks/istok_0_1 3900M > junk360 2>&1 &
addfile.sh istok_0 \?/dbs/tpcc_disks/istok_0_2 3900M > junk361 2>&1 &
addfile.sh istok_0 \?/dbs/tpcc_disks/istok_0_3 3900M > junk362 2>&1 &
addfile.sh istok_0 \?/dbs/tpcc_disks/istok_0_4 3900M > junk363 2>&1 &
addfile.sh istok_0 \?/dbs/tpcc_disks/istok_0_5 3900M > junk364 2>&1 &
addfile.sh istok_0 \?/dbs/tpcc_disks/istok_0_6 3900M > junk365 2>&1 &
addfile.sh istok_0 \?/dbs/tpcc_disks/istok_0_7 3900M > junk366 2>&1 &
addfile.sh ordl_0 \?/dbs/tpcc_disks/ordl_0_1 7500M > junk374 2>&1 &
addfile.sh ordl_0 \?/dbs/tpcc_disks/ordl_0_2 7500M > junk375 2>&1 &
addfile.sh ordl_0 \?/dbs/tpcc_disks/ordl_0_3 7500M > junk376 2>&1 &
addfile.sh ordl_0 \?/dbs/tpcc_disks/ordl_0_4 7500M > junk377 2>&1 &
addfile.sh ordl_0 \?/dbs/tpcc_disks/ordl_0_5 7500M > junk378 2>&1 &
addfile.sh ordl_0 \?/dbs/tpcc_disks/ordl_0_6 7500M > junk379 2>&1 &
addfile.sh ordl_0 \?/dbs/tpcc_disks/ordl_0_7 7500M > junk380 2>&1 &
addfile.sh ordl_0 \?/dbs/tpcc_disks/ordl_0_8 7500M > junk381 2>&1 &
addfile.sh ordl_1 \?/dbs/tpcc_disks/ordl_1_1 7500M > junk382 2>&1 &
addfile.sh ordl_1 \?/dbs/tpcc_disks/ordl_1_2 7500M > junk383 2>&1 &
addfile.sh ordl_1 \?/dbs/tpcc_disks/ordl_1_3 7500M > junk384 2>&1 &
addfile.sh ordl_1 \?/dbs/tpcc_disks/ordl_1_4 7500M > junk385 2>&1 &
addfile.sh ordl_1 \?/dbs/tpcc_disks/ordl_1_5 7500M > junk386 2>&1 &
addfile.sh ordl_1 \?/dbs/tpcc_disks/ordl_1_6 7500M > junk387 2>&1 &
addfile.sh ordl_1 \?/dbs/tpcc_disks/ordl_1_7 7500M > junk388 2>&1 &
addfile.sh ordl_1 \?/dbs/tpcc_disks/ordl_1_8 7500M > junk389 2>&1 &
addfile.sh ordl_2 \?/dbs/tpcc_disks/ordl_2_1 7500M > junk390 2>&1 &
addfile.sh ordl_2 \?/dbs/tpcc_disks/ordl_2_2 7500M > junk391 2>&1 &
addfile.sh ordl_2 \?/dbs/tpcc_disks/ordl_2_3 7500M > junk392 2>&1 &
addfile.sh ordl_2 \?/dbs/tpcc_disks/ordl_2_4 7500M > junk393 2>&1 &
addfile.sh ordl_2 \?/dbs/tpcc_disks/ordl_2_5 7500M > junk394 2>&1 &
addfile.sh ordl_2 \?/dbs/tpcc_disks/ordl_2_6 7500M > junk395 2>&1 &
addfile.sh ordl_2 \?/dbs/tpcc_disks/ordl_2_7 7500M > junk396 2>&1 &
addfile.sh ordl_2 \?/dbs/tpcc_disks/ordl_2_8 7500M > junk397 2>&1 &
addfile.sh ordl_3 \?/dbs/tpcc_disks/ordl_3_1 7500M > junk398 2>&1 &
addfile.sh ordl_3 \?/dbs/tpcc_disks/ordl_3_2 7500M > junk399 2>&1 &
addfile.sh ordl_3 \?/dbs/tpcc_disks/ordl_3_3 7500M > junk400 2>&1 &
addfile.sh ordl_3 \?/dbs/tpcc_disks/ordl_3_4 7500M > junk401 2>&1 &
addfile.sh ordl_3 \?/dbs/tpcc_disks/ordl_3_5 7500M > junk402 2>&1 &
addfile.sh ordl_3 \?/dbs/tpcc_disks/ordl_3_6 7500M > junk403 2>&1 &
addfile.sh ordl_3 \?/dbs/tpcc_disks/ordl_3_7 7500M > junk404 2>&1 &
addfile.sh ordl_3 \?/dbs/tpcc_disks/ordl_3_8 7500M > junk405 2>&1 &
addfile.sh ordl_4 \?/dbs/tpcc_disks/ordl_4_1 7500M > junk406 2>&1 &
addfile.sh ordl_4 \?/dbs/tpcc_disks/ordl_4_2 7500M > junk407 2>&1 &
addfile.sh ordl_4 \?/dbs/tpcc_disks/ordl_4_3 7500M > junk408 2>&1 &
addfile.sh ordl_4 \?/dbs/tpcc_disks/ordl_4_4 7500M > junk409 2>&1 &
addfile.sh ordl_4 \?/dbs/tpcc_disks/ordl_4_5 7500M > junk410 2>&1 &
addfile.sh ordl_4 \?/dbs/tpcc_disks/ordl_4_6 7500M > junk411 2>&1 &
addfile.sh ordl_4 \?/dbs/tpcc_disks/ordl_4_7 7500M > junk412 2>&1 &
addfile.sh ordl_4 \?/dbs/tpcc_disks/ordl_4_8 7500M > junk413 2>&1 &
addfile.sh ordl_5 \?/dbs/tpcc_disks/ordl_5_1 7500M > junk414 2>&1 &
addfile.sh ordl_5 \?/dbs/tpcc_disks/ordl_5_2 7500M > junk415 2>&1 &
addfile.sh ordl_5 \?/dbs/tpcc_disks/ordl_5_3 7500M > junk416 2>&1 &
addfile.sh ordl_5 \?/dbs/tpcc_disks/ordl_5_4 7500M > junk417 2>&1 &
addfile.sh ordl_5 \?/dbs/tpcc_disks/ordl_5_5 7500M > junk418 2>&1 &
addfile.sh ordl_5 \?/dbs/tpcc_disks/ordl_5_6 7500M > junk419 2>&1 &
addfile.sh ordl_5 \?/dbs/tpcc_disks/ordl_5_7 7500M > junk420 2>&1 &
addfile.sh ordl_5 \?/dbs/tpcc_disks/ordl_5_8 7500M > junk421 2>&1 &
addfile.sh ordl_6 \?/dbs/tpcc_disks/ordl_6_1 7500M > junk422 2>&1 &
addfile.sh ordl_6 \?/dbs/tpcc_disks/ordl_6_2 7500M > junk423 2>&1 &
addfile.sh ordl_6 \?/dbs/tpcc_disks/ordl_6_3 7500M > junk424 2>&1 &
addfile.sh ordl_6 \?/dbs/tpcc_disks/ordl_6_4 7500M > junk425 2>&1 &
addfile.sh ordl_6 \?/dbs/tpcc_disks/ordl_6_5 7500M > junk426 2>&1 &
addfile.sh ordl_6 \?/dbs/tpcc_disks/ordl_6_6 7500M > junk427 2>&1 &
addfile.sh ordl_6 \?/dbs/tpcc_disks/ordl_6_7 7500M > junk428 2>&1 &
addfile.sh ordl_6 \?/dbs/tpcc_disks/ordl_6_8 7500M > junk429 2>&1 &
addfile.sh ordl_7 \?/dbs/tpcc_disks/ordl_7_1 7500M > junk430 2>&1 &
addfile.sh ordl_7 \?/dbs/tpcc_disks/ordl_7_2 7500M > junk431 2>&1 &
addfile.sh ordl_7 \?/dbs/tpcc_disks/ordl_7_3 7500M > junk432 2>&1 &
addfile.sh ordl_7 \?/dbs/tpcc_disks/ordl_7_4 7500M > junk433 2>&1 &
addfile.sh ordl_7 \?/dbs/tpcc_disks/ordl_7_5 7500M > junk434 2>&1 &
addfile.sh ordl_7 \?/dbs/tpcc_disks/ordl_7_6 7500M > junk435 2>&1 &
addfile.sh ordl_7 \?/dbs/tpcc_disks/ordl_7_7 7500M > junk436 2>&1 &
addfile.sh ordl_7 \?/dbs/tpcc_disks/ordl_7_8 7500M > junk437 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_1 7500M > junk438 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_2 7500M > junk439 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_3 7500M > junk440 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_4 7500M > junk441 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_5 7500M > junk442 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_6 7500M > junk443 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_7 7500M > junk444 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_8 7500M > junk445 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_9 7500M > junk446 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_10 7500M > junk447 2>&1 &
```

```
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_11 7500M > junk448 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_12 7500M > junk449 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_13 7500M > junk450 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_14 7500M > junk451 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_15 7500M > junk452 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_16 7500M > junk453 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_17 7500M > junk454 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_18 7500M > junk455 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_19 7500M > junk456 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_20 7500M > junk457 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_21 7500M > junk458 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_22 7500M > junk459 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_23 7500M > junk460 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_24 7500M > junk461 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_25 7500M > junk462 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_26 7500M > junk463 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_27 7500M > junk464 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_28 7500M > junk465 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_29 7500M > junk466 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_30 7500M > junk467 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_31 7500M > junk468 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_32 7500M > junk708 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_33 7500M > junk709 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_34 7500M > junk710 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_35 7500M > junk711 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_36 7500M > junk712 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_37 7500M > junk713 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_38 7500M > junk714 2>&1 &
addtempfile.sh temp_0 \?/dbs/tpcc_disks/temp_0_39 7500M > junk715 2>&1 &
addfile.sh icust1_0 \?/dbs/tpcc_disks/icust1_0_1 1400M > junk476 2>&1 &
addfile.sh icust1_0 \?/dbs/tpcc_disks/icust1_0_2 1400M > junk477 2>&1 &
addfile.sh icust1_0 \?/dbs/tpcc_disks/icust1_0_3 1400M > junk478 2>&1 &
addfile.sh icust1_0 \?/dbs/tpcc_disks/icust1_0_4 1400M > junk479 2>&1 &
addfile.sh icust1_0 \?/dbs/tpcc_disks/icust1_0_5 1400M > junk480 2>&1 &
addfile.sh icust1_0 \?/dbs/tpcc_disks/icust1_0_6 1400M > junk481 2>&1 &
addfile.sh icust1_0 \?/dbs/tpcc_disks/icust1_0_7 1400M > junk482 2>&1 &
addfile.sh icust2_0 \?/dbs/tpcc_disks/icust2_0_1 4500M > junk483 2>&1 &
addfile.sh icust2_0 \?/dbs/tpcc_disks/icust2_0_2 4500M > junk484 2>&1 &
addfile.sh icust2_0 \?/dbs/tpcc_disks/icust2_0_3 4500M > junk485 2>&1 &
addfile.sh icust2_0 \?/dbs/tpcc_disks/icust2_0_4 4500M > junk486 2>&1 &
addfile.sh icust2_0 \?/dbs/tpcc_disks/icust2_0_5 4500M > junk487 2>&1 &
addfile.sh icust2_0 \?/dbs/tpcc_disks/icust2_0_6 4500M > junk488 2>&1 &
addfile.sh icust2_0 \?/dbs/tpcc_disks/icust2_0_7 4500M > junk489 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_1 7200M > junk490 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_2 7200M > junk491 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_3 7200M > junk492 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_4 7200M > junk493 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_5 7200M > junk494 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_6 7200M > junk495 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_7 7200M > junk496 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_8 7200M > junk497 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_9 7200M > junk498 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_10 7200M > junk499 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_11 7200M > junk500 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_12 7200M > junk501 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_13 7200M > junk502 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_14 7200M > junk503 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_15 7200M > junk504 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_16 7200M > junk505 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_17 7200M > junk506 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_18 7200M > junk507 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_19 7200M > junk508 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_20 7200M > junk509 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_21 7200M > junk510 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_22 7200M > junk511 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_23 7200M > junk512 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_24 7200M > junk513 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_25 7200M > junk514 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_26 7200M > junk515 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_27 7200M > junk516 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_28 7200M > junk517 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_29 7200M > junk518 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_30 7200M > junk519 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_31 7200M > junk520 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_32 7200M > junk521 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_33 7200M > junk522 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_34 7200M > junk523 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_35 7200M > junk524 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_36 7200M > junk525 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_37 7200M > junk526 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_38 7200M > junk527 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_39 7200M > junk528 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_40 7200M > junk529 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_41 7200M > junk530 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_42 7200M > junk531 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_43 7200M > junk532 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_44 7200M > junk533 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_45 7200M > junk534 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_46 7200M > junk535 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_47 7200M > junk536 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_48 7200M > junk537 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_49 7200M > junk538 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_50 7200M > junk539 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_51 7200M > junk540 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_52 7200M > junk541 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_53 7200M > junk542 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_54 7200M > junk543 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_55 7200M > junk544 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_56 7200M > junk544 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_57 7200M > junk544 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_58 7200M > junk544 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_59 7200M > junk544 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_60 7200M > junk544 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_61 7200M > junk544 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_62 7200M > junk544 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_63 7200M > junk544 2>&1 &
addfile.sh cust_0 \?/dbs/tpcc_disks/cust_0_64 7200M > junk544 2>&1 &
```

step6createddviews.sh

```
#!sh
$SEQLPLUS internal/internal @step6createddviews > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step6createddviews.sql

```
spool step6createddviews.log
@$ORACLE_HOME/rdbms/admin/catalog
@$ORACLE_HOME/rdbms/admin/catproc
@$ORACLE_HOME/rdbms/admin/catparr
spool off
exit sql.sqlcode;
```

step9createware.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @step9createware > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

step9createware.sql

```
spool step9createware.log;
set echo on;
drop table ware;
```

```
set timing on;

create table ware(
        w_id            number(5,0),
        w_ytd           number,
        w_tax           number,
        w_name          varchar2(10),
        w_street_1      varchar2(20),
        w_street_2      varchar2(20),
        w_city          varchar2(20),
        w_state         char(2),
        w_zip           char(9)
)
  partition by range(w_id)
  (
    partition ware_0 values less than (1689)
      tablespace ware_0,
    partition ware_1 values less than (3377)
      tablespace ware_1,
    partition ware_2 values less than (5065)
      tablespace ware_2,
    partition ware_3 values less than (6753)
      tablespace ware_3,
    partition ware_4 values less than (8441)
      tablespace ware_4,
    partition ware_5 values less than (10129)
      tablespace ware_5,
    partition ware_6 values less than (11817)
      tablespace ware_6,
    partition ware_7 values less than (maxvalue)
      tablespace ware_7
  )
  initrans    2
  pctfree    98
  pctused    0
  storage (  freelists 22 freelist groups 43 );
spool off;
set echo off;
exit sql.sqlcode;
```

stepchkpt.sh

```
#!sh

$SEQLPLUS internal/internal @stepchkpt
```

stepchkpt.sql

```
spool stepchkpt.log;

set echo on;

alter system switch logfile;
alter system switch logfile;

set echo off;
spool off;

exit sql.sqlcode;
```

stepcreateuser.sh

```
#!sh

$SEQLPLUS  internal/internal @stepcreateuser > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

stepcreateuser.sql

```
spool stepcreateuser.log;

set echo on;

create user tpcc identified by tpcc;

grant dba to tpcc;

set echo off;
spool off;

exit sql.sqlcode;
```

stepshut.sh

```
#!sh

$SEQLPLUS internal/internal @stepshut > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

stepshut.sql

```
spool stepshut.log;

set echo on;

alter system switch logfile;
alter system switch logfile;

shutdown immediate;

set echo off;
spool off;

exit sql.sqlcode;
```

stepstartb.sh

```
#!sh

$SEQLPLUS internal/internal @stepstartb > junk 2>&1

if test $? -eq 0
then
  exit 1;
else
  exit 0;
fi
```

stepstartb.sql

```
spool stepstartb.log;

set echo on;

startup pfile=p_build.ora open;

set echo off;
spool off;

exit sql.sqlcode;
```

stepusertemp.sh

```
#!sh

$SEQLPLUS internal/internal @stepusertemp > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

stepusertemp.sql

```
spool stepusertemp.log;

set echo on;

alter user tpcc temporary tablespace temp_0;

set echo off;
spool off;

exit sql.sqlcode;
```

tpcc_roll_0_1.sql

```
create rollback segment t_0_1 tablespace roll_0;
create rollback segment t_0_2 tablespace roll_0;
create rollback segment t_0_3 tablespace roll_0;
create rollback segment t_0_4 tablespace roll_0;
create rollback segment t_0_5 tablespace roll_0;
create rollback segment t_0_6 tablespace roll_0;
create rollback segment t_0_7 tablespace roll_0;
create rollback segment t_0_8 tablespace roll_0;
create rollback segment t_0_9 tablespace roll_0;
create rollback segment t_0_10 tablespace roll_0;
create rollback segment t_0_11 tablespace roll_0;
create rollback segment t_0_12 tablespace roll_0;
create rollback segment t_0_13 tablespace roll_0;
create rollback segment t_0_14 tablespace roll_0;
create rollback segment t_0_15 tablespace roll_0;
create rollback segment t_0_16 tablespace roll_0;
create rollback segment t_0_17 tablespace roll_0;
create rollback segment t_0_18 tablespace roll_0;
create rollback segment t_0_19 tablespace roll_0;
create rollback segment t_0_20 tablespace roll_0;
create rollback segment t_0_21 tablespace roll_0;
create rollback segment t_0_22 tablespace roll_0;
create rollback segment t_0_23 tablespace roll_0;
create rollback segment t_0_24 tablespace roll_0;
create rollback segment t_0_25 tablespace roll_0;
create rollback segment t_0_26 tablespace roll_0;
create rollback segment t_0_27 tablespace roll_0;
create rollback segment t_0_28 tablespace roll_0;
create rollback segment t_0_29 tablespace roll_0;
create rollback segment t_0_30 tablespace roll_0;
create rollback segment t_0_31 tablespace roll_0;
create rollback segment t_0_32 tablespace roll_0;
create rollback segment t_0_33 tablespace roll_0;
create rollback segment t_0_34 tablespace roll_0;
create rollback segment t_0_35 tablespace roll_0;
create rollback segment t_0_36 tablespace roll_0;
create rollback segment t_0_37 tablespace roll_0;
create rollback segment t_0_38 tablespace roll_0;
create rollback segment t_0_39 tablespace roll_0;
create rollback segment t_0_40 tablespace roll_0;
create rollback segment t_0_41 tablespace roll_0;
create rollback segment t_0_42 tablespace roll_0;
create rollback segment t_0_43 tablespace roll_0;
create rollback segment t_0_44 tablespace roll_0;
create rollback segment t_0_45 tablespace roll_0;
create rollback segment t_0_46 tablespace roll_0;
create rollback segment t_0_47 tablespace roll_0;
create rollback segment t_0_48 tablespace roll_0;
create rollback segment t_0_49 tablespace roll_0;
create rollback segment t_0_50 tablespace roll_0;
exit;
```

tpcc_roll_0_2.sql

```
create rollback segment t_0_51 tablespace roll_0;
create rollback segment t_0_52 tablespace roll_0;
create rollback segment t_0_53 tablespace roll_0;
create rollback segment t_0_54 tablespace roll_0;
create rollback segment t_0_55 tablespace roll_0;
create rollback segment t_0_56 tablespace roll_0;
create rollback segment t_0_57 tablespace roll_0;
create rollback segment t_0_58 tablespace roll_0;
create rollback segment t_0_59 tablespace roll_0;
create rollback segment t_0_60 tablespace roll_0;
create rollback segment t_0_61 tablespace roll_0;
create rollback segment t_0_62 tablespace roll_0;
create rollback segment t_0_63 tablespace roll_0;
create rollback segment t_0_64 tablespace roll_0;
create rollback segment t_0_65 tablespace roll_0;
create rollback segment t_0_66 tablespace roll_0;
create rollback segment t_0_67 tablespace roll_0;
create rollback segment t_0_68 tablespace roll_0;
create rollback segment t_0_69 tablespace roll_0;
create rollback segment t_0_70 tablespace roll_0;
create rollback segment t_0_71 tablespace roll_0;
create rollback segment t_0_72 tablespace roll_0;
create rollback segment t_0_73 tablespace roll_0;
create rollback segment t_0_74 tablespace roll_0;
create rollback segment t_0_75 tablespace roll_0;
create rollback segment t_0_76 tablespace roll_0;
create rollback segment t_0_77 tablespace roll_0;
create rollback segment t_0_78 tablespace roll_0;
create rollback segment t_0_79 tablespace roll_0;
create rollback segment t_0_80 tablespace roll_0;
create rollback segment t_0_81 tablespace roll_0;
create rollback segment t_0_82 tablespace roll_0;
create rollback segment t_0_83 tablespace roll_0;
create rollback segment t_0_84 tablespace roll_0;
create rollback segment t_0_85 tablespace roll_0;
create rollback segment t_0_86 tablespace roll_0;
create rollback segment t_0_87 tablespace roll_0;
create rollback segment t_0_88 tablespace roll_0;
create rollback segment t_0_89 tablespace roll_0;
create rollback segment t_0_90 tablespace roll_0;
create rollback segment t_0_91 tablespace roll_0;
create rollback segment t_0_92 tablespace roll_0;
create rollback segment t_0_93 tablespace roll_0;
create rollback segment t_0_94 tablespace roll_0;
create rollback segment t_0_95 tablespace roll_0;
create rollback segment t_0_96 tablespace roll_0;
create rollback segment t_0_97 tablespace roll_0;
create rollback segment t_0_98 tablespace roll_0;
create rollback segment t_0_99 tablespace roll_0;
create rollback segment t_0_100 tablespace roll_0;
exit;
```

tpcc_roll_0_3.sql

```
create rollback segment t_0_101 tablespace roll_0;
create rollback segment t_0_102 tablespace roll_0;
create rollback segment t_0_103 tablespace roll_0;
create rollback segment t_0_104 tablespace roll_0;
create rollback segment t_0_105 tablespace roll_0;
create rollback segment t_0_106 tablespace roll_0;
create rollback segment t_0_107 tablespace roll_0;
create rollback segment t_0_108 tablespace roll_0;
create rollback segment t_0_109 tablespace roll_0;
create rollback segment t_0_110 tablespace roll_0;
create rollback segment t_0_111 tablespace roll_0;
create rollback segment t_0_112 tablespace roll_0;
create rollback segment t_0_113 tablespace roll_0;
create rollback segment t_0_114 tablespace roll_0;
create rollback segment t_0_115 tablespace roll_0;
create rollback segment t_0_116 tablespace roll_0;
create rollback segment t_0_117 tablespace roll_0;
create rollback segment t_0_118 tablespace roll_0;
create rollback segment t_0_119 tablespace roll_0;
create rollback segment t_0_120 tablespace roll_0;
create rollback segment t_0_121 tablespace roll_0;
create rollback segment t_0_122 tablespace roll_0;
create rollback segment t_0_123 tablespace roll_0;
create rollback segment t_0_124 tablespace roll_0;
create rollback segment t_0_125 tablespace roll_0;
create rollback segment t_0_126 tablespace roll_0;
create rollback segment t_0_127 tablespace roll_0;
create rollback segment t_0_128 tablespace roll_0;
create rollback segment t_0_129 tablespace roll_0;
create rollback segment t_0_130 tablespace roll_0;
create rollback segment t_0_131 tablespace roll_0;
create rollback segment t_0_132 tablespace roll_0;
create rollback segment t_0_133 tablespace roll_0;
create rollback segment t_0_134 tablespace roll_0;
create rollback segment t_0_135 tablespace roll_0;
create rollback segment t_0_136 tablespace roll_0;
create rollback segment t_0_137 tablespace roll_0;
create rollback segment t_0_138 tablespace roll_0;
create rollback segment t_0_139 tablespace roll_0;
create rollback segment t_0_140 tablespace roll_0;
create rollback segment t_0_141 tablespace roll_0;
create rollback segment t_0_142 tablespace roll_0;
create rollback segment t_0_143 tablespace roll_0;
create rollback segment t_0_144 tablespace roll_0;
create rollback segment t_0_145 tablespace roll_0;
create rollback segment t_0_146 tablespace roll_0;
create rollback segment t_0_147 tablespace roll_0;
create rollback segment t_0_148 tablespace roll_0;
create rollback segment t_0_149 tablespace roll_0;
create rollback segment t_0_150 tablespace roll_0;
exit;
```

tpcc_roll_0_4.sql

```
create rollback segment t_0_151 tablespace roll_0;
create rollback segment t_0_152 tablespace roll_0;
create rollback segment t_0_153 tablespace roll_0;
create rollback segment t_0_154 tablespace roll_0;
create rollback segment t_0_155 tablespace roll_0;
create rollback segment t_0_156 tablespace roll_0;
create rollback segment t_0_157 tablespace roll_0;
create rollback segment t_0_158 tablespace roll_0;
create rollback segment t_0_159 tablespace roll_0;
create rollback segment t_0_160 tablespace roll_0;
create rollback segment t_0_161 tablespace roll_0;
create rollback segment t_0_162 tablespace roll_0;
create rollback segment t_0_163 tablespace roll_0;
create rollback segment t_0_164 tablespace roll_0;
create rollback segment t_0_165 tablespace roll_0;
create rollback segment t_0_166 tablespace roll_0;
create rollback segment t_0_167 tablespace roll_0;
create rollback segment t_0_168 tablespace roll_0;
create rollback segment t_0_169 tablespace roll_0;
create rollback segment t_0_170 tablespace roll_0;
create rollback segment t_0_171 tablespace roll_0;
create rollback segment t_0_172 tablespace roll_0;
create rollback segment t_0_173 tablespace roll_0;
create rollback segment t_0_174 tablespace roll_0;
create rollback segment t_0_175 tablespace roll_0;
create rollback segment t_0_176 tablespace roll_0;
create rollback segment t_0_177 tablespace roll_0;
create rollback segment t_0_178 tablespace roll_0;
create rollback segment t_0_179 tablespace roll_0;
create rollback segment t_0_180 tablespace roll_0;
create rollback segment t_0_181 tablespace roll_0;
create rollback segment t_0_182 tablespace roll_0;
create rollback segment t_0_183 tablespace roll_0;
create rollback segment t_0_184 tablespace roll_0;
create rollback segment t_0_185 tablespace roll_0;
create rollback segment t_0_186 tablespace roll_0;
create rollback segment t_0_187 tablespace roll_0;
create rollback segment t_0_188 tablespace roll_0;
create rollback segment t_0_189 tablespace roll_0;
create rollback segment t_0_190 tablespace roll_0;
create rollback segment t_0_191 tablespace roll_0;
create rollback segment t_0_192 tablespace roll_0;
create rollback segment t_0_193 tablespace roll_0;
create rollback segment t_0_194 tablespace roll_0;
create rollback segment t_0_195 tablespace roll_0;
create rollback segment t_0_196 tablespace roll_0;
create rollback segment t_0_197 tablespace roll_0;
create rollback segment t_0_198 tablespace roll_0;
create rollback segment t_0_199 tablespace roll_0;
create rollback segment t_0_200 tablespace roll_0;
exit;
```

tpcc_roll_1_1.sql

```
create rollback segment t_1_1 tablespace roll_1;
create rollback segment t_1_2 tablespace roll_1;
create rollback segment t_1_3 tablespace roll_1;
create rollback segment t_1_4 tablespace roll_1;
create rollback segment t_1_5 tablespace roll_1;
create rollback segment t_1_6 tablespace roll_1;
create rollback segment t_1_7 tablespace roll_1;
create rollback segment t_1_8 tablespace roll_1;
create rollback segment t_1_9 tablespace roll_1;
create rollback segment t_1_10 tablespace roll_1;
create rollback segment t_1_11 tablespace roll_1;
create rollback segment t_1_12 tablespace roll_1;
create rollback segment t_1_13 tablespace roll_1;
create rollback segment t_1_14 tablespace roll_1;
create rollback segment t_1_15 tablespace roll_1;
create rollback segment t_1_16 tablespace roll_1;
create rollback segment t_1_17 tablespace roll_1;
create rollback segment t_1_18 tablespace roll_1;
create rollback segment t_1_19 tablespace roll_1;
create rollback segment t_1_20 tablespace roll_1;
create rollback segment t_1_21 tablespace roll_1;
create rollback segment t_1_22 tablespace roll_1;
create rollback segment t_1_23 tablespace roll_1;
create rollback segment t_1_24 tablespace roll_1;
create rollback segment t_1_25 tablespace roll_1;
create rollback segment t_1_26 tablespace roll_1;
create rollback segment t_1_27 tablespace roll_1;
create rollback segment t_1_28 tablespace roll_1;
create rollback segment t_1_29 tablespace roll_1;
create rollback segment t_1_30 tablespace roll_1;
create rollback segment t_1_31 tablespace roll_1;
create rollback segment t_1_32 tablespace roll_1;
create rollback segment t_1_33 tablespace roll_1;
create rollback segment t_1_34 tablespace roll_1;
create rollback segment t_1_35 tablespace roll_1;
create rollback segment t_1_36 tablespace roll_1;
create rollback segment t_1_37 tablespace roll_1;
create rollback segment t_1_38 tablespace roll_1;
create rollback segment t_1_39 tablespace roll_1;
create rollback segment t_1_40 tablespace roll_1;
```

```
create rollback segment t_1_41 tablespace roll_1;
create rollback segment t_1_42 tablespace roll_1;
create rollback segment t_1_43 tablespace roll_1;
create rollback segment t_1_44 tablespace roll_1;
create rollback segment t_1_45 tablespace roll_1;
create rollback segment t_1_46 tablespace roll_1;
create rollback segment t_1_47 tablespace roll_1;
create rollback segment t_1_48 tablespace roll_1;
create rollback segment t_1_49 tablespace roll_1;
create rollback segment t_1_50 tablespace roll_1;
exit;
```

tpcc_roll_1_2.sql

```
create rollback segment t_1_51 tablespace roll_1;
create rollback segment t_1_52 tablespace roll_1;
create rollback segment t_1_53 tablespace roll_1;
create rollback segment t_1_54 tablespace roll_1;
create rollback segment t_1_55 tablespace roll_1;
create rollback segment t_1_56 tablespace roll_1;
create rollback segment t_1_57 tablespace roll_1;
create rollback segment t_1_58 tablespace roll_1;
create rollback segment t_1_59 tablespace roll_1;
create rollback segment t_1_60 tablespace roll_1;
create rollback segment t_1_61 tablespace roll_1;
create rollback segment t_1_62 tablespace roll_1;
create rollback segment t_1_63 tablespace roll_1;
create rollback segment t_1_64 tablespace roll_1;
create rollback segment t_1_65 tablespace roll_1;
create rollback segment t_1_66 tablespace roll_1;
create rollback segment t_1_67 tablespace roll_1;
create rollback segment t_1_68 tablespace roll_1;
create rollback segment t_1_69 tablespace roll_1;
create rollback segment t_1_70 tablespace roll_1;
create rollback segment t_1_71 tablespace roll_1;
create rollback segment t_1_72 tablespace roll_1;
create rollback segment t_1_73 tablespace roll_1;
create rollback segment t_1_74 tablespace roll_1;
create rollback segment t_1_75 tablespace roll_1;
create rollback segment t_1_76 tablespace roll_1;
create rollback segment t_1_77 tablespace roll_1;
create rollback segment t_1_78 tablespace roll_1;
create rollback segment t_1_79 tablespace roll_1;
create rollback segment t_1_80 tablespace roll_1;
create rollback segment t_1_81 tablespace roll_1;
create rollback segment t_1_82 tablespace roll_1;
create rollback segment t_1_83 tablespace roll_1;
create rollback segment t_1_84 tablespace roll_1;
create rollback segment t_1_85 tablespace roll_1;
create rollback segment t_1_86 tablespace roll_1;
create rollback segment t_1_87 tablespace roll_1;
create rollback segment t_1_88 tablespace roll_1;
create rollback segment t_1_89 tablespace roll_1;
create rollback segment t_1_90 tablespace roll_1;
create rollback segment t_1_91 tablespace roll_1;
create rollback segment t_1_92 tablespace roll_1;
create rollback segment t_1_93 tablespace roll_1;
create rollback segment t_1_94 tablespace roll_1;
create rollback segment t_1_95 tablespace roll_1;
create rollback segment t_1_96 tablespace roll_1;
create rollback segment t_1_97 tablespace roll_1;
create rollback segment t_1_98 tablespace roll_1;
create rollback segment t_1_99 tablespace roll_1;
create rollback segment t_1_100 tablespace roll_1;
exit;
```

tpcc_roll_1_3.sql

```
create rollback segment t_1_101 tablespace roll_1;
create rollback segment t_1_102 tablespace roll_1;
create rollback segment t_1_103 tablespace roll_1;
create rollback segment t_1_104 tablespace roll_1;
create rollback segment t_1_105 tablespace roll_1;
create rollback segment t_1_106 tablespace roll_1;
create rollback segment t_1_107 tablespace roll_1;
create rollback segment t_1_108 tablespace roll_1;
create rollback segment t_1_109 tablespace roll_1;
create rollback segment t_1_110 tablespace roll_1;
create rollback segment t_1_111 tablespace roll_1;
create rollback segment t_1_112 tablespace roll_1;
create rollback segment t_1_113 tablespace roll_1;
create rollback segment t_1_114 tablespace roll_1;
create rollback segment t_1_115 tablespace roll_1;
create rollback segment t_1_116 tablespace roll_1;
create rollback segment t_1_117 tablespace roll_1;
create rollback segment t_1_118 tablespace roll_1;
create rollback segment t_1_119 tablespace roll_1;
create rollback segment t_1_120 tablespace roll_1;
create rollback segment t_1_121 tablespace roll_1;
create rollback segment t_1_122 tablespace roll_1;
create rollback segment t_1_123 tablespace roll_1;
create rollback segment t_1_124 tablespace roll_1;
create rollback segment t_1_125 tablespace roll_1;
create rollback segment t_1_126 tablespace roll_1;
create rollback segment t_1_127 tablespace roll_1;
create rollback segment t_1_128 tablespace roll_1;
create rollback segment t_1_129 tablespace roll_1;
create rollback segment t_1_130 tablespace roll_1;
create rollback segment t_1_131 tablespace roll_1;
create rollback segment t_1_132 tablespace roll_1;
create rollback segment t_1_133 tablespace roll_1;
create rollback segment t_1_134 tablespace roll_1;
create rollback segment t_1_135 tablespace roll_1;
create rollback segment t_1_136 tablespace roll_1;
create rollback segment t_1_137 tablespace roll_1;
create rollback segment t_1_138 tablespace roll_1;
create rollback segment t_1_139 tablespace roll_1;
create rollback segment t_1_140 tablespace roll_1;
create rollback segment t_1_141 tablespace roll_1;
create rollback segment t_1_142 tablespace roll_1;
create rollback segment t_1_143 tablespace roll_1;
create rollback segment t_1_144 tablespace roll_1;
create rollback segment t_1_145 tablespace roll_1;
create rollback segment t_1_146 tablespace roll_1;
create rollback segment t_1_147 tablespace roll_1;
create rollback segment t_1_148 tablespace roll_1;
create rollback segment t_1_149 tablespace roll_1;
create rollback segment t_1_150 tablespace roll_1;
exit;
```

tpcc_roll_1_4.sql

```
create rollback segment t_1_151 tablespace roll_1;
create rollback segment t_1_152 tablespace roll_1;
create rollback segment t_1_153 tablespace roll_1;
create rollback segment t_1_154 tablespace roll_1;
create rollback segment t_1_155 tablespace roll_1;
create rollback segment t_1_156 tablespace roll_1;
create rollback segment t_1_157 tablespace roll_1;
create rollback segment t_1_158 tablespace roll_1;
create rollback segment t_1_159 tablespace roll_1;
create rollback segment t_1_160 tablespace roll_1;
create rollback segment t_1_161 tablespace roll_1;
create rollback segment t_1_162 tablespace roll_1;
create rollback segment t_1_163 tablespace roll_1;
create rollback segment t_1_164 tablespace roll_1;
create rollback segment t_1_165 tablespace roll_1;
create rollback segment t_1_166 tablespace roll_1;
create rollback segment t_1_167 tablespace roll_1;
create rollback segment t_1_168 tablespace roll_1;
create rollback segment t_1_169 tablespace roll_1;
create rollback segment t_1_170 tablespace roll_1;
create rollback segment t_1_171 tablespace roll_1;
create rollback segment t_1_172 tablespace roll_1;
create rollback segment t_1_173 tablespace roll_1;
create rollback segment t_1_174 tablespace roll_1;
create rollback segment t_1_175 tablespace roll_1;
create rollback segment t_1_176 tablespace roll_1;
create rollback segment t_1_177 tablespace roll_1;
create rollback segment t_1_178 tablespace roll_1;
```

```
create rollback segment t_1_179 tablespace roll_1;
create rollback segment t_1_180 tablespace roll_1;
create rollback segment t_1_181 tablespace roll_1;
create rollback segment t_1_182 tablespace roll_1;
create rollback segment t_1_183 tablespace roll_1;
create rollback segment t_1_184 tablespace roll_1;
create rollback segment t_1_185 tablespace roll_1;
create rollback segment t_1_186 tablespace roll_1;
create rollback segment t_1_187 tablespace roll_1;
create rollback segment t_1_188 tablespace roll_1;
create rollback segment t_1_189 tablespace roll_1;
create rollback segment t_1_190 tablespace roll_1;
create rollback segment t_1_191 tablespace roll_1;
create rollback segment t_1_192 tablespace roll_1;
create rollback segment t_1_193 tablespace roll_1;
create rollback segment t_1_194 tablespace roll_1;
create rollback segment t_1_195 tablespace roll_1;
create rollback segment t_1_196 tablespace roll_1;
create rollback segment t_1_197 tablespace roll_1;
create rollback segment t_1_198 tablespace roll_1;
create rollback segment t_1_199 tablespace roll_1;
create rollback segment t_1_200 tablespace roll_1;
exit;
```

tpcc_roll_2_1.sql

```
create rollback segment t_2_1 tablespace roll_2;
create rollback segment t_2_2 tablespace roll_2;
create rollback segment t_2_3 tablespace roll_2;
create rollback segment t_2_4 tablespace roll_2;
create rollback segment t_2_5 tablespace roll_2;
create rollback segment t_2_6 tablespace roll_2;
create rollback segment t_2_7 tablespace roll_2;
create rollback segment t_2_8 tablespace roll_2;
create rollback segment t_2_9 tablespace roll_2;
create rollback segment t_2_10 tablespace roll_2;
create rollback segment t_2_11 tablespace roll_2;
create rollback segment t_2_12 tablespace roll_2;
create rollback segment t_2_13 tablespace roll_2;
create rollback segment t_2_14 tablespace roll_2;
create rollback segment t_2_15 tablespace roll_2;
create rollback segment t_2_16 tablespace roll_2;
create rollback segment t_2_17 tablespace roll_2;
create rollback segment t_2_18 tablespace roll_2;
create rollback segment t_2_19 tablespace roll_2;
create rollback segment t_2_20 tablespace roll_2;
create rollback segment t_2_21 tablespace roll_2;
create rollback segment t_2_22 tablespace roll_2;
create rollback segment t_2_23 tablespace roll_2;
create rollback segment t_2_24 tablespace roll_2;
create rollback segment t_2_25 tablespace roll_2;
create rollback segment t_2_26 tablespace roll_2;
create rollback segment t_2_27 tablespace roll_2;
create rollback segment t_2_28 tablespace roll_2;
create rollback segment t_2_29 tablespace roll_2;
create rollback segment t_2_30 tablespace roll_2;
create rollback segment t_2_31 tablespace roll_2;
create rollback segment t_2_32 tablespace roll_2;
create rollback segment t_2_33 tablespace roll_2;
create rollback segment t_2_34 tablespace roll_2;
create rollback segment t_2_35 tablespace roll_2;
create rollback segment t_2_36 tablespace roll_2;
create rollback segment t_2_37 tablespace roll_2;
create rollback segment t_2_38 tablespace roll_2;
create rollback segment t_2_39 tablespace roll_2;
create rollback segment t_2_40 tablespace roll_2;
create rollback segment t_2_41 tablespace roll_2;
create rollback segment t_2_42 tablespace roll_2;
create rollback segment t_2_43 tablespace roll_2;
create rollback segment t_2_44 tablespace roll_2;
create rollback segment t_2_45 tablespace roll_2;
create rollback segment t_2_46 tablespace roll_2;
create rollback segment t_2_47 tablespace roll_2;
create rollback segment t_2_48 tablespace roll_2;
create rollback segment t_2_49 tablespace roll_2;
create rollback segment t_2_50 tablespace roll_2;
exit;
```

tpcc_roll_2_2.sql

```
create rollback segment t_2_51 tablespace roll_2;
create rollback segment t_2_52 tablespace roll_2;
create rollback segment t_2_53 tablespace roll_2;
create rollback segment t_2_54 tablespace roll_2;
create rollback segment t_2_55 tablespace roll_2;
create rollback segment t_2_56 tablespace roll_2;
create rollback segment t_2_57 tablespace roll_2;
create rollback segment t_2_58 tablespace roll_2;
create rollback segment t_2_59 tablespace roll_2;
create rollback segment t_2_60 tablespace roll_2;
create rollback segment t_2_61 tablespace roll_2;
create rollback segment t_2_62 tablespace roll_2;
create rollback segment t_2_63 tablespace roll_2;
create rollback segment t_2_64 tablespace roll_2;
create rollback segment t_2_65 tablespace roll_2;
create rollback segment t_2_66 tablespace roll_2;
create rollback segment t_2_67 tablespace roll_2;
create rollback segment t_2_68 tablespace roll_2;
create rollback segment t_2_69 tablespace roll_2;
create rollback segment t_2_70 tablespace roll_2;
create rollback segment t_2_71 tablespace roll_2;
create rollback segment t_2_72 tablespace roll_2;
create rollback segment t_2_73 tablespace roll_2;
create rollback segment t_2_74 tablespace roll_2;
create rollback segment t_2_75 tablespace roll_2;
create rollback segment t_2_76 tablespace roll_2;
create rollback segment t_2_77 tablespace roll_2;
create rollback segment t_2_78 tablespace roll_2;
create rollback segment t_2_79 tablespace roll_2;
create rollback segment t_2_80 tablespace roll_2;
create rollback segment t_2_81 tablespace roll_2;
create rollback segment t_2_82 tablespace roll_2;
create rollback segment t_2_83 tablespace roll_2;
create rollback segment t_2_84 tablespace roll_2;
create rollback segment t_2_85 tablespace roll_2;
create rollback segment t_2_86 tablespace roll_2;
create rollback segment t_2_87 tablespace roll_2;
create rollback segment t_2_88 tablespace roll_2;
create rollback segment t_2_89 tablespace roll_2;
create rollback segment t_2_90 tablespace roll_2;
create rollback segment t_2_91 tablespace roll_2;
create rollback segment t_2_92 tablespace roll_2;
create rollback segment t_2_93 tablespace roll_2;
create rollback segment t_2_94 tablespace roll_2;
create rollback segment t_2_95 tablespace roll_2;
create rollback segment t_2_96 tablespace roll_2;
create rollback segment t_2_97 tablespace roll_2;
create rollback segment t_2_98 tablespace roll_2;
create rollback segment t_2_99 tablespace roll_2;
create rollback segment t_2_100 tablespace roll_2;
exit;
```

tpcc_roll_2_3.sql

```
create rollback segment t_2_101 tablespace roll_2;
create rollback segment t_2_102 tablespace roll_2;
create rollback segment t_2_103 tablespace roll_2;
create rollback segment t_2_104 tablespace roll_2;
create rollback segment t_2_105 tablespace roll_2;
create rollback segment t_2_106 tablespace roll_2;
create rollback segment t_2_107 tablespace roll_2;
create rollback segment t_2_108 tablespace roll_2;
create rollback segment t_2_109 tablespace roll_2;
create rollback segment t_2_110 tablespace roll_2;
create rollback segment t_2_111 tablespace roll_2;
create rollback segment t_2_112 tablespace roll_2;
create rollback segment t_2_113 tablespace roll_2;
create rollback segment t_2_114 tablespace roll_2;
create rollback segment t_2_115 tablespace roll_2;
create rollback segment t_2_116 tablespace roll_2;
create rollback segment t_2_117 tablespace roll_2;
```

```
create rollback segment t_2_118 tablespace roll_2;
create rollback segment t_2_119 tablespace roll_2;
create rollback segment t_2_120 tablespace roll_2;
create rollback segment t_2_121 tablespace roll_2;
create rollback segment t_2_122 tablespace roll_2;
create rollback segment t_2_123 tablespace roll_2;
create rollback segment t_2_124 tablespace roll_2;
create rollback segment t_2_125 tablespace roll_2;
create rollback segment t_2_126 tablespace roll_2;
create rollback segment t_2_127 tablespace roll_2;
create rollback segment t_2_128 tablespace roll_2;
create rollback segment t_2_129 tablespace roll_2;
create rollback segment t_2_130 tablespace roll_2;
create rollback segment t_2_131 tablespace roll_2;
create rollback segment t_2_132 tablespace roll_2;
create rollback segment t_2_133 tablespace roll_2;
create rollback segment t_2_134 tablespace roll_2;
create rollback segment t_2_135 tablespace roll_2;
create rollback segment t_2_136 tablespace roll_2;
create rollback segment t_2_137 tablespace roll_2;
create rollback segment t_2_138 tablespace roll_2;
create rollback segment t_2_139 tablespace roll_2;
create rollback segment t_2_140 tablespace roll_2;
create rollback segment t_2_141 tablespace roll_2;
create rollback segment t_2_142 tablespace roll_2;
create rollback segment t_2_143 tablespace roll_2;
create rollback segment t_2_144 tablespace roll_2;
create rollback segment t_2_145 tablespace roll_2;
create rollback segment t_2_146 tablespace roll_2;
create rollback segment t_2_147 tablespace roll_2;
create rollback segment t_2_148 tablespace roll_2;
create rollback segment t_2_149 tablespace roll_2;
create rollback segment t_2_150 tablespace roll_2;
exit;
```

tpcc_roll_2_4.sql

```
create rollback segment t_2_151 tablespace roll_2;
create rollback segment t_2_152 tablespace roll_2;
create rollback segment t_2_153 tablespace roll_2;
create rollback segment t_2_154 tablespace roll_2;
create rollback segment t_2_155 tablespace roll_2;
create rollback segment t_2_156 tablespace roll_2;
create rollback segment t_2_157 tablespace roll_2;
create rollback segment t_2_158 tablespace roll_2;
create rollback segment t_2_159 tablespace roll_2;
create rollback segment t_2_160 tablespace roll_2;
create rollback segment t_2_161 tablespace roll_2;
create rollback segment t_2_162 tablespace roll_2;
create rollback segment t_2_163 tablespace roll_2;
create rollback segment t_2_164 tablespace roll_2;
create rollback segment t_2_165 tablespace roll_2;
create rollback segment t_2_166 tablespace roll_2;
create rollback segment t_2_167 tablespace roll_2;
create rollback segment t_2_168 tablespace roll_2;
create rollback segment t_2_169 tablespace roll_2;
create rollback segment t_2_170 tablespace roll_2;
create rollback segment t_2_171 tablespace roll_2;
create rollback segment t_2_172 tablespace roll_2;
create rollback segment t_2_173 tablespace roll_2;
create rollback segment t_2_174 tablespace roll_2;
create rollback segment t_2_175 tablespace roll_2;
create rollback segment t_2_176 tablespace roll_2;
create rollback segment t_2_177 tablespace roll_2;
create rollback segment t_2_178 tablespace roll_2;
create rollback segment t_2_179 tablespace roll_2;
create rollback segment t_2_180 tablespace roll_2;
create rollback segment t_2_181 tablespace roll_2;
create rollback segment t_2_182 tablespace roll_2;
create rollback segment t_2_183 tablespace roll_2;
create rollback segment t_2_184 tablespace roll_2;
create rollback segment t_2_185 tablespace roll_2;
create rollback segment t_2_186 tablespace roll_2;
create rollback segment t_2_187 tablespace roll_2;
create rollback segment t_2_188 tablespace roll_2;
create rollback segment t_2_189 tablespace roll_2;
create rollback segment t_2_190 tablespace roll_2;
create rollback segment t_2_191 tablespace roll_2;
create rollback segment t_2_192 tablespace roll_2;
create rollback segment t_2_193 tablespace roll_2;
create rollback segment t_2_194 tablespace roll_2;
create rollback segment t_2_195 tablespace roll_2;
create rollback segment t_2_196 tablespace roll_2;
create rollback segment t_2_197 tablespace roll_2;
create rollback segment t_2_198 tablespace roll_2;
create rollback segment t_2_199 tablespace roll_2;
create rollback segment t_2_200 tablespace roll_2;
exit;
```

tpcc_roll_3_1.sql

```
create rollback segment t_3_1 tablespace roll_3;
create rollback segment t_3_2 tablespace roll_3;
create rollback segment t_3_3 tablespace roll_3;
create rollback segment t_3_4 tablespace roll_3;
create rollback segment t_3_5 tablespace roll_3;
create rollback segment t_3_6 tablespace roll_3;
create rollback segment t_3_7 tablespace roll_3;
create rollback segment t_3_8 tablespace roll_3;
create rollback segment t_3_9 tablespace roll_3;
create rollback segment t_3_10 tablespace roll_3;
create rollback segment t_3_11 tablespace roll_3;
create rollback segment t_3_12 tablespace roll_3;
create rollback segment t_3_13 tablespace roll_3;
create rollback segment t_3_14 tablespace roll_3;
create rollback segment t_3_15 tablespace roll_3;
create rollback segment t_3_16 tablespace roll_3;
create rollback segment t_3_17 tablespace roll_3;
create rollback segment t_3_18 tablespace roll_3;
create rollback segment t_3_19 tablespace roll_3;
create rollback segment t_3_20 tablespace roll_3;
create rollback segment t_3_21 tablespace roll_3;
create rollback segment t_3_22 tablespace roll_3;
create rollback segment t_3_23 tablespace roll_3;
create rollback segment t_3_24 tablespace roll_3;
create rollback segment t_3_25 tablespace roll_3;
create rollback segment t_3_26 tablespace roll_3;
create rollback segment t_3_27 tablespace roll_3;
create rollback segment t_3_28 tablespace roll_3;
create rollback segment t_3_29 tablespace roll_3;
create rollback segment t_3_30 tablespace roll_3;
create rollback segment t_3_31 tablespace roll_3;
create rollback segment t_3_32 tablespace roll_3;
create rollback segment t_3_33 tablespace roll_3;
create rollback segment t_3_34 tablespace roll_3;
create rollback segment t_3_35 tablespace roll_3;
create rollback segment t_3_36 tablespace roll_3;
create rollback segment t_3_37 tablespace roll_3;
create rollback segment t_3_38 tablespace roll_3;
create rollback segment t_3_39 tablespace roll_3;
create rollback segment t_3_40 tablespace roll_3;
create rollback segment t_3_41 tablespace roll_3;
create rollback segment t_3_42 tablespace roll_3;
create rollback segment t_3_43 tablespace roll_3;
create rollback segment t_3_44 tablespace roll_3;
create rollback segment t_3_45 tablespace roll_3;
create rollback segment t_3_46 tablespace roll_3;
create rollback segment t_3_47 tablespace roll_3;
create rollback segment t_3_48 tablespace roll_3;
create rollback segment t_3_49 tablespace roll_3;
create rollback segment t_3_50 tablespace roll_3;
exit;
```

tpcc_roll_3_2.sql

```
create rollback segment t_3_51 tablespace roll_3;
create rollback segment t_3_52 tablespace roll_3;
create rollback segment t_3_53 tablespace roll_3;
create rollback segment t_3_54 tablespace roll_3;
create rollback segment t_3_55 tablespace roll_3;
```

```
create rollback segment t_3_56 tablespace roll_3;
create rollback segment t_3_57 tablespace roll_3;
create rollback segment t_3_58 tablespace roll_3;
create rollback segment t_3_59 tablespace roll_3;
create rollback segment t_3_60 tablespace roll_3;
create rollback segment t_3_61 tablespace roll_3;
create rollback segment t_3_62 tablespace roll_3;
create rollback segment t_3_63 tablespace roll_3;
create rollback segment t_3_64 tablespace roll_3;
create rollback segment t_3_65 tablespace roll_3;
create rollback segment t_3_66 tablespace roll_3;
create rollback segment t_3_67 tablespace roll_3;
create rollback segment t_3_68 tablespace roll_3;
create rollback segment t_3_69 tablespace roll_3;
create rollback segment t_3_70 tablespace roll_3;
create rollback segment t_3_71 tablespace roll_3;
create rollback segment t_3_72 tablespace roll_3;
create rollback segment t_3_73 tablespace roll_3;
create rollback segment t_3_74 tablespace roll_3;
create rollback segment t_3_75 tablespace roll_3;
create rollback segment t_3_76 tablespace roll_3;
create rollback segment t_3_77 tablespace roll_3;
create rollback segment t_3_78 tablespace roll_3;
create rollback segment t_3_79 tablespace roll_3;
create rollback segment t_3_80 tablespace roll_3;
create rollback segment t_3_81 tablespace roll_3;
create rollback segment t_3_82 tablespace roll_3;
create rollback segment t_3_83 tablespace roll_3;
create rollback segment t_3_84 tablespace roll_3;
create rollback segment t_3_85 tablespace roll_3;
create rollback segment t_3_86 tablespace roll_3;
create rollback segment t_3_87 tablespace roll_3;
create rollback segment t_3_88 tablespace roll_3;
create rollback segment t_3_89 tablespace roll_3;
create rollback segment t_3_90 tablespace roll_3;
create rollback segment t_3_91 tablespace roll_3;
create rollback segment t_3_92 tablespace roll_3;
create rollback segment t_3_93 tablespace roll_3;
create rollback segment t_3_94 tablespace roll_3;
create rollback segment t_3_95 tablespace roll_3;
create rollback segment t_3_96 tablespace roll_3;
create rollback segment t_3_97 tablespace roll_3;
create rollback segment t_3_98 tablespace roll_3;
create rollback segment t_3_99 tablespace roll_3;
create rollback segment t_3_100 tablespace roll_3;
exit;
```

tpcc_roll_3_3.sql

```
create rollback segment t_3_101 tablespace roll_3;
create rollback segment t_3_102 tablespace roll_3;
create rollback segment t_3_103 tablespace roll_3;
create rollback segment t_3_104 tablespace roll_3;
create rollback segment t_3_105 tablespace roll_3;
create rollback segment t_3_106 tablespace roll_3;
create rollback segment t_3_107 tablespace roll_3;
create rollback segment t_3_108 tablespace roll_3;
create rollback segment t_3_109 tablespace roll_3;
create rollback segment t_3_110 tablespace roll_3;
create rollback segment t_3_111 tablespace roll_3;
create rollback segment t_3_112 tablespace roll_3;
create rollback segment t_3_113 tablespace roll_3;
create rollback segment t_3_114 tablespace roll_3;
create rollback segment t_3_115 tablespace roll_3;
create rollback segment t_3_116 tablespace roll_3;
create rollback segment t_3_117 tablespace roll_3;
create rollback segment t_3_118 tablespace roll_3;
create rollback segment t_3_119 tablespace roll_3;
create rollback segment t_3_120 tablespace roll_3;
create rollback segment t_3_121 tablespace roll_3;
create rollback segment t_3_122 tablespace roll_3;
create rollback segment t_3_123 tablespace roll_3;
create rollback segment t_3_124 tablespace roll_3;
create rollback segment t_3_125 tablespace roll_3;
create rollback segment t_3_126 tablespace roll_3;
create rollback segment t_3_127 tablespace roll_3;
create rollback segment t_3_128 tablespace roll_3;
create rollback segment t_3_129 tablespace roll_3;
create rollback segment t_3_130 tablespace roll_3;
create rollback segment t_3_131 tablespace roll_3;
create rollback segment t_3_132 tablespace roll_3;
create rollback segment t_3_133 tablespace roll_3;
create rollback segment t_3_134 tablespace roll_3;
create rollback segment t_3_135 tablespace roll_3;
create rollback segment t_3_136 tablespace roll_3;
create rollback segment t_3_137 tablespace roll_3;
create rollback segment t_3_138 tablespace roll_3;
create rollback segment t_3_139 tablespace roll_3;
create rollback segment t_3_140 tablespace roll_3;
create rollback segment t_3_141 tablespace roll_3;
create rollback segment t_3_142 tablespace roll_3;
create rollback segment t_3_143 tablespace roll_3;
create rollback segment t_3_144 tablespace roll_3;
create rollback segment t_3_145 tablespace roll_3;
create rollback segment t_3_146 tablespace roll_3;
create rollback segment t_3_147 tablespace roll_3;
create rollback segment t_3_148 tablespace roll_3;
create rollback segment t_3_149 tablespace roll_3;
create rollback segment t_3_150 tablespace roll_3;
exit;
```

tpcc_roll_3_4.sql

```
create rollback segment t_3_151 tablespace roll_3;
create rollback segment t_3_152 tablespace roll_3;
create rollback segment t_3_153 tablespace roll_3;
create rollback segment t_3_154 tablespace roll_3;
create rollback segment t_3_155 tablespace roll_3;
create rollback segment t_3_156 tablespace roll_3;
create rollback segment t_3_157 tablespace roll_3;
create rollback segment t_3_158 tablespace roll_3;
create rollback segment t_3_159 tablespace roll_3;
create rollback segment t_3_160 tablespace roll_3;
create rollback segment t_3_161 tablespace roll_3;
create rollback segment t_3_162 tablespace roll_3;
create rollback segment t_3_163 tablespace roll_3;
create rollback segment t_3_164 tablespace roll_3;
create rollback segment t_3_165 tablespace roll_3;
create rollback segment t_3_166 tablespace roll_3;
create rollback segment t_3_167 tablespace roll_3;
create rollback segment t_3_168 tablespace roll_3;
create rollback segment t_3_169 tablespace roll_3;
create rollback segment t_3_170 tablespace roll_3;
create rollback segment t_3_171 tablespace roll_3;
create rollback segment t_3_172 tablespace roll_3;
create rollback segment t_3_173 tablespace roll_3;
create rollback segment t_3_174 tablespace roll_3;
create rollback segment t_3_175 tablespace roll_3;
create rollback segment t_3_176 tablespace roll_3;
create rollback segment t_3_177 tablespace roll_3;
create rollback segment t_3_178 tablespace roll_3;
create rollback segment t_3_179 tablespace roll_3;
create rollback segment t_3_180 tablespace roll_3;
create rollback segment t_3_181 tablespace roll_3;
create rollback segment t_3_182 tablespace roll_3;
create rollback segment t_3_183 tablespace roll_3;
create rollback segment t_3_184 tablespace roll_3;
create rollback segment t_3_185 tablespace roll_3;
create rollback segment t_3_186 tablespace roll_3;
create rollback segment t_3_187 tablespace roll_3;
create rollback segment t_3_188 tablespace roll_3;
create rollback segment t_3_189 tablespace roll_3;
create rollback segment t_3_190 tablespace roll_3;
create rollback segment t_3_191 tablespace roll_3;
create rollback segment t_3_192 tablespace roll_3;
create rollback segment t_3_193 tablespace roll_3;
create rollback segment t_3_194 tablespace roll_3;
create rollback segment t_3_195 tablespace roll_3;
create rollback segment t_3_196 tablespace roll_3;
create rollback segment t_3_197 tablespace roll_3;
create rollback segment t_3_198 tablespace roll_3;
create rollback segment t_3_199 tablespace roll_3;
create rollback segment t_3_200 tablespace roll_3;
exit;
```

tpcc_roll_4_1.sql

```
create rollback segment t_4_1 tablespace roll_4;
create rollback segment t_4_2 tablespace roll_4;
create rollback segment t_4_3 tablespace roll_4;
create rollback segment t_4_4 tablespace roll_4;
create rollback segment t_4_5 tablespace roll_4;
create rollback segment t_4_6 tablespace roll_4;
create rollback segment t_4_7 tablespace roll_4;
create rollback segment t_4_8 tablespace roll_4;
create rollback segment t_4_9 tablespace roll_4;
create rollback segment t_4_10 tablespace roll_4;
create rollback segment t_4_11 tablespace roll_4;
create rollback segment t_4_12 tablespace roll_4;
create rollback segment t_4_13 tablespace roll_4;
create rollback segment t_4_14 tablespace roll_4;
create rollback segment t_4_15 tablespace roll_4;
create rollback segment t_4_16 tablespace roll_4;
create rollback segment t_4_17 tablespace roll_4;
create rollback segment t_4_18 tablespace roll_4;
create rollback segment t_4_19 tablespace roll_4;
create rollback segment t_4_20 tablespace roll_4;
create rollback segment t_4_21 tablespace roll_4;
create rollback segment t_4_22 tablespace roll_4;
create rollback segment t_4_23 tablespace roll_4;
create rollback segment t_4_24 tablespace roll_4;
create rollback segment t_4_25 tablespace roll_4;
create rollback segment t_4_26 tablespace roll_4;
create rollback segment t_4_27 tablespace roll_4;
create rollback segment t_4_28 tablespace roll_4;
create rollback segment t_4_29 tablespace roll_4;
create rollback segment t_4_30 tablespace roll_4;
create rollback segment t_4_31 tablespace roll_4;
create rollback segment t_4_32 tablespace roll_4;
create rollback segment t_4_33 tablespace roll_4;
create rollback segment t_4_34 tablespace roll_4;
create rollback segment t_4_35 tablespace roll_4;
create rollback segment t_4_36 tablespace roll_4;
create rollback segment t_4_37 tablespace roll_4;
create rollback segment t_4_38 tablespace roll_4;
create rollback segment t_4_39 tablespace roll_4;
create rollback segment t_4_40 tablespace roll_4;
create rollback segment t_4_41 tablespace roll_4;
create rollback segment t_4_42 tablespace roll_4;
create rollback segment t_4_43 tablespace roll_4;
create rollback segment t_4_44 tablespace roll_4;
create rollback segment t_4_45 tablespace roll_4;
create rollback segment t_4_46 tablespace roll_4;
create rollback segment t_4_47 tablespace roll_4;
create rollback segment t_4_48 tablespace roll_4;
create rollback segment t_4_49 tablespace roll_4;
create rollback segment t_4_50 tablespace roll_4;
exit;
```

tpcc_roll_4_2.sql

```
create rollback segment t_4_51 tablespace roll_4;
create rollback segment t_4_52 tablespace roll_4;
create rollback segment t_4_53 tablespace roll_4;
create rollback segment t_4_54 tablespace roll_4;
create rollback segment t_4_55 tablespace roll_4;
create rollback segment t_4_56 tablespace roll_4;
create rollback segment t_4_57 tablespace roll_4;
create rollback segment t_4_58 tablespace roll_4;
create rollback segment t_4_59 tablespace roll_4;
create rollback segment t_4_60 tablespace roll_4;
create rollback segment t_4_61 tablespace roll_4;
create rollback segment t_4_62 tablespace roll_4;
create rollback segment t_4_63 tablespace roll_4;
create rollback segment t_4_64 tablespace roll_4;
create rollback segment t_4_65 tablespace roll_4;
create rollback segment t_4_66 tablespace roll_4;
create rollback segment t_4_67 tablespace roll_4;
create rollback segment t_4_68 tablespace roll_4;
create rollback segment t_4_69 tablespace roll_4;
create rollback segment t_4_70 tablespace roll_4;
create rollback segment t_4_71 tablespace roll_4;
create rollback segment t_4_72 tablespace roll_4;
create rollback segment t_4_73 tablespace roll_4;
create rollback segment t_4_74 tablespace roll_4;
create rollback segment t_4_75 tablespace roll_4;
create rollback segment t_4_76 tablespace roll_4;
create rollback segment t_4_77 tablespace roll_4;
create rollback segment t_4_78 tablespace roll_4;
create rollback segment t_4_79 tablespace roll_4;
create rollback segment t_4_80 tablespace roll_4;
create rollback segment t_4_81 tablespace roll_4;
create rollback segment t_4_82 tablespace roll_4;
create rollback segment t_4_83 tablespace roll_4;
create rollback segment t_4_84 tablespace roll_4;
create rollback segment t_4_85 tablespace roll_4;
create rollback segment t_4_86 tablespace roll_4;
create rollback segment t_4_87 tablespace roll_4;
create rollback segment t_4_88 tablespace roll_4;
create rollback segment t_4_89 tablespace roll_4;
create rollback segment t_4_90 tablespace roll_4;
create rollback segment t_4_91 tablespace roll_4;
create rollback segment t_4_92 tablespace roll_4;
create rollback segment t_4_93 tablespace roll_4;
create rollback segment t_4_94 tablespace roll_4;
create rollback segment t_4_95 tablespace roll_4;
create rollback segment t_4_96 tablespace roll_4;
create rollback segment t_4_97 tablespace roll_4;
create rollback segment t_4_98 tablespace roll_4;
create rollback segment t_4_99 tablespace roll_4;
create rollback segment t_4_100 tablespace roll_4;
exit;
```

tpcc_roll_4_3.sql

```
create rollback segment t_4_101 tablespace roll_4;
create rollback segment t_4_102 tablespace roll_4;
create rollback segment t_4_103 tablespace roll_4;
create rollback segment t_4_104 tablespace roll_4;
create rollback segment t_4_105 tablespace roll_4;
create rollback segment t_4_106 tablespace roll_4;
create rollback segment t_4_107 tablespace roll_4;
create rollback segment t_4_108 tablespace roll_4;
create rollback segment t_4_109 tablespace roll_4;
create rollback segment t_4_110 tablespace roll_4;
create rollback segment t_4_111 tablespace roll_4;
create rollback segment t_4_112 tablespace roll_4;
create rollback segment t_4_113 tablespace roll_4;
create rollback segment t_4_114 tablespace roll_4;
create rollback segment t_4_115 tablespace roll_4;
create rollback segment t_4_116 tablespace roll_4;
create rollback segment t_4_117 tablespace roll_4;
create rollback segment t_4_118 tablespace roll_4;
create rollback segment t_4_119 tablespace roll_4;
create rollback segment t_4_120 tablespace roll_4;
create rollback segment t_4_121 tablespace roll_4;
create rollback segment t_4_122 tablespace roll_4;
create rollback segment t_4_123 tablespace roll_4;
create rollback segment t_4_124 tablespace roll_4;
create rollback segment t_4_125 tablespace roll_4;
create rollback segment t_4_126 tablespace roll_4;
create rollback segment t_4_127 tablespace roll_4;
create rollback segment t_4_128 tablespace roll_4;
create rollback segment t_4_129 tablespace roll_4;
create rollback segment t_4_130 tablespace roll_4;
create rollback segment t_4_131 tablespace roll_4;
create rollback segment t_4_132 tablespace roll_4;
create rollback segment t_4_133 tablespace roll_4;
create rollback segment t_4_134 tablespace roll_4;
create rollback segment t_4_135 tablespace roll_4;
create rollback segment t_4_136 tablespace roll_4;
create rollback segment t_4_137 tablespace roll_4;
create rollback segment t_4_138 tablespace roll_4;
create rollback segment t_4_139 tablespace roll_4;
create rollback segment t_4_140 tablespace roll_4;
```

```
create rollback segment t_4_141 tablespace roll_4;
create rollback segment t_4_142 tablespace roll_4;
create rollback segment t_4_143 tablespace roll_4;
create rollback segment t_4_144 tablespace roll_4;
create rollback segment t_4_145 tablespace roll_4;
create rollback segment t_4_146 tablespace roll_4;
create rollback segment t_4_147 tablespace roll_4;
create rollback segment t_4_148 tablespace roll_4;
create rollback segment t_4_149 tablespace roll_4;
create rollback segment t_4_150 tablespace roll_4;
exit;
```

tpcc_roll_4_4.sql

```
create rollback segment t_4_151 tablespace roll_4;
create rollback segment t_4_152 tablespace roll_4;
create rollback segment t_4_153 tablespace roll_4;
create rollback segment t_4_154 tablespace roll_4;
create rollback segment t_4_155 tablespace roll_4;
create rollback segment t_4_156 tablespace roll_4;
create rollback segment t_4_157 tablespace roll_4;
create rollback segment t_4_158 tablespace roll_4;
create rollback segment t_4_159 tablespace roll_4;
create rollback segment t_4_160 tablespace roll_4;
create rollback segment t_4_161 tablespace roll_4;
create rollback segment t_4_162 tablespace roll_4;
create rollback segment t_4_163 tablespace roll_4;
create rollback segment t_4_164 tablespace roll_4;
create rollback segment t_4_165 tablespace roll_4;
create rollback segment t_4_166 tablespace roll_4;
create rollback segment t_4_167 tablespace roll_4;
create rollback segment t_4_168 tablespace roll_4;
create rollback segment t_4_169 tablespace roll_4;
create rollback segment t_4_170 tablespace roll_4;
create rollback segment t_4_171 tablespace roll_4;
create rollback segment t_4_172 tablespace roll_4;
create rollback segment t_4_173 tablespace roll_4;
create rollback segment t_4_174 tablespace roll_4;
create rollback segment t_4_175 tablespace roll_4;
create rollback segment t_4_176 tablespace roll_4;
create rollback segment t_4_177 tablespace roll_4;
create rollback segment t_4_178 tablespace roll_4;
create rollback segment t_4_179 tablespace roll_4;
create rollback segment t_4_180 tablespace roll_4;
create rollback segment t_4_181 tablespace roll_4;
create rollback segment t_4_182 tablespace roll_4;
create rollback segment t_4_183 tablespace roll_4;
create rollback segment t_4_184 tablespace roll_4;
create rollback segment t_4_185 tablespace roll_4;
create rollback segment t_4_186 tablespace roll_4;
create rollback segment t_4_187 tablespace roll_4;
create rollback segment t_4_188 tablespace roll_4;
create rollback segment t_4_189 tablespace roll_4;
create rollback segment t_4_190 tablespace roll_4;
create rollback segment t_4_191 tablespace roll_4;
create rollback segment t_4_192 tablespace roll_4;
create rollback segment t_4_193 tablespace roll_4;
create rollback segment t_4_194 tablespace roll_4;
create rollback segment t_4_195 tablespace roll_4;
create rollback segment t_4_196 tablespace roll_4;
create rollback segment t_4_197 tablespace roll_4;
create rollback segment t_4_198 tablespace roll_4;
create rollback segment t_4_199 tablespace roll_4;
create rollback segment t_4_200 tablespace roll_4;
exit;
```

tpcc_roll_5_1.sql

```
create rollback segment t_5_1 tablespace roll_5;
create rollback segment t_5_2 tablespace roll_5;
create rollback segment t_5_3 tablespace roll_5;
create rollback segment t_5_4 tablespace roll_5;
create rollback segment t_5_5 tablespace roll_5;
create rollback segment t_5_6 tablespace roll_5;
create rollback segment t_5_7 tablespace roll_5;
create rollback segment t_5_8 tablespace roll_5;
create rollback segment t_5_9 tablespace roll_5;
create rollback segment t_5_10 tablespace roll_5;
create rollback segment t_5_11 tablespace roll_5;
create rollback segment t_5_12 tablespace roll_5;
create rollback segment t_5_13 tablespace roll_5;
create rollback segment t_5_14 tablespace roll_5;
create rollback segment t_5_15 tablespace roll_5;
create rollback segment t_5_16 tablespace roll_5;
create rollback segment t_5_17 tablespace roll_5;
create rollback segment t_5_18 tablespace roll_5;
create rollback segment t_5_19 tablespace roll_5;
create rollback segment t_5_20 tablespace roll_5;
create rollback segment t_5_21 tablespace roll_5;
create rollback segment t_5_22 tablespace roll_5;
create rollback segment t_5_23 tablespace roll_5;
create rollback segment t_5_24 tablespace roll_5;
create rollback segment t_5_25 tablespace roll_5;
create rollback segment t_5_26 tablespace roll_5;
create rollback segment t_5_27 tablespace roll_5;
create rollback segment t_5_28 tablespace roll_5;
create rollback segment t_5_29 tablespace roll_5;
create rollback segment t_5_30 tablespace roll_5;
create rollback segment t_5_31 tablespace roll_5;
create rollback segment t_5_32 tablespace roll_5;
create rollback segment t_5_33 tablespace roll_5;
create rollback segment t_5_34 tablespace roll_5;
create rollback segment t_5_35 tablespace roll_5;
create rollback segment t_5_36 tablespace roll_5;
create rollback segment t_5_37 tablespace roll_5;
create rollback segment t_5_38 tablespace roll_5;
create rollback segment t_5_39 tablespace roll_5;
create rollback segment t_5_40 tablespace roll_5;
create rollback segment t_5_41 tablespace roll_5;
create rollback segment t_5_42 tablespace roll_5;
create rollback segment t_5_43 tablespace roll_5;
create rollback segment t_5_44 tablespace roll_5;
create rollback segment t_5_45 tablespace roll_5;
create rollback segment t_5_46 tablespace roll_5;
create rollback segment t_5_47 tablespace roll_5;
create rollback segment t_5_48 tablespace roll_5;
create rollback segment t_5_49 tablespace roll_5;
create rollback segment t_5_50 tablespace roll_5;
exit;
```

tpcc_roll_5_2.sql

```
create rollback segment t_5_51 tablespace roll_5;
create rollback segment t_5_52 tablespace roll_5;
create rollback segment t_5_53 tablespace roll_5;
create rollback segment t_5_54 tablespace roll_5;
create rollback segment t_5_55 tablespace roll_5;
create rollback segment t_5_56 tablespace roll_5;
create rollback segment t_5_57 tablespace roll_5;
create rollback segment t_5_58 tablespace roll_5;
create rollback segment t_5_59 tablespace roll_5;
create rollback segment t_5_60 tablespace roll_5;
create rollback segment t_5_61 tablespace roll_5;
create rollback segment t_5_62 tablespace roll_5;
create rollback segment t_5_63 tablespace roll_5;
create rollback segment t_5_64 tablespace roll_5;
create rollback segment t_5_65 tablespace roll_5;
create rollback segment t_5_66 tablespace roll_5;
create rollback segment t_5_67 tablespace roll_5;
create rollback segment t_5_68 tablespace roll_5;
create rollback segment t_5_69 tablespace roll_5;
create rollback segment t_5_70 tablespace roll_5;
create rollback segment t_5_71 tablespace roll_5;
create rollback segment t_5_72 tablespace roll_5;
create rollback segment t_5_73 tablespace roll_5;
create rollback segment t_5_74 tablespace roll_5;
create rollback segment t_5_75 tablespace roll_5;
create rollback segment t_5_76 tablespace roll_5;
create rollback segment t_5_77 tablespace roll_5;
create rollback segment t_5_78 tablespace roll_5;
```

```
create rollback segment t_5_79 tablespace roll_5;
create rollback segment t_5_80 tablespace roll_5;
create rollback segment t_5_81 tablespace roll_5;
create rollback segment t_5_82 tablespace roll_5;
create rollback segment t_5_83 tablespace roll_5;
create rollback segment t_5_84 tablespace roll_5;
create rollback segment t_5_85 tablespace roll_5;
create rollback segment t_5_86 tablespace roll_5;
create rollback segment t_5_87 tablespace roll_5;
create rollback segment t_5_88 tablespace roll_5;
create rollback segment t_5_89 tablespace roll_5;
create rollback segment t_5_90 tablespace roll_5;
create rollback segment t_5_91 tablespace roll_5;
create rollback segment t_5_92 tablespace roll_5;
create rollback segment t_5_93 tablespace roll_5;
create rollback segment t_5_94 tablespace roll_5;
create rollback segment t_5_95 tablespace roll_5;
create rollback segment t_5_96 tablespace roll_5;
create rollback segment t_5_97 tablespace roll_5;
create rollback segment t_5_98 tablespace roll_5;
create rollback segment t_5_99 tablespace roll_5;
create rollback segment t_5_100 tablespace roll_5;
exit;
```

tpcc_roll_5_3.sql

```
create rollback segment t_5_101 tablespace roll_5;
create rollback segment t_5_102 tablespace roll_5;
create rollback segment t_5_103 tablespace roll_5;
create rollback segment t_5_104 tablespace roll_5;
create rollback segment t_5_105 tablespace roll_5;
create rollback segment t_5_106 tablespace roll_5;
create rollback segment t_5_107 tablespace roll_5;
create rollback segment t_5_108 tablespace roll_5;
create rollback segment t_5_109 tablespace roll_5;
create rollback segment t_5_110 tablespace roll_5;
create rollback segment t_5_111 tablespace roll_5;
create rollback segment t_5_112 tablespace roll_5;
create rollback segment t_5_113 tablespace roll_5;
create rollback segment t_5_114 tablespace roll_5;
create rollback segment t_5_115 tablespace roll_5;
create rollback segment t_5_116 tablespace roll_5;
create rollback segment t_5_117 tablespace roll_5;
create rollback segment t_5_118 tablespace roll_5;
create rollback segment t_5_119 tablespace roll_5;
create rollback segment t_5_120 tablespace roll_5;
create rollback segment t_5_121 tablespace roll_5;
create rollback segment t_5_122 tablespace roll_5;
create rollback segment t_5_123 tablespace roll_5;
create rollback segment t_5_124 tablespace roll_5;
create rollback segment t_5_125 tablespace roll_5;
create rollback segment t_5_126 tablespace roll_5;
create rollback segment t_5_127 tablespace roll_5;
create rollback segment t_5_128 tablespace roll_5;
create rollback segment t_5_129 tablespace roll_5;
create rollback segment t_5_130 tablespace roll_5;
create rollback segment t_5_131 tablespace roll_5;
create rollback segment t_5_132 tablespace roll_5;
create rollback segment t_5_133 tablespace roll_5;
create rollback segment t_5_134 tablespace roll_5;
create rollback segment t_5_135 tablespace roll_5;
create rollback segment t_5_136 tablespace roll_5;
create rollback segment t_5_137 tablespace roll_5;
create rollback segment t_5_138 tablespace roll_5;
create rollback segment t_5_139 tablespace roll_5;
create rollback segment t_5_140 tablespace roll_5;
create rollback segment t_5_141 tablespace roll_5;
create rollback segment t_5_142 tablespace roll_5;
create rollback segment t_5_143 tablespace roll_5;
create rollback segment t_5_144 tablespace roll_5;
create rollback segment t_5_145 tablespace roll_5;
create rollback segment t_5_146 tablespace roll_5;
create rollback segment t_5_147 tablespace roll_5;
create rollback segment t_5_148 tablespace roll_5;
create rollback segment t_5_149 tablespace roll_5;
create rollback segment t_5_150 tablespace roll_5;
exit;
```

tpcc_roll_5_4.sql

```
create rollback segment t_5_151 tablespace roll_5;
create rollback segment t_5_152 tablespace roll_5;
create rollback segment t_5_153 tablespace roll_5;
create rollback segment t_5_154 tablespace roll_5;
create rollback segment t_5_155 tablespace roll_5;
create rollback segment t_5_156 tablespace roll_5;
create rollback segment t_5_157 tablespace roll_5;
create rollback segment t_5_158 tablespace roll_5;
create rollback segment t_5_159 tablespace roll_5;
create rollback segment t_5_160 tablespace roll_5;
create rollback segment t_5_161 tablespace roll_5;
create rollback segment t_5_162 tablespace roll_5;
create rollback segment t_5_163 tablespace roll_5;
create rollback segment t_5_164 tablespace roll_5;
create rollback segment t_5_165 tablespace roll_5;
create rollback segment t_5_166 tablespace roll_5;
create rollback segment t_5_167 tablespace roll_5;
create rollback segment t_5_168 tablespace roll_5;
create rollback segment t_5_169 tablespace roll_5;
create rollback segment t_5_170 tablespace roll_5;
create rollback segment t_5_171 tablespace roll_5;
create rollback segment t_5_172 tablespace roll_5;
create rollback segment t_5_173 tablespace roll_5;
create rollback segment t_5_174 tablespace roll_5;
create rollback segment t_5_175 tablespace roll_5;
create rollback segment t_5_176 tablespace roll_5;
create rollback segment t_5_177 tablespace roll_5;
create rollback segment t_5_178 tablespace roll_5;
create rollback segment t_5_179 tablespace roll_5;
create rollback segment t_5_180 tablespace roll_5;
create rollback segment t_5_181 tablespace roll_5;
create rollback segment t_5_182 tablespace roll_5;
create rollback segment t_5_183 tablespace roll_5;
create rollback segment t_5_184 tablespace roll_5;
create rollback segment t_5_185 tablespace roll_5;
create rollback segment t_5_186 tablespace roll_5;
create rollback segment t_5_187 tablespace roll_5;
create rollback segment t_5_188 tablespace roll_5;
create rollback segment t_5_189 tablespace roll_5;
create rollback segment t_5_190 tablespace roll_5;
create rollback segment t_5_191 tablespace roll_5;
create rollback segment t_5_192 tablespace roll_5;
create rollback segment t_5_193 tablespace roll_5;
create rollback segment t_5_194 tablespace roll_5;
create rollback segment t_5_195 tablespace roll_5;
create rollback segment t_5_196 tablespace roll_5;
create rollback segment t_5_197 tablespace roll_5;
create rollback segment t_5_198 tablespace roll_5;
create rollback segment t_5_199 tablespace roll_5;
create rollback segment t_5_200 tablespace roll_5;
exit;
```

tpcc_roll_6_1.sql

```
create rollback segment t_6_1 tablespace roll_6;
create rollback segment t_6_2 tablespace roll_6;
create rollback segment t_6_3 tablespace roll_6;
create rollback segment t_6_4 tablespace roll_6;
create rollback segment t_6_5 tablespace roll_6;
create rollback segment t_6_6 tablespace roll_6;
create rollback segment t_6_7 tablespace roll_6;
create rollback segment t_6_8 tablespace roll_6;
create rollback segment t_6_9 tablespace roll_6;
create rollback segment t_6_10 tablespace roll_6;
create rollback segment t_6_11 tablespace roll_6;
create rollback segment t_6_12 tablespace roll_6;
create rollback segment t_6_13 tablespace roll_6;
create rollback segment t_6_14 tablespace roll_6;
create rollback segment t_6_15 tablespace roll_6;
create rollback segment t_6_16 tablespace roll_6;
create rollback segment t_6_17 tablespace roll_6;
```

```
create rollback segment t_6_18 tablespace roll_6;
create rollback segment t_6_19 tablespace roll_6;
create rollback segment t_6_20 tablespace roll_6;
create rollback segment t_6_21 tablespace roll_6;
create rollback segment t_6_22 tablespace roll_6;
create rollback segment t_6_23 tablespace roll_6;
create rollback segment t_6_24 tablespace roll_6;
create rollback segment t_6_25 tablespace roll_6;
create rollback segment t_6_26 tablespace roll_6;
create rollback segment t_6_27 tablespace roll_6;
create rollback segment t_6_28 tablespace roll_6;
create rollback segment t_6_29 tablespace roll_6;
create rollback segment t_6_30 tablespace roll_6;
create rollback segment t_6_31 tablespace roll_6;
create rollback segment t_6_32 tablespace roll_6;
create rollback segment t_6_33 tablespace roll_6;
create rollback segment t_6_34 tablespace roll_6;
create rollback segment t_6_35 tablespace roll_6;
create rollback segment t_6_36 tablespace roll_6;
create rollback segment t_6_37 tablespace roll_6;
create rollback segment t_6_38 tablespace roll_6;
create rollback segment t_6_39 tablespace roll_6;
create rollback segment t_6_40 tablespace roll_6;
create rollback segment t_6_41 tablespace roll_6;
create rollback segment t_6_42 tablespace roll_6;
create rollback segment t_6_43 tablespace roll_6;
create rollback segment t_6_44 tablespace roll_6;
create rollback segment t_6_45 tablespace roll_6;
create rollback segment t_6_46 tablespace roll_6;
create rollback segment t_6_47 tablespace roll_6;
create rollback segment t_6_48 tablespace roll_6;
create rollback segment t_6_49 tablespace roll_6;
create rollback segment t_6_50 tablespace roll_6;
exit;
```

tpcc_roll_6_2.sql

```
create rollback segment t_6_51 tablespace roll_6;
create rollback segment t_6_52 tablespace roll_6;
create rollback segment t_6_53 tablespace roll_6;
create rollback segment t_6_54 tablespace roll_6;
create rollback segment t_6_55 tablespace roll_6;
create rollback segment t_6_56 tablespace roll_6;
create rollback segment t_6_57 tablespace roll_6;
create rollback segment t_6_58 tablespace roll_6;
create rollback segment t_6_59 tablespace roll_6;
create rollback segment t_6_60 tablespace roll_6;
create rollback segment t_6_61 tablespace roll_6;
create rollback segment t_6_62 tablespace roll_6;
create rollback segment t_6_63 tablespace roll_6;
create rollback segment t_6_64 tablespace roll_6;
create rollback segment t_6_65 tablespace roll_6;
create rollback segment t_6_66 tablespace roll_6;
create rollback segment t_6_67 tablespace roll_6;
create rollback segment t_6_68 tablespace roll_6;
create rollback segment t_6_69 tablespace roll_6;
create rollback segment t_6_70 tablespace roll_6;
create rollback segment t_6_71 tablespace roll_6;
create rollback segment t_6_72 tablespace roll_6;
create rollback segment t_6_73 tablespace roll_6;
create rollback segment t_6_74 tablespace roll_6;
create rollback segment t_6_75 tablespace roll_6;
create rollback segment t_6_76 tablespace roll_6;
create rollback segment t_6_77 tablespace roll_6;
create rollback segment t_6_78 tablespace roll_6;
create rollback segment t_6_79 tablespace roll_6;
create rollback segment t_6_80 tablespace roll_6;
create rollback segment t_6_81 tablespace roll_6;
create rollback segment t_6_82 tablespace roll_6;
create rollback segment t_6_83 tablespace roll_6;
create rollback segment t_6_84 tablespace roll_6;
create rollback segment t_6_85 tablespace roll_6;
create rollback segment t_6_86 tablespace roll_6;
create rollback segment t_6_87 tablespace roll_6;
create rollback segment t_6_88 tablespace roll_6;
create rollback segment t_6_89 tablespace roll_6;
create rollback segment t_6_90 tablespace roll_6;
create rollback segment t_6_91 tablespace roll_6;
create rollback segment t_6_92 tablespace roll_6;
create rollback segment t_6_93 tablespace roll_6;
create rollback segment t_6_94 tablespace roll_6;
create rollback segment t_6_95 tablespace roll_6;
create rollback segment t_6_96 tablespace roll_6;
create rollback segment t_6_97 tablespace roll_6;
create rollback segment t_6_98 tablespace roll_6;
create rollback segment t_6_99 tablespace roll_6;
create rollback segment t_6_100 tablespace roll_6;
exit;
```

tpcc_roll_6_3.sql

```
create rollback segment t_6_101 tablespace roll_6;
create rollback segment t_6_102 tablespace roll_6;
create rollback segment t_6_103 tablespace roll_6;
create rollback segment t_6_104 tablespace roll_6;
create rollback segment t_6_105 tablespace roll_6;
create rollback segment t_6_106 tablespace roll_6;
create rollback segment t_6_107 tablespace roll_6;
create rollback segment t_6_108 tablespace roll_6;
create rollback segment t_6_109 tablespace roll_6;
create rollback segment t_6_110 tablespace roll_6;
create rollback segment t_6_111 tablespace roll_6;
create rollback segment t_6_112 tablespace roll_6;
create rollback segment t_6_113 tablespace roll_6;
create rollback segment t_6_114 tablespace roll_6;
create rollback segment t_6_115 tablespace roll_6;
create rollback segment t_6_116 tablespace roll_6;
create rollback segment t_6_117 tablespace roll_6;
create rollback segment t_6_118 tablespace roll_6;
create rollback segment t_6_119 tablespace roll_6;
create rollback segment t_6_120 tablespace roll_6;
create rollback segment t_6_121 tablespace roll_6;
create rollback segment t_6_122 tablespace roll_6;
create rollback segment t_6_123 tablespace roll_6;
create rollback segment t_6_124 tablespace roll_6;
create rollback segment t_6_125 tablespace roll_6;
create rollback segment t_6_126 tablespace roll_6;
create rollback segment t_6_127 tablespace roll_6;
create rollback segment t_6_128 tablespace roll_6;
create rollback segment t_6_129 tablespace roll_6;
create rollback segment t_6_130 tablespace roll_6;
create rollback segment t_6_131 tablespace roll_6;
create rollback segment t_6_132 tablespace roll_6;
create rollback segment t_6_133 tablespace roll_6;
create rollback segment t_6_134 tablespace roll_6;
create rollback segment t_6_135 tablespace roll_6;
create rollback segment t_6_136 tablespace roll_6;
create rollback segment t_6_137 tablespace roll_6;
create rollback segment t_6_138 tablespace roll_6;
create rollback segment t_6_139 tablespace roll_6;
create rollback segment t_6_140 tablespace roll_6;
create rollback segment t_6_141 tablespace roll_6;
create rollback segment t_6_142 tablespace roll_6;
create rollback segment t_6_143 tablespace roll_6;
create rollback segment t_6_144 tablespace roll_6;
create rollback segment t_6_145 tablespace roll_6;
create rollback segment t_6_146 tablespace roll_6;
create rollback segment t_6_147 tablespace roll_6;
create rollback segment t_6_148 tablespace roll_6;
create rollback segment t_6_149 tablespace roll_6;
create rollback segment t_6_150 tablespace roll_6;
exit;
```

tpcc_roll_6_4.sql

```
create rollback segment t_6_151 tablespace roll_6;
create rollback segment t_6_152 tablespace roll_6;
create rollback segment t_6_153 tablespace roll_6;
create rollback segment t_6_154 tablespace roll_6;
create rollback segment t_6_155 tablespace roll_6;
```

```
create rollback segment t_6_156 tablespace roll_6;
create rollback segment t_6_157 tablespace roll_6;
create rollback segment t_6_158 tablespace roll_6;
create rollback segment t_6_159 tablespace roll_6;
create rollback segment t_6_160 tablespace roll_6;
create rollback segment t_6_161 tablespace roll_6;
create rollback segment t_6_162 tablespace roll_6;
create rollback segment t_6_163 tablespace roll_6;
create rollback segment t_6_164 tablespace roll_6;
create rollback segment t_6_165 tablespace roll_6;
create rollback segment t_6_166 tablespace roll_6;
create rollback segment t_6_167 tablespace roll_6;
create rollback segment t_6_168 tablespace roll_6;
create rollback segment t_6_169 tablespace roll_6;
create rollback segment t_6_170 tablespace roll_6;
create rollback segment t_6_171 tablespace roll_6;
create rollback segment t_6_172 tablespace roll_6;
create rollback segment t_6_173 tablespace roll_6;
create rollback segment t_6_174 tablespace roll_6;
create rollback segment t_6_175 tablespace roll_6;
create rollback segment t_6_176 tablespace roll_6;
create rollback segment t_6_177 tablespace roll_6;
create rollback segment t_6_178 tablespace roll_6;
create rollback segment t_6_179 tablespace roll_6;
create rollback segment t_6_180 tablespace roll_6;
create rollback segment t_6_181 tablespace roll_6;
create rollback segment t_6_182 tablespace roll_6;
create rollback segment t_6_183 tablespace roll_6;
create rollback segment t_6_184 tablespace roll_6;
create rollback segment t_6_185 tablespace roll_6;
create rollback segment t_6_186 tablespace roll_6;
create rollback segment t_6_187 tablespace roll_6;
create rollback segment t_6_188 tablespace roll_6;
create rollback segment t_6_189 tablespace roll_6;
create rollback segment t_6_190 tablespace roll_6;
create rollback segment t_6_191 tablespace roll_6;
create rollback segment t_6_192 tablespace roll_6;
create rollback segment t_6_193 tablespace roll_6;
create rollback segment t_6_194 tablespace roll_6;
create rollback segment t_6_195 tablespace roll_6;
create rollback segment t_6_196 tablespace roll_6;
create rollback segment t_6_197 tablespace roll_6;
create rollback segment t_6_198 tablespace roll_6;
create rollback segment t_6_199 tablespace roll_6;
create rollback segment t_6_200 tablespace roll_6;
exit;
```

tpcc_roll_7_1.sql

```
create rollback segment t_7_1 tablespace roll_7;
create rollback segment t_7_2 tablespace roll_7;
create rollback segment t_7_3 tablespace roll_7;
create rollback segment t_7_4 tablespace roll_7;
create rollback segment t_7_5 tablespace roll_7;
create rollback segment t_7_6 tablespace roll_7;
create rollback segment t_7_7 tablespace roll_7;
create rollback segment t_7_8 tablespace roll_7;
create rollback segment t_7_9 tablespace roll_7;
create rollback segment t_7_10 tablespace roll_7;
create rollback segment t_7_11 tablespace roll_7;
create rollback segment t_7_12 tablespace roll_7;
create rollback segment t_7_13 tablespace roll_7;
create rollback segment t_7_14 tablespace roll_7;
create rollback segment t_7_15 tablespace roll_7;
create rollback segment t_7_16 tablespace roll_7;
create rollback segment t_7_17 tablespace roll_7;
create rollback segment t_7_18 tablespace roll_7;
create rollback segment t_7_19 tablespace roll_7;
create rollback segment t_7_20 tablespace roll_7;
create rollback segment t_7_21 tablespace roll_7;
create rollback segment t_7_22 tablespace roll_7;
create rollback segment t_7_23 tablespace roll_7;
create rollback segment t_7_24 tablespace roll_7;
create rollback segment t_7_25 tablespace roll_7;
create rollback segment t_7_26 tablespace roll_7;
create rollback segment t_7_27 tablespace roll_7;
create rollback segment t_7_28 tablespace roll_7;
create rollback segment t_7_29 tablespace roll_7;
create rollback segment t_7_30 tablespace roll_7;
create rollback segment t_7_31 tablespace roll_7;
create rollback segment t_7_32 tablespace roll_7;
create rollback segment t_7_33 tablespace roll_7;
create rollback segment t_7_34 tablespace roll_7;
create rollback segment t_7_35 tablespace roll_7;
create rollback segment t_7_36 tablespace roll_7;
create rollback segment t_7_37 tablespace roll_7;
create rollback segment t_7_38 tablespace roll_7;
create rollback segment t_7_39 tablespace roll_7;
create rollback segment t_7_40 tablespace roll_7;
create rollback segment t_7_41 tablespace roll_7;
create rollback segment t_7_42 tablespace roll_7;
create rollback segment t_7_43 tablespace roll_7;
create rollback segment t_7_44 tablespace roll_7;
create rollback segment t_7_45 tablespace roll_7;
create rollback segment t_7_46 tablespace roll_7;
create rollback segment t_7_47 tablespace roll_7;
create rollback segment t_7_48 tablespace roll_7;
create rollback segment t_7_49 tablespace roll_7;
create rollback segment t_7_50 tablespace roll_7;
exit;
```

tpcc_roll_7_2.sql

```
create rollback segment t_7_51 tablespace roll_7;
create rollback segment t_7_52 tablespace roll_7;
create rollback segment t_7_53 tablespace roll_7;
create rollback segment t_7_54 tablespace roll_7;
create rollback segment t_7_55 tablespace roll_7;
create rollback segment t_7_56 tablespace roll_7;
create rollback segment t_7_57 tablespace roll_7;
create rollback segment t_7_58 tablespace roll_7;
create rollback segment t_7_59 tablespace roll_7;
create rollback segment t_7_60 tablespace roll_7;
create rollback segment t_7_61 tablespace roll_7;
create rollback segment t_7_62 tablespace roll_7;
create rollback segment t_7_63 tablespace roll_7;
create rollback segment t_7_64 tablespace roll_7;
create rollback segment t_7_65 tablespace roll_7;
create rollback segment t_7_66 tablespace roll_7;
create rollback segment t_7_67 tablespace roll_7;
create rollback segment t_7_68 tablespace roll_7;
create rollback segment t_7_69 tablespace roll_7;
create rollback segment t_7_70 tablespace roll_7;
create rollback segment t_7_71 tablespace roll_7;
create rollback segment t_7_72 tablespace roll_7;
create rollback segment t_7_73 tablespace roll_7;
create rollback segment t_7_74 tablespace roll_7;
create rollback segment t_7_75 tablespace roll_7;
create rollback segment t_7_76 tablespace roll_7;
create rollback segment t_7_77 tablespace roll_7;
create rollback segment t_7_78 tablespace roll_7;
create rollback segment t_7_79 tablespace roll_7;
create rollback segment t_7_80 tablespace roll_7;
create rollback segment t_7_81 tablespace roll_7;
create rollback segment t_7_82 tablespace roll_7;
create rollback segment t_7_83 tablespace roll_7;
create rollback segment t_7_84 tablespace roll_7;
create rollback segment t_7_85 tablespace roll_7;
create rollback segment t_7_86 tablespace roll_7;
create rollback segment t_7_87 tablespace roll_7;
create rollback segment t_7_88 tablespace roll_7;
create rollback segment t_7_89 tablespace roll_7;
create rollback segment t_7_90 tablespace roll_7;
create rollback segment t_7_91 tablespace roll_7;
create rollback segment t_7_92 tablespace roll_7;
create rollback segment t_7_93 tablespace roll_7;
create rollback segment t_7_94 tablespace roll_7;
create rollback segment t_7_95 tablespace roll_7;
create rollback segment t_7_96 tablespace roll_7;
create rollback segment t_7_97 tablespace roll_7;
create rollback segment t_7_98 tablespace roll_7;
create rollback segment t_7_99 tablespace roll_7;
create rollback segment t_7_100 tablespace roll_7;
exit;
```

tpcc_roll_7_3.sql

```
create rollback segment t_7_101 tablespace roll_7;
create rollback segment t_7_102 tablespace roll_7;
create rollback segment t_7_103 tablespace roll_7;
create rollback segment t_7_104 tablespace roll_7;
create rollback segment t_7_105 tablespace roll_7;
create rollback segment t_7_106 tablespace roll_7;
create rollback segment t_7_107 tablespace roll_7;
create rollback segment t_7_108 tablespace roll_7;
create rollback segment t_7_109 tablespace roll_7;
create rollback segment t_7_110 tablespace roll_7;
create rollback segment t_7_111 tablespace roll_7;
create rollback segment t_7_112 tablespace roll_7;
create rollback segment t_7_113 tablespace roll_7;
create rollback segment t_7_114 tablespace roll_7;
create rollback segment t_7_115 tablespace roll_7;
create rollback segment t_7_116 tablespace roll_7;
create rollback segment t_7_117 tablespace roll_7;
create rollback segment t_7_118 tablespace roll_7;
create rollback segment t_7_119 tablespace roll_7;
create rollback segment t_7_120 tablespace roll_7;
create rollback segment t_7_121 tablespace roll_7;
create rollback segment t_7_122 tablespace roll_7;
create rollback segment t_7_123 tablespace roll_7;
create rollback segment t_7_124 tablespace roll_7;
create rollback segment t_7_125 tablespace roll_7;
create rollback segment t_7_126 tablespace roll_7;
create rollback segment t_7_127 tablespace roll_7;
create rollback segment t_7_128 tablespace roll_7;
create rollback segment t_7_129 tablespace roll_7;
create rollback segment t_7_130 tablespace roll_7;
create rollback segment t_7_131 tablespace roll_7;
create rollback segment t_7_132 tablespace roll_7;
create rollback segment t_7_133 tablespace roll_7;
create rollback segment t_7_134 tablespace roll_7;
create rollback segment t_7_135 tablespace roll_7;
create rollback segment t_7_136 tablespace roll_7;
create rollback segment t_7_137 tablespace roll_7;
create rollback segment t_7_138 tablespace roll_7;
create rollback segment t_7_139 tablespace roll_7;
create rollback segment t_7_140 tablespace roll_7;
create rollback segment t_7_141 tablespace roll_7;
create rollback segment t_7_142 tablespace roll_7;
create rollback segment t_7_143 tablespace roll_7;
create rollback segment t_7_144 tablespace roll_7;
create rollback segment t_7_145 tablespace roll_7;
create rollback segment t_7_146 tablespace roll_7;
create rollback segment t_7_147 tablespace roll_7;
create rollback segment t_7_148 tablespace roll_7;
create rollback segment t_7_149 tablespace roll_7;
create rollback segment t_7_150 tablespace roll_7;
exit;
```

tpcc_roll_7_4.sql

```
create rollback segment t_7_151 tablespace roll_7;
create rollback segment t_7_152 tablespace roll_7;
create rollback segment t_7_153 tablespace roll_7;
create rollback segment t_7_154 tablespace roll_7;
create rollback segment t_7_155 tablespace roll_7;
create rollback segment t_7_156 tablespace roll_7;
create rollback segment t_7_157 tablespace roll_7;
create rollback segment t_7_158 tablespace roll_7;
create rollback segment t_7_159 tablespace roll_7;
create rollback segment t_7_160 tablespace roll_7;
create rollback segment t_7_161 tablespace roll_7;
create rollback segment t_7_162 tablespace roll_7;
create rollback segment t_7_163 tablespace roll_7;
create rollback segment t_7_164 tablespace roll_7;
create rollback segment t_7_165 tablespace roll_7;
create rollback segment t_7_166 tablespace roll_7;
create rollback segment t_7_167 tablespace roll_7;
create rollback segment t_7_168 tablespace roll_7;
create rollback segment t_7_169 tablespace roll_7;
create rollback segment t_7_170 tablespace roll_7;
create rollback segment t_7_171 tablespace roll_7;
create rollback segment t_7_172 tablespace roll_7;
create rollback segment t_7_173 tablespace roll_7;
create rollback segment t_7_174 tablespace roll_7;
create rollback segment t_7_175 tablespace roll_7;
create rollback segment t_7_176 tablespace roll_7;
create rollback segment t_7_177 tablespace roll_7;
create rollback segment t_7_178 tablespace roll_7;
create rollback segment t_7_179 tablespace roll_7;
create rollback segment t_7_180 tablespace roll_7;
create rollback segment t_7_181 tablespace roll_7;
create rollback segment t_7_182 tablespace roll_7;
create rollback segment t_7_183 tablespace roll_7;
create rollback segment t_7_184 tablespace roll_7;
create rollback segment t_7_185 tablespace roll_7;
create rollback segment t_7_186 tablespace roll_7;
create rollback segment t_7_187 tablespace roll_7;
create rollback segment t_7_188 tablespace roll_7;
create rollback segment t_7_189 tablespace roll_7;
create rollback segment t_7_190 tablespace roll_7;
create rollback segment t_7_191 tablespace roll_7;
create rollback segment t_7_192 tablespace roll_7;
create rollback segment t_7_193 tablespace roll_7;
create rollback segment t_7_194 tablespace roll_7;
create rollback segment t_7_195 tablespace roll_7;
create rollback segment t_7_196 tablespace roll_7;
create rollback segment t_7_197 tablespace roll_7;
create rollback segment t_7_198 tablespace roll_7;
create rollback segment t_7_199 tablespace roll_7;
create rollback segment t_7_200 tablespace roll_7;
exit;
```

undml.sh

```
#!/bin/sh
svrmgrl <<!
connect tpcc/tpcc;

set echo on;
  alter table ware enable table lock;
  alter table dist enable table lock;
  alter table cust enable table lock;
  alter table hist enable table lock;
  alter table item enable table lock;
  alter table stok enable table lock;
  alter table ordr enable table lock;
  alter table nord enable table lock;
  alter table ordl enable table lock;
  alter table tpcc_audit_tab enable table lock;
  alter table tpcc_back_res  enable table lock;
  alter table tpcc_del_res   enable table lock;
  alter table tpcc_new_res   enable table lock;
  alter table tpcc_ord_res   enable table lock;
  alter table tpcc_pay_res   enable table lock;
  alter table tpcc_run_desc  enable table lock;
  alter table tpcc_run_int   enable table lock;
  alter table tpcc_sto_res   enable table lock;
  alter table tpcc_tpm       enable table lock;
  alter table tpcc_user_res  enable table lock;
  alter table bench_del_res  enable table lock;
  alter table bench_new_res  enable table lock;
  alter table bench_ord_res  enable table lock;
  alter table bench_pay_res  enable table lock;
  alter table bench_run_int  enable table lock;
  alter table bench_sto_res  enable table lock;
  alter table bench_user_res enable table lock;
set echo off;
exit;
!
```

undml.sql

```
REM=========================================================
========+
REM       Copyright (c) 1996  Oracle Corp, Redwood Shores, CA      |
REM                  OPEN SYSTEMS PERFORMANCE GROUP                 |
REM                       All Rights Reserved                       |
REM=========================================================
========+
REM FILENAME
REM     undml.sql
REM DESCRIPTION
REM     Enable table locks for TPC-C tables.
REM USAGE
REM     sqlplus tpcc/tpcc @undml
REM=========================================================
========*/

connect tpcc/tpcc;
spool undml.log;
set echo on;

   alter table ware enable table lock;
   alter table dist enable table lock;
   alter table cust enable table lock;
   alter table hist enable table lock;
   alter table item enable table lock;
   alter table stok enable table lock;
   alter table ordr enable table lock;
   alter table nord enable table lock;
   alter table ordl enable table lock;

set echo off;
spool off;
```

views.sql

```
connect tpcc/tpcc;
set echo on;

create or replace view wh_cust
(w_id, w_tax, c_id, c_d_id, c_w_id, c_discount, c_last, c_credit)
as select w.w_id, w.w_tax,
          c.c_id, c.c_d_id, c.c_w_id, c.c_discount, c.c_last, c.c_credit
   from cust c, ware w
   where w.w_id = c.c_w_id;

create or replace view wh_dist
(w_id, d_id, d_tax, d_next_o_id, w_tax )
as select w.w_id, d.d_id, d.d_tax, d.d_next_o_id, w.w_tax
   from dist d, ware w
   where w.w_id = d.d_w_id;

create  or replace view stock_item
(i_id, s_w_id, i_price, i_name, i_data, s_data, s_quantity,
 s_order_cnt, s_ytd, s_remote_cnt,
 s_dist_01, s_dist_02, s_dist_03, s_dist_04, s_dist_05,
 s_dist_06, s_dist_07, s_dist_08, s_dist_09, s_dist_10)
as
 select i.i_id, s_w_id, i.i_price, i.i_name, i.i_data, s_data, s_quantity,
 s_order_cnt, s_ytd, s_remote_cnt,
 s_dist_01, s_dist_02, s_dist_03, s_dist_04, s_dist_05,
 s_dist_06, s_dist_07, s_dist_08, s_dist_09, s_dist_10
  from stok s, item i
 where i.i_id = s.s_i_id;

set echo off;
```

Appendix C

scr_util.c

```
/*+*****************************************************************
*******
 *                                                                  *
 *  COPYRIGHT (c) 1999 BY                                           *
 *  COMPAQ COMPUTER CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                            *
 *                                                                  *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY  OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                    *
 *                                                                  *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY
COMPAQ  COMPUTER   *
 *  CORPORATION.                                                    *
 *                                                                  *
 *  COMPAQ ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS    *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY COMPAQ.
*
 *                                                                  *
 *                                                                  *

*******************************************************************
*****-*/
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include <ctype.h>
#include <stdarg.h>
#include <stdtypes.h>

#ifdef _WIN32
# include <nt_lib.h>
#endif

#include <prte.h>

#include <common.h>

#include <tpccstruct.h>
#include <tpccerr.h>
#include <config.h>

#define SCR_UTIL_C
#include <scr_util.h>


/* Static functions */
static print_getnet_debug(char *name, char *ret_string, char *def_string);
```

```
int
GetCallingThreadLimit( int *pCallingThreadLimit )
{
 *pCallingThreadLimit = PRTEget_worker_thread_limit( );

 return( ERR_SUCCESS );
}


int
GetConfigValue( char *Name, int *Len, char *Value )
{
 int                                    Status;

 Status = GetConfigValueTX( Name, Len, Value );
 if( SCR_E_SUCCESS == Status )
 {
  /* Convert to FFE style error code. */
  Status = ERR_SUCCESS;
 }

 return( Status );
}



int
GetConfigValueTX( char *Name, int *Len, char *Value )
{
 char                                   *locValue;
 int                                    locLen;
 char
            FullName[PRTE_MAX_NET_VAR_NAME_LENGTH+1];

 strcpy( FullName, Sut_name );
 strcat( FullName, "_" );
 strcat( FullName, Name );
 locValue = PRTEget_network_variable( FullName );

 if( NULL == locValue )
 {
  Value[0] = '\0';
  *Len = 0;
  return( ERR_CANT_FIND_VALUE );
 }
 else
 {
  locLen = strlen( locValue );
  if( locLen < *Len )
  {
   memcpy( Value, locValue, locLen+1 );
   *Len = locLen;
   return( SCR_E_SUCCESS );
  }
  else
  {
   return( ERR_VALUE_TOO_LONG );
  }
 }
}


/*******************************************************************
*********/
/*                                                                  */
/* Name: GetConnectInfo                                             */
/*                                                                  */
/* Purpose: This routine gets all the information about the FEs that we      */
/*          need.                                                   */
/*                                                                  */
/* Inputs:                                                          */
/*                                                                  */
/* Outputs:                                                         */
/*                                                                  */
/* Function Value:                                                  */
/*                                                                  */
/* Notes:                                                           */
/*                                                                  */
```

```
/*****************************************************************
*********/
int32
GetConnectInfo( int32 RunMode, int32 *pTpccUsers, int32 *pNumFEs,
                               int32 *pAdminUsers,
                               int32 *pNumConnects, connect_t **ppConnects )
{
#define COUNT_UNSPECIFIED -1
#define CONNECT_INFO            "CONNECT%d"
  char                          Field[11];
  int                           Index;
  char                          NetVarName[50];
  int32                         NetVarNum;
  int                           NumCountUnspec;
  char                          *pConnect;
  char                          Script[PATH_MAX];
  char                          *pTmpChar;
  int                           Remainder;
  int                           UcntSpec;
  int                           TotalUcnt;
  int                           TotalDcnt;
  int                           Ucnt;

  /* Get the name of the user emulation script. */
  if( NULL == ( pTmpChar = PRTEget_network_variable( "USER_SCRIPT_NAME"
)))
  {
    Script[0] = '\0';
  }
  else
  {
    strcpy( Script, pTmpChar );
  }

  /* Get total count of districts. */
  if( NULL == ( pTmpChar = PRTEget_network_variable(
"TOTAL_DISTRICT_COUNT" )))
  {
    return( SCRerr( SCR_E_TOTAL_DISTRICT_COUNT_NOT_SET ));
  }
  else
  {
    TotalDcnt = atoi( pTmpChar );
  }

  /* Get total count of users. */
  if( NULL == ( pTmpChar = PRTEget_network_variable( "TOTAL_USER_COUNT"
)))
  {
    TotalUcnt = COUNT_UNSPECIFIED;
  }
  else
  {
    TotalUcnt = atoi( pTmpChar );
  }

  /* Collect user connect info */
  *pNumConnects = 0;
  NetVarNum = 1;
  while( TRUE )
  {
    sprintf( NetVarName, CONNECT_INFO, NetVarNum++ );
    if( NULL == (pConnect = PRTEget_network_variable( NetVarName )))
    {
      break;
    }
    /* Check if there is a placeholder. I.e. an FE not to be used. */
    if( '\0' == pConnect[0] )
    {
      continue;
    }
    (*pNumConnects)++;
  }
  if( 0 == *pNumConnects )
  {
    return( SCRerr( SCR_E_NO_CONNECTS ));
  }

  /* Now we know how many connects there are, so we can allocate space. */
  *ppConnects = ( connect_t * ) malloc( *pNumConnects * sizeof( **ppConnects ));
  if( NULL == *ppConnects )
  {
    return( SCRerr( SCR_E_MALLOC_CONNECT_ARRAY ));
  }

  /* Now we need to get the connect info */
  *pNumFEs = 0;
  *pAdminUsers = 0;
  NumCountUnspec = 0;
  UcntSpec = 0;
  Index = 0;
  NetVarNum = 1;
  while( TRUE )
  {
    sprintf( NetVarName, CONNECT_INFO, NetVarNum++ );
    if( NULL == (pConnect = PRTEget_network_variable( NetVarName )))
    {
      break;
    }
    /* Check if there is a placeholder. I.e. an entry not to be used. */
    if( '\0' == pConnect[0] )
    {
      continue;
    }

    /* Parse the connect information */
    pTmpChar = SCR_next_list_element( pConnect, (*ppConnects)[Index].RTEName
);
    if( '\0' == (*ppConnects)[Index].RTEName[0] )
    {
      free( *ppConnects );
      SCRlog( "Parsing %s.", NetVarName );
      return( SCRerr( SCR_E_NO_RTE_IN_CONNECT ));
    }

    pTmpChar = SCR_next_list_element( pTmpChar, (*ppConnects)[Index].FEName );

    pTmpChar = SCR_next_list_element( pTmpChar, Field );
    if( (*ppConnects)[Index].ConfConn = ( 'T' == Field[0] || 't' == Field[0] ))
    {
      (*pNumFEs)++;
    }

    pTmpChar = SCR_next_list_element( pTmpChar, Field );
    if( (*ppConnects)[Index].AdminConn = ( 'T' == Field[0] || 't' == Field[0] ))
    {
      (*pAdminUsers)++;
    }

    pTmpChar = SCR_next_list_element( pTmpChar, Field );
    if( '\0' == Field[0] )
    {
      (*ppConnects)[Index].Ucnt = COUNT_UNSPECIFIED;
      NumCountUnspec++;
    }
    else
    {
      UcntSpec += (*ppConnects)[Index].Ucnt = atoi( Field );
    }

    pTmpChar = SCR_next_list_element( pTmpChar, (*ppConnects)[Index].Script );
    if( '\0' == (*ppConnects)[Index].Script[0] )
    {
      if( '\0' == Script[0] )
      {
                free( *ppConnects );
                SCRlog( "Parsing %s.", NetVarName );
                return( SCRerr( SCR_E_USER_SCRIPT_NAME_NOT_SET ));
      }
      strcpy( (*ppConnects)[Index].Script, Script );
    }
    Index++;
  }

  /* Check that all specified user counts agree. */
  if( 0 == NumCountUnspec )
  {
    if( COUNT_UNSPECIFIED != TotalUcnt && UcntSpec != TotalUcnt )
    {
      free( *ppConnects );
```

```
    return( SCRerr( SCR_E_USER_COUNT_MISMATCH ));
   }
   *pTpccUsers = UcntSpec;
  }
  else
  {
   if( COUNT_UNSPECIFIED == TotalUcnt )
   {
    TotalUcnt = TotalDcnt;
   }
   /* Distribute the user counts */
   *pTpccUsers = 0;
   Ucnt = ( TotalUcnt - UcntSpec ) / NumCountUnspec;
   Remainder = ( TotalUcnt - UcntSpec ) % NumCountUnspec;
   for( Index = 0; Index < *pNumConnects; Index++ )
   {
    if( COUNT_UNSPECIFIED == (*ppConnects)[Index].Ucnt )
    {
            (*ppConnects)[Index].Ucnt = Ucnt + ( Remainder-- > 0 ? 1 : 0 );
    }
    *pTpccUsers += (*ppConnects)[Index].Ucnt;
   }
  }

  /* Make sure all specified counts are in sync. */
  if( END_TO_END_MODE == RunMode &&
     TotalDcnt != *pTpccUsers )
  {
   free( *ppConnects );
   return( SCRerr( SCR_E_DISTRICT_USER_COUNT_MISMATCH ));
  }

# pragma message ("FIXME: GetConnectInfo: Should warn tester if user count won't
fully access all districts for the last warehouse.")
# pragma message ("FIXME: GetConnectInfo: Should warn tester if user count on a
given network card exceeds limit (currently 1023).")

  return( SCR_E_SUCCESS );
}


int32
SCRerr( int32 ErrCode )
{
  char                  ErrFmt[] = "\007ERROR( %d ): %s\n\n";
  int32                 Idx;
  char                  *ErrString;

  Idx = 0;
  while( '\0' != ScrErrMsgs[Idx].ErrMsg[0] &&
         ErrCode != ScrErrMsgs[Idx].ErrCode )
  {
   Idx++;
  }
  if( '\0' != ScrErrMsgs[Idx].ErrMsg[0] )
  {
   ErrString = ScrErrMsgs[Idx].ErrMsg;
  }
  else
  {
   /* No SCR error message defined for this error code, check TPCC errors. */
   Idx = 0;
   while( '\0' != errorMsgs[Idx].szMsg[0] &&
          ErrCode != errorMsgs[Idx].iError )
   {
    Idx++;
   }
   if( '\0' != errorMsgs[Idx].szMsg[0] )
   {
    ErrString = errorMsgs[Idx].szMsg;
   }
   else
   {
    ErrString = ScrErrMsgs[SCR_E_NO_ERR_MSG - SCR_E_BASE].ErrMsg;
   }
  }

  SCRlog( ErrFmt, ErrCode, ErrString );
  return( ErrCode );
}

void
SCRlog( char *Format, ... )
{
  char                  Buf[8192];
  va_list       VarArgs;


  /* Expand the format string and arguments */
  if( NULL != Format ) {
   va_start( VarArgs, Format );
   vsprintf( Buf, Format, VarArgs );
   va_end( VarArgs );
  }
  else
  {
   strcpy( Buf, "[NULL]" );
  }

  PRTEsend_console_message( Buf );
  PRTEto_log( 555, Buf );

  return;
}



/* SCR_inverse_random
   Inputs: Mean - average value needed from function
           Iexp - pointer to double for storage of result
   Return: NONE
   Implementation:  Does inverse-random function using drand48.
           Will try again if value caluclated is > Max
*/
void
SCR_inverse_random(double Mean, double Max, double *Iexp) {
  double Randval, lval;

  do {
   Randval = drand48();
   if( 0.0 == Randval )
   {
    /* We have to treat a zero value returned from the random number */
    /* generator as a special case because log( 0.0 ) would result in */
    /* an infinite negative number.  Passing an infinite negative number */
    /* onto the following equation would result in a floating point */
    /* exception.  However, mathematically, negating an infinite negative */
    /* number and then scaling it by the desired mean value would still */
    /* result in an infinite positive number which would excced the maximum */
    /* requested value.  Exceeding the maximum requested value causes a */
    /* loop to get a new random value, which is what we shall do now. */
    continue;
   }
   lval = log(Randval);
   *Iexp  = -1.0 * Mean * lval;

  } while (*Iexp > Max );
}

char *
SCR_getnet_int ( char *varname, char *default_str, int *pInt) {
  char *pnet;

  if (!varname) return NULL;

  pnet = PRTEget_network_variable (varname);
  if (pnet) *pInt = atoi(pnet);
  else *pInt = atoi(default_str);
#ifdef DEBUG
  print_getnet_debug(varname,pnet,default_str);
#endif

  return pnet;
}

char *
SCR_getnet_long ( char *varname, char *default_str, long *pLong) {
```

```
  char *pnet;

  if (!varname) return NULL;

  pnet = PRTEget_network_variable (varname);
  if (pnet) *pLong = atol(pnet);
  else *pLong = atol(default_str);
#ifdef DEBUG
  print_getnet_debug(varname,pnet,default_str);
#endif

  return pnet;
}

char *
SCR_getnet_float ( char *varname, char *default_str, float *pFloat) {
  char *pnet;

  if (!varname) return NULL;

  pnet = PRTEget_network_variable (varname);
  if (pnet) *pFloat = (float) atof(pnet);
  else *pFloat = (float) atof(default_str);
#ifdef DEBUG
  print_getnet_debug(varname,pnet,default_str);
#endif

  return pnet;
}

char *
SCR_getnet_double ( char *varname, char *default_str, double *pDouble) {
  char *pnet;

  if (!varname) return NULL;

  pnet = PRTEget_network_variable (varname);
  if (pnet) *pDouble = atof(pnet);
  else *pDouble = atof(default_str);

  return pnet;
}


char *
SCR_getnet_string ( char *varname, char *default_str, char *pString) {
  char *pnet;

  if (!varname) return NULL;

  pnet = PRTEget_network_variable (varname);

  if (pnet != NULL) strcpy(pString,pnet);

  else {
    if (default_str != NULL) strcpy(pString,default_str);
    else (pString[0] = '\0');
  }
#ifdef DEBUG
  print_getnet_debug(varname,pnet,default_str);
#endif

  return pString;
}

char *
SCR_getnet_bool ( char *varname, char *default_str, BOOL *pBool) {
  char *pnet;
  char first_letter;

  if (!varname) return NULL;

  pnet = PRTEget_network_variable (varname);
  if (pnet)
  {
    first_letter = pnet[0];
  }
  else
  {
    first_letter = default_str[0];
    pnet = default_str;
  }
  switch (first_letter)
  {
    case 't':
    case 'T':
    case '1':
      *pBool = TRUE;
      break;

    case 'f':
    case 'F':
    case '0':
    default:
      *pBool = FALSE;
      break;
    }
#ifdef DEBUG
  print_getnet_debug(varname,pnet,default_str);
#endif

  return pnet;
}

#ifdef DEBUG
static
print_getnet_debug(char *name, char *ret_string, char *def_string) {
  char temp_string[256];

  SCRlog( "Netvar: name = %s netvar = %s, default = %s\n",
                name, ret_string, def_string );
}
#endif


/*********************************************************************
********/
/*                                                                   */
/* Name: SCR_next_list_element                                       */
/*                                                                   */
/* Purpose: Return an element in Element and a pointer to the next element   */
/*          in a comma separated list                                */
/* Inputs:  List -- pointer to list                                  */
/*          Element-- pointer to string for storing element          */
/* Outputs: Returns pointer to next element in list                  */
/*                                                                   */
/*********************************************************************
********/

char
*SCR_next_list_element( char *List, char *Element )
{
#define EAT_LEADING_WHITESPACE(p) while( isspace(*(p)) ) (p)++
#define EAT_TRAILING_WHITESPACE(p) while( isspace(*((p)-1)) ) (p)--
#define IS_SEPARATOR(c) (( ',' == (c)))
  char *pTmpChar;

  pTmpChar = Element;
  if( NULL != pTmpChar )
  {
    *pTmpChar = '\0';                   /* Null terminate Element */
  }

  if( List && *List )
  {
    EAT_LEADING_WHITESPACE( List );
    while( '\0' != *List && !IS_SEPARATOR( *List ))
    {
      if( '\\' == *List )
      {
                List++;                                         /* Skip the escape char
*/
                if( '\0' == *List )
                {
                  break;
                }
      }
      if( NULL != pTmpChar )
      {
                *pTmpChar++ = *List;        /* Copy the element to Element */
```

```
    }
    List++;
   }
   if( NULL != pTmpChar )
   {
    EAT_TRAILING_WHITESPACE( pTmpChar );
    *pTmpChar = '\0';                    /* Null terminate Element */
   }

   /* Now eat up trailing separator */
   if( IS_SEPARATOR( *List ))
   {
    List++;
   }

   return( List );
  }
  else
  {
   return( NULL );
  }
}

/*****************************************************************
********/
/*                                                               */
/* Name: SCR_count_elements                                      */
/*                                                               */
/* Purpose: Return count of elements in an element list
            */
/* Inputs:  List -- pointer to list                              */
/* Outputs: Returns count of elements                            */
/*                                                               */
/*****************************************************************
********/


int
SCR_count_elements( char *List )
{
  char *pTmpChar;
  int Count = 0;

  /* Now count elements */
  pTmpChar = List;
  while( NULL != ( pTmpChar = SCR_next_list_element( pTmpChar, NULL )))
  {
    Count++;
  }

  /* This is a bit of a hack.  If the list contains one zero length element,  */
  /* the count will return 0.  What we really want is 1.  So update the count.*/
  if( 0 == Count )
  {
    Count++;
  }

  return Count;
}


void
SCR_make_delta_time_string( char *timestr, double DeltaSeconds )
{
#define ONE_MINUTE 60
#define ONE_HOUR (60*ONE_MINUTE)
  int                                  Hours;
  int                                  Minutes;
  int                                  Seconds;
  int                                  MilliSeconds;

  Seconds = (int)DeltaSeconds;
  MilliSeconds = (int)( DeltaSeconds*1000.0 - Seconds*1000.0 );

  Hours = Seconds / ONE_HOUR;
  Seconds = Seconds % ONE_HOUR;

  Minutes = Seconds / ONE_MINUTE;
  Seconds = Seconds % ONE_MINUTE;

  sprintf(timestr,"%2d:%02d:%02d.%03d", Hours, Minutes, Seconds, MilliSeconds);
}

int
SetConfigValueTX( char *Name, char *Value )
{
 char
                FullName[PRTE_MAX_NET_VAR_NAME_LENGTH+1];

 strcpy( FullName, Sut_name );
 strcat( FullName, "_" );
 strcat( FullName, Name );

 PRTEset_network_variable( FullName, Value );

 return( SCR_E_SUCCESS );
}
```

scr_util.h

```
#ifndef SCR_UTIL_H
#define SCR_UTIL_H
/*+*****************************************************************************
*******
 *                                                                 *
 *  COPYRIGHT (c) 1999 BY                                          *
 *  DIGITAL COMPUTER CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                           *
 *                                                                 *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS HEREBY   *
 *  TRANSFERRED.                                                   *
 *                                                                 *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND SHOULD NOT BE CONSTRUED AS  A COMMITMENT BY  DIGITAL
COMPUTER   *
 *  CORPORATION.                                                   *
 *                                                                 *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                                 *
 *                                                                 *

*****************************************************************
*****-*/

#define ATOI(pStr,pNext,val)\
  {\
    int                                  ival;\
    char                         *pSrc = (pStr);\
    ival = 0;\
    while ('0' <= *pSrc && *pSrc <= '9')\
    {\
      ival = (ival * 10) + (*pSrc - '0');\
      pSrc++;\
    }\
    (pNext) = pSrc;\
    (val) = ival;\
  }

#define ITOA(val,pStr,pNext)\
  {\
    int                                  ival;\
    char                         *pDst = (pStr);\
    char                         Tmp[12]; /* size of base10 32 bit # -2,147,483,648 */\
```

```
  char                    *pTmp = &Tmp[0];\
  ival = (val);\
  if( 0 > ival )\
  {\
    *pDst++ = '-';\
    ival = -ival;\
  }\
  *pTmp++ = '\0';\
  if( 0 == ival )\
  {\
     *pTmp++ = '0';\
  }\
  while( 0 < ival )\
  {\
    *pTmp++ = ( ival % 10 ) + '0';\
    ival /= 10;\
  }\
  while( '\0' != ( *pDst++ = *--pTmp ));\
  (pNext) = pDst;\
 }


typedef struct _connect_t
{
  BOOL                          ConfConn;
  BOOL                          ConfMod;
  BOOL                          AdminConn;
  int32                         AideID;
  char                          RTEName[MAXHOSTNAMELEN];
  char                          FEName[MAXHOSTNAMELEN];
  int32                         Ucnt;
  char                          Script[PATH_MAX];
} connect_t;


union dataptr {
  double *pDouble;
  int *pInt;
  float *pFloat;
};


int  GetCallingThreadLimit( int *pCallingThreadLimit );
int32 GetConnectInfo( int32 RunMode, int32 *pTpccUsers, int32 *pNumFEs,
                      int32 *pAdminUsers,
                      int32 *pNumConnects, connect_t **ppConnects );

void SCR_inverse_random (double Mean, double Max, double *Iexp);

char *SCR_getnet_int     (char *varname, char *default_str, int *pInt);
char *SCR_getnet_float   (char *varname, char *default_str, float *pFloat);
char *SCR_getnet_long    (char *varname, char *default_str, long *pLong);
char *SCR_getnet_double (char *varname, char *default_str, double *pDouble);
char *SCR_getnet_bool    (char *varname, char *default_str, BOOL *pBool);
char *SCR_getnet_string (char *varname, char *default_str, char *pString);
void          SCRlog( char *Format, ... );
int32         SCRerr( int32 ErrCode );
char *SCR_next_list_element (char *List, char *Element);
int SCR_count_elements (char *List);
void SCR_make_delta_time_string( char *timestr, double DeltaSeconds );
int SetConfigValueTX( char *Name, char *Value );
int GetConfigValueTX( char *Name, int *Len, char *Value );

/* #define's for all errors we can detect. */

#define       SCR_E_SUCCESS
              (0)

#define SCR_E_BASE                                                    200000

#define SCR_E_NO_ERR_MSG
              (SCR_E_BASE+1)
#define SCR_E_INIT
              (SCR_E_BASE+2)
#define SCR_E_MAKING_RUN_DIR
              (SCR_E_BASE+3)
#define SCR_E_NO_REDUCER
              (SCR_E_BASE+4)
#define SCR_E_CKPT_INT_TOO_LONG
              (SCR_E_BASE+5)
#define SCR_E_CKPT_INT_GREATER_THAN_MEASUREMENT_INT
              (SCR_E_BASE+6)
#define SCR_E_MEASUREMENT_NOT_INTEGRAL_MULTIPLE
              (SCR_E_BASE+7)
#define SCR_E_WARMUP_TOO_SHORT
              (SCR_E_BASE+8)
#define SCR_E_CANT_PARSE_SERVICE_CONTROL_PAGE
              (SCR_E_BASE+9)
#define SCR_E_CANT_STOP_WEB
              (SCR_E_BASE+10)
#define SCR_E_CANT_START_WEB
              (SCR_E_BASE+11)
#define SCR_E_AIDE_TIMEOUT_STOP_FE
              (SCR_E_BASE+12)
#define SCR_E_GET_BE_INFO
              (SCR_E_BASE+13)
#define SCR_E_INVALID_RUN_MODE
              (SCR_E_BASE+14)
#define SCR_E_AUDIT_FUNC_NETVARS
              (SCR_E_BASE+15)
#define SCR_E_BE_NAMES
              (SCR_E_BASE+16)
#define SCR_E_MALLOC_BE_ARRAY
              (SCR_E_BASE+17)
#define SCR_E_CODE_VERSION_NOT_SET
              (SCR_E_BASE+18)
#define SCR_E_RUN_DIR_NOT_SET
              (SCR_E_BASE+19)
#define SCR_E_CODE_VERSION_MISMATCH
              (SCR_E_BASE+20)
#define SCR_E_OPEN_BIN_LOG_FILE
              (SCR_E_BASE+21)
#define SCR_E_WRITE_BIN_LOG_FILE_HEADER_SIZE
              (SCR_E_BASE+22)
#define SCR_E_WRITE_BIN_LOG_FILE_HEADER
              (SCR_E_BASE+23)
#define SCR_E_MASTER_ID_NOT_SET
              (SCR_E_BASE+24)
#define SCR_E_RUN_NUMBER_NOT_SET
              (SCR_E_BASE+25)
#define SCR_E_VERSION_NUMBER_NOT_SET
              (SCR_E_BASE+26)
#define SCR_E_REDUCER_UPDATE_INTERVAL_NOT_SET
              (SCR_E_BASE+27)
#define SCR_E_REDUCER_HEADER_INTERVAL_NOT_SET
              (SCR_E_BASE+28)
#define SCR_E_OPEN_AIDE_DATA_FILE
              (SCR_E_BASE+29)
#define SCR_E_DATABASE_TYPE_NOT_SET
              (SCR_E_BASE+30)
#define SCR_E_INVALID_DATABASE_TYPE
              (SCR_E_BASE+31)
#define SCR_E_AIDE_INIT_TIMEOUT
              (SCR_E_BASE+32)
#define SCR_E_TOTAL_DISTRICT_COUNT_NOT_SET
              (SCR_E_BASE+33)
#define SCR_E_NO_CONNECTS
              (SCR_E_BASE+34)
#define SCR_E_NO_RTE_IN_CONNECT
              (SCR_E_BASE+35)
#define SCR_E_CALC_VALID_TXN_MIX
              (SCR_E_BASE+36)
#define SCR_E_REDUCER_INIT_TIMEOUT
              (SCR_E_BASE+37)
#define SCR_E_LOAD_FE_TIMEOUT
              (SCR_E_BASE+38)
#define SCR_E_CONNECT_APPLICATION
              (SCR_E_BASE+39)
#define SCR_E_SEND
              (SCR_E_BASE+40)
#define SCR_E_WAIT_FOR
              (SCR_E_BASE+41)
#define SCR_E_HTTP_STATUS_OFFSET
              (SCR_E_BASE+42)
#define SCR_E_FORM_TYPE_OFFSET
              (SCR_E_BASE+43)
#define SCR_E_WEB_PAGE_TOO_SHORT
              (SCR_E_BASE+44)
#define SCR_E_WEB_PAGE_STATUS_CODE
              (SCR_E_BASE+45)
```

```
#define SCR_E_ERROR_STATUS_CODE
        (SCR_E_BASE+46)
#define SCR_E_ERROR_WEB_PAGE
        (SCR_E_BASE+47)
#define SCR_E_WRONG_WEB_PAGE
        (SCR_E_BASE+48)
#define SCR_E_AIDE_TASK_TIMEOUT
        (SCR_E_BASE+49)
#define SCR_E_AIDES_FAILED_TASK
        (SCR_E_BASE+50)
#define SCR_E_BE_CKPT_TIMEOUT
        (SCR_E_BASE+51)
#define SCR_E_BE_CKPT_FAILED
        (SCR_E_BASE+52)
#define SCR_E_AIDE_CKPT_INIT_TIMEOUT
        (SCR_E_BASE+53)
#define SCR_E_USER_LOGIN_TIMEOUT
        (SCR_E_BASE+54)
#define SCR_E_USER_LOGIN_FAILED
        (SCR_E_BASE+55)
#define SCR_E_LOG_DIR_NOT_SET
        (SCR_E_BASE+56)
#define SCR_E_CGI_SCRIPT_NAME_NOT_SET
        (SCR_E_BASE+57)
#define SCR_E_INVALID_TXN_TYPE_REQUESTED
        (SCR_E_BASE+58)
#define SCR_E_USER_START_FAILED
        (SCR_E_BASE+59)
#define SCR_E_USER_START_TIMEOUT
        (SCR_E_BASE+60)
#define SCR_E_USER_STOP_FAILED
        (SCR_E_BASE+61)
#define SCR_E_USER_STOP_TIMEOUT
        (SCR_E_BASE+62)
#define SCR_E_AIDE_TIMEOUT_START_FE
        (SCR_E_BASE+63)
#define SCR_E_AIDES_FAILED_START
        (SCR_E_BASE+64)
#define SCR_E_UNUSED_4
        (SCR_E_BASE+65)
#define SCR_E_UNUSED_5
        (SCR_E_BASE+66)
#define SCR_E_AIDE_CKPT_RUN_TIMEOUT
        (SCR_E_BASE+67)
#define SCR_E_AIDE_CKPT_STOP_TIMEOUT
        (SCR_E_BASE+68)
#define SCR_E_WRITE_SENDERS_ID
        (SCR_E_BASE+69)
#define SCR_E_WRITE_MSG_TYPE
        (SCR_E_BASE+70)
#define SCR_E_WRITE_MSG_SIZE
        (SCR_E_BASE+71)
#define SCR_E_WRITE_MSG
        (SCR_E_BASE+72)
#define SCR_E_AIDE_EXIT_TIMEOUT
        (SCR_E_BASE+73)
#define SCR_E_OPEN_C_LAST_FILE
        (SCR_E_BASE+74)
#define SCR_E_SYNC_USER_TIMEOUT
        (SCR_E_BASE+75)
#define SCR_E_MALLOC_CONNECT_ARRAY
        (SCR_E_BASE+76)
#define SCR_E_MALLOC_USER_STOP_ARRAY
        (SCR_E_BASE+77)
#define SCR_E_AIDES_FAILED_STOP
        (SCR_E_BASE+78)
#define SCR_E_ACTION_NOT_SET
        (SCR_E_BASE+79)
#define SCR_E_UNUSED_1
        (SCR_E_BASE+80)
#define SCR_E_DELI_OFFSET
        (SCR_E_BASE+81)
#define SCR_E_DURA_PARSE_WAREHOUSE
        (SCR_E_BASE+82)
#define SCR_E_DURA_PARSE_DISTRICT
        (SCR_E_BASE+83)
#define SCR_E_DURA_PARSE_ORDER_ID
        (SCR_E_BASE+84)
#define SCR_E_GREETING
        (SCR_E_BASE+85)
#define SCR_E_CONNECT_ABORTED
        (SCR_E_BASE+86)
#define SCR_E_MENU_BAR
        (SCR_E_BASE+87)
#define SCR_E_AIDE_GET_LOG_SIZE
        (SCR_E_BASE+88)
#define SCR_E_RUN_ID_NOT_SET
        (SCR_E_BASE+89)
#define SCR_E_AIDE_SKIPPING_STATS
        (SCR_E_BASE+90)
#define SCR_E_AIDE_SWITCH_LOG
        (SCR_E_BASE+91)
#define SCR_E_NO_VALUE
        (SCR_E_BASE+92)
#define SCR_E_NO_TYPE
        (SCR_E_BASE+93)
#define SCR_E_NO_DATA
        (SCR_E_BASE+94)
#define SCR_E_UNUSED_6
        (SCR_E_BASE+95)
#define SCR_E_REDUCER_ID_NOT_SET
        (SCR_E_BASE+96)
#define SCR_E_RTE_NAMES_NOT_SET
        (SCR_E_BASE+97)
#define SCR_E_USER_COUNT_MISMATCH
        (SCR_E_BASE+98)
#define SCR_E_LOAD_FE_FAILURE
        (SCR_E_BASE+99)
#define SCR_E_USER_CREATE_FAILED
        (SCR_E_BASE+100)
#define SCR_E_USER_INIT_FAILED
        (SCR_E_BASE+101)
#define SCR_E_USER_INIT_TIMEOUT
        (SCR_E_BASE+102)
#define SCR_E_FIRST_USER_ID_NOT_SET
        (SCR_E_BASE+103)
#define SCR_E_VALUE_MODIFIED
        (SCR_E_BASE+104)
#define SCR_E_CANT_PARSE_GET_REG_PAGE
        (SCR_E_BASE+105)
#define SCR_E_CANT_INIT_TIME_COUNTER
        (SCR_E_BASE+106)
#define SCR_E_USER_SCRIPT_NAME_NOT_SET
        (SCR_E_BASE+107)
#define SCR_E_WRONG_DATA_ITEM_COUNT
        (SCR_E_BASE+108)
#define SCR_E_AIDE_TIMEOUT_CONFIG_FE
        (SCR_E_BASE+109)
#define SCR_E_AIDES_FAILED_CONFIG
        (SCR_E_BASE+110)
#define SCR_E_NOT_CORRECT_TRANS
        (SCR_E_BASE+111)
#define SCR_E_INVALID_CONNECT
        (SCR_E_BASE+112)
#define SCR_E_INVALID_DISCONNECT
        (SCR_E_BASE+113)
#define SCR_E_UI_NOT_SET
        (SCR_E_BASE+114)
#define SCR_E_UI_LOAD_FAILED
        (SCR_E_BASE+115)
#define SCR_E_NO_PARSE_INIT_FAILED
        (SCR_E_BASE+116)
#define SCR_E_DURA_PARSE_EXECUTION_STATUS
        (SCR_E_BASE+117)
#define SCR_E_UI_UNLOAD_FAILED
        (SCR_E_BASE+118)
#define SCR_E_AIDE_CKPT_DISCONNECT_TIMEOUT
        (SCR_E_BASE+119)
#define SCR_E_DISTRICT_USER_COUNT_MISMATCH
        (SCR_E_BASE+120)
#define SCR_E_REDUCER_CREATE_FAILED
        (SCR_E_BASE+121)
#define SCR_E_AIDE_CREATE_FAILED
        (SCR_E_BASE+122)

#define SCR_E_MAX_ERROR
        (SCR_E_BASE+122)

/* Error message strucure. */
typedef struct _scr_e_msg_t
```

```
{
  int32                 ErrCode;     /* error id of message */
  char                  ErrMsg[256];          /* message to sent to browser */
} scr_e_msg_t;

#ifdef SCR_UTIL_C
/* Error messages for all errors we can detect. */
scr_e_msg_t ScrErrMsgs[] =
{
  { SCR_E_SUCCESS, "Success, no error." },
  { SCR_E_NO_ERR_MSG, "No error message available for this error code." },
  { SCR_E_INIT, "Failed to initialize." },
  { SCR_E_MAKING_RUN_DIR, "Failed to make the run directory." },
  { SCR_E_NO_REDUCER, "Failed to get the reducer's PRTE user id." },
  { SCR_E_CKPT_INT_TOO_LONG, "Checkpoint interval > 1800 seconds, this
violates the benchmark specification." },
  { SCR_E_CKPT_INT_GREATER_THAN_MEASUREMENT_INT, "Checkpoint
interval > measurement interval, this violates the benchmark specification." },
  { SCR_E_MEASUREMENT_NOT_INTEGRAL_MULTIPLE, "Measurement
interval is not an integral multiple of checkpoint interval, this violates the benchmark
specification." },
  { SCR_E_WARMUP_TOO_SHORT, "The warmup interval is too short to support
requested checkpoint interval." },
  { SCR_E_CANT_PARSE_SERVICE_CONTROL_PAGE, "Error parsing the service
control response page." },
  { SCR_E_CANT_STOP_WEB, "Unable to stop the Web Server." },
  { SCR_E_CANT_START_WEB, "Unable to start the Web Server." },
  { SCR_E_AIDE_TIMEOUT_STOP_FE, "Timed out waiting for aides to stop
application code." },
  { SCR_E_GET_BE_INFO, "Problem getting back end information." },
  { SCR_E_INVALID_RUN_MODE, "Invalid run mode specified." },
  { SCR_E_AUDIT_FUNC_NETVARS, "Problem getting audit function network
variables." },
  { SCR_E_BE_NAMES, "Unable to get BE_NAMES network variable." },
  { SCR_E_MALLOC_BE_ARRAY, "Unable to malloc space for BE array." },
  { SCR_E_CODE_VERSION_NOT_SET, "Code version network variable not set." },
  { SCR_E_RUN_DIR_NOT_SET, "Run directory network variable not set." },
  { SCR_E_CODE_VERSION_MISMATCH, "Code version mismatch with Master."
},
  { SCR_E_OPEN_BIN_LOG_FILE, "Failed to open the binary data log file." },
  { SCR_E_WRITE_BIN_LOG_FILE_HEADER_SIZE, "Error writing binary data log
file header size." },
  { SCR_E_WRITE_BIN_LOG_FILE_HEADER, "Error writing binary data log file
header." },
  { SCR_E_MASTER_ID_NOT_SET, "Master ID network variable not set." },
  { SCR_E_RUN_NUMBER_NOT_SET, "Run number network variable not set." },
  { SCR_E_VERSION_NUMBER_NOT_SET, "Version number network variable not
set." },
  { SCR_E_REDUCER_UPDATE_INTERVAL_NOT_SET, "Reducer update interval
network variable not set." },
  { SCR_E_REDUCER_HEADER_INTERVAL_NOT_SET, "Reducer header update
interval network variable not set." },
  { SCR_E_OPEN_AIDE_DATA_FILE, "Error opening aide data file." },
  { SCR_E_DATABASE_TYPE_NOT_SET, "Database type network variable not
set." },
  { SCR_E_INVALID_DATABASE_TYPE, "Invalid database type" },
  { SCR_E_AIDE_INIT_TIMEOUT, "Timed out waiting for Aides to initialize." },
  { SCR_E_TOTAL_DISTRICT_COUNT_NOT_SET, "TOTAL_DISTRICT_COUNT
network variable not set. " },
  { SCR_E_NO_CONNECTS, "No connect network variables have been specified." },
  { SCR_E_NO_RTE_IN_CONNECT, "No RTE was specified in a CONNECT
network variable." },
  { SCR_E_CALC_VALID_TXN_MIX, "Failed to calculate a valid transaction mix."
},
  { SCR_E_REDUCER_INIT_TIMEOUT, "Timed out waiting for Reducer to
initialize." },
  { SCR_E_LOAD_FE_TIMEOUT, "Timed out waiting for FE(s) to load." },
  { SCR_E_CONNECT_APPLICATION, "Error trying to connect to the application
(SUT)." },
  { SCR_E_SEND, "Error sending data to the application (SUT)." },
  { SCR_E_WAIT_FOR, "Error waiting for reply from application (SUT)." },
  { SCR_E_HTTP_STATUS_OFFSET, "Couldn't determine HTTP status offset." },
  { SCR_E_FORM_TYPE_OFFSET, "Couldn't determine the form type offset." },
  { SCR_E_WEB_PAGE_TOO_SHORT, "Received a web page that is too short to be
from our dll." },
  { SCR_E_WEB_PAGE_STATUS_CODE, "Couldn't find form status code in web
page." },
  { SCR_E_ERROR_STATUS_CODE, "Received an error page from our dll, but
couldn't parse the status code." },
  { SCR_E_ERROR_WEB_PAGE, "Received and error page from our dll." },
  { SCR_E_WRONG_WEB_PAGE, "Received a valid web page from our dll, but it
isn't the one we expected." },
  { SCR_E_AIDE_TASK_TIMEOUT, "Timed out waiting for all aides to perform a
task." },
  { SCR_E_AIDES_FAILED_TASK, "One or more aides failed to perform a task." },
  { SCR_E_BE_CKPT_TIMEOUT, "Timed out waiting for all aides to do their
backend checkpoint init." },
  { SCR_E_BE_CKPT_FAILED, "One or more aides failed to do their backend
checkpoint init." },
  { SCR_E_AIDE_CKPT_INIT_TIMEOUT, "Timed out waiting for all aides to do
their checkpoint init." },
  { SCR_E_USER_LOGIN_TIMEOUT, "Timed out waiting for all users to confirm
login." },
  { SCR_E_USER_LOGIN_FAILED, "A user failed to log in successfully." },
  { SCR_E_LOG_DIR_NOT_SET, "Log dir network variable not set." },
  { SCR_E_CGI_SCRIPT_NAME_NOT_SET, "CGI script name network variable not
set. " },
  { SCR_E_INVALID_TXN_TYPE_REQUESTED, "An invalid transcation type was
requested." },
  { SCR_E_USER_START_FAILED, "User(s) failed to start doing transactions." },
  { SCR_E_USER_START_TIMEOUT, "Timed out waiting for all users to start doing
transactions." },
  { SCR_E_USER_STOP_FAILED, "User(s) failed to confirm stop." },
  { SCR_E_USER_STOP_TIMEOUT, "Timed out waiting for all users to confirm
stop." },
  { SCR_E_AIDE_TIMEOUT_START_FE, "Timed out waiting for aides to start
application code." },
  { SCR_E_AIDES_FAILED_START, "Not all aides successfully started application
code." },
  { SCR_E_UNUSED_4, "Unused message 4." },
  { SCR_E_UNUSED_5, "Unused message 5." },
  { SCR_E_AIDE_CKPT_RUN_TIMEOUT, "Timed out waiting for aides to start
checkpoint loop." },
  { SCR_E_AIDE_CKPT_STOP_TIMEOUT, "Timed out waiting for aides to stop
checkpoint loop." },
  { SCR_E_WRITE_SENDERS_ID, "Failed to write sender's id to binary log file." },
  { SCR_E_WRITE_MSG_TYPE, "Failed to write message type to binary log file." },
  { SCR_E_WRITE_MSG_SIZE, "Failed to write message size to binary log file." },
  { SCR_E_WRITE_MSG, "Failed to write message to binary log file." },
  { SCR_E_AIDE_EXIT_TIMEOUT, "Timed out waiting for all aides to exit." },
  { SCR_E_OPEN_C_LAST_FILE, "Unable to open file to write C_LAST constant."
},
  { SCR_E_SYNC_USER_TIMEOUT, "Timed out on delay to allow users to catch up
with in sync offset." },
  { SCR_E_MALLOC_CONNECT_ARRAY, "Unable to malloc space for the connect
array." },
  { SCR_E_MALLOC_USER_STOP_ARRAY, "Unable to malloc space for the user
stop array." },
  { SCR_E_AIDES_FAILED_STOP, "Not all aides successfully stopped application
code." },
  { SCR_E_ACTION_NOT_SET, "Action network variable not set." },
  { SCR_E_UNUSED_1, "Unused message 1." },
  { SCR_E_DELI_OFFSET, "Couldn't determine deli record offset." },
  { SCR_E_DURA_PARSE_WAREHOUSE, "Couldn't locate warehouse ID in
response page." },
  { SCR_E_DURA_PARSE_DISTRICT, "Couldn't locate district ID in response
page." },
  { SCR_E_DURA_PARSE_ORDER_ID, "Couldn't locate order ID in response page."
},
  { SCR_E_GREETING, "Failed to get the Greeting Form." },
  { SCR_E_CONNECT_ABORTED, "Asked to stop while logging in." },
  { SCR_E_MENU_BAR, "Failed to get the Menu Bar Form." },
  { SCR_E_AIDE_GET_LOG_SIZE, "Failed to get log size." },
  { SCR_E_RUN_ID_NOT_SET, "Run ID network variable not set." },
  { SCR_E_AIDE_SKIPPING_STATS, "WARNING: Unable to read file -- skipping
Oracle stats." },
  { SCR_E_AIDE_SWITCH_LOG, "Failed to switch log." },
  { SCR_E_NO_VALUE, "No value found." },
  { SCR_E_NO_TYPE, "No type found." },
  { SCR_E_NO_DATA, "No data found." },
  { SCR_E_UNUSED_6, "Unused message 6." },
  { SCR_E_REDUCER_ID_NOT_SET, "REDUCER_ID network variable not set." },
  { SCR_E_RTE_NAMES_NOT_SET, "RTE names network variable not set." },
  { SCR_E_USER_COUNT_MISMATCH, "The count of users in each CONNECT do
not sum to the TOTAL_USER_COUNT." },
  { SCR_E_LOAD_FE_FAILURE, "FE(s) failed to load correctly." },
  { SCR_E_USER_CREATE_FAILED, "The creation of TPC-C users failed." },
  { SCR_E_USER_INIT_FAILED, "User(s) failed to confirm initialization." },
  { SCR_E_USER_INIT_TIMEOUT, "Timed out waiting for all users to confirm
initialization." },
```

```
  { SCR_E_FIRST_USER_ID_NOT_SET, "FIRST_USER_ID network variable not
set." },
  { SCR_E_VALUE_MODIFIED, "Registry setting was modified." },
  { SCR_E_CANT_PARSE_GET_REG_PAGE, "Error parsing GetRegistry response
page." },
  { SCR_E_CANT_INIT_TIME_COUNTER, "Error initializing time value used for
scale calculation of tpmC." },
  { SCR_E_USER_SCRIPT_NAME_NOT_SET, "User script name not specified in
CONNECT nor is USER_SCRIPT_NAME network variable set." },
  { SCR_E_WRONG_DATA_ITEM_COUNT, "Configuration value list must have
either one value or match the number of CONNECT# statements." },
  { SCR_E_AIDE_TIMEOUT_CONFIG_FE, "Timed out waiting for all aides to
configure FEs." },
  { SCR_E_AIDES_FAILED_CONFIG, "Not all aides successfully configured FEs."
},
  { SCR_E_NOT_CORRECT_TRANS, "Did not recieve the expected transaction
response." },
  { SCR_E_INVALID_CONNECT, "The specified connection mode is invalid." },
  { SCR_E_INVALID_DISCONNECT, "The specified disconnect mode is invalid." },
  { SCR_E_UI_NOT_SET, "The UI network variable is either not defined or not
properly defined. " },
  { SCR_E_UI_LOAD_FAILED, "UI load failed. " },
  { SCR_E_NO_PARSE_INIT_FAILED, "Unable to initialize NewOrder parsing
function." },
  { SCR_E_DURA_PARSE_EXECUTION_STATUS, "Unable to initialize durability
parsing function." },
  { SCR_E_UI_UNLOAD_FAILED, "UI unload failed. " },
  { SCR_E_AIDE_CKPT_DISCONNECT_TIMEOUT, "Timed out waiting for all
aides to do their checkpoint disconnect." },
  { SCR_E_DISTRICT_USER_COUNT_MISMATCH, "The specified district and
user count(s) don't agree."},
  { SCR_E_REDUCER_CREATE_FAILED, "Failed to create Reducer." },
  { SCR_E_AIDE_CREATE_FAILED, "Failed to create Aides." },
  { 0, "" }
};
#else
extern scr_e_msg_t ScrErrMsgs[];
#endif /* SCR_UTIL_C */

#endif /* SCR_UTIL_H */
```

tpcc.c

```
/*+          FILE:       TPCC.C
 *                       Microsoft TPC-C Kit Ver. 3.00.000
 *                       Audited 08/23/96          By Francois Raab
 *
 *                       Copyright Microsoft, 1996
 *                       Copyright Digital Equipment Corp., 1997
 *
 *          PURPOSE:  Main module for TPCC.DLL which is an ISAPI service dll.
 *          Author:             Philip Durr
 *                                        philipdu@Microsoft.com
 *
 *          MODIFICATIONS:
 *
 *                       Routines substantially modified by:
 *                                  Anne Bradley          Digital Equipment
Corp.
 *                                  Bill Carr      Digital Equipment Corp.
 *
 */
/*+*****************************************************************************
*******
 *                                                  *
 *  COPYRIGHT (c) 1997, 2000 BY                     *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                            *
 *                                                  *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY  OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                    *
 *                                                  *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                    *
 *                                                  *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                  *
 *                                                  *

*****************************************************************************
*****-*/


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

#define TPCC_C

#include <tpccerr.h>
#include <tpccstruct.h>
#include <tpccapi.h>
#include <httpext.h>

#include <tpcc.h>
#include <web_ui.h>

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

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

```
   else
     *szDest++ = *szPic;
   szPic++;
  }
  *szDest = 0;

  return;
}

/* FUNCTION: int ParseNewOrderQuery( char *pProcessedQuery[],
 *                                                  NewOrderData
*pNewOrderData )
 *
 * PURPOSE:              This function extracts and validates the new order query
 *                       from an http command string.
 *
 * ARGUMENTS:    char        *pProcessedQuery[]    array of char* that
points to
 *                                                              the value
of each name-value
 *                                                              pair.
 *                NewOrderData  *pNewOrderData    pointer to new order
data
 *                                                              structure
 *
 * RETURNS:      int         ERR_SUCCESS                     input data
successfully parsed
 *                           error_code              reason for failure
 *
 * COMMENTS:     None
 *
 */

int ParseNewOrderQuery(char *pQueryString, NewOrderData *pNewOrderData)
{
  char    *ptr;
  int                   i;
  short         items;
  char          *pProcessedQuery[MAXNEWORDERVALS];


  PARSE_QUERY_STRING(pQueryString, MAXNEWORDERVALS,
                                newOrderStrs, pProcessedQuery);

  if ( !GetValuePtr(pProcessedQuery, DID, &ptr) )
    return ERR_NEWORDER_FORM_MISSING_DID;

  GetNumeric(ptr, &pNewOrderData->d_id);
  if(0 == pNewOrderData->d_id)
    return ERR_NEWORDER_DISTRICT_INVALID;


  if ( !GetValuePtr(pProcessedQuery, CID, &ptr) )
    return ERR_NEWORDER_CUSTOMER_KEY;

  if( !GetNumeric(ptr, &pNewOrderData->c_id))
    return ERR_NEWORDER_CUSTOMER_INVALID;

  pNewOrderData->o_all_local = 1;


  for(i=0, items=0; i<MAX_OL; i++)
  {
    if( !GetValuePtr(pProcessedQuery, i*3+IID00, &ptr))
      return ERR_NEWORDER_MISSING_IID_KEY;
    if(*ptr != '&' && *ptr)
    {
      if(!GetNumeric(ptr, &pNewOrderData->o_ol[items].ol_i_id))
                return ERR_NEWORDER_ITEMID_INVALID;

      if(!GetValuePtr(pProcessedQuery, i*3+SP00, &ptr))
                return ERR_NEWORDER_MISSING_SUPPW_KEY;
      if(!GetNumeric(ptr, &pNewOrderData->o_ol[items].ol_supply_w_id))
                return ERR_NEWORDER_SUPPW_INVALID;
      if ( pNewOrderData->o_all_local &&
                  pNewOrderData->o_ol[items].ol_supply_w_id !=
                  pNewOrderData->w_id )
                pNewOrderData->o_all_local = 0;
      if(!GetValuePtr(pProcessedQuery, i*3+QTY00, &ptr))
                return ERR_NEWORDER_MISSING_QTY_KEY;
      if(!GetNumeric(ptr, &pNewOrderData->o_ol[items].ol_quantity))
                return ERR_NEWORDER_QTY_INVALID;
      if ( pNewOrderData->o_ol[items].ol_i_id >= 1000000 ||
                   pNewOrderData->o_ol[items].ol_i_id < 1 )
                return ERR_NEWORDER_ITEMID_RANGE;
      if ( pNewOrderData->o_ol[items].ol_quantity >= 100 ||
                   pNewOrderData->o_ol[items].ol_quantity < 1 )
                return ERR_NEWORDER_QTY_RANGE;
      items++;
    }
    else
    {
      if(!GetValuePtr(pProcessedQuery, i*3+SP00, &ptr))
                return ERR_NEWORDER_MISSING_SUPPW_KEY;
      if(*ptr != '&' && *ptr)
                return ERR_NEWORDER_SUPPW_WITHOUT_ITEMID;

      if(!GetValuePtr(pProcessedQuery, i*3+QTY00, &ptr))
                return ERR_NEWORDER_MISSING_QTY_KEY;
      if(*ptr != '&' && *ptr)
                return ERR_NEWORDER_QTY_WITHOUT_ITEMID;
    }
  }
  if ( items == 0 )
    return ERR_NEWORDER_NOITEMS_ENTERED;

  pNewOrderData->o_ol_cnt = items;

  return ERR_SUCCESS;
}

/* FUNCTION: int ParseOrderStatusQuery( char *pProcessedQuery[],
 *                                                  OrderStatusData
*pOrderStatusData )
 *
 * PURPOSE:              This function extracts and validates the order status query
 *                       from an http command string.
 *
 * ARGUMENTS:    char        *pProcessedQuery[]    array of char* that
points to
 *                                                              the value
of each name-value
 *                                                              pair.
 *                OrderStatusData *pOrderStatusData pointer to new order
data
 *                                                              structure
 *
 * RETURNS:      int         ERR_SUCCESS                     input data
successfully parsed
 *                           error_code              reason for failure
 *
 * COMMENTS:     None
 *
 */
int ParseOrderStatusQuery(char *pQueryString,
                                        OrderStatusData *pOrderStatusData)
{
  char          szTmp[26];
  char    *ptr;
  char          *pSzTmp;
  char          *pProcessedQuery[MAXORDERSTATUSVALS];


  PARSE_QUERY_STRING(pQueryString, MAXORDERSTATUSVALS,
                                orderStatusStrs, pProcessedQuery);

  if ( !GetValuePtr(pProcessedQuery, DID, &ptr) )
    return ERR_ORDERSTATUS_MISSING_DID_KEY;
  if ( !GetNumeric(ptr, &pOrderStatusData->d_id) )
    return ERR_ORDERSTATUS_DID_INVALID;

  if ( !GetValuePtr(pProcessedQuery, CID, &ptr) )
    return ERR_ORDERSTATUS_MISSING_CID_KEY;

  if ( *ptr == '&' || !(*ptr))
  {
    pSzTmp = szTmp;
    pOrderStatusData->c_id = INVALID_C_ID;
    if ( !GetValuePtr(pProcessedQuery, CLT_O, &ptr) )
      return ERR_ORDERSTATUS_MISSING_CLT_KEY;
```

```
  while(*ptr != '&' && *ptr)
  {
   *pSzTmp = *ptr;
   pSzTmp++;
   ptr++;
  }
  *pSzTmp = '\0';
  _strupr( szTmp );
  strcpy(pOrderStatusData->c_last, szTmp);
  if ( strlen(pOrderStatusData->c_last) > 16 )
   return ERR_ORDERSTATUS_CLT_RANGE;
 }
 else
 {
  if (!GetNumeric(ptr, &pOrderStatusData->c_id))
   return ERR_ORDERSTATUS_CID_INVALID;
  if ( !GetValuePtr(pProcessedQuery, CLT_O, &ptr) )
   return ERR_ORDERSTATUS_MISSING_CLT_KEY;
  if ( *ptr != '&' && *ptr)
   return ERR_ORDERSTATUS_CID_AND_CLT;
 }

 return ERR_SUCCESS;
}

/* FUNCTION: int ParsePaymentQuery( char *pProcessedQuery[],
 *                                  PaymentData *pPaymentData )
 *
 * PURPOSE:          This function extracts and validates the payment query
 *                   from an http command string.
 *
 * ARGUMENTS:        char          *pProcessedQuery[]    array of char* that
points to
 *                                                       the value
of each name-value
 *                                                       pair.
 *                   PaymentData  *pPaymentData          pointer to payment
data
 *                                                       structure
 *
 * RETURNS:          int           ERR_SUCCESS           input data
successfully parsed
 *                                 error_code            reason for failure
 *
 * COMMENTS:         None
 *
 */

int ParsePaymentQuery(char *pQueryString, PaymentData *pPaymentData)
{
 char        szTmp[26];
 char        *ptr;
 char    *pPtr;
 char        *pSzTmp;
 char        *pProcessedQuery[MAXPAYMENTVALS];


 PARSE_QUERY_STRING(pQueryString, MAXPAYMENTVALS,
                    paymentStrs, pProcessedQuery);

 if ( !GetValuePtr(pProcessedQuery, DID, &ptr) )
  return ERR_PAYMENT_MISSING_DID_KEY;
 if ( !GetNumeric(ptr, &pPaymentData->d_id) )
  return ERR_PAYMENT_DISTRICT_INVALID;

 if ( !GetValuePtr(pProcessedQuery, CID, &ptr) )
  return ERR_PAYMENT_MISSING_CID_KEY;


 if(*ptr == '&' || !(*ptr))
 {
  pPaymentData->c_id = INVALID_C_ID;
  pSzTmp = szTmp;
  if ( !GetValuePtr(pProcessedQuery, CLT_P, &ptr) )
   return ERR_PAYMENT_MISSING_CLT;
  if (*ptr == '&' || !(*ptr))
   return ERR_PAYMENT_MISSING_CID_CLT;
  while(*ptr != '&' && *ptr)
  {
   *pSzTmp = *ptr;
   pSzTmp++;
   ptr++;
  }
  *pSzTmp = '\0';
  _strupr( szTmp );

  strcpy(pPaymentData->c_last, szTmp);
  if ( strlen(pPaymentData->c_last) > 16 )
   return ERR_PAYMENT_LAST_NAME_TO_LONG;
 }
 else
 {
  if (!GetNumeric(ptr, &pPaymentData->c_id))
    return ERR_PAYMENT_CUSTOMER_INVALID;
  if ( !GetValuePtr(pProcessedQuery, CLT_P, &ptr) )
   return ERR_PAYMENT_MISSING_CLT_KEY;
  if(*ptr != '&' && *ptr)
   return ERR_PAYMENT_CID_AND_CLT;
 }

 if ( !GetValuePtr(pProcessedQuery, CDI, &ptr) )
  return ERR_PAYMENT_MISSING_CDI_KEY;
 if ( !GetNumeric(ptr, &pPaymentData->c_d_id) )
  return ERR_PAYMENT_CDI_INVALID;

 if ( !GetValuePtr(pProcessedQuery, CWI, &ptr) )
  return ERR_PAYMENT_MISSING_CWI_KEY;

 if ( !GetNumeric(ptr, &pPaymentData->c_w_id) )
  return ERR_PAYMENT_CWI_INVALID;



 if ( !GetValuePtr(pProcessedQuery, HAM, &ptr) )
  return ERR_PAYMENT_MISSING_HAM_KEY;

 pPtr = ptr;
 while( *pPtr != '&' && *pPtr)
 {
  if ( *pPtr == '.' )
  {
   pPtr++;
   if ( !*pPtr )
            break;
   if ( *pPtr < '0' || *pPtr > '9' )
            return ERR_PAYMENT_HAM_INVALID;
   pPtr++;
   if ( !*pPtr )
            break;
   if ( *pPtr < '0' || *pPtr > '9' )
            return ERR_PAYMENT_HAM_INVALID;
   if ( !*pPtr )
            return ERR_PAYMENT_HAM_INVALID;
  }
  else if ( *pPtr < '0' || *pPtr > '9' )
   return ERR_PAYMENT_HAM_INVALID;
  pPtr++;
 }

 pPaymentData->h_amount = atof(ptr);
 if ( pPaymentData->h_amount >= 10000.00 || pPaymentData->h_amount < 0 )
  return ERR_PAYMENT_HAM_RANGE;

 return ERR_SUCCESS;
}
```

tpcc.h

```
#ifndef TPCC_H
#define TPCC_H

/*+*****************************************************************************
*******
 *                                                                  *
 *  COPYRIGHT (c) 1997, 2000 BY                                     *
 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*
```

```
 *  ALL RIGHTS RESERVED.                                        *
 *                                                              *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                *
 *                                                              *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL
EQUIPMENT   *
 *  CORPORATION.                                                *
 *                                                              *
 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
 *                                                              *
 *                                                              *

**********************************************************************
*****-*/

/*+
 * Abstract: This is the header file for web_ui.c.  it contains the
 *              function prototypes for the routines that are called outside web_ui.c
 *
 * Author: A Bradley
 * Creation Date: May 1997
 *
 * Modified history:
 *
 *
 */


#if defined WEB_UI_C || defined TPCC_C

void FormatString(char *szDest, char *szPic, char *szSrc);

int ParseNewOrderQuery(char *pQueryString, NewOrderData *pNewOrderData);
int ParsePaymentQuery(char *pQueryString, PaymentData *pPaymentData);
int ParseOrderStatusQuery(char *pQueryString,
                                        OrderStatusData *pOrderStatusData);
#endif /* defined WEB_UI_C || defined TPCC_C */

#endif /* TPCC_H */
```

tpcc_gen.c

```
/*+**********************************************************************
*******
 *                                                              *
 *  COPYRIGHT (c) 1999 BY                                       *
 *  COMPAQ COMPUTER CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                        *
 *                                                              *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                *
 *                                                              *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY
COMPAQ  COMPUTER   *
 *  CORPORATION.                                                *
 *                                                              *
 *  COMPAQ ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS   *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY COMPAQ.
*
 *                                                              *
 *                                                              *

**********************************************************************
*****-*/

#include <stdio.h>
#include <math.h>
#include <string.h>
#include <time.h>
#include <stdtypes.h>

#ifdef _WIN32
# include <nt_lib.h>
#else
# include <sys/param.h>                 /* to get MAXHOSTNAMELEN */
#endif

#include <common.h>

#include <tpccstruct.h>
#include <tpccapi.h>
#include <tpccerr.h>

#include <scr_util.h>

#include <msg.h>
#include <tx_api.h>
#include <tpcc_gen.h>

/* Constants used in NURand functions */
#define C_ID_CONST 511
#define ITEM_CONST 4093


/* Values used to generate random data */
#define PCT_C_LAST 60

/* Macro to do random number functions */
#define NURand(A, x, y, C) \
    ((((random_int(0, A) | random_int(x, y)) + C) % (y-x+1)) + x)

/* Values needed to make lastnames */
const char *c_table[10] = {"BAR", "OUGHT", "ABLE", "PRI", "PRES", "ESE",
                           "ANTI", "CALLY", "ATION", "EING"};
const int c_length[10] = {3,5,4,3,4,3,4,5,5,4};

void
get_wdid( txn_context_t *pContext, int *pw_id, int *pd_id )
{
 if( BACKEND_MODE == pContext->RunMode )
 {
   *pw_id = random_int( 1, pContext->max_w_id );
   *pd_id = random_int( 1, DISTRICTS_PER_WAREHOUSE );
 }
 else
 {
   *pw_id = pContext->w_id;
   *pd_id = pContext->d_id;
 }
}

void
get_no_data( txn_context_t *pContext, pNewOrderData pNewOrder,
                    int *context )
{
 int remote_txn = 0;
 int line;

 pNewOrder->CC = (CallersContext)pContext;
```

```
  *context = NEW_ORDER_TXN;

  get_wdid( pContext, &pNewOrder->w_id, &pNewOrder->ld_id );

  pNewOrder->d_id = random_int( 1, DISTRICTS_PER_WAREHOUSE );

  pNewOrder->c_id = NURand( 1023, 1, MAX_C_ID, C_ID_CONST );

  pNewOrder->o_ol_cnt = random_int( MIN_OL, MAX_OL );
  for( line = 0; line < pNewOrder->o_ol_cnt; line++ )
  {
   /* ol_supplying warehouse is remote 1% of the time */
   if(( random_int( 0, 999 ) < 990) || ( pContext->max_w_id == 1 ))
   {
    pNewOrder->o_ol[line].ol_supply_w_id = pNewOrder->w_id;
   }
   else
   {
    remote_txn++;
    pNewOrder->o_ol[line].ol_supply_w_id =
                random_int( 1, pContext->max_w_id-1 );
    if( pNewOrder->o_ol[line].ol_supply_w_id >= pNewOrder->w_id )
    {
                pNewOrder->o_ol[line].ol_supply_w_id++;
    }
   }

   /* set last pNewOrder->o_ol[line].ol_i_id to ILLEGAL if rollback required */
   if(( line == pNewOrder->o_ol_cnt-1 ) &&
      ( random_int( 0, 999 ) < 10 ))
   {
    pNewOrder->o_ol[line].ol_i_id = ILLEGAL_ITEM;
    *context |= ILLEGAL_ITEM_TXN;
   }
   else
   {
    pNewOrder->o_ol[line].ol_i_id =
                NURand( 8191, 1, MAX_I_ID_POPULATED, ITEM_CONST );
   }

   pNewOrder->o_ol[line].ol_quantity = random_int( 1, MAX_OL_QUANTITY );

  }          /* for loop */

  *context |= ( pNewOrder->o_ol_cnt << OL_CNT_BIT_POS );
  *context |= ( remote_txn << REMOTE_TXN_BIT_POS );

}                               /* get_no_data */

void
get_pt_data( txn_context_t *pContext, pPaymentData pPayment,
                int *context )
{
  char                          *ptr;
  int                           index, i1, i2, i3;

  pPayment->CC = (CallersContext)pContext;

  *context = PAYMENT_TXN;

  get_wdid( pContext, &pPayment->w_id, &pPayment->ld_id );

  pPayment->d_id = random_int( 1, DISTRICTS_PER_WAREHOUSE );

  pPayment->c_id = INVALID_C_ID;

  ptr = pPayment->c_last;
  if( random_int( 0,99 ) < PCT_C_LAST )
  {
   /* use C_LAST */
   *context |= NON_KEY_ACCESS;
   index = NURand( 255, 0, 999, pContext->CLast );
   i1 = index/100;                                      /* hundreds digit */
   i2 = ( index - i1 * 100 ) / 10;              /* tens digit */
   i3 = index - ( i1 * 100 ) - ( i2 * 10 );/* ones digit */

   memcpy( ptr, c_table[i1], c_length[i1] );
   ptr += c_length[i1];
   memcpy( ptr, c_table[i2], c_length[i2] );
   ptr += c_length[i2];
   memcpy( ptr, c_table[i3], c_length[i3] );
   ptr += c_length[i3];
   *ptr = '\0';
  }
  else
  {
   /* use c_id within [1..MAX_C_ID] */
   pPayment->c_id = NURand( 1023, 1, MAX_C_ID, C_ID_CONST );
  }

  /* 15% remote payment */
  if(( random_int( 0,999 ) < 850 ) || ( pContext->max_w_id == 1 ))
  {
   pPayment->c_w_id = pPayment->w_id;
   pPayment->c_d_id = pPayment->ld_id;
  }
  else
  {
   *context |= REMOTE_PAYMENT;
   pPayment->c_w_id = random_int( 1, pContext->max_w_id-1 );
   if( pPayment->c_w_id >= pPayment->w_id )
   {
    pPayment->c_w_id++;
   }
   pPayment->c_d_id = random_int( 1, DISTRICTS_PER_WAREHOUSE );
  }

  /* h_amount in the within [1.00..MAX_PT_AMOUNT_CENTS/100] */
  pPayment->h_amount = ((double)random_int( 100, MAX_PT_AMOUNT_CENTS )
/ 100.0);

}                               /* get_pt_data */

void
get_os_data( txn_context_t *pContext, pOrderStatusData pOrderStatus,
                int *context )
{
  char                          *ptr;
  int                           index, i1, i2, i3;

  pOrderStatus->CC = (CallersContext)pContext;

  *context = ORDER_STATUS_TXN;

  get_wdid( pContext, &pOrderStatus->w_id, &pOrderStatus->ld_id );

  pOrderStatus->d_id = random_int( 1, DISTRICTS_PER_WAREHOUSE );

  pOrderStatus->c_id = INVALID_C_ID;

  ptr = pOrderStatus->c_last;
  if( random_int( 0,99 ) < PCT_C_LAST )
  {
   /* use C_LAST */
   *context |= NON_KEY_ACCESS;
   index = NURand( 255, 0, 999, pContext->CLast );
   i1 = index/100;                                      /* hundreds digit */
   i2 = ( index - i1 * 100 ) / 10;              /* tens digit */
   i3 = index - ( i1 * 100 ) - ( i2 * 10 );/* ones digit */

   memcpy( ptr, c_table[i1], c_length[i1] );
   ptr += c_length[i1];
   memcpy( ptr, c_table[i2], c_length[i2] );
   ptr += c_length[i2];
   memcpy( ptr, c_table[i3], c_length[i3] );
   ptr += c_length[i3];
   *ptr = '\0';
  }
  else
  {
   /* use c_id within [1..MAX_C_ID] */
   pOrderStatus->c_id = NURand( 1023, 1, MAX_C_ID, C_ID_CONST );
  }

}                               /* get_os_data */

void
get_dy_data( txn_context_t *pContext, pDeliveryData pDelivery,
                int *context )
```

```
{
  pDelivery->CC = (CallersContext)pContext;

  *context = INT_DELIVERY_TXN;

  get_wdid( pContext, &pDelivery->w_id, &pDelivery->ld_id );

  /* generate o_carrier_id in the range [1..MAX_CARRIER_ID] */
  pDelivery->o_carrier_id = random_int( 1, MAX_CARRIER_ID );

  /* enter time of request */
  pDelivery->queue_time = time( NULL );

}                               /* get_dy_data */

void
get_sl_data( txn_context_t *pContext, pStockLevelData pStockLevel,
                int *context )
{
  pStockLevel->CC = (CallersContext)pContext;

  *context = STOCK_LEVEL_TXN;

  get_wdid( pContext, &pStockLevel->w_id, &pStockLevel->ld_id );

  /* generate threshold in the range
[MIN_SL_THRESHOLD..MAX_SL_THRESHOLD] */
  pStockLevel->threshold = random_int( MIN_SL_THRESHOLD,
MAX_SL_THRESHOLD );

}                               /* get_sl_data */

void
get_cp_data( txn_context_t *pContext, pCheckpointData pCheckpoint,
                int *context )
{
  pCheckpoint->CC = (CallersContext)pContext;

  *context = CHECKPOINT_TXN;

  get_wdid( pContext, &pCheckpoint->w_id, &pCheckpoint->ld_id );

  pCheckpoint->how_many = 1;
  pCheckpoint->interval = 0;

}                               /* get_cp_data */
```

tpcc_gen.h

```
#ifndef TPCC_GEN_H
#define TPCC_GEN_H
/*+**********************************************************************
*******
 *                                                                      *
 *  COPYRIGHT (c) 1999 BY                                               *
 *  COMPAQ COMPUTER CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                                *
 *                                                                      *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY  OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                        *
 *                                                                      *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY
COMPAQ  COMPUTER   *
 *  CORPORATION.                                                        *
 *                                                                      *
 *  COMPAQ ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS    *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY COMPAQ.
*
 *                                                                      *
 *                                                                      *

**********************************************************************
******-*/

void get_wdid( txn_context_t *pContext, int *pw_id, int *pd_id );
void get_no_data(txn_context_t *pUdata, pNewOrderData pNewOrder, int *context);
void get_pt_data(txn_context_t *pUdata, pPaymentData pPayment, int *context);
void get_os_data(txn_context_t *pUdata, pOrderStatusData pOrderStatus,
                        int *context);
void get_dy_data(txn_context_t *pUdata, pDeliveryData pDelivery, int *context);
void get_sl_data(txn_context_t *pUdata, pStockLevelData pStockLevel,
                        int *context);
void get_cp_data(txn_context_t *pUdata, pCheckpointData pCheckpoint,
                        int *context);

#endif
```

tpcc_master.c

```
/*+**********************************************************************
*******
 *                                                                      *
 *  COPYRIGHT (c) 1999 BY                                               *
 *  COMPAQ COMPUTER CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                                *
 *                                                                      *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY  OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                        *
 *                                                                      *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY
COMPAQ  COMPUTER   *
 *  CORPORATION.                                                        *
 *                                                                      *
 *  COMPAQ ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS    *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY COMPAQ.
*
 *                                                                      *
 *                                                                      *

**********************************************************************
******-*/
/*
 * Abstract: The Master script is the script responsible for timing when
 *          the users should be logged in, when they should rampup, how
 *          long their steady state is. etc...
 *
 * Author: ???
 *
 * Creation Date: ???
 *
 * Modification history:
 *
 * X-1 W. Carr 17-Jan-1997
 *    Added new format header.  Checked in code that prevents the audit
 *    scripts from running.  Added new code to pass counter reset message
 *    to all scripts as part of run-time data reduction code.
 *
 *
```

```
*
*/


#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <time.h>

#ifdef _WIN32
# include <direct.h>
# include <nt_lib.h>
# include <windows.h>
# include <process.h>
#else
# include <dirent.h>
# include <sys/time.h>
# include <unistd.h>
#endif

#include <stdtypes.h>
#include <prte.h>
#include <common.h>

#include <tpccstruct.h>
#include <tpccapi.h>
#include <tpccerr.h>
#include <reg.h>
#include <config.h>

#include <scr_util.h>

#define NEED_MSG_NAMES
#include <msg.h>
#include <netvar.h>
#include <audit_util.h>


#ifdef _WIN32
# define MKDIR(Dir,Mode) mkdir(Dir)
  typedef int mode_t;
#else
# define MKDIR(Dir,Mode) mkdir(Dir,Mode)
#endif

#define MAX_RUN_DIR_NAME_LEN 3 /* Number padded with leading 0's */
#define LOG_DIR "logs"     /* appended to the run dir */
#define maximum(A,B) ((A) > (B) ? (A) : (B))

#define NODELAY

#define REG_KEY_STR "CONF_FE_NT_REG_KEY_"
#define REG_VALUE_STR "CONF_FE_NT_REG_VALUE_"
#define REG_TYPE_STR "CONF_FE_NT_REG_TYPE_"
#define REG_DATA_STR "CONF_FE_NT_REG_DATA_"

typedef struct _be_t
{
  char          Name[MAXHOSTNAMELEN];
  double LogSize;
  double LogUsed;
} be_t;

/* This is the master data structure.  It contains all relevent information
   for coordinating the testing. */
typedef struct _master_data_t
{
  int32                 AdminUsers;
  int32                 RequestsSent;
  int32                 ReplysReceived;
  int32                 AidesStatus;
  BOOL                  MsgTimedOut;
  double        MsgTimeout;

  char                  CgiScriptName[PATH_MAX];
  BOOL                  DoCheckpoints;
  double        CkptStartOffset;

  /* FE Info */
  int32                         RunMode;
  int32                         TpccUsers;
  int32                         NumFEs;
  int32                         NumConnects;
  connect_t             *pConnects;
  BOOL                          FERestartViaReboot;
  int32                         RestartFEs;

  double        Proximity;
  pid_t                 OsPid;
  char                  RunDir[PATH_MAX];
  int                   CheckpointInterval;
  double        WarmUpTime;
  double        SteadyStateTime;
  double        MeasurementInterval;
  double        CoolDownTime;
  int                   ReducerID;
  int32                 ReducerStatus;
  char                  ReducerScriptName[PATH_MAX];
  char                  ReducerScriptNode[MAXHOSTNAMELEN];
  char                  OutputDir[PATH_MAX];
  char                  RunNumStr[8];
  int                   VersionNum;
  long                  MasterSeed;
  char                  *pVersion; /* Holds the System Version. */
  int32                 NumBE;
  be_t                  *pBEs;
  char                  AideScriptName[PATH_MAX];

  char                  BeUname[32];
  char                  DbUname[32];
  char                  DbPassword[32];
  char                  DbName[32];
  char                  OraStatScriptPath[PATH_MAX];
  char                  CustomBeforeTestScript[PATH_MAX];
  char                  CustomAfterTestScript[PATH_MAX];


  task_data_t   BeTasksData;
} master_data_t;


/* Prototypes */
void            BinMsgHandler( void *Context, int SendersID, int Len, void *Msg );
int32           Config( master_data_t *pContext );
int32           ConfigFEs( master_data_t *pContext );
int32           ConfigFEsSendMsg( master_data_t *pContext,
                                  config_fe_msg_t *pConfigFEMsg, char
*pDataVar );
int32           GetBeInfo(master_data_t *pContext);
void            GetFeUserCounts(master_data_t *pContext, int32 State);
void            GetSeed( master_data_t *pContext );
int32           Init( master_data_t *pContext );
int32           InitAides( master_data_t *pContext );
int32           InitFEs( master_data_t *pContext );
int32           InitReducer( master_data_t *pContext );
int32           MakeRunDir(char *DirName, char *RunDir);
int32           ParseUsersPerFE(master_data_t *pContext, char *string);
int32   PreTestHook(master_data_t *pContext);
int32           StartFEs( master_data_t *pContext );
int32           StopAides( master_data_t *pContext );
int32           StopFEs( master_data_t *pContext );
int32           StopReducer( master_data_t *pContext );
void            TellAidesTo(master_data_t *pContext, int32 Task, int32 State);
int32           TestPeriod( master_data_t *pContext );


void
main(int argc, char *argv[])
{
  master_data_t                         Context;
  master_data_t                         *pContext = &Context;
  int32                                 Status;

  /* Initialize with PRTE. */
  PRTEinit( argc, argv );
```

```
 /* Initialize. */
 if( SCR_E_SUCCESS != ( Status = Init( pContext )))
 {
  ( void ) SCRerr( Status );
  ( void ) SCRerr( SCR_E_INIT );
 }

 /* Select the proper routine for the run mode. */
 if( SCR_E_SUCCESS == Status )
 {
  switch( pContext->RunMode )
  {
  case END_TO_END_MODE:
  case BACKEND_MODE:
    Status = TestPeriod( pContext );
    break;
  case CONFIG_MODE:
    Status = Config( pContext );
    break;
  default:
    break;
  }
 }

 SCRlog( "Master: Blowing this popsicle stand." );

 PRTEexit( );
}



/*****************************************************************
*********/
/*                                                               */
/* Name: BinMsgHandler                                           */
/*                                                               */
/* Purpose: This is the message handler that gets registered with PRTE for    */
/*          processing binary messages sent to the Master.                    */
/*                                                               */
/* Inputs:                                                       */
/*        Context: Pointer to the context structure. This structure is set   */
/*                 up elsewhere, and registered with PRTE to be passed in.   */
/*        SendersID: User ID number of the user that sent us the binary      */
/*                   message.                                                */
/*        Len: Length of the binary message in bytes.                        */
/*        Msg: Pointer to the actual data being sent.                        */
/*                                                               */
/* Outputs: None.                                                */
/*                                                               */
/*                                                               */
/* Function Value: void                                          */
/*                                                               */
/* Notes: This funtion is called on our behalf by PRTE. In order for this     */
/*        to happen, we must be inside a call to some PRTE function. If we    */
/*        are running code outside a PRTE function, PRTE will not interrupt   */
/*        us (i.e. no signal is sent). It is only when we are inside a PRTE   */
/*        function call (e.g. PRTEdelay, etc.) that PRTE will check to see if */
/*        we have a message that needs processing.                           */
/*                                                               */
/*****************************************************************
*********/
void
BinMsgHandler( void *Context, int SendersID, int Len, void *Msg )
{
  int32                                 Index;
  checkpoint_status_msg_t       *pCheckpointStatusMsg;
  master_data_t                         *pContext;
  generic_msg_t                         *pMsg;
  generic_status_msg_t          *pStatusMsg;
  char                                  TmpStr[256];

  /* Effectively convert void * parameters to correct types. */
  pMsg = Msg;
  pContext = Context;

  /* Process messaged based on its type. */
  switch( pMsg->Type )
  {
   case DATA_PROCESSING_LOOP_OVER:
    PRTEresume( );
    break;

   case DATA_POST_PROCESS_REPLY_MSG:
    pStatusMsg = (generic_status_msg_t *) Msg;
    if( SCR_E_SUCCESS == pStatusMsg->Status )
    {
            SCRlog( "\tTest results successfully generated." );
    }
    else
    {
            SCRlog( "\tError generating test results ( %d ).",
                    pStatusMsg->Status );
    }
    PRTEresume( );

    break;

   case AIDELOG_POST_PROCESS_REPLY_MSG:
    pStatusMsg = (generic_status_msg_t *) Msg;
    if( ERR_SUCCESS == pStatusMsg->Status )
    {
            SCRlog( "\tAide logs successfully processed." );
    }
    else
    {
            SCRlog( "\tError processing aide logs( %d ).",
                    pStatusMsg->Status );
    }
    PRTEresume_users( User_id, User_id );

    break;

   case AIDE_EXIT_REPLY_MSG:
    pStatusMsg = (generic_status_msg_t *) Msg;
    pContext->ReplysReceived++;
    if( SCR_E_SUCCESS != pStatusMsg->Status &&
             SCR_E_SUCCESS == pContext->AidesStatus )
    {
            /* Note that we've had a failure. */
            pContext->AidesStatus = pStatusMsg->Status;
    }
    if( pContext->ReplysReceived == pContext->RequestsSent )
    {
            /* We've heard back from everyone, so break ourselves out of delay. */
            pContext->MsgTimedOut = FALSE;
            PRTEdelay( 0.0 );
    }
    break;

   case AIDE_INIT_REPLY_MSG:
    pStatusMsg = (generic_status_msg_t *) Msg;
    if( SCR_E_SUCCESS != pStatusMsg->Status &&
             SCR_E_SUCCESS == pContext->AidesStatus )
    {
            /* Note that we've had a failure. */
            pContext->AidesStatus = pStatusMsg->Status;
    }
    pContext->ReplysReceived++;
    if( pContext->ReplysReceived == pContext->RequestsSent )
    {
            /* We've heard back from everyone, so break ourselves out of delay. */
            pContext->MsgTimedOut = FALSE;
            PRTEdelay( 0.0 );
    }
    break;

   case CHECKPOINT_CONNECT_REPLY_MSG:
    pCheckpointStatusMsg = (checkpoint_status_msg_t *) Msg;
    pContext->ReplysReceived++;

    if( SCR_E_SUCCESS == pCheckpointStatusMsg->Data.Status )
    {
            SCRlog( "\t\t%s - checkpoint sub-system initialized.",
                    pCheckpointStatusMsg->Data.FEName );
    }
    else
    {
            if( SCR_E_SUCCESS == pContext->AidesStatus )
            {
```

```
            pContext->AidesStatus = pCheckpointStatusMsg->Data.Status;
          }
          SCRlog( "\t\t%s - checkpoint subsystem initialization FAILED ( %d ).",
                  pCheckpointStatusMsg->Data.FEName,
                  pCheckpointStatusMsg->Data.Status );
  }

  if( pContext->ReplysReceived == pContext->RequestsSent )
  {
          /* We've heard back from everyone, so break ourselves out of delay. */
          pContext->MsgTimedOut = FALSE;
          PRTEdelay( 0.0 );
  }

  break;

case CHECKPOINT_RUN_REPLY_MSG:
  pCheckpointStatusMsg = (checkpoint_status_msg_t *) Msg;
  pContext->ReplysReceived++;

  if( SCR_E_SUCCESS == pCheckpointStatusMsg->Data.Status )
  {
          SCRlog( "\t\t%s - checkpoint loop started.",
                  pCheckpointStatusMsg->Data.FEName );
  }
  else
  {
          if( SCR_E_SUCCESS == pContext->AidesStatus )
          {
            pContext->AidesStatus = pCheckpointStatusMsg->Data.Status;
          }
          SCRlog( "\t\t%s - checkpoint loop NOT entered ( %d ).",
                  pCheckpointStatusMsg->Data.FEName,
                  pCheckpointStatusMsg->Data.Status );
  }

  if( pContext->ReplysReceived == pContext->RequestsSent )
  {
          /* We've heard back from everyone, so break ourselves out of delay. */
          pContext->MsgTimedOut = FALSE;
          PRTEdelay( 0.0 );
  }

  break;

case CHECKPOINT_STOP_REPLY_MSG:
  pCheckpointStatusMsg = (checkpoint_status_msg_t *) Msg;
  pContext->ReplysReceived++;

  if( SCR_E_SUCCESS == pCheckpointStatusMsg->Data.Status )
  {
          SCRlog( "\t\t%s - checkpoint loop ran successfully.",
                  pCheckpointStatusMsg->Data.FEName );
  }
  else
  {
          if( SCR_E_SUCCESS == pContext->AidesStatus )
          {
            pContext->AidesStatus = pCheckpointStatusMsg->Data.Status;
          }
          SCRlog( "\t\t%s - checkpoint loop FAILED ( %d ).",
                  pCheckpointStatusMsg->Data.FEName,
                  pCheckpointStatusMsg->Data.Status );
  }

  if( pContext->ReplysReceived == pContext->RequestsSent )
  {
          /* We've heard back from everyone, so break ourselves out of delay. */
          pContext->MsgTimedOut = FALSE;
          PRTEdelay( 0.0 );
  }

  break;

case CHECKPOINT_DISCONNECT_REPLY_MSG:
  pCheckpointStatusMsg = (checkpoint_status_msg_t *) Msg;
  pContext->ReplysReceived++;

  if( SCR_E_SUCCESS == pCheckpointStatusMsg->Data.Status )
  {
          SCRlog( "\t\t%s - checkpoint sub-system shutdown.",
                  pCheckpointStatusMsg->Data.FEName );
  }
  else
  {
          if( SCR_E_SUCCESS == pContext->AidesStatus )
          {
            pContext->AidesStatus = pCheckpointStatusMsg->Data.Status;
          }
          SCRlog( "\t\t%s - checkpoint subsystem shutdown FAILED ( %d ).",
                  pCheckpointStatusMsg->Data.FEName,
                  pCheckpointStatusMsg->Data.Status );
  }

  if( pContext->ReplysReceived == pContext->RequestsSent )
  {
          /* We've heard back from everyone, so break ourselves out of delay. */
          pContext->MsgTimedOut = FALSE;
          PRTEdelay( 0.0 );
  }

  break;

case CONFIG_FE_REPLY_MSG:
  pStatusMsg = (generic_status_msg_t *) Msg;
  pContext->ReplysReceived++;
  if( SCR_E_VALUE_MODIFIED == pStatusMsg->Status )
  {
          for( Index = 0; Index < pContext->NumConnects; Index++ )
          {
            if( SendersID == pContext->pConnects[Index].AideID )
            {
              pContext->pConnects[Index].ConfMod = TRUE;
              break;
            }
          }
  }
  else if( SCR_E_SUCCESS != pStatusMsg->Status &&
           SCR_E_SUCCESS == pContext->AidesStatus )
  {
          /* Note that we've had a failure. */
          pContext->AidesStatus = pStatusMsg->Status;
  }
  if( pContext->ReplysReceived == pContext->RequestsSent )
  {
          /* We've heard back from everyone, so break ourselves out of delay. */
          pContext->MsgTimedOut = FALSE;
          PRTEdelay( 0.0 );
  }
  break;

case ORA_SWITCHLOG_REPLY_MSG:
case GET_LOGSIZE_REPLY_MSG:
case ORA_DOSTATS_REPLY_MSG:
case DO_CUSTOM_SCRIPT_REPLY_MSG:
case INIT_AUDIT_TABLE_REPLY_MSG:
case BE_CKPTDATA_REPLY_MSG:
  pStatusMsg = (generic_status_msg_t *) Msg;
  if( SCR_E_SUCCESS != pStatusMsg->Status &&
      SCR_E_SUCCESS == pContext->AidesStatus )
  {
          /* Note that we've had a failure. */
          pContext->AidesStatus = pStatusMsg->Status;
  }
  pContext->ReplysReceived++;
  if( pContext->ReplysReceived == pContext->RequestsSent )
  {
          /* We've heard back from everyone, so break ourselves out of delay. */
          pContext->MsgTimedOut = FALSE;
          PRTEdelay( 0.0 );
  }
  break;

case START_FE_REPLY_MSG:
  pStatusMsg = (generic_status_msg_t *) Msg;
  pContext->ReplysReceived++;
  if( SCR_E_SUCCESS != pStatusMsg->Status &&
      SCR_E_SUCCESS == pContext->AidesStatus )
  {
          /* Note that we've had a failure. */
```

```
            pContext->AidesStatus = pStatusMsg->Status;
    }
    if( pContext->ReplysReceived == pContext->RequestsSent )
    {
            /* We've heard back from everyone, so break ourselves out of delay. */
            pContext->MsgTimedOut = FALSE;
            PRTEdelay( 0.0 );
    }
    break;

   case STOP_FE_REPLY_MSG:
    pStatusMsg = (generic_status_msg_t *) Msg;
    pContext->ReplysReceived++;
    if( SCR_E_SUCCESS != pStatusMsg->Status &&
          SCR_E_SUCCESS == pContext->AidesStatus )
    {
            /* Note that we've had a failure. */
            pContext->AidesStatus = pStatusMsg->Status;
    }
    if( pContext->ReplysReceived == pContext->RequestsSent )
    {
            /* We've heard back from everyone, so break ourselves out of delay. */
            pContext->MsgTimedOut = FALSE;
            PRTEdelay( 0.0 );
    }
    break;

   case REDUCER_INIT_REPLY_MSG:
    pStatusMsg = (generic_status_msg_t *) Msg;
    pContext->ReducerStatus = pStatusMsg->Status;
    pContext->MsgTimedOut = FALSE;
    PRTEdelay( 0.0 );
    break;

   case USER_LOGIN_REPLY_MSG:
   case USER_START_REPLY_MSG:
   case USER_STOP_REPLY_MSG:
    pStatusMsg = (generic_status_msg_t *) Msg;
    pContext->ReducerStatus = pStatusMsg->Status;
    PRTEresume( );
    break;

   case LOAD_FE_REPLY_MSG:
    pStatusMsg = (generic_status_msg_t *) Msg;
    pContext->ReplysReceived++;
    if( SCR_E_SUCCESS != pStatusMsg->Status &&
          SCR_E_SUCCESS == pContext->AidesStatus )
    {
            /* Note that we've had a failure. */
            pContext->AidesStatus = pStatusMsg->Status;
    }
    if( pContext->ReplysReceived == pContext->RequestsSent )
    {
            /* We've heard back from everyone, so break ourselves out of delay. */
            pContext->MsgTimedOut = FALSE;
            PRTEdelay( 0.0 );
    }
    break;

   default:
    sprintf( TmpStr, "Master received unknown user message type: %d",
              pMsg->Type );
    SCRlog( TmpStr );
  }
}



int32
Config( master_data_t *pContext )
{
  char                  *pTmpStr;
  int32                 Status;
  char                  TmpStr[256];

  Status = SCR_E_SUCCESS;

  SCRlog( "\n\n\n\t\t\t\t\t..START OF CONFIG..\n\n" );

  /* Lie about a reducer existing for the aides will initialize. */
  ITOA( 0, TmpStr, pTmpStr );
  PRTEset_network_variable( "REDUCER_ID", TmpStr );

  if( SCR_E_SUCCESS == Status )
  {
   Status = InitAides( pContext );
  }

  if( SCR_E_SUCCESS == Status )
  {
   Status = ConfigFEs( pContext );
  }

  if( SCR_E_SUCCESS == Status )
  {
   Status = StopAides( pContext );
  }

  SCRlog( "\n\n\n\n\t\t\t\t\t..END OF CONFIG..\n\n\n" );

  return( Status );
}


/*****************************************************************************
*********/
/*                                                                           */
/* Name: ConfigFEs                                                           */
/*                                                                           */
/* Purpose: This function causes the FEs to be configed.                     */
/*                                                                           */
/* Inputs:                                                                   */
/*                                                                           */
/* Outputs:                                                                  */
/*                                                                           */
/* Function Value:                                                           */
/*                                                                           */
/* Notes:                                                                    */
/*                                                                           */
/*****************************************************************************
*********/
int32
ConfigFEs( master_data_t *pContext )
{
  int                           Code;
  int32                         Index;
  char                          NetVarName[50];
  int32                         NetVarNum;
  char                          *pTmp;
  char                          *pValue;
  config_fe_msg_t       ConfigFEMsg;
  int32                         Status;

  /* Check if we have any FEs to configure. */
  if( 0 == pContext->NumFEs )
  {
   return( SCR_E_SUCCESS );
  }

  /* Initialize stuff. */
  for( Index = 0; Index < pContext->NumConnects; Index++ )
  {
   if( ! pContext->pConnects[Index].ConfConn )
   {
    continue;
   }
   pContext->pConnects[Index].ConfMod = FALSE;
   SCRlog( "\tChecking configuration of %s.",
                pContext->pConnects[Index].FEName );
  }

  /* All the known FE parameters go into the same key, get them by name. */
  strcpy( ConfigFEMsg.Data.Key, TPCC_CLASS );
  strcpy( ConfigFEMsg.Data.Type, "string" );

  for( Code = FFE_C_BASE + 1; Code <= FFE_C_MAX_PARAM; Code++ )
  {
   /* Get the configuration value name in order to get its data */
   pValue = ConfigCodeToStr( Code );
   strcpy( ConfigFEMsg.Data.Value, pValue );
```

```
  /* Now collect data and send messages. */
  Status = ConfigFEsSendMsg( pContext, &ConfigFEMsg, pValue );
  if( SCR_E_SUCCESS != Status )
  {
    return( Status );
  }
}

/* In order to set a registry entry, you must have the key, value, type */
/* and data. This code assumes that these pieces of info are grouped    */
/* in sets of predetermined network variables. So we'll keep setting    */
/* entries as long as we keep finding the next set.                     */
NetVarNum = 1;
sprintf( NetVarName, "%s%d", REG_KEY_STR, NetVarNum );
while( NULL != (pTmp = PRTEget_network_variable( NetVarName )))
{
  /* We got a key back, so the rest of this set should also be there. */
  /* So save the key, and get the rest of the set.                    */
  strcpy( ConfigFEMsg.Data.Key, pTmp );

  /* Get the value. */
  sprintf( NetVarName, "%s%d", REG_VALUE_STR, NetVarNum );
  if( NULL == (pTmp = PRTEget_network_variable( NetVarName )))
  {
    /* We have an error here. */
    SCRlog( "Getting network variable: %s", ConfigFEMsg.Data.Value );
    return( SCRerr( SCR_E_NO_VALUE ));
  }
  else
  {
    strcpy( ConfigFEMsg.Data.Value, pTmp );
  }

  /* Get the type. */
  sprintf( NetVarName, "%s%d", REG_TYPE_STR, NetVarNum );
  if( NULL == (pTmp = PRTEget_network_variable( NetVarName )))
  {
    /* We have an error here. */
    SCRlog( "Getting network variable: %s", ConfigFEMsg.Data.Type );
    return( SCRerr( SCR_E_NO_TYPE ));
  }
  else
  {
    strcpy( ConfigFEMsg.Data.Type, pTmp );
  }

  /* Now collect data and send messages. */
  sprintf( NetVarName, "%s%d", REG_DATA_STR, NetVarNum );
  Status = ConfigFEsSendMsg( pContext, &ConfigFEMsg, NetVarName );
  if( SCR_E_SUCCESS != Status )
  {
    return( Status );
  }

  /* Move on to the next entry set. */
  NetVarNum++;
  sprintf( NetVarName, "%s%d", REG_KEY_STR, NetVarNum );
} /* while */

/* Check if any of the FEs need to be restarted. */
Status = SCR_E_SUCCESS;
for( Index = 0; Index < pContext->NumConnects; Index++ )
{
  if( ! pContext->pConnects[Index].ConfConn )
  {
    continue;
  }
  if( pContext->pConnects[Index].ConfMod )
  {
    Status = SCR_E_VALUE_MODIFIED;
    SCRlog( "\tSuccessfully modified configuration of %s.",
                pContext->pConnects[Index].FEName );
  }
  else
  {
    SCRlog( "\tSuccessfully checked configuration of %s.",
                pContext->pConnects[Index].FEName );
  }
}

if( SCR_E_VALUE_MODIFIED != Status && SCR_E_SUCCESS != Status )
{
  SCRlog( "\tFailed to configure the FEs. Pausing.\n" );
  PRTEpause();
}

/* Check if FEs should be restarted. */
if(( FE_RESTART_ALWAYS == pContext->RestartFEs ) ||
   ( FE_RESTART_MODIFIED == pContext->RestartFEs &&
     SCR_E_VALUE_MODIFIED == Status ))
{
  if( SCR_E_SUCCESS != ( Status = StopFEs( pContext )))
  {
    SCRlog( "\tFailed to stop the FEs. Pausing.\n" );
    PRTEpause();
  }
  if( SCR_E_SUCCESS != ( Status = StartFEs( pContext )))
  {
    SCRlog( "\tFailed to start the FEs. Pausing.\n" );
    PRTEpause();
  }
}
else
{
  /* Clear the potential MODIFIED status. */
  Status = SCR_E_SUCCESS;
}

return( Status );
}


int32
ConfigFEsSendMsg( master_data_t *pContext, config_fe_msg_t *pConfigFEMsg,
                  char *pDataVar )
{
  int                   Count;
  int32                 Index;
  char                  *pTmp;
  int32                 Status;

  Status = SCR_E_SUCCESS;
  if( NULL == (pTmp = PRTEget_network_variable( pDataVar )))
  {
    /* We have an error here. */
    SCRlog( "Getting network variable: %s", pDataVar );
    return( SCRerr( SCR_E_NO_DATA ));
  }

  /* Check for correct number of data values. */
  Count = SCR_count_elements( pTmp );
  if( 1 != Count && pContext->NumFEs != Count )
  {
    SCRlog( "Parsing: %s %s", pDataVar, pTmp );
    return( SCRerr( SCR_E_WRONG_DATA_ITEM_COUNT ));
  }

  /* send  reg_msg_t to each aide */
  pContext->RequestsSent = 0;
  pContext->ReplysReceived = 0;
  pContext->AidesStatus = SCR_E_SUCCESS;
  pContext->MsgTimedOut = TRUE;
  pConfigFEMsg->Type = CONFIG_FE_REQUEST_MSG;
  for( Index = 0; Index < pContext->NumConnects; Index++ )
  {
    if( ! pContext->pConnects[Index].ConfConn )
    {
      continue;
    }
    if( 1 == Count )
    {
      /* Always get the first element.  Use the element routine to  */
      /* eliminate any quoting problems.
                 */
      (void)SCR_next_list_element( pTmp, pConfigFEMsg->Data.Data );
    }
    else
    {
      /* Get the next element. */
```

```
    pTmp = SCR_next_list_element( pTmp, pConfigFEMsg->Data.Data );
   }
   strcpy( pConfigFEMsg->Data.FEName, pContext->pConnects[Index].FEName );
   PRTEmessage_to_user_binary( sizeof( *pConfigFEMsg ), pConfigFEMsg,
                                                    pContext-
>pConnects[Index].AideID,
                                                    pContext-
>pConnects[Index].AideID );
   pContext->RequestsSent++;
  }

  PRTEdelay( pContext->MsgTimeout );

  if( pContext->MsgTimedOut )
  {
   Status = SCRerr( SCR_E_AIDE_TIMEOUT_CONFIG_FE );
   SCRlog( "\tPausing...\n" );
   PRTEpause( );
  }

  if( SCR_E_SUCCESS != pContext->AidesStatus )
  {
   Status = SCRerr( pContext->AidesStatus );
   Status = SCRerr( SCR_E_AIDES_FAILED_CONFIG );
  }

  return( Status );
}


int32
GetBeInfo( master_data_t *pContext )
{
  char        *List;
  char        *pTmpChar;
  char        ElementString[32];
  int32       BeIndex;

  /* Get the number of FEs we're dealing with, so we can allocate structures.*/
  if( NULL == ( List = PRTEget_network_variable( "BE_NAMES" )))
  {
   return( SCRerr( SCR_E_BE_NAMES ));
  }

  pContext->NumBE = SCR_count_elements(List);

  /* Now we now how many BEs there are, so we can allocate space. */
  if( NULL == ( pContext->pBEs =
                        ( be_t * ) malloc( pContext->NumBE * sizeof( be_t ))))
  {
   return( SCRerr( SCR_E_MALLOC_BE_ARRAY ));
  }

  SCRlog( "\tThe following BEs will be used." );
  pTmpChar = List;
  BeIndex = 0;
  while( pTmpChar = SCR_next_list_element( pTmpChar, ElementString ))
  {
   strcpy( pContext->pBEs[BeIndex].Name, ElementString );
   SCRlog( "\t\t%s", pContext->pBEs[BeIndex].Name );
   BeIndex++;
  }

  return( SCR_E_SUCCESS );
}


void
GetSeed(master_data_t *pContext)
{

  char temp_string[128];
  struct timeval current_time;
  FILE   *SeedFile;
  char   SeedFileName[256];  /* some max file name size. dj. */

  /* Get the seed value. */
  SCR_getnet_long("SEED",SEED_DEFAULT,&pContext->MasterSeed);
  if (0 == pContext->MasterSeed)
  {
   gettimeofday( &current_time, NULL);
   pContext->MasterSeed = pContext->OsPid * current_time.tv_sec;
  }

  /* Announce seed value to PRTE community. */
  sprintf(temp_string,"%ld",pContext->MasterSeed);
  PRTEset_network_variable("SEED",temp_string);

  /* Open audit file to record seed. */
  sprintf(SeedFileName, "%sm%sseed.v%d", pContext->RunDir,
                pContext->RunNumStr, pContext->VersionNum);
  if (NULL == (SeedFile = fopen(SeedFileName,"w")))
  {
   SCRlog( "Master: Failed to open seed file." );
  }
  else
  {
   /* Write seed out to audit file in the run directory. */
   fprintf(SeedFile, "%s", temp_string);
   fclose(SeedFile);
  }

  /* Log seed to console and private log file. */
  SCRlog( "\tSeed = %ld\n", pContext->MasterSeed );
}



/*************************************************************
 * Init()                                                    *
 *                                                           *
 * Inputs:  None                                             *
 *                                                           *
 * Output:  None                                             *
 *                                                           *
 * Written by:  Tareef Kawaf  (Feb. 1995)                    *
 *                                                           *
 * Dependencies: None.                                       *
 *                                                           *
 * Description:                                              *
 *  Init() will retrieve the value of all                    *
 * network variables that would affect the execution of this  *
 * test, as seen by the master.                              *
 *                                                           *
 *                                                           *
 *************************************************************/


int32
Init( master_data_t *pContext )
{
  int32                 CLast;
  double          CkptProximAddOffset;
  double          CkptsPerInterval;
  FILE                  *Fd;
  char                  FileName[PATH_MAX];
  char                  IDString[7];
  int                   Index;
  int32                 RunId;
  int32                 Status;
  struct timeval        TimeStamp;
  char                  TmpStr[256]; /* arbitrary length */


  /* Announce that we're initializing. */
  SCRlog( "Master: Initializing.\n" );

  /* Initialize context to zero. */
  memset( pContext, 0, sizeof( *pContext ));

  /* Set the software version for this build.  NOTE we should confirm this */
  /* is the same version as PRTE!                                          */
  pContext->pVersion = VERSION; /* Get System Version from compile time. */
  SCRlog( "\tSoftware Version = %s\n", pContext->pVersion );
  PRTEset_network_variable("TPCC_SCRIPTS_VERSION", pContext->pVersion);

  /* Get the operating system process id. */
```

```
pContext->OsPid = getpid();
SCRlog( "\tOS PID = %d", pContext->OsPid );

/* Announce the PRTE user id. */
SCRlog( "\tMaster PRTE User ID = %d\n", User_id );

/* Make sure we flush stuff as we write it out. */
Flush_log = TRUE;

/* /Get the test results directory, and create a run directory under it. */
SCR_getnet_string( "TEST_RESULTS_DIR", TEST_RESULTS_DIR_DEFAULT,
                   pContext->OutputDir );

/* Make the run directory, log directory and announce them to the PRTE */
/* community.                                                          */
if( -1 == ( RunId = MakeRunDir( pContext->OutputDir, pContext->RunDir )))
{
  return( SCRerr( SCR_E_MAKING_RUN_DIR ));
}

sprintf( TmpStr, "%d", RunId );
PRTEset_network_variable( "RUN_ID", TmpStr );

/* We now have enough to build up the log file name and path. This means */
/* master can now play in the asci world.                                */
strcpy( Log_path, pContext->RunDir );
strcpy( Log_name, "master.log" );

/* Master doesn't connect to any SUTs, so all we only need to enable  */
/* PRTEto_log calls.
                                                                      */
Logging_type = ( SCRIPT_SPECIFIC_LOGGING );

SCRlog( "\tLogFile = %s%s\n", Log_path, Log_name );

/* Register binary message handler with PRTE. */
PRTEcatch_message_binary( BinMsgHandler, pContext );

/* Master is now ready to play in the binary message world, so go ahead and */
/* announce his user id to the PRTE community.                              */
sprintf( IDString, "%d", User_id );
PRTEset_network_variable( "MASTER_ID", IDString );

/* Get the test run number. */
SCR_getnet_string( "RUN_NUMBER", RUN_NUMBER_DEFAULT,
                   pContext->RunNumStr );
SCRlog( "\tRunNumber = %s.", pContext->RunNumStr );

/* Get the test version number. */
SCR_getnet_int( "VERSION_NUMBER", VERSION_NUMBER_DEFAULT,
                &pContext->VersionNum );
SCRlog( "\tVersionNumber = %d\n", pContext->VersionNum );

/* Get timing lengths. */
SCR_getnet_double( "WARMUP_TIME", WARMUP_TIME_DEFAULT,
                   &pContext->WarmUpTime );
if( 0 < pContext->WarmUpTime )
{
  SCRlog( "\tWarmup = %7.2f", pContext->WarmUpTime );
}

SCR_getnet_double( "STEADY_STATE_TIME",
STEADY_STATE_TIME_DEFAULT,
                   &pContext->SteadyStateTime );
if( 0 < pContext->SteadyStateTime )
{
  SCRlog( "\tSteady State Time = %7.2f", pContext->SteadyStateTime );
}

SCR_getnet_double( "MEASUREMENT_INTERVAL",
MEASUREMENT_INTERVAL_DEFAULT,
                   &pContext->MeasurementInterval );
if( 0 < pContext->MeasurementInterval )
{
  SCRlog( "\tMeasurement Interval= %7.2f", pContext->MeasurementInterval );
}

SCR_getnet_double( "COOLDOWN_TIME", COOLDOWN_TIME_DEFAULT,
                   &pContext->CoolDownTime );
if( 0 < pContext->CoolDownTime )
{
  SCRlog( "\tCooldown = %7.2f\n", pContext->CoolDownTime );
}

/* Get the message timeout value. */
SCR_getnet_double( "MSG_TIMEOUT", MSG_TIMEOUT_DEFAULT,
                   &( pContext->MsgTimeout ));
SCRlog( "\tMessage Timeout = %5.2f\n", pContext->MsgTimeout );

/* Set the seed. Requires RunDir, RunNumStr, RunVer. */
GetSeed( pContext );

/* Check out the value for C_LAST.  Users are the ones who need this, */
/* but we don't want all of them printing it's value out, so master   */
/* does it. Also, master logs it to a file in the run directory.      */
SCR_getnet_int( "C_LAST", C_LAST_DEFAULT, &CLast );
SCRlog( "\tC_LAST = %d\n", CLast );
sprintf( FileName, "%srte%sclast.v%d", pContext->RunDir,
         pContext->RunNumStr, pContext->VersionNum );
if( NULL == (Fd = fopen( FileName, "w" )))
{
  ( void ) SCRerr( SCR_E_OPEN_C_LAST_FILE );
}
else
{
  gettimeofday( &TimeStamp, NULL );
  fprintf( Fd, "Date: %s\n\n", ctime( &TimeStamp.tv_sec ));
  fprintf( Fd, "C_LAST constant = %d.\n", CLast );
  fclose( Fd );
}

/* Get RTE info */
Status = GetConnectInfo( pContext->RunMode, &pContext->TpccUsers,
                         &pContext->NumFEs, &pContext-
>AdminUsers,
                         &pContext->NumConnects, &pContext-
>pConnects );
if( SCR_E_SUCCESS != Status )
{
  SCRlog( "Error ( %d ) getting CONNECT information.", Status );
  return( Status );
}

/* Get the run mode. */
SCR_getnet_int( "RUN_MODE", RUN_MODE_DEFAULT,
                &pContext->RunMode );
switch( pContext->RunMode )
{
  case END_TO_END_MODE:
    SCRlog( "\tRunning in end-to-end mode.\n" );
    break;
  case BACKEND_MODE:
    SCRlog( "\tRunning in rte based backend run (C') mode.\n" );
    break;
  case CONFIG_MODE:
    SCRlog( "\tRunning in config mode.\n" );
    break;
  default:
    SCRlog( "RunMode = %d", pContext->RunMode );
    return( SCRerr( SCR_E_INVALID_RUN_MODE ));
    break;
}

SCRlog( "\tConnects." );
#define CONNECT_TABLE_HEADER_FORMAT "\t  %-12s %-12s %-5s %-5s %5s
%s"
#define CONNECT_TABLE_ENTRY_FORMAT  "\t  %-12s %-12s %-5s %-5s %5d
%s"
SCRlog( CONNECT_TABLE_HEADER_FORMAT,
        "RTE", "FE", "Conf.", "Admin", "Users", "Script");
for( Index = 0; Index < pContext->NumConnects; Index++ )
{
  SCRlog( CONNECT_TABLE_ENTRY_FORMAT,
          (pContext->pConnects)[Index].RTEName,
          (pContext->pConnects)[Index].FEName,
          (pContext->pConnects)[Index].ConfConn ? "  T" : "  F",
          (pContext->pConnects)[Index].AdminConn ? "  T" : "  F",
          (pContext->pConnects)[Index].Ucnt,
          (pContext->pConnects)[Index].Script );
}
```

```
  SCRlog( "\tTotal TPC-C Users = %d.\n", pContext->TpccUsers );
  SCRlog( "\tTotal FE Count = %d.\n", pContext->NumFEs );
  SCRlog( "\tTotal Admin Users = %d.\n", pContext->AdminUsers );

  /* How to handle FEs after reconfig. */
  SCR_getnet_bool( "FE_RESTART_VIA_REBOOT",
FE_RESTART_VIA_REBOOT_DEFAULT,
                        &pContext->FERestartViaReboot );
  SCR_getnet_int( "FE_RESTART_MODE", FE_RESTART_MODE_DEFAULT,
                        &pContext->RestartFEs );

  /* Get support user info. */
  SCR_getnet_string( "ADMIN_SCRIPT_NAME",
ADMIN_SCRIPT_NAME_DEFAULT,
                        pContext->AideScriptName );
  SCRlog( "\tAide Script Name = %s.", pContext->AideScriptName );

  SCR_getnet_string( "REDUCER_SCRIPT_NAME",
REDUCER_SCRIPT_NAME_DEFAULT,
                        pContext->ReducerScriptName );
  SCRlog( "\tReducer Script Name = %s.", pContext->ReducerScriptName );
  SCR_getnet_string( "REDUCER_SCRIPT_NODE",
REDUCER_SCRIPT_NODE_DEFAULT,
                        pContext->ReducerScriptNode );
  SCRlog( "\tReducer Script Node = %s.", pContext->ReducerScriptNode );

  /* Get the user dll name. We don't print out the name of the dll here */
  /* because the reducer will print out all the dll info.                */
  SCR_getnet_string( "CGI_SCRIPT_NAME", CGI_SCRIPT_NAME_DEFAULT,
                        pContext->CgiScriptName );

  if( 0 < pContext->AdminUsers )
  {
    /* Get the checkpoint settings. */
    SCR_getnet_int( "CHECKPOINT_INTERVAL",
CHECKPOINT_INTERVAL_DEFAULT,
                        &pContext->CheckpointInterval );

    SCRlog( "\tCheckpoint Interval = %d", pContext->CheckpointInterval );


    if( 0 < pContext->CheckpointInterval )
    {
      SCR_getnet_double( "CKPT_PROXIMITY_ADDITIONAL_OFFSET",

CKPT_PROXIMITY_ADDITIONAL_OFFSET_DEFAULT,
                              &CkptProximAddOffset );

      /* Verify that the checkpoints will be legal with regard to the */
      /* benchmark specification (v3.3.2 Sec. 5.5.2.2).               */

      /* Checkpoint interval must be less than or equal to 1800 seconds. */
      if( pContext->CheckpointInterval > 1800 )
      {
            return( SCRerr( SCR_E_CKPT_INT_TOO_LONG ));
      }

      /* Checkpoint interval must be less than or equal to measurement int. */
      if( pContext->CheckpointInterval > pContext->MeasurementInterval )
      {
            return( SCRerr(
SCR_E_CKPT_INT_GREATER_THAN_MEASUREMENT_INT ));
      }

      /* If checkpoint is less than measurement interval, measurement */
      /* interval must be integral multiple of checkpoint interval.   */
      if( (pContext->CheckpointInterval < pContext->MeasurementInterval) &&
            ( (int32) pContext->MeasurementInterval %
              pContext->CheckpointInterval != 0 ))
      {
            return( SCRerr(
SCR_E_MEASUREMENT_NOT_INTEGRAL_MULTIPLE ));
      }

      /* If there are fewer than 4 checkpoints in the measurement interval */
      /* then guard zones must be observed.                                 */
      CkptsPerInterval = ceil( pContext->MeasurementInterval /
                              pContext->CheckpointInterval );
      if( 4 > CkptsPerInterval )
      {
            pContext->Proximity = (pContext->MeasurementInterval *
                              (2 / (CkptsPerInterval+3))) / 2;
      }

      /* Now that we know the requested values are legal by the spec. make */
      /* sure the values are legal for this implementation.  Specifically, */
      /* make sure the warm up period is long enough to support the        */
      /* requested checkpoint interval.                                    */
      if( 0 >= (pContext->CkptStartOffset = pContext->WarmUpTime +
                        pContext->Proximity + CkptProximAddOffset -
                        pContext->CheckpointInterval))
      {
            return( SCRerr( SCR_E_WARMUP_TOO_SHORT ));
      }

      /* Since we've made it through all our checks, we can do checkpoints. */
      pContext->DoCheckpoints = TRUE;
      if( (0 < pContext->Proximity) && (0 < CkptProximAddOffset) )
      {
            SCRlog( "\tProximity = %7.2f",
                        pContext->Proximity );
            SCRlog( "\tAdditional Proximity Offset = %7.2f",
                        CkptProximAddOffset );
            SCRlog( "\tTotal Checkpoint Offset = %7.2f\n",
                        pContext->Proximity + CkptProximAddOffset );
      }
      else if( 0 < pContext->Proximity )
      {
            SCRlog( "\tProximity = %7.2f\n",
                        pContext->Proximity );
      }
      else if( 0 < CkptProximAddOffset )
      {
            SCRlog( "\tCheckpoint Offset = %7.2f\n",
                        CkptProximAddOffset );
      }
    }
    else
    {
      SCRlog( "Master: Checkpoints will not be done." );
    }
  }

  /* If we're not going all the way to a real database, then there is  */
  /* no point in doing any of this.
                */
  if( 0 < pContext->AdminUsers )
  {
    SCRlog( "\tAudit utilities will be run." );

    /* Get all the BE info we're going to need. */
    if( SCR_E_SUCCESS != ( Status = GetBeInfo( pContext )))
    {
      ( void ) SCRerr( SCR_E_GET_BE_INFO );
      return( Status );
    }

    Status = UTILGetAuditFunctionNetworkVariables( &pContext->BeTasksData );
    if( SCR_E_SUCCESS != Status )
    {
      SCRlog( "Error getting audit function net vars." );
      return( Status );
    }
    else
    {
      SCRlog( "\t\tDatabase type is %s", pContext->BeTasksData.DbTypeStr );
    }
  }

  return( SCR_E_SUCCESS );
} /* End of Init */



int32
InitAides( master_data_t *pContext )
{
  int32                         AideID;
  int32                         Index;
  int32                         Status;
```

```
  Status = SCR_E_SUCCESS;

  if( 0 < pContext->NumFEs || 0 < pContext->AdminUsers )
  {
    SCRlog( "\tAides being created and initialized.\n\n" );

    pContext->RequestsSent = 0;
    pContext->ReplysReceived = 0;
    pContext->AidesStatus = SCR_E_SUCCESS;
    pContext->MsgTimedOut = FALSE;

    for( Index = 0; Index < pContext->NumConnects; Index++ )
    {
      if( ! pContext->pConnects[Index].ConfConn &&
                ! pContext->pConnects[Index].AdminConn )
      {
                pContext->pConnects[Index].AideID = 0;
                continue;
      }
      AideID = PRTEexecute_script( pContext->AideScriptName,
                                                        pContext-
>pConnects[Index].RTEName,
                                                        1, pContext-
>pConnects[Index].FEName, 0.0 );
      if( -1 == AideID )
      {
                Status = SCRerr( SCR_E_AIDE_CREATE_FAILED );
                SCRlog( "\tPausing.\n" );
                PRTEpause();
                return( Status );
      }

      PRTEstart_users( AideID, AideID );

      /* Keep track of the aide's ID */
      pContext->pConnects[Index].AideID = AideID;
      pContext->RequestsSent++;
    }

    pContext->MsgTimedOut = TRUE;
    PRTEdelay( pContext->MsgTimeout );
    SCRlog( "Master: Resuming.\n" );

    if( pContext->MsgTimedOut )
    {
      /* We timed out waiting to hear from the aides. */
      Status = SCRerr( SCR_E_AIDE_INIT_TIMEOUT );
      SCRlog( "\tPausing indefinitely....\n\n" );
      PRTEpause();
    }

    if( SCR_E_SUCCESS != pContext->AidesStatus )
    {
      Status = SCRerr( pContext->AidesStatus );
      SCRlog( "\tNot all aides initialized correctly. Pausing.\n" );
      PRTEpause();
    }
    else
    {
      SCRlog( "\tAides successfully initialized.\n" );
    }
  }

  return( Status );
}


/*****************************************************************************
**********/
/*                                                                           */
/* Name: InitFEs                                                             */
/*                                                                           */
/* Purpose: This function causes the application code to be loaded in the FE. */
/*                                                                           */
/* Inputs:                                                                   */
/*                                                                           */
/* Outputs:                                                                  */
/*                                                                           */
/* Function Value:                                                           */
/*                                                                           */
/* Notes:                                                                    */
/*                                                                           */
/*****************************************************************************
**********/
int32
InitFEs( master_data_t *pContext )
{
  int32                                 Index;
  load_fe_msg_t                         LoadFEMsg;
  int32                                 Status;

  /* Check if we have any FEs to start. */
  if( 0 == pContext->NumFEs )
  {
    return( SCR_E_SUCCESS );
  }

  /* Get the timeout value to use when loading the user dll. */
  SCR_getnet_int( "LOAD_FE_TIMEOUT", LOAD_FE_TIMEOUT_DEFAULT,
                        &LoadFEMsg.Data.Timeout );
  SCRlog( "\tFE(s) being loaded.  FE Load Timeout = %d\n",
                LoadFEMsg.Data.Timeout );

  LoadFEMsg.Type = LOAD_FE_REQUEST_MSG;

  pContext->RequestsSent = 0;
  pContext->ReplysReceived = 0;
  pContext->AidesStatus = SCR_E_SUCCESS;
  pContext->MsgTimedOut = TRUE;

  for( Index = 0; Index < pContext->NumConnects; Index++)
  {
    if( ! pContext->pConnects[Index].ConfConn )
    {
      continue;
    }

    strcpy( LoadFEMsg.Data.FEName, pContext->pConnects[Index].FEName );
#pragma message ("FIXME: the cgi script (dll) name is possibly different.  This code
should parse a comma separated list?  Not here, but where the CgiScriptName is
originally loaded.")
    strcpy( LoadFEMsg.Data.CgiScriptName, pContext->CgiScriptName );

    PRTEmessage_to_user_binary( sizeof( LoadFEMsg ), &LoadFEMsg,
                                                        pContext-
>pConnects[Index].AideID,
                                                        pContext-
>pConnects[Index].AideID );
    pContext->RequestsSent++;
  }

  PRTEdelay( LoadFEMsg.Data.Timeout );

  Status = SCR_E_SUCCESS;
  if( pContext->MsgTimedOut )
  {
    /* We timed out waiting to hear from the aide(s). */
    Status = SCRerr( SCR_E_LOAD_FE_TIMEOUT );
  }
  else if( SCR_E_SUCCESS != pContext->AidesStatus )
  {
    Status = SCRerr( pContext->AidesStatus );
    Status = SCRerr( SCR_E_LOAD_FE_FAILURE );
  }
  else
  {
    SCRlog( "\tFE(s) successfully loaded.\n" );
  }

  if( SCR_E_SUCCESS != Status )
  {
    SCRlog( "\tFailed to load FEs' application code. Pausing.\n" );
    PRTEpause();
  }

  return( Status );
}
```

```
int32
InitReducer( master_data_t *pContext )
{
  int32                                 Status;

  Status = SCR_E_SUCCESS;

  /* Tell reducer to initialize. We'll resume ourselves from in the*/
  /* binary message handler once we hear back from reducer.                */
  SCRlog( "\tReducer being created.\n\n" );

  pContext->ReducerStatus = SCR_E_SUCCESS;
  pContext->MsgTimedOut = FALSE;

  pContext->ReducerID = PRTEexecute_script( pContext->ReducerScriptName,
                                                                  pContext-
>ReducerScriptNode,
                                                                  1, "", 0.0 );
  if( -1 == pContext->ReducerID )
  {
    Status = SCRerr( SCR_E_REDUCER_CREATE_FAILED );
    SCRlog( "\tPausing.\n" );
    PRTEpause();
  }
  else
  {
    SCRlog( "\tReducer being told to initialize.\n\n" );
    PRTEstart_users( pContext->ReducerID, pContext->ReducerID );

    pContext->MsgTimedOut = TRUE;
    PRTEdelay( pContext->MsgTimeout );
    SCRlog( "Master: Resuming.\n" );
  }

  if( pContext->MsgTimedOut )
  {
    /* We timed out waiting to hear from the reducer. */
    Status = SCRerr( SCR_E_REDUCER_INIT_TIMEOUT );
    SCRlog( "\tPausing.\n" );
    PRTEpause();
  }

  if( SCR_E_SUCCESS != pContext->ReducerStatus )
  {
    SCRlog( "\tReducer failed to initialize correctly. Pausing.\n" );
    PRTEpause();
  }
  else
  {
    SCRlog( "\tReducer successfully initialized.\n" );
  }

  return( Status );
}



/*********************************************************************
*********/
/*                                                                    */
/* Name: MakeRunDir                                                   */
/*                                                                    */
/* Purpose: This function creates the run results directory.          */
/*                                                                    */
/* Inputs:                                                            */
/*        DirName: Pointer to the parent directory where there run directory */
/*                 should be created.                                 */
/*                                                                    */
/* Outputs: Returns the run-dir number, or -1, there was an error     */
/*                                                                    */
/*                                                                    */
/* Function Value: a char * to a memory location where the name of the newly  */
/*                 created directory is (i.e. a null terminated string).      */
/*                                                                    */
/* Notes:                                                             */
/*                                                                    */
/*********************************************************************
*********/
int32
MakeRunDir(char *DirName, char *RunDir)
{
#ifdef _WIN32
  HANDLE                find_handle = INVALID_HANDLE_VALUE;
  WIN32_FIND_DATA   find_data;
#else
  DIR           *pDir;
  struct dirent *Entry;
#endif
  char          *c;
  char                          FormatStr[MAX_RUN_DIR_NAME_LEN + 3];    /* + 3 is
for %0 d/. */
  int           MaxDir=0;
  int           NewDir=0;
  int           Len;
  mode_t        Mode = 0770;

  /* Make sure we got something for input. */
  if( NULL == DirName )
    return -1;

  /* Now open the directory, read all entries, and find the current max. */
  /* In the WIN32 version, things are based on a group of files, in the  */
  /* Unix version, things are based on a directory.                      */
#ifdef _WIN32
  strcpy(RunDir, DirName);
  strcat(RunDir,"/*");
  find_handle = FindFirstFile(( LPTSTR )RunDir, &find_data );
  if( INVALID_HANDLE_VALUE == find_handle )
    return -1;

  do
  {
    c = find_data.cFileName;
    while( (*c != '\0') && ('0' <= *c) && (*c <= '9') )
    {
      c++;
    }

    if ( '\0' == *c)
    {
      MaxDir = maximum(MaxDir, atoi(find_data.cFileName));
    }
  } while( FindNextFile( find_handle, &find_data ));

  (void)FindClose( find_handle );
#else
  if (NULL == (pDir = opendir(DirName)))
  {
    return -1;
  }

  /* Check each entry in the directory. */
  for (Entry = readdir(pDir); Entry != NULL; Entry = readdir(pDir))
  {
    /* Make sure the name is all numbers. */
    c = Entry->d_name;
    while( (c != '\0') && ('0' <= *c) && (*c <= '9') )
    {
      c++;
    }

    if ( '\0' == *c)
    {
      MaxDir = maximum(MaxDir, atoi(Entry->d_name));
    }
  }

  closedir(pDir);
#endif

  /* The next dir will be "1 more" than the max found. */
  NewDir = MaxDir + 1;

  /* Move the parent directory name into the run directory. */
  strcpy(RunDir,DirName);

  /* Make sure we end with a directory seperator. */
```

```
  Len = strlen(RunDir);
  if ('/' != RunDir[Len])
  {
    RunDir[Len++] = '/';
  }

  /* NOTE: The value of 3 is hardcoded below.  This allows up to 999 run */
  /* directories which should be reasonable enough for anyone. Also note */
  /* that we need to bump the value of Len, so that it continues to      */
  /* point to the end of the full name.                                  */
  sprintf(FormatStr,"%%0%dd/", MAX_RUN_DIR_NAME_LEN);
  sprintf(&RunDir[Len], FormatStr, NewDir);
  Len += 4;   /* Size of field above plus "/" */

  /* Now actually make the run directory. */
  if ( -1 == MKDIR(RunDir, Mode) )
  {
    return -1;
  }
  SCRlog("\tRun Dir = %s", RunDir);
  PRTEset_network_variable("RUN_DIR", RunDir);

  /* Now temporarily add the log dir to the run dir, make the log dir, and */
  /* then remove the log portion of the path so we're just left with run   */
  /* dir again.                                                            */
  strcat(RunDir, LOG_DIR);
  if (-1 == MKDIR(RunDir, Mode))
  {
    return -1;
  }
  PRTEset_network_variable("LOG_DIR", RunDir);
  RunDir[Len] = '\0';

  return NewDir;
}


/*********************************************************************
*********/
/*                                                                   */
/* Name: StartFEs                                                    */
/*                                                                   */
/* Purpose: This function causes the FEs to be started.              */
/*                                                                   */
/* Inputs:                                                           */
/*                                                                   */
/* Outputs:                                                          */
/*                                                                   */
/* Function Value:                                                   */
/*                                                                   */
/* Notes:                                                            */
/*                                                                   */
/*********************************************************************
*********/
int32
StartFEs( master_data_t *pContext )
{
  int32                 Index;
  start_fe_msg_t        StartFEMsg;
  start_fe_msg_t        *pStartFEMsg = &StartFEMsg;
  double                StartTimeout;
  int32                 Status;

  Status = SCR_E_SUCCESS;
  if( 0 == pContext->NumFEs )
  {
    return( SCR_E_SUCCESS );
  }

#pragma message ("FIXME: Should there be a separate start timeout as network
variable?")
  StartTimeout = pContext->MsgTimeout;

  pContext->RequestsSent = 0;
  pContext->ReplysReceived = 0;
  pContext->AidesStatus = SCR_E_SUCCESS;
  pContext->MsgTimedOut = TRUE;
  pStartFEMsg->Type = START_FE_REQUEST_MSG;
  pStartFEMsg->Data.Timeout = StartTimeout;
  for( Index = 0; Index < pContext->NumConnects; Index++ )
  {
    if( ! pContext->pConnects[Index].ConfConn )
    {
      continue;
    }
    else if( FE_RESTART_ALWAYS != pContext->RestartFEs &&
             ! pContext->pConnects[Index].ConfMod )
    {
      continue;
    }
    strcpy( pStartFEMsg->Data.FEName, pContext->pConnects[Index].FEName );
    PRTEmessage_to_user_binary( sizeof( *pStartFEMsg ), pStartFEMsg,
                                pContext-
>pConnects[Index].AideID,
                                pContext-
>pConnects[Index].AideID );
    pContext->RequestsSent++;
    SCRlog( "\tStarting %s.", pContext->pConnects[Index].FEName );
  }

  /* Now wait for replies. */
  if( 0 == pContext->RequestsSent )
  {
    return( SCR_E_SUCCESS );
  }

  /* Wait as long as the Aides do to start FEs plus the msg timeout. */
  PRTEdelay( StartTimeout + pContext->MsgTimeout );

  if( pContext->MsgTimedOut )
  {
    Status = SCRerr( SCR_E_AIDE_TIMEOUT_START_FE );
    SCRlog( "Pausing...\n" );
    PRTEpause( );
  }

  if( SCR_E_SUCCESS != pContext->AidesStatus )
  {
    Status = SCRerr( pContext->AidesStatus );
    Status = SCRerr( SCR_E_AIDES_FAILED_START );
  }
  else
  {
    for( Index = 0; Index < pContext->NumConnects; Index++ )
    {
      if( ! pContext->pConnects[Index].ConfConn )
      {
        continue;
      }
      else if( FE_RESTART_ALWAYS != pContext->RestartFEs &&
               ! pContext->pConnects[Index].ConfMod )
      {
        continue;
      }
      SCRlog( "\tSuccessfully started %s.",
              pContext->pConnects[Index].FEName );
    }
  }

  return( Status );
}



int32
StopAides( master_data_t *pContext )
{
  generic_msg_t                 GenericMsg;
  int32                         Index;
  int32                         Status;

  Status = SCR_E_SUCCESS;

  if( 0 < pContext->NumFEs || 0 < pContext->AdminUsers )
  {
    SCRlog( "\tAides being told to exit." );
    pContext->RequestsSent = 0;
    GenericMsg.Type = AIDE_EXIT_REQUEST_MSG;
    pContext->ReplysReceived = 0;
    pContext->AidesStatus = SCR_E_SUCCESS;
```

```
    pContext->MsgTimedOut = TRUE;

    for( Index = 0; Index < pContext->NumConnects; Index++ )
    {
      if( ! pContext->pConnects[Index].ConfConn )
      {
                continue;
      }
      PRTEmessage_to_user_binary( sizeof( GenericMsg ), &GenericMsg,
                                                        pContext-
>pConnects[Index].AideID,
                                                        pContext-
>pConnects[Index].AideID );
      pContext->RequestsSent++;
    }

    PRTEdelay( pContext->MsgTimeout );

    if( pContext->MsgTimedOut )
    {
      Status = SCRerr( SCR_E_AIDE_EXIT_TIMEOUT );
    }

    if( SCR_E_SUCCESS != pContext->AidesStatus )
    {
      Status = SCRerr( pContext->AidesStatus );
      SCRlog( "Not all aides exited successfully." );
    }
    else
    {
      SCRlog( "\t%d Aides exited successfully.\n", pContext->ReplysReceived );
    }
  }

  return( Status );
}


/*****************************************************************************
*********/
/*                                                                           */
/* Name: StopFEs                                                             */
/*                                                                           */
/* Purpose: This function causes the FEs to be stopped.                      */
/*                                                                           */
/* Inputs:                                                                   */
/*                                                                           */
/* Outputs:                                                                  */
/*                                                                           */
/* Function Value:                                                           */
/*                                                                           */
/* Notes:                                                                    */
/*                                                                           */
/*****************************************************************************
*********/
int32
StopFEs( master_data_t *pContext )
{
  int32                         Index;
  stop_fe_msg_t                 StopFEMsg;
  stop_fe_msg_t                 *pStopFEMsg = &StopFEMsg;
  double                StopTimeout;
  int32                         Status;

  Status = SCR_E_SUCCESS;
  if( 0 == pContext->NumFEs )
  {
    return( SCR_E_SUCCESS );
  }

#pragma message ("FIXME: Should there be a separate stop timeout as network
variable?")
  StopTimeout = pContext->MsgTimeout;

  pContext->RequestsSent = 0;
  pContext->ReplysReceived = 0;
  pContext->AidesStatus = SCR_E_SUCCESS;
  pContext->MsgTimedOut = TRUE;
  pStopFEMsg->Type = STOP_FE_REQUEST_MSG;
  pStopFEMsg->Data.Timeout = StopTimeout;
  pStopFEMsg->Data.Reboot = pContext->FERestartViaReboot;
  for( Index = 0; Index < pContext->NumConnects; Index++ )
  {
    if( ! pContext->pConnects[Index].ConfConn )
    {
      continue;
    }
    else if( FE_RESTART_ALWAYS != pContext->RestartFEs &&
                ! pContext->pConnects[Index].ConfMod )
    {
      continue;
    }
    strcpy( pStopFEMsg->Data.FEName, pContext->pConnects[Index].FEName );
    PRTEmessage_to_user_binary( sizeof( *pStopFEMsg ), pStopFEMsg,
                                                        pContext-
>pConnects[Index].AideID,
                                                        pContext-
>pConnects[Index].AideID );
    pContext->RequestsSent++;
    SCRlog( "\tStoping %s.", pContext->pConnects[Index].FEName );
  }

  /* Now wait for replies. */
  if( 0 == pContext->RequestsSent )
  {
    return( SCR_E_SUCCESS );
  }

  /* Wait as long as the Aides do to stop FEs plus the msg timeout. */
  PRTEdelay( StopTimeout + pContext->MsgTimeout );

  if( pContext->MsgTimedOut )
  {
    Status = SCRerr( SCR_E_AIDE_TIMEOUT_STOP_FE );
    SCRlog( "Pausing...\n" );
    PRTEpause( );
  }

  if( SCR_E_SUCCESS != pContext->AidesStatus )
  {
    Status = SCRerr( pContext->AidesStatus );
    Status = SCRerr( SCR_E_AIDES_FAILED_STOP );
  }
  else
  {
    for( Index = 0; Index < pContext->NumConnects; Index++ )
    {
      if( ! pContext->pConnects[Index].ConfConn )
      {
                continue;
      }
      else if( FE_RESTART_ALWAYS != pContext->RestartFEs &&
                  ! pContext->pConnects[Index].ConfMod )
      {
                continue;
      }
      SCRlog( "\tSuccessfully stopped %s.",
                  pContext->pConnects[Index].FEName );
    }
  }

  return( Status );
}


int32
StopReducer( master_data_t *pContext )
{
  /* Nothing to be done here. */
  return( SCR_E_SUCCESS );
}


void
TellAidesTo( master_data_t *pContext, int32 Task, int32 State )
{
  int32 BeIndex;
  int32 Index;
  generic_dotask_msg_t SendMsg;
```

```
  /* Check to see if any aides are configured */
  if (pContext->AdminUsers == 0)
    return;


  SCRlog( "\t%d aides being told to %s.", pContext->AdminUsers, MsgNames[Task]);
  SendMsg.Type = Task;
  SendMsg.Data.State = State;

  pContext->RequestsSent = 0;
  pContext->ReplysReceived = 0;
  pContext->AidesStatus = SCR_E_SUCCESS;
  pContext->MsgTimedOut = TRUE;

  for( Index = 0, BeIndex = 0; Index < pContext->NumConnects; Index++ )
  {
    if( ! pContext->pConnects[Index].AdminConn )
    {
      continue;
    }
    SendMsg.Data.HostId = BeIndex + 1;
    strcpy( SendMsg.Data.SutName, pContext->pBEs[BeIndex].Name );
    PRTEmessage_to_user_binary( sizeof( SendMsg ), &SendMsg,
                                                    pContext-
>pConnects[Index].AideID,
                                                    pContext-
>pConnects[Index].AideID );
    pContext->RequestsSent++;
    BeIndex++;
  }

  PRTEdelay( pContext->MsgTimeout );

  if( pContext->MsgTimedOut )
  {
    SCRlog( "\tTimed out waiting for %d aides to %s.",
                ( pContext->RequestsSent - pContext->ReplysReceived ),
                MsgNames[Task] );
    SCRlog( "\tPausing.\n\n" );
    PRTEpause();
  }
  else if( SCR_E_SUCCESS != pContext->AidesStatus )
  {
    ( void )SCRerr( pContext->AidesStatus );
    SCRlog( "\tAides failed to %s.", MsgNames[Task] );
    SCRlog( "\tPausing.\n\n" );
    PRTEpause();
  }
  else
  {
    SCRlog( "\t%d aides successfully did %s.\n", pContext->AdminUsers,
                MsgNames[Task]);
  }

  return;
}


int32
TestPeriod( master_data_t *pContext )
{
  checkpoint_connect_msg_t      CheckpointConnect;
  checkpoint_run_msg_t          CheckpointRun;
  checkpoint_disconnect_msg_t   CheckpointDisconnect;
  generic_msg_t                         GenericMsg;
  int32                                 Index;
  measurement_info_msg_t        Measurement;
  double                        TimeStamp;
  double                        TimeStamp2;
  double                        DelayTime;
  double                        StartTime;
  int32                                 Status;


  SCRlog( "\n\n\n\t\t\t\t\t..START OF TEST..\n\n" );

  /* Tell reducer to initialize. */
  Status = InitReducer( pContext );

  /* Tell any aides to initialize. */
  Status = InitAides( pContext );

  /* Configure the FEs */
  Status = ConfigFEs( pContext );

  /* Make sure the user application is loaded on all FEs. */
  Status = InitFEs( pContext );

  /* If Checkpoints requested then do Back-end Checkpoint stuff */
  if( pContext->DoCheckpoints )
  {
    TellAidesTo( pContext, BE_CKPTDATA_REQUEST_MSG, BEFORE );
  }

  /* For Oracle, do switch logfile.  If stats requested, do stats startup
     script, too */
  if( pContext->BeTasksData.DbType == DB_TYPE_ORACLE )
  {
    TellAidesTo( pContext, ORA_SWITCHLOG_REQUEST_MSG, BEFORE );

    if( '\0' != pContext->BeTasksData.OraStatScriptPath[0] )
    {
      TellAidesTo( pContext, ORA_DOSTATS_REQUEST_MSG, BEFORE );
    }
  }

  /* If GetAllAuditFiles specified -- get the log size info */
  if( pContext->BeTasksData.GetAllAuditFiles )
  {
    TellAidesTo( pContext, INIT_AUDIT_TABLE_REQUEST_MSG, BEFORE );
    TellAidesTo( pContext, GET_LOGSIZE_REQUEST_MSG, BEFORE );
  }

  /* If custom script specified -- go do it */
  if( '\0' != pContext->BeTasksData.CustomBeforeTestScript[0] )
  {
    TellAidesTo( pContext, DO_CUSTOM_SCRIPT_REQUEST_MSG, BEFORE );
  }

  /* Tell the reducer to log in users. */
  SCRlog("\tPausing while users log in.\n\n");
  pContext->ReducerStatus = SCR_E_SUCCESS;
  PRTEresume_users( pContext->ReducerID, pContext->ReducerID );
  PRTEpause();


  SCRlog("Master: Resuming.\n");

  if( SCR_E_SUCCESS != pContext->ReducerStatus )
  {
    SCRlog( "\tUsers failed to log in successfully. Pausing.\n" );
    PRTEpause();
  }
  else
  {
    SCRlog( "\tAll users logged in successfully.\n" );
  }

  /* If we're doing checkpoints, have the aides init checkpoint stuff. */
  if( pContext->DoCheckpoints )
  {
    SCRlog( "\t%d Aides initializing checkpoint sub-systems.",
                        pContext->AdminUsers );
    CheckpointConnect.Type = CHECKPOINT_CONNECT_REQUEST_MSG;
    strcpy( CheckpointConnect.Data.Dll, pContext->CgiScriptName );
    pContext->RequestsSent = 0;
    pContext->ReplysReceived = 0;
    pContext->AidesStatus = SCR_E_SUCCESS;
    pContext->MsgTimedOut = TRUE;

    for( Index = 0; Index < pContext->NumConnects; Index++ )
    {
      if( ! pContext->pConnects[Index].AdminConn )
      {
                continue;
      }
      strcpy( CheckpointConnect.Data.FEName, pContext-
>pConnects[Index].FEName);
```

```
    PRTEmessage_to_user_binary( sizeof( CheckpointConnect ),
                                                &CheckpointConnect,
                                                pContext-
>pConnects[Index].AideID,
                                                pContext-
>pConnects[Index].AideID );
    pContext->RequestsSent++;
   }

   PRTEdelay( pContext->MsgTimeout );

   if( pContext->MsgTimedOut )
   {
    ( void ) SCRerr( SCR_E_AIDE_CKPT_INIT_TIMEOUT );
    SCRlog( "Pausing indefinitely...\n\n" );
    PRTEpause();
   }

   if( SCR_E_SUCCESS != pContext->AidesStatus )
   {
    Status = SCRerr( pContext->AidesStatus );
    SCRlog( "Not all aides did checkpoint initialization successfully." );
    SCRlog( "Pausing...\n\n" );
    PRTEpause();
   }

   SCRlog( "\t%d Aides successfully initialized checkpoint sub-systems.\n",
                pContext->ReplysReceived );
  }

  /* Tell the reducer to resume users for transactions. */
  SCRlog("\tPausing while users start doing transactions.\n\n");
  pContext->ReducerStatus = SCR_E_SUCCESS;
  PRTEresume_users( pContext->ReducerID, pContext->ReducerID );
  PRTEpause();


  SCRlog("Master: Resuming.\n");

  if( SCR_E_SUCCESS != pContext->ReducerStatus )
  {
   SCRlog( "\tUsers failed to resume successfully. Pausing.\n" );
   PRTEpause();
  }
  else
  {
   SCRlog( "\tAll users starting transactions successfully.\n" );
  }

  /* Warmup */
  TimeStamp = PRTEtimer_begin( "x" );
  PRTEtimer_cancel();

  SCRlog( "\n\n\n\n\t\t\t\tStart of Warmup\n\n");

  /* If we're running checkpoints, have the aides start doing them. */
  if( pContext->DoCheckpoints )
  {
   SCRlog( "\t%d Aides starting checkpoint loop.",
                pContext->AdminUsers );

   /* Calculate when the checkpoints should start. This is done by finding */
   /* whent the checkpoint during the start of measurement needs to start  */
   /* and back calculating when to start the first one which is during     */
   /* warmup.                                                              */
   CheckpointRun.Type = CHECKPOINT_RUN_REQUEST_MSG;
   StartTime = TimeStamp + pContext->CkptStartOffset;
   CheckpointRun.Data.StartTime = StartTime;
   CheckpointRun.Data.Interval = pContext->CheckpointInterval;
   strcpy( CheckpointRun.Data.Dll, pContext->CgiScriptName );

   pContext->RequestsSent = 0;
   pContext->ReplysReceived = 0;
   pContext->AidesStatus = SCR_E_SUCCESS;
   pContext->MsgTimedOut = TRUE;

   for( Index = 0; Index < pContext->NumConnects; Index++ )
   {
    if( ! pContext->pConnects[Index].AdminConn )
    {
                continue;
    }
    strcpy( CheckpointRun.Data.FEName, pContext->pConnects[Index].FEName );
    PRTEmessage_to_user_binary( sizeof( CheckpointRun ), &CheckpointRun,
                                                pContext-
>pConnects[Index].AideID,
                                                pContext-
>pConnects[Index].AideID );
    pContext->RequestsSent++;
   }

   PRTEdelay( pContext->MsgTimeout );

   if( pContext->MsgTimedOut )
   {
    ( void ) SCRerr( SCR_E_AIDE_CKPT_RUN_TIMEOUT );
    SCRlog( "Pausing indefinitely...\n\n" );
    PRTEpause();
   }
   else if( SCR_E_SUCCESS != pContext->AidesStatus )
   {
    Status = SCRerr( pContext->AidesStatus );
    SCRlog( "Not all aides entered checkpoint loop.." );
    SCRlog( "Pausing...\n\n" );
    PRTEpause();
   }
   else
   {
    SCRlog( "\t%d Aides successfully entered checkpoint loop.\n",
                pContext->ReplysReceived );
   }
  }

  /* Now that we're done, find out how much time is left in warmup, and */
  /* sleep for that time.                                               */
  TimeStamp2 = PRTEtimer_begin( "x" );
  PRTEtimer_cancel();
  DelayTime = pContext->WarmUpTime - (TimeStamp2 - TimeStamp);

  PRTEdelay( DelayTime );

  /* Measurement */
  TimeStamp = PRTEtimer_begin( "x" );
  PRTEtimer_cancel();

  SCRlog( "\n\n\n\n\t\t\t\tStart of Measurement\n\n" );

  Measurement.Type = MEASUREMENT_INFO_MSG;
  Measurement.Data.Length = pContext->SteadyStateTime;
  Measurement.Data.StartTime = TimeStamp;
  PRTEmessage_to_user_binary(sizeof(Measurement), &Measurement,
                                        pContext->ReducerID, pContext-
>ReducerID);

  TimeStamp2 = PRTEtimer_begin( "x" );
  PRTEtimer_cancel();
  DelayTime = pContext->SteadyStateTime - (TimeStamp2 - TimeStamp);
  PRTEdelay( DelayTime );


  /* Cooldown */
  TimeStamp = PRTEtimer_begin( "x" );
  PRTEtimer_cancel();

  SCRlog( "\n\n\n\n\t\t\t\tStart of Cooldown" );

  TimeStamp2 = PRTEtimer_begin( "x" );
  PRTEtimer_cancel();
  DelayTime = pContext->CoolDownTime - (TimeStamp2 - TimeStamp);
  PRTEdelay( DelayTime );

  SCRlog( "\n\n\n\n\t\t\t\tEnd of Cooldown\n\n");

  /* If we were doing checkpoints, have the aides stop now. */
  if( pContext->DoCheckpoints )
  {
   SCRlog( "\t%d Aides stopping checkpoint loop.",
                                        pContext->AdminUsers );

   GenericMsg.Type = CHECKPOINT_STOP_REQUEST_MSG;
```

```
    pContext->RequestsSent = 0;
    pContext->ReplysReceived = 0;
    pContext->AidesStatus = SCR_E_SUCCESS;
    pContext->MsgTimedOut = TRUE;

    for( Index = 0; Index < pContext->NumConnects; Index++ )
    {
      if( ! pContext->pConnects[Index].AdminConn )
      {
                continue;
      }
      PRTEmessage_to_user_binary( sizeof( GenericMsg ), &GenericMsg,
                                                        pContext-
>pConnects[Index].AideID,
                                                        pContext-
>pConnects[Index].AideID );
      pContext->RequestsSent++;
    }

    PRTEdelay( pContext->MsgTimeout );

    if( pContext->MsgTimedOut )
    {
      ( void ) SCRerr( SCR_E_AIDE_CKPT_STOP_TIMEOUT );
      SCRlog( "Pausing indefinitely...\n\n" );
      PRTEpause();
    }

    if( SCR_E_SUCCESS != pContext->AidesStatus )
    {
      Status = SCRerr( pContext->AidesStatus );
      SCRlog( "Not all aides successfully ran checkpoint loop." );
      SCRlog( "Pausing...\n\n" );
      PRTEpause();
    }

    SCRlog( "\t%d Aides successfully ran checkpoint loop.\n",
                    pContext->ReplysReceived );

  }

  /* Break reducer out of his "infinite process data" loop. */
  PRTEstop_users( pContext->ReducerID, pContext->ReducerID );
  PRTEpause();

  /* Tell the reducer to stop users. */
  SCRlog("\tPausing while users stop doing transactions.\n\n");
  pContext->ReducerStatus = SCR_E_SUCCESS;
  PRTEresume_users( pContext->ReducerID, pContext->ReducerID );
  PRTEpause();

  SCRlog("Master: Resuming.\n");

  if( SCR_E_SUCCESS != pContext->ReducerStatus )
  {
    SCRlog( "\tUsers failed to stop successfully. Pausing.\n" );
    PRTEpause();
  }
  else
  {
    SCRlog( "\tAll users successfully stopped doing transactions.\n" );
  }


  /* If we're doing checkpoints, have the aides disconnect checkpoint stuff. */
  if( pContext->DoCheckpoints )
  {
    SCRlog( "\t%d Aides disconnecting checkpoint sub-systems.",
                    pContext->AdminUsers );
    CheckpointDisconnect.Type = CHECKPOINT_DISCONNECT_REQUEST_MSG;
    strcpy( CheckpointDisconnect.Data.Dll, pContext->CgiScriptName );
    pContext->RequestsSent = 0;
    pContext->ReplysReceived = 0;
    pContext->AidesStatus = SCR_E_SUCCESS;
    pContext->MsgTimedOut = TRUE;

    for( Index = 0; Index < pContext->NumConnects; Index++ )
    {
      if( ! pContext->pConnects[Index].AdminConn )
      {
                continue;
      }
      strcpy( CheckpointDisconnect.Data.FEName,
                    pContext->pConnects[Index].FEName );
      PRTEmessage_to_user_binary( sizeof( CheckpointDisconnect ),
                                                        &CheckpointDisconnect,
                                                        pContext-
>pConnects[Index].AideID,
                                                        pContext-
>pConnects[Index].AideID );
      pContext->RequestsSent++;
    }

    PRTEdelay( pContext->MsgTimeout );

    if( pContext->MsgTimedOut )
    {
      ( void ) SCRerr( SCR_E_AIDE_CKPT_DISCONNECT_TIMEOUT );
      SCRlog( "Pausing indefinitely...\n\n" );
      PRTEpause();
    }

    if( SCR_E_SUCCESS != pContext->AidesStatus )
    {
      Status = SCRerr( pContext->AidesStatus );
      SCRlog( "Not all aides did checkpoint disconnect successfully." );
      SCRlog( "Pausing...\n\n" );
      PRTEpause();
    }

    SCRlog( "\t%d Aides successfully disconnected checkpoint sub-systems.\n",
                    pContext->ReplysReceived );
  }

  /* Tell the reducer to post process data. */
  SCRlog("\tPost process binary logfile.\n\n");

  pContext->ReducerStatus = SCR_E_SUCCESS;
  PRTEresume_users( pContext->ReducerID, pContext->ReducerID );
  PRTEpause();

  SCRlog("Master: Resuming.\n");


  /* Do post-test back-end functions */


  /* If Checkpoints requested then do Back-end Checkpoint stuff */
  if( pContext->DoCheckpoints )
  {
    TellAidesTo( pContext, BE_CKPTDATA_REQUEST_MSG, AFTER );
  }


  /* For Oracle, do switch logfile.  If stats requested, do stats end
     script, too */

  /* If GetAllAuditFiles specified -- get the log size info */
  if( pContext->BeTasksData.GetAllAuditFiles )
  {
    TellAidesTo( pContext, GET_LOGSIZE_REQUEST_MSG, AFTER );
  }

  if( pContext->BeTasksData.DbType == DB_TYPE_ORACLE )
  {
    if( pContext->BeTasksData.OraStatScriptPath[0] )
      TellAidesTo( pContext, ORA_DOSTATS_REQUEST_MSG, AFTER );

    TellAidesTo( pContext, ORA_SWITCHLOG_REQUEST_MSG, AFTER );
  }


  /* If custom script specified -- go do it */
  if( pContext->BeTasksData.CustomAfterTestScript[0] )
    TellAidesTo( pContext, DO_CUSTOM_SCRIPT_REQUEST_MSG, AFTER );

  /* Tell the reducer to post process data. */
  SCRlog("\tPost process aide logfiles.\n\n");
```

```
  pContext->ReducerStatus = SCR_E_SUCCESS;
  PRTEresume_users( pContext->ReducerID, pContext->ReducerID );
  PRTEpause();

  SCRlog("Master: Resuming.\n");


  /* Tell any aides to exit. */
  Status = StopAides( pContext );

  Status = StopReducer( pContext );

  free( pContext->pConnects );

  SCRlog( "\n\n\n\n\t\t\t\t..END OF TEST..\n\n\n" );

#pragma message ("FIXME: This routine needs to keep a running track of the status of
all operations.")
  return( Status );
}
```

tpcc_user.c

```
/*+*********************************************************************
*******
 *                                                                     *
 *  COPYRIGHT (c) 1999 BY                                              *
 *  COMPAQ COMPUTER CORPORATION, MAYNARD, MASSACHUSETTS.
*
 *  ALL RIGHTS RESERVED.                                               *
 *                                                                     *
 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED   *
 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND
WITH THE   *
 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR
ANY OTHER   *
 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE
AVAILABLE TO ANY   *
 *  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE
IS  HEREBY   *
 *  TRANSFERRED.                                                       *
 *                                                                     *
 *  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE
WITHOUT NOTICE   *
 *  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY
COMPAQ  COMPUTER   *
 *  CORPORATION.                                                       *
 *                                                                     *
 *  COMPAQ ASSUMES NO RESPONSIBILITY FOR THE USE  OR
RELIABILITY OF ITS    *
 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY COMPAQ.
*
 *                                                                     *
 *                                                                     *

**********************************************************************
*****-*/


/* This script is the main script for each tpcc user. It
   handles communications with other scripts, control
   of the transaction mix and initialiazion and shutdown of
   of each user.  It calls routines in tpcc_gen.c to
   get the actual data for each transaction.
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <errno.h>
#ifdef _WIN32
# include <nt_lib.h>
# include <time.h>
#endif

#include <stdtypes.h>
#include <prte.h>
#include <common.h>

#include <tpccstruct.h>
#include <tpccapi.h>
#include <tpccerr.h>
#include <transpool.h>
#include <config.h>

#include <scr_util.h>

#include <msg.h>
#include <netvar.h>
#include <tx_api.h>

/* Maximum values set for strings and such */
#define MAXSTRINGSIZE   4096

typedef struct _user_context_t {
  long                                  myseed;
  char                                  *Version;
  txn_context_t                         TxnContext;
} user_context_t;

/* Prototypes */
void BinMsgHandler(void *Context, int SendersID, int Len, void *Msg);
void DisconnectExit(user_context_t *pContext, int status, int Disconnect);
int  Init(user_context_t *pContext);

/* globals variables */

void
main(int argc, char *argv[])
{
  generic_status_msg_t    StatusMsg;
  user_login_reply_msg_t LoginMsg;
  user_context_t          Context;
  user_context_t          *pContext = &Context;

  /* Init PRTE */
  PRTEinit(argc,argv);

  /* Register binary message handler with PRTE. */
  /* This must be done before the user pauses so he can wake himself in */
  /* response to a transaction message.                                 */
  PRTEcatch_message_binary(BinMsgHandler, pContext);

  /* Initialize transaction timing records. */
  pContext->TxnContext.TxnTiming.Data.MenuStart = 0.0;
  pContext->TxnContext.TxnTiming.Data.MenuResponse = 0.0;
  pContext->TxnContext.TxnTiming.Data.TxnStart = 0.0;
  pContext->TxnContext.TxnTiming.Data.TxnResponse = 0.0;

  /* Let the reducer know we are initialized. */
  pContext->TxnContext.ReducerID =
    atoi( PRTEwait_for_network_variable( "REDUCER_ID" ));
  StatusMsg.Status = SCR_E_SUCCESS;
  StatusMsg.Type = USER_INIT_REPLY_MSG;
  PRTEmessage_to_user_binary( sizeof( StatusMsg ), &StatusMsg,
                                          pContext->TxnContext.ReducerID,
                                          pContext->TxnContext.ReducerID );

  /* Wait to be told to continue */
  PRTEpause( );

  LoginMsg.Type = USER_LOGIN_REPLY_MSG;

  /* Get variables to figure out what to do */
  if( SCR_E_SUCCESS != ( LoginMsg.Status = Init( pContext )))
  {
    /* Exit, but don't need to disconnect since we haven't connected yet. */
    PRTEmessage_to_user_binary( sizeof( LoginMsg ), &LoginMsg,
                                                pContext->TxnContext.ReducerID,
                                                pContext->TxnContext.ReducerID
);
    DisconnectExit ( pContext, LoginMsg.Status, FALSE );
  }
```

```
 if( SCR_E_SUCCESS != ( LoginMsg.Status = StartupTxn( &pContext->TxnContext
)))
 {
  /* Disconnect application and exit. */
  PRTEmessage_to_user_binary( sizeof( LoginMsg ), &LoginMsg,
                                                  pContext->TxnContext.ReducerID,
                                                  pContext->TxnContext.ReducerID
);
  DisconnectExit( pContext, LoginMsg.Status, FALSE );
 }

 /* Connect to the SUT and database*/
 LoginMsg.Status = ConnectTxn( &pContext->TxnContext, CONNECT_TRANS,
                                                  &LoginMsg.Data );
 if( SCR_E_SUCCESS != LoginMsg.Status )
 {
  /* Disconnect application and exit. */
  PRTEmessage_to_user_binary( sizeof( LoginMsg ), &LoginMsg,
                                                  pContext->TxnContext.ReducerID,
                                                  pContext->TxnContext.ReducerID
);
  DisconnectExit( pContext, LoginMsg.Status, FALSE );
 }

 LoginMsg.Status = SCR_E_SUCCESS;

 PRTEmessage_to_user_binary( sizeof(LoginMsg), &LoginMsg,
                                          pContext->TxnContext.ReducerID,
                                          pContext->TxnContext.ReducerID );

 /* Now execute the transaction loop. */
 StatusMsg.Status = DequeueExecLoopTxn( &pContext->TxnContext );

 /* We're done doing transactions either because the test is over, or */
 /* because we encountered and error. In either event, disconnect the */
 /* application and exit.                                             */
 DisconnectExit( pContext, StatusMsg.Status, TRUE );
}




/*********************************************************************
*********/
/*                                                                   */
/* Name: BinMsgHandler                                               */
/*                                                                   */
/* Purpose: This is the message handler that gets registered with PRTE for    */
/*          processing binary messages sent to the User.             */
/*                                                                   */
/* Inputs:                                                           */
/*        Context: Pointer to the context structure. This structure is set   */
/*                 up elsewhere, and registered with PRTE to be passed in.   */
/*        SendersID: User ID number of the user that sent us the binary      */
/*                   message.                                        */
/*        Len: Length of the binary message in bytes.                */
/*        Msg: Pointer to the actual data being sent.                */
/*                                                                   */
/* Outputs: This routine records the data sent into the user and uses it to   */
/*          define the next transaction.                             */
/*                                                                   */
/* Function Value: void                                              */
/*                                                                   */
/* Notes: This funtion is called on our behalf by PRTE. In order for this     */
/*        to happen, we must be inside a call to some PRTE function. If we    */
/*        are running code outside a PRTE function, PRTE will not interrupt   */
/*        us (i.e. no signal is sent). It is only when we are inside a PRTE   */
/*       function call (e.g. PRTEdelay, etc.) that PRTE will check to see if  */
/*       we have a message that needs processing.                    */
/*                                                                   */
/*********************************************************************
*********/
void
BinMsgHandler( void *Context, int SendersID, int Len, void *Msg )
{
 user_context_t              *pContext;

 /* NOTE: These structures are prone to possible bugs if the user sending  */
 /* the message is on one platform with one type of structure padding      */
 /* rule, and the receiver is on another platform with a different padding */
 /* rule.                                                                  */
 generic_msg_t                       *pMsg;
 start_msg_t                         *pStartMsg;
 txn_msg_t                 *pTxnMsg;

 /* Effectively convert void * parameters to correct types. */
 pMsg = Msg;
 pContext = Context;

 /* Process messaged based on its type. */
 switch( pMsg->Type ) {
  case USER_LOGIN_REQUEST_MSG:
   PRTEresume( );
   break;

  case USER_START_REQUEST_MSG:
   pStartMsg = Msg;
   EnqueueTxn( &pContext->TxnContext, TXN_CACHE_SIZE, &pStartMsg-
>Data[0] );
   break;

  case TXN_MSG:
   pTxnMsg = Msg;
   EnqueueTxn( &pContext->TxnContext, 1, &pTxnMsg->Data );
   break;

  case USER_STOP_REQUEST_MSG:
   PRTEstop( );
   PRTEdelay( 0.0 );
   break;

  default:
   SCRlog( "User: Received unknown message type: %d", *pMsg );
 }
}


void
DisconnectExit(user_context_t *pContext, int Status, int Disconnect)
{
 user_exit_msg_t          ExitMsg;
 char                                tmp_buf[128];
 int                                 LocStatus;

 /* Disconnect from any application if told to. */
 if ( TRUE == Disconnect )
 {
  LocStatus = DisconnectTxn( &pContext->TxnContext, DISCONNECT_TRANS );
  if( SCR_E_SUCCESS != LocStatus )
  {
   SCRerr( LocStatus );
   SCRlog( "User %d: Failed to disconnect from application.", User_id );
  }
 }

 LocStatus = ShutdownTxn( &pContext->TxnContext );
 if( SCR_E_SUCCESS != LocStatus )
 {
  SCRerr( LocStatus );
  SCRlog( "User %d: Failed to shutdown application.", User_id );
 }

 /* Make sure we get our log file flushed out by turning on flushing, */
 /* and sending a final message.                                      */
 Flush_log = TRUE;
 if (Status == SCR_E_SUCCESS)
 {
  sprintf(tmp_buf,"User %d: Exited gracefully as a swan.", User_id);
  PRTEto_log(110, tmp_buf);
 }
 else
 {
  SCRlog( "User %d: Exited with fatal error (%d).", User_id, Status );
  SCRerr( Status );
 }

 /* Log our exit status with the reducer. */
  ExitMsg.Type = USER_EXIT_MSG;
```

```
    ExitMsg.Data.Status = Status;
    ExitMsg.Data.ExitTime = PRTEtimer_begin ( "x" );
    PRTEtimer_cancel();
    PRTEmessage_to_user_binary( sizeof(ExitMsg), &ExitMsg,
                                                 pContext->TxnContext.ReducerID,
                                                 pContext->TxnContext.ReducerID
);

  PRTEexit();
}



int
Init( user_context_t *pContext )
{
  char temp_string[128];
  char *pNetVarStr;
  char *master_seed;
  int  temp_int;

  /* Get the log file and logging type set up. It is important to do this */
  /* first because we need these in place for PRTEto_log calls to work.   */

  /* LOG_DIR is a network variable set by master. */
  SCR_getnet_string("LOG_DIR", LOG_DIR_DEFAULT, temp_string);
  if ('\0' == temp_string[0] )
  {
    return( SCRerr( SCR_E_LOG_DIR_NOT_SET ));
  }
  else
  {
    strcpy (Log_path, temp_string);
    strcpy (Err_path, temp_string);
  }

  /* The error log name is built based on the user id. */
  sprintf(temp_string, "user_%d.err",User_id);
  strcpy (Err_name, temp_string);

  /* The log name is built based on the user id. */
  sprintf(temp_string, "user_%d.log",User_id);
  strcpy (Log_name, temp_string);

  /* Get/set the logging type.  Logging_type is a bit mask.             */
  /* Its values are defined in prte.h.                                  */
  SCR_getnet_int("TPCC_USER_LOG_TYPE",
TPCC_USER_LOG_TYPE_DEFAULT,
                                &Logging_type);
  Logging_type |= (USER_SUT_DATA_LOGGING);

  /* Set our version based on compile command line defined constant. */
  pContext->Version = VERSION;

  /* Verify our version with the master's. */
  pNetVarStr = PRTEget_network_variable("TPCC_SCRIPTS_VERSION");
  if( NULL == pNetVarStr )
  {
    /* The required network variable was not set. */
    return( SCRerr( SCR_E_CODE_VERSION_NOT_SET ));
  }
  else if( STRING_MATCH != strcmp( pContext->Version, pNetVarStr ))
  {
    SCRlog( "User code version = %s, Master code version = %s.",
                  pContext->Version, pNetVarStr );
    return( SCRerr( SCR_E_CODE_VERSION_MISMATCH ));
  }


  /* Get/set the flush log type. */
  SCR_getnet_int("TPCC_USER_FLUSH_LOG",
TPCC_USER_FLUSH_LOG_DEFAULT,
                                &Flush_log);

  /* Find out if we're running in RTE run mode. */
  SCR_getnet_int( "RUN_MODE", RUN_MODE_DEFAULT,
                                  &pContext->TxnContext.RunMode );

  /* Get the cgi script name. */
  SCR_getnet_string( "CGI_SCRIPT_NAME", CGI_SCRIPT_NAME_DEFAULT,
                                       pContext->TxnContext.CgiScriptName );
  if( '\0' == *pContext->TxnContext.CgiScriptName )
  {
    return( SCRerr( SCR_E_CGI_SCRIPT_NAME_NOT_SET ));
  }

  /* Get the seed from the master (or input file) and make it unique for each */
  /* user  - then initialize the random functions.  Force the seed to be odd. */
  master_seed = PRTEwait_for_network_variable("SEED");
  pContext->myseed = ( atol( master_seed ) * User_id * 2000 ) + 1;

  PRTErandomize(((unsigned) pContext->myseed % 100)); /* seed PRTE random */
  srand48(pContext->myseed);                          /* seeds the rand48 random
function */

  /* Find out if we're running a durability test. */
  SCR_getnet_bool( "DURABILITY_LOGGING",
DURABILITY_LOGGING_DEFAULT,
                                &pContext->TxnContext.DurabilityLogging );

  /* Calculate the w_id and d_id for this User */
  pNetVarStr = PRTEget_network_variable( "FIRST_USER_ID" );
  if( NULL == pNetVarStr )
  {
    return( SCRerr( SCR_E_FIRST_USER_ID_NOT_SET ));
  }

  temp_int = User_id - atoi( pNetVarStr );
  pContext->TxnContext.w_id = ( temp_int / 10 ) + 1;
  pContext->TxnContext.d_id = ( temp_int % 10 ) + 1;

  /* Get the value we should use for C_LAST. */
  SCR_getnet_int( "C_LAST", C_LAST_DEFAULT, &( pContext-
>TxnContext.CLast ));

  return( SCR_E_SUCCESS );
}
```

Appendix D

WFTPCC1

```
ident                   "WFTPCC1"

options                 AUTOFS
options                 BIN_COMPAT
options                 BSD_TTY
options                 BUFCACHE_STATS
options                 CDFS
options                 COMPAT_43
options                 DLB
options                 DLI
options                 FFM_FS
options                 INET
options                 INOCACHE_STATS
options                 LABELS
options                 LDTTY
options                 MACH
options                 MACH_IPC_TCACHE
options                 MACH_IPC_WWA
options                 MACH_IPC_XXXHACK
options                 MSFS
options                 NFS
options                 NFS_SERVER
options                 OSF
options                 QUOTA
options                 RPTY
options                 STAT_TIME
options                 STREAMS
options                 STRKINFO
options                 SYSV_COFF
options                 UERF
options                 UFS
options                 UIPC
options                 VAGUE_STATS
options                 XTISO
options                 _LMF_
options     BPARM
options     DEC_AUDIT
options     KDEBUG
options     NTP_TIME
options     PACKETFILTER
options     PROCFS
options     SL
options     TIMOD
options     FFM_FS
options     TIRDWR
#options    IPTUNNEL
#options   IPV6
options   PCKT
options   PPP
options SNMPINFO

#
# Standard options.
#
options                 UNIX_LOCKS
options                 SER_COMPAT
options                 RT_PREEMPT
options                 RT_SCHED
options                 RT_SCHED_RQ
options                 RT_PML
options                 RT_TIMER
options                 RT_SEM
options                 RT_CSEM
options                 RT_IPC
#
makeoptionsCCOMPRESS="-compress"
makeoptionsCDEBUGOPTS="-g3"
makeoptionsPROFOPTS="-DPROFILING -DPROFTYPE=4"
makeoptionsLOADADDR="fffffc0000630000"
#
# Special options (see configuring the kernel chapter
# in the Guide to System Administration)
#
cpu                     "ALPHAWILDFIRE"
makeoptionsLOADADDR="ffffffff00000000 -Nu"
machine                 alpha

config                  vmunix      swap generic

bus         isa0    at *
callout after_c "../bin/mkdata isa"


#
# Static Driver Definitions
#
config_driver mchan
config_driver tu
config_driver alt
config_driver ata
config_driver fdi
config_driver lp
config_driver ace
config_driver gpc
config_driver emx
config_driver scsi
config_driver isp
config_driver pci
callout after_c "../bin/mkdata pci"
config_driver wfiop
config_driver wfqbb

#
# Pseudodevice Definitions (see configuring the
# kernel chapter in the Guide to System Administration)
#
pseudo-device           ether
pseudo-device           loop
pseudo-device           lsm         1
pseudo-device           lsm_vmted   0
pseudo-device           rt_hab
pseudo-device           soe_two_hab
pseudo-device           svid_three_hab
pseudo-device           svr_four_hab
pseudo-device           sysv_hab
pseudo-device   prf   6
pseudo-device           ws
```

p_run_1.ora

```
# ----------------------------------------------------
# p_run for node1 of 8 node tpcc
# ----------------------------------------------------
ifile=$ORACLE_HOME/dbs/p_run_common.ora
#_ipc_net = alt8
#_ipc_net = mc0
_ipc_net = lo0
instance_number                                 = 1
thread                                          = 1
rollback_segments  =
(t_0_1,t_0_2,t_0_3,t_0_4,t_0_5,t_0_6,t_0_7,t_0_8,t_0_9,t_0_10,t_0_11,t_0_12,t_0_13
,t_0_14,t_0_15,t_0_16,t_0_17,t_0_18,t_0_19,t_0_20,t_0_21,t_0_22,t_0_23,t_0_24,t_0
_25,t_0_26,t_0_27,t_0_28,t_0_29,
t_0_30,t_0_31,t_0_32,t_0_33,t_0_34,t_0_35,t_0_36,t_0_37,t_0_38,t_0_39,t_0_40,
t_0_41,t_0_42,t_0_43,t_0_44,t_0_45,t_0_46,t_0_47,t_0_48,t_0_49,t_0_50,t_0_51,t_0
_52,t_0_53,t_0_54,t_0_55,t_0_56,t_0_57,t_0_58,t_0_59,t_0_60,t_0_61,t_0_62,
t_0_63,t_0_64,t_0_65,t_0_66,t_0_67,t_0_68,t_0_69,t_0_70,t_0_71,t_0_72,t_0_73,t_0
_74,t_0_75,t_0_76,t_0_77,t_0_78,t_0_79,t_0_80,t_0_81,t_0_82,t_0_83,t_0_84,t_0_85
,t_0_86,t_0_87,t_0_88,t_0_89,t_0_90,t_0_91,t_0_92,t_0_93,t_0_94,t_0_95,t_0_96,t_0
_97,t_0_98,t_0_99,t_0_100)
#rollback_segments  =
(t_0_101,t_0_102,t_0_103,t_0_104,t_0_105,t_0_106,t_0_107,t_0_108,t_0_109,t_0_11
0,t_0_111,t_0_112,t_0_113,t_0_114,t_0_115,t_0_116,t_0_117,t_0_118,t_0_119,t_0_1
```

```
20,t_0_121,t_0_122,t_0_123,t_0_124,t_0_125,t_0_126,t_0_127,t_0_128,t_0_129,
t_0_130)
#rollback_segments =
(t_0_131,t_0_132,t_0_133,t_0_134,t_0_135,t_0_136,t_0_137,t_0_138,t_0_139,t_0_14
0,
t_0_141,t_0_142,t_0_143,t_0_144,t_0_145,t_0_146,t_0_147,t_0_148,t_0_149,t_0_15
0,t_0_151,t_0_152,t_0_153,t_0_154,t_0_155,t_0_156,t_0_157,t_0_158,t_0_159,t_0_1
60,t_0_161,t_0_162,
t_0_163,t_0_164,t_0_165,t_0_166,t_0_167,t_0_168,t_0_169,t_0_170,t_0_171,t_0_17
2,t_0_173,t_0_174,t_0_175,t_0_176,t_0_177,t_0_178,t_0_179,t_0_180,t_0_181,t_0_1
82,t_0_183,t_0_184,t_0_185,t_0_186,t_0_187,t_0_188,t_0_189,t_0_190,t_0_191,t_0_
192,t_0_193,t_0_194,t_0_195,t_0_196,t_0_197,t_0_198,t_0_199,t_0_200)
```

p_run_2.ora

```
# ----------------------------------------------------
# p_run for node2 of 8 node tpcc
# ----------------------------------------------------
ifile=$ORACLE_HOME/dbs/p_run_common.ora
#_ipc_net = alt9
_ipc_net = lo0
instance_number                                 = 2
thread                                          = 2
rollback_segments =
(t_1_1,t_1_2,t_1_3,t_1_4,t_1_5,t_1_6,t_1_7,t_1_8,t_1_9,t_1_10,t_1_11,t_1_12,t_1_13
,t_1_14,t_1_15,t_1_16,t_1_17,t_1_18,t_1_19,t_1_20,t_1_21,t_1_22,t_1_23,t_1_24,t_1
_25,t_1_26,t_1_27,t_1_28,t_1_29,
t_1_30,t_1_31,t_1_32,t_1_33,t_1_34,t_1_35,t_1_36,t_1_37,t_1_38,t_1_39,t_1_40,
t_1_41,t_1_42,t_1_43,t_1_44,t_1_45,t_1_46,t_1_47,t_1_48,t_1_49,t_1_50,t_1_51,t_1
_52,t_1_53,t_1_54,t_1_55,t_1_56,t_1_57,t_1_58,t_1_59,t_1_60,t_1_61,t_1_62,
t_1_63,t_1_64,t_1_65,t_1_66,t_1_67,t_1_68,t_1_69,t_1_70,t_1_71,t_1_72,t_1_73,t_1
_74,t_1_75,t_1_76,t_1_77,t_1_78,t_1_79,t_1_80,t_1_81,t_1_82,t_1_83,t_1_84,t_1_85
,t_1_86,t_1_87,t_1_88,t_1_89,t_1_90,t_1_91,t_1_92,t_1_93,t_1_94,t_1_95,t_1_96,t_1
_97,t_1_98,t_1_99,t_1_100)
#rollback_segments =
(t_1_101,t_1_102,t_1_103,t_1_104,t_1_105,t_1_106,t_1_107,t_1_108,t_1_109,t_1_11
0,t_1_111,t_1_112,t_1_113,t_1_114,t_1_115,t_1_116,t_1_117,t_1_118,t_1_119,t_1_1
20,t_1_121,t_1_122,t_1_123,t_1_124,t_1_125,t_1_126,t_1_127,t_1_128,t_1_129,
t_1_130)
#rollback_segments =
(t_1_131,t_1_132,t_1_133,t_1_134,t_1_135,t_1_136,t_1_137,t_1_138,t_1_139,t_1_14
0,
t_1_141,t_1_142,t_1_143,t_1_144,t_1_145,t_1_146,t_1_147,t_1_148,t_1_149,t_1_15
0,t_1_151,t_1_152,t_1_153,t_1_154,t_1_155,t_1_156,t_1_157,t_1_158,t_1_159,t_1_1
60,t_1_161,t_1_162,
t_1_163,t_1_164,t_1_165,t_1_166,t_1_167,t_1_168,t_1_169,t_1_170,t_1_171,t_1_17
2,t_1_173,t_1_174,t_1_175,t_1_176,t_1_177,t_1_178,t_1_179,t_1_180,t_1_181,t_1_1
82,t_1_183,t_1_184,t_1_185,t_1_186,t_1_187,t_1_188,t_1_189,t_1_190,t_1_191,t_1_
192,t_1_193,t_1_194,t_1_195,t_1_196,t_1_197,t_1_198,t_1_199,t_1_200)
```

p_run_3.ora

```
# ----------------------------------------------------
# p_run for node3 of 8 node tpcc
# ----------------------------------------------------
ifile=$ORACLE_HOME/dbs/p_run_common.ora
#_ipc_net = alt10
_ipc_net = lo0
instance_number                                 = 3
thread                                          = 3
rollback_segments =
(t_2_1,t_2_2,t_2_3,t_2_4,t_2_5,t_2_6,t_2_7,t_2_8,t_2_9,t_2_10,t_2_11,t_2_12,t_2_13
,t_2_14,t_2_15,t_2_16,t_2_17,t_2_18,t_2_19,t_2_20,t_2_21,t_2_22,t_2_23,t_2_24,t_2
_25,t_2_26,t_2_27,t_2_28,t_2_29,
t_2_30,t_2_31,t_2_32,t_2_33,t_2_34,t_2_35,t_2_36,t_2_37,t_2_38,t_2_39,t_2_40,
t_2_41,t_2_42,t_2_43,t_2_44,t_2_45,t_2_46,t_2_47,t_2_48,t_2_49,t_2_50,t_2_51,t_2
_52,t_2_53,t_2_54,t_2_55,t_2_56,t_2_57,t_2_58,t_2_59,t_2_60,t_2_61,t_2_62,
t_2_63,t_2_64,t_2_65,t_2_66,t_2_67,t_2_68,t_2_69,t_2_70,t_2_71,t_2_72,t_2_73,t_2
_74,t_2_75,t_2_76,t_2_77,t_2_78,t_2_79,t_2_80,t_2_81,t_2_82,t_2_83,t_2_84,t_2_85
,t_2_86,t_2_87,t_2_88,t_2_89,t_2_90,t_2_91,t_2_92,t_2_93,t_2_94,t_2_95,t_2_96,t_2
_97,t_2_98,t_2_99,t_2_100)
#rollback_segments =
(t_2_101,t_2_102,t_2_103,t_2_104,t_2_105,t_2_106,t_2_107,t_2_108,t_2_109,t_2_11
0,t_2_111,t_2_112,t_2_113,t_2_114,t_2_115,t_2_116,t_2_117,t_2_118,t_2_119,t_2_1
20,t_2_121,t_2_122,t_2_123,t_2_124,t_2_125,t_2_126,t_2_127,t_2_128,t_2_129,
t_2_130)
#rollback_segments =
(t_2_131,t_2_132,t_2_133,t_2_134,t_2_135,t_2_136,t_2_137,t_2_138,t_2_139,t_2_14
0,
t_2_141,t_2_142,t_2_143,t_2_144,t_2_145,t_2_146,t_2_147,t_2_148,t_2_149,t_2_15
0,t_2_151,t_2_152,t_2_153,t_2_154,t_2_155,t_2_156,t_2_157,t_2_158,t_2_159,t_2_1
60,t_2_161,t_2_162,
t_2_163,t_2_164,t_2_165,t_2_166,t_2_167,t_2_168,t_2_169,t_2_170,t_2_171,t_2_17
2,t_2_173,t_2_174,t_2_175,t_2_176,t_2_177,t_2_178,t_2_179,t_2_180,t_2_181,t_2_1
82,t_2_183,t_2_184,t_2_185,t_2_186,t_2_187,t_2_188,t_2_189,t_2_190,t_2_191,t_2_
192,t_2_193,t_2_194,t_2_195,t_2_196,t_2_197,t_2_198,t_2_199,t_2_200)
```

p_run_4.ora

```
# ----------------------------------------------------
# p_run for node4 of 8 node tpcc
# ----------------------------------------------------
ifile=$ORACLE_HOME/dbs/p_run_common.ora
#_ipc_net = alt11
_ipc_net = lo0
instance_number                                 = 4
thread                                          = 4
rollback_segments =
(t_3_1,t_3_2,t_3_3,t_3_4,t_3_5,t_3_6,t_3_7,t_3_8,t_3_9,t_3_10,t_3_11,t_3_12,t_3_13
,t_3_14,t_3_15,t_3_16,t_3_17,t_3_18,t_3_19,t_3_20,t_3_21,t_3_22,t_3_23,t_3_24,t_3
_25,t_3_26,t_3_27,t_3_28,t_3_29,
t_3_30,t_3_31,t_3_32,t_3_33,t_3_34,t_3_35,t_3_36,t_3_37,t_3_38,t_3_39,t_3_40,
t_3_41,t_3_42,t_3_43,t_3_44,t_3_45,t_3_46,t_3_47,t_3_48,t_3_49,t_3_50,t_3_51,t_3
_52,t_3_53,t_3_54,t_3_55,t_3_56,t_3_57,t_3_58,t_3_59,t_3_60,t_3_61,t_3_62,
t_3_63,t_3_64,t_3_65,t_3_66,t_3_67,t_3_68,t_3_69,t_3_70,t_3_71,t_3_72,t_3_73,t_3
_74,t_3_75,t_3_76,t_3_77,t_3_78,t_3_79,t_3_80,t_3_81,t_3_82,t_3_83,t_3_84,t_3_85
,t_3_86,t_3_87,t_3_88,t_3_89,t_3_90,t_3_91,t_3_92,t_3_93,t_3_94,t_3_95,t_3_96,t_3
_97,t_3_98,t_3_99,t_3_100)
#rollback_segments =
(t_3_101,t_3_102,t_3_103,t_3_104,t_3_105,t_3_106,t_3_107,t_3_108,t_3_109,t_3_11
0,t_3_111,t_3_112,t_3_113,t_3_114,t_3_115,t_3_116,t_3_117,t_3_118,t_3_119,t_3_1
20,t_3_121,t_3_122,t_3_123,t_3_124,t_3_125,t_3_126,t_3_127,t_3_128,t_3_129,
t_3_130)
#rollback_segments =
(t_3_131,t_3_132,t_3_133,t_3_134,t_3_135,t_3_136,t_3_137,t_3_138,t_3_139,t_3_14
0,
t_3_141,t_3_142,t_3_143,t_3_144,t_3_145,t_3_146,t_3_147,t_3_148,t_3_149,t_3_15
0,t_3_151,t_3_152,t_3_153,t_3_154,t_3_155,t_3_156,t_3_157,t_3_158,t_3_159,t_3_1
60,t_3_161,t_3_162,
t_3_163,t_3_164,t_3_165,t_3_166,t_3_167,t_3_168,t_3_169,t_3_170,t_3_171,t_3_17
2,t_3_173,t_3_174,t_3_175,t_3_176,t_3_177,t_3_178,t_3_179,t_3_180,t_3_181,t_3_1
82,t_3_183,t_3_184,t_3_185,t_3_186,t_3_187,t_3_188,t_3_189,t_3_190,t_3_191,t_3_
192,t_3_193,t_3_194,t_3_195,t_3_196,t_3_197,t_3_198,t_3_199,t_3_200)
```

p_run_5.ora

```
# ----------------------------------------------------
# p_run for node6 of 8 node tpcc
# ----------------------------------------------------
ifile=$ORACLE_HOME/dbs/p_run_common.ora
#_ipc_net = alt12
_ipc_net = lo0
instance_number                                 = 5
thread                                          = 5
rollback_segments =
(t_4_1,t_4_2,t_4_3,t_4_4,t_4_5,t_4_6,t_4_7,t_4_8,t_4_9,t_4_10,t_4_11,t_4_12,t_4_13
,t_4_14,t_4_15,t_4_16,t_4_17,t_4_18,t_4_19,t_4_20,t_4_21,t_4_22,t_4_23,t_4_24,t_4
_25,t_4_26,t_4_27,t_4_28,t_4_29,
t_4_30,t_4_31,t_4_32,t_4_33,t_4_34,t_4_35,t_4_36,t_4_37,t_4_38,t_4_39,t_4_40,
t_4_41,t_4_42,t_4_43,t_4_44,t_4_45,t_4_46,t_4_47,t_4_48,t_4_49,t_4_50,t_4_51,t_4
_52,t_4_53,t_4_54,t_4_55,t_4_56,t_4_57,t_4_58,t_4_59,t_4_60,t_4_61,t_4_62,
t_4_63,t_4_64,t_4_65,t_4_66,t_4_67,t_4_68,t_4_69,t_4_70,t_4_71,t_4_72,t_4_73,t_4
_74,t_4_75,t_4_76,t_4_77,t_4_78,t_4_79,t_4_80,t_4_81,t_4_82,t_4_83,t_4_84,t_4_85
,t_4_86,t_4_87,t_4_88,t_4_89,t_4_90,t_4_91,t_4_92,t_4_93,t_4_94,t_4_95,t_4_96,t_4
_97,t_4_98,t_4_99,t_4_100)
```

```
#rollback_segments  =
(t_4_101,t_4_102,t_4_103,t_4_104,t_4_105,t_4_106,t_4_107,t_4_108,t_4_109,t_4_11
0,t_4_111,t_4_112,t_4_113,t_4_114,t_4_115,t_4_116,t_4_117,t_4_118,t_4_119,t_4_1
20,t_4_121,t_4_122,t_4_123,t_4_124,t_4_125,t_4_126,t_4_127,t_4_128,t_4_129,
t_4_130)
#rollback_segments  =
(t_4_131,t_4_132,t_4_133,t_4_134,t_4_135,t_4_136,t_4_137,t_4_138,t_4_139,t_4_14
0,
t_4_141,t_4_142,t_4_143,t_4_144,t_4_145,t_4_146,t_4_147,t_4_148,t_4_149,t_4_15
0,t_4_151,t_4_152,t_4_153,t_4_154,t_4_155,t_4_156,t_4_157,t_4_158,t_4_159,t_4_1
60,t_4_161,t_4_162,
t_4_163,t_4_164,t_4_165,t_4_166,t_4_167,t_4_168,t_4_169,t_4_170,t_4_171,t_4_17
2,t_4_173,t_4_174,t_4_175,t_4_176,t_4_177,t_4_178,t_4_179,t_4_180,t_4_181,t_4_1
82,t_4_183,t_4_184,t_4_185,t_4_186,t_4_187,t_4_188,t_4_189,t_4_190,t_4_191,t_4_
192,t_4_193,t_4_194,t_4_195,t_4_196,t_4_197,t_4_198,t_4_199,t_4_200)
```

p_run_6.ora

```
# ----------------------------------------------------
# p_run for node6 of 8 node tpcc
# ----------------------------------------------------
ifile=$ORACLE_HOME/dbs/p_run_common.ora
#_ipc_net = alt13
_ipc_net = lo0
instance_number                                 = 6
thread                                          = 6
rollback_segments  =
(t_5_1,t_5_2,t_5_3,t_5_4,t_5_5,t_5_6,t_5_7,t_5_8,t_5_9,t_5_10,t_5_11,t_5_12,t_5_13
,t_5_14,t_5_15,t_5_16,t_5_17,t_5_18,t_5_19,t_5_20,t_5_21,t_5_22,t_5_23,t_5_24,t_5
_25,t_5_26,t_5_27,t_5_28,t_5_29,
t_5_30,t_5_31,t_5_32,t_5_33,t_5_34,t_5_35,t_5_36,t_5_37,t_5_38,t_5_39,t_5_40,
t_5_41,t_5_42,t_5_43,t_5_44,t_5_45,t_5_46,t_5_47,t_5_48,t_5_49,t_5_50,t_5_51,t_5
_52,t_5_53,t_5_54,t_5_55,t_5_56,t_5_57,t_5_58,t_5_59,t_5_60,t_5_61,t_5_62,
t_5_63,t_5_64,t_5_65,t_5_66,t_5_67,t_5_68,t_5_69,t_5_70,t_5_71,t_5_72,t_5_73,t_5
_74,t_5_75,t_5_76,t_5_77,t_5_78,t_5_79,t_5_80,t_5_81,t_5_82,t_5_83,t_5_84,t_5_85
,t_5_86,t_5_87,t_5_88,t_5_89,t_5_90,t_5_91,t_5_92,t_5_93,t_5_94,t_5_95,t_5_96,t_5
_97,t_5_98,t_5_99,t_5_100)
#rollback_segments  =
(t_5_101,t_5_102,t_5_103,t_5_104,t_5_105,t_5_106,t_5_107,t_5_108,t_5_109,t_5_11
0,t_5_111,t_5_112,t_5_113,t_5_114,t_5_115,t_5_116,t_5_117,t_5_118,t_5_119,t_5_1
20,t_5_121,t_5_122,t_5_123,t_5_124,t_5_125,t_5_126,t_5_127,t_5_128,t_5_129,
t_5_130)
#rollback_segments  =
(t_5_131,t_5_132,t_5_133,t_5_134,t_5_135,t_5_136,t_5_137,t_5_138,t_5_139,t_5_14
0,
t_5_141,t_5_142,t_5_143,t_5_144,t_5_145,t_5_146,t_5_147,t_5_148,t_5_149,t_5_15
0,t_5_151,t_5_152,t_5_153,t_5_154,t_5_155,t_5_156,t_5_157,t_5_158,t_5_159,t_5_1
60,t_5_161,t_5_162,
t_5_163,t_5_164,t_5_165,t_5_166,t_5_167,t_5_168,t_5_169,t_5_170,t_5_171,t_5_17
2,t_5_173,t_5_174,t_5_175,t_5_176,t_5_177,t_5_178,t_5_179,t_5_180,t_5_181,t_5_1
82,t_5_183,t_5_184,t_5_185,t_5_186,t_5_187,t_5_188,t_5_189,t_5_190,t_5_191,t_5_
192,t_5_193,t_5_194,t_5_195,t_5_196,t_5_197,t_5_198,t_5_199,t_5_200)
```

p_run_7.ora

```
# ----------------------------------------------------
# p_run for node7 of 8 node tpcc
# ----------------------------------------------------
ifile=$ORACLE_HOME/dbs/p_run_common.ora
#_ipc_net = alt14
_ipc_net = lo0
instance_number                                 = 7
thread                                          = 7
rollback_segments  =
(t_6_1,t_6_2,t_6_3,t_6_4,t_6_5,t_6_6,t_6_7,t_6_8,t_6_9,t_6_10,t_6_11,t_6_12,t_6_13
,t_6_14,t_6_15,t_6_16,t_6_17,t_6_18,t_6_19,t_6_20,t_6_21,t_6_22,t_6_23,t_6_24,t_6
_25,t_6_26,t_6_27,t_6_28,t_6_29,
t_6_30,t_6_31,t_6_32,t_6_33,t_6_34,t_6_35,t_6_36,t_6_37,t_6_38,t_6_39,t_6_40,
t_6_41,t_6_42,t_6_43,t_6_44,t_6_45,t_6_46,t_6_47,t_6_48,t_6_49,t_6_50,t_6_51,t_6
_52,t_6_53,t_6_54,t_6_55,t_6_56,t_6_57,t_6_58,t_6_59,t_6_60,t_6_61,t_6_62,
t_6_63,t_6_64,t_6_65,t_6_66,t_6_67,t_6_68,t_6_69,t_6_70,t_6_71,t_6_72,t_6_73,t_6
_74,t_6_75,t_6_76,t_6_77,t_6_78,t_6_79,t_6_80,t_6_81,t_6_82,t_6_83,t_6_84,t_6_85
,t_6_86,t_6_87,t_6_88,t_6_89,t_6_90,t_6_91,t_6_92,t_6_93,t_6_94,t_6_95,t_6_96,t_6
_97,t_6_98,t_6_99,t_6_100)
#rollback_segments  =
(t_6_101,t_6_102,t_6_103,t_6_104,t_6_105,t_6_106,t_6_107,t_6_108,t_6_109,t_6_11
0,t_6_111,t_6_112,t_6_113,t_6_114,t_6_115,t_6_116,t_6_117,t_6_118,t_6_119,t_6_1
20,t_6_121,t_6_122,t_6_123,t_6_124,t_6_125,t_6_126,t_6_127,t_6_128,t_6_129,
t_6_130)
#rollback_segments  =
(t_6_131,t_6_132,t_6_133,t_6_134,t_6_135,t_6_136,t_6_137,t_6_138,t_6_139,t_6_14
0,
t_6_141,t_6_142,t_6_143,t_6_144,t_6_145,t_6_146,t_6_147,t_6_148,t_6_149,t_6_15
0,t_6_151,t_6_152,t_6_153,t_6_154,t_6_155,t_6_156,t_6_157,t_6_158,t_6_159,t_6_1
60,t_6_161,t_6_162,
t_6_163,t_6_164,t_6_165,t_6_166,t_6_167,t_6_168,t_6_169,t_6_170,t_6_171,t_6_17
2,t_6_173,t_6_174,t_6_175,t_6_176,t_6_177,t_6_178,t_6_179,t_6_180,t_6_181,t_6_1
82,t_6_183,t_6_184,t_6_185,t_6_186,t_6_187,t_6_188,t_6_189,t_6_190,t_6_191,t_6_
192,t_6_193,t_6_194,t_6_195,t_6_196,t_6_197,t_6_198,t_6_199,t_6_200)
```

p_run_8.ora

```
# ----------------------------------------------------
# p_run for node8 of 8 node tpcc
# ----------------------------------------------------
ifile=$ORACLE_HOME/dbs/p_run_common.ora
#_ipc_net = alt15
_ipc_net = lo0
instance_number                                 = 8
thread                                          = 8
rollback_segments  =
(t_7_1,t_7_2,t_7_3,t_7_4,t_7_5,t_7_6,t_7_7,t_7_8,t_7_9,t_7_10,t_7_11,t_7_12,t_7_13
,t_7_14,t_7_15,t_7_16,t_7_17,t_7_18,t_7_19,t_7_20,t_7_21,t_7_22,t_7_23,t_7_24,t_7
_25,t_7_26,t_7_27,t_7_28,t_7_29,
t_7_30,t_7_31,t_7_32,t_7_33,t_7_34,t_7_35,t_7_36,t_7_37,t_7_38,t_7_39,t_7_40,
t_7_41,t_7_42,t_7_43,t_7_44,t_7_45,t_7_46,t_7_47,t_7_48,t_7_49,t_7_50,t_7_51,t_7
_52,t_7_53,t_7_54,t_7_55,t_7_56,t_7_57,t_7_58,t_7_59,t_7_60,t_7_61,t_7_62,
t_7_63,t_7_64,t_7_65,t_7_66,t_7_67,t_7_68,t_7_69,t_7_70,t_7_71,t_7_72,t_7_73,t_7
_74,t_7_75,t_7_76,t_7_77,t_7_78,t_7_79,t_7_80,t_7_81,t_7_82,t_7_83,t_7_84,t_7_85
,t_7_86,t_7_87,t_7_88,t_7_89,t_7_90,t_7_91,t_7_92,t_7_93,t_7_94,t_7_95,t_7_96,t_7
_97,t_7_98,t_7_99,t_7_100)
#rollback_segments  =
(t_7_101,t_7_102,t_7_103,t_7_104,t_7_105,t_7_106,t_7_107,t_7_108,t_7_109,t_7_11
0,t_7_111,t_7_112,t_7_113,t_7_114,t_7_115,t_7_116,t_7_117,t_7_118,t_7_119,t_7_1
20,t_7_121,t_7_122,t_7_123,t_7_124,t_7_125,t_7_126,t_7_127,t_7_128,t_7_129,
t_7_130)
#rollback_segments  =
(t_7_131,t_7_132,t_7_133,t_7_134,t_7_135,t_7_136,t_7_137,t_7_138,t_7_139,t_7_14
0,
t_7_141,t_7_142,t_7_143,t_7_144,t_7_145,t_7_146,t_7_147,t_7_148,t_7_149,t_7_15
0,t_7_151,t_7_152,t_7_153,t_7_154,t_7_155,t_7_156,t_7_157,t_7_158,t_7_159,t_7_1
60,t_7_161,t_7_162,
t_7_163,t_7_164,t_7_165,t_7_166,t_7_167,t_7_168,t_7_169,t_7_170,t_7_171,t_7_17
2,t_7_173,t_7_174,t_7_175,t_7_176,t_7_177,t_7_178,t_7_179,t_7_180,t_7_181,t_7_1
82,t_7_183,t_7_184,t_7_185,t_7_186,t_7_187,t_7_188,t_7_189,t_7_190,t_7_191,t_7_
192,t_7_193,t_7_194,t_7_195,t_7_196,t_7_197,t_7_198,t_7_199,t_7_200)
```

p_run_common.ora

```
#=============================================================
=====
# FILENAME
#     p_run_common.ora
# DESCRIPTION
#     Oracle parameter file for running TPC-C in OPS mode.
#=============================================================
=====
control_files               = (?/dbs/tpcc_disks/control_001)
compatible                  = 8.1.5
db_name                     = tpcc
db_files                    = 748
timed_statistics                        = FALSE
db_writer_processes         = 1
db_block_lru_latches        = 18
cpu_count                   = 4
```

```
parallel_server                 = TRUE
db_block_buffers                = 5491744 # 10.47gig
buffer_pool_recycle             = (buffers:13180,lru_latches:5)
buffer_pool_keep                =(buffers:3844220,lru_latches:5)
log_buffer                      = 8192000
enqueue_resources               = 600
_db_writer_max_writes           = 256 # 640
_db_writer_chunk_writes         = 64  # 100
db_block_max_dirty_target       = 0
fast_start_io_target            = 0
shared_pool_size                = 200000000
shared_pool_reserved_size       = 2500000
cursor_space_for_time           = TRUE
max_dump_file_size                      = 900000
db_block_checking               = FALSE
replication_dependency_tracking         = FALSE
transaction_auditing                    = FALSE
recovery_parallelism                    = 80
parallel_max_servers            = 80
dml_locks                       = 300
hash_join_enabled               = FALSE
log_archive_start               = FALSE
log_checkpoint_timeout          = 0
log_checkpoint_interval         = 0
log_checkpoints_to_alert        = TRUE
sort_area_size                  = 1048576
hash_area_size                  = 1048576
hash_multiblock_io_count        = 1
open_cursors                    = 140
processes                       = 140
db_file_multiblock_read_count           = 1
_db_file_noncontig_mblock_read_count=1
distributed_transactions        = 0
max_rollback_segments           = 140
lm_ress                         = 1431024
lm_locks                        = 2660000
gc_releasable_locks             = 5000
gc_rollback_locks       = "0-1631=50EACH"
gc_files_to_locks       = "1=3000EACH:"  # system
# cust 7000
gc_files_to_locks       = "9,91-97=56000:105-112=56000:123-130=56000:"
gc_files_to_locks       = "131-133,138-142=56000:149-156=56000:"
gc_files_to_locks       = "252-259=56000:270-277=56000:278,289-295=56000:296=1:"
# stock 12000
gc_files_to_locks       = "88-90,98-104=120000:113=12000:114-
122=96000:134=12000:"
gc_files_to_locks       = "135-137,143-148=96000:"
gc_files_to_locks       = "157=12000:158-159,200,246-251=96000:"
gc_files_to_locks       = "260=12000:261-269=96000:279=12000:280-288=96000:"
gc_files_to_locks       = "297=12000:298-306=96000:307=12000:308-
316=96000:317=8:"
# all others ************
gc_files_to_locks       = "2-8,10-87,160-199,201-245=8EACH:" # all others
gc_files_to_locks       = "327,328=20EACH" # temp
gc_files_to_locks       = "326=20EACH" # temperm2 bench tables
#
java_pool_size          = 4K
mts_max_servers         = 0
_db_aging_stay_count    = 1
_spin_count                     = 3000
_lm_rcv_buffer_size             = 1048576
_interconnect_checksum          = FALSE
_lm_cache_res_cleanup           = 100
#_lm_direct_sends                = lkmgr  # needed for tcp version
#_lm_sync_timeout                = 120000  # in centi secs
```

sysconfigtab

```
#
# ****************************************************************
# *                                                              *
# *    Copyright Compaq Computer Corporation, 2000               *
# *                                                              *
# *   The software contained on this media  is  proprietary  to  *
# *   and embodies the confidential technology of Compaq   *
# *   Computer Corporation.  Possession, use,  duplication  or   *
# *   dissemination of the software and media is authorized only  *
# *   pursuant to a valid written license from Compaq Computer    *
# *   Corporation.                                               *
# *                                                              *
# *   RESTRICTED RIGHTS LEGEND   Use, duplication, or disclosure  *
# *   by the U.S. Government is subject to restrictions  as  set  *
# *   forth in Subparagraph (c)(1)(ii)  of  DFARS  252.227-7013,  *
# *   or  in  FAR 52.227-19, as applicable.                      *
# *                                                              *
# ****************************************************************
#
# HISTORY
#
# (c) Copyright 1990, 1991, 1992, 1993 OPEN SOFTWARE FOUNDATION, INC.
# ALL RIGHTS RESERVED
#
#
# OSF/1 1.2

#
#
# The supported method of changing the information in this
# file (sysconfigtab) is by using the sysconfigdb command.
#
# If you insist upon changing this file by direct editing,
# then it would be advisable to add new information at the end
# rather than the beginning or middle of the file, in order
# to reduce confusion during installation and reconciliation
# of user changes and update changes.
#
# See the appropriate documentation such as the release notes
# for the new version being installed as well as the appropriate
# reference pages.
#

#
#   Contiguous Memory Allocation for loadable driver subsystems
#
cma_dd:



#
#   EISA_Option = Board_Id Function_Name Driver_Name Type
#
eisa:
#
# %%%EISA
#
  EISA_Option = Board_Id - ADP0001, Function_Name - AHA1740, Driver_Name -
aha, Int_Aft_Attach - 1
  EISA_Option = Board_Id - ADP0002, Function_Name - AHA1740, Driver_Name -
aha, Int_Aft_Attach - 1
  EISA_Option = Board_Id - DEC2500, Driver_Name - envram
  EISA_Option = Board_Id - ISA1010, Function_Name  - 'COM,1' Driver_Name - ace
  EISA_Option = Board_Id - ISA1010, Function_Name - 'COM,2', Driver_Name - ace
  EISA_Option = Board_Id - ISA1010, Function_Name - PAR, Driver_Name - lp
  EISA_Option = Board_Id - DEC2A01, Function_Name - SYSMEM, Int_Aft_Attach
- 1
  EISA_Option = Board_Id - DEC2A01, Function_Name - 'ACECOM,1',
Driver_Name - ace
  EISA_Option = Board_Id - DEC2A01, Function_Name - 'ACECOM,2',
Driver_Name - ace
  EISA_Option = Board_Id - DEC2A01, Function_Name - PAR, Driver_Name - lp
  EISA_Option = Board_Id - MLX0070, Driver_Name - xcr, Int_Aft_Attach - 1
  EISA_Option = Board_Id - MLX0075, Driver_Name - xcr,  Int_Aft_Attach - 1
  EISA_Option = Board_Id - MLX0077, Driver_Name - xcr,  Int_Aft_Attach - 1
  EISA_Option = Board_Id - DEC5000, Function_Name - SYSMEM, Int_Aft_Attach
- 1
  EISA_Option = Board_Id - DEC5000, Function_Name - 'ACECOM,1',
Driver_Name - ace
  EISA_Option = Board_Id - DEC5000, Function_Name - 'ACECOM,2',
Driver_Name - ace
  EISA_Option = Board_Id - DEC5000, Function_Name - PAR, Driver_Name - lp
  EISA_Option = Board_Id - DEC5000, Function_Name - 'KBD,MOUSE',
Driver_Name - gpc
  EISA_Option = Board_Id - DEC5100, Function_Name - SYSMEM, Int_Aft_Attach
- 1
  EISA_Option = Board_Id - DEC5100, Function_Name - 'ACECOM,1',
Driver_Name - ace
```

```
  EISA_Option = Board_Id - DEC5100, Function_Name - 'ACECOM,2',
Driver_Name - ace
  EISA_Option = Board_Id - DEC5100, Function_Name - PAR, Driver_Name - lp
  EISA_Option = Board_Id - DEC5100, Function_Name - 'KBD,MOUSE',
Driver_Name - gpc
  EISA_Option = Board_Id - DEC5301, Function_Name - SYSMEM, Int_Aft_Attach
- 1
  EISA_Option = Board_Id - DEC5301, Function_Name - 'ACECOM,1',
Driver_Name - ace
  EISA_Option = Board_Id - DEC5301, Function_Name - 'ACECOM,2',
Driver_Name - ace
  EISA_Option = Board_Id - DEC5301, Function_Name - PAR, Driver_Name - lp
  EISA_Option = Board_Id - DEC5301, Function_Name - 'KBD,MOUSE',
Driver_Name - gpc
  EISA_Option = Board_Id - DEC6000, Function_Name - 'SYSMEM', Int_Aft_Attach
- 1
  EISA_Option = Board_Id - DEC6000, Function_Name - 'KBD,MOUSE',
Driver_Name - gpc
  EISA_Option = Board_Id - DEC6400, Function_Name - SYSMEM, Driver_Name -
Null
  EISA_Option = Board_Id - DEC6400, Function_Name - 'ACECOM,1',
Driver_Name - ace
  EISA_Option = Board_Id - DEC6400, Function_Name - 'ACECOM,2',
Driver_Name - ace
  EISA_Option = Board_Id - DEC6400, Function_Name - PAR, Driver_Name - lp
  EISA_Option = Board_Id - DEC6400, Function_Name - 'KBD,MOUSE',
Driver_Name - gpc
#
#   ISA_Option = Board_Id Function_Name Driver_Name Type
#
isa:
#
# %%%ISA
#
  ISA_Option =  Function_Name - 'KBD,MOUSE', Driver_Name - gpc
  ISA_Option =  Function_Name - COM, Driver_Name - ace
  ISA_Option =  Function_Name - LPT, Driver_Name - lp
  ISA_Option =  Function_Name - 'ISA--SCC', Driver_Name - iscc



#
# Pcisw_Version Vendor_Id Ddevice_Id Rev Base Sub Pif Sub_Vid Sub_Did
Vid_Mo_Flag Did_Mo_Flag
# Rev_Mo_Flag Base_Mo_Flag Sub_Mo_Flag Pif_Mo_Flag Sub_Vid_Mo_Flag
Driver_Name Type
#
pci:
#
# %%%PCI
#

#
#  PCI to something bridge adapters
#
#  At present these are not used in system configuration, but are present in
#  this file for informational purposes.  In some cases, the "driver" may
#  not exist, and in other cases the driver may exist but be unprepared to
#  parse these entries, so they are all commented out at this time.

#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1013, Device_Id - 0x1100,
Driver_Name - pcmcia, Type - A, Adpt_Config - N
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1011, Device_Id - 0x1, Base -
6, Sub - 4, Driver_Name - pci, Type - A, Adpt_Config - N
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1011, Base - 6, Sub - 4,
Driver_Name - pci, Type - A, Adpt_Config - N
#   PCI_Option =  PCI_SE_Rev - 0x210, Base - 6, Sub - 4, Driver_Name - pci, Type -
A, Adpt_Config - N
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x8086, Device_Id - 0x482,
Driver_Name - eisa, Type - A, Adpt_Config - N
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x8086, Device_Id - 0x484,
Driver_Name - isa, Type - A, Adpt_Config - N
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1011, Device_Id - 0x10,
Driver_Name - vba, Type - A, Adpt_Config - N
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x10E3, Device_Id - 0x00,
Driver_Name - vba, Type - A, Adpt_Config - N



#
#  TC_Option = Modname Driver_Name
#
tc:
#
# %%%TC
#
   TC_Option =  Modname - 'PMTNV-AA', Driver_Name - nvtc, Int_Aft_Attach - 1,
Type - C, Adpt_Config - N
   TC_Option =  Modname - 'PMAP-AA ', Driver_Name - nvtc, Int_Aft_Attach - 1,
Type - C, Adpt_Config - N
   TC_Option =  Modname - 'PMAZ-AA ', Driver_Name - asc, Int_Aft_Attach - 1,
Type - A, Adpt_Config - N
   TC_Option =  Modname - 'PMAZ-DS ', Driver_Name - tcds, Int_Aft_Attach - 1,
Type - A, Adpt_Config - N
   TC_Option =  Modname - 'PMAZ-FS ', Driver_Name - tcds, Int_Aft_Attach - 1,
Type - A, Adpt_Config - N
   TC_Option =  Modname - 'PMAZB-AA', Driver_Name - tcds, Int_Aft_Attach - 1,
Type - A, Adpt_Config - N
   TC_Option =  Modname - 'PMAZB-AB', Driver_Name - tcds, Int_Aft_Attach - 1,
Type - A, Adpt_Config - N
   TC_Option =  Modname - 'PMAZC-AA', Driver_Name - tcds, Int_Aft_Attach - 1,
Type - A, Adpt_Config - N
   TC_Option =  Modname - 'KZTSA-AA', Driver_Name - tza, Int_Aft_Attach - 1,
Type - A, Adpt_Config - N
   TC_Option =  Modname - 'PMAGB-BA', Driver_Name - fb, Int_Aft_Attach - 1,
Type - C, Adpt_Config - N
   TC_Option =  Modname - 'PMAG-RO ', Driver_Name - fb, Type - C, Adpt_Config -
N
   TC_Option =  Modname - 'PMAG-JA ', Driver_Name - fb, Type - C, Adpt_Config -
N
   TC_Option =  Modname - 'PMADC   ', Driver_Name - pv, Int_Aft_Attach - 1, Type
- C, Adpt_Config - N
   TC_Option =  Modname - 'PMAGC-AA', Driver_Name - pv, Int_Aft_Attach - 1,
Type - C, Adpt_Config - N
   TC_Option =  Modname - 'PMAGC-BA', Driver_Name - pv, Int_Aft_Attach - 1,
Type - C, Adpt_Config - N
   TC_Option =  Modname - 'PMAGC-DA', Driver_Name - pvl, Int_Aft_Attach - 1,
Type - C, Adpt_Config - N
   TC_Option =  Modname - 'PMAGC-EA', Driver_Name - pvl, Int_Aft_Attach - 1,
Type - C, Adpt_Config - N
   TC_Option =  Modname - 'PMAGD   ', Driver_Name - fb, Int_Aft_Attach - 1, Type
- C, Adpt_Config - N
   TC_Option =  Modname - 'PMAGD-AA', Driver_Name - fb, Int_Aft_Attach - 1,
Type - C, Adpt_Config - N
   TC_Option =  Modname - 'PMAGD-BA', Driver_Name - fb, Int_Aft_Attach - 1,
Type - C, Adpt_Config - N
# Disabled due to lack of X support in Platinum, device driver is being pulled from the
release
#   TC_Option =  Modname - "KWS_TD", Confname - kws_td, Int_Aft_Attach- 1,
Type - C, Adpt_Config - N

#
# fdi: FLOPPY
#
fdi:
  ISA_Option =  Function_Name - FLOPPY, Driver_Name - fdi
  EISA_Option = Board_Id - ADP0002, Function_Name - 'MSD,FPYCTL',
Driver_Name - fdi
  EISA_Option = Board_Id - DEC2400, Function_Name - 'MSD,FPYCTL',
Driver_Name - fdi
  EISA_Option = Board_Id - DEC2A01, Function_Name - 'MSD,FPYCTL',
Driver_Name - fdi
  EISA_Option = Board_Id - DEC5000, Function_Name - 'MSD,FPYCTL',
Driver_Name - fdi
  EISA_Option = Board_Id - DEC5100, Function_Name - 'MSD,FPYCTL',
Driver_Name - fdi
  EISA_Option = Board_Id - DEC5301, Function_Name - 'MSD,FPYCTL',
Driver_Name - fdi
  EISA_Option = Board_Id - DEC6000, Function_Name - 'MSD,FPYCTL',
Driver_Name - fdi

#
# ln: LANCE Ethernet
#
ln:
  TC_Option =  Modname - 'PMAD-AA ', Driver_Name - ln, Type - C, Adpt_Config -
N
  TC_Option =  Modname - 'PMAD-BA ', Driver_Name - ln, Type - C, Adpt_Config -
N
```

#

```
  EISA_Option = Board_Id - DEC4220, Function_Name - "NET,ETH", Driver_Name - ln
  CMA_Option = Size - 0x10000, Alignment - 0x10000, Addrlimit - 0, Type - 0x1f, Flag - 0

#
# le: LeMAC Ethernet
#
le:
  ISA_Option =  Function_Name - "DE200-LE", Driver_Name - le

#
# el: 3Com EtherLink III Ethernet
#
el:
  PCMCIA_Option = Manufact_Name - "3Com Corporation", Product_Name - "3C589", Manufact_Id - 0x101, Card_Rev - 0, Func_Code - 6, Driver_Name - el, Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 0, Type - C, Adpt_Config - N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1, Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 0, Func_Num - 0

  PCMCIA_Option = Manufact_Name - "3Com Corporation", Product_Name - "3C589D", Manufact_Id - 0x101, Card_Rev - 0, Func_Code - 6, Driver_Name - el, Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 0, Type - C, Adpt_Config - N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1, Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 0, Func_Num - 0

  ISA_Option =  Function_Name - 3C509, Driver_Name - el, Type - C

  PCMCIA_Option = Manufact_Name - "3Com Corporation", Product_Name - "3C562B/3C563B", Manufact_Id - 0x101, Card_Rev - 0, Func_Code - 6, Driver_Name - el, Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 0, Type - C, Adpt_Config - N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1, Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 1, Func_Num - 0

#
# ee: i82558/9 PCI 10/100 Ethernet
#
ee:
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x8086, Device_Id - 0x1229, Driver_Name - ee

#
# tu: TULIP Ethernet/Fast Ethernet
#
tu:
  EISA_Option = Board_Id - DEC4250, Function_Name - Null, Driver_Name - tu, Type - C, Adpt_Config - N
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1011, Device_Id - 2, Driver_Name - tu
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1011, Device_Id - 0x14, Driver_Name - tu
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1011, Device_Id - 0x9, Driver_Name - tu
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1011, Device_Id - 0x19, Driver_Name - tu

#
# alt: DEGPA Gigabit Ethernet
#
alt:
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x12AE, Device_Id - 0x1, Driver_Name - alt

#
# fza: DEFZA FDDI
#
fza:
  TC_Option =  Modname - "PMAF-AA ", Driver_Name - fza, Type - C, Adpt_Config - N

#
# fta: PDQ FDDI
#
fta:
  TC_Option =  Modname - "PMAF-FA ", Driver_Name - fta, Type - C, Adpt_Config - N
  TC_Option =  Modname - "PMAF-FS ", Driver_Name - fta, Type - C, Adpt_Config - N
  TC_Option =  Modname - "PMAF-FD ", Driver_Name - fta, Type - C, Adpt_Config - N
  TC_Option =  Modname - "PMAF-FU ", Driver_Name - fta, Type - C, Adpt_Config - N
  EISA_Option = Board_Id - DEC3001, Driver_Name - fta
  EISA_Option = Board_Id - DEC3002, Driver_Name - fta
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1011, Device_Id - 0xF, Driver_Name - fta

#
# tra: Token-Ring
#
tra:
  TC_Option =  Modname - "PMAT-AA ", Driver_Name - tra, Type - C, Adpt_Config - N
  ISA_Option =  Function_Name - DW110, Driver_Name - tra
  ISA_Option =  Function_Name - TKRNG, Driver_Name - tra
  EISA_Option = Board_Id - PRO6000, Driver_Name - tra
  EISA_Option = Board_Id - PRO6001, Driver_Name - tra
  EISA_Option = Board_Id - PRO6002, Driver_Name - tra
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x104C, Device_Id - 0x0508, Driver_Name - tra
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x10DA, Device_Id - 0x0508, Driver_Name - tra
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x10EF, Device_Id - 0x8154, Driver_Name - tra

#
# lta: ATM
#
# NOTE:  There must be one CMA_Option line for each dynamically
#       loaded Turbo Channel lta device (DGLTA-FA)
#
lta:
  TC_Option =  Modname - "DGLTA-FA",Driver_Name - lta, Type - C, Adpt_Config - N
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1011, Device_Id - 0x16, Driver_Name - lta
#  CMA_Option = Size - 0x23D00, Alignment - 0x10000, Addrlimit - 0, Type - 0x1d, Flag - 2

#
# lfa:  ForeRunner HE Series ATM Adapter
#
lfa:
#   PCI_Option = PCI_SE_Rev - 0x210, Vendor_Id - 0x1127, Device_Id - 0x400, Rev - 0, Base - 0, Sub - 0, Pif - 0, Sub_Vid - 0, Sub_Did - 0, Vid_Mo_Flag - 1, Did_Mo_Flag - 1, Rev_Mo_Flag - 0, Base_Mo_Flag - 0, Sub_Mo_Flag - 0, Pif_Mo_Flag - 0, Sub_Vid_Mo_Flag - 0, Sub_Did_Mo_Flag - 0, Driver_Name - lfa, Type - C, Adpt_Config - N

#
# PCMCIA_Option = Manufact_Name Product_Name Manufact_Id Card_Rev Func_Code
#                       Driver_Name Loadable_Flag Unload_Flag Intr_Handler_Flag Type
#                               Adpt_Config Man_Name_Mo_Flag Prd_Name_Mo_Flag Mid_Mo_Flag Card_Rev_Mo_Flag
#                               Multi_Func_Flag Func_Num
#
pcmcia:
#
# %%%PCMCIA
#
  PCMCIA_Option = Manufact_Name - "AT&T Paradyne", Product_Name - "KeepInTouch Card", Manufact_Id - 0, Card_Rev - 0, Func_Code - 2, Driver_Name - ace, Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 1, Type - C, Adpt_Config - N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1, Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 0, Func_Num - 0
  PCMCIA_Option = Manufact_Name - "Digital", Product_Name - "PCMCIA V.32bis 14,400 Fax", Manufact_Id - 0, Card_Rev - 0, Func_Code - 2, Driver_Name - ace, Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 1, Type - C, Adpt_Config - N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1, Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 0, Func_Num - 0
  PCMCIA_Option = Manufact_Name - "MEGAHERTZ", Product_Name - "XJ2288", Manufact_Id - 0, Card_Rev - 0, Func_Code - 2, Driver_Name - ace, Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 1, Type - C, Adpt_Config - N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1, Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 0, Func_Num - 0
  PCMCIA_Option = Manufact_Name - "3Com Corporation", Product_Name - "3C562B/3C563B", Manufact_Id - 0x101, Card_Rev - 0, Func_Code - 2, Driver_Name
```

```
- ace, Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 1, Type - C,
Adpt_Config - N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1, Mid_Mo_Flag -
0, Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 1, Func_Num - 1
  PCMCIA_Option = Manufact_Name - 'AD PC-CARD', Product_Name -
'RC288ACL', Manufact_Id - 0, Card_Rev - 0, Func_Code - 2, Driver_Name - ace,
Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 1, Type - C, Adpt_Config -
N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1, Mid_Mo_Flag - 0,
Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 0, Func_Num - 0
  PCMCIA_Option = Manufact_Name - 'AD2880WRLD', Product_Name -
'International_V.34_PC-Card_Modem', Manufact_Id - 0, Card_Rev - 0, Func_Code -
2, Driver_Name - ace, Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 1,
Type - C, Adpt_Config - N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1,
Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 0, Func_Num - 0
#
# The following are the ONLY ATA cards that have been tested.  Others are known
# to violate various parts of the spec and therefore fail.  Some devices also
# require ddr.dbase entries due to broken firmware which causes them to self-
# destruct (the Maxtor DDR entry is an example).  Beware if you attempt using
# such devices.
#
  PCMCIA_Option = Manufact_Name - 'INTEGRAL PERIPHERALS',
Product_Name - 'ATA CARD', Manufact_Id - 0, Card_Rev - 0, Func_Code - 4,
Driver_Name - ata, Loadable_Flag - 0, Unload_Flag - 0, Intr_Handler_Flag - 0, Type -
A, Adpt_Config - N, Man_Name_Mo_Flag - 1, Prd_Name_Mo_Flag - 1,
Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0, Multi_Func_Flag - 0, Func_Num - 0
  PCMCIA_Option = Manufact_Name - 'IBM', Product_Name - '', Manufact_Id - 0,
Card_Rev - 0, Func_Code - 4, Driver_Name - ata, Loadable_Flag - 0, Unload_Flag - 0,
Intr_Handler_Flag - 0, Type - A, Adpt_Config - N, Man_Name_Mo_Flag - 1,
Prd_Name_Mo_Flag - 0, Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0,
Multi_Func_Flag - 0, Func_Num - 0
  PCMCIA_Option = Manufact_Name - 'Maxtor', Product_Name - '', Manufact_Id - 0,
Card_Rev - 0, Func_Code - 4, Driver_Name - ata, Loadable_Flag - 0, Unload_Flag - 0,
Intr_Handler_Flag - 0, Type - A, Adpt_Config - N, Man_Name_Mo_Flag - 1,
Prd_Name_Mo_Flag - 0, Mid_Mo_Flag - 0, Card_Rev_Mo_Flag - 0,
Multi_Func_Flag - 0, Func_Num - 0


#
#  qvision: CMPQ Qvision SVGA
#
qvision:
  EISA_Option = Board_Id - CPQ3011, Driver_Name - qvision
  EISA_Option = Board_Id - CPQ3111, Driver_Name - qvision
  EISA_Option = Board_Id - CPQ3112, Driver_Name - qvision
#  PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x0e11, Device_Id - 0x3032,
Driver_Name - qvision

#
#  cirrus: Cirrus SVGA
#
cirrus:
  EISA_Option = Board_Id - DEC5000, Function_Name - VID, Driver_Name - cirrus

#
#  ati64:  ATI Mach64 SVGA
#
ati64:
  EISA_Option = Board_Id - ISA6400, Driver_Name - ati64_vga
  ISA_Option =  Function_Name - 'MACH64', Driver_Name - ati64_vga
#  PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1002, Device_Id - 0x4358,
Driver_Name - ati
#  PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1002, Device_Id - 0x4758,
Driver_Name - ati
#  PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1002, Device_Id - 0x4354,
Driver_Name - ati

#
#  s3trio:  S3 Trio 32/64 SVGA
#
s3trio:
#  PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x5333, Device_Id - 0x8811,
Driver_Name - trio
#  PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x5333, Device_Id - 0x8810,
Driver_Name - trio

#
#  wd:  Western Digital WD90C24 SVGA
#
  ISA_Option =  Function_Name - 'WD90', Driver_Name - wd

#
#  vga: Standard 640x480 VGA
#
vga:
  EISA_Option = Board_Id - PHI8041, Driver_Name - vga
  ISA_Option =  Function_Name - 'ISA--VGA', Driver_Name - vga
#  PCI_Option =  PCI_SE_Rev - 0x210, Base - 3, Sub - 0, Driver_Name - vga
#  PCI_Option =  PCI_SE_Rev - 0x210, Base - 0, Sub - 1, Driver_Name - vga

#
#  other SVGAs
#
#s3v864:
##  PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x5333, Device_Id - 0x88c0,
Driver_Name - svision
#
#ati32:
#  PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x1002, Device_Id - 0x4158,
Driver_Name - ati_vga, Type - C, Adpt_Config - N

#
# VBA_Option = Manufact_Name Product_Name Bus_Instance Driver_Name
#            Driver_Instance Csr1 Csr2 Vector Bus_Prioity
#            Type Adpt_Config
#
vb:
#
# %%%VB
#
  VBA_Option = Manufact_Name - 'Digital', Product_Name - 'VME Backplane
Network Driver',  Bus_Instance - 0, Driver_Name - vb, Driver_Instance - 0, Csr1 - 0,
Csr2 - 0, Vector - 0x1150, Bus_Priority - 7, Type - C, Adpt_Config - N
#
# pfm: Pseudo driver needed for performace measurements ... kprofile() and/or
uprofile()
#
pfm:
  Module_Config_Name = pfm
  Device_Dir = /dev
  Device_Char_Major = Any
  Device_Char_Minor = 0
  Device_Char_Files = pfcntr
  Device_User = root
  Device_Group = 0
  Device_Mode = 444
  Device_Major_Req = Same

#
# USB drivers.  Supported options are: Driver_Name, Vendor_Id, Product_Id,
#            Release_Num, Device_Class, Device_Sub_Class, Device_Protocol,
#            Configuration_Value, Interface_Number, Interface_Sub_Class,
#            Interface_Protocol, Interface_Class
DEC_USBhub:
            USB_Class_Driver = Driver_Name - usb_hub, Release_Num - 0x1000,
Device_Class - 0x09, Device_Sub_Class - 0x1, Config_Device - 0x1
            USB_Class_Driver = Driver_Name - usb_hub, Release_Num - 0x1000,
Device_Class - 0x09, Device_Sub_Class - 0x0, Config_Device - 0x1
DEC_USBmouse:
            USB_Class_Driver = Driver_Name - usb_mouse, Release_Num - 0x1000,
Device_Class - 0xff, Interface_Class - 0x03, Interface_Sub_Class - 0x01,
Interface_Protocol - 0x02, Config_Interface - 0x1
DEC_USBkeyboard:
            USB_Class_Driver = Driver_Name - usb_keyboard, Release_Num -
0x1000, Device_Class - 0xff, Interface_Class - 0x03, Interface_Sub_Class - 0x01,
Interface_Protocol - 0x01, Config_Interface - 0x1
DEC_USBhid:
            USB_Class_Driver = Driver_Name - usb_hid, Release_Num - 0x1000,
Device_Class - 0xff, Interface_Class - 0x03, Interface_Sub_Class - 0x01,
Config_Interface - 0x1

#
# KGPSA Fibre Channel Adapter
#
emx:
#            memlog_flags = 0xffffffff
#            memlog_size = 32768
            panic_on_no_context = 1

#  PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x10df, Device_Id - 0x1ae5,
Driver_Name - emx, Type - A, Adpt_Config - N
```

```
#   PCI_Option =  PCI_SE_Rev - 0x210, Vendor_Id - 0x10df, Device_Id - 0xf700,
Driver_Name - emx, Type - A, Adpt_Config - N


# Entries merged forward from /etc/sysconfigtab

# *    Copyright (c) Digital Equipment Corporation, 1991, 1999    *
# *   All Rights Reserved. Unpublished rights reserved under   *
# *   the copyright laws of the United States.                 *
# *                                                            *
# *   and embodies the confidential technology of Digital   *
# *   Equipment Corporation. Possession, use, duplication or   *
# *   pursuant to a valid written license from Digital Equipment  *
#       aio_max_num = 16384
#               aio_percpu_data = 1
#               ipqmaxlen=2048
#               ipqs=8
#               tcbhashnum=8
#               tcbhashsize=8192
#       gh-fail-if-no-mem = 1
#       Device_Major_Req = Same

generic:
 memberid=1
 msgbuf_size=16384
 rt_preempt_opt=1
 kmem-debug=0
 act_vers_high=1450159080085520384
 act_vers_low=148
 rolls_ver_lookup=0
 lockmode=2
 locktype = 0

ipc:
                msg_max = 32768
                msg_mnb = 16384
                msg_mni = 256
                msg_tql = 40
                sem_aem = 32768
                sem_mni = 1024
                sem_msl = 2000
                sem_opm = 2000
                sem_ume = 2000
                sem_vmx = 320000
                shm_max = 2139095040
                shm_mni = 1024
                shm_seg = 512
                ssm_threshold = 0
                shm_allocate_striped = 0

rt:
                aio_task_max_num = 16384

io:
                basic_dma_window_size = 2048

proc:
#               give_boost = 0
                max_per_proc_address_space = 128000000000
                max_per_proc_data_size = 128000000000
                max_proc_per_user = 8192
                max_threads_per_user = 8192
                max_users = 8192
                per_proc_address_space = 128000000000
                per_proc_data_size = 128000000000

rm:
                rm_no_inheritance = 1

net:
                netisrthreads = 8
inet:
        ipqmaxlen=2048
        ipqs=8
        ipport_userreserved=20000
        tcbhashnum=8
        tcbhashsize=8192
                udpcksum = 0
                udp_recvspace = 65536
                udp_sendspace = 65536
                udp_ttl = 10

vm:
 swapdevice=/dev/disk/dsk239b,/dev/disk/dsk1c,/dev/disk/dsk2c,/dev/disk/dsk5c
 vm_page_free_reserved=20
 vm_page_free_min=30
                dump_user_pte_pages = 1
                new_wire_method = 1
                rad_gh_regions[0] = 15032
                rad_gh_regions[1] = 15032
                rad_gh_regions[2] = 15032
                rad_gh_regions[3] = 15032
                rad_gh_regions[4] = 15032
                rad_gh_regions[5] = 15032
                rad_gh_regions[6] = 15032
                rad_gh_regions[7] = 15032
                replicate_user_text = 1
                vm-swap-eager = 0
                vm_troll_interval = 0

lsm:
 lsm_rootdev_is_volume=0
                Enable_LSM_Stats = 0

pcount:
                Subsystem_Description = pcount device driver
                Module_Config_Name = pcount
                Module_Type = Dynamic
#       Device_Major_Req = Same
                Device_Char_Major = ANY
                Device_Char_Minor = 0
                Device_Char_Files = pcount0

clubase:
 cluster_expected_votes=1
 cluster_name=tpcc
 cluster_node_name=wftpcc1
 cluster_node_inter_name=wftpcc1-mc0
 cluster_node_votes=1
 tnc_node_info=10.0.0.1
 tnc_node_mask=255.255.255.0
 global_root_major=19
 global_root_minor=3877
 tnc_interface=mc0
 cluster_seqdisk_major=19
 cluster_seqdisk_minor=3907
 cluster_qdisk_major=0
 cluster_qdisk_minor=0
 cluster_qdisk_votes=0

sec:
 acl_mode = disable

rdg:
                max_objs=4096
                max_async_req=1024
                msg_size = 2048
#               max_sessions = 1024
                max_sessions = 2048
                rdg_max_auto_msg_wires = 2000
# debug       rdg_max_auto_msg_wires=0

cfs:
                cfslog_max_size=163840

drd:
                drd_enable_drd=0
dlm:
                dlm_scn_switch=1


]
```

Appendix E

Auditor Attestation

Benchmark Sponsor:

Dave Stanley
Compaq Computer
110 Spit Brook Road
ZKO2-3/M31
Nashua NH, 03062

November 9, 2000

I verified the performance of the following configuration:

Platform:	**Compaq AlphaServer GS320 Model 6/731 c/s**
Operating system:	**Compaq Tru64 UNIX V5.1**
Database Manager:	**Oracle8 V8.1.7**
Transaction Manager:	**Compaq DB Web Connector V1.1**

The performance was measured according to the requirements of the TPC-C Benchmark. The results were:

CPU's Speed	Memory	Disks	NewOrder 90% Response Time	**tpmC**
Server: **AlphaServer GS320 Model 6/731**				
32 x Alphachip 21264A (731 MHz)	128 GB (4 MB cache per cpu)	854 x 18 GB 20 x 9 GB	1.826 Seconds	**155,179.25**
Eight Clients: **Compaq ProLiant ML370** (Specification for each)				
2 x Pentium III (600 MHz)	640 MB	1 x 18 GB	n/a	n/a

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

- The database records were the proper size
- The database was properly scaled and populated
- The required ACID properties were met
- The transactions were correctly implemented
- Input data was generated according to the specified percentages
- The transaction cycle times included the required keying and think times
- The reported response times were correctly measured.
- All 90% response times were under the specified maximums

- At least 90% of all delivery transactions met the 80 Second completion time limit
- The reported measurement interval was 120 minutes (7200 seconds)
- The reported measurement interval was representative of steady state conditions
- Four checkpoints were taken during the reported measurement interval
- The repeatability of the measured performance was verified
- The 180 day storage requirement was correctly computed
- The system pricing was verified for major components and maintenance

Additional Audit Notes:

> The measured system included (724) DS-RZ1DA-VW (9 GB disks) that were substituted by (724) DS-RZ1EA-VW(18 GB disks) in the priced configuration. Based on the specifications of these disks and on additional performance data collected on these disks, it is my opinion that this substitution meets all the TPC requirements to ensure that it has no material effect on the reported performance.

Respectfully Yours,

François Raab
President

Appendix F

Price Quotations

8-Nov-00 Quote No: 131788

Compaq Computer Corp.

Maria Lopez

SERVER HARDWARE:	Part Num.	Qty	Unit Price	Extension
Compaq AlphaServer GS320 6/731 Model 32	DA-320EC-AA	1	412920	$412,920
GS160/320 ADD-on CPU UNIX SMP	3X-KN8AA-AD	31	14260	$442,060
GS80/160/320 2GB MEMORY OPT	3X-KN8AA-CB	32	16120	$515,840
GS80/160/320 2GB MEMORY UPG	3X-KN8AA-CU	32	13640	$436,480
PCI to Fibre Adapter	DS-KGPSA-CA	52	2480	$128,960
PCI to UltraSCSI Adapter	KZPBA-CA	2	285	$570
D shf 180w Dif Blw mtrc blue	DS-BA356-JF	2	908	$1,816
16 bit SCSI for KZPSC/BA356	BN31S-1E	2	72	$144
1600 Watt Bulk Power Converter	H7506-AA	4	6200	$24,800
GS160/320 PCI Expansion Box	3X-DWWPA-BA	8	7440	$59,520
GS80/160/320 Expansion Cab	H9A20-AA	2	3100	$6,200
15M Fibre Op Cable 50/125	234457-B23	52	141	$7,332
PCI to GbE SX Adapter	DEGPA-SA	8	1208	$9,664
PCI Memory Channel Two Controller	CCMAB-AA	1	1854	$1,854
5YR 5x9 4/HR GS320 25-32 CPU	*FM-W64HR-60*	*1*	*381570*	*$381,570*
SC-SC Dual FO Cbl, MM, PP, 4.5M	BN34B-4E	8	60	$480
VT510,WHITE,NORTH AMER,NO KEY	VT510-AA	1	245	$245
US& Canada/English KB	PCXLA-NA	1	16	$16
SUBTOTAL SERVER HARDWARE:				**$2,430,471**
STORAGE:				
ESA 12000 Storage Array	380620-001	18	54950	$989,100
5YR 5X9/4HR 24 Slot Enc/No Drives	*FM-CG4HR-60*	*36*	*1635*	*$58,860*
5YR 5x9/4hr; HSG80 w/Cache	*FM-CA4HR-60*	*72*	*2022*	*$145,584*
5YR 5X9/4HR 41U/42U CAB	*FM-CB4HR-60*	*18*	*1338*	*$24,084*
15M Fibre Op Cable 50/125	234457-B23	144	141	$20,304
9.1GB 7200rpm UWSCSI Disk**	DS-RZ1DA-VW	21	403	$8,463
18.2GB 7200rpm UWSCSI Disk**	DS-RZ1EA-VW	940	741	$696,540
16 Port Fibre Switch	DS-DSGGB-AB	13	22143	$287,859
Short Wave Optical GBIC	380561-B21	196	305	$59,780
5YR 5X9/4HR $35k-$44999	*FM-NB4HR-60*	*13*	*10797*	*$140,361*
12/24GB 4mm DAT Tape	TLZ10-LB	1	704	$704
5YR 5X9/4HR DAT TAPE	*FM-4M4HR-60*	*1*	*1019*	*$1,019*
SUBTOTAL STORAGE:				**$2,432,658**
Network:				
GS-Router 16 Slot Chassis	DGSRA-BA	2	6197	$12,394
5YR 5X9/4HR; NTWK $7000 - $9000	*FM-N74HR-60*	*2*	*2294*	*$4,588*
GS-/R 64MB Switch Control Pro	DGSRC-BA	2	5577	$11,154
5YR 5X9/4HR; NTWK $7000 - $9000	*FM-N74HR-60*	*2*	*2294*	*$4,588*
GS/Rtr 2-P GEEth SX 16MB Line Card	DGSRS-AB	8	4957	$39,656
5YR 5X9/4HR; NTWK $7000 - $9000	*FM-N74HR-60*	*8*	*2294*	*$18,352*
GS/R 16-Slot Switching Fabric Module	DGSRD-AA	4	3097	$12,388
5YR 5X9/4HR NTWK $3000-$4999	*FM-N54HR-60*	*4*	*1080*	*$4,320*
GS/Rtr 8-P Fast Ethernet 16MB Line Card	DGSRT-AB	17	3717	$63,189
5YR 5X9/4HR; NTWK $5000-$6999	*FM-N64HR-60*	*17*	*1619*	*$27,523*
NETWORK SUBTOTAL:				**$198,152**

SERVER SOFTWARE:				
5YR TRU64UNIX BRZ9x5	*FM-W16U9-60*	*1*	*27259*	*$27,259*
5YR GS160/320 TRU64 UNIX SMP	*FM-W16SM-60*	*31*	*2074*	*$64,294*
5YR Tru64Unix O/S & LP	*FM-CDDST-60*	*1*	*10899*	*$10,899*
HSG ACS SF All/Lic PCRM Pkg.	128697-B21	72	4340	$312,480
GIGAsw/Rtr All Lic/Mcd Pkg	QB-64PAA-WA	1	2477	$2,477
*5YR 5x9 HS*80/60 ACS Dual SW*	*FM-DBAL6-60*	*36*	*7488*	*$269,568*
*5YR 5x9 HS*80 Platform SW*	*FM-PBAT1-60*	*36*	*1211*	*$43,596*
Digital UNIX Alpha CDRM	QA-MT4AA-H8	1	245	$245
SERVER SOFTWARE SUBTOTAL:				**$730,818**
TOTAL SERVER:				**$5,792,099**
CLIENT HARDWARE:				
ML370T01 P600-256 128 US	195293-001	8	2390	$19,120
Processor 600Mhz	128292-B21	*8*	623	$4,984
256MB SDRAM DIMM Memory Option	128278-B21	16	747	$11,952
Ethernet Dual Channel	338456-B21	8	177	$1,416
Gigabit Daughter CD Upgrade	338456-B23	8	353	$2,824
18GB Drive	388144-B22	8	345	$2,760
5YR 5x9 4HR WorkGroup Svr	*FM-LO4HR-60*	*8*	*2192*	*$17,536*
SC-SC Dual FO Cbl, MM, PP, 4.5M	BN34B-4E	8	60	$480
V700 15" Color Monitor	325900-001	10	79	$790
CLIENT HARDWARE SUBTOTAL:				**$61,862**
CLIENT SOFTWARE:				
Application Optimizer WNT Server Edition	QB-641AA-SA	8	279	$2,232
Application Optimizer WNT Server Service	*QT-641AA-XA*	*5*	*58*	*$2,320*
CLIENT SOFTWARE SUBTOTAL:				**$4,552**
TOTAL CLIENT:				**$66,414**
Configuration Total:				**$5,858,513**

The Compaq AlphaServer GS320 carries a five (5) year, 5x9/4HR response warranty. The storage have a five (5) year on-site, 4-hour, 5-day per week response with a five (5) year return to manufacture warranty. All other products carry a standard warranty of five (5) years on-site; 4-hour x 5-days per week.
38% Discount based on total dollar volume: Server Hardware and Software
20'% Discount based on total dollar volume: FM part numbers.

Traditional Compaq Intel Based Client Hardware Priced at US1plus%

Valid: This quote is valid for 60days from date
Terms: TBD
Delivery: 15 Days ARO
Shipping: FOB Origin
Warranty: Manufactures New Equipment
Installation: Included

Sincerely,

Philip K. Nolan
Vice President
Phone No: 201-666-1122 exten:111
Fax No: 201-666-0956
e-mail: phil@icssolutions.com

QUOTE

NetCruiser Technologies, Inc
3005 Hadley Road Unit #1
South Plainfield, NJ 07080
(908) 222-8811

ORDER NUMBER 0000136
ORDER DATE. 10/31/00

SALESPERSON 0002
CUSTOMER NO. 00-A0USER0

SOLD TO:
Yahoo Order (USA)/DBA FormReady
,

SHIP TO:
Compaq computer Corp
Maria Lopez
,

CONFIRM TO:

CUSTOMER P.O	SHIP VIA	F.O.B.	TERMS
	UPS G		VISA Card

ITEM NUMBER	UNIT	ORDERED	SHIPPED	BACK ORDER	PRICE	AMOUNT
CT1008D/P	EACH	1	0	0	18.00	18.00
8 Port Desktop Hub/Plan			WHSE: 000			

Net Order:	18.00
Less Discount:	0.00
Freight:	0.00
Sales Tax:	0.00
Order Total:	**18.00**