Hewlett-Packard Company

TPC Benchmark™ C Full Disclosure Report for HP Integrity rx5670 Using Oracle Database 10g Standard Edition and Red Hat Enterprise Linux AS 3

Third Edition
December 31, 2003

Third Edition – December 31, 2003

Table of Contents

Preface

The TPC Benchmark C was developed by the Transaction Processing Performance Council (TPC). The TPC was founded to define transaction processing benchmarks and to disseminate objective, verifiable performance data to the industry. This full disclosure report is based on the TPC Benchmark C Standard Specifications Version 5.0, released March 7, 2001.

TPC Benchmark C Overview

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

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

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

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

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

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

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

Abstract

Overview

This report documents the methodology and results of the TPC Benchmark C test conducted on the hp integrity rx5670. The operating system used for the benchmark was Red Hat Enterprise Linux AS 3 . The DBMS used was Oracle Database 10g Standard Edition.

TPC Benchmark C Metrics

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

136110.975tpmC
$3.94 per tpmC
Available as of March 5, 2004*. Hardware is available now.

Standard and Executive Summary Statements

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

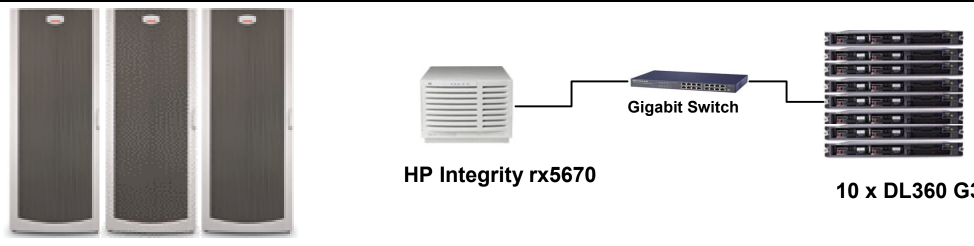
Auditor

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

	HP Integrity rx5670 C/S with 10 ProLiant DL360-G3	TPC-C Rev. 5.1 Report Date: December 31, 2003

Total System Cost	TPC-C Throughput	Price/Performance	Availability Date
$536,783	**136110.98**	**$3.94**	**March 5, 2004*** ***Hardware available now**

Processors	Database Manager	Operating System	Other Software	Number of Users
4 x 1.5GHz Intel Itanium 2 6M Processors – Server 20 x Xeon 2.4GHz – Client	Oracle Database 10g Standard Edition	Red Hat Enterprise Linux AS 3	BEA Tuxedo 8.1	**108000**



3 HP Rack 9142 containing: 24 X 4314R Storage Works Enclosure with 14X 18.2 GB 15K drives each and 2X Storage Works MSA 1000s each with 10X 36.4 GB 15K drives each

System Components	**Server** Quantity	**Server** Description	**Each Client** Quantity	**Each Client** Description
Processor	4	1.5GHz Itanium 2 6M w/ 6MB Cache	2	2.4GHz Xeon w/ 256K cache
Memory	48	2GB	4	1024MB
Disk Controllers	8	HP SMART Array Controller 5304/128	1	Integrated SMART 5i Controller
	1	Integrated SCSI Controller		
	1	hp StorageWorks fca2214DC		
Disk Drives	336	18GB 15K SCSI Drives	1	36 GB 15K SCSI Drive
	20	36GB 15K SCSI Drives		
Total Storage		6843.80 GB		36 GB
Tape Drives	1	20/40 GB DAT		

Hewlett-Packard Company	HP Integrity rx5670 C/S				TPC-C Rev. 5.1 Report Date: December 31, 2003	
Description	**Price Key**	**Part Number**	**Unit Price**	**Qty**	**Extended Price**	**3 Yr Maint Price**
hp server rx5670,1.5GHz Itanium 2 w/ 6MB iL3 cache,0 MB RAM, 0 disk	1	A6838B	$26,494	1	$26,494	
CPU upgrade Itanium 2, 1.5GHz w/ 6MB iL3 cache	1	A9810A	$8,250	3	$24,750	
8GB PC2100 DDR-SDRAM (4x2GB DIMMs)	1	A6835A	$16,000	12	$192,000	
Memory Carrier Board	1	A6747A	$1,981	2	$3,962	
HP 36GB, 15krpm Ultra320 hot-swap disk	1	A7049A	$819	1	$819	
HP Rackmount Kit Factory	1	A5580A	$134	1	$134	
DVD Rom drive	1	A5557B	$450	1	$450	
Graphics USB Card	1	A6869A	$349	1	$349	
HP USB keyboard and mouse	1	A7861A	$32	1	$32	
HP Smart Array Controller 5304	2	283551-B21	$2,247	8	$17,976	
hp StorageWorks fca2214DC	2	321835-B21	$2,500	1	$2,500	
5m LC to LC Cable Kit	2	221692-B22	$82	2	$164	
S5500 15 carbon / silver monitor	2	261602-001	$129	1	$129	
HP Rack Model 9142 (42U - Opal) - Flat Pallet	2	120663-B21	$1,321	3	$3,963	
HP Power Distribution Unit 120-240V	1	E7671A	$145	3	$435	
UPS R1500 XR	2	204404-001	$866	1	$866	
HP Hardware Support 3 yr, 24x7, 4 hr rx5670	1	H4405Y-6BO	$7,052	1		$7,052
HP Hardware Support 3 yr, 24x7, 4 hr addt'l CPU	1	H4405Y-6BP	$1,153	3		$3,459
20/40 GB DAT Drive, External	1	C5687B	$1,450	1	$1,450	
Storageworks Modular SAN Array 1000	2	201723-B22	$9,995	2	$19,990	
FM-FC724-36 3YR 24x7 4HR RA4X/MSA1x/SA	2	402164-002	$3,538	2		$7,076
Storageworks Enclosure Model 4314R	2	190209-001	$2,955	24	$70,920	
FM-4E724-36 3YR 24X7 4HR EMPTY DISK ENCL	2	171242-002	$157	24		$3,768
18GB, 15krpm Ultra320 Wide disk	2	286775-B22	$299	336	$100,464	
18GB, 15krpm Ultra320 Wide disk (10% spares)	2	286775-B22	$299	34	$10,166	
36GB, 15krpm Ultra320 Wide disk	2	286776-B22	$519	20	$10,380	
36GB, 15krpm Ultra320 Wide disk (10% spares)	2	286776-B22	$519	2	$1,038	
	Server Subtotal				**$489,431**	**$21,355**
Oracle Database 10g Standard Edition, processor 3 yrd. unlimited users+A64	3	run-time	$7,500	4	$30,000	
Oracle Database Server Support Package 3 years	3	run-time	$2,000	3		$6,000
Red Hat Enterprise Linux AS for Itanium Processor (Ver. 3 Std. Edi.)	4		$1,992	1	$1,992	
2 Addi. Yrs Subs. to Red Hat Ent. Linux AS for Itanium (Ver. 3 Std. Edi.)	4		$1,992	2		$3,984
	Server Software Subtotal				**$31,992**	**$9,984**
HP ProLiant DL360R03 X2.4-512KB/533, 512MB	2	292887-001	$2,199	10	$21,990	
2.4GHz Xeon processor kit	2	292891-B21	$639	10	$6,390	
2Gb Reg PC2100 2X1Gb	2	300680-B21	$1,300	20	$26,000	
36GB, 15krpm Ultra320 Wide disk	2	286776-B22	$519	10	$5,190	
FM-I0724-36-3yr 24X7 4HR 300 SERIES SVR	2	162657-002	$949	10		$9,490
	Client Subtotal				**$59,570**	**$9,490**
Red Hat Enterprise Linux ES (version 3 Standard Edition)	4		$799	10	$7,990	
2 Addi. Yrs Subs. to Red Hat Ent. Linux ES (Ver. 3 Std Edi.)	4		$799	20		$15,980
BEA Tuxedo 8.1 Teir 1	5		$1,200	10	$12,000	7560
	Client Software Subtotal				**$19,990**	**$23,540**
GS508TNA 8PORT 10/100/1000BTX COPPER GIGABIT SWITCH	6	GS508TNA	$587	3	$1,760	
	Connectivity Subtotal				**$1,760**	
HP's Large Configuration Discount *					-$122,963	-$5,566
Oracle Mandatory E-Business Discount (license and support)					-$1,800	
	Total:				**$477,980**	**$58,803**
Price Key: 1-HP at 22% discount, 2-HP at 16% discount, 3-Oracle (Oracle pricing contact: MaryBeth Pierantoni, mary.beth.pierantoni@oracle.com, 650-506-2118. Please see Appendix E of FDR), 4-Red Hat 5-BEA, 6-Netgear	**3 year cost of ownership:**					**$536,783**
			tpmC:			**136110.98**
* All discounts are based on US list prices and for similar quantities and configurations			**$/tpmC:**			**$3.94**

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. Results independently audited by Lorna Livingtree of Performance Metrics Inc. Original Report Date 5 September, 2003 and repriced on 17 September, 2003.

Numerical Quantities Summary

MQTH, Computed Maximum Qualified Throughput — 136110.975 tpmC

Response Times (in seconds)	Average	90%	Maximum
New-Order	0.302	0.459	18.672
Payment	0.211	0.318	98.901
Order-Status	0.239	0.364	6.971
Delivery (interactive portion)	0.101	0.102	7.169
Delivery (deferred portion)	0.023	0.037	258.549
Stock-Level	0.193	0.289	42.502
Menu	0.101	0.102	1.830

Transaction Mix, in percent of total transaction	
New-Order	44.915%
Payment	43.020%
Order-Status	4.020%
Delivery	4.020%
Stock-Level	4.020%

Emulation Delay (in seconds)	Resp.Time	Menu
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

Keying/Think Times (in seconds)	Min.	Average	Max.
New-Order	18.005/0.00	18.009/12.085	18.018/120.752
Payment	3.010/0.00	3.019/12.015	3.027/120.085
Order-Status	2.010/0.00	2.019/10.014	2.027/99.823
Delivery (interactive)	2.010/0.00	2.019/5.025	2.027/50.049
Stock-Level	2.010/0.00	2.019/5.014	2.026/49.782

Test Duration	
Ramp-up time	133minutes
Measurement interval	120 minutes
Transactions (all types) completed during measurement interval	37,040,792
Ramp down time	36 minutes

Checkpointing	
Number of checkpoints	5
Checkpoint interval	30 minutes

General Items

Application Code and Definition Statements

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

Appendix A contains all source code implemented in this benchmark.

Test Sponsor

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

This benchmark was sponsored by Hewlett Packard Company. The benchmark was developed and engineered by Hewlett Packard Company and Oracle Corporation. Testing took place at HP Database Performance Engineering Laboratory in Houston, Texas.

Parameter Settings

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

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

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

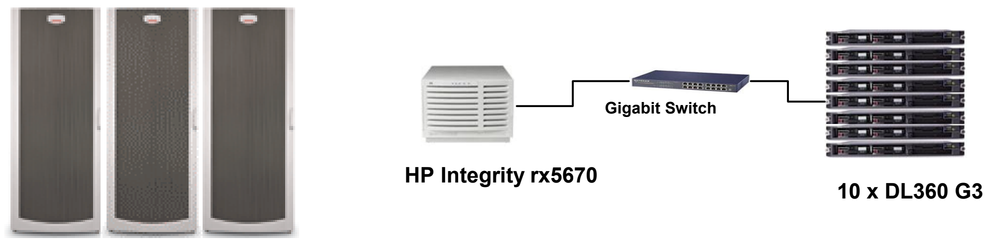
Appendix C contains the tunable parameters for the database, the operating system, and the transaction monitor.

Configuration Items

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

The configuration diagram for both the tested and priced system are the same and included on the following page

Figure 1. Benchmarked and Priced Configuration



3 HP Rack 9142 containing: 24 X 4314R Storage Works Enclosure with 14X 18.2 GB 15K drives each and 2X Storage Works MSA 1000s each with 10X 36.4 GB 15K drives each

Clause 1 Related Items

Table Definitions

Listing must be provided for all table definition statements and all other statements used to set up the database.
Appendix B contains the code used to define and load the database tables.

Physical Organization of Database

The physical organization of tables and indices within the database must be disclosed.
336 disks used in the benchmark had a capacity of 18.2GB 15K rpm, and 20 disks used in the benchmark had a capacity of 36.4 GB 15K rpm.

Controller	Storage	Unformatted Capacity	Contents
1. hp SMART Array Controller 5304	3 Storageworks Enclosure Model 4314R (3 x 14 x 18.2 GB 15K rpm disk drives)	764GB	Tables, Indexes
2. hp SMART Array Controller 5304	3 Storageworks Enclosure Model 4314R (3 x 14 x 18.2 GB 15K rpm disk drives)	764GB	Tables, Indexes
3. hp SMART Array Controller 5304	3 Storageworks Enclosure Model 4314R (3 x 14 x 18.2 GB 15K rpm disk drives)	764GB	Tables, Indexes
4. hp SMART Array Controller 5304	3 Storageworks Enclosure Model 4314R (3 14 x 18.2 GB 15K rpm disk drives)	764GB	Tables, Indexes
5. hp SMART Array Controller 5304	3 Storageworks Enclosure Model 4314R (3 x14 x 18.2 GB 15K rpm disk drives)	764GB	Tables, Indexes
6. hp SMART Array Controller 5304	3 Storageworks Enclosure Model 4314R (3 x14 x 18.2 GB 15K rpm disk drives)	764GB	Tables, Indexes
7. hp SMART Array Controller 5304	3 Storageworks Enclosure Model 4314R (3 x 14 x 18.2 GB 15K rpm disk drives)	764GB	Tables, Indexes
8. hp SMART Array Controller 5304	3 Storageworks Enclosure Model 4314R (3 x 14 x 18.2 GB 15K rpm disk drives)	764GB	Tables, Indexes
9. hp StorageWorks fca2214DC	Port -1 Storageworks Modular SAN Array 1000 (10 x 36.4 GB 15K rpm disk drives)	362GB	Redo Logs
	Port 2- Storageworks Modular SAN Array 1000 (10 x 36.4 GB 15K rpm disk drives)	362GB	Redo Logs

Priced Configuration:

All hardware and software remained the same between the benchmarked and priced configurations.

Insert and Delete Operations

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

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

Partitioning

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

None.

Replication, Duplication or Additions

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

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

Clause 2 Related Items

Random Number Generation

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

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

Input/Output Screen Layout

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

All screen layouts followed the specifications exactly.

Priced Terminal Feature Verification

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

The terminal attributes were verified by the auditor manually exercising each specification on a representative ProLiant DL360R.

Presentation Manager or Intelligent Terminal

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

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

Transaction Statistics

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

Table 2. 1 Transaction Statistics

Statistic		Value
New Order	Home warehouse order lines	99.00%
	Remote warehouse order lines	1.00%
	Rolled back transactions	1.00%
	Average items per order	10.00
Payment	Home warehouse	85.00%
	Remote warehouse	15.00%
	Accessed by last name	60.01%
Order Status	Accessed by last name	60.04%
Delivery	Skipped transactions	None
Transaction Mix	New Order	44.915%
	Payment	43.020%
	Order status	4.020%
	Delivery	4.025%
	Stock level	4.020%

Queuing Mechanism

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

BEA Tuxedo on each client system served as the queuing mechanism to the database. Each delivery request was submitted to BEA Tuxedo asynchronously with control being returned to the client process immediately and the deferred delivery part completing asynchronously.

Clause 3 Related Items

Transaction System Properties (ACID)

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

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

Atomicity

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

Completed Transactions

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

Aborted Transactions

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

Consistency

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

Consistency conditions one through four were tested using a shell script to issue queries to the database. The results of the queries verified that the database was consistent for all four tests.

A run was executed under full load over two hours with checkpoints.

The shell script was executed again. The result of the same queries verified that the database remained consistent after the run.

Isolation

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

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

Durability

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

Durable Media Failure

Durability from media failure was demonstrated on a database scaled for 20000 warehouses. The standard driving mechanism was used to generate the transaction load of 1080000 users. The fully scaled database under full load would also have passed the following test.

Loss of Data

To demonstrate recovery from a permanent failure of durable medium containing TPC-C tables, the following steps were executed:

1. A partition on a disk was backed up.
2. The total number of New Orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving the beginning count. Consistency check 3 was verified before run.
3. The RTE was started with 20000 users
4. The test was allowed to run for a minimum of 10 minutes.
5. The backed up partition was overwritten with garbage information.
6. Oracle10g recorded errors about corrupt data on the partition. The database and the RTE were then shut down.
7. The database partition which was backed up in Step 1 was restored.
8. The database was then started. The database was recovered using the recover command from SQLPLUS. The database was opened and Oracle 10g performed instance recovery.
9. Consistency conditions were executed and verified.
10. Step 2 was repeated and the difference between the first and second counts was noted.
11. An RTE report was generated for the entire run time giving the number of NEW-ORDERS successfully returned to the RTE.
12. The counts in step 9 and 10 were compared and the results verified that all committed transactions had been successfully recovered.
13. Samples were taken from the RTE files and used to query the database to demonstrate successful transactions had corresponding rows in the ORDER table.

Loss of Log

To demonstrate recovery from a permanent failure of durable medium containing TPC-C tables, the following steps were executed:

1. The total number of New Orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving the beginning count. Consistency check 3 was verified before run.
2. The RTE was started with 20000 users.
3. The test was allowed to run for a minimum of 10 minutes.
4. A log disk containing log information was removed.
5. The system continued running because the logs are mirrored.
6. The database and the RTE were then shut down.
7. The database was then started. Consistency conditions were executed and verified.
8. Step 1 was repeated and the difference between the first and second counts was noted.
9. An RTE report was generated for the entire run time giving the number of NEW-ORDERS successfully returned to the RTE.
10. The counts in step 7 and 8 were compared and the results verified that all committed transactions had been successfully recovered.
11. Samples were taken from the RTE files and used to query the database to demonstrate successful transactions had corresponding rows in the ORDER table.

Instantaneous Interruption, Loss of Memory

Because loss of power erases the contents of memory, the instantaneous interruption and the loss of memory tests were combined into a single test. This test was executed on a fully scaled database of 10800 warehouses under a full load of 108000 users. The following steps were executed:

1. The total number of New Orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving the beginning count.
2. The RTE was started with 108000 users.
3. The test was allowed to run for a minimum of 10 minutes.
4. A checkpoint was issued.
5. Upon completion of the checkpoint a system crash and loss of memory were induced by turning all six of the computers in the cluster off. No battery backup or Uninterruptible Power Supply (UPS) were used to preserve the contents of memory.
6. The RTE was shutdown.
7. Power was restored and one of the systems restarted.
8. Oracle10g was restarted and performed an automatic recovery.
9. Consistency conditions were executed and verified.
10. Step 1 was repeated and the difference between the first and second counts was noted.
11. An RTE report was generated for the entire run time giving the number of NEW-ORDERS successfully returned to the RTE.
12. The counts in step 9 and 10 were compared and the results verified that all committed transactions had been successfully recovered.
13. Samples were taken from the RTE files and used to query the database to demonstrate successful transactions had corresponding rows in the ORDER table.

Clause 4 Related Items

Initial Cardinality of Tables

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

Table 4.1 Number of Rows for Server

Table	Occurrences
Warehouse	10800
District	108000
Customer	324000000
History	324000000
Order	324000000
New Order	97200000
Order Line	3244354000
Stock	10800000000
Item	100000
Unused Warehouses	0

Database Layout

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

The benchmarked configuration used eight Smart Array Controllers with three StorageWorks Enclosure 4314Rs with 14 disk drives each for the database. Array accelerator cache for data volumes were set to 100% write.

Two hp StorageWorks MSA1000s each with 10 disks were used for database redo log. Each MSA1000s had one hardware RAID 0 volume. hp StorageWorks MSA1000s were connected to the system using hp StorageWorks fca2214DC, dual port fibre channel HBA. The two RAID 0 volumes were mirrord (RAID 1) at the OS level using the mkraid utility of Linux OS.
Array accelerator cache were set to 100% write on hp StorageWorks MSA1000s.

Section 1.2 of this report details the distribution of database tables and logs across all disks. The code that creates the database and tables are included in Appendix B.

Type of Database

A statement must be provided that describes:

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

Oracle Database 10g Edition is a relational DBMS.

Anonymous block PL/SQL and stored procedures were accessed through the ORACLE Call Interface. Application code is included in Appendix A.

Database Mapping

The mapping of database partitions/replications must be explicitly described.
The database was not replicated. The tables were not partitioned.

60 Day Space

Details of the 60 day space computations along with proof that the database is configured to sustain 8 hours of growth for the dynamic tables (Order, Order-Line, and History) must be disclosed.

SEGMENT	BLOCKS	BLOCK_SIZE	REQUIRED	STATIC	DYNAMIC	OVERSIZE
CUSTCLUSTER	130959360	2048	130959360	130959360	0	0
DISTCLUSTER	141389	2048	141389	141389	0	0
HIST	14336000	2048	11796480	0	9830400	2539520
ICUST1	4028900	2048	4028850	4028850	0	50
ICUST2	9152640	2048	9152640	9152640	0	0
IDIST	32768	2048	1323	1323	0	31445
IITEM	1000	2048	1000	1000	0	0
IORDR2	8314880	2048	6881280	6881280	0	1433600
ISTOK	12311040	2048	12311040	12311040	0	0
ITEMCLUSTER	8400	2048	8400	8400	0	0
IWARE	300	2048	263	263	0	37
NORDCLUSTER	2088960	2048	1344000	1344000	0	744960
ORDRCLUSTER	19968000	16384	18478080	0	15398400	1489920
STOKCLUSTER	227028480	2048	227028480	227028480	0	0
SYSTEM	102400	2048	102400	102400	0	0
WARECLUSTER	14175	2048	14175	14175	0	0

STATIC	DYNAMIC	OVERSIZE	DAILY_GROW		SPACE60	
783949200	266035200	33337944	53207040		3976371600	KB
				Required=	**3792.163467**	**GB**
		Numbers of disks	336	**Configured=**	**6048**	**GB**
		Capacity	18			

Log Space Calculation		
redo size after test	2.6E+11	
redo size before test	5.4092E+10	
redo size during the test	2.0591E+11	
total new orders during the test	41717358	
redo size per new order	4935.78716	bytes
redo size for 8 hours @ 136049.517	3.2233E+11	bytes
Required=	**300.189015**	**GB**
Required with Mirroring=	**600.37803**	
Number of disks	20	
Capacity	36	
Configured=	**720**	**GB**

Clause 5 Related Items

Throughput

Measured tpmC must be reported

Measured tpmC	136110.975tpmC
Price per tpmC	$3.94 per tpmC

Response Times

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

Table 5.1: Response Times

Type	Average	Maximum	90th %
New-Order	0.302	18.672	0.459
Payment	0.211	6.971	0.318
Order-Status	0.239	7.169	0.364
Interactive Delivery	0.101	0.176	0.102
Deferred Delivery	0.023	258.549	0.037
Stock-Level	0.193	1.830	0.289
Menu	0.101	0.257	0.102

Keying and Think Times

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

Table 5.2: Keying Times

Type	Minimum	Average	Maximum
New-Order	18.005	18.009	18.018
Payment	3.010	3.019	3.027
Order-Status	2.010	2.019	2.027
Interactive Delivery	2.010	2.019	2.027
Stock-Level	2.010	2.019	2.026

Table 5.3: Think Times

Type	Minimum	Average	Maximum
New-Order	0.000	12.085	120.752
Payment	0.000	12.015	120.085
Order-Status	0.000	10.014	99.823
Interactive Delivery	0.000	5.025	50.049
Stock-Level	0.000	5.014	49.782

Response Time Frequency Distribution Curves and Other Graphs

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

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

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

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

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

Figure 5.1: Response Times Frequency Distribution for New Order Transactions

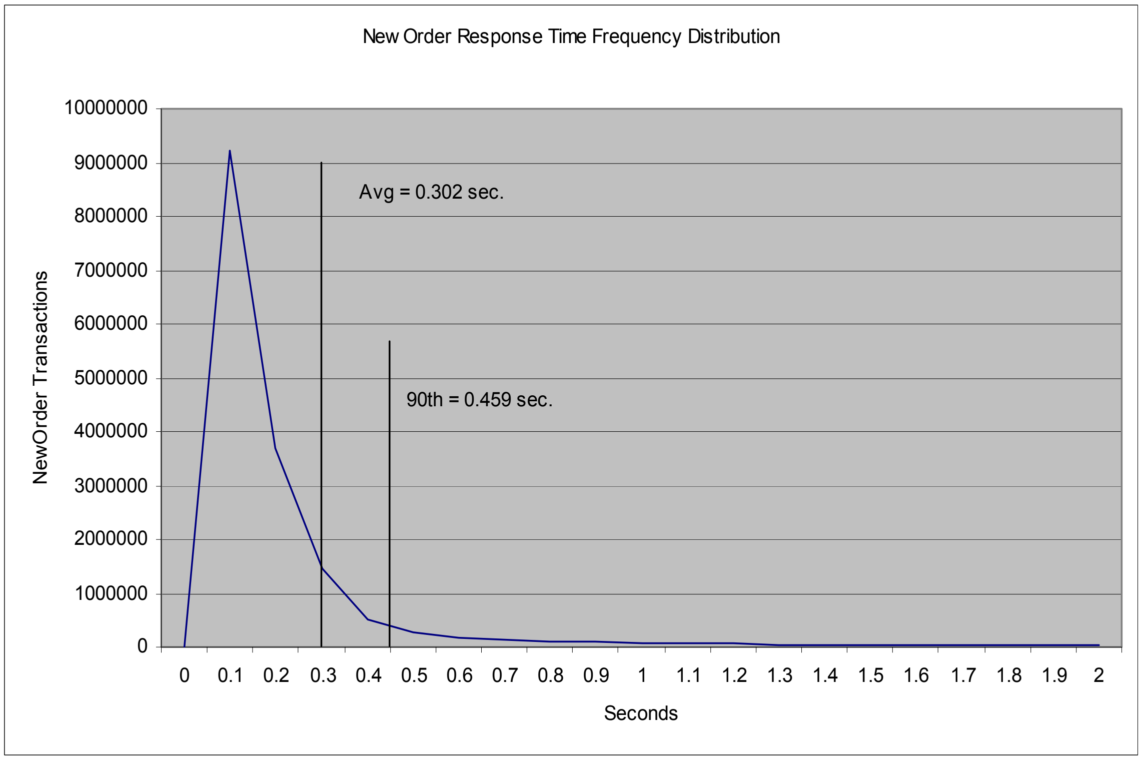


Figure 5.2: Response Times Frequency Distribution for Payment Transactions

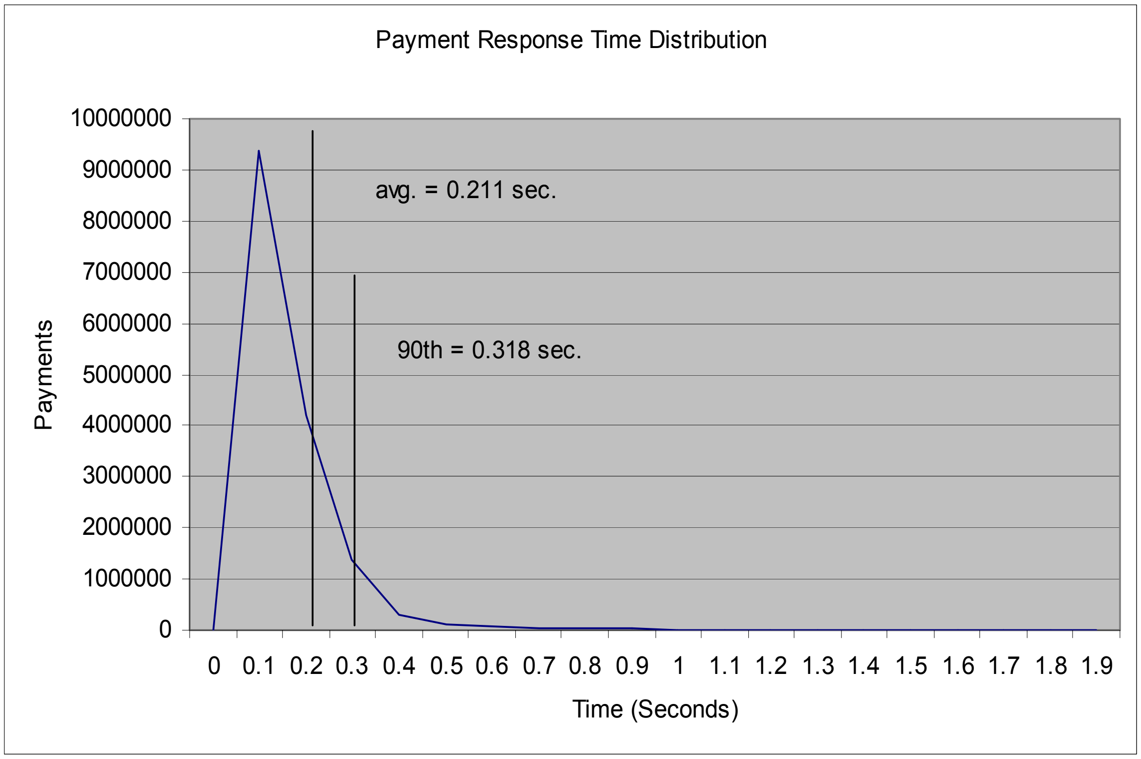


Figure 5.3: Response Times Frequency Distribution for Order Status Transactions

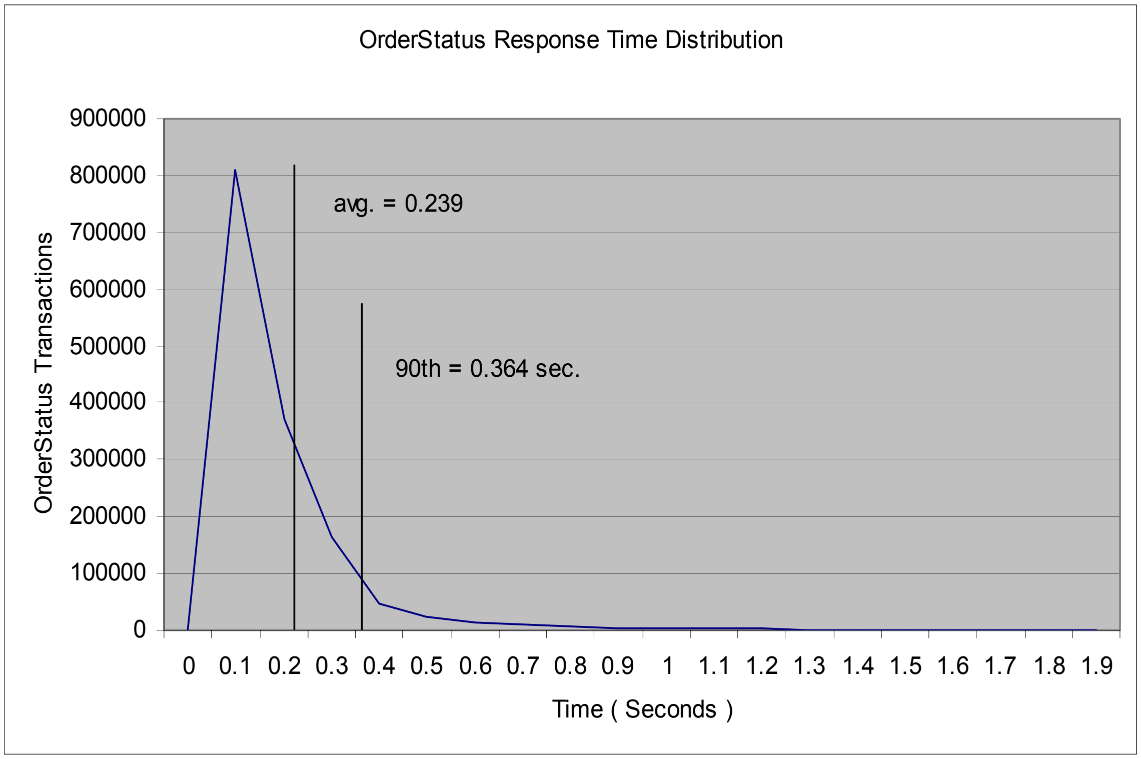


Figure 5.4: Response Times Frequency Distribution for Delivery Transactions

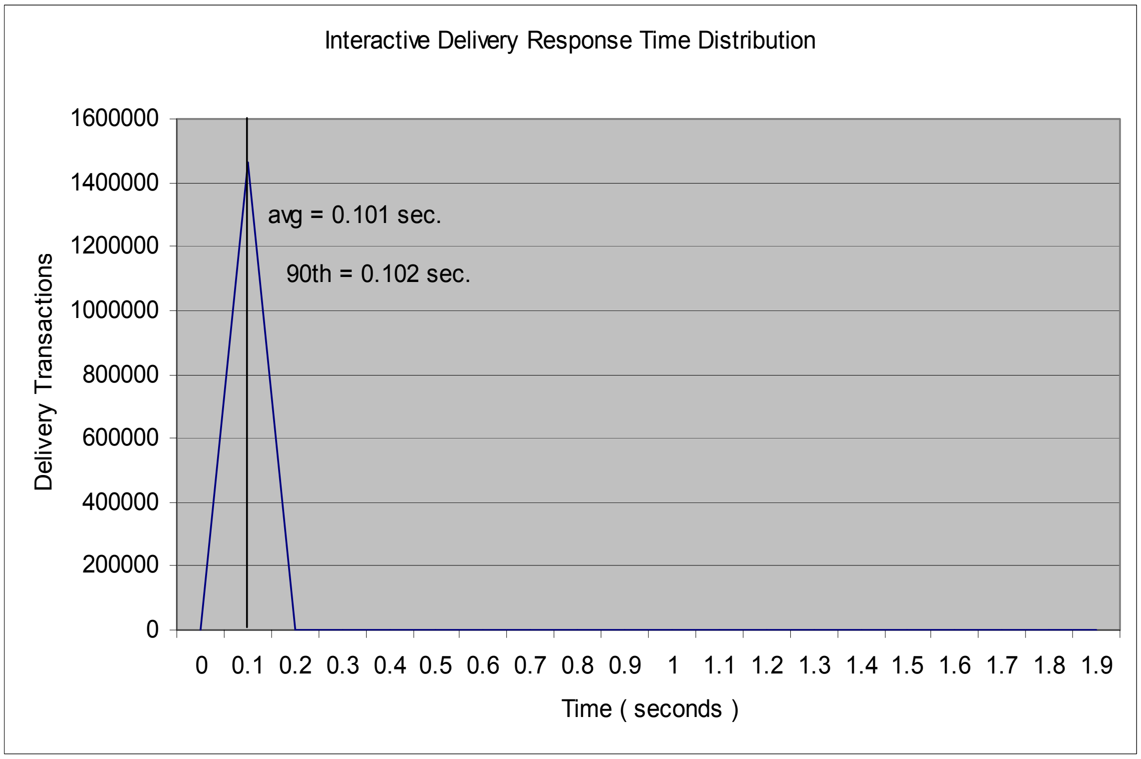


Figure 5.5: Response Times Frequency Distribution for Stock Level Transactions

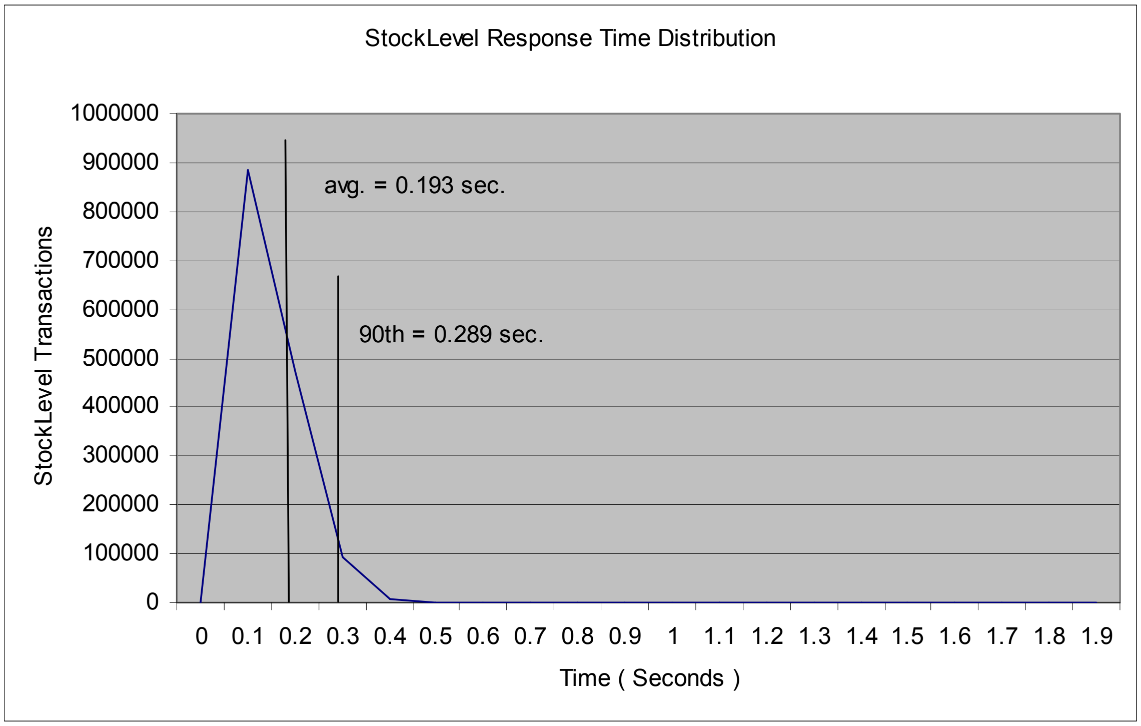


Figure 5.6: Response Time versus Throughput

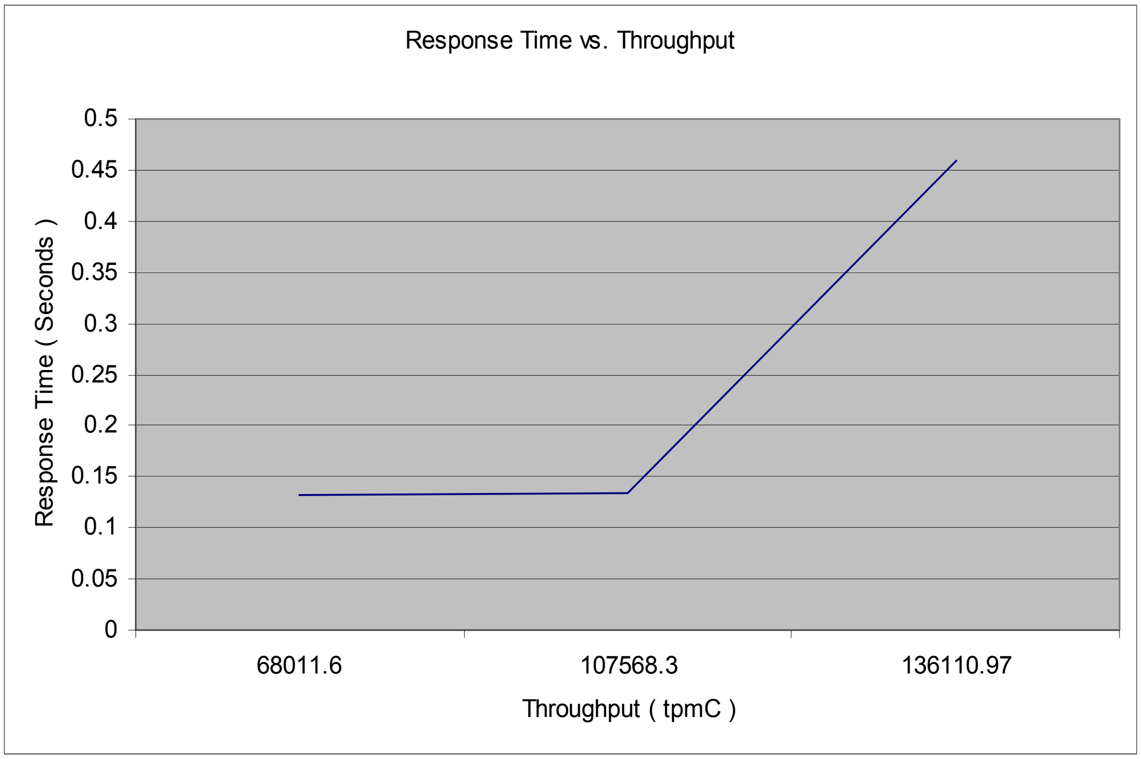


Figure 5.7: Think Times distribution for New Order Transactions

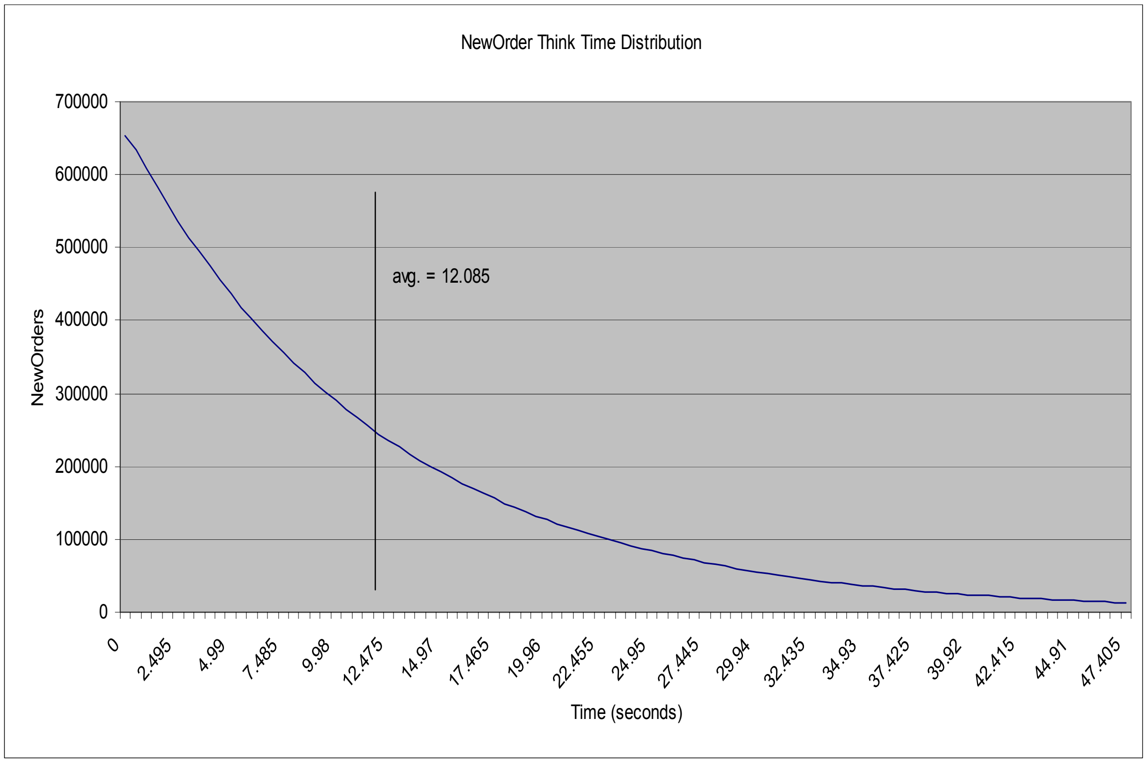
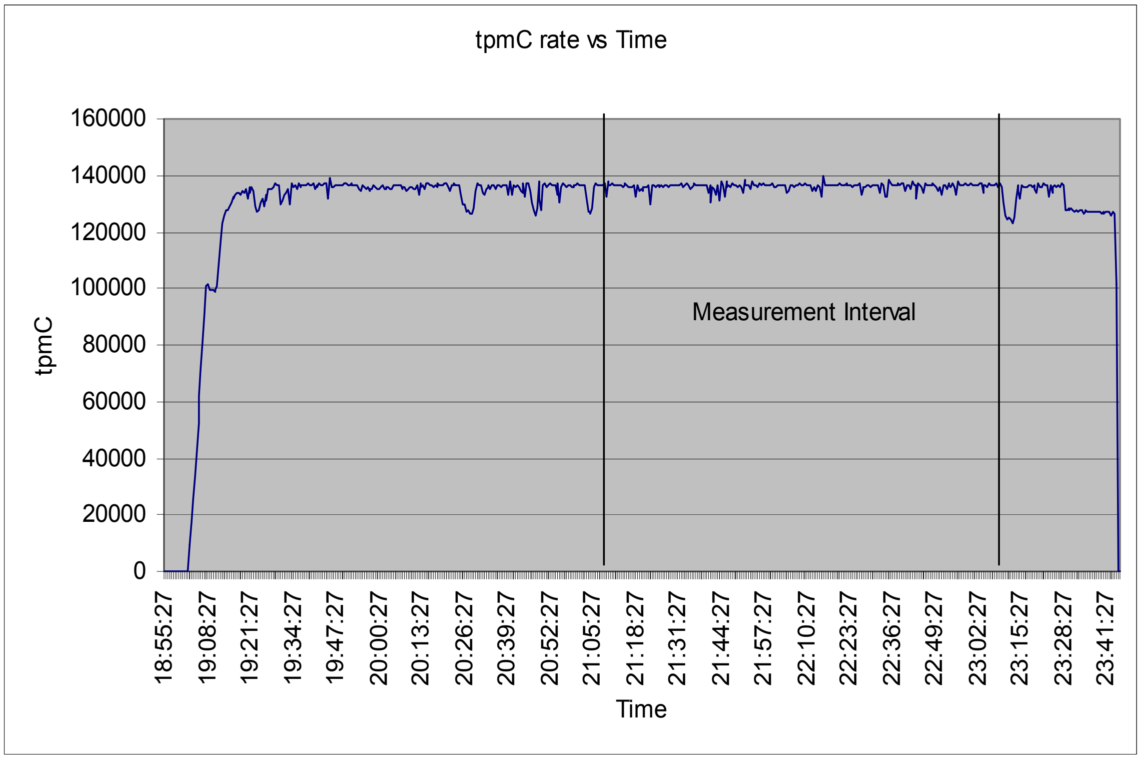


Figure 5.8: Throughput versus Time



Steady State Determination

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

Steady state was determined using real time monitor utilities from both the operating system and the RTE. Steady state was further confirmed by the throughput data collected during the run and graphed in Figure 5.8.

Work Performed During Steady State

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

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 Apache Web 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 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.

Measurement Period Duration

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

The reported measured interval was 7200 seconds.

Regulation of Transaction Mix

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

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

Transaction Statistics

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

Table 5.4: Transaction Statistics

Statistic		Value
New Order	Home warehouse order lines	99.00%
	Remote warehouse order lines	1.00%
	Rolled back transactions	1.00%
	Average items per order	10.00
Payment	Home warehouse	85.00%
	Remote warehouse	15.00%
	Accessed by last name	60.01%
Order Status	Accessed by last name	60.04%
Delivery	Skipped transactions	0
Transaction Mix	New Order	44.915%
	Payment	43.020%
	Order status	4.020%
	Delivery	4.025%
	Stock level	4.020%

Checkpoint Count and Location

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

A checkpoint is the process of writing all modified data pages to disk. The TPC-C benchmark on hp integrity rx5670 was set up to checkpoint within every 24 minutes. One checkpoint occurred during the warm-up period and 5 checkpoints occurred during the measurement period.

Checkpoint Duration

The start time and duration in seconds of at least the four longest checkpoints during the measurement Interval must be disclosed.

Start	End	Duration
19:53:12	20:16:00	0:22:48
20:18:23	20:41:54	0:23:31
20:44:18	21:07:46	0:23:28
21:10:08	21:32:57	0:22:49
21:35:17	21:58:09	0:22:52
22:00:30	22:23:32	0:23:02
22:25:55	22:49:05	0:23:10
22:51:28	23:14:50	0:23:22
23:17:15	23:41:22	0:24:07

Clause 6 Related Items

RTE Descriptions

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

PRTE Software was used to simulate terminal users, generate random data and record response times. This package ran on systems that are distinct from the system under test. PRTE command file used is included in Appendix A.

Emulated Components

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

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.

The driver system consisted of 10 ProLiant servers.

Functional Diagrams

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

The diagram in Section 1 shows the tested and priced benchmark configurations.

Networks

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

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

Section 1 of this report contains detailed diagrams of both the benchmark configuration and the priced configuration. In the tested configuration, the server system and ten client systems were connected to 8 port 1000BaseT Ethernet switch.
In the tested configuration there were ten driver systems (RTE), each of them connected to a client systems using 1000BaseT Ethernet switches.

Operator Intervention

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

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

Clause 7 Related Items

System Pricing

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

The total 3 year price of the entire configuration must be reported, including: hardware, software, and maintenance charges. Separate component pricing is recommended. The basis of all discounts used must be disclosed.

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

Availability, Throughput, and Price Performance

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

A statement of the measured tpmC as well as the respective calculations for the 3-year pricing, price/performance (price/tpmC), and the availability date must be included.

- **Maximum Qualified Throughput 136110.975 tpmC**
- **Price per tpmC $3.94 per tpmC**
- **Available March 5, 2004**
- **Hardware Available Now**

All hardware components are available now.

Country Specific Pricing

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

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

Usage Pricing

For any usage pricing, the sponsor must disclose:

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

The component pricing based on usage is shown below:

- Oracle Database 10g Standard Edition
- Red Hat Enterprise Linux AS 3
- Red Hat Enterprise Linux ES
- BEA Tuxedo CTS 8.1

Clause 9 Related Items

Auditor's Report

The auditor's name, address, phone number, and a copy of the auditor's attestation letter indicating compliance must be included in the Full Disclosure Report.

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

Lorna Livingtree
Performance Metrics Inc.
2229 Benita Dr. Suite 101
Rancho Cordova, CA 95670
916-635-2822

Availability of the Full Disclosure Report

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

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

Transaction Processing Performance Council
Presidio of San Francisco
Building 572B (surface)
P.O. Box 29920 (mail) San Francisco, CA 94129-0920
Voice: 415-561-6272
Fax: 415-561-6120
Email: info@tpc.org

or

Hewlett Packard Company
Database Performance Engineering
P.O. Box 692000
Houston, TX 77269-2000

TPC Benchmark C Full Disclosure Reports are available at www.tpc.org

September 4, 2003

Mr. Raghunath Othayoth and
Mr. Bryon Georgson
Database Performance Engineers
Hewlett-Packard Company
20555 SH 249
Houston, TX 77070

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

Platform:	HP Integrity rx5670 – 4P
Database Manager:	Oracle10i Database Standard Edition
Operating System:	Red Hat Linux Advanced Server IA64
Transaction Monitor:	BEA Tuxedo 8.1

System Under Test: HP Integrity rx 5670 with:				
CPU's	Memory	Disks (total)	90% Response	TpmC
4 Itanium 2 @ 1.5 Ghz	Main: 96 GB Cache: 6MB	336 @ 18.2GB 20 @ 36 GB 1 @ 36 GB (OS)	0.30	136,110.98

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

- The transactions were correctly implemented.
- The database files were properly sized.
- The database was properly scaled with 10,800 warehouses.
- The ACID properties were successfully demonstrated.
- Log loss and data loss durability were demonstrated on a subset of the SUT configured with a database properly populated for 2,000 warehouses.
- Input data was generated according to the specified percentages.
- Eight hours of mirrored log space was present on the tested system.
- Eight hours of growth space for the dynamic tables was present on the tested system.

- The data for the 60 day space calculation was verified.
- The controller cache for the log disks was disabled.
- The steady state portion of the test was 120 minutes.
- More than one checkpoint was taken before the measured interval opened.
- Four complete checkpoints was taken during the measured interval.
- The system pricing was checked for major components and maintenance.
- Third party quotes were verified for compliance.

Auditor Notes: None

Sincerely,

Lorna Livingtree

Lorna Livingtree
Auditor

Appendix A: Source Code

```
-------------------------------------------------
      Makefile
-------------------------------------------------

##
##  Makefile -- Build procedure for sample tpcc Apache module
##  Autogenerated via ``apxs -n tpcc -O2''.
##

builddir=.
top_srcdir=/usr/src/redhat/BUILD/httpd-2.0.36
top_builddir=/usr/src/redhat/BUILD/httpd-2.0.36
#include /usr/src/redhat/BUILD/httpd-2.0.36/build/special.mk

#   the used tools
APXS=/usr/sbin/apxs
#APXS=/usr/local/ap2/sbin/apxs
APACHECTL=/usr/sbin/apachectl
TUXDIR=/home/bea/tuxedo8.1
ORAHOME=/home/oracle/OraHome1

#   additional user defines, includes and libraries
#DEF=-Dmy_define=my_value
#LIB=-Lmy/lib/dir -lmylib
APACHEINC=-I/usr/include/httpd
#APACHEINC=-I/usr/local/ap2/include/apache
INC=-I. $(APACHEINC) $(ORAINC) $(TUXINC)
DEF=-Wall
TUXINC=-I/home/bea/tuxedo8.1/include
ORAINC=-I/home/oracle/OraHome1/rdbms/demo -
I/home/oracle/OraHome1/rdbms/public -
I/home/oracle/OraHome1/network/public

AP_LIBS =   $(top_builddir)/lib/libapr.a

TUX_LIBS =      $(TUXDIR)/lib/libtux.a \
                $(TUXDIR)/lib/libbuft.a \
                $(TUXDIR)/lib/libengine.a \
                $(TUXDIR)/lib/libtrpc.a \
                $(TUXDIR)/lib/libfml.a \
                $(TUXDIR)/lib/libfml32.a

LINUX_LIBS =  /usr/lib/libpthread.a \
    /usr/lib/libdl.a \
    /usr/lib/libm.a

ORA_LIBS =  -L$(ORAHOME)/rdbms/lib/ \
    -L$(ORAHOME)/lib/ \
    -lclntsh



TUX_SRV_OBJS =  tux_srv.o \
    oracle_db8.o \
    oracle_txns8.o \
    logfile_tux.o \
    util.o

MOD_TPCC_OBJS = mod_tpcc.o \
    logfile_mod.o \
    tpcc.o \
    tux_cli.o \
    util.o

#   the default target
#tpcc: local-shared-build

#   compile the DSO file
mod_tpcc.so: $(MOD_TPCC_OBJS)
  $(APXS) -Wc,-O2 -c $(DEF) $(INC) $(LIB) -L$(TUXDIR)/lib
$(MOD_TPCC_OBJS) -ltux -lbuft -lfml -lfml32 -lengine -ldl -lpthread

mod_tpcc.o: mod_tpcc.c
  gcc -O2 -c -DEAPI $(DEF) $(INC) $(LIB) mod_tpcc.c

logfile_mod.o: logfile_mod.c
  gcc -O2 -c $(DEF) $(INC) $(LIB) logfile_mod.c

logfile_tux.o: logfile_tux.c
  gcc -O2 -c $(DEF) $(INC) $(LIB) logfile_tux.c

tpcc.o: tpcc.c
  gcc -O2 -c $(DEF) $(INC) $(LIB) tpcc.c

util.o: util.c
  gcc -O2 -c $(DEF) $(INC) $(LIB) util.c

tux_cli.o: tux_cli.c
  gcc -O2 -c $(DEF) $(INC) $(LIB) tux_cli.c

oracle_db8.o: oracle_db8.c
  gcc -O2 -c $(DEF) $(INC) $(LIB) oracle_db8.c

oracle_txns8.o: oracle_txns8.c
  gcc -O2 -c $(DEF) $(INC) $(LIB) oracle_txns8.c

tux_srv.o: tux_srv.c
  gcc -O2 -c $(DEF) $(INC) $(LIB) tux_srv.c

delirpt: delirpt.c
  gcc -O2 -o delirpt delirpt.c

#tuxora: $(TUX_SRV_OBJS)
# gcc  $(TUX_SRV_OBJS) $(TUX_LIBS) -Wl,-rpath $(TUXDIR)/lib
$(ORAHOME)/lib/libclntst9.a $(LINUX_LIBS) -o tuxora

BS-7dc9.o: BS-7dc9.c
  gcc -c  -I$(TUXDIR)/include BS-7dc9.c

BS-deli.o: BS-deli.c
  gcc -c  -I$(TUXDIR)/include BS-deli.c

BS-deli1.o: BS-deli1.c
  gcc -c  -I$(TUXDIR)/include BS-deli1.c

BS-deli2.o: BS-deli2.c
  gcc -c  -I$(TUXDIR)/include BS-deli2.c

BS-deli3.o: BS-deli3.c
  gcc -c  -I$(TUXDIR)/include BS-deli3.c

BS-deli4.o: BS-deli4.c
  gcc -c  -I$(TUXDIR)/include BS-deli4.c

BS-deli5.o: BS-deli5.c
  gcc -c  -I$(TUXDIR)/include BS-deli5.c

BS-payo.o: BS-payo.c
  gcc -c  -I$(TUXDIR)/include BS-payo.c

BS-ordo.o: BS-ordo.c
  gcc -c  -I$(TUXDIR)/include BS-ordo.c

BS-stoo.o: BS-stoo.c
  gcc -c  -I$(TUXDIR)/include BS-stoo.c

BS-newo.o: BS-newo.c
  gcc -c  -I$(TUXDIR)/include BS-newo.c

BS-tpcc.o: BS-tpcc.c
  gcc -c  -I$(TUXDIR)/include BS-tpcc.c

tuxora: $(TUX_SRV_OBJS)
  gcc -o tuxora -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-7dc9.o -ltux -
lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS) $(ORA_LIBS)

tpccora: $(TUX_SRV_OBJS) BS-tpcc.o
  gcc -o tpccora -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-tpcc.o -ltux -
lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS) $(ORA_LIBS)

deliora: $(TUX_SRV_OBJS) BS-deli.o
  gcc -o deliora -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-deli.o -ltux -
lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS)  $(ORA_LIBS)

deliora1: $(TUX_SRV_OBJS) BS-deli1.o
  gcc -o deliora1 -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-deli1.o -ltux
-lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS)  $(ORA_LIBS)

deliora2: $(TUX_SRV_OBJS) BS-deli2.o
  gcc -o deliora2 -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-deli2.o -ltux
-lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS)  $(ORA_LIBS)

deliora3: $(TUX_SRV_OBJS) BS-deli3.o
  gcc -o deliora3 -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-deli3.o -ltux
-lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS)  $(ORA_LIBS)

deliora4: $(TUX_SRV_OBJS) BS-deli4.o
  gcc -o deliora4 -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-deli4.o -ltux
-lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS)  $(ORA_LIBS)

deliora5: $(TUX_SRV_OBJS) BS-deli5.o
  gcc -o deliora5 -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-deli5.o -ltux
-lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS)  $(ORA_LIBS)

stoora: $(TUX_SRV_OBJS) BS-stoo.o
```

```
  gcc -o stoora -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-stoo.o -ltux -
lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS) $(ORA_LIBS)

ordora: $(TUX_SRV_OBJS) BS-ordo.o
  gcc -o ordora -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-ordo.o -ltux -
lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS) $(ORA_LIBS)

payora: $(TUX_SRV_OBJS) BS-payo.o
  gcc -o payora -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-payo.o -ltux -
lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS) $(ORA_LIBS)

newora: $(TUX_SRV_OBJS) BS-newo.o
  gcc -o newora -L${TUXDIR}/lib $(TUX_SRV_OBJS) BS-newo.o -ltux -
lbuft -lfml -lfml32 -lengine -ldl -lpthread /usr/lib/libcrypt.a
$(LINUX_LIBS) $(ORA_LIBS)

tpccora:
#   install the shared object file into Apache
install: install-modules

replace:
  cp .libs/mod_tpcc.so /etc/httpd/modules
  cp tpccora $(TUXDIR)
  cp deliora? $(TUXDIR)

#installallclients:
# rcp [td]*ora cl101:/home/bea/tuxedo8.0
# rcp .libs/mod_tpcc.so cl101:/usr/local/ap2/lib/apache
# rcp [td]*ora cl102:/home/bea/tuxedo8.0
# rcp .libs/mod_tpcc.so cl102:/usr/local/ap2/lib/apache
# rcp [td]*ora cl103:/home/bea/tuxedo8.0
# rcp .libs/mod_tpcc.so cl103:/usr/local/ap2/lib/apache
# rcp [td]*ora cl104:/home/bea/tuxedo8.0
# rcp .libs/mod_tpcc.so cl104:/usr/local/ap2/lib/apache
# rcp [td]*ora cl105:/home/bea/tuxedo8.0
# rcp .libs/mod_tpcc.so cl105:/usr/local/ap2/lib/apache
# rcp [td]*ora cl106:/home/bea/tuxedo8.0
# rcp .libs/mod_tpcc.so cl106:/usr/local/ap2/lib/apache
# rcp [td]*ora cl107:/home/bea/tuxedo8.0
# rcp .libs/mod_tpcc.so cl107:/usr/local/ap2/lib/apache
# rcp [td]*ora cl108:/home/bea/tuxedo8.0
# rcp .libs/mod_tpcc.so cl108:/usr/local/ap2/lib/apache

#installcl78:
# scp [td]*ora cl78:/home/bea/tuxedo8.0
# scp .libs/mod_tpcc.so cl78:/usr/local/ap2/lib/apache

#   cleanup
clean:
  -rm -f mod_tpcc.o mod_tpcc.so

cleanall:
  -rm -f *.o .libs/mod_tpcc.so

#   simple test
test: reload
  lynx -mime_header http://localhost/tpcc

#   reload the module by installing and restarting Apache
reload: install restart

#   the general Apache start/restart/stop procedures
start:
  $(APACHECTL) start
restart:
  $(APACHECTL) restart
stop:
  $(APACHECTL) stop



-------------------------------------------------
      BS-7dc9.c
-------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void dy_transaction _((TPSVCINFO *));
extern void no_transaction _((TPSVCINFO *));
extern void os_transaction _((TPSVCINFO *));
extern void pt_transaction _((TPSVCINFO *));
extern void sl_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"dy_transaction1", (char*)"dy_transaction1", (void (*)
_((TPSVCINFO *))) dy_transaction, 0, 0 },
  { (char*)"no_transaction", (char*)"no_transaction", (void (*)
_((TPSVCINFO *))) no_transaction, 1, 0 },
  { (char*)"os_transaction", (char*)"os_transaction", (void (*)
_((TPSVCINFO *))) os_transaction, 2, 0 },
  { (char*)"pt_transaction", (char*)"pt_transaction", (void (*)
_((TPSVCINFO *))) pt_transaction, 3, 0 },
  { (char*)"sl_transaction", (char*)"sl_transaction", (void (*)
_((TPSVCINFO *))) sl_transaction, 4, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



-------------------------------------------------
      BS-deli1.c
-------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void dy_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"dy_transaction1", (char*)"dy_transaction1", (void (*)
_((TPSVCINFO *))) dy_transaction, 0, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
```

```
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



-------------------------------------------------
      BS-deli2.c
-------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void dy_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"dy_transaction2", (char*)"dy_transaction2", (void (*)
_((TPSVCINFO *))) dy_transaction, 0, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



-------------------------------------------------
      BS-deli3.c
-------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void dy_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"dy_transaction3", (char*)"dy_transaction3", (void (*)
_((TPSVCINFO *))) dy_transaction, 0, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



------------------------------------------------
      BS-deli4.c
------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void dy_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"dy_transaction4", (char*)"dy_transaction4", (void (*)
_((TPSVCINFO *))) dy_transaction, 0, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



------------------------------------------------
      BS-deli5.c
------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void dy_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"dy_transaction5", (char*)"dy_transaction5", (void (*)
_((TPSVCINFO *))) dy_transaction, 0, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



------------------------------------------------
      BS-newo.c
------------------------------------------------

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

#if defined(__cplusplus)
```

```
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void no_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"no_transaction", (char*)"no_transaction", (void (*)
_((TPSVCINFO *))) no_transaction, 0, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



-------------------------------------------------
      BS-ordo.c
-------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void os_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"os_transaction", (char*)"os_transaction", (void (*)
_((TPSVCINFO *))) os_transaction, 0, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



-------------------------------------------------
      BS-payo.c
-------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void pt_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"pt_transaction", (char*)"pt_transaction", (void (*)
_((TPSVCINFO *))) pt_transaction, 0, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



-------------------------------------------------
      BS-stoo.c
-------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void sl_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"sl_transaction", (char*)"sl_transaction", (void (*)
_((TPSVCINFO *))) sl_transaction, 0, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



-------------------------------------------------
      BS-tpcc.c
-------------------------------------------------

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

#if defined(__cplusplus)
extern "C" {
#endif
extern int _tmrunserver _((int));
extern void no_transaction _((TPSVCINFO *));
extern void os_transaction _((TPSVCINFO *));
extern void pt_transaction _((TPSVCINFO *));
extern void sl_transaction _((TPSVCINFO *));
#if defined(__cplusplus)
}
#endif

static struct tmdsptchtbl_t _tmdsptchtbl[] = {
  { (char*)"no_transaction", (char*)"no_transaction", (void (*)
_((TPSVCINFO *))) no_transaction, 0, 0 },
  { (char*)"os_transaction", (char*)"os_transaction", (void (*)
_((TPSVCINFO *))) os_transaction, 1, 0 },
  { (char*)"pt_transaction", (char*)"pt_transaction", (void (*)
_((TPSVCINFO *))) pt_transaction, 2, 0 },
  { (char*)"sl_transaction", (char*)"sl_transaction", (void (*)
_((TPSVCINFO *))) sl_transaction, 3, 0 },
  { NULL, NULL, NULL, 0, 0 }
};

#ifndef _TMDLLIMPORT
#define _TMDLLIMPORT
#endif

#if defined(__cplusplus)
extern "C" {
#endif
_TMDLLIMPORT extern struct xa_switch_t tmnull_switch;
#if defined(__cplusplus)
}
#endif

typedef void (*tmp_void_cast)();
typedef void (*tmp_voidvoid_cast)(void);
typedef int (*tmp_intchar_cast)(int, char **);
typedef int (*tmp_int_cast)(int);
static struct tmsvrargs_t tmsvrargs = {
  NULL,
  &_tmdsptchtbl[0],
  0,
  (tmp_intchar_cast)tpsvrinit,
  (tmp_voidvoid_cast)tpsvrdone,
  (tmp_int_cast)_tmrunserver, /* PRIVATE  */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  NULL,     /* RESERVED */
  (tmp_intchar_cast)tpsvrthrinit,
  (tmp_voidvoid_cast)tpsvrthrdone
};

struct tmsvrargs_t *
#ifdef _TMPROTOTYPES
_tmgetsvrargs(void)
#else
_tmgetsvrargs()
#endif
{
  tmsvrargs.reserved1 = NULL;
  tmsvrargs.reserved2 = NULL;
  tmsvrargs.xa_switch = &tmnull_switch;
  return(&tmsvrargs);
}

int
#ifdef _TMPROTOTYPES
main(int argc, char **argv)
#else
main(argc,argv)
int argc;
char **argv;
#endif
{
#ifdef TMMAINEXIT
#include "mainexit.h"
#endif

  return( _tmstartserver( argc, argv, _tmgetsvrargs()));
}



-------------------------------------------------
      delirpt.c
-------------------------------------------------

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

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

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

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

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

#define TRUE  1
#define FALSE 0

typedef int BOOL;

typedef struct _DelTime
{
  struct tm dtime;
  int   wMilliseconds;
} DelTime;

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

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

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

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


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

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

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

  if ( (iError=Init()) != ERR_SUCCESS )
  {
    ErrorMessage(iError);
    Restore();
    return -1;
  }

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

  Restore();

  return 0;
}

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

static int Init(void)
{
  int iError;

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

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

static void Restore(void)
```

```
{
  CloseLogFile();
  return;
}

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

static int DoReport(void)
{
  int iRc;

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

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

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

  ResetLogFile();

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

      if ( iLines % 10 == 0 )
        printf("Reading Report Line:\t%d\r", iLines);
    }
  }
  printf("                                        \r");
  if ( iLines == 0 )
  {
    printf("No deliveries found.\n");
  }
  else
  {
    fAverage = (iTotalResponse / iLines)/1000.0;
    printf("Total Deliveries:       %u\n", iLines);
    printf("Total Response Times:  %10.3f (sec)\n",
(iTotalResponse/1000.0));
    printf("Average Response Time: %10.3f (sec)\n", fAverage);
  }
```

```
  return ERR_SUCCESS;
}

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

int Percentile90th(void)
{
  RPTLINE reportLine;
  int   iBucketSize;
  int   i;
  long    iMaxSeconds;
  int   iTotalBuckets;
  double  iTotal;
  double  i90thPercent;
  short *psBuckets;
  char  szDelivery[128];

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

  ResetLogFile();

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

  printf("Max Response Time = %f (sec)\n", iMaxSeconds/1000.0);

  iTotalBuckets = iMaxSeconds + 2;

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

  iBucketSize = iTotalBuckets * sizeof(short);

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

/**
  ZeroMemory(psBuckets, iBucketSize);
**/

  for (i=0; i < iTotalBuckets; i++)
    psBuckets[i]=0;


  iTotal = 0;

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

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

  printf("Done filling buckets\n");
  fflush(stdout);

  i90thPercent = iTotal * .9;

  printf(" i90thPercent = %f\n", i90thPercent );
  fflush(stdout);
  for(i=0, iTotal = 0.0; iTotal < i90thPercent; iTotal +=
(double)psBuckets[i] )
    i++;

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

  free(psBuckets);

  return ERR_SUCCESS;
}

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

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

  ResetLogFile();

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

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

  return ERR_SUCCESS;
}

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

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

  iRptStartTime = (pRptLine->start.dtime.tm_hour * 3600000) +
(pRptLine->start.dtime.tm_min * 60000) + (pRptLine-
>start.dtime.tm_sec * 1000) + pRptLine->start.wMilliseconds;
  iRptEndTime = (pRptLine->end.dtime.tm_hour * 3600000) +
(pRptLine->end.dtime.tm_min * 60000) + (pRptLine->end.dtime.tm_sec
* 1000) + pRptLine->end.wMilliseconds;

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

  if ( !OverMidnight ) {
    if ( iStartTime <= iRptStartTime && iEndTime >= iRptEndTime )
      return FALSE;
  }
  else {
    if ( pRptLine->day == StartDay ) {
      if ( iStartTime <= iRptStartTime )
        return FALSE;
    }
    else {
      if ( iEndTime >= iRptEndTime )
        return FALSE;
    }
  }

  return TRUE;
}

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

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

  if ( !fpLog )
    return ERR_CANNOT_OPEN_RESULTS_FILE;

  return ERR_SUCCESS;
}

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

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

  return;
}


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

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

  return;
}

/* FUNCTION: static BOOL LogEOF(int iOperation)
 *
 * PURPOSE: This function tracks and reports the end of file
condition
 *      on the delilog file.
 *
 * ARGUMENTS: int iOperation  requested operation this can be:
 *                  LOGFILE_READ_EOF  check log file flag return
current state
 *                  LOGFILE_CLEAR_EOF clear end of log file flag
 *                  LOGFILE_SET_EOF   set flag end of log file
reached
 *
 *
 * RETURNS:   None
 *
 * COMMENTS:  None
 *
 */

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

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

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

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

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

  }

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

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

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

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

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

  pRptLine->end.dtime.tm_year = pRptLine->start.dtime.tm_year;
  pRptLine->end.dtime.tm_mon = pRptLine->start.dtime.tm_mon;
  pRptLine->end.dtime.tm_mday = pRptLine->start.dtime.tm_mday;

  pRptLine->day=(pRptLine->start.dtime.tm_mon*100) + pRptLine-
>start.dtime.tm_mday;
  if (StartDay == 0) {
    StartDay=pRptLine->day;
    printf("Setting Start Day to %d\n", StartDay);
  }

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

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

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

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

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

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

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

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

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

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

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

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

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

  return FALSE;
}

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

static BOOL ParseDate(char *szDate, DelTime *pTime)
{
  if ( !isdigit(*szDate) || !isdigit(*(szDate+1)) ||
!isdigit(*(szDate+2)) || !isdigit(*(szDate+3)) || *(szDate+4) !=
'/' ||
     !isdigit(*(szDate+5)) || !isdigit(*(szDate+6)) || *(szDate+7)
!= '/' ||
     !isdigit(*(szDate+8)) || !isdigit(*(szDate+9)) )
     return TRUE;
  pTime->dtime.tm_year = atoi(szDate);

  pTime->dtime.tm_mon= atoi(szDate+5);

  pTime->dtime.tm_mday = atoi(szDate+8);

  if ( pTime->dtime.tm_mon > 12 || pTime->dtime.tm_mon < 0 ||
pTime->dtime.tm_mday > 31 || pTime->dtime.tm_mday < 0 )
    return TRUE;

  return FALSE;
}

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

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

  pTime->dtime.tm_hour = atoi(szTime);
  pTime->dtime.tm_min = atoi(szTime+3);
  pTime->dtime.tm_sec = atoi(szTime+6);
  pTime->wMilliseconds = atoi(szTime+9);

  if ( pTime->dtime.tm_hour > 23 || pTime->dtime.tm_hour < 0 ||
    pTime->dtime.tm_min > 59 || pTime->dtime.tm_min < 0 ||
    pTime->dtime.tm_sec > 59 || pTime->dtime.tm_sec < 0 ||
    pTime->wMilliseconds < 0 )
    return TRUE;

  if ( pTime->wMilliseconds > 999 )
  {
    pTime->dtime.tm_sec += (pTime->wMilliseconds/1000);
    pTime->wMilliseconds = pTime->wMilliseconds  % 1000;
  }

  return FALSE;
}

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

static void ErrorMessage(int iError)
{
  int i;

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

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

/* FUNCTION: BOOL GetParameters(int argc, char *argv[])
 *
 * PURPOSE: This function parses the command line passed in to the
delivery executable, initializing
 *      and filling in global variable parameters.
 *
 * ARGUMENTS: int   argc  number of command line arguments passed
to delivery
 *        char  *argv[] array of command line argument pointers
 *
 * RETURNS:   BOOL  FALSE parameter read successfull
 *            TRUE  user has requested parameter information screen
be displayed.
 *
 * COMMENTS:  None
 *
 */

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

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

  for(i=0; i<argc; i++)
  {
    if ( argv[i][0] == '-' || argv[i][0] == '/' )
    {
      switch(argv[i][1])
      {
        case 'S':
        case 's':
          if ( ParseTime(argv[i]+2, &startTime) )
            return TRUE;
          iStartTime = (startTime.dtime.tm_hour * 3600000) +
(startTime.dtime.tm_min * 60000) + (startTime.dtime.tm_sec * 1000)
+ startTime.wMilliseconds;
          break;
        case 'E':
        case 'e':
          if ( ParseTime(argv[i]+2, &endTime) )
            return TRUE;
          iEndTime = (endTime.dtime.tm_hour * 3600000) +
(endTime.dtime.tm_min * 60000) + (endTime.dtime.tm_sec * 1000) +
endTime.wMilliseconds;
          if (iStartTime > iEndTime)
            OverMidnight=1;
          break;
        case 'R':
        case 'r':
          iReport = atoi(argv[i]+2);
          if ( iReport > 4 || iReport < 1 )
            iReport = 4;
          break;
        case '?':
          return TRUE;
      }
    }
  }
  return FALSE;
}

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

static void PrintParameters(void)
{
  printf("DELIRPT:\n\n");
  printf("Parameter
Default\n");
  printf("---------------------------------------------------------
-------------\n");
  printf("-S Start Time HH:MM:SS:MMM
All    \n");
  printf("-E End Time HH:MM:SS:MMM
All    \n");
  printf("-R 1)Average Response, 2)90th 3) Skipped 4) All
All    \n");
  printf("-? This help screen\n\n");
  printf("Note:  Command line switches are NOT case sensitive.\n");

  return;
}
```

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

static void cls(void)
{
  system("clear");

  return;
}

/* FUNCTION: BOOL IsNumeric(char *ptr)
 *
 * PURPOSE: This function determines if a string is numeric. It
fails if any characters other
 *      than numeric and null terminator are present.
 *
 * ARGUMENTS: char      *ptr  pointer to string to check.
 *
 * RETURNS:   BOOL  FALSE if string is not all numeric
 *            TRUE  if string contains only numeric characters i.e.
'0' - '9'
 *
 * COMMENTS:  A comma is counted as a valid delimiter.
 *
 */

static BOOL IsNumeric(char *ptr)
{
  if ( *ptr == 0 )
    return FALSE;

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


------------------------------------------------
      logfile_mod.c
------------------------------------------------

/*+******************************************************************
*********
 *
*

 *  COPYRIGHT (c) 1997 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: W Carr
 * Creation Date: October 1997
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                      - Conversion to run under Linux and Apache
 *
*/

#include <stdio.h>
#include <stdarg.h>
#include <time.h>
#include <sys/time.h>
#include <errno.h>
#include <unistd.h>

#include "apr_thread_mutex.h"

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

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

#include <tpcc.h>

static FILE *LogFile;

static char   t1[1];
static apr_thread_mutex_t * ErrCriticalSection;
static apr_thread_mutex_t * LogCriticalSection;


/* FUNCTION: void TPCCOpenLog( void )
 *
 * PURPOSE: This function opens the log file.
 *
 * ARGUMENTS: None
 *
 * RETURNS: None
 *
 * COMMENTS:  None
 *
 */
BOOL
TPCCOpenLog( apr_pool_t *pool )
{
  char      szFile[FILENAMESIZE];

  apr_thread_mutex_create(&LogCriticalSection, 0, pool);

  strcpy( szFile, szTpccLogPath );
  strcat( szFile, "tpcclog" );

  if (LogFile = fopen( szFile, "a")) {
    apr_thread_mutex_create(&ErrCriticalSection, 0, pool);
    return TRUE;
  }
  else
  {
    return FALSE;
  }
}


/* FUNCTION: void TPCCCloseLog( void )
 *
 * PURPOSE: This function closes the log file.
 *
 * ARGUMENTS: None
 *
 * RETURNS: None
 *
 * COMMENTS:  None
 *
 */
BOOL
TPCCCloseLog( void )
{
  fclose( LogFile );

  return TRUE;
}

/* FUNCTION: void TPCCLog( char *szType, char *szStr )
 *
 * PURPOSE: This function reports the date, time, operation and
 *    string to the log file.
 *
 * ARGUMENTS: char    *szType String containing the operation type
 *        i.e. Query or Response.
 *    char  *szStr  String associated with the operation.
 *
 * RETURNS: None
 *
 * COMMENTS:  None
 *
 */
void
TPCCLog( char *fmt, ... )
{
  va_list   marker;
  char      szArg[4096];
  struct timezone       tz;
  struct timeval        tv;
  struct tm             systemTime;
  struct tm             *pst;
  int                   len, ret;

  va_start( marker, fmt );
  vsprintf( szArg, fmt, marker );
  va_end( marker );

  pst=&systemTime;
  ret=gettimeofday(&tv, &tz);

  apr_thread_mutex_lock( LogCriticalSection );

  pst=localtime(&tv.tv_sec);

  len = fprintf( stderr,
     "[%ld] %2.2d/%2.2d/%2.2d %2.2d:%2.2d:%2.2d\t%s\r\n",
     getpid(),
     1900+pst->tm_year, pst->tm_mon+1, pst->tm_mday,
     pst->tm_hour, pst->tm_min, pst->tm_sec,
     szArg );
  apr_thread_mutex_unlock( LogCriticalSection );
}

void
TPCCErrInternal( char *szTmp, int len )
{
  int     dwWriteLen;
  FILE      *ErrFile;
  char      szFile[FILENAMESIZE];

  apr_thread_mutex_lock( ErrCriticalSection );

  strcpy( szFile, szTpccLogPath );
  strcat( szFile, "tpccerr" );

  ErrFile = fopen( szFile, "a");

  if (ErrFile) {
    len = fprintf( ErrFile, "%s\n", szTmp);
    fclose( ErrFile );
  }

  apr_thread_mutex_unlock( ErrCriticalSection );
}


void
TPCCErr( char *fmt, ...)
{
  va_list   marker;
  char      szTmp[4096];
  char      szArg[4096];
  struct timezone tz;
  struct timeval  tv;
  struct tm   systemTime;
  struct tm   *pst;
  int     len, ret;

  va_start( marker, fmt );
  vsprintf( szArg, fmt, marker );
  va_end( marker );

  pst=&systemTime;
  ret=gettimeofday(&tv, &tz);
  pst=localtime(&tv.tv_sec);

  len = sprintf( szTmp,
     "%2.2d/%2.2d/%2.2d %2.2d:%2.2d:%2.2d\t%s\r\n",
     1900+pst->tm_year, pst->tm_mon+1, pst->tm_mday,
     pst->tm_hour, pst->tm_min, pst->tm_sec,
     szArg );

  TPCCErrInternal( szTmp, len );
}


void
TPCCTransactionErr( pConnData pConn, char *fmt, ...)
{
  va_list   marker;
  char      szTmp[4096];
  char      szArg[4096];
  struct timezone       tz;
  struct timeval        tv;
  struct tm             systemTime;
  struct tm             *pst;
  int                   len, ret;

  va_start( marker, fmt );
  vsprintf( szArg, fmt, marker );
  va_end( marker );

  pst=&systemTime;
  ret=gettimeofday(&tv, &tz);
  pst=localtime(&tv.tv_sec);
  len = sprintf( szTmp,
     "%2.2d/%2.2d/%2.2d %2.2d:%2.2d:%2.2d\tTransaction error. w_id:
%d, ld_id: %d, pCC: %x, status: %d, dbstatus: %d, %s\r\n",
     1900+pst->tm_year, pst->tm_mon+1, pst->tm_mday,
     pst->tm_hour, pst->tm_min, pst->tm_sec,
     pConn->w_id, pConn->ld_id, pConn->pCC,
     pConn->status, pConn->dbstatus,
     szArg );

  TPCCErrInternal( szTmp, len );
}


------------------------------------------------
      logfile_tux.c
------------------------------------------------

/*+*****************************************************************
*********
 *
*
 *  COPYRIGHT (c) 1997 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: W Carr
 * Creation Date: October 1997
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                      - Conversion to run under Linux and Apache
 *
*/

#include <stdio.h>
#include <stdarg.h>
#include <time.h>
#include <sys/time.h>

#include <tpccstruct.h>

static FILE *LogFile;
```

```
void
TPCCErr( char *fmt, ...)
{
  va_list   marker;
  char      szTmp[4096];
  char      szArg[4096];
  struct timezone tz;
  struct timeval  tv;
  struct tm   systemTime;
  struct tm   *pst;
  int     len, ret;

  va_start( marker, fmt );
  vsprintf( szArg, fmt, marker );
  va_end( marker );

  pst=&systemTime;
  ret=gettimeofday(&tv, &tz);
  pst=localtime(&tv.tv_sec);

  len = userlog( "%2.2d/%2.2d/%2.2d %2.2d:%2.2d:%2.2d\t%s\r\n",
     1900+pst->tm_year, pst->tm_mon+1, pst->tm_mday,
     pst->tm_hour, pst->tm_min, pst->tm_sec,
     szArg );

  if (len < 0)
  printf("TPCCErr: Error writing to Tuxedo userlog\n");

}


-------------------------------------------------
      mod_tpcc.c
-------------------------------------------------

/*+*****************************************************************
**********
 *
*
 *  COPYRIGHT (c) 1997 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
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                      - Conversion to run under Linux and Apache
 *      - Additions by Joe Orton to support Apache 2.0
*/
#include "httpd.h"
#include "http_config.h"
#include "http_protocol.h"
#include "ap_config.h"
#include "ap_mpm.h"
#include "apr_thread_mutex.h"

#include <stdio.h>
#include <stdarg.h>
#include <malloc.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

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

#define MOD_TPCC_C

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

#include <tpcc.h>
#include <mod_tpcc.h>

#ifdef FFE_DEBUG
# include <crtdbg.h>
static int tmpDbgFlag;
static _HFILE hMemFile;
#endif

int tpcc_handler(request_rec *req);
static int tpcc_post_config(apr_pool_t *p, apr_pool_t *plog,
apr_pool_t *ptemp, server_rec *s);
static void tpcc_child_init(apr_pool_t *p, server_rec *s);
static apr_status_t tpcc_child_exit(void *data);

#define FORMMAXSIZE 4096

#define MYFILE  "/etc/httpd/logs/tpcc.log"
#define BOGUS "Bogus File!"
#define GOOD "Good File!"

int     LogFD;
int myerrno;
int max_threads;

static void tpcc_register_hooks(apr_pool_t *p)
{
  fprintf(stderr, "register()");

    ap_hook_handler(tpcc_handler, NULL, NULL, APR_HOOK_MIDDLE);
    ap_hook_post_config(tpcc_post_config, NULL, NULL,
APR_HOOK_MIDDLE);
/*
    ap_hook_child_init(tpcc_child_init, NULL, NULL,
APR_HOOK_MIDDLE);
*/
}

/* Dispatch list for API hooks */
module AP_MODULE_DECLARE_DATA tpcc_module = {
    STANDARD20_MODULE_STUFF,
    NULL,                  /* create per-dir    config structures
*/
    NULL,                  /* merge  per-dir    config structures
*/
    NULL,                  /* create per-server config structures
*/
    NULL,                  /* merge  per-server config structures
*/
    NULL,                  /* table of config file commands
*/
    tpcc_register_hooks  /* register hooks                       */
};

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

#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_D_STR_2 PT_CREDIT + 1
#define PT_DISC PT_D_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 WDID(w_id,d_id) (w_id*10+(d_id-1))

#define PANIC_FORM_SIZE 4096

#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)\
  {\
    apr_thread_mutex_lock( gpForms->critSec[type] );\
    FFE_ASSERT( gpForms->iNextFreeForm[type] <= gpForms-
>iMaxIndex[type] );\
    szForm = gpForms->index[gpForms->iFirstFormIndex[type] +\
          gpForms->iNextFreeForm[type]++];\
    apr_thread_mutex_unlock( gpForms->critSec[type] );\
  }
#define UNRESERVE_FORM(type,szForm)\
  {\
    apr_thread_mutex_lock( gpForms->critSec[type] );\
    FFE_ASSERT( gpForms->iNextFreeForm[type] > 0 );\
    gpForms->index[gpForms->iFirstFormIndex[type] +\
       --gpForms->iNextFreeForm[type]] = szForm;\
    apr_thread_mutex_unlock( gpForms->critSec[type] );\
  }

#define RESERVE_RESPONSE(type,szResponse)\
  {\
    apr_thread_mutex_lock( gpResponses->critSec[type] );\
    FFE_ASSERT(gpResponses->iNextFreeResponse[type]<=gpResponses-
>iMaxIndex[type]);\
    szResponse = gpResponses->index[gpResponses-
>iFirstResponseIndex[type] +\
             gpResponses->iNextFreeResponse[type]++];\
    apr_thread_mutex_unlock( gpResponses->critSec[type] );\
  }
#define UNRESERVE_RESPONSE(type,szResponse)\
  {\
    apr_thread_mutex_lock( gpResponses->critSec[type] );\
    FFE_ASSERT(gpResponses->iNextFreeResponse[type] > 0 );\
    gpResponses->index[gpResponses->iFirstResponseIndex[type] +\
      --gpResponses->iNextFreeResponse[type]] = szResponse;\
    apr_thread_mutex_unlock( gpResponses->critSec[type] );\
  }

#define RESERVE_PANIC_FORM(szForm)\
  {\
    apr_thread_mutex_lock( gpPanicForms->critSec );\
    FFE_ASSERT( gpPanicForms->iNextFree <= gpPanicForms->iMaxIndex
);\
    szForm = gpPanicForms->index[gpPanicForms->iNextFree++];\
    apr_thread_mutex_unlock( gpPanicForms->critSec );\
  }

#define UNRESERVE_PANIC_FORM(szForm)\
  {\
    apr_thread_mutex_lock( gpPanicForms->critSec );\
    FFE_ASSERT( gpPanicForms->iNextFree > 0 );\
    gpPanicForms->index[--gpPanicForms->iNextFree] = szForm;\
    apr_thread_mutex_unlock( 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 NAME=3 VALUE=W00>"
"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></BODY>";

static char
szWelcomeForm[sizeof(szWelcomeFormTemplate)+FILENAMESIZE];
static int  iWelcomeFormLen;

static char szMainMenuFormTemplate[] =
"<BODY><FORM ACTION=%s METHOD=GET>"
"<INPUT TYPE=hidden NAME=3 VALUE=M%06d>"
"%55.55s<BR>"
"Select Desired Transaction.<BR>"
MENU_BAR
"</FORM></BODY>";

static char szDeliveryFormTemp2i[] =
"<INPUT TYPE=hidden NAME=3 VALUE=D######>"
"<INPUT TYPE=hidden NAME=6 VALUE=0>"
"<PRE>                                  Delivery<BR>"
"Warehouse: #####<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></BODY>";

static char szDeliveryFormTemp2p[] =
"<INPUT TYPE=hidden NAME=3 VALUE=d######>"
"<PRE>                                  Delivery<BR>"
"Warehouse: #####<BR><BR>"
"Carrier Number: ##<BR><BR>"
"Execution Status: Delivery has been queued.<BR>"
"</PRE>"
MENU_BAR
"</FORM></BODY>";

static char szNewOrderFormTemp2i[] =
"<INPUT TYPE=hidden NAME=3 VALUE=N######>"
"<PRE>                                 New Order<BR>"
"Warehouse: #####"
"  District: <INPUT NAME=8 SIZE=2>
Date:<BR>"
"Customer:  <INPUT NAME=9 SIZE=4>   "
"Name:                    Credit:      %Disc:<BR>"
"Order Number:           Number of Lines:           "
"W_tax:          D_tax:<BR><BR>"
" Supp_W  Item_Id  Item Name                  Qty  Stock  B/G  "
"Price    Amount<BR>"
" <INPUT NAME=A SIZE=5>  <INPUT NAME=B SIZE=6>"
"                            <INPUT NAME=C SIZE=1><BR>"
" <INPUT NAME=D SIZE=5>  <INPUT NAME=E SIZE=6>"
"                            <INPUT NAME=F SIZE=1><BR>"
" <INPUT NAME=G SIZE=5>  <INPUT NAME=H SIZE=6>"
"                            <INPUT NAME=I SIZE=1><BR>"
" <INPUT NAME=J SIZE=5>  <INPUT NAME=K SIZE=6>"
"                            <INPUT NAME=L SIZE=1><BR>"
" <INPUT NAME=M SIZE=5>  <INPUT NAME=N SIZE=6>"
"                            <INPUT NAME=O SIZE=1><BR>"
" <INPUT NAME=P SIZE=5>  <INPUT NAME=Q SIZE=6>"
"                            <INPUT NAME=R SIZE=1><BR>"
" <INPUT NAME=S SIZE=5>  <INPUT NAME=T SIZE=6>"
"                            <INPUT NAME=U SIZE=1><BR>"
```

```
" <INPUT NAME=V SIZE=5>  <INPUT NAME=W SIZE=6>"
"                            <INPUT NAME=X SIZE=1><BR>"
" <INPUT NAME=a SIZE=5>  <INPUT NAME=b SIZE=6>"
"                            <INPUT NAME=c SIZE=1><BR>"
" <INPUT NAME=d SIZE=5>  <INPUT NAME=e SIZE=6>"
"                            <INPUT NAME=f SIZE=1><BR>"
" <INPUT NAME=g SIZE=5>  <INPUT NAME=h SIZE=6>"
"                            <INPUT NAME=i SIZE=1><BR>"
" <INPUT NAME=j SIZE=5>  <INPUT NAME=k SIZE=6>"
"                            <INPUT NAME=l SIZE=1><BR>"
" <INPUT NAME=m SIZE=5>  <INPUT NAME=n SIZE=6>"
"                            <INPUT NAME=o SIZE=1><BR>"
" <INPUT NAME=p SIZE=5>  <INPUT NAME=q SIZE=6>"
"                            <INPUT NAME=r SIZE=1><BR>"
" <INPUT NAME=s SIZE=5>  <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></BODY>";

static char szNewOrderFormTemp2p[] =
"<INPUT TYPE=hidden NAME=3 VALUE=n######>"
"<PRE>                                 New Order<BR>"
"Warehouse: #####"
"  District: ##                         Date: ###################
<BR>"
"Customer:  ####   Name: ################   Credit: ##   "
"%Disc: #####            <BR>"
"Order Number: ########  Number of Lines: ##        "
"W_tax: #####   D_tax: #####  <BR><BR>"
" Supp_W  Item_Id  Item Name                  Qty  Stock  B/G  "
"Price    Amount<BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"  #####   ######   ########################  ##    ###    #   "
"$######  $#######  <BR>"
"Execution Status: ########################                    "
"Total:  $########  <BR>"
"</PRE>"
MENU_BAR
"</FORM></BODY>";

static char szOrderStatusFormTemp2i[] =
"<INPUT TYPE=hidden NAME=3 VALUE=O######>"
"<PRE>                               Order-Status<BR>"
"Warehouse: #####  District: <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></BODY>";

static char szOrderStatusFormTemp2p[] =
"<INPUT TYPE=hidden NAME=3 VALUE=o######>"
"<PRE>                               Order-Status<BR>"
"Warehouse: #####  District: ##<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></BODY>";

static char szPaymentFormTemp2i[] =
"<INPUT TYPE=hidden NAME=3 VALUE=P######>"
"<PRE>                                      Payment<BR>"
"Date:  <BR><BR>"
"Warehouse: #####                           District: <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></BODY>";

static char szPaymentFormTemp2p[] =
"<INPUT TYPE=hidden NAME=3 VALUE=p######>"
"<PRE>                                      Payment<BR>"
"Date: ################### <BR><BR>"
"Warehouse: #####                           District: ##<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></BODY>";

static char szStockLevelFormTemp2i[] =
"<INPUT TYPE=hidden NAME=3 VALUE=S######>"
"<PRE>                                    Stock-Level<BR>"
"Warehouse: #####  District: ##<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></BODY>";

static char szStockLevelFormTemp2p[] =
"<INPUT TYPE=hidden NAME=3 VALUE=s######>"
"<PRE>                                    Stock-Level<BR>"
"Warehouse: #####  District: ##<BR><BR>"
"Stock Level Threshold: ##<BR><BR>"
"low stock: ###"
"</PRE>"
MENU_BAR
"</FORM></BODY>";

static char szErrorFormTemplate[] =
"<BODY><FORM ACTION=%s METHOD=GET>"
"<INPUT TYPE=hidden NAME=3 VALUE=e%06d>"
"Error: %d (%s): %s"
MENU_BAR
"</FORM></BODY>";

static char szResponseHeaderTemplate[] =
"####\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_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( request_rec *req );
void CheckpointCmd( request_rec *req, int w_id, int ld_id );
void CheckpointStartupCmd( request_rec *req, int w_id, int ld_id );
void ClearCmd( request_rec *req );
void ExitCmd( request_rec *req );
void MenuCmd( request_rec *req, int w_id, int ld_id );
void SubmitCmd( request_rec *req, int *w_id, int *ld_id );
void MemoryCheckCmd( request_rec *req, 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);

void Log( char *szType, char *szStr );
void MakePanicPool( int dwResponseSize, apr_pool_t *p );
void MakeTemplatePool( int dwFormSize, int dwResponseSize,
apr_pool_t *p);
void MakeTransactionPool( int dwTransactionPoolSize, apr_pool_t
*p);
void DeletePanicPool( void );
void DeleteTemplatePool( void );
void DeleteTransactionPool( void );

int ProcessDeliveryQuery( request_rec *req,
        char *the_request,
        int w_id, int ld_id );
int ProcessNewOrderQuery( request_rec *req,
        char *the_request,
        int w_id, int ld_id );
int ProcessOrderStatusQuery( request_rec *req,
           char *the_request,
           int w_id, int ld_id );
int ProcessPaymentQuery( request_rec *req,
       char *the_request,
       int w_id, int ld_id );
int ProcessStockLevelQuery( request_rec *req,
          char *the_request,
          int w_id, int ld_id );

int ProcessQueryString(request_rec *req);

void PutNumeric( int iInt, int iFieldSize, char *pChar );
void SendErrorResponse( request_rec *req, int iError,
      int iErrorType, char *szMsg, int w_id, int ld_id,
      pConnData pConn );
void SendMainMenuForm( request_rec *req,
           int w_id, int ld_id, char *szStatus );
void SendResponse(request_rec *req, char *szStr, int iStrLen);
void SendWelcomeForm(request_rec *req);
#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
{
  apr_thread_mutex_t * critSec;
#ifdef FFE_DEBUG
  int iMaxIndex;
#endif
  int iNextFree;
  char  *index[1];
  char  forms[PANIC_FORM_SIZE];
} PanicStruct, *pPanicStruct;

typedef struct
{
  apr_thread_mutex_t * 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
{
  apr_thread_mutex_t * 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 = FALSE;

static apr_thread_mutex_t * startupspinlock;
static BOOL             startupFlag     = FALSE;

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, int
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
 *    int 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
 *
 */

static int tpcc_post_config(apr_pool_t *p, apr_pool_t *plog,
apr_pool_t *ptemp, server_rec *s)
{
  if (iInitStatus == FALSE) {
    apr_thread_mutex_create( &startupspinlock, 0, p);

    LogFD=open(MYFILE, O_CREAT|O_RDWR);
    myerrno=errno;
    MyLogFile=fdopen(LogFD, "a+");
    if (LogFD == -1)
    {
          printf("Bad file open, errno=%d\n", myerrno);
    }

    iInitStatus=TRUE;

    TPCCOpenLog(s->process->pool);

    ap_mpm_query(AP_MPMQ_MAX_THREADS, &max_threads);

#if (DEBUG == 1)
        fprintf(MyLogFile, "tpcc_post_config, pid=%d\n", getpid());
        fprintf(MyLogFile, "s->path: %s\n", s->path);
        fprintf(MyLogFile, "s->port: %d\n", s->port);
        fprintf(MyLogFile, "s->server_hostname: %s\n", s-
>server_hostname);
        fprintf(MyLogFile, "s->error_fname: %s\n", s->error_fname);
        fprintf(MyLogFile, "Max threads = %d\n", max_threads);
        fflush(MyLogFile);
#endif

  }

  return OK;
}

static void tpcc_child_init(apr_pool_t *p, server_rec *s)
{

#if (DEBUG == 1)
        fprintf(MyLogFile, "In tpcc_child_init\n");
        fflush(MyLogFile);
#endif

}

static apr_status_t tpcc_child_exit(void *data)
{
#if (DEBUG == 1)
        fprintf(MyLogFile, "In tpcc_child_exit\n");
        fflush(MyLogFile);
#endif

    TPCCShutdown( );

    DeleteTransactionPool( );
    DeleteTemplatePool( );
    DeletePanicPool( );

    TPCCCloseLog( );
}


/* FUNCTION: int tpcc_handler(request_rec *req)
 *
 * PURPOSE: This function is the main entry point for the TPCC DLL.
 *    The internet service calls this function passing in the
 *    http string.
 *
 * ARGUMENTS: request_rec *req  structure ptr containing the
 *            internet service information.
 *
 * RETURNS: int HSE_STATUS_SUCCESS  connection can be dropped if
 *            error
 *      HSE_STATUS_SUCCESS_AND_KEEP_CONN  keep connect valid
 *            comment sent
 *
 * COMMENTS:  None
 *
 */

int tpcc_handler(request_rec *req)
{
  int     status;
  int     dbstatus;

  /*  TPCCLog("now in handler"); */

  if ( ! startupFlag ) {
    apr_thread_mutex_lock( startupspinlock );
    if ( ! startupFlag ) {

#if (DEBUG == 1)
        fprintf(MyLogFile, "tpcc_handler: Startup Section\n");
#endif

  if ( ERR_SUCCESS != ( iInitStatus = ReadRegistrySettings( )))
        MakePanicPool( 50, req->pool );  /* make room for error
messages */
  else {
        dbstatus = TPCCStartup( );
    if( ERR_DB_SUCCESS != dbstatus ) {
      iInitStatus = dbstatus;
    }
  }

  {
    apr_pool_t *ppool = req->server->process->pool;

    strcpy(szModName, req->uri);

    MakeTemplatePool(max_threads, max_threads, ppool);
    MakePanicPool(max_threads, ppool);
    MakeTransactionPool(max_threads, ppool);
  }

      startupFlag = TRUE;
    }
    apr_thread_mutex_unlock( startupspinlock );
  }

#if (DEBUG == 1)
        fprintf(MyLogFile, "tpcc_handler: iInitStatus=%d\n",
iInitStatus);
#endif
  if( ERR_SUCCESS != iInitStatus )
  {
    SendErrorResponse(req, iInitStatus, ERR_TYPE_WEBDLL, NULL, -1,
-1, NULL);
    return TRUE;
  }


#if (DEBUG == 1)
        fprintf(MyLogFile, "req->the_request: %s\n", req-
>the_request);
        fprintf(MyLogFile, "req->unparsed_uri: %s\n", req-
>unparsed_uri);
        fprintf(MyLogFile, "req->uri: %s\n", req->uri);
        fprintf(MyLogFile, "req->filename: %s\n", req->filename);
        fprintf(MyLogFile, "req->args: %s\n", req->args);
        fflush(MyLogFile);
#endif
  /* process http query */
  status = ProcessQueryString(req);

  /* finish up with status returned by Processing functions */
  return OK;
}
```

```
/* FUNCTION: void SendErrorResponse( request_rec *req, int iError,
 *              int iErrorType, char *szMsg,
 *              int w_id, int ld_id )
 *
 * PURPOSE: This function displays an error form in the client
browser.
 *
 * ARGUMENTS: request_rec *req  IIS context structure pointer
 *            unique to this connection.
 *    int iError      id of error message
 *    int iErrorType    error type, ERR_TYPE_SQL,
 *            ERR_TYPE_DBLIB, ERR_TYPE_WEBDLL
 *    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
 *
 * COMMENTS:  If the error type is ERR_TYPE_WEBDLL the szMsg
parameter
 *    may be NULL because it is ignored. If the error type is
 *    ERR_TYPE_SQL or ERR_TYPE_DBLIB then the szMsg parameter
 *    contains the text of the error message, so the szMsg
 *    parameter cannot be NULL.
 *
 */

void
SendErrorResponse( request_rec *req, int iError, int iErrorType,
      char *szMsg, int w_id, int ld_id, pConnData pConn )
{
  int ii;

  static char szNoMsg[] = "";
  char    *szErrorTypeMsg;
  char    *szErrorMsg;
  char    *szForm;
  int   iStrLen;

  if ( !szMsg  )
    szMsg = szNoMsg;

#if (DEBUG == 1)
        fprintf(MyLogFile, "Entering SendErrorResponse\n");
        fflush(MyLogFile);
#endif

  RESERVE_PANIC_FORM( szForm );

#if (DEBUG == 1)
        fprintf(MyLogFile, "After Reserve Form\n");
        fflush(MyLogFile);
#endif

  if( ERR_TYPE_WEBDLL == iErrorType )
  {
    ii = 0;
    while( '\0' != errorMsgs[ii].szMsg[0] && iError !=
errorMsgs[ii].iError )
      ii++;
#if (DEBUG == 1)
        fprintf(MyLogFile, "After while\n");
        fflush(MyLogFile);
#endif
    if ( '\0' == errorMsgs[ii].szMsg[0] )
      ii = 1;  /* ERR_NO_MESSAGE */
    szErrorTypeMsg = "TPCCWEB";
    szErrorMsg = errorMsgs[ii].szMsg;
  }
  else if( ERR_TYPE_DBLIB == iErrorType )
  {
    szErrorTypeMsg = "DBLIB";
    szErrorMsg = szMsg;
  }
#if (DEBUG == 1)
        fprintf(MyLogFile, "After Reserve Form\n");
        fflush(MyLogFile);
#endif

/*
  if( NULL != pConn )
    TPCCTransactionErr( pConn, "%s(%d): %s\r\n",
      szErrorTypeMsg, iError, szErrorMsg );
  else
*/
    TPCCErr( "%s(%d): %s\r\n", szErrorTypeMsg, iError, szErrorMsg
);
#if (DEBUG == 1)
        fprintf(MyLogFile, "szErrorMsg=%s\n", szErrorMsg);
        fflush(MyLogFile);
#endif

  iStrLen = sprintf( szForm, szErrorFormTemplate, req->uri,
         WDID(w_id,ld_id), iError, szErrorTypeMsg, szErrorMsg );

#if (DEBUG == 1)
        fprintf(MyLogFile, "szForm=%s\n", szForm);
        fflush(MyLogFile);
#endif

#if (DEBUG == 1)
        fprintf(MyLogFile, "SendErrorResponse: Before
SendResponse\n");
        fflush(MyLogFile);
#endif

  SendResponse(req, szForm, iStrLen);

#if (DEBUG == 1)
        fprintf(MyLogFile, "SendErrorResponse: After
SendResponse\n");
        fflush(MyLogFile);
#endif
  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(request_rec *req, 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:
 *    request_rec *req    Server context structure.
 *    char      *szForm   form to pass to browser.
 *    int     iStrLen   length of form excluding null.
 *
 * RETURNS:
 *    None
 *
 * COMMENTS:
 */

void
SendResponse(request_rec *req, char *szForm, int iStrLen)
{
  int   lpbSize, numpad;
  char    szHeader1[10];
  char    headerpad[5];

  lpbSize = iStrLen;

#if (DEBUG == 1)
        fprintf(MyLogFile, "Entering SendResponse\n");
        fflush(MyLogFile);
#endif

  sprintf(szHeader1, "%d\0", lpbSize);
  apr_table_setn(req->headers_out, "Keep-Alive", "1");
/*
  apr_table_setn(req->headers_out, "Content-Length", szHeader1);
*/


  numpad=MAXPAD-(strlen(szHeader1));

#if (DEBUG == 1)
        fprintf(MyLogFile, "Header Pad = %s\n", szHeader1);
        fprintf(MyLogFile, "numpad = %d\n", numpad);
        fflush(MyLogFile);
#endif

  if (numpad > 0)
  {
    sprintf(headerpad, "%s\0", "P");
    while (--numpad > 0)
      strcat(headerpad, (char *)"P");
  }

  apr_table_set(req->headers_out, "PRTE PAD", headerpad);
#if (DEBUG == 1)
        fprintf(MyLogFile, "Header Pad = %s\n", headerpad);
        fflush(MyLogFile);
#endif

  req->content_type = "text/html";
/*
  apr_send_http_header(req);
*/

  ap_rputs(szForm, req);
}

/* 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( request_rec *req,
 *            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: request_rec *req  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( request_rec *req, 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(req, deliveryForm, giFormLen[DELIVERY_FORM]);

  UNRESERVE_FORM( DELIVERY_FORM, deliveryForm );
}

/* FUNCTION: void SendNewOrderForm( request_rec *req,
 *            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: request_rec *req  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( request_rec *req, 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(req, newOrderForm, giFormLen[NEW_ORDER_FORM]);

  UNRESERVE_FORM( NEW_ORDER_FORM, newOrderForm );
}

/* FUNCTION: void SendPaymentForm(request_rec *req,
 *          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:
 *    request_rec *req  pointer to structure passed in
 *            the internet
 *    int     w_id  warehouse id
 *    int     ld_id login district id
 *
 * RETURNS: None
 *
 * COMMENTS:  None
 *
 */

void
SendPaymentForm( request_rec *req, 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(req, paymentForm, giFormLen[PAYMENT_FORM]);

  UNRESERVE_FORM( PAYMENT_FORM, paymentForm );
}

/* FUNCTION: void SendOrderStatusForm(request_rec *req,
 *          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: request_rec *req  ptr to structure passed in the
 *             internet.
 *    int   w_id  warehouse id
 *    int   ld_id login district id
 *
 * RETURNS: None
 *
 * COMMENTS:  None
 *
 */

void
SendOrderStatusForm( request_rec *req, 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(req, orderStatusForm, giFormLen[ORDER_STATUS_FORM]);

  UNRESERVE_FORM( ORDER_STATUS_FORM, orderStatusForm );
}

/* FUNCTION: void SendStockLevelForm(request_rec *req,
 *          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: request_rec *req  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( request_rec *req, 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(req, stockLevelForm, giFormLen[STOCK_LEVEL_FORM]);

  UNRESERVE_FORM( STOCK_LEVEL_FORM, stockLevelForm );
}

/* FUNCTION: void SendMainMenuForm(request_rec *req,
 *          int w_id, int ld_id, char *szStatus)
 *
 * PURPOSE: This function sends the main menu form to the browser.
 *
 * ARGUMENTS: request_rec *req  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( request_rec *req,
      int w_id, int ld_id, char *szStatus )
{
  char  *szForm;
  int iStrLen;
  static char *szNoStatus = "";
  char  *pszStatus;

  pszStatus = ( NULL == szStatus ) ? szNoStatus : szStatus;

#if (DEBUG == 1)
        fprintf(MyLogFile, "Before RESERVE_PANIC_FORM\n");
        fflush(MyLogFile);
#endif

  RESERVE_PANIC_FORM( szForm );

#if (DEBUG == 1)
        fprintf(MyLogFile, "Before SendMainMenuForm\n");
        fflush(MyLogFile);
#endif
  iStrLen = sprintf( szForm, szMainMenuFormTemplate,
         req->uri, WDID(w_id,ld_id), pszStatus );

  SendResponse(req, szForm, iStrLen);

  UNRESERVE_PANIC_FORM( szForm );
}


/* FUNCTION: void SendWelcomeForm(request_rec *req)
 *
 * PURPOSE: This function sends the welcome form to the browser.
 *
 * ARGUMENTS:   None
 *
 * RETURNS: None
 *
 * COMMENTS:  The welcome form is generated on initialization.
 */

void
SendWelcomeForm(request_rec *req)
{
  char    *mod_name;

#if (DEBUG == 1)
        fprintf(MyLogFile, "SendWelcomeForm 1\n");
        fflush(MyLogFile);
#endif
    mod_name = strrchr( req->uri, '/' );
    if( NULL != mod_name )
      mod_name++;
    else
      {
        fprintf(MyLogFile, "SendWelcomeForm: Null mod_name\n");
        return;
      }

  iWelcomeFormLen = sprintf(szWelcomeForm, szWelcomeFormTemplate,
mod_name);

#if (DEBUG == 1)
        fprintf(MyLogFile, "SendWelcomeForm 2\n");
        fflush(MyLogFile);
#endif

  SendResponse( req, szWelcomeForm, iWelcomeFormLen );
}

/* FUNCTION: int ProcessQueryString(request_rec *req)
 *
 * PURPOSE: This function extracts the relevent information out
 *    of the http command passed in from the browser.
 *
 * ARGUMENTS: request_rec *req  IIS context structure pointer
 *           unique to this connection.
 *
 * RETURNS: int       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.
 */

int
ProcessQueryString(request_rec *req)
{
  static char *beginptr = "Begin";
  char *ptr;
  char *cmdptr;
  int cFormID;
  int w_id;
  int ld_id;
  int status;
  int retcode;

  w_id = 0;
  ld_id = 0;

#if (DEBUG == 1)
        fprintf(MyLogFile, "Starting QueryString 1\n");
        fprintf(MyLogFile, "&ptr=%x\n", &ptr);
        fflush(MyLogFile);
#endif
  if ( GetCharKeyValuePtr( req->args, '3', &ptr ))
  {
    cFormID = *ptr++;
    if( !GetWDID( ptr, &w_id, &ld_id, &ptr )) {
#if (DEBUG == 1)
        fprintf(MyLogFile, "Calling SendErrorResponse\n");
        fflush(MyLogFile);
#endif
      SendErrorResponse( req, ERR_W_ID_INVALID, ERR_TYPE_WEBDLL,
NULL,
       w_id, ld_id, NULL );
      return TRUE;
    }
  }
  else
    cFormID = '\0';

  /* now figure out what command we have and execute it */
  if ( !GetCharKeyValuePtr( ptr, '0', &cmdptr ))
  {
    if(  req->args == NULL ) {
      cmdptr = beginptr;
    }
    else {
      SendErrorResponse( req, ERR_COMMAND_UNDEFINED,
ERR_TYPE_WEBDLL,
       NULL, w_id, ld_id, NULL );
      return TRUE;
    }
  }

  if( '\0' == cFormID && !MATCHES_BEGIN( cmdptr )) {
    SendErrorResponse( req, ERR_INVALID_FORM_AND_CMD_NOT_BEGIN,
           ERR_TYPE_WEBDLL, NULL, w_id, ld_id, NULL );
    return TRUE;
  }

  status = TRUE;
  if( MATCHES_PROCESS( cmdptr ))
  {
#if (DEBUG == 1)
        fprintf(MyLogFile, "Matches Process\n");
        fflush(MyLogFile);
#endif

    if( 'N' == cFormID )
      retcode = ProcessNewOrderQuery( req, ptr, w_id, ld_id );
    else if( 'P' == cFormID )
      retcode = ProcessPaymentQuery( req, ptr, w_id, ld_id );
    else if( 'D' == cFormID )
      retcode = ProcessDeliveryQuery( req, ptr, w_id, ld_id );
    else if( 'O' == cFormID )
      retcode = ProcessOrderStatusQuery( req, ptr, w_id, ld_id );
    else if( 'S' == cFormID )
      retcode = ProcessStockLevelQuery( req, ptr, w_id, ld_id );
    else {
      SendErrorResponse( req, ERR_INVALID_FORM, ERR_TYPE_WEBDLL,
NULL,
       w_id, ld_id, NULL );
      return TRUE;
    }

    if( ERR_DB_PENDING == retcode )
      status = TRUE;
    else if( ERR_DB_SUCCESS != retcode ) {
#if (DEBUG == 1)
```

```
        fprintf(MyLogFile, "Here We Are Again!!!\n");
        fflush(MyLogFile);
#endif
      if (!apr_table_get(req->headers_out, "PRTE PAD"))
      {
        SendErrorResponse( req, retcode, ERR_TYPE_WEBDLL, NULL,
w_id, ld_id, NULL );
      }
      return TRUE;
    }
  }
  else if( MATCHES_BEGIN( cmdptr ))
    BeginCmd( req );
  else if( MATCHES_NEWORDER( cmdptr ))
    SendNewOrderForm( req, w_id, ld_id );
  else if( MATCHES_PAYMENT( cmdptr ))
    SendPaymentForm( req, w_id, ld_id );
  else if( MATCHES_ORDERSTATUS( cmdptr ))

    SendOrderStatusForm( req, w_id, ld_id );
  else if( MATCHES_STOCKLEVEL( cmdptr ))
    SendStockLevelForm( req, w_id, ld_id );
  else if( MATCHES_DELIVERY( cmdptr ))
    SendDeliveryForm( req, w_id, ld_id );
  else if( MATCHES_SUBMIT( cmdptr ))
    SubmitCmd( req, &w_id, &ld_id );
  else if( MATCHES_MENU( cmdptr ))
    MenuCmd( req, w_id, ld_id );
  else if( MATCHES_EXIT( cmdptr ))
    ExitCmd( req );
  else if( MATCHES_CLEAR( cmdptr ))
    ClearCmd( req );
  else
    SendErrorResponse( req, ERR_COMMAND_UNDEFINED, ERR_TYPE_WEBDLL,
           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[] = "********************";
  char tmpbuff[10];
  double dtmp;


  pChar += (iFieldSize - 1);

  dtmp=dVal*100.0;
  if(0 > dVal)
  {
    bSignFlag = FALSE;
    iInt = abs((int)( dtmp ));
  }
  else
  {
    /* iInt = (int)( dtmp ); */
    sprintf(tmpbuff,"%.0f",dtmp);
    iInt = (int)(atoi(tmpbuff));
  }
  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. */
    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: void TPCCDeliveryResponse( request_rec *req,
 *          int retcode,
 *          DeliveryData *deliveryData )
 *
 * PURPOSE: This function fills in the values and returns the
 *    response form to the browser.
 *
 * ARGUMENTS: request_rec *req
 *    int      retcode  return code from db
 *    DeliveryData    *deliveryData pointer to the delivery
 *            data structure.
 *
 * RETURNS: none
 *
 * COMMENTS:  none
 */

void
TPCCDeliveryResponse( int retcode, pDeliveryData pDelivery,
        pDeliveryData CompletedDeliveries[DELIVERY_RESPONSE_COUNT]
)
{
  int ssCnt = 0;
  char *szOutput;
  int iOutputLen;
  PutStrStruct StrStruct[2];
  char *deliveryForm;
  request_rec *req;

  req = pDelivery->pCC;

  if ( ERR_DB_PENDING == retcode )
  {
    return;
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {

    SendErrorResponse( req, ERR_DELIVERY_NOT_PROCESSED,
           ERR_TYPE_WEBDLL, NULL,
           pDelivery->w_id, pDelivery->ld_id,
           (pConnData)pDelivery );

    return;
  }
  else  if ( ERR_DB_SUCCESS != retcode )
  {
    SendErrorResponse( req, ERR_DB_DELIVERY_NOT_QUEUED,
           ERR_TYPE_WEBDLL, NULL,
           pDelivery->w_id, pDelivery->ld_id,
           (pConnData)pDelivery );

    return;
  }

  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]);

  UNRESERVE_TRANSACTION_STRUCT( DELIVERY_TRANS, pDelivery );

  PUT_STRING(NULL, 0, 0, StrStruct[ssCnt]);
  PutHTMLStrings(StrStruct, deliveryForm,
giResponseLen[DELIVERY_RESPONSE],
     &szOutput, &iOutputLen);

  SendResponse(req, szOutput, iOutputLen);

  UNRESERVE_RESPONSE( DELIVERY_RESPONSE, deliveryForm );

  if( szOutput != deliveryForm )
    UNRESERVE_PANIC_FORM( szOutput );
}

/* FUNCTION: void TPCCNewOrderResponse(request_rec *req,
 *          int retcode,
 *          NewOrderData *newOrderData )
 *
 * PURPOSE: This function fills in the values and returns the
 *    response form to the browser.
 *
 * ARGUMENTS: request_rec *req  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
 */

void
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;
  char szStatus[80];
  request_rec *req;

  req = pNewOrder->pCC;

  if ( ERR_DB_PENDING == retcode )
  {
    return;
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {
    SendErrorResponse( req, ERR_NEW_ORDER_NOT_PROCESSED,
           ERR_TYPE_WEBDLL, NULL,
           pNewOrder->w_id, pNewOrder->ld_id,
           (pConnData)pNewOrder );
    return;
  }
  else if( ERR_DB_SUCCESS != retcode && ERR_DB_NOT_COMMITED !=
retcode )
  {
    sprintf( szStatus,
       "Item number is not valid, or DB error = %d",
       pNewOrder->dbstatus );
    SendErrorResponse( req, ERR_DB_ERROR,
           ERR_TYPE_WEBDLL, NULL,
           pNewOrder->w_id, pNewOrder->ld_id,
           (pConnData)pNewOrder );
    return;
  }
  else if ( ERR_DB_SUCCESS == retcode )
  {
    bValid = TRUE;
    execution_status = "Transaction commited.";
  }
  else if ( ERR_DB_NOT_COMMITED == retcode )
  {
    bValid = FALSE;
    execution_status = "Item number is not valid.";
  }

  RESERVE_RESPONSE( NEW_ORDER_RESPONSE, newOrderForm );

  if(bValid)
  {
    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]);

    /* 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]);
    PutNumeric(pNewOrder->o_id,
         newOrderResponseIndexes[NO_OID].iLen,

&newOrderForm[newOrderResponseIndexes[NO_OID].iStartIndex]);
    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)].iStart
Index-1],
       szDollar, 1);
      PutFloat2(pNewOrder->o_ol[i].i_price,
    newOrderResponseIndexes[NO_PRICE+(i*8)].iLen,

  &newOrderForm[newOrderResponseIndexes[NO_PRICE+(i*8)].iStartIndex
]);

memcpy(&newOrderForm[newOrderResponseIndexes[NO_AMT+(i*8)].iStartIn
dex-1],
       szDollar, 1);
      PutFloat2(pNewOrder->o_ol[i].ol_amount,
    newOrderResponseIndexes[NO_AMT+(i*8)].iLen,

  &newOrderForm[newOrderResponseIndexes[NO_AMT+(i*8)].iStartIndex])
;

    }
    /* need to blank out the rest of the unused item rows */
    jj = NO_AMT + ((i-1)*8) + 1;
    for(kk=i; kk<15; 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
  {
    /* will need to blank out any fields not entered when not valid
*/
    /* space for discount */

memcpy(&newOrderForm[newOrderResponseIndexes[NO_DISC].iStartIndex],
     szBlanks, newOrderResponseIndexes[NO_DISC].iLen);
    /*the actual order number */
    PutNumeric(pNewOrder->o_id,
         newOrderResponseIndexes[NO_OID].iLen,

&newOrderForm[newOrderResponseIndexes[NO_OID].iStartIndex]);
    /* space for number of lines, w_tax, and d_tax */
    for(kk=0; kk<3; kk++)
    {

memcpy(&newOrderForm[newOrderResponseIndexes[NO_LINES+kk].iStartInd
ex],
       szBlanks, newOrderResponseIndexes[NO_LINES+kk].iLen);
    }
    /* spaces for each of the fields in the row items */
    jj = NO_S_WID;
    for(kk=0; kk<15; 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);

#ifdef FFE_DEBUG
  pNewOrder->iStage |= UNRESERVING;
#endif

  UNRESERVE_TRANSACTION_STRUCT( NEW_ORDER_TRANS, pNewOrder );

  SendResponse(req, szOutput, iOutputLen);

  UNRESERVE_RESPONSE( NEW_ORDER_RESPONSE, newOrderForm );

  if( szOutput != newOrderForm )
    UNRESERVE_PANIC_FORM( szOutput );
```

```
}

/* FUNCTION: void TPCCPaymentResponse(request_rec *req,
 *          int retcode,
 *          PaymentData *paymentData)
 *
 * PURPOSE: This function fills in the values and returns the
 *    response form to the browser.
 *
 * ARGUMENTS: request_rec *req  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
 */

void
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;
  request_rec *req;

  req = pPayment->pCC;

  if ( ERR_DB_PENDING == retcode )
  {
    return;
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {
    SendErrorResponse( req, ERR_PAYMENT_NOT_PROCESSED,
           ERR_TYPE_WEBDLL, NULL,
           pPayment->w_id, pPayment->ld_id,
           (pConnData)pPayment );
    return;
  }
  else if ( ERR_DB_NOT_COMMITED == retcode )
  {
    SendErrorResponse( req, ERR_PAYMENT_INVALID_CUSTOMER,
           ERR_TYPE_WEBDLL, NULL,
           pPayment->w_id, pPayment->ld_id,
           (pConnData)pPayment );
    return;
  }
  else if ( ERR_DB_SUCCESS != retcode )
  {
    SendErrorResponse( req, ERR_DB_ERROR,
           ERR_TYPE_WEBDLL, NULL,
           pPayment->w_id, pPayment->ld_id,
           (pConnData)pPayment );
    return;
  }


  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].iStartInde
x],
   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->d_street_2,
       paymentResponseIndexes[PT_D_STR_2].iLen,
       paymentResponseIndexes[PT_D_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].iStartIn
dex],
       szBlanks, paymentResponseIndexes[PT_CUST_DATA+i].iLen);
    }
  }

  PUT_STRING(NULL, 0, 0, StrStruct[ssCnt]);

  PutHTMLStrings(StrStruct, paymentForm,
giResponseLen[PAYMENT_RESPONSE],
     &szOutput, &iOutputLen);

#ifdef FFE_DEBUG
  pPayment->iStage |= UNRESERVING;
#endif

  UNRESERVE_TRANSACTION_STRUCT( PAYMENT_TRANS, pPayment );

  SendResponse(req, szOutput, iOutputLen);

  UNRESERVE_RESPONSE( PAYMENT_RESPONSE, paymentForm );

  if( szOutput != paymentForm )
    UNRESERVE_PANIC_FORM( szOutput );
}

/* FUNCTION: void TPCCOrderStatusResponse( int retcode,
 *              OrderStatusData *orderStatusData)
 *
 * PURPOSE: This function fills in the values and returns the
 *    response form to the browser.
 *
 * ARGUMENTS: request_rec *req  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
 */

void
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;
  request_rec *req;

  req = pOrderStatus->pCC;

  if ( ERR_DB_PENDING == retcode )
  {
    return;
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {
    SendErrorResponse( req, ERR_ORDER_STATUS_NOT_PROCESSED,
           ERR_TYPE_WEBDLL, NULL,
           pOrderStatus->w_id, pOrderStatus->ld_id,
           (pConnData)pOrderStatus );
    return;
  }
  else if ( ERR_DB_NOT_COMMITED == retcode )
  {
    SendErrorResponse( req, ERR_NOSUCH_CUSTOMER,
           ERR_TYPE_WEBDLL, NULL,
           pOrderStatus->w_id, pOrderStatus->ld_id,
           (pConnData)pOrderStatus );
    return;
  }
  else  if ( ERR_DB_SUCCESS != retcode )
  {
    SendErrorResponse( req, ERR_DB_ERROR,
           ERR_TYPE_WEBDLL, NULL,
           pOrderStatus->w_id, pOrderStatus->ld_id,
           (pConnData)pOrderStatus );
  return;
  }

  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].iStartI
ndex],
   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)].iStartI
ndex]);
    PutNumeric(pOrderStatus->s_ol[i].ol_i_id,
         orderStatusResponseIndexes[OS_IID+(i*5)].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_IID+(i*5)].iStartInd
ex]);
    PutNumeric(pOrderStatus->s_ol[i].ol_quantity,
         orderStatusResponseIndexes[OS_QTY+(i*5)].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_QTY+(i*5)].iStartInd
ex]);

memcpy(&orderStatusForm[orderStatusResponseIndexes[OS_AMT+(i*5)].iS
tartIndex-1],
       szDollar, 1);
    PutFloat2(pOrderStatus->s_ol[i].ol_amount,
        orderStatusResponseIndexes[OS_AMT+(i*5)].iLen,

&orderStatusForm[orderStatusResponseIndexes[OS_AMT+(i*5)].iStartInd
ex]);
    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)
].iStartIndex],
     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<15; 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);

#ifdef FFE_DEBUG
  pOrderStatus->iStage |= UNRESERVING;
#endif

  UNRESERVE_TRANSACTION_STRUCT( ORDER_STATUS_TRANS, pOrderStatus );

  SendResponse(req, szOutput, iOutputLen);

  UNRESERVE_RESPONSE( ORDER_STATUS_RESPONSE, orderStatusForm );

  if( szOutput != orderStatusForm )
    UNRESERVE_PANIC_FORM( szOutput );
}
```

```
/* FUNCTION: void 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: request_rec *req  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
 */

void
TPCCStockLevelResponse( int retcode, StockLevelData *pStockLevel )
{
  char *stockLevelForm;
  request_rec *req;

  req = pStockLevel->pCC;

  if ( ERR_DB_PENDING == retcode )
  {
    return;
  }
  else if ( ERR_DB_DEADLOCK_LIMIT == retcode )
  {
    SendErrorResponse( req, ERR_STOCKLEVEL_NOT_PROCESSED,
           ERR_TYPE_WEBDLL, NULL,
           pStockLevel->w_id, pStockLevel->ld_id,
           (pConnData)pStockLevel );
    return;
  }
  else if ( ERR_DB_SUCCESS != retcode )
  {
    SendErrorResponse( req, ERR_DB_ERROR,
           ERR_TYPE_WEBDLL, NULL,
           pStockLevel->w_id, pStockLevel->ld_id,
           (pConnData)pStockLevel );
    return;
  }

  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]);

#ifdef FFE_DEBUG
  pStockLevel->iStage |= UNRESERVING;
#endif

  UNRESERVE_TRANSACTION_STRUCT( STOCK_LEVEL_TRANS, pStockLevel );

  SendResponse(req, stockLevelForm,
         giResponseLen[STOCK_LEVEL_RESPONSE]);

  UNRESERVE_RESPONSE( STOCK_LEVEL_RESPONSE, stockLevelForm );
}


/* FUNCTION: int ProcessDeliveryQuery( request_rec *req,
 *
 * 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: request_rec *req  ptr to the structure
 *            containing the internet server
 *            information.
 *
 * RETURNS: int   status
 *
 * COMMENTS:  None
 *
 */
```

```
int
ProcessDeliveryQuery( request_rec *req, char *the_request,
          int w_id, int ld_id )
{
  int     retcode;
  char      *ptr;
  char      *deliveryVals[MAXDELIVERYVALS];
  pDeliveryData   pDelivery;
  pDeliveryData   CompletedDeliveries[DELIVERY_RESPONSE_COUNT];

  RESERVE_TRANSACTION_STRUCT( DELIVERY_TRANS, pDelivery );

  pDelivery->w_id = w_id;
  pDelivery->ld_id = ld_id;
  pDelivery->pCC = req;

  PARSE_QUERY_STRING(the_request, 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 > 10 || pDelivery->o_carrier_id < 1
)
    return ERR_DELIVERY_CARRIER_ID_RANGE;

#ifdef FFE_DEBUG
  pDelivery->iStage |= CALLING_LH;
#endif
  retcode = TPCCDelivery( pDelivery );

#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));
  pDelivery->iStage |= CALLING_RESP;
#endif
  TPCCDeliveryResponse( retcode, pDelivery, CompletedDeliveries );

  return retcode;
}

/* FUNCTION: int ProcessNewOrderQuery( request_rec *req,
 *
 * 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: request_rec *req  ptr to structure containing
 *            internet server info
 *
 * RETURNS: int   status
 *
 * COMMENTS:  None
 *
 */

int
ProcessNewOrderQuery( request_rec *req, char *the_request,
          int w_id, int ld_id )
{
  int     retcode;
  NewOrderData    *pNewOrder;

  RESERVE_TRANSACTION_STRUCT( NEW_ORDER_TRANS, pNewOrder );

  pNewOrder->w_id = w_id;
  pNewOrder->ld_id = ld_id;
  pNewOrder->pCC = req;

  if ( ERR_SUCCESS != ( retcode = ParseNewOrderQuery( the_request,
                pNewOrder )))
    return retcode;

#ifdef FFE_DEBUG
  pNewOrder->iStage |= CALLING_LH;
#endif
  retcode = TPCCNewOrder( pNewOrder );

  if (pNewOrder->status > 0)
  {
        retcode=pNewOrder->status;
  }

#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));

  pNewOrder->iStage |= CALLING_RESP;
#endif
  TPCCNewOrderResponse( retcode, pNewOrder );

  return retcode;
}
```

```
/* FUNCTION: int ProcessOrderStatusQuery( request_rec *req,
 *
 * 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: request_rec *req  ptr to structure that contains
 *            the internet server info.
 *
 * RETURNS: int   status
 *
 * COMMENTS:  None
 *
 */

int
ProcessOrderStatusQuery( request_rec *req, char *the_request,
       int w_id, int ld_id )
{
  int     retcode;
  OrderStatusData *pOrderStatus;

  RESERVE_TRANSACTION_STRUCT( ORDER_STATUS_TRANS, pOrderStatus );

  pOrderStatus->w_id = w_id;
  pOrderStatus->ld_id = ld_id;
  pOrderStatus->pCC = req;

  if( ERR_SUCCESS != ( retcode = ParseOrderStatusQuery(
the_request,
            pOrderStatus )))
    return retcode;

#ifdef FFE_DEBUG
  pOrderStatus->iStage |= CALLING_LH;
#endif
  retcode = TPCCOrderStatus( pOrderStatus );

  if (pOrderStatus->status > 0)
        retcode=ERR_DB_ERROR;

#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));
  pOrderStatus->iStage |= CALLING_RESP;
#endif
  TPCCOrderStatusResponse( retcode, pOrderStatus );


  return retcode;
}

/* FUNCTION: int ProcessPaymentQuery( request_rec *req,
 *
 * 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: request_rec *req  ptr to structure that contains
 *            the internet server info.
 *
 * RETURNS: int   status
 *
 * COMMENTS:  None
 *
 */

int
ProcessPaymentQuery( request_rec *req, char *the_request,
         int w_id, int ld_id )
{
  int     retcode;
  PaymentData   *pPayment;

  RESERVE_TRANSACTION_STRUCT( PAYMENT_TRANS, pPayment );

  pPayment->w_id = w_id;
  pPayment->ld_id = ld_id;
  pPayment->pCC = req;

  if( ERR_SUCCESS != ( retcode = ParsePaymentQuery( the_request,
              pPayment )))
    return retcode;

#ifdef FFE_DEBUG
  pPayment->iStage |= CALLING_LH;
#endif
  retcode = TPCCPayment( pPayment );

  if (pPayment->status > 0)
  retcode=ERR_DB_ERROR;

#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));
  pPayment->iStage |= CALLING_RESP;
#endif
  TPCCPaymentResponse( retcode, pPayment );
  return retcode;
}

/* FUNCTION: int ProcessStockLevelQuery( request_rec *req,
 *
 * 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: request_rec *req  ptr to structure that contains
 *            the internet server info.
 *    int   iSyncId   client browser sync id
 *
 * RETURNS: int   status
 *
 * COMMENTS:  None
 *
 */

int
ProcessStockLevelQuery( request_rec *req, char *the_request,
      int w_id, int ld_id )
{
  char          *ptr;
  int   retcode;
  char    *stockLevelVals[MAXSTOCKLEVELVALS];
  StockLevelData  *pStockLevel;

#if (DEBUG == 1)
        fprintf(MyLogFile, "Entering ProcessStockLevelQuery\n");
  fflush(MyLogFile);
#endif

  RESERVE_TRANSACTION_STRUCT( STOCK_LEVEL_TRANS, pStockLevel );

  pStockLevel->w_id = w_id;
  pStockLevel->ld_id = ld_id;
  pStockLevel->pCC = req;

  PARSE_QUERY_STRING(the_request, 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;

#ifdef FFE_DEBUG
  pStockLevel->iStage |= CALLING_LH;
#endif

  retcode = TPCCStockLevel( pStockLevel );

  if (pStockLevel->status > 0)
        retcode=ERR_DB_ERROR;


#ifdef FFE_DEBUG
  _ASSERT(VALID_DB_ERR(retcode));
  pStockLevel->iStage |= CALLING_RESP;
#endif
  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( int 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( int dwResponseSize, apr_pool_t *p )
{
  int iMallocSize;
  char *pForm;
  int 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));

#if (DEBUG == 1)
        fprintf(MyLogFile, "gpPanicForms malloc=%d\n",
iMallocSize);
        fflush(MyLogFile);
#endif

  gpPanicForms = malloc( iMallocSize );
  apr_thread_mutex_create( &gpPanicForms->critSec, 0, p );
#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 )
{
  free( gpPanicForms );
}

/* FUNCTION: void MakeTemplatePool( int dwFormSize, int
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( int dwFormSize, int dwResponseSize, apr_pool_t
*p)
{
  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;
  int 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
);
  MakeResponseHeader( );

  /* 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])));
#if (DEBUG == 1)
        fprintf(MyLogFile, "gpForms malloc=%d\n", iMallocSize);
        fflush(MyLogFile);
#endif
  gpForms = malloc( iMallocSize );

  for( jj = 0; jj < NUMBER_POOL_FORM_TYPES; jj++ )
  {
    apr_thread_mutex_create( &gpForms->critSec[jj], 0, p );
    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])));
#if (DEBUG == 1)
        fprintf(MyLogFile, "gpResponses malloc=%d\n", iMallocSize);
        fflush(MyLogFile);
#endif
  gpResponses = malloc( iMallocSize );

  for( jj = 0; jj < NUMBER_POOL_RESPONSE_TYPES; jj++ )
  {
    apr_thread_mutex_create( &gpResponses->critSec[jj], 0, p );
#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 )
{
  free( gpResponses );

  free( gpForms );

  free( gpPanicForms );

}

/* FUNCTION: void MakeTransactionPool( int dwTransactionPoolSize )
 *
 * 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
MakeTransactionPool( int dwTransactionPoolSize , apr_pool_t *p)
{
  int iMaxSize;
  int iSize;
  char *data;
  int ii;
  /**** set up transaction data pool used during async operation
****/
  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));
  iMaxSize = MAX(iMaxSize,sizeof(LoginData));
#if 1
  iSize = (((char *)&gpTransactionPool->index - (char
*)gpTransactionPool) +
     (((char *)gpTransactionPool->data - (char *)gpTransactionPool-
>index)
      * dwTransactionPoolSize ) +
     (sizeof( char ) * iMaxSize * dwTransactionPoolSize ));
#else
  iSize = (((char *)&gpTransactionPool->index - (char
*)gpTransactionPool) +
     (((char *)gpTransactionPool->data - (char *)gpTransactionPool-
>index)
      * dwTransactionPoolSize ) +
     (sizeof( char ) * iMaxSize * dwTransactionPoolSize ));
#endif

#if (DEBUG == 1)
        fprintf(MyLogFile, "gpTransaction malloc=%d\n", iSize);
        fflush(MyLogFile);
#endif
  gpTransactionPool = malloc( iSize );

  apr_thread_mutex_create( &gpTransactionPool->critSec, 0, p );
#ifdef FFE_DEBUG
  gpTransactionPool->iMaxIndex = dwTransactionPoolSize - 1;
  gpTransactionPool->iTransactionSize = iMaxSize;
  gpTransactionPool->iHistoryId = 0;
#endif
  gpTransactionPool->iNextFree = 0;

  /* careful here, the data is not right after index[0] as the
structure */
  /* defines.  We have wedged 'NumUsers + total' indexes in
between. */
  data = ((char *)&gpTransactionPool->index[0] +
    (((char *)&gpTransactionPool->data[0] -
      (char *)&gpTransactionPool->index[0]) *
     dwTransactionPoolSize ));

  for( ii = 0; ii < dwTransactionPoolSize; ii++ ) {
    gpTransactionPool->index[ii] = data;
    data += iMaxSize;
  }
}

/* FUNCTION: void DeleteTransactionPool( void )
 *
 * PURPOSE: This function destroys the array of transaction data
 *    structures used by the threads as they process a transaction.
 *
 * ARGUMENTS: none
 *
 * RETURNS: none
 *
 * COMMENTS:  none
 */
void
DeleteTransactionPool( void )
{
  free( gpTransactionPool );
}

/* FUNCTION: void BeginCmd( request_rec *req )
 *
 * PURPOSE: This routine is executed in response to the browser
query
 *    'CMD=Begin&Server=??????'.
 *
 * ARGUMENTS: request_rec *req  IIS context structure pointer
 *            unique to this connection.
 *            at login.
 * RETURNS: None
 *
 * COMMENTS:  Specification of a server machine is required.
 */

void
BeginCmd( request_rec *req )
{
  SendWelcomeForm(req);
}


/* FUNCTION: void ClearCmd(request_rec *req)
 *
 * PURPOSE: This resets all terminals and resets the log file.
 *
 * ARGUMENTS: request_rec *req  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(request_rec *req)
{
  if ( bLog )
  {
    TPCCCloseLog( );
    TPCCOpenLog( req->server->process->pool);
  }

  SendWelcomeForm(req);
}

/* FUNCTION: void ExitCmd(request_rec *req,
 *
 * PURPOSE: This function deallocates the terminal associated with
 *    the browser and presents the login screen.
 *
 * ARGUMENTS: request_rec *req  IIS context structure pointer
 *            unique to this connection.
 * RETURNS: None
 *
 * COMMENTS:  None
 *
 */

void
ExitCmd( request_rec *req )
{
/*
  TPCCDisconnect( req );
*/

  SendWelcomeForm( req );
}

/* FUNCTION: void MenuCmd( request_rec *req,
 *
 * PURPOSE: This function displays the main menu.
 *
 * ARGUMENTS: request_rec *req  IIS context structure pointer
 *            unique to this connection.
 * RETURNS: None
 *
 * COMMENTS:  None
 *
 */

void
MenuCmd( request_rec *req, int w_id, int ld_id )
{
  SendMainMenuForm(req, w_id, ld_id, NULL);
}


/* FUNCTION: void SubmitCmd( request_rec *req )
 *
 * PURPOSE: This function assigns a unique terminal id to the
calling
 *    browser.
 *
 * ARGUMENTS: request_rec *req  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( request_rec *req, int *w_id, int *ld_id )
{
  int iStatus;
  LoginData login;
  char  *ptr;

  if ( !GetCharKeyValuePtr( req->args, '4', &ptr ) ||
       ( 0 == ( *w_id = atoi( ptr ))) ||
       ( *w_id < 0 ))
  {
    SendErrorResponse( req, ERR_W_ID_INVALID, ERR_TYPE_WEBDLL,
           NULL, *w_id, -1, NULL );
    goto SubmitError;
  }

  if ( !GetCharKeyValuePtr( req->args, '5', &ptr ) ||
       ( 0 == ( *ld_id = atoi( ptr ))) ||
       ( *ld_id > 10 ) ||
       ( *ld_id < 0 ))
  {
    SendErrorResponse( req, ERR_D_ID_INVALID, ERR_TYPE_WEBDLL,
           NULL, *w_id, *ld_id, NULL );
    goto SubmitError;
  }

  login.w_id = *w_id;
  login.ld_id = *ld_id;
  login.pCC = req;
  strcpy( login.szServer, gszServer );
  strcpy( login.szDatabase, gszDatabase );
  strcpy( login.szUser, gszUser );
  strcpy( login.szPassword, gszPassword );
  sprintf( login.szApplication, "TPCC" );
  iStatus = TPCCConnect( &login );
  if( ERR_DB_SUCCESS != iStatus )
  {
    SendErrorResponse( req, iStatus, ERR_TYPE_WEBDLL,
           NULL, *w_id, *ld_id, NULL );
    goto SubmitError;
  }

  SendMainMenuForm(req, *w_id, *ld_id, NULL);
  return;

SubmitError:
  return;

}


/* 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;

  if (szIPtr == NULL)
    return 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 = (int) (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;
}

/* FUNCTION: void CheckMemory(void *param)
 *
 * PURPOSE: This function loops calling _CrtCheckMemory()
 *
 * ARGUMENTS:
 *    void  *param      not used
 *
 * RETURNS: nothing
 *
 * COMMENTS:
 */

#ifdef FFE_DEBUG

unsigned __stdcall
CheckMemory(void *param)
{
  while (TRUE)
  {
    _ASSERTE(_CrtCheckMemory());
    Sleep(1000);
  }

  return 0;
}

#endif
```

```
------------------------------------------------
      mod_tpcc.h
------------------------------------------------

#ifndef MOD_TPCC_H
#define MOD_TPCC_H
/*+*****************************************************************
*********
 *
*
 *  COPYRIGHT (c) 1997 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
 *
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                      - Conversion to run under Linux and Apache
 *
 */

/* function prototypes */
BOOL GetNumeric(char *ptr, int *iValue);
BOOL GetValuePtr(char *pProcessedQuery[], int iIndex, char
**pValue);

/* define indexes for parsing the query string */
/* for the payment, orderstatus and new order txns */
#define DID 0
#define CID DID+1
/* more for the order status txn */
#define CLT_O CID+1
#define MAXORDERSTATUSVALS CLT_O + 1
/* for the stocklevel txn */
#define TT  0
#define MAXSTOCKLEVELVALS TT + 1
/* for the delivery txn */
#define QUEUETIME 0
#define OCD 1
#define MAXDELIVERYVALS OCD + 1
/* more for the payment txn */
#define CWI   CID + 1
#define CDI   CWI + 1
#define CLT_P CDI + 1
#define HAM   CLT_P + 1
#define MAXPAYMENTVALS HAM + 1
/* more for the neworder txn */
#define SP00  CID + 1
#define IID00 SP00 + 1
```

```
#define QTY00 IID00 + 1
#define SP01  QTY00 + 1
#define IID01 SP01 + 1
#define QTY01 IID01 + 1
#define SP02  QTY01 + 1
#define IID02 SP02 + 1
#define QTY02 IID02 + 1
#define SP03  QTY02 + 1
#define IID03 SP03 + 1
#define QTY03 IID03 + 1
#define SP04  QTY03 + 1
#define IID04 SP04 + 1
#define QTY04 IID04 + 1
#define SP05  QTY04 + 1
#define IID05 SP05 + 1
#define QTY05 IID05 + 1
#define SP06  QTY05 +1
#define IID06 SP06 + 1
#define QTY06 IID06 + 1
#define SP07  QTY06 + 1
#define IID07 SP07 + 1
#define QTY07 IID07 + 1
#define SP08  QTY07 + 1
#define IID08 SP08 + 1
#define QTY08 IID08 + 1
#define SP09  QTY08 + 1
#define IID09 SP09 + 1
#define QTY09 IID09 + 1
#define SP10  QTY09 + 1
#define IID10 SP10 + 1
#define QTY10 IID10 + 1
#define SP11  QTY10 + 1
#define IID11 SP11 + 1
#define QTY11 IID11 + 1
#define SP12  QTY11 + 1
#define IID12 SP12 + 1
#define QTY12 IID12 + 1
#define SP13  QTY12 + 1
#define IID13 SP13 + 1
#define QTY13 IID13 + 1
#define SP14  QTY13 + 1
#define IID14 SP14 + 1
#define QTY14 IID14 + 1
#define MAXNEWORDERVALS QTY14 + 1

#if 0
#define PARSE_QUERY_STRING(pQueryString,varMax,charTable,valTable)\
  {\
    int ii;\
    char *ptr, *tmpPtr;\
    ptr = pQueryString;\
    for (ii=0; ii < varMax; ii++)\
    {\
      if ( !(tmpPtr=strstr(ptr, stringTable[ii])) )\
        valTable[ii] = NULL;\
      else\
      {\
        ptr = tmpPtr;\
        if ( !(ptr=strchr(ptr, '=')) )\
    valTable[ii] = NULL;\
        else\
    valTable[ii] = ++ptr;\
      }\
    }\
  }
#else
#define PARSE_QUERY_STRING(pQueryString,varMax,charTable,valTable)\
{\
  int ii;\
  char *ptr;\
  int iKey;\
  ptr = pQueryString;\
  for( ii=0; ii<varMax; ii++ ) {\
    iKey = charTable[ii];\
    valTable[ii] = NULL;\
    if( iKey == *ptr && '=' == *++ptr ) {\
      valTable[ii] = ++ptr;\
    }\
    while( *ptr ) {\
      if( '&' == *ptr ) {\
  ptr++;\
  break;\
      }\
      ptr++;\
    }\
  }\
}
#endif

typedef struct _FORMINDEXES
{
  int iStartIndex;    // index into the form char array for values
  int iLen;           // length of the current value field
} FORM_INDEXES;

GLOBAL(FORM_INDEXES deliveryFormIndexesI[4], { 0 } );
GLOBAL(FORM_INDEXES deliveryFormIndexesP[33], { 0 });
GLOBAL(FORM_INDEXES newOrderFormIndexes[4], { 0 });
GLOBAL(FORM_INDEXES newOrderResponseIndexes[136], { 0 });
GLOBAL(FORM_INDEXES orderStatusFormIndexes[4], { 0 });
GLOBAL(FORM_INDEXES orderStatusResponseIndexes[88], { 0 });
GLOBAL(FORM_INDEXES paymentFormIndexes[4], { 0 });
GLOBAL(FORM_INDEXES paymentResponseIndexes[38], { 0 });
GLOBAL(FORM_INDEXES stockLevelFormIndexes[5], { 0 });
GLOBAL(FORM_INDEXES stockLevelResponseIndexes[7], { 0 });

#ifdef MOD_TPCC_C
char deliveryStrs[] = {'6', '7'};
char newOrderStrs[] = {
'8', '9',
'A', 'B', 'C',
'D', 'E', 'F',
'G', 'H', 'I',
'J', 'K', 'L',
'M', 'N', 'O',
'P', 'Q', 'R',
'S', 'T', 'U',
'V', 'W', 'X',
'a', 'b', 'c',
'd', 'e', 'f',
'g', 'h', 'i',
'j', 'k', 'l',
'm', 'n', 'o',
'p', 'q', 'r',
's', 't', 'u'};
char orderStatusStrs[] = {'8', '9', 'Y'};
char paymentStrs[] = {'8', '9', 'Z', 'v', 'Y', 'w'};
char stockLevelStrs[] = {'x'};
#else
extern char deliveryStrs[];
extern char newOrderStrs[];
extern char orderStatusStrs[];
extern char paymentStrs[];
extern char stockLevelStrs[];
#endif /* MOD_TPCC_C */
GLOBAL(char szModName[FILENAMESIZE], { 0 });
#endif /* MOD_TPCC_H */



-------------------------------------------------
      mod_tpcc_template.c
-------------------------------------------------

/*
**  mod_tpcc.c -- Apache sample tpcc module
**  [Autogenerated via ``apxs -n tpcc -g'']
**
**  To play with this sample module, first compile it into a
**  DSO file and install it into Apache's libexec directory
**  by running:
**
**    $ apxs -c -i mod_tpcc.c
**
**  Then activate it in Apache's httpd.conf file, for instance
**  for the URL /tpcc, as follows:
**
**    #   httpd.conf
**    LoadModule tpcc_module libexec/mod_tpcc.so
**    <Location /tpcc>
**    SetHandler tpcc
**    </Location>
**
**  Then after restarting Apache via
**
**    $ apachectl restart
**
**  you immediately can request the URL /%NAME and watch for the
**  output of this module. This can be achieved for instance via:
**
**    $ lynx -mime_header http://localhost/tpcc
**
**  The output should be similar to the following one:
**
**    HTTP/1.1 200 OK
**    Date: Tue, 31 Mar 1998 14:42:22 GMT
**    Server: Apache/1.3.4 (Unix)
**    Connection: close
**    Content-Type: text/html
**
**    The sample page from mod_tpcc.c
*/

#include "httpd.h"
#include "http_config.h"
#include "http_protocol.h"
#include "ap_config.h"

/* The sample content handler */
static int tpcc_handler(request_rec *r)
{
    r->content_type = "text/html";
    ap_send_http_header(r);
    if (!r->header_only)
        ap_rputs("The sample page from mod_tpcc.c\n", r);
    return OK;
}

/* Dispatch list of content handlers */
static const handler_rec tpcc_handlers[] = {
    { "tpcc", tpcc_handler },
    { NULL, NULL }
```

```
};

/* Dispatch list for API hooks */
module MODULE_VAR_EXPORT tpcc_module = {
    STANDARD_MODULE_STUFF,
    NULL,                  /* module initializer
*/
    NULL,                  /* create per-dir    config structures
*/
    NULL,                  /* merge  per-dir    config structures
*/
    NULL,                  /* create per-server config structures
*/
    NULL,                  /* merge  per-server config structures
*/
    NULL,                  /* table of config file commands
*/
    tpcc_handlers,       /* [#8] MIME-typed-dispatched handlers */
    NULL,                  /* [#1] URI to filename translation
*/
    NULL,                  /* [#4] validate user id from request
*/
    NULL,                  /* [#5] check if the user is ok _here_
*/
    NULL,                  /* [#3] check access by host address
*/
    NULL,                  /* [#6] determine MIME type
*/
    NULL,                  /* [#7] pre-run fixups
*/
    NULL,                  /* [#9] log a transaction
*/
    NULL,                  /* [#2] header parser
*/
    NULL,                  /* child_init
*/
    NULL,                  /* child_exit
*/
    NULL                   /* [#0] post read-request
*/
#ifdef EAPI
   ,NULL,                  /* EAPI: add_module
*/
    NULL,                  /* EAPI: remove_module
*/
    NULL,                  /* EAPI: rewrite_command
*/
    NULL                   /* EAPI: new_connection
*/
#endif
};



-------------------------------------------------

      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) 1998 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 <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/timeb.h>
#include <sys/time.h>

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

#define ORACLE_DB_C

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

#define DEADLOCKRETRIES 6

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

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

/* prototypes */
int ORAReadRegistrySettings(void);
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(void)== fd)
    {
      return(ERR_DB_ERROR);
    }
    while (fgets((char *)linebuf+nulpt, SQL_BUF_SIZE,fd))
    {
        nulpt = strlen((char *)linebuf);
    }
    return(nulpt);
}

int getfile(char *filename, text *filebuf)
{
  text parsbuf[SQL_BUF_SIZE];

  strcpy(parsbuf, szTpccLogPath);
  strcat(parsbuf, filename);
  return(sqlfile(parsbuf, filebuf));
```

```
}


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

  return ERR_DB_SUCCESS;

}

int TPCCShutdownDB(void)
{

  bTpccExit = TRUE;

  /* Add Oracle specific code */

  return ERR_DB_SUCCESS;
}


int ocierror(char *fname, int lineno, OraContext *p, sword status)
{
  text 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");
    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(CallersContext *pCC, int iTermId,
int iSyncId,
 *  OraContext **dbproc, char *server, char *database, char *user,
 *  char *password, char *app, int *spid, long *pack_size)
 *
 * PURPOSE: This function opens the sql connection for use.
 *
 * ARGUMENTS: CallersContext  *pCC  passed in structure pointer
from inetsrv.
 *    int     iTermId   terminal id of browser
 *    int     iSyncId   sync id of browser
 *    OraContext    **dbproc  pointer to returned OraContext
 *    char      *server   SQL server name
 *    char      *database SQL server database
 *    char      *user   user name
 *    char      *password user password
 *    char      *app    pointer to returned application array
 *    int     *spid   pointer to returned spid
 *    long      *pack_size  pointer to returned default pack size
 *
 * RETURNS:   int 0 if successful
 *        1 if an error occurs
 *
 * COMMENTS:  None
 *
 */


int TPCCConnectDB(OraContext **dbproc, 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;
  char userstr[256];

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

  p = *dbproc;

  /* 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;

  sprintf(userstr,"%s/%s@%s",
    pLogin->szUser,pLogin->szPassword,pLogin->szServer);

/* OCIEnvCreate doesn't work on Linux
  OCIEnvCreate(&(p->tpcenv), OCI_DEFAULT | OCI_OBJECT, NULL, NULL,
NULL, NULL, (size_t) 0, NULL);
*/

  OCIERROR(p,OCIInitialize(OCI_DEFAULT|OCI_OBJECT,(dvoid *)0, NULL,
NULL, NULL));
  OCIERROR(p,OCIEnvInit(&(p->tpcenv), OCI_DEFAULT, NULL, (dvoid
**)0));

  OCIERROR(p,OCIHandleAlloc((dvoid *)p->tpcenv, (dvoid **)&(p-
>tpcsrv), OCI_HTYPE_SERVER,
     0 , (dvoid **)0));
  OCIERROR(p,OCIHandleAlloc((dvoid *)p->tpcenv, (dvoid **)&(p-
>errhp), OCI_HTYPE_ERROR,
     0 , (dvoid **)0));
  OCIERROR(p,OCIHandleAlloc((dvoid *)p->tpcenv, (dvoid **)&(p-
>datecvterrhp), OCI_HTYPE_ERROR,
     0 , (dvoid **)0));
  OCIERROR(p,OCIHandleAlloc((dvoid *)p->tpcenv, (dvoid **)&(p-
>tpcsvc), OCI_HTYPE_SVCCTX,
     0 , (dvoid **)0));
  if (RECOVERR != (OCIERROR(p, OCIServerAttach(p->tpcsrv, p->errhp,
                (text *)0, 0, OCI_DEFAULT))))
/*
  if (RECOVERR != (OCIERROR(p, OCIServerAttach(p->tpcsrv, p->errhp,
               userstr, strlen(userstr),
               OCI_DEFAULT))));
*/
    /*    return IRRECERR;  */
    return ERR_DB_ERROR;

  /*
OCIERROR(p, OCIServerAttach(p->tpcsrv, p->errhp,
               userstr, strlen(userstr),
               OCI_DEFAULT));*/
    /*
    {
      return IRRECERR;
    }
    */
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-
>szUser,
       (ub4)strlen(pLogin->szUser),OCI_ATTR_USERNAME, p->errhp);
OCIAttrSet((dvoid *)p->tpcusr, OCI_HTYPE_SESSION,
       (dvoid *)pLogin->szPassword,
       (ub4)strlen(pLogin->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), 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(OraContext *dbproc, CallersContext *pCC){

  /* Add Oracle specific code */


   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(OraContext *dbproc, pStockLevelData
pStockLevel)
{


  int tries,status;
  StockLevelData *pbindvars;
#ifdef DEBUG
  struct timeval tmp1,tmp2;
  struct timezone tz;
  unsigned delta;
#endif

  pbindvars = &dbproc->bindvars.info.stockLevel;

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

#ifdef DEBUG
  gettimeofday (&tmp1, &tz);
#endif


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

    status = tkvcs(dbproc);
  }
```

```
#ifdef DEBUG
  gettimeofday (&tmp2, &tz);
  delta=(tmp2.tv_sec-tmp1.tv_sec)*1000000+tmp2.tv_usec-
tmp1.tv_usec;
  if (delta > 60000000) {
  TPCCErr("SL:%10.10d:%5.5d\n", delta,pbindvars->w_id);
  }
#endif

  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
 *
 */

int TPCCNewOrderDB( OraContext *dbproc, pNewOrderData  pNewOrder)
{

  int tries,status;
  int ii;
  int jj;
  int datebufsize;
#ifdef DEBUG
  struct timeval tmp1,tmp2;
  struct timezone tz;
  unsigned delta;
#endif
  OCIError *datecvterrhp = dbproc->datecvterrhp;
  unsigned char localcr_date[7];

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

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

  ntemp->n_retry = 0;

  memcpy(pbindvars, pNewOrder, sizeof(NewOrderData));
  for(jj= 0; jj<MAX_OL; jj++)
    {
      ntemp->nol_i_id[jj] = pbindvars->o_ol[jj].ol_i_id;
      ntemp->nol_supply_w_id[jj] = pbindvars-
>o_ol[jj].ol_supply_w_id;
      ntemp->nol_quantity[jj] = pbindvars->o_ol[jj].ol_quantity;
    }
#ifdef DEBUG

  gettimeofday(&tmp1, &tz);
#endif

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

#ifdef DEBUG
  gettimeofday(&tmp2, &tz);
  delta=(tmp2.tv_sec-tmp1.tv_sec)*1000000+tmp2.tv_usec-
tmp1.tv_usec;
  if (delta > 60000000) {
  TPCCErr("NO:%10.10d:%5.5d:%2.2d\n", delta,pbindvars-
>w_id,pbindvars->d_id);
  }
#endif

  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++)
    {
      pNewOrder->o_ol[ii].ol_i_id = ntemp->nol_i_id[ii];
      pNewOrder->o_ol[ii].ol_supply_w_id = ntemp-
>nol_supply_w_id[ii];
      pNewOrder->o_ol[ii].ol_quantity = ntemp->nol_quantity[ii];
      strncpy(pNewOrder->o_ol[ii].i_name, ntemp->i_name[ii], 24);
      pNewOrder->o_ol[ii].s_quantity = ntemp->s_quantity[ii];
      pNewOrder->o_ol[ii].i_price = ntemp->i_price[ii]/100.0;
      pNewOrder->o_ol[ii].ol_amount = ntemp->nol_amount[ii]/100.0;
      pNewOrder->o_ol[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(OraContext *dbproc, pPaymentData pPayment)
{
  int tries;
  int status;
  int datebufsize;
  float ftmp;
#ifdef DEBUG
  struct timeval tmp1, tmp2;
  struct timezone tz;
  unsigned delta;
#endif
  OCIError *datecvterrhp = dbproc->datecvterrhp;

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

  ptemp->p_retry = 0;

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

  /* the db is stored in pennies - convert input to cents. */
  ftmp=pbindvars->h_amount*100.0;
  ptemp->h_amount = (int)(ftmp);
#ifdef DEBUG
  gettimeofday(&tmp1, &tz);
#endif

  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);

  }
```

```
#ifdef DEBUG
  gettimeofday(&tmp2, &tz);
  delta=(tmp2.tv_sec-tmp1.tv_sec)*1000000+tmp2.tv_usec-
tmp1.tv_usec;
  if (delta > 60000000) {
  TPCCErr("PY:%10.10d:%5.5d:%2.2d:%5.5d:%2.2d\n", delta,pbindvars-
>w_id,pbindvars->d_id,pbindvars->c_w_id,pbindvars->c_d_id);
  }
#endif

  memcpy(pPayment, pbindvars, sizeof(PaymentData));
  /* datebufsize = the size of c_since_str in paytemp struct */
 datebufsize=11;
  /* 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 *) "DD-
MM-YYYY", 10, (text *) 0, 0, &datebufsize, &ptemp->c_since_str);
  /* cvtdmy(ptemp->customer_sdate, ptemp->c_since_str); */
  /* datebufsize = the size of h_date string in paytemp struct */
 datebufsize=DATE_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 = (float)(ptemp->c_credit_lim)/100.0;
  pPayment->c_discount = (float)(ptemp->c_discount)*100.0;
  pPayment->c_balance = (float)(pPayment->c_balance)/100.0;
  pPayment->h_amount = (float)(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(OraContext *dbproc, pOrderStatusData
pOrderStatus)
{

  int tries,status;
  int ii;
#ifdef DEBUG
  struct timeval tmp1, tmp2;
  struct timezone tz;
  unsigned delta;
#endif
  OrderStatusData *pbindvars = &(dbproc-
>bindvars.info.orderStatus);
  ordtemp *otemp = &(dbproc->tempvars.ord);

  memcpy(pbindvars, pOrderStatus, sizeof(OrderStatusData));
#ifdef DEBUG
  gettimeofday (&tmp1, &tz);
#endif

  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);
 }
#ifdef DEBUG
  gettimeofday(&tmp2, &tz);
  delta=(tmp2.tv_sec-tmp1.tv_sec)*1000000+tmp2.tv_usec-
tmp1.tv_usec;
  if (delta > 60000000) {
  TPCCErr("OS:%10.10d:%5.5d:%2.2d\n", delta,pbindvars-
>w_id,pbindvars->d_id);
  }
#endif

  if (status == ERR_DB_ERROR)
  {
   TPCCErr("TPCCOrderStatusDB %d\n",status);
   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( OraContext *dbproc, pDeliveryData pDeliveryData
)
{


  int retries = 0;
  int status;
  DeliveryData *pbindvars;
#ifdef DEBUG
  struct timeval tmp1, tmp2;
  struct timezone tz;
  unsigned delta;
  gettimeofday(&tmp1, &tz);
#endif

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

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

    status = tkvcd(pDeliveryData, dbproc);
  }
#ifdef DEBUG
  gettimeofday(&tmp2, &tz);
  delta=(tmp2.tv_sec-tmp1.tv_sec)*1000000+tmp2.tv_usec-
tmp1.tv_usec;
  if (delta > 60000000) {
  TPCCErr("DY:%10.10d:%5.5d\n", delta,pbindvars->w_id);
  }
#endif

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


}

int TPCCGetLastDBErrorDB(OraContext *dbproc)
{


  /* Add Oracle specific code */

  return ERR_DB_SUCCESS;

}


/* FUNCTION: int TPCCCheckpointDB(CallersContext *pCC, int iTermId,
int iSyncId, OraContext *dbproc, int deadlock_retry, Checkpoint
*pCheckpoint
 *
 * PURPOSE: This function does a checkpoinnt 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
 *    Checkpoint  *Checkpoint pointer to Checkpoint data
 *    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
 *
 */


#define CHECKPOINT_TXT "alter system switch logfile"

int TPCCCheckpointDB (OraContext *dbproc, pCheckpointData
pCheckpoint ) {

  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 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.
-*/
/*+
-*/
/*+
-*/
/****************************************************************
*********-*/

/*
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                          Conversion to run under Linux and Apache
 *      11/22/2002  Bryon Georgson HP
 *            Conversion to latest oracle 10i kit.
 *
*/

#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 MSG_SIZ 512
#define DATE_SIZ 20  /* DD-MM-YYYY.HH:MI:SS  plus null terminator
*/
#define NITEMS 15
#define NDISTS 10
#define ROWIDLEN 20
#define OCIROWLEN 20
#define DEL_DATE_LEN 7
#define SQL_BUF_SIZE 16384

#define FULLDATE "dd-mon-yy.hh24:mi:ss"
#define SHORTDATE "dd-mm-yyyy"

#ifndef NULLP
# define NULLP(x)  (x  *)NULL
#endif /* NULLP */

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

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


struct _delctx {
   ub2 del_d_id_len[NDISTS];
   ub2 del_o_id_len[NDISTS];
   ub2 w_id_len;
   ub2 d_id_len[NDISTS];
   ub2 o_c_id_len[NDISTS];
   ub2 sums_len[NDISTS];
   ub2 carrier_id_len;
   ub2 ordcnt_len;
   ub2 del_date_len;
#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
   ub2 inum_len;
#endif
   int del_o_id[NDISTS];
   int del_d_id[NDISTS];
   int o_c_id[NDISTS];
   int sums[NDISTS];
   OCIDate del_date;
   int carrier_id;
   int ordcnt;
   ub4 del_o_id_rcnt;
   ub4 del_d_id_rcnt;
   ub4 o_c_id_rcnt;
   ub4 sums_rcnt;
   int retry;
#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
   char inum[10];
#endif
   OCIStmt *curp1;
   OCIStmt *curp2;

   OCIBind *w_id_bp;
   OCIBind *d_id_bp;
   OCIBind *o_id_bp;
   OCIBind *o_c_id_bp;
   OCIBind *cr_date_bp;
   OCIBind *ordcnt_bp;
   OCIBind *sums_bp;
   OCIBind *del_date_bp;
   OCIBind *carrier_id_bp;
   OCIBind *retry_bp;
   int norow;
};
typedef struct _delctx delctx;


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



struct _newctx {
  ub2 nol_i_id_len[NITEMS];
  ub2 nol_supply_w_id_len[NITEMS];
  ub2 nol_quantity_len[NITEMS];
  ub2 nol_amount_len[NITEMS];
  ub2 s_quantity_len[NITEMS];
  ub2 i_name_len[NITEMS];
  ub2 i_price_len[NITEMS];
  ub2 s_dist_info_len[NITEMS];
  ub2 ol_o_id_len[NITEMS];
  ub2 ol_number_len[NITEMS];
  ub2 cons_len[NITEMS];
  ub2 s_remote_len[NITEMS];
  ub2 s_quant_len[NITEMS];
  ub2 ol_dist_info_len[NITEMS];
  sb2 s_bg_len[NITEMS];

  int ol_o_id[NITEMS];
  int ol_number[NITEMS];

  int s_remote[NITEMS];
  char s_dist_info[NITEMS][25];

  OCIStmt *curn1;
  OCIBind *ol_i_id_bp;
  OCIBind *ol_supply_w_id_bp;
  OCIBind *i_price_bp;
  OCIBind *i_name_bp;
  OCIBind *s_bg_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;
  OCIStmt *curn2;
  OCIBind *ol_quantity_bp;
  OCIBind *s_remote_bp;
  OCIBind *s_quantity_bp;
  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_o_id_bp;
  OCIBind *ol_amount_bp;

  sb2 w_id_len;
  ub2 d_id_len;
  ub2 c_id_len;
  ub2 o_all_local_len;
  ub2 o_ol_cnt_len;
  ub2 w_tax_len;
  ub2 d_tax_len;
  ub2 o_id_len;
  ub2 c_discount_len;
  ub2 c_credit_len;
  ub2 c_last_len;
  ub2 retries_len;
  ub2 cr_date_len;


  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 {
   ub2 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;
   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 *c_id_bp;
   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_last_bp;
    OCIBind *c_last_bp4;
    OCIBind *o_id_bp;
    OCIBind *c_rowid_bp;
    OCIBind *o_rowid_bp;
    OCIDefine *c_rowid_dp;
    OCIDefine *c_last_dp;
    OCIDefine *c_last_dp1;
    OCIDefine *c_id_dp;
    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_rowid_dp1;
    OCIDefine *o_rowid_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 *c_rowid_cust;
    OCIRowid *o_rowid;
    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;
  ub2 w_id_len;

  OCIBind *d_id_bp;
  OCIBind *d_id_bp1;
  ub2 d_id_len;

  OCIBind *c_w_id_bp;
  OCIBind *c_w_id_bp1;
  ub2 c_w_id_len;

  OCIBind *c_d_id_bp;
  OCIBind *c_d_id_bp1;
  ub2 c_d_id_len;

  OCIBind *c_id_bp;
  OCIBind *c_id_bp1;
  ub2 c_id_len;

  OCIBind *h_amount_bp;
  OCIBind *h_amount_bp1;
  ub2 h_amount_len;

  OCIBind *c_last_bp;
  OCIBind *c_last_bp1;
  ub2 c_last_len;

  OCIBind *w_street_1_bp;
  OCIBind *w_street_1_bp1;
  ub2 w_street_1_len;

  OCIBind *w_street_2_bp;
  OCIBind *w_street_2_bp1;
  ub2 w_street_2_len;

  OCIBind *w_city_bp;
  OCIBind *w_city_bp1;
  ub2 w_city_len;

  OCIBind *w_state_bp;
  OCIBind *w_state_bp1;
  ub2 w_state_len;

  OCIBind *w_zip_bp;
  OCIBind *w_zip_bp1;
  ub2 w_zip_len;

  OCIBind *d_street_1_bp;
  OCIBind *d_street_1_bp1;
  ub2 d_street_1_len;

  OCIBind *d_street_2_bp;
  OCIBind *d_street_2_bp1;
  ub2 d_street_2_len;

  OCIBind *d_city_bp;
  OCIBind *d_city_bp1;
  ub2 d_city_len;

  OCIBind *d_state_bp;
  OCIBind *d_state_bp1;
  ub2 d_state_len;

  OCIBind *d_zip_bp;
  OCIBind *d_zip_bp1;
  ub2 d_zip_len;

  OCIBind *c_first_bp;
  OCIBind *c_first_bp1;

  ub2 c_first_len;

  OCIBind *c_middle_bp;
  OCIBind *c_middle_bp1;
  ub2 c_middle_len;

  OCIBind *c_street_1_bp;
  OCIBind *c_street_1_bp1;
  ub2 c_street_1_len;

  OCIBind *c_street_2_bp;
  OCIBind *c_street_2_bp1;
  ub2 c_street_2_len;


  OCIBind *c_city_bp;
  OCIBind *c_city_bp1;
  ub2 c_city_len;

  OCIBind *c_state_bp;
  OCIBind *c_state_bp1;
  ub2 c_state_len;

  OCIBind *c_zip_bp;
  OCIBind *c_zip_bp1;
  ub2 c_zip_len;

  OCIBind *c_phone_bp;
  OCIBind *c_phone_bp1;
  ub2 c_phone_len;

  OCIBind *c_since_bp;
  OCIBind *c_since_bp1;
  ub2 c_since_len;

  OCIBind *c_credit_bp;
  OCIBind *c_credit_bp1;
  ub2 c_credit_len;

  OCIBind *c_credit_lim_bp;
  OCIBind *c_credit_lim_bp1;
  ub2 c_credit_lim_len;

  OCIBind *c_discount_bp;
  OCIBind *c_discount_bp1;
  ub2 c_discount_len;

  OCIBind *c_balance_bp;
  OCIBind *c_balance_bp1;
  ub2 c_balance_len;

  OCIBind *c_data_bp;
  OCIBind *c_data_bp1;
  ub2 c_data_len;

  OCIBind *h_date_bp;
  OCIBind *h_date_bp1;
  ub2 h_date_len;

  OCIBind *retries_bp;
  OCIBind *retries_bp1;
  ub2 retries_len;

  OCIBind *cr_date_bp;
  OCIBind *cr_date_bp1;
  ub2 cr_date_len;

  OCIBind *byln_bp;
  ub2 byln_len;
};
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_SIZ];
  OCIDate cr_date;
} deltemp;

typedef struct _newtemp {
  char entry_date[DATE_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];
  double c_discount;
  double w_tax;
  double d_tax;
  int n_retry;
} newtemp;

typedef struct _ordtemp {
  OCIDate entry_date;
  char entry_date_str[DATE_SIZ + 1];
  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][11];
} ordtemp;

typedef struct _paytemp {
  char h_date[DATE_SIZ];
  OCIDate customer_sdate;
  char c_since_str[11];
  OCIDate cr_date;
  double 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 */
  TransactionData 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 OCIBNDPL(stmp,bndp,p,sqlvar,progv,progvl,ftype,alen) \
            DISCARD ocierror(__FILE__,__LINE__,(p), \
              OCIBindByName((stmp),&(bndp),(p->errhp),(CONST text
*)(sqlvar), \
               (sb4)strlen((CONST char *)(sqlvar)),
(dvoid*)(progv),(progvl),(ftype),\
                  NULLP(dvoid),(alen), NULLP(ub2),
0,NULLP(ub4),OCI_DEFAULT))

#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 OCIBNDPLA(stmp,bndp,p,sqlvar,progv,progv1,ftype,alen,ms,cu)
\
  DISCARD ocierror(__FILE__,__LINE__,(p), \
   OCIBindByName((stmp),&(bndp),(p->errhp),(const char *)(sqlvar),
\
     (sb4)strlen((CONST char *) (sqlvar)),(void *)(progv), \
     (progv1),(ftype),NULL,(alen),NULL,(ms),(cu),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,p,pos,progv,progvl,ftype,indp,alen,arcode) \

      OCIDefineByPos((stmp),&(dfnp),(p->errhp),(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 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.
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP Corporation
 *                      - Conversion to run under Linux
 *      10/31/2002      Bryon Georgson, HP Corporation
 *                      - Conversion to Oracle 10i
 *
 */

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

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

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

#include <tpcc.h>

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

/* prototypes */
int getfile(char *filename, text *filebuf);


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+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",
                        day,month,year,hour,min,sec);
        return;
}


/* 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) \
      order by ol_o_id desc "

int 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);

}

int 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);
    TPCCErr("Error in StockLevel Transaction warehouse: %d \tcurs
errcode: %d\n",p->bindvars.info.stockLevel.w_id,errcode);
    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_INIT "BEGIN inittpcc.init_pay; END;"


int 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 */
   DISCARD OCIERROR(p,OCIHandleAlloc(p->tpcenv, (dvoid **)(&(pctx-
>curpi)),
          OCI_HTYPE_STMT,0,(dvoid**)0));
   DISCARD OCIERROR(p,OCIHandleAlloc(p->tpcenv, (dvoid **)(&(pctx-
>curp0)),
          OCI_HTYPE_STMT,0,(dvoid**)0));
   DISCARD 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);
   DISCARD OCIERROR(p,OCIStmtPrepare(pctx->curpi, p->errhp, stmbuf,
          strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));
   DISCARD OCIERROR(p,OCIStmtExecute(p->tpcsvc,pctx->curpi,p-
>errhp,1,0,
    NULLP(CONST OCISnapshot),NULLP(OCISnapshot),OCI_DEFAULT));
   /* customer id !=  0, go by customer id */
   if(ERR_DB_ERROR == getfile("paynz.sql",stmbuf))
  {
    TPCCErr("Error opening the file paynz.sql");
    return ERR_DB_ERROR;
  }
   DISCARD OCIERROR(p,OCIStmtPrepare(pctx->curp0, p->errhp, stmbuf,
          strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));
   /* customer id ==  0, go by last name */
   if(ERR_DB_ERROR == getfile("payz.sql",stmbuf))
  {
    TPCCErr("Error opening the file payz.sql");
    return ERR_DB_ERROR;
  }
   DISCARD OCIERROR(p,OCIStmtPrepare(pctx->curp1, p->errhp, stmbuf,
          strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));
  pctx->w_id_len = SIZ(pPay->w_id);
  pctx->d_id_len = SIZ(pPay->d_id);
  pctx->c_w_id_len = SIZ(pPay->c_w_id);
  pctx->c_d_id_len = SIZ(pPay->c_d_id);
  pctx->c_id_len = 0;
  pctx->h_amount_len = SIZ(ptemp->h_amount);
  pctx->c_last_len = 0;
  pctx->w_street_1_len = 0;
  pctx->w_street_2_len = 0;
  pctx->w_city_len = 0;
  pctx->w_state_len = 0;
  pctx->w_zip_len = 0;
  pctx->d_street_1_len = 0;
  pctx->d_street_2_len = 0;
  pctx->d_city_len = 0;
  pctx->d_state_len = 0;
  pctx->d_zip_len = 0;
  pctx->c_first_len = 0;
  pctx->c_middle_len = 0;
  pctx->c_street_1_len = 0;
  pctx->c_street_2_len = 0;
  pctx->c_city_len = 0;
  pctx->c_state_len = 0;
  pctx->c_zip_len = 0;
  pctx->c_phone_len = 0;
  pctx->c_since_len = 0;
  pctx->c_credit_len = 0;
  pctx->c_credit_lim_len = 0;
  pctx->c_discount_len = 0;
  pctx->c_balance_len = sizeof(double);
  pctx->c_data_len = 0;
  pctx->h_date_len = 0;
  pctx->retries_len = 0;
  pctx->cr_date_len = sizeof(ptemp->cr_date);

   /* bind variables */

   OCIBNDPL(pctx->curp0, pctx->w_id_bp, p,":w_id",ADR(pPay-
>w_id),SIZ(int),
          SQLT_INT, NULL);
   OCIBNDPL(pctx->curp0, pctx->d_id_bp, p,":d_id",ADR(pPay-
>d_id),SIZ(int),
          SQLT_INT, 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);
   OCIBNDPL(pctx->curp0, pctx->h_amount_bp,
p,":h_amount",ADR(ptemp->h_amount),
         SIZ(int),SQLT_INT, &pctx->h_amount_len);
   OCIBNDPL(pctx->curp0, pctx->c_last_bp, p,":c_last",pPay->c_last,
         SIZ(pPay->c_last),SQLT_STR, &pctx->c_last_len);
   OCIBNDPL(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_len);
   OCIBNDPL(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_len);
   OCIBNDPL(pctx->curp0, pctx->w_city_bp, p,":w_city",pPay->w_city,
    SIZ(pPay->w_city),SQLT_STR, &pctx->w_city_len);
   OCIBNDPL(pctx->curp0, pctx->w_state_bp, p,":w_state",pPay-
>w_state,
     SIZ(pPay->w_state), SQLT_STR, &pctx->w_state_len);
   OCIBNDPL(pctx->curp0, pctx->w_zip_bp, p,":w_zip",pPay->w_zip,
     SIZ(pPay->w_zip), SQLT_STR, &pctx->w_zip_len);
   OCIBNDPL(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_len);
   OCIBNDPL(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_len);
   OCIBNDPL(pctx->curp0, pctx->d_city_bp, p,":d_city",pPay->d_city,
     SIZ(pPay->d_city), SQLT_STR, &pctx->d_city_len);
   OCIBNDPL(pctx->curp0, pctx->d_state_bp, p,":d_state",pPay-
>d_state,
     SIZ(pPay->d_state), SQLT_STR, &pctx->d_state_len);
   OCIBNDPL(pctx->curp0, pctx->d_zip_bp, p,":d_zip",pPay->d_zip,
     SIZ(pPay->d_zip), SQLT_STR, &pctx->d_zip_len);
   OCIBNDPL(pctx->curp0, pctx->c_first_bp, p,":c_first",pPay-
>c_first,
     SIZ(pPay->c_first), SQLT_STR, &pctx->c_first_len);
   OCIBNDPL(pctx->curp0, pctx->c_middle_bp, p,":c_middle", pPay-
>c_middle,2,
          SQLT_AFC, &pctx->c_middle_len);
   OCIBNDPL(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_len);
   OCIBNDPL(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_len);
   OCIBNDPL(pctx->curp0, pctx->c_city_bp, p,":c_city",pPay->c_city,
     SIZ(pPay->c_city), SQLT_STR, &pctx->c_city_len);
   OCIBNDPL(pctx->curp0, pctx->c_state_bp, p,":c_state",pPay-
>c_state,
     SIZ(pPay->c_state), SQLT_STR,&pctx->c_state_len);
   OCIBNDPL(pctx->curp0, pctx->c_zip_bp, p,":c_zip",pPay->c_zip,
     SIZ(pPay->c_zip), SQLT_STR, &pctx->c_zip_len);
   OCIBNDPL(pctx->curp0, pctx->c_phone_bp, p,":c_phone",pPay-
>c_phone,
     SIZ(pPay->c_phone), SQLT_STR, &pctx->c_phone_len);
   OCIBNDPL(pctx->curp0,pctx->c_since_bp,p,":c_since",
          ADR(ptemp->customer_sdate),SIZ(ptemp-
>customer_sdate),SQLT_ODT,
          &pctx->c_since_len);
   OCIBNDPL(pctx->curp0, pctx->c_credit_bp, p,":c_credit",pPay-
>c_credit,
          SIZ(pPay->c_credit),SQLT_CHR, &pctx->c_credit_len);
   OCIBNDPL(pctx->curp0,pctx->c_credit_lim_bp,p,":c_credit_lim",
           ADR(ptemp->c_credit_lim),SIZ(int),SQLT_INT,&pctx-
>c_credit_lim_len);
   OCIBNDPL(pctx->curp0, pctx->c_discount_bp, p,":c_discount",
           ADR(ptemp->c_discount),SIZ(ptemp->c_discount),SQLT_FLT,
           &pctx->c_discount_len);
   OCIBNDPL(pctx->curp0,pctx->c_balance_bp,p,":c_balance",ADR(pPay-
>c_balance),
           SIZ(pPay->c_balance),SQLT_FLT, &pctx->c_balance_len);
   OCIBNDPL(pctx->curp0, pctx->c_data_bp, p,":c_data",pPay->c_data,
     SIZ(pPay->c_data),SQLT_STR, &pctx->c_data_len);
   OCIBNDPL(pctx->curp0, pctx->retries_bp, p,":retry",ADR(ptemp-
>p_retry),
           SIZ(ptemp->p_retry), SQLT_INT, &pctx->retries_len);
   OCIBNDPL(pctx->curp0, pctx->cr_date_bp, p,":cr_date",ADR(ptemp-
>cr_date),
           SIZ(ptemp->cr_date),SQLT_ODT, &pctx->cr_date_len);

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

   OCIBNDPL(pctx->curp1, pctx->w_id_bp1, p,":w_id",ADR(pPay-
>w_id),SIZ(int),
          SQLT_INT, &pctx->w_id_len);
   OCIBNDPL(pctx->curp1, pctx->d_id_bp1, p,":d_id",ADR(pPay->d_id),
SIZ(int),
          SQLT_INT, &pctx->d_id_len);
   OCIBND(pctx->curp1, pctx->c_w_id_bp1, p,":c_w_id",ADR(pPay-
>c_w_id),SIZ(int),
    SQLT_INT);
   OCIBND(pctx->curp1, pctx->c_d_id_bp1, p,":c_d_id",ADR(pPay-
>c_d_id),SIZ(int),
    SQLT_INT);
   OCIBNDPL(pctx->curp1, pctx->c_id_bp1, p,":c_id",ADR(pPay->c_id),
SIZ(int),
          SQLT_INT, &pctx->c_id_len);
   OCIBNDPL(pctx->curp1,pctx->h_amount_bp1,p,":h_amount",ADR(ptemp-
>h_amount),
          SIZ(int),SQLT_INT, &pctx->h_amount_len);
   OCIBND(pctx->curp1,pctx->c_last_bp1, p,":c_last",pPay->c_last,
          SIZ(pPay->c_last), SQLT_STR);
   OCIBNDPL(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_len);
   OCIBNDPL(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_len);
   OCIBNDPL(pctx->curp1,pctx->w_city_bp1,p,":w_city",pPay->w_city,
          SIZ(pPay->w_city),SQLT_STR, &pctx->w_city_len);
   OCIBNDPL(pctx->curp1, pctx->w_state_bp1, p,":w_state",pPay-
>w_state,
     SIZ(pPay->w_state), SQLT_STR, &pctx->w_state_len);
   OCIBNDPL(pctx->curp1, pctx->w_zip_bp1, p,":w_zip",pPay->w_zip,
     SIZ(pPay->w_zip), SQLT_STR, &pctx->w_zip_len);
   OCIBNDPL(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_len);
   OCIBNDPL(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_len);
   OCIBNDPL(pctx->curp1, pctx->d_city_bp1, p,":d_city", pPay-
>d_city,
           SIZ(pPay->d_city), SQLT_STR, &pctx->d_city_len);
   OCIBNDPL(pctx->curp1, pctx->d_state_bp1, p,":d_state", pPay-
>d_state,
           SIZ(pPay->d_state), SQLT_STR, &pctx->d_state_len);
   OCIBNDPL(pctx->curp1, pctx->d_zip_bp1, p,":d_zip",pPay->d_zip,
     SIZ(pPay->d_zip), SQLT_STR, &pctx->d_zip_len);
   OCIBNDPL(pctx->curp1, pctx->c_first_bp1, p,":c_first",pPay-
>c_first,
           SIZ(pPay->c_first), SQLT_STR, &pctx->c_first_len);
   OCIBNDPL(pctx->curp1, pctx->c_middle_bp1, p,":c_middle", pPay-
>c_middle,2,
           SQLT_AFC, &pctx->c_middle_len);
   OCIBNDPL(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_len);
   OCIBNDPL(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_len);
   OCIBNDPL(pctx->curp1, pctx->c_city_bp1, p,":c_city",pPay-
>c_city,
     SIZ(pPay->c_city),SQLT_STR, &pctx->c_city_len);
   OCIBNDPL(pctx->curp1, pctx->c_state_bp1, p,":c_state",pPay-
>c_state,
     SIZ(pPay->c_state),SQLT_STR, &pctx->c_state_len);
   OCIBNDPL(pctx->curp1, pctx->c_zip_bp1, p,":c_zip",pPay->c_zip,
     SIZ(pPay->c_zip), SQLT_STR, &pctx->c_zip_len);
   OCIBNDPL(pctx->curp1, pctx->c_phone_bp1, p,":c_phone",pPay-
>c_phone,
     SIZ(pPay->c_phone), SQLT_STR, &pctx->c_phone_len);
   OCIBNDPL(pctx->curp1, pctx->c_since_bp1, p,":c_since",
           ADR(ptemp->customer_sdate),SIZ(ptemp-
>customer_sdate),SQLT_ODT,
     &pctx->c_since_len);
   OCIBNDPL(pctx->curp1, pctx->c_credit_bp1, p,":c_credit", pPay-
>c_credit,
           SIZ(pPay->c_credit),SQLT_CHR, &pctx->c_credit_len);
   OCIBNDPL(pctx->curp1, pctx->c_credit_lim_bp1, p,":c_credit_lim",
           ADR(ptemp->c_credit_lim),SIZ(int), SQLT_INT,&pctx-
>c_credit_lim_len);
   OCIBNDPL(pctx->curp1, pctx->c_discount_bp1, p,":c_discount",
           ADR(ptemp->c_discount),SIZ(ptemp->c_discount),SQLT_FLT,
           &pctx->c_discount_len);
   OCIBNDPL(pctx->curp1,pctx-
>c_balance_bp1,p,":c_balance",ADR(pPay->c_balance),
           SIZ(double),SQLT_FLT, &pctx->c_balance_len);
   OCIBNDPL(pctx->curp1, pctx->c_data_bp1, p,":c_data",pPay-
>c_data,
     SIZ(pPay->c_data), SQLT_STR, &pctx->c_data_len);
   OCIBNDPL(pctx->curp1, pctx->retries_bp1, p,":retry", ADR(ptemp-
>p_retry),
           SIZ(int), SQLT_INT, &pctx->retries_len);
   OCIBNDPL(pctx->curp1, pctx->cr_date_bp1, p,":cr_date",
ADR(ptemp->cr_date),
           SIZ(ptemp->cr_date), SQLT_ODT, &pctx->cr_date_len);

 return (ERR_DB_SUCCESS);
}


int 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(localcr_date);
  cvtdmyhms(localcr_date,ptemp->h_date);
  OCIDateFromText(datecvterrhp,ptemp->h_date,strlen(ptemp-
>h_date),"DD-MM-YYYY HH24:MI:SS",21,(text *) 0, 0,&ptemp->cr_date);
  pctx->w_id_len = SIZ(pPay->w_id);
  pctx->d_id_len = SIZ(pPay->d_id);
  pctx->c_w_id_len = 0;
  pctx->c_d_id_len = 0;
  pctx->c_id_len = 0;
  pctx->h_amount_len = SIZ(ptemp->h_amount);
  pctx->c_last_len = SIZ(pPay->c_last);
  pctx->w_street_1_len = 0;
  pctx->w_street_2_len = 0;
  pctx->w_city_len = 0;
  pctx->w_state_len = 0;
  pctx->w_zip_len = 0;
  pctx->d_street_1_len = 0;
  pctx->d_street_2_len = 0;
  pctx->d_city_len = 0;
  pctx->d_state_len = 0;
  pctx->d_zip_len = 0;
  pctx->c_first_len = 0;
  pctx->c_middle_len = 0;
  pctx->c_street_1_len = 0;
  pctx->c_street_2_len = 0;
  pctx->c_city_len = 0;
  pctx->c_state_len = 0;
  pctx->c_zip_len = 0;
  pctx->c_phone_len = 0;
  pctx->c_since_len = 0;
  pctx->c_credit_len = 0;
  pctx->c_credit_lim_len = 0;
  pctx->c_discount_len = 0;
  pctx->c_balance_len = sizeof(double);
  pctx->c_data_len = 0;
  pctx->h_date_len = 0;
  pctx->retries_len = 0;
  pctx->cr_date_len = sizeof(ptemp->cr_date);
  pctx->retries_len = sizeof(ptemp->p_retry);
  if(pPay->byname)
  {
    execstatus=OCIStmtExecute(p->tpcsvc,pctx->curp1,p->errhp,1,0,
               NULLP(CONST OCISnapshot),NULLP(OCISnapshot),
               OCI_DEFAULT|OCI_COMMIT_ON_SUCCESS);
  }
  else
  {
    execstatus=OCIStmtExecute(p->tpcsvc,pctx->curp0,p->errhp,1,0,
               NULLP(CONST OCISnapshot),NULLP(OCISnapshot),
               OCI_DEFAULT|OCI_COMMIT_ON_SUCCESS);
  }
  if(execstatus != OCI_SUCCESS) {
     errcode = OCIERROR(p,execstatus);
     TPCCErr("Error in Payment Transaction curp0 or curp1 errcode:
%d\n",
         errcode);
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     errcode = OCIERROR(p,execstatus);
     if((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR) ||
         (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_last, c_d_id, c_w_id, c_first"

#define SQL_ORD_CUR1 "SELECT /*+ USE_NL(cust) INDEX_DESC(ordr
iordr2) */ \
                 c_id, c_balance, c_first, c_middle, c_last, \
                 o_id, o_entry_d, o_carrier_id, o_ol_cnt,
ordr.rowid \
                 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_c_id DESC, o_d_id DESC, o_w_id DESC,
o_id DESC"

#define SQL_ORD_CUR2 "SELECT /*+ USE_NL(cust) INDEX_DESC (ordr
iordr2) */ \
                 c_balance, c_first, c_middle, c_last, \
                 o_id, o_entry_d, o_carrier_id, o_ol_cnt,
ordr.rowid \
                 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 DESC, o_d_id DESC, o_w_id DESC,
o_id DESC"

#define SQL_ORD_CUR3 "SELECT /*+ ORDERED USE_NL(ordl) CLUSTER
(ordl) */ \
                ol_i_id,ol_supply_w_id,ol_quantity,ol_amount,
ol_delivery_d \
                FROM ordr, ordl \
                WHERE ordr.rowid = :ordr_rowid \
                  AND o_id = ol_o_id AND ol_d_id = o_d_id AND
ol_w_id = o_w_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 "

int tkvcoinit (OrderStatusData *pOrd,
     OraContext *p)
{
  int i;
  text stmbuf[8192];
  ordtemp *otemp = &(p->tempvars.ord);
  ordctx *octx = &(p->octx);
  DISCARD memset(octx,(char)0,sizeof(ordctx));
  octx->cs = 1;
  octx->norow = 0;
  octx->somerows = 10;
/* get the rowid handles */
   OCIERROR(p,OCIDescriptorAlloc((dvoid *)p->tpcenv,(dvoid
**)&octx->o_rowid,
    (ub4)OCI_DTYPE_ROWID, (size_t) 0, (dvoid **)0));
  for(i=0;i<100;i++) {
   DISCARD OCIERROR(p,OCIDescriptorAlloc(p->tpcenv,
          (dvoid**)&octx-
>c_rowid_ptr[i],OCI_DTYPE_ROWID,0,(dvoid**)0));
  }
   DISCARD OCIERROR(p,
   OCIHandleAlloc(p->tpcenv,(dvoid**)&octx-
>curo0,OCI_HTYPE_STMT,0,(dvoid**)0));
   DISCARD OCIERROR(p,
   OCIHandleAlloc(p->tpcenv,(dvoid**)&octx-
>curo1,OCI_HTYPE_STMT,0,(dvoid**)0));
   DISCARD OCIERROR(p,
   OCIHandleAlloc(p->tpcenv,(dvoid**)&octx-
>curo2,OCI_HTYPE_STMT,0,(dvoid**)0));
   DISCARD OCIERROR(p,
   OCIHandleAlloc(p->tpcenv,(dvoid**)&octx-
>curo3,OCI_HTYPE_STMT,0,(dvoid**)0));
   DISCARD 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 of
rowid's back*/
   DISCARD sprintf((char *) stmbuf, SQL_ORD_CUR0);
   DISCARD OCIERROR(p,
   OCIStmtPrepare(octx->curo0,p->errhp,stmbuf,(ub4)strlen((char
*)stmbuf),
             OCI_NTV_SYNTAX,OCI_DEFAULT));
   DISCARD 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 */
   DISCARD sprintf((char *) stmbuf, SQL_ORD_CUR1);
   DISCARD OCIERROR(p,
   OCIStmtPrepare(octx->curo1,p->errhp,stmbuf,(ub4)strlen((char
*)stmbuf),
           OCI_NTV_SYNTAX,OCI_DEFAULT));
   DISCARD 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  */
   DISCARD sprintf((char *) stmbuf, SQL_ORD_CUR2);
   DISCARD OCIERROR(p,
   OCIStmtPrepare(octx->curo2,p->errhp,stmbuf,(ub4)strlen((char
*)stmbuf),
           OCI_NTV_SYNTAX,OCI_DEFAULT));
   DISCARD OCIERROR(p,
   OCIAttrSet(octx->curo2,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
        OCI_ATTR_PREFETCH_ROWS,p->errhp));
   DISCARD sprintf((char *) stmbuf, SQL_ORD_CUR3);
   DISCARD OCIERROR(p,
   OCIStmtPrepare(octx->curo3,p->errhp,stmbuf,(ub4)strlen((char
*)stmbuf),
           OCI_NTV_SYNTAX,OCI_DEFAULT));
   DISCARD OCIERROR(p,
   OCIAttrSet(octx->curo3,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
        OCI_ATTR_PREFETCH_ROWS,p->errhp));
   DISCARD sprintf((char *) stmbuf, SQL_ORD_CUR4);
   DISCARD OCIERROR(p,
   OCIStmtPrepare(octx->curo4,p->errhp,stmbuf,(ub4)strlen((char
*)stmbuf),

           OCI_NTV_SYNTAX,OCI_DEFAULT));
   DISCARD 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_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_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_len = sizeof(int);
   octx->ol_d_id_len = sizeof(int);
   octx->ol_o_id_len = sizeof(int);

   /* bind variables */

   /* 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);
   OCIDFNRA(octx->curo0,octx->c_rowid_dp,p,1,octx->c_rowid_ptr,
        SIZ(OCIRowid*), SQLT_RDD, NULL, octx->c_rowid_len, NULL);
   OCIBND(octx->curo1,octx->c_rowid_bp,p,":cust_rowid",&octx-
>c_rowid_cust,
    sizeof(octx->c_rowid_ptr[0]),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);
   OCIDEF(octx->curo1,octx->o_rowid_dp1,p->errhp,10,ADR(octx-
>o_rowid),
    SIZ(OCIRowid*),SQLT_RDD);

/* 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->o_rowid_dp2,p->errhp,9,ADR(octx-
>o_rowid),SIZ(OCIRowid*),
           SQLT_RDD);

/* Bind for last cursor - 3 */
   OCIBND (octx->curo3,octx->o_rowid_bp,p,":ordr_rowid",ADR(octx-
>o_rowid),
    SIZ(OCIRowid*),SQLT_RDD);
   OCIDFNRA(octx->curo3,octx->ol_i_id_dp,p,1,otemp-
>loc_ol_i_id,SIZ(int),
           SQLT_INT,NULL,octx->ol_i_id_len,NULL);
   OCIDFNRA(octx->curo3,octx->ol_supply_w_id_dp,p,2,
           otemp->loc_ol_supply_w_id,SIZ(int),SQLT_INT,NULL,
           octx->ol_supply_w_id_len, NULL);
   OCIDFNRA(octx->curo3,octx->ol_quantity_dp,p,3,otemp-
>loc_ol_quantity,
           SIZ(int),SQLT_INT,NULL,octx->ol_quantity_len,NULL);
   OCIDFNRA(octx->curo3,octx->ol_amount_dp,p,4,otemp-
>loc_ol_amount,
      SIZ(int), SQLT_INT,NULL, octx->ol_amount_len, NULL);
   OCIDFNRA(octx->curo3,octx->ol_d_base_dp,p,5,otemp-
>loc_ol_delivery_date,
            SIZ(OCIDate),SQLT_ODT,NULL,octx-
>ol_delivery_d_len,NULL);
   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);
 }

int 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_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_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;
   if(pOrd->byname)
     {
     cbctx->reexec = FALSE;
     execstatus=OCIStmtExecute(p->tpcsvc,octx->curo0,p-
>errhp,100,0,
                NULLP(CONST
OCISnapshot),NULLP(OCISnapshot),OCI_DEFAULT);
     if ((execstatus != OCI_NO_DATA) && (execstatus !=
OCI_SUCCESS))
     /* will get OCI_NO_DATA if <100 found */
     {
        errcode = OCIERROR(p,execstatus);
        TPCCErr("Error in OrderStatus Transaction curo0 errcode:
%d\n",errcode);
        if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
||
            (errcode == SNAPSHOT_TOO_OLD))
        {
            DISCARD 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 */
     DISCARD OCIAttrGet(octx->curo0,OCI_HTYPE_STMT,&rcount,NULL,
                        OCI_ATTR_ROW_COUNT, p->errhp);
     octx->cust_idx=(rcount)/2 ;
     }
     else
     {
     /* count  the number of rows */
     execstatus = OCIStmtExecute(p->tpcsvc,octx->curo4,p-
>errhp,1,0,
                NULLP(CONST
OCISnapshot),NULLP(OCISnapshot),OCI_DEFAULT);
     if ((execstatus != OCI_NO_DATA) && (execstatus !=
OCI_SUCCESS))
     {
      errcode = OCIERROR(p,execstatus);
      TPCCErr("Error in OrderStatus Transaction curo0
errcode:%d\n",errcode);
      if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
         || (errcode == SNAPSHOT_TOO_OLD))
       {
         DISCARD OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
   return RECOVERR;
       } else {
         return ERR_DB_ERROR;
       }
     }
     if (octx->rcount+1 < 2*10)
         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,NULLP(CONST
OCISnapshot),NULLP(OCISnapshot),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);
      TPCCErr("Error in Transaction OrderStatus curo0 errcode:
%d\n",errcode);
      if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
           || (errcode == SNAPSHOT_TOO_OLD))
      {
         DISCARD OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
   return RECOVERR;
       } else {
         return ERR_DB_ERROR;
       }
     }
     octx->cust_idx=0;
   }
  }

  octx->c_rowid_cust=octx->c_rowid_ptr[octx->cust_idx];
  execstatus=OCIStmtExecute(p->tpcsvc,octx->curo1,p->errhp,1,0,
            NULLP(CONST
OCISnapshot),NULLP(OCISnapshot),OCI_DEFAULT);
  if (execstatus != OCI_SUCCESS)
   {
      errcode = OCIERROR(p,execstatus);
      TPCCErr("Error in Transaction OrderStatus curo1
errcode:%d\n",errcode);
      DISCARD 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,
                NULLP(CONST
OCISnapshot),NULLP(OCISnapshot),OCI_DEFAULT);
     if (execstatus != OCI_SUCCESS)
```

```
      {
         errcode = OCIERROR(p,execstatus);
         TPCCErr("Error in Transaction OrderStatus curo2
errcode:%d\n",errcode);
         DISCARD 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_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,
              NULLP(CONST OCISnapshot),NULLP(OCISnapshot),
              OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
   if (execstatus != OCI_SUCCESS)
     {
         errcode = OCIERROR(p,execstatus);
         TPCCErr("Error in Transaction OrderStatus curo3
errcode:%d\n",errcode);
         DISCARD OCITransCommit(p->tpcsvc,p->errhp,OCI_DEFAULT);
         if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
              || (errcode == SNAPSHOT_TOO_OLD))
         {
     return RECOVERR;
         } else {
           return ERR_DB_ERROR;
         }
   }
   /* clean up and convert the delivery dates */
  for (i = 0; i < pOrd->o_ol_cnt; i++) {
     octx->ol_delivery_d_len[i]=sizeof(otemp-
>ol_delivery_date_str[i]);
     DISCARD OCIERROR(p, OCIDateToText(p->errhp,&otemp-
>loc_ol_delivery_date[i],
        (const text*)SHORTDATE,(ub1)strlen(SHORTDATE),(text*)0,0,
        (ub4 *)&octx->ol_delivery_d_len[i],otemp-
>ol_delivery_date_str[i]));
   }
  /* convert the order entry date */
    DISCARD 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,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 SQLTXT "BEGIN inittpcc.init_del; END;"

#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 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 sum(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"




int tkvcdinit (DeliveryData *pDel,
     OraContext *p)
{
```

```
   delctx *dctx = &(p->dctx);
   text stmbuf[SQL_BUF_SIZE];
   DISCARD memset(dctx,(char)0,sizeof(delctx));

   DISCARD OCIHandleAlloc(p->tpcenv, (dvoid **)&dctx->curp1,
OCI_HTYPE_STMT, 0,
        (dvoid **)0);
   DISCARD sprintf ((char *)stmbuf, SQLTXT);
   DISCARD OCIStmtPrepare(dctx->curp1,p->errhp,stmbuf,
       (ub4)strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT);
   DISCARD OCIERROR(p,
          OCIStmtExecute(p->tpcsvc,dctx->curp1,p-
>errhp,1,0,NULLP(OCISnapshot),
      NULLP(OCISnapshot), OCI_DEFAULT));
   DISCARD OCIHandleAlloc(p->tpcenv,(dvoid **)&dctx-
>curp2,OCI_HTYPE_STMT,0,(dvoid**)0);
   if(ERR_DB_ERROR == getfile("tkvcpdel.sql",stmbuf))
    {
      TPCCErr("Error opening the file tkvcpdel.sql");
      return ERR_DB_ERROR;
    }
   DISCARD OCIStmtPrepare(dctx->curp2,p->errhp,stmbuf,
       (ub4)strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT);
   OCIBNDPL(dctx->curp2,dctx->w_id_bp,p,":w_id",ADR(pDel-
>w_id),SIZ(int),SQLT_INT, &dctx->w_id_len);
   OCIBNDPL(dctx->curp2,dctx->ordcnt_bp,p,":ordcnt",ADR(dctx-
>ordcnt),
     SIZ(int),SQLT_INT, &dctx->ordcnt_len);
   OCIBNDPL(dctx->curp2,dctx->del_date_bp,p,":now",
         ADR(dctx->del_date),SIZ(OCIDate),SQLT_ODT, &dctx-
>del_date_len);
   OCIBNDPL(dctx->curp2,dctx->carrier_id_bp,p,":carrier_id",
     ADR(dctx->carrier_id), SIZ(int),SQLT_INT, &dctx-
>carrier_id_len);
   OCIBNDPLA(dctx->curp2, dctx->d_id_bp, p,":d_id",
             dctx->del_d_id, SIZ(int),SQLT_INT, dctx->del_d_id_len,
                      NDISTS, &dctx->del_d_id_rcnt);
   OCIBNDPLA(dctx->curp2, dctx->o_id_bp, p,":order_id",
              dctx->del_o_id,SIZ(int),SQLT_INT, dctx-
>del_o_id_len,NDISTS,
                      &dctx->del_o_id_rcnt);
   OCIBNDPLA(dctx->curp2, dctx->sums_bp, p,"sums",
              dctx->sums,SIZ(int),SQLT_INT, dctx->sums_len,NDISTS,
                      &dctx->sums_rcnt);
   OCIBNDPLA(dctx->curp2, dctx->o_c_id_bp, p,":o_c_id",
              dctx->o_c_id,SIZ(int),SQLT_INT, dctx-
>o_c_id_len,NDISTS,
                   &dctx->o_c_id_rcnt);

    OCIBND (dctx->curp2,dctx->retry_bp,p,":retry",
        ADR(dctx->retry),SIZ(int),SQLT_INT);
   return (ERR_DB_SUCCESS);
}

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

  invalid = 0;

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

   /* initialization for array operations */
   dctx->w_id_len=sizeof(int);
   dctx->carrier_id_len=sizeof(int);
   dctx->carrier_id=pDel->o_carrier_id;
   for (i = 0; i < NDISTS; i++) {
      dctx->del_o_id_len[i]= sizeof(int);
      dctx->del_o_id[i]=0;
   }
   dctx->del_date_len=DEL_DATE_LEN;
   DISCARD memcpy (&dctx->del_date,&dtemp-
>cr_date,sizeof(OCIDate));


   dctx->retry=0;

   execstatus=OCIStmtExecute(p->tpcsvc,dctx->curp2,p->errhp,1,0,
    NULLP(CONST OCISnapshot),NULLP(OCISnapshot),OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS) {
     errcode = OCIERROR(p,execstatus);
     TPCCErr("Error in Delivery Transaction curp2
errcode:%d\n",errcode);
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     errcode = OCIERROR(p,execstatus);
     if((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR) ||
       (errcode == SNAPSHOT_TOO_OLD)) {
        return(RECOVERR);
     } else {
        return ERR_DB_ERROR;
     }
  }
  for(i=0;i<NDISTS;i++)
    {
      pDel->o_id[i]=0;
    }
    for(i=0;i<dctx->del_o_id_rcnt;i++)
      pDel->o_id[dctx->del_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
  DISCARD free(&dctx);
}

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

#define NOSQLTXT2ops "UPDATE stok SET s_order_cnt = s_order_cnt +
1, \
   s_ytd = s_ytd + :ol_quantity, s_remote_cnt = s_remote_cnt +
:s_remote, \
   s_quantity = s_quantity - :ol_quantity + \
   DECODE (SIGN (s_quantity - :ol_quantity - 10), -1, 91, 0) \
   WHERE s_i_id = :ol_i_id AND s_w_id = :ol_supply_w_id"


#define NOSQLTXT2 "BEGIN inittpcc.init_no(:idx1arr); END;"




int tkvcninit (NewOrderData *pNew,
         OraContext *p)
{
  newctx *nctx = &(p->nctx);
  newtemp *ntemp = &(p->tempvars.new);
  int execstatus;
  int errcode;
  text stmbuf[SQL_BUF_SIZE];
  DISCARD memset(nctx,(char)0,sizeof(newctx));
  nctx->cs = 1;
  nctx->norow=0;
  nctx->w_id_len = sizeof(pNew->w_id);
  nctx->d_id_len = sizeof(pNew->d_id);
  nctx->c_id_len = sizeof(pNew->c_id);
  nctx->o_all_local_len = sizeof(pNew->o_all_local);
  nctx->o_ol_cnt_len = sizeof(pNew->o_ol_cnt);
  nctx->w_tax_len = 0;
  nctx->d_tax_len = 0;
  nctx->o_id_len = sizeof(pNew->o_id);
  nctx->c_discount_len = 0;
  nctx->c_credit_len = 0;
  nctx->c_last_len = 0;
  nctx->retries_len = sizeof(ntemp->n_retry);
  nctx->cr_date_len = sizeof(ntemp->cr_date);
  /* open first cursor */
  DISCARD OCIERROR(p,OCIHandleAlloc(p->tpcenv,(dvoid **)(&nctx-
>curn1),
      OCI_HTYPE_STMT, 0, (dvoid**)0));
  if(ERR_DB_ERROR == getfile("tkvcpnew.sql",stmbuf))
  {
    TPCCErr("Error opening the file tkvcpnew.sql");
    return ERR_DB_ERROR;
  }
  DISCARD OCIERROR(p,OCIStmtPrepare(nctx->curn1, p->errhp, stmbuf,
     strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));
  /* bind variables */
  OCIBNDPL(nctx->curn1,nctx->w_id_bp,p,":w_id",ADR(pNew-
>w_id),SIZ(pNew->w_id),
          SQLT_INT, &nctx->w_id_len);
  OCIBNDPL(nctx->curn1,nctx->d_id_bp,p,":d_id",ADR(pNew-
>d_id),SIZ(pNew->d_id),
          SQLT_INT, &nctx->d_id_len);
  OCIBNDPL(nctx->curn1,nctx->c_id_bp,p,":c_id",ADR(pNew-
>c_id),SIZ(pNew->c_id),
          SQLT_INT, &nctx->c_id_len);
  OCIBNDPL(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_len);
  OCIBNDPL(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_len);
  OCIBNDPL(nctx->curn1,nctx->w_tax_bp,p,":w_tax",ADR(ntemp->w_tax),
    SIZ(ntemp->w_tax),SQLT_FLT,&nctx->w_tax_len);
  OCIBNDPL(nctx->curn1,nctx->d_tax_bp,p,":d_tax",ADR(ntemp->d_tax),
    SIZ(ntemp->d_tax),SQLT_FLT,&nctx->d_tax_len);
  OCIBNDPL(nctx->curn1,nctx->o_id_bp,p,":o_id",ADR(pNew-
>o_id),SIZ(pNew->o_id),
          SQLT_INT,&nctx->o_id_len);
  OCIBNDPL(nctx->curn1,nctx->c_discount_bp,p,":c_discount",
          ADR(ntemp->c_discount),SIZ(ntemp->c_discount),SQLT_FLT,
          &nctx->c_discount_len);
  OCIBNDPL(nctx->curn1,nctx->c_credit_bp,p,":c_credit",pNew-
>c_credit,
          SIZ(pNew->c_credit),SQLT_CHR,&nctx->c_credit_len);
  OCIBNDPL(nctx->curn1,nctx->c_last_bp,p,":c_last",pNew->c_last,
          SIZ(pNew->c_last),SQLT_STR,&nctx->c_last_len);
  OCIBNDPL(nctx->curn1, nctx->retries_bp, p, ":retry",ADR(ntemp-
>n_retry),
          SIZ(ntemp->n_retry),SQLT_INT, &nctx->retries_len);
  OCIBNDPL(nctx->curn1,nctx->cr_date_bp,p,":cr_date",ADR(ntemp-
>cr_date),
          SIZ(ntemp->cr_date),SQLT_ODT,&nctx->cr_date_len);
  OCIBNDPLA(nctx->curn1,nctx->ol_i_id_bp,p,":ol_i_id",ntemp-
>nol_i_id,
          SIZ(int),SQLT_INT,nctx->nol_i_id_len,NITEMS,&nctx-
>nol_i_count);
  OCIBNDPLA(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_len,
    NITEMS,&nctx->nol_s_count);
  OCIBNDPLA(nctx->curn1,nctx->ol_quantity_bp,p,":ol_quantity",
    ntemp->nol_quantity,SIZ(int),SQLT_INT,nctx->nol_quantity_len,
          NITEMS,&nctx->nol_q_count);
  OCIBNDPLA(nctx->curn1,nctx->i_price_bp,p,":i_price",ntemp-
>i_price,
          SIZ(int),SQLT_INT,nctx->i_price_len,NITEMS,&nctx-
>nol_item_count);
  OCIBNDPLA(nctx->curn1,nctx->i_name_bp,p,":i_name",ntemp->i_name,
          SIZ(pNew->o_ol[0].i_name),SQLT_STR,nctx-
>i_name_len,NITEMS,
    &nctx->nol_name_count);
  OCIBNDPLA(nctx->curn1,nctx->s_quantity_bp,p,":s_quantity",ntemp-
>s_quantity,
          SIZ(int),SQLT_INT,nctx->s_quant_len,NITEMS,&nctx-
>nol_qty_count);
  OCIBNDPLA(nctx->curn1,nctx->s_bg_bp,p,":brand_generic",ntemp-
>brand_generic,
          SIZ(char),SQLT_CHR,nctx->s_bg_len,NITEMS,&nctx-
>nol_bg_count);
  OCIBNDPLA(nctx->curn1,nctx->ol_amount_bp,p,":ol_amount",ntemp-
>nol_amount,
          SIZ(int), SQLT_INT,nctx->nol_amount_len,NITEMS,&nctx-
>nol_am_count);
  OCIBNDPLA(nctx->curn1,nctx->s_remote_bp,p,":s_remote",nctx-
>s_remote,
          SIZ(int),SQLT_INT,nctx->s_remote_len,NITEMS,&nctx-
>s_remote_count);

  /* open second cursor */
  DISCARD OCIERROR(p,OCIHandleAlloc(p->tpcenv, (dvoid **)(&nctx-
>curn2),
           OCI_HTYPE_STMT, 0, (dvoid**)0));
  DISCARD sprintf ((char *) stmbuf, NOSQLTXT2);
  DISCARD 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];
  ub4 idx1arr_count;
  ub2 idx;
  for (idx=0;idx<NITEMS;idx++)
  {
   idx1arr[idx] = idx + 1;
   idx1arr_len[idx] = sizeof(int);
  }
  idx1arr_count=NITEMS;
  pNew->o_ol_cnt=NITEMS;

/* Bind array */
  OCIBNDPLA(nctx-
>curn2,idx1arr_bp,p,":idx1arr",idx1arr,SIZ(int),SQLT_INT,
    idx1arr_len,NITEMS,&idx1arr_count);
  execstatus = OCIStmtExecute(p->tpcsvc,nctx->curn2,p->errhp,1,0,
             NULLP(CONST
OCISnapshot),NULLP(OCISnapshot),OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS)
  {
   DISCARD 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 retries = 0;
  int i;
  int rcount;
  statusCnt = 0;                        /* number of invalid items
*/
  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_len = sizeof(pNew->w_id);
   nctx->d_id_len = sizeof(pNew->d_id);
   nctx->c_id_len = sizeof(pNew->c_id);
   nctx->o_all_local_len = sizeof(pNew->o_all_local);
   nctx->o_ol_cnt_len = sizeof(pNew->o_ol_cnt);
   nctx->w_tax_len = 0;
   nctx->d_tax_len = 0;
   nctx->o_id_len = sizeof(pNew->o_id);
   nctx->c_discount_len = 0;
   nctx->c_credit_len = 0;
   nctx->c_last_len = 0;
   nctx->retries_len = sizeof(retries);
   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;

   /* initialization for array operations */
   for (i = 0; i < pNew->o_ol_cnt; i++) {
      nctx->ol_number[i] = i + 1;
      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_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->s_remote_len[i] = sizeof(int);
      nctx->s_quant_len[i] = sizeof(int);
      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_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_o_id_len[i] = 0;
      nctx->ol_number_len[i] = 0;
      nctx->ol_dist_info_len[i] = 0;
      nctx->s_remote_len[i] = 0;
      nctx->s_quant_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);
   /* did the txn succeed? */
   /* sth added return of ERR_DB_NOT_COMMITED for Invalid Item */
   if (rcount != pNew->o_ol_cnt)
   {
    statusCnt = rcount - pNew->o_ol_cnt;
    pNew->o_ol_cnt = rcount;
    return (ERR_DB_NOT_COMMITED);
   }
   if(execstatus != OCI_SUCCESS) {
     OCITransRollback(p->tpcsvc,p->errhp,OCI_DEFAULT);
     errcode = OCIERROR(p,execstatus);
     TPCCErr ("Error in NewOrder Transaction curn1
errcode:%d\n",errcode);
     if((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR) ||
       (errcode == SNAPSHOT_TOO_OLD)) {
  retries++;
  return (RECOVERR);
     }
     else
     {
  return (ERR_DB_ERROR);
     }
   }

/* calculate total amount */
  pNew->total_amount = 0.0;
  for (i=0;i<pNew->o_ol_cnt;i++)
  {
     pNew->total_amount += ntemp->nol_amount[i];
  }
  pNew->total_amount *= ((double)(1-ntemp->c_discount)) *
(double)(1.0 + ((double)(ntemp->d_tax))+((double)(ntemp->w_tax)));
  pNew->total_amount = pNew->total_amount/100;
  return (ERR_DB_SUCCESS);
}

void tkvcndone (newctx *pnctx)
{
   newctx nctx = *pnctx;
   if(NULL != nctx.curn1)
     DISCARD OCIHandleFree((dvoid *)nctx.curn1,OCI_HTYPE_STMT);
   if(NULL != nctx.curn2)
     DISCARD OCIHandleFree((dvoid *)nctx.curn2,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 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.
*

 *
*
 *
*

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

/*
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                      - Conversion to run under Linux and Apache
 *
 */

#include <stdio.h>
#include <stdarg.h>
#include <malloc.h>
#include <stdlib.h>
```

```
#include <string.h>
#include <time.h>

#include "apr_thread_mutex.h"

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

#define TPCC_C

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

#include <tpcc.h>
#include <mod_tpcc.h>

#define _strupr(x)      { \
                        int strupr_pos; \
                        for (strupr_pos=0; strupr_pos <
strlen(x);strupr_pos++) \
                          x[strupr_pos] = toupper(x[strupr_pos]); \
                        }

/* 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<15; 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 = 0;
    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;
    if (pOrderStatusData->c_id==0)
      return ERR_ORDERSTATUS_CID_INVALID;
  }

  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 = 0;
    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(pPaymentData->c_id==0)
      return ERR_PAYMENT_CUSTOMER_INVALID;
  }

  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;
}


/* FUNCTION: BOOL ReadRegistrySettings(void)
 *
 * PURPOSE: This function reads the Linux TPCC configuration file
for
 *    startup parameters.
 *
 * ARGUMENTS:   None
 *
 * RETURNS: None
 *
 * COMMENTS:  This function also sets up required operation
variables to
 *    their default value so if registry is not setup the default
 *    values will be used.
 *
 */

int ReadRegistrySettings(void)
{
  char  szTmp[FILENAMESIZE];
  int   status;
  int iTmp;

  status = GetConfigValue("PATH", (char *)&szTmp);
  if ( status != ERROR_SUCCESS )
    return ERR_CANT_FIND_PATH_VALUE;
  strcpy(szTpccLogPath, szTmp);

  status = GetConfigValue("Server", (char *)&szTmp);
  if ( status != ERROR_SUCCESS )
    /* required */
    return ERR_CANT_FIND_SERVER_VALUE;
  strcpy(gszServer, szTmp);

  status = GetConfigValue("Database", (char *)&szTmp);
  if ( status != ERROR_SUCCESS )
    /* required */
    return ERR_CANT_FIND_DATABASE_VALUE;
  strcpy(gszDatabase, szTmp);

  status = GetConfigValue("User", (char *)&szTmp);
  if ( status != ERROR_SUCCESS )
    /* required */
    return ERR_CANT_FIND_USER_VALUE;
  strcpy(gszUser, szTmp);

  status = GetConfigValue("Password", (char *)&szTmp);
  if ( status != ERROR_SUCCESS )
    /* required */
    return ERR_CANT_FIND_PASSWORD_VALUE;
  strcpy(gszPassword, szTmp);

  status = GetConfigValue("LOG", (char *)&szTmp);
  if ( status == ERROR_SUCCESS  && 0 == strcmp(szTmp, "ON") )
    bLog = TRUE;

  status = GetConfigValue("MaxConnections", (char *)&szTmp);
  if ( status == ERROR_SUCCESS && 0 != (iTmp = atoi(szTmp)) )
    iMaxConnections = iTmp;

  status = GetConfigValue("NumDeliveryServers", (char *)&szTmp);
  if ( status == ERROR_SUCCESS && 0 != (iTmp = atoi(szTmp)) )
    iDeliveryServers = iTmp;

  return ERR_SUCCESS;

}



-------------------------------------------------
      tpcc.h
-------------------------------------------------

#ifndef TPCC_H
#define TPCC_H

/*+*****************************************************************
*********
 *
*
 *  COPYRIGHT (c) 1997 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
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                        Conversion to run under Linux and Apache
 *
*/

#define ERROR_SUCCESS 1
#define FILENAMESIZE 256
#define DEBUG 0
#define MAXPAD 6

#define itoa(x,y)       sprintf(y, "%d", x)

#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 */

BOOL ReadRegistrySettings(void);

/* global variables */
#ifdef MOD_TPCC_C
#define GLOBAL(thing,initializer) thing = initializer
#else
#define GLOBAL(thing,initializer) extern thing
#endif /* TPCC_C */

GLOBAL(int iMaxConnections,25);
GLOBAL(BOOL bLog,FALSE);
GLOBAL(int iDeadlockRetry,3);
GLOBAL(char szTpccLogPath[FILENAMESIZE],{'\0'});
GLOBAL(int iMaxWareHouses,500);
GLOBAL(char gszServer[32],{'\0'});
GLOBAL(char gszDatabase[32],"tpcc");
GLOBAL(char gszUser[32],"oracle");
GLOBAL(char gszPassword[32],{'\0'});
GLOBAL(pTransactionPoolStruct gpTransactionPool,{0});
GLOBAL(FILE *MyLogFile, {0});
GLOBAL(int iDeliveryServers,1);

#endif /* TPCC_H */


-------------------------------------------------
      tpccapi.h
-------------------------------------------------

#ifndef TPCCAPI_H
#define TPCCAPI_H
/*+*****************************************************************
*********
 *
*
 *  COPYRIGHT (c) 1996 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
**
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                        Conversion to run under Linux and Apache
 *
*/


#define DELIVERY_RESPONSE_COUNT 2

int TPCCGetTransportData( pTransportData pTransport );

int TPCCStartup( );
int TPCCStartupDB( );

int TPCCConnect( pLoginData pLogin );
int TPCCConnectDB(OraContext  **dbproc, pLoginData pLogin );

int TPCCDelivery( pDeliveryData pDelivery);
int TPCCDeliveryDeferred( pDeliveryData ppDelivery );
int TPCCDeliveryDB( OraContext *dbproc, pDeliveryData pDelivery );

int TPCCNewOrder( pNewOrderData pNewOrder );
int TPCCNewOrderDB( OraContext *dbproc, pNewOrderData pNewOrder );

int TPCCOrderStatus( pOrderStatusData pOrderStatus );
int TPCCOrderStatusDB( OraContext *dbproc, pOrderStatusData
pOrderStatus );

int TPCCPayment( pPaymentData pPayment );
int TPCCPaymentDB( OraContext *dbproc, pPaymentData pPayment );

int TPCCStockLevel( pStockLevelData pStockLevel );
int TPCCStockLevelDB( OraContext *dbproc, pStockLevelData
pStockLevel );

int TPCCCheckpoint( pCheckpointData pCheckpoint );
int TPCCCheckpointDB( OraContext *dbproc, pCheckpointData
pCheckpoint );

int TPCCDisconnect( pCallersContext pCC );
int TPCCDisconnectDB( OraContext *dbproc, pCallersContext pCC );

int TPCCShutdown( void );
int TPCCShutdownDB( void );

void TPCCDeliveryResponse( int retcode, pDeliveryData pDelivery,
       pDeliveryData CompletedDeliveries[DELIVERY_RESPONSE_COUNT]
);

void TPCCDeliveryDeferredResponse( int retcode, pDeliveryData
pDelivery );

void TPCCNewOrderResponse( int retcode, pNewOrderData pNewOrder );

void TPCCOrderStatusResponse( int retcode, pOrderStatusData
pOrderStatus );

void TPCCPaymentResponse( int retcode, pPaymentData pPayment );

void TPCCStockLevelResponse( int retcode, pStockLevelData
pStockLevel );

void TPCCResponseComplete( CallersContext *pCC );

void ErrorMessage( CallersContext *pCC, int iError, int iErrorType,
       char *pszMesasge );

int TPCCGetTransportErrorString( int iErrorCode, int iBufSize, char
*pBuffer );
int TPCCGetDBErrorString( int iErrorCode, int iBufSize, char
*pBuffer );

BOOL TPCCOpenLog( apr_pool_t *pool );

BOOL TPCCCloseLog( void );

void TPCCLog( char *fmt, ... );

void TPCCErr( char *fmt, ... );

void TPCCTransactionErr( pConnData pConn, char *fmt, ... );

int GetConfigValue(char *option, char *value);

#endif /* TPCCAPI_H */
```

```
-------------------------------------------------
      tpccerr.h
-------------------------------------------------

#ifndef TPCCERR_H
#define TPCCERR_H

/*  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
 *
 * Modification history:
 *
 *
 *
*/

/*#pragma message ("FIXME: the error types need to be made DB non-
specific") */
#define ERR_TYPE_WEBDLL             1
#define ERR_TYPE_SQL              2
#define ERR_TYPE_DBLIB              3

#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
/* NOTE: Be sure to update MAX_ERR if new error code is added. */
#define ERR_DB_MAX_ERR          11

#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_DISCONNECT    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_PAYMENT_MISSING_CID_CLT   1115
#define ERR_CANT_FIND_MAXDBCONNECTIONS_VALUE    1116

/* 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 TPCC_C
SERRORMSG errorMsgs[] =
{
  { 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_DISCONNECT, "Failed to start delivery pipe
disconnect thread."},
  { 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, "Synchronous transport requires
delivery server connect to database." },
  { 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_CANT_FIND_MAXDBCONNECTIONS_VALUE, "MaxDBConnections value
not set in TPCC key." },
  { 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." },
  { 0, "" }
};
#else
extern SERRORMSG errorMsgs[];
#endif /* TPCC_C */

#endif /* TPCCERR_H */


-------------------------------------------------
```

```
      tpccstruct.h
-------------------------------------------------

#ifndef TPCCSTRUCT_H
#define TPCCSTRUCT_H

#include "apr_thread_mutex.h"

/*****************************************************************
**********
******************************  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)
**
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                      - Conversion to run under Linux and Apache
 *
 */

#include <time.h>

/*
#include <sys/types.h>
*/

#define BOOLEAN int
#define BOOL int
#define VMS     0
#define LINEMAX 256
#define FALSE   0
#ifndef TRUE
#define TRUE    1
#endif

#define MAX_OL  15


#ifdef FFE_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 ALL_STAGES 0x01ff

  /*                           users * scale * hours * min * txn/no
*/
# define HISTORY_SIZE ((int)( 5000 *   1.2 *     2 *  60 *
2.22222))

# define TRANSACTION_DEBUG_INFO\
  int iStage;\
  int dwThreadId;\
  int dwXPThreadId;\
  int iSynchronous;\
  int iType;\
  int iReserveHistoryId;\
  int iUnreserveHistoryId;\

# define INIT_TRANSACTION(type,pData)\
  gpTransactionPool->iHistoryId++;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iFailure = 0;\
  _ASSERT( gpTransactionPool->iNextFree <= gpTransactionPool-
>iMaxIndex );\
  memset( pData, 0x01, gpTransactionPool->iTransactionSize );\
  pData->iStage = 0;\
  pData->dwThreadId = GetCurrentThreadId();\
  pData->dwXPThreadId = 0;\
  pData->iType = type;\
  pData->iReserveHistoryId = gpTransactionPool->iHistoryId;\
  pData->iUnreserveHistoryId = 0;\
  gpTransactionPool->History[gpTransactionPool->iHistoryId].iOpCode
= 1;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iReserveHistoryId = gpTransactionPool->iHistoryId;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iUnreserveHistoryId = 0;\
  gpTransactionPool->History[gpTransactionPool->iHistoryId].iType =
type;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].dwThreadId = pData->dwThreadId;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].dwXPThreadId = pData->dwXPThreadId;\
  gpTransactionPool->History[gpTransactionPool->iHistoryId].pTrans
= pData;

# define CHECK_TRANSACTION(type,pData)\
  gpTransactionPool->iHistoryId++;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iFailure++;\
  _ASSERT( gpTransactionPool->iNextFree > 0 );\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iFailure++;\
  _ASSERT(((pData->iStage) | ALL_STAGES) == ALL_STAGES);\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iFailure++;\
  if( pData->iSynchronous == 1 )\
    _ASSERT((pData->dwThreadId == GetCurrentThreadId( )));\
  else if( pData->iSynchronous == 0 )\
    _ASSERT((pData->dwXPThreadId == GetCurrentThreadId( )));\
  else\
    _ASSERT(FALSE);\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iFailure++;\
  _ASSERT((pData->iType==type));\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iFailure++;\
  _ASSERT((gpTransactionPool->History[pData-
>iReserveHistoryId].pTrans) == pData);\
  pData->iUnreserveHistoryId = gpTransactionPool->iHistoryId;\
  gpTransactionPool->History[gpTransactionPool->iHistoryId].iOpCode
= 2;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iReserveHistoryId = pData->iReserveHistoryId;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].iUnreserveHistoryId = gpTransactionPool->iHistoryId;\
  gpTransactionPool->History[gpTransactionPool->iHistoryId].iType =
type;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].dwThreadId = pData->dwThreadId;\
  gpTransactionPool->History[gpTransactionPool-
>iHistoryId].dwXPThreadId = pData->dwXPThreadId;\
  gpTransactionPool->History[gpTransactionPool->iHistoryId].pTrans
= pData;

#else /* FFE_DEBUG */

# define TRANSACTION_DEBUG_INFO

# define INIT_TRANSACTION(type,pData)

# define CHECK_TRANSACTION(type,pData)

#endif /* FFE_DEBUG */

# define NUMBER_POOL_TRANS_TYPES 5
# define DELIVERY_TRANS 0
# define NEW_ORDER_TRANS 1
# define ORDER_STATUS_TRANS 2
# define PAYMENT_TRANS 3
# define STOCK_LEVEL_TRANS 4

#define RESERVE_TRANSACTION_STRUCT(type,pData)\
  apr_thread_mutex_lock( gpTransactionPool->critSec );\
  pData = gpTransactionPool->index[gpTransactionPool->iNextFree];\
  INIT_TRANSACTION(type,pData);\
  gpTransactionPool->iNextFree++;\
  apr_thread_mutex_unlock( gpTransactionPool->critSec );

#define UNRESERVE_TRANSACTION_STRUCT(type,pData)\
  apr_thread_mutex_lock( gpTransactionPool->critSec );\
  CHECK_TRANSACTION(type,pData);\
  gpTransactionPool->index[--gpTransactionPool->iNextFree] =
pData;\
  apr_thread_mutex_unlock( gpTransactionPool->critSec );

typedef struct
{
  apr_thread_mutex_t * critSec;
  int iNextFree;
#ifdef FFE_DEBUG
  int iMaxIndex;
  int iTransactionSize;
  int iHistoryId;
  struct
  {
    int   iOpCode;
    int   iFailure;
    int   iReserveHistoryId;
    int   iUnreserveHistoryId;
    int   iType;
    int dwThreadId;
    int dwXPThreadId;
    void  *pTrans;
  }   History[HISTORY_SIZE];
#endif
  void  *index[1];
  char  data[1];
} TransactionPoolStruct, *pTransactionPoolStruct;
```

```
/*
**  Data structures descriptions for IO data for each transaction
type
**
*/

typedef void CallersContext;
typedef void *pCallersContext;
typedef void *DBContext;

#define INVALID_DB_CONTEXT NULL

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 \
    TRANSACTION_DEBUG_INFO\
    int     w_id;\
    int     ld_id;\
    CallersContext *pCC;\
    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     delta_time;        /* in milliseconds */\
    struct timeval  tbegin;\
    struct timeval  tend;\
    int         o_carrier_id;

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, *pNewOrderData;

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, *pOrderStatusData;

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, *pPaymentData;

typedef struct _StockLevelData {
    CONN_DATA
    int     threshold;
    int     low_stock;
} StockLevelData, *pStockLevelData;

typedef struct _CheckpointData {
    CONN_DATA
    int     how_many;
    int     interval;
} CheckpointData, *pCheckpointData;

/*
**  Data structure for input & output data
*/

typedef struct _TransactionData {
    int type;
    union {
  DeliveryData delivery;
  NewOrderData newOrder;
  OrderStatusData orderStatus;
  PaymentData payment;
  StockLevelData stockLevel;
        CheckpointData checkpoint;
    }  info;
} TransactionData, *pTransactionData;


typedef struct _TransportData {
  BOOLEAN asynchronous;
  BOOLEAN generic;
  int   num_gc;
  int   num_dy;
  int   num_no;
  int   num_os;
  int   num_pt;
  int   num_sl;
  BOOLEAN dy_use_transport;
  int   num_dy_servers;
  int   num_queued_deliveries;
  int   num_queued_responses;
} TransportData, *pTransportData;

/* Data structure for passing connection information */
typedef struct _LoginData {
  CONN_DATA
  char      szServer[32];
  char      szDatabase[32];
  char      szUser[32];
  char      szPassword[32];
  char      szApplication[32];
} LoginData, *pLoginData;

#endif /* TPCCSTRUCT_H */


-------------------------------------------------
      tux_cli.c
-------------------------------------------------

/*+***************************************************************
*********
 *
*
 *  COPYRIGHT (c) 1997 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.
*
 *
*
 *
*
 *  Updated November 20, 2001 - Susan Georgson
*
 *  Converted tpcc_fct.c file to tux_cli.c
*
 *  Changed transaction monitor from DB Web Connector to Tuxedo
*

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

/*
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                      - Conversion to run under Linux
 *
 */

#include <stdlib.h>   /* stg - added for change to Tuxedo */
#include <string.h>
#include <stdio.h>
#include <sys/time.h>

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

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

#include <tpcc.h>

#include <pthread.h>

/* tuxedo include files */
#include <atmi.h>

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

#define TOTAL_ADMIN_CONNECTIONS 1

#define FILENAMESIZE 256

static pthread_key_t initkey;

static pthread_once_t initkey_once = PTHREAD_ONCE_INIT;

static void doinit(void)
{
  pthread_key_create(&initkey, NULL);
}

/* Returns non-zero if thread has been initialized already. */
static int IsInited(void)
{
  void *p;
  pthread_once(&initkey_once, doinit);
  p = pthread_getspecific(initkey);
  return (p == NULL);
}

static void NowInited(void)
{
  pthread_setspecific(initkey, (void *)1); /* non-NULL value. */
}

/* stg - IsTuxInit is added to check if Tuxedo has been initialized
*/
/* If Tuxedo has not been initialized, then Tuxedo is initialized
during */
/* this function.  */
/*
 * FUNCTION int IsTuxInit
 */
int
IsTuxInit()
{
  TPINIT *tpinitbuf;

  int retcode = -1;
  int count = 0;
  static int num_tpinits = 0;
  TPCONTEXT_T mycontext;
  char  myenv[255];

#if (DEBUG == 1)
        fprintf(MyLogFile, "Entering IsTuxInit\n");
  fflush(MyLogFile);
#endif
  if(IsInited())
    {
      while(count < 20)
      {
    if(NULL == (tpinitbuf = (TPINIT *) tpalloc("TPINIT", NULL,
                       sizeof(TPINIT))))
    {
      TPCCErr("error with tpalloc - %d - %d", tperrno,count);
    }
    else
    {

#if (DEBUG == 1)
/*
        tpgetctxt(&mycontext,0);
        fprintf(MyLogFile, "tpgetctxt before=%d\n", mycontext);
  tpsetctxt(TPNULLCONTEXT,0);
*/
        tpgetctxt(&mycontext,0);
        fprintf(MyLogFile, "before tpinit, pid=%d, mycontext=%d\n",
getpid(),mycontext);
/*
  if (tuxgetenv("NLSPATH") != NULL) {
    fprintf(MyLogFile, "NLSPATH=%s\n", myenv);
  }
  else
    fprintf(MyLogFile, "NLSPATH=NULL\n");
*/
#endif
      tpinitbuf->flags |= TPMULTICONTEXTS;
      itoa(++num_tpinits, tpinitbuf->cltname);
      retcode = tpinit(tpinitbuf);

            tpgetctxt(&mycontext,0);
            fprintf(MyLogFile, "Back from tpinit, pid=%d,
cltname=%s, retcode=%d, context=%d\n", getpid(),tpinitbuf->cltname,
retcode, mycontext);
      fflush(MyLogFile);

      if(-1 != retcode)
      {
        NowInited();
        tpfree((char*)tpinitbuf);
        break;
      }
      else
      {
        TPCCErr("error with TPINIT - %s (%d) - %d\n\t\t..%s..",
          tpstrerror(tperrno),
          tperrno,
          count,
          tpstrerrordetail( tperrordetail( 0 ), 0 ));
        tpfree((char*)tpinitbuf);
      }
    }

    count++;
    if(count > 50)
      {
        retcode = -1;
        TPCCErr("exceeded 50 trys in TPINIT");
      }


    sleep(10);
      }
/*
      sleep(50);
*/
      if( -1 != retcode)
  return ERR_DB_SUCCESS;
      else
  return(retcode);

    }
  return ERR_DB_SUCCESS;
}

/* stg - end IsTuxInit function */


/* 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)
{
  int ii;

  for( ii = 0; errorMsgs[ii].szMsg[0]; ii++ ) {
    if ( iError == errorMsgs[ii].iError ) {
      TPCCErr( "*Error(%d): %s\r\n", iError, errorMsgs[ii].szMsg );
      return;
    }
  }

  TPCCErr( "*Error(%d): Unknown Error.\r\n", iError );
  return;
}


int TPCCDelivery( pDeliveryData pDelivery)
{
  int                           retcode;
  struct timezone   tz;

  time( &pDelivery->queue_time );

  gettimeofday(&pDelivery->tbegin, &tz);

  retcode = TPCCDeliveryDeferred(pDelivery);

  if ( ERR_DB_PENDING != retcode )
  {
    if( ERR_DB_SUCCESS != retcode)
    {
      /* send a flag to the reducer to mark an error on the
delivery */
      pDelivery->queue_time = 1;
      DELIErrorMessage(retcode);
    }

  }

  return ERR_DB_SUCCESS;
}


/* stg - begin Tuxedo change of TPCCDelivery Deferred */
/*
 * FUNCTION int TPCCDelivery
 */

int
TPCCDeliveryDeferred( pDeliveryData ppDelivery )
{
  int retcode = ERR_DB_SUCCESS;

  pDeliveryData retptr;
  int dysiz = sizeof(DeliveryData);
  int ds;
  char svcname[100];
#if (DEBUG == 1)
        fprintf(MyLogFile, "Entering TPCCDeliveryDeferred\n");
  fflush(MyLogFile);
#endif

  /* check to see that the database is connected. */
  if( ERR_DB_SUCCESS != IsTuxInit() )
    {
      TPCCErr("IsTuxInit - delivery ");
      return ERR_DB_ERROR;
    }

  /* allocate memory and copy over data */
  if(NULL == ( retptr= (pDeliveryData) tpalloc("CARRAY", NULL,
dysiz)))
    {
      TPCCErr("tp alloc in delivery");
      return ERR_DB_ERROR;
    }
  memcpy( retptr, ppDelivery, dysiz);

  /* Call tuxedo for Delivery */

  ds=ppDelivery->w_id;
  ds=(ds % iDeliveryServers)+1;
  sprintf(svcname, "dy_transaction%d", ds);

  retcode = tpacall(svcname, (char
*)retptr,dysiz,TPNOREPLY|TPSIGRSTRT|TPNOTIME);
  if( -1 == retcode )
    {
      TPCCErr("tpcall - delivery: %d", tperrno);
      tpfree((char*) retptr);
      return ERR_DB_ERROR;
    }
/*
  memcpy(ppDelivery, retptr, dysiz);
*/
  tpfree((char*) retptr);
  return ERR_DB_SUCCESS;
}

/* stg - end Tuxedo change of TPCCDelivery Deferred */



/* stg - begin Tuxedo change of TPCCNewOrder */
/*
 * FUNCTION int TPCCNewOrder
 */
int
TPCCNewOrder( pNewOrderData ppNewOrder )
{
  int retcode = ERR_DB_SUCCESS;

  pNewOrderData retptr;
  int nosiz = sizeof(NewOrderData);

#if (DEBUG == 1)
        fprintf(MyLogFile, "Entering TPCCNewOrder\n");
  fflush(MyLogFile);
#endif

  /* check to see that the database is connected. */
  if( ERR_DB_SUCCESS != IsTuxInit() )
    {
      TPCCErr("IsTuxInit - new order: %d ", tperrno);
      return ERR_DB_ERROR;
    }

  /* allocate memory and copy over data */
  if(NULL == ( retptr= (pNewOrderData) tpalloc("CARRAY", NULL,
nosiz)))
    {
      TPCCErr("tp alloc in neworder: %d ", tperrno);
      return ERR_DB_ERROR;
    }
  memcpy( retptr, ppNewOrder, nosiz);

  /* Call tuxedo for New Order */
  retcode = tpcall("no_transaction", (char *)retptr, nosiz,
       (char**)&retptr, (long *)&nosiz, TPSIGRSTRT|TPNOTIME);

  if( -1 == retcode )
    {
      TPCCErr("tpcall - new order: %d", tperrno);
      tpfree((char*) retptr);
      return ERR_DB_ERROR;
    }
  memcpy(ppNewOrder, retptr, nosiz);
  tpfree((char*) retptr);
  return ERR_DB_SUCCESS;
}

/* stg - end Tuxedo change of TPCCNewOrder */



/* stg - begin Tuxedo change of TPCCOrderStatus */
/*
 * FUNCTION int TPCCOrderStatus
 */
int
TPCCOrderStatus( pOrderStatusData ppOrderStatus )
{
  int retcode = ERR_DB_SUCCESS;

  pOrderStatusData retptr;
  long ossiz = sizeof(OrderStatusData);

#if (DEBUG == 1)
        fprintf(MyLogFile, "Entering TPCCOrderStatus\n");
  fflush(MyLogFile);
#endif

  /* check to see that the database is connected. */
  if( ERR_DB_SUCCESS != IsTuxInit() )
    {
      TPCCErr("IsTuxInit - order status");
      return ERR_DB_ERROR;
    }

  /* allocate memory and copy over data */
  if(NULL == ( retptr= (pOrderStatusData) tpalloc("CARRAY", NULL,
ossiz)))
    {
      TPCCErr("tp alloc in order status: %d", tperrno);
      return ERR_DB_ERROR;
    }
  memcpy( retptr, ppOrderStatus, ossiz);

  /* Call tuxedo for Order Status */
  retcode = tpcall("os_transaction", (char *)retptr, ossiz,
       (char**)&retptr, (long *)&ossiz, TPSIGRSTRT|TPNOTIME);
#if (DEBUG == 1)
        fprintf(MyLogFile, "TPCCOrderStatus:tpcall returned $d\n",
retcode);
  fflush(MyLogFile);
#endif
  if( -1 == retcode )
    {
      TPCCErr("tpcall - order status");
      tpfree((char*) retptr);
      return ERR_DB_ERROR;
    }
  memcpy(ppOrderStatus, retptr, ossiz);
  tpfree((char*) retptr);
  return ERR_DB_SUCCESS;
}

/* stg - end Tuxedo change of TPCCOrderStatus */


/* stg - begin Tuxedo change of TPCCPayment */
/*
 * FUNCTION int TPCCPayment
 */
int
TPCCPayment( pPaymentData ppPayment )
{
  int retcode = ERR_DB_SUCCESS;

  pPaymentData retptr;
  long ptsiz = sizeof(PaymentData);

#if (DEBUG == 1)
        fprintf(MyLogFile, "Entering TPCCPayment\n");
  fflush(MyLogFile);
#endif

  /* check to see that the database is connected. */
  if( ERR_DB_SUCCESS != IsTuxInit() )
    {
      TPCCErr("IsTuxInit - payment ");
      return ERR_DB_ERROR;
    }

  /* allocate memory and copy over data */
  if(NULL == ( retptr= (pPaymentData) tpalloc("CARRAY", NULL,
ptsiz)))
    {
      TPCCErr("tp alloc in payment");
      return ERR_DB_ERROR;
    }
  memcpy( retptr, ppPayment, ptsiz);

  /* Call tuxedo for Payment */
  retcode = tpcall("pt_transaction", (char *)retptr, ptsiz,
       (char**)&retptr, &ptsiz, TPSIGRSTRT|TPNOTIME);
  if( -1 == retcode )
    {
      TPCCErr("tpcall - payment: %d ", tperrno);
      tpfree((char*) retptr);
      return ERR_DB_ERROR;
    }
  memcpy(ppPayment, retptr, ptsiz);
  tpfree((char*) retptr);
  return ERR_DB_SUCCESS;
}

/* stg - end Tuxedo change of TPCCPayment */
```

```
/* stg - begin Tuxedo change of TPCCStockLevel */
/*
 * FUNCTION int TPCCStockLevel
 */
int
TPCCStockLevel( pStockLevelData ppStockLevel )
{
  int retcode = ERR_DB_SUCCESS;

  pStockLevelData retptr;
  int slsiz = sizeof(StockLevelData);


#if (DEBUG == 1)
        fprintf(MyLogFile, "Entering TPCCStockLevel\n");
  fflush(MyLogFile);
#endif
  /* check to see that the database is connected. */
  if( ERR_DB_SUCCESS != IsTuxInit() )
    {
      TPCCErr("IsTuxInit - stock level ");
      return ERR_DB_ERROR;
    }

  /* allocate memory and copy over data */
  if(NULL == ( retptr= (pStockLevelData) tpalloc("CARRAY", NULL,
slsiz)))
    {
      TPCCErr("tp alloc in stock level");
      return ERR_DB_ERROR;
    }
  memcpy( retptr, ppStockLevel, slsiz);

  /* Call tuxedo for Stock Level */
  retcode = tpcall("sl_transaction", (char *)retptr, slsiz,
       (char**)&retptr, (long *)&slsiz, TPSIGRSTRT|TPNOTIME);
  if( -1 == retcode )
    {
      TPCCErr("tpcall - stock level: %d", tperrno);
      tpfree((char*) retptr);
      return ERR_DB_ERROR;
    }
  memcpy(ppStockLevel, retptr, slsiz);
  tpfree((char*) retptr);
  return ERR_DB_SUCCESS;
}

/* stg - end Tuxedo change of TPCCStockLevel */


/*
**++
**  FUNCTION NAME: TPCCStartup
**--
*/
int
TPCCStartup()
{
  return ERR_SUCCESS;
}


/*
**++
**  FUNCTION NAME: TPCCConnect
**--
*/
int
TPCCConnect( pLoginData pLogin )
{

  if( 0 != strcmp( pLogin->szServer, gszServer ))
    return ERR_SERVER_MISMATCH;

  if( 0 != strcmp( pLogin->szDatabase, gszDatabase ))
    return ERR_DATABASE_MISMATCH;

  if( 0 != strcmp( pLogin->szUser, gszUser ))
    return ERR_USER_MISMATCH;

  if( 0 != strcmp( pLogin->szPassword, gszPassword ))
    return ERR_PASSWORD_MISMATCH;

  return ERR_DB_SUCCESS;
}

/*
**++
**  FUNCTION NAME: TPCCDisconnect
**--
*/

int
TPCCDisconnect( pCallersContext pCC )
{
  return ERR_DB_SUCCESS;
}


/* stg - added for TuxShutdown function for Tuxedo */
```

```
/*
 *  FUNCTION int TuxShutdown
 */
int
TuxShutdown()
{
  return ERR_DB_SUCCESS;
}

/*
**++
**  FUNCTION NAME: TPCCShutdown
**--
*/
int
TPCCShutdown( void )
{
  int     retcode;

  /* shut down the servers listed in the TUXCONFIG file (ubb* file)
*/
  retcode = system("tmshutdown -y");
  if (retcode != 0)
    {
      TPCCErr("Error shutting the tuxedo servers down.");
      return retcode;
    }

  return(TuxShutdown());
}

/* stg - don't need the following for Tuxedo - I think! */
#if 0
void __cdecl
force_connect( void *arglist )
{
  LoginData   login;
  int     txnType;

  login.w_id = 0;
  login.ld_id = 0;
  login.pCC = 0;
  login.szApplication[0] = '\0';
  strcpy( login.szServer, gszServer );
  strcpy( login.szDatabase, gszDatabase );
  strcpy( login.szUser, gszUser );
  strcpy( login.szPassword, gszPassword );

  txnType = (int) arglist;
  switch ( txnType ) {
  case TYPE_DY:
    dy_transaction_init( STDL_SYNCHRONOUS, &login,
       (struct io_login_wksp *)&login );
    break;

  case TYPE_NO:
    no_transaction_init( STDL_SYNCHRONOUS, &login,
       (struct io_login_wksp *)&login );
    break;

  case TYPE_OS:
    os_transaction_init( STDL_SYNCHRONOUS, &login,
       (struct io_login_wksp *)&login );
    break;

  case TYPE_PT:
    pt_transaction_init( STDL_SYNCHRONOUS, &login,
       (struct io_login_wksp *)&login );
    break;

  case TYPE_SL:
    sl_transaction_init( STDL_SYNCHRONOUS, &login,
       (struct io_login_wksp *)&login );
    break;

  case TYPE_GC:
    gc_transaction_init( STDL_SYNCHRONOUS, &login,
       (struct io_login_wksp *)&login );
    break;
  }
  if ( login.status != ERR_DB_SUCCESS) {
    /** Only store the first failure **/
    if ( ERR_DB_SUCCESS == gInitRetStatus )
      gInitRetStatus = ERR_FORCE_CONNECT_THREAD_FAILED;

    TPCCErr( "Connect Transaction returned %8X\r\n", login.status
);
   }
  if ( InterlockedDecrement( &gForceAllThreadsCtr ) == 0 )
    SetEvent( gForceAllThreadsEvent );
  return;
}
#endif /*stg - end #if 0 section */


-------------------------------------------------
      tux_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.
*
 *
*
 *
*

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

/*
 *
 *
 * Modification history:
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                      - Conversion to run under Linux
 *
 */

#include <errno.h>
#include <unistd.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/time.h>
#include <fcntl.h>

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

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

#include <tpcc.h>

#define NOWHAT

#include <atmi.h>


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

/* dbproc pointer for db connection */
DBContext DBC;

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

FILE  *LogFile;
FILE  *MyLogFile;

#define MAXNUMDIGITS 10

char      szTpccLogPath[FILENAMESIZE];
char      szNumber[MAXNUMDIGITS];



/* 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;
*/

  pDelivery->delta_time = ((pDelivery->tend.tv_sec - pDelivery-
>tbegin.tv_sec) * 1000) + (int) ceil((pDelivery->tend.tv_usec -
pDelivery->tbegin.tv_usec)/1000);

  memcpy( &start, localtime( &pDelivery->tbegin.tv_sec), sizeof(
start ));
  memcpy( &end, localtime( &pDelivery->tend.tv_sec), sizeof( end
));

  fprintf( fpLog,
           "%4.4d/%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",
           1900+start.tm_year, start.tm_mon+1, start.tm_mday,
           start.tm_hour, start.tm_min, start.tm_sec,
     (int) pDelivery->tbegin.tv_usec/1000, end.tm_hour,
     end.tm_min, end.tm_sec, (int) pDelivery->tend.tv_usec/1000,
           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] );

  fflush(fpLog);

  return;
}







/*
**++
**  FUNCTION NAME: tpsvrinit
**--
*/
int
tpsvrinit( int argc, char *argv[] )
{
/*
  BOOL      bLog;
*/
  /* stg next two lines not needed for v6 web ora tux app code
    StartupData   Startup;
    pStartupData    pStartup = &Startup; */
    int status, myfd;
    char  szTmp[FILENAMESIZE];
    LoginData  login;

    /* to avoid compiler errors */
    argc = argc;
    argv = argv;

    /* used for debugging the server code */
/*
    sleep(30000);
*/

    userlog("Starting tpcc server");

    /* Get login data from file settings */
  status = GetConfigValue("Server", (char *)&szTmp);
  if ( status != ERROR_SUCCESS )
    return ERR_CANT_FIND_SERVER_VALUE;
  strcpy(login.szServer, szTmp);


  status = GetConfigValue("Database", (char *)&szTmp);
  if ( status != ERROR_SUCCESS )
    return ERR_CANT_FIND_DATABASE_VALUE;
  strcpy(login.szDatabase, szTmp);


  status = GetConfigValue("User", (char *)&szTmp);
  if ( status != ERROR_SUCCESS )
    return ERR_CANT_FIND_USER_VALUE;
  strcpy(login.szUser, szTmp);


  status = GetConfigValue("Password", (char *)&szTmp);
  if ( status != ERROR_SUCCESS )
    return ERR_CANT_FIND_PASSWORD_VALUE;
  strcpy(login.szPassword, szTmp);

  /* Get Path registry value */
  status = GetConfigValue("PATH", (char *)&szTmp);
  if (status != ERROR_SUCCESS )
    return ERR_CANT_FIND_PATH_VALUE;
  strcpy (szTpccLogPath, szTmp);

  if (ERROR_SUCCESS == status)
  {
    /* set application name */
/*     strcpy( pStartup->Login.databaseLogin.szApplication,
"TUX_SRV" ); */

    TPCCStartupDB();

/* populate LoginData login structure like in tpcc_fct.c */
/* Server, Database, User and Password already populated into login
above */
   login.w_id = 0;
   login.ld_id = 0;
   login.pCC = 0;
   login.szApplication[0] = '\0';

   strcpy(szTmp, szTpccLogPath);
   strcat(szTmp, "delilog");
   itoa(getpid(), szNumber);
   strcat(szTmp, szNumber);
   myfd = fdopen(szTmp, O_WRONLY|O_CREAT|O_DIRECT);
   fpLog = fdopen(myfd, "w");
   if ( NULL == fpLog )
      return ERR_CANNOT_CREATE_RESULTS_FILE;

    status = TPCCConnectDB( (OraContext  **)&DBC, &login );

    if(ERR_DB_SUCCESS != status)
    {
      TPCCErr( "tpsvrinit : Error logging into db." );
      return ERR_DB_ERROR;
    }
    TPCCErr( "Finished TPCCConnectDB, dbprocptr = %8X\r\n", DBC );
  }
  else
  {
    TPCCErr("tpsvrinit : could not get configuration settings");
  }


  return (0);
}


/*
**++
**  FUNCTION NAME: tpsvrdone
**--
*/
void tpsvrdone(void)
{
  TPCCShutdownDB();
  return;
}

/*
**++
**  FUNCTION NAME: dy_transaction
**--
*/
void
dy_transaction( TPSVCINFO *dy_wksp )
{

//  struct timeval  tv;
  struct timezone tz;
//  struct tm   tmp1,tmp2;

  pDeliveryData ptr;

  ptr = (pDeliveryData)dy_wksp->data;

  /* Additional Delivery error logging
  gettimeofday(&tv, &tz);
  memcpy( &tmp1, localtime( &ptr->tbegin.tv_sec), sizeof( tmp1 ));
  memcpy( &tmp2, localtime( &tv), sizeof( tmp2 ));

  TPCCErr( "%2.2d:%2.2d:%2.2d:%3.3d,"
           "%2.2d:%2.2d:%2.2d:%3.3d,"
           "%5.5d",
           tmp1.tm_hour, tmp1.tm_min, tmp1.tm_sec,
           (int) ptr->tbegin.tv_usec/1000, tmp2.tm_hour,
           tmp2.tm_min, tmp2.tm_sec, (int) tv.tv_usec/1000,
           ptr->w_id);
   */

  ptr->status = TPCCDeliveryDB( DBC, ptr );

  gettimeofday(&ptr->tend, &tz);

  /* update log */
  DELILog( ptr );

  if(ERR_DB_ERROR != ptr->status)
    tpreturn(TPSUCCESS, ptr->status, dy_wksp->data, dy_wksp->len,
0);
  else
    tpreturn(TPFAIL, ptr->status, dy_wksp->data, 0L, 0);
}

/*
**++
**  FUNCTION NAME: no_transaction
**--
*/
void
no_transaction( TPSVCINFO *no_wksp )
{
  pNewOrderData ptr;

  ptr = (pNewOrderData)no_wksp->data;

  ptr->status = TPCCNewOrderDB( DBC, ptr );
  if(ERR_DB_ERROR != ptr->status)
    tpreturn(TPSUCCESS, ptr->status, no_wksp->data, no_wksp->len,
0);
  else
    tpreturn(TPFAIL, ptr->status, no_wksp->data, 0L, 0);
}

/*
**++
**  FUNCTION NAME: os_transaction
**--
*/
void
os_transaction( TPSVCINFO *os_wksp )
{
  pOrderStatusData ptr;

  ptr = (pOrderStatusData)os_wksp->data;

  ptr->status = TPCCOrderStatusDB( DBC, ptr );
  if(ERR_DB_ERROR != ptr->status)
    tpreturn(TPSUCCESS, ptr->status, os_wksp->data, os_wksp->len,
0);
  else
  {
    TPCCErr("os_transaction: %d\n",ptr->status);
    tpreturn(TPFAIL, ptr->status, os_wksp->data, 0L, 0);
  }
}

/*
**++
**  FUNCTION NAME: pt_transaction
**--
*/
void
pt_transaction( TPSVCINFO *pt_wksp )
{
  pPaymentData ptr;

  ptr = (pPaymentData)pt_wksp->data;

  ptr->status = TPCCPaymentDB( DBC, ptr );
  if(ERR_DB_ERROR != ptr->status)
    tpreturn(TPSUCCESS, ptr->status, pt_wksp->data,
sizeof(PaymentData), 0);
  else
    tpreturn(TPFAIL, ptr->status, pt_wksp->data, 0L, 0);

}

/*
**++
**  FUNCTION NAME: sl_transaction
**--
*/
void
sl_transaction( TPSVCINFO *sl_wksp )
{
  pStockLevelData ptr;

  ptr = (pStockLevelData)sl_wksp->data;

  ptr->status = TPCCStockLevelDB( DBC, ptr );
  if(ERR_DB_ERROR != ptr->status)
    tpreturn(TPSUCCESS, ptr->status, sl_wksp->data, sl_wksp->len,
0);
  else
    tpreturn(TPFAIL, ptr->status, sl_wksp->data, 0L, 0);
}


------------------------------------------------
      util.c
------------------------------------------------

/*
 *
 *
 *      08/01/2002      Andrew Bond, HP
 *                      - Configuration values are stored in a
filesystem file under Linux
 *                        rather than the Windows registry.
 *
 */

#include <stdio.h>


#define MAXCFGLINE 255
#define CONFIGFILENAME "/usr/local/etc/tpcc.conf"


/*  FUNCTION:  int GetConfigValue(char *option, char *value)
 *
 *  Read the Linux tpcc configuration file
 *
*/
int GetConfigValue(char *option, char *value)
{
FILE    *cfFD;
char    line[MAXCFGLINE];
char optname[MAXCFGLINE];
char *poptname, *tmpValue, *linep;
int full_len, half_len, len;
short notfound=1;

poptname=(char *)&optname;

cfFD=fopen(CONFIGFILENAME, "r");

if (cfFD == NULL)
{
        printf("Error opening file\n");
        return -1;
}
linep=(char *)&line;

while ((fgets(linep, MAXCFGLINE, cfFD) != NULL) && (notfound))
{
  tmpValue=(char *)index(linep, '=');

  if (tmpValue==NULL)
  {
        printf("Equals sign not found\n");
        continue;
  }

  full_len=strlen(linep);
  half_len=strlen(tmpValue);

  strncpy(poptname,linep, full_len-half_len);
  optname[full_len-half_len] = '\0';
  tmpValue++;

  if (!strcmp(optname, option))
  {
        len=strlen(tmpValue);
        strncpy(value, tmpValue, len-1);
        value[len-1] = '\0';
        notfound=0;
  }
}

fclose(cfFD);

if (notfound)
  return(0);
else
  return(1);


}


------------------------------------------------
      paynz.sql
------------------------------------------------

DECLARE /* paynz */
      not_serializable         EXCEPTION;
      PRAGMA EXCEPTION_INIT(not_serializable,-8177);
      deadlock                 EXCEPTION;
      PRAGMA EXCEPTION_INIT(deadlock,-60);
      snapshot_too_old         EXCEPTION;
      PRAGMA EXCEPTION_INIT(snapshot_too_old,-1555);
   BEGIN
      LOOP BEGIN
            UPDATE ware
               SET w_ytd = w_ytd + :h_amount
             WHERE w_id = :w_id
         RETURNING w_name, w_street_1, w_street_2, w_city, w_state,
w_zip
              INTO inittpcc.ware_name, :w_street_1, :w_street_2,
:w_city,
                   :w_state, :w_zip;

           UPDATE  cust
              SET  c_balance = c_balance - :h_amount,
                   c_ytd_payment = c_ytd_payment + :h_amount,
                   c_payment_cnt = c_payment_cnt+1
          WHERE    c_id = :c_id AND c_d_id = :c_d_id AND
                   c_w_id = :c_w_id
         RETURNING rowid, c_first, c_middle, c_last, 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
              INTO inittpcc.cust_rowid,:c_first, :c_middle,
:c_last, :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;
            IF SQL%NOTFOUND THEN
              raise NO_DATA_FOUND;
            END IF;


         IF :c_credit = 'BC' THEN
            UPDATE cust
         SET c_data = substr ((to_char (:c_id) || ' ' ||
                                    to_char (:c_d_id) || ' ' ||
                                    to_char (:c_w_id) || ' ' ||
                                    to_char (:d_id) || ' ' ||
                                    to_char (:w_id) || ' ' ||
                                    to_char (:h_amount/100,
'9999.99') || ' | ')
                                    || c_data, 1, 500)
             WHERE rowid = inittpcc.cust_rowid
         RETURNING substr(c_data,1, 200)
              INTO :c_data;

         END IF;

            UPDATE dist
               SET d_ytd = d_ytd + :h_amount
             WHERE d_id = :d_id
               AND d_w_id = :w_id
         RETURNING d_name, d_street_1, d_street_2, d_city,d_state,
d_zip
              INTO
inittpcc.dist_name,:d_street_1,:d_street_2,:d_city,:d_state,
                   :d_zip;
          IF SQL%NOTFOUND THEN
            raise NO_DATA_FOUND;
          END IF;


         INSERT INTO hist  (h_c_id, h_c_d_id, h_c_w_id, h_d_id,
h_w_id,
                               h_amount, h_date, h_data)
         VALUES
            (:c_id, :c_d_id, :c_w_id, :d_id, :w_id, :h_amount,
             :cr_date, inittpcc.ware_name || '    ' ||
inittpcc.dist_name);
         EXIT;

         EXCEPTION
            WHEN not_serializable OR deadlock OR snapshot_too_old
THEN
               ROLLBACK;
               :retry := :retry + 1;
         END;

      END LOOP;
   END;


-------------------------------------------------
      payz.sql
-------------------------------------------------

DECLARE /* payz */
      not_serializable         EXCEPTION;
      PRAGMA EXCEPTION_INIT(not_serializable,-8177);
      deadlock                 EXCEPTION;
      PRAGMA EXCEPTION_INIT(deadlock,-60);
      snapshot_too_old         EXCEPTION;
      PRAGMA EXCEPTION_INIT(snapshot_too_old,-1555);
   BEGIN
      LOOP BEGIN
         UPDATE ware
            SET w_ytd = w_ytd+:h_amount
            WHERE w_id = :w_id
            RETURNING w_name,
                      w_street_1, w_street_2, w_city, w_state,
w_zip
              INTO inittpcc.ware_name,
                   :w_street_1, :w_street_2, :w_city, :w_state,
:w_zip;


         SELECT rowid
         BULK COLLECT INTO inittpcc.row_id
         FROM cust
         WHERE c_d_id = :c_d_id AND c_w_id = :c_w_id AND c_last =
:c_last
         ORDER BY c_last, c_d_id, c_w_id, c_first;

       inittpcc.c_num := sql%rowcount;
       inittpcc.cust_rowid := inittpcc.row_id((inittpcc.c_num) /
2);

        UPDATE cust
          SET c_balance = c_balance - :h_amount,
              c_ytd_payment = c_ytd_payment+ :h_amount,
              c_payment_cnt = c_payment_cnt+1
        WHERE rowid = inittpcc.cust_rowid
        RETURNING
                c_id, c_first, c_middle, c_last, 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
            INTO :c_id, :c_first, :c_middle, :c_last,
                 :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 := ' ';
         IF :c_credit = 'BC' THEN
            UPDATE cust
              SET c_data = substr ((to_char (:c_id) || ' ' ||
                                    to_char (:c_d_id) || ' ' ||
                                    to_char (:c_w_id) || ' ' ||
                                    to_char (:d_id) || ' ' ||
                                    to_char (:w_id) || ' ' ||
                                    to_char (:h_amount/100,
'9999.99') || ' | ')
                                    || c_data, 1, 500)
              WHERE rowid = inittpcc.cust_rowid
              RETURNING substr(c_data,1, 200)
              INTO :c_data;

         END IF;

         UPDATE dist
            SET d_ytd = d_ytd+:h_amount
            WHERE d_id = :d_id
              AND d_w_id = :w_id
            RETURNING  d_name, d_street_1, d_street_2, d_city,
           d_state, d_zip
            INTO inittpcc.dist_name, :d_street_1, :d_street_2,
:d_city,
                 :d_state, :d_zip;

            IF  SQL%NOTFOUND
      THEN
    raise NO_DATA_FOUND;
            END IF;

         INSERT INTO hist (h_c_id, h_c_d_id, h_c_w_id, h_d_id,
h_w_id,
                              h_amount, h_date, h_data)
             VALUES (:c_id, :c_d_id, :c_w_id, :d_id, :w_id,
:h_amount,
             :cr_date, inittpcc.ware_name || '    ' ||
inittpcc.dist_name);

         EXIT;

         EXCEPTION
            WHEN not_serializable OR deadlock OR snapshot_too_old
THEN
               ROLLBACK;
               :retry := :retry + 1;
         END;

      END LOOP;
   END;


-------------------------------------------------
      tkvcinin.sql
-------------------------------------------------

-- The initnew package for storing variables used in the
-- New Order anonymous block

CREATE OR REPLACE PACKAGE inittpcc
AS
 TYPE intarray IS TABLE OF INTEGER INDEX BY BINARY_INTEGER;
 TYPE distarray IS TABLE OF VARCHAR(24) INDEX BY BINARY_INTEGER;
 nulldate       DATE;
 TYPE rowidarray IS TABLE OF ROWID INDEX BY PLS_INTEGER;
 s_dist     distarray;
 idx1arr    intarray;
 s_remote   intarray;
 dist                   intarray;
 row_id                 rowidarray;
 cust_rowid             rowid;
 dist_name              VARCHAR2(11);
 ware_name              VARCHAR2(11);
 c_num                  PLS_INTEGER;

 PROCEDURE init_no(idxarr intarray);
 PROCEDURE init_del;
 PROCEDURE init_pay;
END inittpcc;
/
show errors;

CREATE OR REPLACE PACKAGE BODY inittpcc AS
  PROCEDURE init_no (idxarr  intarray)
  IS
  BEGIN
        -- initialize null date
   nulldate := TO_DATE('01-01-1811', 'MM-DD-YYYY');
    idx1arr := idxarr;
  END init_no;

  PROCEDURE init_del
  IS
  BEGIN
    FOR i IN 1 .. 10 LOOP
      dist(i) := i;
    END LOOP;
  END init_del;

  PROCEDURE init_pay IS
  BEGIN
    NULL;
  END init_pay;

END inittpcc;
/
show errors
exit


-------------------------------------------------
      tkvcpdel.sql
-------------------------------------------------

declare
  TYPE numarray IS TABLE OF NUMBER INDEX BY BINARY_INTEGER;
  TYPE numlist is varray (10) of number;
  dist numarray;
  amt numarray ;
  cnt pls_integer;

  not_serializable EXCEPTION;
  PRAGMA EXCEPTION_INIT(not_serializable, -8177);
  deadlock         EXCEPTION;
  PRAGMA EXCEPTION_INIT(deadlock, -60);
  snapshot_too_old EXCEPTION;
  PRAGMA EXCEPTION_INIT(snapshot_too_old, -1555);

BEGIN
  LOOP BEGIN
    FORALL d IN 1..10
      DELETE FROM nord
        WHERE no_d_id = inittpcc.dist(d)
          AND no_w_id = :w_id
          AND ROWNUM <= 1
        RETURNING no_d_id, no_o_id BULK COLLECT INTO :d_id,
:order_id;

    :ordcnt := 10;


    FORALL o in 1.. :ordcnt
      UPDATE ordr SET o_carrier_id = :carrier_id
      WHERE o_id = :order_id (o)
        AND o_d_id = :d_id(o)
        AND o_w_id = :w_id
      RETURNING o_c_id BULK COLLECT INTO :o_c_id;

    FORALL o in 1.. :ordcnt
      UPDATE ordl SET ol_delivery_d = :now
      WHERE ol_w_id = :w_id
        AND ol_d_id = :d_id(o)
        AND ol_o_id = :order_id(o)
      RETURNING sum(ol_amount) BULK COLLECT INTO  :sums;

    FORALL c IN 1.. :ordcnt
      UPDATE cust
         SET c_balance = c_balance + :sums(c),
                         c_delivery_cnt = c_delivery_cnt + 1
       WHERE c_w_id = :w_id
         AND c_d_id = :d_id(c)
         AND c_id = :o_c_id(c);
    COMMIT;
    EXIT;
    EXCEPTION
      WHEN not_serializable OR deadlock OR snapshot_too_old
      THEN
        ROLLBACK;
        :retry := :retry + 1;
    END;

  END LOOP; -- for retry
END;


-------------------------------------------------
      tkvcpnew.sql
-------------------------------------------------


-- New Order Anonymous block

  DECLARE
      idx                      PLS_INTEGER;
      dummy_local              PLS_INTEGER;
      cache_ol_cnt             PLS_INTEGER;
      not_serializable         EXCEPTION;
      PRAGMA EXCEPTION_INIT(not_serializable,-8177);
      deadlock                 EXCEPTION;
      PRAGMA EXCEPTION_INIT(deadlock,-60);
      snapshot_too_old         EXCEPTION;
      PRAGMA EXCEPTION_INIT(snapshot_too_old,-1555);

   PROCEDURE u1 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                             THEN s_quantity +91
                             ELSE s_quantity
                        END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_01,
                    i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
                  ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                        THEN 'G'
                        ELSE 'B'
                        END)
             END
        BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                          :ol_amount,:brand_generic;
   END u1;

   PROCEDURE u2 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                             THEN s_quantity +91
                             ELSE s_quantity
                        END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_02,
                  i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
                  ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                        THEN 'G'
                        ELSE 'B'
                        END)
             END
        BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                          :ol_amount,:brand_generic;
   END u2;

   PROCEDURE u3 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                             THEN s_quantity +91
                             ELSE s_quantity
                        END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_03,
                    i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
```

```
                ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                      THEN 'G'
                      ELSE 'B'
                      END)
          END
       BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                        :ol_amount,:brand_generic;
   END u3;

   PROCEDURE u4 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                           THEN s_quantity +91
                           ELSE s_quantity
                      END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_04,
                    i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
                  ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                        THEN 'G'
                        ELSE 'B'
                        END)
            END
       BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                        :ol_amount,:brand_generic;
   END u4;

   PROCEDURE u5 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                           THEN s_quantity +91
                           ELSE s_quantity
                      END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_05,
                    i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
                  ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                        THEN 'G'
                        ELSE 'B'
                        END)
            END
       BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                        :ol_amount,:brand_generic;
   END u5;

   PROCEDURE u6 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                           THEN s_quantity +91
                           ELSE s_quantity
                      END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_06,
                    i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
                  ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                        THEN 'G'
                        ELSE 'B'
                        END)
            END
       BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                        :ol_amount,:brand_generic;
   END u6;

   PROCEDURE u7 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                           THEN s_quantity +91
                           ELSE s_quantity
                      END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_07,
                    i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
                  ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                        THEN 'G'
                        ELSE 'B'
                        END)
            END
       BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                        :ol_amount,:brand_generic;
   END u7;

   PROCEDURE u8 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                           THEN s_quantity +91
                           ELSE s_quantity
                      END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_08,
                    i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
                  ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                        THEN 'G'
                        ELSE 'B'
                        END)
            END
       BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                        :ol_amount,:brand_generic;
   END u8;

   PROCEDURE u9 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                           THEN s_quantity +91
                           ELSE s_quantity
                      END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_09,
                    i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
                  ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                        THEN 'G'
                        ELSE 'B'
                        END)
            END
       BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                        :ol_amount,:brand_generic;
   END u9;

   PROCEDURE u10 IS
   BEGIN
        FORALL idx IN 1 .. cache_ol_cnt
          UPDATE  stock_item
          SET s_order_cnt = s_order_cnt + 1,
          s_ytd = s_ytd + :ol_quantity(idx),
          s_remote_cnt = s_remote_cnt + :s_remote(idx),
          s_quantity = (CASE WHEN s_quantity < :ol_quantity (idx) +
10
                           THEN s_quantity +91
                           ELSE s_quantity
                      END) - :ol_quantity(idx)
          WHERE i_id = :ol_i_id(idx)
          AND s_w_id = :ol_supply_w_id(idx)
          RETURNING i_price, i_name, s_quantity, s_dist_10,
                    i_price*:ol_quantity(idx),
            CASE WHEN i_data NOT LIKE '%ORIGINAL%'
                 THEN 'G'
                  ELSE (CASE WHEN s_data NOT LIKE '%ORIGINAL%'
                        THEN 'G'
                        ELSE 'B'
                        END)
            END
        BULK COLLECT INTO :i_price, :i_name, :s_quantity,
inittpcc.s_dist,
                        :ol_amount,:brand_generic;
   END u10;

   PROCEDURE fix_items IS
     rows_lost                      PLS_INTEGER;
     max_index                      PLS_INTEGER;
     temp_index                     PLS_INTEGER;
   BEGIN
     idx := 1;
     rows_lost := 0;
     max_index := dummy_local;

     WHILE (max_index != cache_ol_cnt) LOOP

      WHILE (idx <= sql%rowcount AND
                   sql%bulk_rowcount(idx + rows_lost) = 1)
      LOOP
        idx := idx + 1;
      END LOOP;

      temp_index := max_index;
      WHILE (temp_index >= idx + rows_lost) LOOP
         :ol_amount(temp_index + 1)      :=
:ol_amount(temp_index);
         :i_price(temp_index + 1)        :=  :i_price(temp_index);
         :i_name(temp_index + 1)         :=  :i_name(temp_index);
         :s_quantity(temp_index + 1)     :=
:s_quantity(temp_index);
         inittpcc.s_dist(temp_index + 1) :=
inittpcc.s_dist(temp_index);
         :brand_generic(temp_index + 1)  :=
:brand_generic(temp_index);
         temp_index := temp_index - 1;
      END LOOP;

      IF (idx + rows_lost <= cache_ol_cnt) THEN
        :i_price(idx + rows_lost)        :=   0;
        :i_name(idx + rows_lost)         :=   'NO ITEM';
        :s_quantity(idx + rows_lost)     :=   0;
        inittpcc.s_dist(idx + rows_lost) := NULL;
        :brand_generic(idx + rows_lost)  := ' ';
        :ol_amount(idx + rows_lost)      := 0;
        rows_lost := rows_lost + 1;
        max_index := max_index + 1;
      END IF;

    END LOOP;
   END fix_items;

   BEGIN
      LOOP BEGIN
         cache_ol_cnt := :o_ol_cnt;

         UPDATE dist SET d_next_o_id = d_next_o_id + 1
          WHERE d_id = :d_id AND  d_w_id = :w_id
          RETURNING d_tax, d_next_o_id-1
          INTO :d_tax, :o_id;

         SELECT c_discount, c_last, c_credit, w_tax
          INTO :c_discount, :c_last, :c_credit , :w_tax
          FROM cust , ware
          WHERE c_id = :c_id AND c_d_id = :d_id AND c_w_id = w_id
          AND  w_id = :w_id;


         INSERT INTO nord (no_o_id, no_d_id, no_w_id)
          VALUES (:o_id, :d_id, :w_id);

         INSERT INTO ordr  (o_id,o_d_id, o_w_id, o_c_id, o_entry_d,
                      o_carrier_id, o_ol_cnt, o_all_local)
          VALUES (:o_id, :d_id, :w_id, :c_id,
                  :cr_date, 11, :o_ol_cnt, :o_all_local);


         dummy_local :=  :d_id;

         IF (dummy_local < 6) THEN
           IF (dummy_local < 3) THEN
             IF (dummy_local = 1) THEN
                u1;
             ELSE
                u2;
             END IF;
           ELSE
             IF (dummy_local = 3) THEN
                u3;
             ELSIF (dummy_local = 4) then
                u4;
             ELSE
                u5;
             END IF;
           END IF;
         ELSE
           IF (dummy_local < 8) THEN
             IF (dummy_local = 6) THEN
                u6;
             ELSE
                u7;
             END IF;
           ELSE
             IF (dummy_local = 8) THEN
                u8;
             ELSIF (dummy_local = 9) then
                u9;
             ELSE
                u10;
             END IF;
           END IF;
         END IF;

        dummy_local := sql%rowcount;

        IF (dummy_local != cache_ol_cnt )  THEN fix_items; END IF;

        FORALL idx IN 1..dummy_local
         INSERT INTO ordl
            (ol_o_id, ol_d_id, ol_w_id, ol_number, ol_delivery_d,
ol_i_id,
                  ol_supply_w_id,
ol_quantity,ol_amount,ol_dist_info)
         VALUES (:o_id, :d_id, :w_id, inittpcc.idx1arr(idx),
inittpcc.nulldate,
                 :ol_i_id(idx), :ol_supply_w_id(idx),
                 :ol_quantity(idx), :ol_amount(idx),
inittpcc.s_dist(idx));

        IF (dummy_local != :o_ol_cnt) THEN
           :o_ol_cnt := dummy_local;
           ROLLBACK;
        END IF;

    EXIT;

    EXCEPTION
           WHEN not_serializable OR deadlock OR snapshot_too_old
THEN
               ROLLBACK;
               :retry := :retry + 1;
         END;
      END LOOP;
   END;


------------------------------------------------
      views.sql
------------------------------------------------


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
/





# PRTE COMMAND FILE
# C_LAST        is the constant value used for customer last names.
database.set network_variable C_LAST      87
```

Appendix B: Database Design

```
--------------------------------------------------

                addfile.sh
--------------------------------------------------

#!/bin/sh
# $1 = tablespace name
# $2 = filename
# $3 = size
# $4 = temporary ts (1) or not (0)
# global variable $tpcc_listfiles, does not execute sql

if expr x$tpcc_listfiles = xt > /dev/null; then
  echo $2 $3 >> $tpcc_bench/files.dat
  exit 0
fi

if expr $4 = 1 > /dev/null; then
  altersql="alter tablespace $1 add tempfile '$2' size $3 reuse;"
else
  altersql="alter tablespace $1 add datafile '$2' size $3 reuse autoextend on;"
fi

$tpcc_sqlplus $tpcc_user_pass <<!
   spool addfile_$1.log
   set echo on
  $altersql
   set echo off
   spool off
   exit ;
!


--------------------------------------------------
                addts.sh
--------------------------------------------------

#!/bin/sh
# $1 = tablespace name
# $2 = filename
# $3 = size
# $4 = uniform size
# $5 = block size
# $6 = temporary ts (1) or not (0)
# $7 = bitmapped manage (t) or not (f)
# global variable $tpcc_listfiles, does not execute sql

if expr x$tpcc_listfiles = xt > /dev/null; then
  echo $2 $3 >> $tpcc_bench/files.dat
  exit 0
fi


if expr $5 = auto > /dev/null; then
  bssql=
else
  bssql="blocksize $5"
fi

if expr $6 = 1 > /dev/null; then
  createsql="create temporary tablespace $1 tempfile '$2' size $3 reuse extent management local
uniform size $4;"
else
  if expr x$7 = xt > /dev/null; then
    autospace=auto
  else
    autospace=manual
  fi
    createsql="create tablespace $1 datafile '$2' size $3 reuse extent management local uniform size $4
segment space management $autospace $bssql nologging ;"
fi

$tpcc_sqlplus $tpcc_user_pass <<!
   spool createts_$1.log
   set echo on
   drop tablespace $1 including contents;
 $createsql
   set echo off
   spool off
   exit ;
!


--------------------------------------------------
                analyze.sql
--------------------------------------------------

spool analyze.log;
set echo on;

alter user system temporary tablespace temp_0;
connect system/manager

execute dbms_stats.GATHER_TABLE_STATS (OWNNAME=>'TPCC', -
                                TABNAME=>'STOK', -
                                PARTNAME=>NULL, -
                                ESTIMATE_PERCENT=>1, -
                                BLOCK_SAMPLE=>TRUE, -
                                METHOD_OPT=>'FOR ALL COLUMNS SIZE 1', -
                                DEGREE=>10, -
                                GRANULARITY=>'DEFAULT', -
                                CASCADE=>TRUE);

execute dbms_stats.GATHER_TABLE_STATS (OWNNAME=>'TPCC', -
                                TABNAME=>'CUST', -
                                PARTNAME=>NULL, -
                                ESTIMATE_PERCENT=>1, -
                                BLOCK_SAMPLE=>TRUE, -
                                METHOD_OPT=>'FOR ALL COLUMNS SIZE 1', -
                                DEGREE=>10, -
                                GRANULARITY=>'DEFAULT', -
                                CASCADE=>TRUE);

execute dbms_stats.GATHER_TABLE_STATS (OWNNAME=>'TPCC', -
                                TABNAME=>'ORDR', -
                                PARTNAME=>NULL, -
                                ESTIMATE_PERCENT=>1, -
                                BLOCK_SAMPLE=>TRUE, -
                                METHOD_OPT=>'FOR ALL COLUMNS SIZE 1', -
                                DEGREE=>10, -
                                GRANULARITY=>'DEFAULT', -
                                CASCADE=>TRUE);

execute dbms_stats.GATHER_TABLE_STATS (OWNNAME=>'TPCC', -
                                TABNAME=>'ORDL', -
                                PARTNAME=>NULL, -
                                ESTIMATE_PERCENT=>1, -
                                BLOCK_SAMPLE=>TRUE, -
                                METHOD_OPT=>'FOR ALL COLUMNS SIZE 1', -
                                DEGREE=>10, -
                                GRANULARITY=>'DEFAULT', -
                                CASCADE=>TRUE);

execute dbms_stats.GATHER_TABLE_STATS (OWNNAME=>'TPCC', -
                                TABNAME=>'NORD', -
                                PARTNAME=>NULL, -
                                ESTIMATE_PERCENT=>1, -
                                BLOCK_SAMPLE=>TRUE, -
                                METHOD_OPT=>'FOR ALL COLUMNS SIZE 1', -
                                DEGREE=>10, -
                                GRANULARITY=>'DEFAULT', -
                                CASCADE=>TRUE);

execute dbms_stats.GATHER_TABLE_STATS(OWNNAME=>'TPCC', -
                                TABNAME=>'HIST', -
                                PARTNAME=>NULL, -
                                ESTIMATE_PERCENT=>1, -
                                BLOCK_SAMPLE=>TRUE, -
                                METHOD_OPT=>'FOR ALL COLUMNS SIZE 1', -
                                DEGREE=>10, -
                                GRANULARITY=>'DEFAULT', -
                                CASCADE=>TRUE);

execute dbms_stats.GATHER_TABLE_STATS(OWNNAME=>'TPCC', -
                                TABNAME=>'DIST', -
                                PARTNAME=>NULL, -
                                ESTIMATE_PERCENT=>1, -
                                BLOCK_SAMPLE=>TRUE, -
                                METHOD_OPT=>'FOR ALL COLUMNS SIZE 1', -
                                DEGREE=>10, -
                                GRANULARITY=>'DEFAULT', -
```

```
                              CASCADE=>TRUE);

execute dbms_stats.GATHER_TABLE_STATS(OWNNAME=>'TPCC', -
                              TABNAME=>'ITEM', -
                              PARTNAME=>NULL, -
                              ESTIMATE_PERCENT=>10, -
                              BLOCK_SAMPLE=>TRUE, -
                              METHOD_OPT=>'FOR ALL COLUMNS SIZE 1', -
                              DEGREE=>1, -
                              GRANULARITY=>'DEFAULT', -
                              CASCADE=>TRUE);

execute dbms_stats.GATHER_TABLE_STATS(OWNNAME=>'TPCC', -
                              TABNAME=>'WARE', -
                              PARTNAME=>NULL, -
                              ESTIMATE_PERCENT=>10, -
                              BLOCK_SAMPLE=>TRUE, -
                              METHOD_OPT=>'FOR ALL COLUMNS SIZE 1', -
                              DEGREE=>10, -
                              GRANULARITY=>'DEFAULT', -
                              CASCADE=>TRUE);



set echo off;
spool off;

exit sql.sqlcode;


--------------------------------------------------
              assigntemp.sql
--------------------------------------------------

spool assigntemp.log;

set echo on;

alter user tpcc temporary tablespace temp_0;

set echo off;
spool off;

exit ;


--------------------------------------------------
              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

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 $oracle_dba/$oracle_dba_password;
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;


-------------------------------------------------
                createdb.sql
-------------------------------------------------

/* created automatically by  /home/weshi/tpcc10800/scripts/buildcreatedb.sh Fri May 16 10:30:59
PDT 2003 */
spool createdb.log

set echo on

shutdown abort

startup pfile=p_create.ora nomount
create database tpcc
  controlfile reuse
  maxinstances 1
  datafile '$tpcc_disks_location/system_001' size 200M reuse
  logfile '$tpcc_disks_location/log_1' size 18500M reuse,
          '$tpcc_disks_location/log_2' size 18500M reuse
  sysaux datafile '$tpcc_disks_location/aux.df' size 120M reuse ;

create undo tablespace undo_ts datafile
   '$tpcc_disks_location/roll01' size 8096M reuse blocksize 8K;

set echo off
exit sql.sqlcode


-------------------------------------------------
                createindex_icust1.sql
-------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:31:53
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createindex_icust1.log ;
    set echo on ;
    drop index icust1 ;
      create unique index icust1 on cust ( c_w_id
, c_d_id
, c_id )
  pctfree 1  initrans 3
  storage ( buffer_pool default )
  parallel
  tablespace icust1_0 ;
    set echo off
    spool off
    exit sql.sqlcode;


-------------------------------------------------
                createindex_icust2.sql
-------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:31:54
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createindex_icust2.log ;
    set echo on ;
    drop index icust2 ;
      create unique index icust2 on cust ( c_last
, c_w_id
, c_d_id
, c_first
, c_id )
  pctfree 1  initrans 3
  storage ( buffer_pool default )
  parallel
  tablespace icust2_0 ;
    set echo off
    spool off
    exit sql.sqlcode;


--------------------------------------------------
                createindex_idist.sql
--------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:31:57
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createindex_idist.log ;
    set echo on ;
    drop index idist ;
      create unique index idist on dist ( d_w_id
, d_id )
  pctfree 5  initrans 3
  storage ( buffer_pool default )
  parallel
  tablespace idist_0 ;
    set echo off
    spool off
    exit sql.sqlcode;


--------------------------------------------------
                createindex_iitem.sql
--------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:32:00
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createindex_iitem.log ;
    set echo on ;
    drop index iitem ;
      create unique index iitem on item ( i_id )
  pctfree 5  initrans 4
  storage ( buffer_pool default )
  parallel
  tablespace iitem_0 ;
    set echo off
    spool off
    exit sql.sqlcode;


--------------------------------------------------
                createindex_inord.sql
--------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:32:08
PDT 2003 */
set timing on
  exit 0;


--------------------------------------------------
                createindex_iordl.sql
--------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:32:07
PDT 2003 */
set timing on
  exit 0;


--------------------------------------------------
                createindex_iordr1.sql
--------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:32:02
PDT 2003 */
set timing on
  exit 0;


--------------------------------------------------
                createindex_iordr2.sql
--------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:32:03
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createindex_iordr2.log ;
    set echo on ;
    drop index iordr2 ;
      create unique index iordr2 on ordr ( o_c_id
, o_d_id
, o_w_id
, o_id )
  pctfree 25  initrans 4
  storage ( buffer_pool default )
  parallel
  tablespace iordr2_0 ;
    set echo off
    spool off
    exit sql.sqlcode;


--------------------------------------------------
                createindex_istok.sql
--------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:31:59
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createindex_istok.log ;
    set echo on ;
    drop index istok ;
      create unique index istok on stok ( s_i_id
, s_w_id )
  pctfree 1  initrans 3
  storage ( buffer_pool default )
  parallel 16
  tablespace istok_0 ;
    set echo off
    spool off
    exit sql.sqlcode;


--------------------------------------------------
                createindex_iware.sql

--------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreateindex.sh Fri May 16 10:31:51
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createindex_iware.log ;
    set echo on ;
    drop index iware ;
      create unique index iware on ware ( w_id )
  pctfree 1  initrans 3
  storage ( buffer_pool default )
  parallel
  tablespace iware_0 ;
    set echo off
    spool off
    exit sql.sqlcode;


--------------------------------------------------
                createmisc.sh
--------------------------------------------------

#!/bin/sh

$tpcc_sqlplus $tpcc_sqlplus_args << !
$tpcc_internal_connect

spool createmisc.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;

REM
REM begin plsql_mon.sql
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;

REM
REM end plsql_mon.sql
REM

REM
REM  begin cre_tab.sql
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;

REM
REM  end cre_tab.sql
REM

REM
REM  begin views.sql
REM

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;

REM
REM  end views.sql
REM


REM
REM  begin dml.sql
REM
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;

set echo off;

REM
REM  end dml.sql
REM

REM
REM  begin extent.sql
REM

$SYS_CONNECTION_STRING

@$tpcc_sql_dir/extent

@$tpcc_sql_dir/freeext

exit sql.sqlcode;

!
```

```
-------------------------------------------------
                 createstoredprocs.sql
-------------------------------------------------

spool createstoredprocs.log
@$tpcc_sql_dir/tkvcinin.sql
spool off
exit sql.sqlcode;
```

```
-------------------------------------------------
                 createtable_cust.sql
-------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreatetable.sh Fri May 16 10:31:04
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createtable_cust.log
    set echo on
      drop cluster custcluster including tables ;

create cluster custcluster (
  c_id number
, c_d_id number
, c_w_id number
```

```
  )
  single table
  hashkeys 324000000
  hash is ( (c_id * ( 10800 * 10 ) + c_w_id * 10 + c_d_id) )
  size 250
  pctfree 0  initrans 3
  storage ( buffer_pool recycle )
  tablespace cust_0;

create table cust (
  c_id number
, c_d_id number
, c_w_id number
, 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
);
    set echo off
    spool off
    exit sql.sqlcode;


-------------------------------------------------
                  createtable_dist.sql
-------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreatetable.sh Fri May 16 10:31:15
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createtable_dist.log
    set echo on
      drop cluster distcluster including tables ;

create cluster distcluster (
  d_id number
, d_w_id number
  )
  single table
  hashkeys 108000
  hash is ( ((d_w_id * 10) + d_id) )

    initrans 4
  storage ( buffer_pool default )
  tablespace dist_0;

create table dist (
  d_id number
, d_w_id number
, d_ytd number
, d_next_o_id number
, d_tax 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)
)
cluster distcluster (
  d_id
, d_w_id
);
    set echo off
    spool off
    exit sql.sqlcode;


-------------------------------------------------
                  createtable_hist.sql
-------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreatetable.sh Fri May 16 10:31:21
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createtable_hist.log
    set echo on
      drop table hist ;

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)
)
  pctfree 5  initrans 4
  storage ( buffer_pool recycle )
  tablespace hist_0 ;
    set echo off
    spool off
    exit sql.sqlcode;


-------------------------------------------------
                  createtable_item.sql
-------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreatetable.sh Fri May 16 10:31:33
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createtable_item.log
    set echo on
      drop cluster itemcluster including tables ;

create cluster itemcluster (
  i_id number(6,0)
  )
  single table
  hashkeys 100000
  hash is ( i_id )
  size 120
  pctfree 0  initrans 3
  storage ( buffer_pool keep )
  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
);
    set echo off
    spool off
    exit sql.sqlcode;


-------------------------------------------------
                  createtable_nord.sql
-------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreatetable.sh Fri May 16 10:31:44
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createtable_nord.log
    set echo on
        drop cluster nordcluster_queue including tables ;

  create cluster nordcluster_queue (
    no_w_id number
, no_d_id number
, no_o_id number SORT
    )

    hashkeys 108000
    hash is ( (no_w_id - 1) * 10 + no_d_id - 1 )
    size 190
    tablespace nord_0;

  create table nord (
    no_w_id number
, no_d_id number
, no_o_id number sort
    , constraint nord_uk primary key ( no_w_id
, no_d_id
, no_o_id )
  )
  cluster nordcluster_queue (
    no_w_id
, no_d_id
, no_o_id
  );
    set echo off
    spool off
    exit sql.sqlcode;
```

```
------------------------------------------------
                createtable_ordl.sql
------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreatetable.sh Fri May 16 10:31:40
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createtable_ordl.log
    set echo on
        create table ordl (
    ol_w_id number
, ol_d_id number
, ol_o_id number sort
, ol_number number sort
, 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 ordl_uk primary key (ol_w_id, ol_d_id, ol_o_id, ol_number  )) CLUSTER
ordrcluster_queue(ol_w_id, ol_d_id, ol_o_id, ol_number)
;
    set echo off
    spool off
    exit sql.sqlcode;
```

```
------------------------------------------------
                createtable_ordr.sql
------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreatetable.sh Fri May 16 10:31:37
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createtable_ordr.log
    set echo on
        drop cluster ordrcluster_queue including tables ;

  create cluster ordrcluster_queue (
    o_w_id number
, o_d_id number
, o_id number SORT
, o_number number SORT
    )

    hashkeys 108000
    hash is ( (o_w_id - 1) * 10 + o_d_id - 1 )
    size 2980
    pctfree 5
    tablespace ordr_0;

  create table ordr (
    o_id number sort
, 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
    , constraint ordr_uk primary key ( o_w_id
, o_d_id
, o_id )
  )
  cluster ordrcluster_queue (
    o_w_id
, o_d_id
, o_id
  );
    set echo off
    spool off
    exit sql.sqlcode;
```

```
------------------------------------------------
                createtable_stok.sql
------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreatetable.sh Fri May 16 10:31:24
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createtable_stok.log
    set echo on
      drop cluster stokcluster including tables ;

create cluster stokcluster (
  s_i_id number
, s_w_id number
  )
  single table
  hashkeys 1080000000
  hash is ( (s_i_id * 10800 + s_w_id) )
  size 350
  pctfree 0  initrans 3
  storage ( buffer_pool keep )
  tablespace stok_0;

create table stok (
  s_i_id number
, s_w_id number
, 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
);
    set echo off
    spool off
    exit sql.sqlcode;
```

```
------------------------------------------------
                createtable_ware.sql
------------------------------------------------

/* created automatically by /home/weshi/tpcc10800/scripts/buildcreatetable.sh Fri May 16 10:31:00
PDT 2003 */
set timing on
    set sqlblanklines on
    spool createtable_ware.log
    set echo on
      drop cluster warecluster including tables ;

create cluster warecluster (
  w_id number(5,0)
  )
  single table
  hashkeys 10800
  hash is ( (w_id) )

    initrans 2
  storage ( buffer_pool default )
  tablespace ware_0;

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)
)
cluster warecluster (
  w_id
);
    set echo off
    spool off
    exit sql.sqlcode;


------------------------------------------------
                createts.sh
------------------------------------------------

#created automatically by /home/weshi/tpcc10800/scripts/buildcreatets.sh Fri May 16 10:29:24 PDT
2003
# Tablespace ware, ts size 27M (27000K)
# each file 1687K (1687K)
# extents 1444K (1444K)
# 16 files
$tpcc_createts ware 1 1       27M 27000K unix 0       0 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for ware failed.  Exiting.
      exit 0
    fi
# Tablespace cust, ts size 321G (336181640K)
# each file 6840M (7004160K)
# extents 1749760K (1749760K)
# 48 files
$tpcc_createts cust 40 1       8388606K 100M unix 0       1 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for cust failed.  Exiting.
      exit 0
    fi
# Tablespace dist, ts size 264M (270000K)
# each file 17M (17408K)
# extents 15565K (15565K)
# 16 files
$tpcc_createts dist 1 1       264M 263M unix 0       41 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for dist failed.  Exiting.
      exit 0
    fi
# Tablespace hist, ts size 24G (24321707K)
# each file 1480M (1515520K)
# extents 99942K (99942K)
# 16 files
$tpcc_createts hist 4 1       7000M 200M unix 0       42 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for hist failed.  Exiting.
      exit 0
    fi
# Tablespace stok, ts size 441G (461425781K)
# each file 7040M (7208960K)
# extents 1800960K (1800960K)
# 64 files
$tpcc_createts stok 56 1       8388606K 100M unix 0       46 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for stok failed.  Exiting.
      exit 0
    fi
# Tablespace item, ts size 16M (15868K)
# each file 1M (1024K)
# extents 876K (876K)
# 16 files
$tpcc_createts item 1 1       16M 16000K unix 0       102 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for item failed.  Exiting.
      exit 0
    fi
# Tablespace ordr, ts size 299G (312868980K)
# each file 19000M (19456000K)
# extents 101370K (101370K)
# 16 files
$tpcc_createts ordr 8 1       39000M 300M unix 0       103 4 16K t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for ordr failed.  Exiting.
      exit 0
    fi
# Tablespace nord, ts size 3G (2439503K)
# each file 149M (152576K)
# extents 74752K (74752K)
# 16 files
$tpcc_createts nord 4 1       1020M 250M unix 0       111 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for nord failed.  Exiting.
      exit 0
    fi
# Tablespace iware, ts size 14M (13500K)
# each file 1M (1024K)
# extents 55K (55K)
# 16 files
$tpcc_createts iware 1 1       14M 500K unix 0       115 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for iware failed.  Exiting.
      exit 0
    fi
# Tablespace icust1, ts size 10G (9936000K)
# each file 607M (621568K)
# extents 2303K (2303K)
# 16 files
$tpcc_createts icust1 2 1       6000M 600K unix 0       116 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for icust1 failed.  Exiting.
      exit 0
    fi
# Tablespace icust2, ts size 60G (61978500K)
# each file 3780M (3870720K)
# extents 14361K (14361K)
# 16 files
$tpcc_createts icust2 8 1       8000M 15M unix 0       118 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for icust2 failed.  Exiting.
      exit 0
    fi
# Tablespace idist, ts size 53M (54000K)
# each file 3375K (3375K)
# extents 180K (180K)
# 16 files
$tpcc_createts idist 1 1       54000K 180K unix 0       126 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for idist failed.  Exiting.
      exit 0
    fi
# Tablespace istok, ts size 27G (28215000K)
# each file 1720M (1761280K)
# extents 6533K (6533K)
# 16 files
$tpcc_createts istok 4 1       7100M 100M unix 0       127 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for istok failed.  Exiting.
      exit 0
    fi
# Tablespace iitem, ts size 3M (2560K)
# each file 1M (1024K)
# extents 55K (55K)
# 16 files
$tpcc_createts iitem 1 1       4M 55K unix 0       131 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for iitem failed.  Exiting.
      exit 0
    fi
# Tablespace iordr2, ts size 15G (15378660K)
# each file 939M (961536K)
# extents 3565K (3565K)
# 16 files
$tpcc_createts iordr2 8 1       2030M 200M unix 0       132 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for iordr2 failed.  Exiting.
      exit 0
    fi
# Tablespace temp, ts size 119G (123957000K)
# each file 7560M (7741440K)
# extents 1934080K (1934080K)
# 16 files
$tpcc_createts temp 16 1       7560M 1934080K unix 1       140 4 auto t
    if expr $? != 0 > /dev/null; then
      echo Creating tablespace for temp failed.  Exiting.
      exit 0
    fi

------------------------------------------------
                createuser.sql
------------------------------------------------

spool createusertpcc.log;

set echo on;

create user tpcc identified by tpcc;

grant dba to tpcc;

set echo off;
spool off;

exit ;


------------------------------------------------
                cs_cpu.sql
------------------------------------------------

rem
rem
rem=================================================================+
rem       Copyright (c) 1997  Oracle Corp, Redwood Shores, CA       |
rem                       All Rights Reserved                       |
rem=================================================================+
rem FILENAME
rem     cs_cpu.sql
rem DESCRIPTION
rem     Create Table for CPU Specific Process Stat
rem=================================================================
rem usage: sqlplus tpcc/tpcc @cs_cpu.sql

connect tpcc/tpcc
set echo on

DROP TABLE pre_cpu_stats;
DROP TABLE post_cpu_stats;
DROP TABLE cpu_stats;

rem
rem  CPU statistics.
rem

       CREATE TABLE cpu_stats
       (
          runname         VARCHAR2(20),
            cpu_id          NUMBER,
          dpc_cpu         NUMBER,
          interrupt_cpu   NUMBER,
          priv_cpu        NUMBER,
          processor_cpu   NUMBER,
          user_cpu        NUMBER,
          interrupt_rate   NUMBER
       );

rem
rem  Save Begining CPU Stat Values
rem

       CREATE TABLE pre_cpu_stats
       (
          runname         VARCHAR2(20),
            cpu_id          NUMBER,
          dpc_cpu         NUMBER,
          interrupt_cpu   NUMBER,
          priv_cpu        NUMBER,
          processor_cpu   NUMBER,
          user_cpu        NUMBER,
          interrupt_rate   NUMBER
       );


rem
rem  Save Ending CPU Stat Values
rem

       CREATE TABLE post_cpu_stats
       (
          runname         VARCHAR2(20),
            cpu_id          NUMBER,
          dpc_cpu         NUMBER,
          interrupt_cpu   NUMBER,
          priv_cpu        NUMBER,
          processor_cpu   NUMBER,
          user_cpu        NUMBER,
          interrupt_rate   NUMBER
       );
commit;
set echo off;


-------------------------------------------------
                cs_os.sql
-------------------------------------------------

rem
rem
rem=================================================================+
rem        Copyright (c) 1997  Oracle Corp, Redwood Shores, CA       |
rem                        All Rights Reserved                        |
rem=================================================================+
rem FILENAME
rem     cs_os.sql
rem DESCRIPTION
rem     Create Table for OS Specific Process Stat
rem=================================================================
rem usage: sqlplus tpcc/tpcc @cs_os.sql

connect tpcc/tpcc
set echo on

DROP TABLE pre_os_stats;
DROP TABLE post_os_stats;
DROP TABLE os_stats;
rem
rem  OS statistics.
rem

       CREATE TABLE os_stats
       (
          runname         VARCHAR2(20),
          time            NUMBER,
          syscall         NUMBER,
          intr            NUMBER,
          cswitch         NUMBER,
          freads          NUMBER,
          fwrites         NUMBER,
          fcontrolops     NUMBER,
          priv_cpu        NUMBER,
          user_cpu        NUMBER,
          processor_cpu   NUMBER,
            interrupt_cpu   NUMBER
       );

rem
rem  Save Begining OS Stat Values
rem

       CREATE TABLE pre_os_stats
       (
          runname         VARCHAR2(20),
          time            NUMBER,
          syscall         NUMBER,
          intr            NUMBER,
          cswitch         NUMBER,
          freads          NUMBER,
          fwrites         NUMBER,
          fcontrolops     NUMBER,
          priv_cpu        NUMBER,
          user_cpu        NUMBER,
          processor_cpu   NUMBER,
            interrupt_cpu   NUMBER
       );


rem
rem  Save Ending OS Stat Values
rem

       CREATE TABLE post_os_stats
       (
          runname         VARCHAR2(20),
          time            NUMBER,
          syscall         NUMBER,
          intr            NUMBER,
          cswitch         NUMBER,
          freads          NUMBER,
          fwrites         NUMBER,
          fcontrolops     NUMBER,
          priv_cpu        NUMBER,
          user_cpu        NUMBER,
          processor_cpu   NUMBER,
            interrupt_cpu   NUMBER
       );
commit;
set echo off;


-------------------------------------------------
                cs_proc.sql
-------------------------------------------------


rem
rem
rem=================================================================+
rem        Copyright (c) 1997  Oracle Corp, Redwood Shores, CA       |
rem                        All Rights Reserved                        |
rem=================================================================+
rem FILENAME
rem     cs_proc.sql
rem DESCRIPTION
rem     Create Table for OS Specific Process Stats
rem=================================================================
rem Usage: sqlplus tpcc/tpcc @cs_proc.sql

connect tpcc/tpcc
set echo on

DROP TABLE process_stats;
DROP TABLE pre_process_stats;
DROP TABLE post_process_stats;

rem
rem  Resource usage for a process.
```

```
rem

    CREATE TABLE process_stats
    (
        runname       VARCHAR2(20),
        user_cpu      NUMBER,
        priv_cpu      NUMBER,
        processor_cpu NUMBER,
        pagefaults    NUMBER
    );



rem
rem  Save Begining Resource Values for a process.
rem

    CREATE TABLE pre_process_stats
    (
        runname       VARCHAR2(20),
        user_cpu      NUMBER,
        priv_cpu      NUMBER,
        processor_cpu NUMBER,
        pagefaults    NUMBER
    );


rem
rem  Save Ending Resource Values for a process.
rem

    CREATE TABLE post_process_stats
    (
        runname       VARCHAR2(20),
        user_cpu      NUMBER,
        priv_cpu      NUMBER,
        processor_cpu NUMBER,
        pagefaults    NUMBER
    );
commit;
set echo off


-------------------------------------------------
                cs_thread.sql
-------------------------------------------------

rem
rem
rem=================================================================+
rem        Copyright (c) 1997  Oracle Corp, Redwood Shores, CA       |
rem                     All Rights Reserved                       |
rem=================================================================+
rem FILENAME
rem    cs_thread.sql
rem DESCRIPTION
rem    Create Table for thread statistics
rem=================================================================
rem Usage: sqlplus tpcc/tpcc @cs_thread.sql

connect tpcc/tpcc
set echo on

DROP TABLE thread_stats;
DROP TABLE pre_thread_stats;
DROP TABLE post_thread_stats;

rem
rem  Resource usage for a thread.
rem

    CREATE TABLE thread_stats
    (
        runname       VARCHAR2(20),
         thread_id   VARCHAR2(10),
        user_cpu      NUMBER,
        priv_cpu      NUMBER,
        processor_cpu NUMBER,
        ctxswitch     NUMBER
    );



rem
rem  Save Begining Resource Values for a thread.
rem

    CREATE TABLE pre_thread_stats
    (
        runname       VARCHAR2(20),
         thread_id   VARCHAR2(10),
        user_cpu      NUMBER,
        priv_cpu      NUMBER,
        processor_cpu NUMBER,
        ctxswitch     NUMBER
    );


rem
rem  Save Ending Resource Values for a thread.
rem

    CREATE TABLE post_thread_stats
    (
        runname       VARCHAR2(20),
         thread_id   VARCHAR2(10),
        user_cpu      NUMBER,
        priv_cpu      NUMBER,
        processor_cpu NUMBER,
        ctxswitch     NUMBER
    );
commit;
set echo off


-------------------------------------------------
                cs_tpcc.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

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;


-------------------------------------------------
                dml.sql
-------------------------------------------------

REM==================================================================+
REM        Copyright (c) 1996  Oracle Corp, Redwood Shores, CA       |
REM                   OPEN SYSTEMS PERFORMANCE GROUP                 |
REM                        All Rights Reserved                       |
REM==================================================================+
REM FILENAME
REM     dml.sql
REM DESCRIPTION
REM     Disable table locks for TPC-C tables.
REM USAGE
REM     sqlplus tpcc/tpcc dml.sql
REM===================================================================

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;

set echo off;

connect $oracle_dba/$oracle_dba_password;


-------------------------------------------------
                driver.sh
-------------------------------------------------

#!/bin/sh

. ./stepenv.sh

if expr $# \< 1 > /dev/null; then
  echo "$0 <starting stepname> <optional: only>"
  echo OR use:
  echo "$0 buildcreate  - to build the database creation scripts"
  echo "$0 create       - to create the database (after buildcreate)"
  echo "$0 steps        - to list individual steps"
  exit 1
fi

if expr x$1 = xsteps > /dev/null; then
  echo stepnames are from creation scripts: $tpcc_create_steps
  echo or running steps: $tpcc_steps
  echo "use the 'only' option to only do that step (otherwise all steps after will also be executed.)"
  echo "   (e.g.  $0 listfiles only)"
  echo "use the 'through' option to do a sequence of steps (inclusively.)"
  echo "   (e.g.  $0 shutdowndb through startupdb-p_build)"
  exit 1
fi

startstep=$1
controlcmd=$2
endstep=$3

# Aliases for special steps
if test $startstep = buildcreate; then
  startstep=`echo $tpcc_create_steps | cut -d' ' -f1`
fi

if test $startstep = create; then
  startstep=`echo $tpcc_steps | cut -d' ' -f1`
fi

if test "x$controlcmd" = x; then
  endstep=
  # Since endstep is null it won't match any other steps, so we keep going.
elif test "x$controlcmd" = xonly; then
  controlcmd=only
  # this is allowed
elif test "x$controlcmd" = xthrough; then
  actualstep=f
  for step in $tpcc_create_steps $tpcc_steps ; do
    if test "x$step" = "x$endstep"; then
      actualstep=t
    fi
  done
  if test $actualstep = f; then
    echo "Invalid step $endstep.  Use $0 steps  to show steps."
    exit 1
  fi
else
  echo "Invalid syntax.  Use $0 by itself for help."
  exit 1
fi

echo Starting from step: $startstep

dostep=f
for step in $tpcc_create_steps $tpcc_steps ; do
  if expr $step = $startstep > /dev/null; then
    dostep=t
  fi

  if expr $dostep = t > /dev/null; then
    echo $step
    cd $tpcc_bench
    $tpcc_scripts/`echo $step | cut -d- -f1`.sh `echo $step | sed -e's/-*$/-/' | cut -d- -f2- | sed -e's/-/ /g'`
    lasterror=$?
    cd $tpcc_bench
    if test -n "`find $tpcc_bench/scripts -name '*.log'`"; then
     mv *.log `find $tpcc_bench/scripts -name '*.log'` $tpcc_bench/log/
    else
      mv *.log $tpcc_bench/log/
    fi

    if expr $lasterror != 0 > /dev/null; then
      if expr $lasterror != 99 > /dev/null; then
        echo Step $step failed.  Stopping driver.
        exit 1
      else
        echo Step $step has completed and requested stop.  Stopping driver.
        exit 0
      fi
    fi
    if test "x$controlcmd" = xonly; then
      exit 0
    fi
    if test "x$endstep" = "x$step"; then
      echo The driver reached the last desired step.  Stopping driver.
      exit 0
    fi
  fi
done

if expr $dostep = f > /dev/null; then
  echo No such step: $1
fi
```

```
-------------------------------------------------
                extent.sql
-------------------------------------------------
```

```
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 'sys/change_on_install as sysdba' @extent
REM==================================================================*/
      set space 2
      set pagesize 2000
      set echo off
      set termout off
      set verify off
      set feedback off
      spool extent.rpt
      select substr(e.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 * t.block_size / 1048576 size_MB
      from   dba_extents e, dba_tablespaces t
      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 e.tablespace_name <> 'SYSTEM'
        AND e.tablespace_name = t.tablespace_name
      order by e.tablespace_name, segment_name, extent_id, file_id;

      select substr(e.tablespace_name,1,8) tspace,
        substr(segment_name,1,11) segment,
        sum(blocks) tot_blk, sum(blocks) * t.block_size / 1048576 size_MB
      from   dba_extents e, dba_tablespaces t
      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 e.tablespace_name <> 'SYSTEM'
        AND e.tablespace_name = t.tablespace_name
      group by e.tablespace_name, segment_name, t.block_size
      order by e.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 'sys/change_on_install as sysdba' @freeext
REM==================================================================*/
      set space 2
      set pagesize 2000
      set echo off
      set termout off
      set verify off
      set feedback off
      spool freeextent.rpt
      select substr(e.tablespace_name,1,8) tspace, file_id, block_id, blocks,
             blocks * t.block_size / 1048576 size_MB
      from dba_free_space e, dba_tablespaces t
      where e.tablespace_name = t.tablespace_name
      order by e.tablespace_name, file_id, block_id;

      select substr(e.tablespace_name,1,8) tspace, sum(blocks) tot_blk,
             sum(blocks) * t.block_size / 1048576 size_MB
      from dba_free_space e, dba_tablespaces t
      where e.tablespace_name = t.tablespace_name
      group by e.tablespace_name, t.block_size
      order by e.tablespace_name;
```

```
-------------------------------------------------
                loadcust.sh
-------------------------------------------------
```

```
#created automatically by /home/weshi/tpcc10800/scripts/evenload.sh Fri May 16 10:31:48 PDT 
2003
rm loadcust*.log
cd $tpcc_bench
allprocs=
$tpcc_load -M 10800 -c  -b 1 -e 1350 >> loadcust0.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -c  -b 1351 -e 2700 >> loadcust1.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -c  -b 2701 -e 4050 >> loadcust2.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -c  -b 4051 -e 5400 >> loadcust3.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -c  -b 5401 -e 6750 >> loadcust4.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -c  -b 6751 -e 8100 >> loadcust5.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -c  -b 8101 -e 9450 >> loadcust6.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -c  -b 9451 -e 10800 >> loadcust7.log 2>&1 &
allprocs="$allprocs ${!}"
error=0
for curproc in $allprocs; do
  wait $curproc
  error=`expr $? + $error`
done
exit `expr $error != 0`
```

```
-------------------------------------------------
                loaddist.sh
-------------------------------------------------
```

```
cd $tpcc_bench
$tpcc_load -M $tpcc_scale -d > loaddist.log 2>&1
```

```
-------------------------------------------------
                loadfixordrordl.sh
-------------------------------------------------
```

```
#created automatically by /home/weshi/tpcc10800/scripts/evenload.sh Fri May 16 10:31:50 PDT 
2003
date
rm loadfixordrordl*.log
cd $tpcc_bench
allprocs=
$tpcc_updateordrordl -M 10800 -o  -b 1 -e 1350 >> loadfixordrordl0.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_updateordrordl -M 10800 -o  -b 1351 -e 2700 >> loadfixordrordl1.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_updateordrordl -M 10800 -o  -b 2701 -e 4050 >> loadfixordrordl2.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_updateordrordl -M 10800 -o  -b 4051 -e 5400 >> loadfixordrordl3.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_updateordrordl -M 10800 -o  -b 5401 -e 6750 >> loadfixordrordl4.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_updateordrordl -M 10800 -o  -b 6751 -e 8100 >> loadfixordrordl5.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_updateordrordl -M 10800 -o  -b 8101 -e 9450 >> loadfixordrordl6.log 2>&1 &
allprocs="$allprocs ${!}"
```

```
$tpcc_updateordrordl -M 10800 -o  -b 9451 -e 10800 >> loadfixordrordl7.log 2>&1 &
allprocs="$allprocs ${!}"
error=0
for curproc in $allprocs; do
  wait $curproc
  error=`expr $? + $error`
done
date
exit `expr $error != 0`


-------------------------------------------------
                loadhist.sh
-------------------------------------------------

#created automatically by /home/weshi/tpcc10800/scripts/evenload.sh Fri May 16 10:31:46 PDT
2003
rm loadhist*.log
cd $tpcc_bench
allprocs=
$tpcc_load -M 10800 -h  -b 1 -e 1350 >> loadhist0.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -h  -b 1351 -e 2700 >> loadhist1.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -h  -b 2701 -e 4050 >> loadhist2.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -h  -b 4051 -e 5400 >> loadhist3.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -h  -b 5401 -e 6750 >> loadhist4.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -h  -b 6751 -e 8100 >> loadhist5.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -h  -b 8101 -e 9450 >> loadhist6.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -h  -b 9451 -e 10800 >> loadhist7.log 2>&1 &
allprocs="$allprocs ${!}"
error=0
for curproc in $allprocs; do
  wait $curproc
  error=`expr $? + $error`
done
exit `expr $error != 0`


-------------------------------------------------
                loaditem.sh
-------------------------------------------------

cd $tpcc_bench
$tpcc_load -M $tpcc_scale -i > loaditem.log 2>&1


-------------------------------------------------
                loadnord.sh
-------------------------------------------------

#created automatically by /home/weshi/tpcc10800/scripts/evenload.sh Fri May 16 10:31:47 PDT
2003
rm loadnord*.log
cd $tpcc_bench
allprocs=
$tpcc_load -M 10800 -n  -b 1 -e 1350 >> loadnord0.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -n  -b 1351 -e 2700 >> loadnord1.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -n  -b 2701 -e 4050 >> loadnord2.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -n  -b 4051 -e 5400 >> loadnord3.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -n  -b 5401 -e 6750 >> loadnord4.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -n  -b 6751 -e 8100 >> loadnord5.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -n  -b 8101 -e 9450 >> loadnord6.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -n  -b 9451 -e 10800 >> loadnord7.log 2>&1 &
allprocs="$allprocs ${!}"
error=0
for curproc in $allprocs; do
  wait $curproc
  error=`expr $? + $error`
done
exit `expr $error != 0`


-------------------------------------------------
                loadordrordl.sh
-------------------------------------------------

#created automatically by /home/weshi/tpcc10800/scripts/evenload.sh Fri May 16 10:31:48 PDT
2003
rm loadordrordl*.log
cd $tpcc_bench
allprocs=
$tpcc_load -M 10800 -o $tpcc_disks_location/dummy0.dat -b 1 -e 1350 >> loadordrordl0.log 2>&1
&
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -o $tpcc_disks_location/dummy1.dat -b 1351 -e 2700 >> loadordrordl1.log
2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -o $tpcc_disks_location/dummy2.dat -b 2701 -e 4050 >> loadordrordl2.log
2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -o $tpcc_disks_location/dummy3.dat -b 4051 -e 5400 >> loadordrordl3.log
2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -o $tpcc_disks_location/dummy4.dat -b 5401 -e 6750 >> loadordrordl4.log
2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -o $tpcc_disks_location/dummy5.dat -b 6751 -e 8100 >> loadordrordl5.log
2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -o $tpcc_disks_location/dummy6.dat -b 8101 -e 9450 >> loadordrordl6.log
2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -o $tpcc_disks_location/dummy7.dat -b 9451 -e 10800 >> loadordrordl7.log
2>&1 &
allprocs="$allprocs ${!}"
error=0
for curproc in $allprocs; do
  wait $curproc
  error=`expr $? + $error`
done
exit `expr $error != 0`


-------------------------------------------------
                loadstok.sh
-------------------------------------------------

#created automatically by /home/weshi/tpcc10800/scripts/evenload.sh Fri May 16 10:31:49 PDT
2003
rm loadstok*.log
cd $tpcc_bench
allprocs=
$tpcc_load -M 10800 -S  -j 1 -k 12500 >> loadstok0.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -S  -j 12501 -k 25000 >> loadstok1.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -S  -j 25001 -k 37500 >> loadstok2.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -S  -j 37501 -k 50000 >> loadstok3.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -S  -j 50001 -k 62500 >> loadstok4.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -S  -j 62501 -k 75000 >> loadstok5.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -S  -j 75001 -k 87500 >> loadstok6.log 2>&1 &
allprocs="$allprocs ${!}"
$tpcc_load -M 10800 -S  -j 87501 -k 100000 >> loadstok7.log 2>&1 &
allprocs="$allprocs ${!}"
error=0
for curproc in $allprocs; do
  wait $curproc
  error=`expr $? + $error`
done
exit `expr $error != 0`


-------------------------------------------------
                loadware.sh
-------------------------------------------------

cd $tpcc_bench
$tpcc_load -M $tpcc_scale -w > loadware.log 2>&1


-------------------------------------------------
                orst_cre.sql
-------------------------------------------------

rem
rem
rem ==================================================================+
rem         Copyright (c) 1996  Oracle Corp, Redwood Shores, CA       |
rem                   OPEN SYSTEMS PERFORMANCE GROUP                  |
rem                         All Rights Reserved                       |
rem ==================================================================+
rem  FILENAME
rem       orst_cre.sql
rem  DESCRIPTION
rem       Drop and Create Tables for Oracle Statistics
rem ===================================================================*/
rem
rem  Usage: sqlplus internal/internal @orst_cre.sql
rem
        connect $oracle_dba/$oracle_dba_password;
        SET ECHO ON;
rem        SET TERMOUT OFF;

        DROP TABLE save_sysstat;
        DROP TABLE save_latch;
        DROP TABLE save_rollstat;
        DROP TABLE save_filestat;
        DROP TABLE save_rowcache;
        DROP TABLE save_parameter;
        DROP TABLE save_wait;
        DROP TABLE save_fwait;
        DROP TABLE save_event;
        DROP TABLE save_lockact;
        DROP TABLE save_fping;
        DROP TABLE save_fping2;
        DROP TABLE save_ping;
        DROP TABLE save_ping2;
        DROP TABLE save_blkping;
        DROP TABLE save_blkping2;
        DROP TABLE save_kclwait;
        DROP TABLE save_sqlarea;
        DROP TABLE save_time;
        DROP TABLE save_dfile;
        DROP TABLE save_rsrc;
        DROP TABLE save_circuit;
    DROP TABLE save_dispatcher;
    DROP TABLE save_queue;
    DROP TABLE save_server;

        DROP TABLE tmp_sysstat;
        DROP TABLE tmp_latch;
        DROP TABLE tmp_filestat;
        DROP TABLE tmp_rollstat;
        DROP TABLE tmp_rowcache;
        DROP TABLE tmp_wait;
        DROP TABLE tmp_fwait;
        DROP TABLE tmp_event;
        DROP TABLE tmp_fping;
        DROP TABLE tmp_fping2;
        DROP TABLE tmp_kclwait;
        DROP TABLE blockclass;
        DROP TABLE tmp_lockact;
        DROP TABLE zero_lockact;
        DROP TABLE tmp_sqlarea;
        DROP TABLE tmp_time;
        DROP TABLE tmp_circuit;
    DROP TABLE tmp_dispatcher;
    DROP TABLE tmp_queue;
    DROP TABLE tmp_server;
    DROP TABLE tmp_rsrc;

rem
rem  save_sysstat corresponds to v$sysstat and v$statname
rem
        CREATE TABLE save_sysstat
          (
            hid             NUMBER,
            run             NUMBER,
        name      VARCHAR2(64),
            statistic#      NUMBER,
        value      NUMBER
        );

rem
rem  save_latch corresponds to v$latch and v$latchname
rem

        CREATE TABLE save_latch
          (
            hid              NUMBER,
            run              NUMBER,
            name              VARCHAR2(64),
        latch#          NUMBER,
        gets        NUMBER,
        misses     NUMBER,
        sleeps      NUMBER,
        immediate_gets     NUMBER,
        immediate_misses NUMBER
        );

rem
rem  save_rollstat corresponds to v$rollstat and v$rollname
rem
        CREATE TABLE save_rollstat
          (
            hid             NUMBER,
            run             NUMBER,
            name            VARCHAR2(30),
            USN             NUMBER,
            EXTENTS         NUMBER,
            RSSIZE          NUMBER,
            WRITES          NUMBER,
            XACTS           NUMBER,
            GETS            NUMBER,
            WAITS           NUMBER,
            OPTSIZE         NUMBER,
            HWMSIZE         NUMBER,
            SHRINKS         NUMBER,
            WRAPS           NUMBER,
            EXTENDS         NUMBER,
            AVESHRINK       NUMBER,
            AVEACTIVE       NUMBER
        );

rem
rem  save_filestat corresponds to v$filestat and v$dbfile;
rem
        CREATE TABLE save_filestat
          (
            hid             NUMBER,
            run             NUMBER,
            FILE#           NUMBER,
            PHYRDS          NUMBER,
            PHYWRTS         NUMBER,
            PHYBLKRD        NUMBER,
            PHYBLKWRT       NUMBER,
            READTIM         NUMBER,
            WRITETIM        NUMBER,
            NAME            VARCHAR2(257)
          );

rem
rem  save_rowcache corresponds to v$rowcache
rem
          CREATE TABLE save_rowcache
          (
            hid             NUMBER,
            run             NUMBER,
            cache#          NUMBER,
            type            VARCHAR2(11),
            subordinate#    NUMBER,
            parameter       VARCHAR2(32),
            count           NUMBER,
            usage           NUMBER,
            fixed           NUMBER,
            gets            NUMBER,
            getmisses       NUMBER,
            scans           NUMBER,
            scanmisses      NUMBER,
            scancompletes   NUMBER,
            modifications   NUMBER,
            flushes         NUMBER
          );

rem
rem Create table to hold values in v$parameter
rem

          CREATE TABLE save_parameter
          (
            hid             NUMBER,
            run             NUMBER,
            NAME            VARCHAR2(64),
            VALUE           VARCHAR2(512)
          );

rem
rem  save_wait corresponds to v$wait_stat
rem
          CREATE TABLE save_wait
          (
            hid             NUMBER,
            run             NUMBER,
            class           VARCHAR2(18),
            count           NUMBER,
            time            NUMBER
          );

rem
rem  save_fwait corresponds to X$KCBFWAIT
rem
          CREATE TABLE save_fwait
          (
            hid             NUMBER,
            run             NUMBER,
            addr            VARCHAR2(20),
            indx            NUMBER,
        count           NUMBER,
        time            NUMBER
    );

rem
rem  save_event corresponds to v$system_event
rem
    CREATE TABLE save_event
    (
        hid             NUMBER,
        run             NUMBER,
        event           VARCHAR2(64),
        total_waits     NUMBER,
        time_waited     NUMBER,
        average_wait    NUMBER
    );

rem
rem  save_lockact corresponds to v$lock_activity
rem
    CREATE TABLE save_lockact
    (
        hid             NUMBER,
        run             NUMBER,
        from_val        VARCHAR2(4),
        to_val          VARCHAR2(4),
        action_val      VARCHAR2(51),
        counter         NUMBER
    );

rem
rem  save_fping corresponds to file_ping
rem
    CREATE TABLE save_fping
    (
        hid             NUMBER,
        run             NUMBER,
        file_id         NUMBER,
        file_name       VARCHAR2(257),
        ts_name         VARCHAR2(30),
        x_to_n          NUMBER
    );

rem
rem  save_fping2 corresponds to file_ping with extended ping stats
rem
    CREATE TABLE save_fping2
    (
        hid             NUMBER,
        run             NUMBER,
        file_id         NUMBER,
        file_name       VARCHAR2(257),
        ts_name         VARCHAR2(30),
        x2n             NUMBER,
        x2s             NUMBER,
        x2ssx           NUMBER,
        s2n             NUMBER,
        cic             NUMBER,
        crt             NUMBER,
        hping           NUMBER,
        sping           NUMBER
    );

rem
rem  save_ping corresponds to v$ping
rem
    CREATE TABLE save_ping
    (
        hid             NUMBER,
        run             NUMBER,
        tablespace_name VARCHAR2(30),
        file_name       VARCHAR2(257),
        kind            VARCHAR2(12),
        status          VARCHAR2(4),
        xnc             NUMBER
    );

rem
rem  save_ping2 corresponds to v$ping with extended ping stats
rem
    CREATE TABLE save_ping2
    (
        hid             NUMBER,
        run             NUMBER,
        tablespace_name VARCHAR2(30),
        file#           NUMBER,
        kind            VARCHAR2(12),
        status          VARCHAR2(4),
        hping           NUMBER,
        sping           NUMBER
    );

rem
rem  save_blkping corresponds to v$ping
rem
    CREATE TABLE save_blkping
    (
        hid             NUMBER,
        run             NUMBER,
        tablespace_name VARCHAR2(30),
        file_name       VARCHAR2(257),
        kind            VARCHAR2(12),
        block#          NUMBER,
        status          VARCHAR2(4),
        xnc             NUMBER
    );

rem
rem  save_blkping2 corresponds to v$ping with extended ping stats
rem
    CREATE TABLE save_blkping2
    (
        hid             NUMBER,
        run             NUMBER,
        tablespace_name VARCHAR2(30),
        file#           NUMBER,
        kind            VARCHAR2(12),
        block#          NUMBER,
        status          VARCHAR2(4),
        hping           NUMBER,
        sping           NUMBER,
        lock_element_addr  RAW(4)
    );

rem
rem  save_kclwait corresponds to v$kclwait
rem
    CREATE TABLE save_kclwait
    (
        hid             NUMBER,
        run             NUMBER,
        indx            NUMBER,
        pings           NUMBER,
        hpings          NUMBER,
        spings          NUMBER,
        wpings          NUMBER
    );

rem
rem  save_sqlarea corresponds to v$sqlarea
rem
    CREATE TABLE save_sqlarea
    (
        hid             NUMBER,
        run             NUMBER,
        sql_text        VARCHAR2(1000),
        executions      NUMBER,
        buffer_gets     NUMBER,
        disk_reads      NUMBER,
        serializable_aborts      NUMBER
    );

rem
rem  save_time records duration of each run
rem
    CREATE TABLE save_time
    (
        hid             NUMBER,
        run             NUMBER,
        rtime           NUMBER
    );

rem
rem  save_dfile maps oracle datafile to physical disks and nodes
rem
    CREATE TABLE save_dfile
    (
        file#           NUMBER,
        group#          NUMBER,
        gname           VARCHAR2(20),
        node#           NUMBER,
        nname           VARCHAR2(20),
        disk#           NUMBER,
        dname           VARCHAR2(20)
    );

rem
rem  save_rsrc corresponds to v$rsrc_consumer_group
rem
    CREATE TABLE save_rsrc
    (
        hid             NUMBER,
            run            NUMBER,
            NAME               VARCHAR2(32),
            ACTIVE_SESSIONS     NUMBER,
            EXECUTION_WAITERS   NUMBER,
            REQUESTS           NUMBER,
            CPU_WAIT_TIME       NUMBER,
            CPU_WAITS          NUMBER,
            CONSUMED_CPU_TIME   NUMBER,
            YIELDS             NUMBER,
            SESSIONS_QUEUED     NUMBER
          );

        CREATE TABLE save_circuit
        (
        hid        NUMBER,
        run        NUMBER,
        circuit        raw(4),
        msg0    NUMBER,
        msg1    NUMBER,
        msgs    NUMBER,
        bytes    NUMBER,
        breaks        NUMBER
        );

        CREATE TABLE save_dispatcher
        (
        hid        NUMBER,
        run        NUMBER,
        paddr    raw(4),
        msgs    NUMBER,
        bytes    NUMBER,
        breaks        NUMBER,
        idle        NUMBER,
        busy        NUMBER
        );

        CREATE TABLE save_server
        (
        hid        NUMBER,
        run        NUMBER,
        name     VARCHAR2(20),
        msgs    NUMBER,
        bytes    NUMBER,
        breaks        NUMBER,
        idle        NUMBER,
        busy        NUMBER,
        requests NUMBER
        );

        CREATE TABLE save_queue
        (
        hid        NUMBER,
        run        NUMBER,
        paddr    raw(4),
        wait      NUMBER,
        totalq    NUMBER
        );

rem
rem  tmp_sysstat corresponds to v$sysstat
rem
        CREATE TABLE tmp_sysstat
          (
        hid        NUMBER,
        state      VARCHAR2(10),
        statistic#NUMBER,
        value     NUMBER
        );

rem
rem  tmp_latch corresponds to v$latch
rem

        CREATE TABLE tmp_latch
          (
        hid        NUMBER,
        state       VARCHAR2(10),
        latch#          NUMBER,
        gets        NUMBER,
        misses     NUMBER,
        sleeps     NUMBER,
        immediate_gets     NUMBER,
        immediate_misses NUMBER
        );

rem
rem  tmp_filestat corresponds to v$filestat
rem
          CREATE TABLE tmp_filestat
          (
        hid        NUMBER,
          state      VARCHAR2(10),
              FILE#          NUMBER,
              PHYRDS         NUMBER,
              PHYWRTS        NUMBER,
              PHYBLKRD       NUMBER,
              PHYBLKWRT      NUMBER,
              READTIM        NUMBER,
              WRITETIM       NUMBER
            );

rem
rem  tmp_rollstat corresponds to v$rollstat
rem
            CREATE TABLE tmp_rollstat
            (
          hid        NUMBER,
          state         VARCHAR2(10),
              USN            NUMBER,
              EXTENTS        NUMBER,
              RSSIZE         NUMBER,
              WRITES         NUMBER,
              XACTS          NUMBER,
              GETS           NUMBER,
              WAITS          NUMBER,
              OPTSIZE        NUMBER,
              HWMSIZE        NUMBER,
              SHRINKS        NUMBER,
              WRAPS          NUMBER,
              EXTENDS        NUMBER,
              AVESHRINK      NUMBER,
              AVEACTIVE      NUMBER
            );

rem
rem  tmp_rowcache corresponds to v$rowcache
rem
            CREATE TABLE tmp_rowcache
            (
          hid        NUMBER,
              state          VARCHAR2(10),
              cache#         NUMBER,
              type           VARCHAR2(11),
              subordinate#   NUMBER,
              parameter      VARCHAR2(32),
              count          NUMBER,
              usage          NUMBER,
              fixed          NUMBER,
              gets           NUMBER,
              getmisses      NUMBER,
              scans          NUMBER,
              scanmisses     NUMBER,
              scancompletes   NUMBER,
              modifications  NUMBER,
              flushes        NUMBER
            );

rem
rem  tmp_wait corresponds to v$wait_stat
rem
            CREATE TABLE tmp_wait
            (
              hid            NUMBER,
              state          VARCHAR2(10),
              class          VARCHAR2(18),
              count          NUMBER,
              time           NUMBER
            );

rem
rem  tmp_fwait corresponds to X$KCBFWAIT
rem
            CREATE TABLE tmp_fwait
            (
              hid            NUMBER,
              state          VARCHAR2(10),
              addr           VARCHAR2(20),
              indx           NUMBER,
              count          NUMBER,
              time           NUMBER
            );

rem
rem  tmp_event corresponds to v$system_event
rem
            CREATE TABLE tmp_event
            (
              hid            NUMBER,
              state          VARCHAR2(10),
              event          VARCHAR2(64),
              total_waits     NUMBER,
              time_waited     NUMBER,
            average_wait    NUMBER
        );

rem
rem  tmp_fping corresponds to file_ping
rem
        CREATE TABLE tmp_fping
        (
            hid             NUMBER,
            state           VARCHAR2(10),
            file_id         NUMBER,
            file_name       VARCHAR2(257),
            ts_name         VARCHAR2(30),
            x_to_n          NUMBER
        );

rem
rem  tmp_fping2 corresponds to file_ping with extended ping stats
rem
        CREATE TABLE tmp_fping2
        (
            hid             NUMBER,
            state           VARCHAR2(10),
            file_id         NUMBER,
            file_name       VARCHAR2(257),
            ts_name         VARCHAR2(30),
            x2n             NUMBER,
            x2s             NUMBER,
            x2ssx           NUMBER,
            s2n             NUMBER,
            cic             NUMBER,
            crt             NUMBER,
            hping           NUMBER,
            sping           NUMBER
        );

rem
rem  tmp_kclwait corresponds to v$kclwait
rem
        CREATE TABLE tmp_kclwait
        (
            hid             NUMBER,
            state           VARCHAR2(10),
            indx            NUMBER,
            pings           NUMBER,
            hpings          NUMBER,
            spings          NUMBER,
            wpings          NUMBER
        );

rem
rem  blockclass contains mapping from KCBC index to text
rem
      CREATE TABLE blockclass
        (
            indx            NUMBER,
            name            VARCHAR2(24)
        );

  INSERT INTO blockclass VALUES (0, 'NONE');
  INSERT INTO blockclass VALUES (1, 'DATA');
  INSERT INTO blockclass VALUES (2, 'SORT');
  INSERT INTO blockclass VALUES (3, 'SAVE UNDO');
  INSERT INTO blockclass VALUES (4, 'SEG HDR');
  INSERT INTO blockclass VALUES (5, 'SAVE UNDO SEG HDR');
  INSERT INTO blockclass VALUES (6, 'FREE LIST|EXTENT MAP');
  INSERT INTO blockclass VALUES (7, 'UNDO HDR');
  INSERT INTO blockclass VALUES (8, 'UNDO');
  INSERT INTO blockclass VALUES (9, 'UNDO HDR');
  INSERT INTO blockclass VALUES (10, 'UNDO');

rem Are there more...???

rem
rem  tmp_lockact corresponds to v$lock_activity
rem
        CREATE TABLE tmp_lockact
        (
            hid             NUMBER,
            state           VARCHAR2(10),
            from_val        VARCHAR2(4),
            to_val          VARCHAR2(4),
            action_val      VARCHAR2(51),
            counter         NUMBER
        );

rem
rem  zero_lockact corresponds to v$lock_activity with no activity
rem
        CREATE TABLE zero_lockact
        (
            from_val        VARCHAR2(4),
            to_val          VARCHAR2(4),
            action_val      VARCHAR2(51),
            counter         NUMBER
        );

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('NULL', 'S',    'Lock buffers for read',                              0);

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('NULL', 'X',    'Lock buffers for write',                             0);

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('S',    'NULL', 'Make buffers CR (no write)',                         0);

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('S',    'X',    'Upgrade read lock to write',                         0);

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('X',    'NULL', 'Make buffers CR (write dirty buffers)',              0);

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('X',    'S',    'Downgrade write lock to read (write dirty buffers)', 0);

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('X',    'SSX',  'Write transaction table/undo blocks',                0);

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('SSX',  'NULL', 'Transaction table/undo blocks (write dirty buffers)', 0);

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('SSX',  'S',    'Make transaction table/undo block available share',  0);

  INSERT INTO zero_lockact (from_val, to_val, action_val, counter) VALUES
    ('SSX',  'X',    'Rearm transaction table write mechanism',            0);

rem
rem  tmp_sqlarea corresponds to v$sqlarea
rem
        CREATE TABLE tmp_sqlarea
        (
            hid             NUMBER,
            state           VARCHAR2(10),
            sql_text        VARCHAR2(1000),
            executions      NUMBER,
            buffer_gets     NUMBER,
            disk_reads      NUMBER,
            serializable_aborts      NUMBER
        );

rem
rem  tmp_time records begin and end time
rem
        CREATE TABLE tmp_time
        (
            hid             NUMBER,
            state           VARCHAR2(10),
            timestamp       DATE
        );

      CREATE TABLE tmp_circuit
      (
      hid        NUMBER,
      state      VARCHAR2(10),
      circuit       raw(4),
      msg0       NUMBER,
      msg1       NUMBER,
      msgs       NUMBER,
      bytes      NUMBER,
      breaks        NUMBER
      );

      CREATE TABLE tmp_dispatcher
      (
      hid        NUMBER,
      state      VARCHAR2(10),
         paddr      raw(4),
      msgs       NUMBER,
      bytes      NUMBER,
      breaks        NUMBER,
      idle       NUMBER,
      busy       NUMBER
      );

      CREATE TABLE tmp_server
      (
      hid        NUMBER,
      state      VARCHAR2(10),
      name       VARCHAR2(20),
      msgs       NUMBER,
      bytes      NUMBER,
        breaks        NUMBER,
        idle       NUMBER,
        busy       NUMBER,
        requests NUMBER
        );

        CREATE TABLE tmp_queue
        (
        hid        NUMBER,
        state      VARCHAR2(10),
        paddr      raw(4),
        wait            NUMBER,
        totalq            NUMBER
        );
rem
rem  tmp_rsrc corresponds to v$rsrc_consumer_group
rem
          CREATE TABLE tmp_rsrc
          (
             hid            NUMBER,
             state           VARCHAR2(10),
             NAME                VARCHAR2(32),
             ACTIVE_SESSIONS     NUMBER,
             EXECUTION_WAITERS   NUMBER,
             REQUESTS            NUMBER,
             CPU_WAIT_TIME       NUMBER,
             CPU_WAITS           NUMBER,
             CONSUMED_CPU_TIME   NUMBER,
             YIELDS             NUMBER,
             SESSIONS_QUEUED     NUMBER
          );

  COMMIT;
  SET ECHO OFF;
```

```
-------------------------------------------------
                p_build.ora
-------------------------------------------------

compatible = 10.0.0.0.0
db_name = tpcc
control_files = /home/oracle/dev/raw/control_001
parallel_max_servers = 100
recovery_parallelism = 40
db_files = 524
db_cache_size   = 5000M
db_8k_cache_size  = 512M
db_16k_cache_size  = 7000M
dml_locks = 500
log_buffer = 10485760
processes =1024
sessions = 5000
transactions = 100
shared_pool_size = 500M
cursor_space_for_time = TRUE
db_block_size = 2048
undo_management = auto
undo_retention = 2
UNDO_TABLESPACE = undo_ts
_in_memory_undo = false
```

```
-------------------------------------------------
                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;
```

```
-------------------------------------------------
                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 tpcc/tpcc;
        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;
```

```
--------------------------------------------------
            stepenv.sh
--------------------------------------------------

# forces any env variables we set to be exported
set -a
tpcc_kit=t
tpcc_bench=$PWD
tpcc_scripts=$tpcc_bench/scripts
tpcc_require=$tpcc_scripts/require_vars.sh
tpcc_lcm=$tpcc_scripts/lcm.sh
tpcc_tokilobytes=$tpcc_scripts/tokilobytes.sh
tpcc_fromkilobytes=$tpcc_scripts/fromkilobytes.sh
tpcc_estsize=$tpcc_scripts/estsize.sh
tpcc_notneg=$tpcc_scripts/notneg.sh
tpcc_isneg=$tpcc_scripts/isneg.sh


# need a better way to check for bc, may
# resort to checking each directory in path
# if this doesn't work
#11/7/02 - alex.ni this is causing too many problems
#because systems have bc in some odd place.  typically
#mangled cygwin installs w/ mksnt/cygwin mixes
#if test -x /usr/bin/bc -o -x /bin/bc; then
tpcc_bcexpr=$tpcc_scripts/bcexpr.sh
#else
#tpcc_bcexpr=expr
#fi


# the ksh version is a bit faster, so we want
# to use it if we have ksh.  Otherwise we have
# a compatible version.
if test -x /bin/ksh; then
tpcc_createts=$tpcc_scripts/createts.ksh
else
tpcc_createts=$tpcc_scripts/createts.sh
fi

tpcc_tabledata=$tpcc_scripts/tabledata.sh
tpcc_load=$tpcc_bench/benchrun/bin/tpccload.exe
tpcc_createtablespaces=$tpcc_scripts/createtablespaces.sh

##
tpcc_sqlplus=cat
tpcc_sqlplus_args='/nolog'
tpcc_internal_connect='connect / as sysdba'
tpcc_user_pass='tpcc/tpcc'
tpcc_dba_user_pass='system/manager'
oracle_dba=system
oracle_dba_password=manager
tpcc_sqlplus_args=
tpcc_user_pass=
tpcc_sqlplus=sqlplus
tpcc_user_pass='tpcc/tpcc'

# import options generated by gui
. ${tpcc_bench}/options.sh

#8gb oracle filesize limit (in k)
tpcc_fsize_limit_k=8388608
#2gb - 1k oracle extent limit (in k)
tpcc_extent_limit_k=2097151

# Runlen calculations should be in hours, but
# this was the old calculation, which assumed
# minutes, and also 8 times:
# tpcc_runlen=`$tpcc_bcexpr 8 \* 60 \* $tpcc_runlen`
# we just want to keep the value as it is.

tpcc_system_size=200M
tpcc_logfile_size=`$tpcc_bcexpr 20 + \( $tpcc_scale \)`M
tpcc_undo_size=`$tpcc_bcexpr 2 \* $tpcc_scale`
if test $tpcc_undo_size -gt 8096; then
  tpcc_undo_size=8096
fi
tpcc_undo_size="${tpcc_undo_size}M"

tpcc_undo_bs=8K

tpcc_statspack_size=`$tpcc_bcexpr 1 \* $tpcc_scale`
if test $tpcc_statspack_size -gt 2048; then
  tpcc_statspack_size=2048
fi
tpcc_statspack_size="${tpcc_statspack_size}M"

tpcc_sysaux_size=120M

# fixed table params

#table list (note temp is always at the end since it may use numbers from other tables, and it's not
included in these lists)
tpcc_table_list='ware cust dist hist stok item ordr ordl nord'
tpcc_index_list='iware icust1 icust2 idist istok iitem iordr1 iordr2 iordl inord'
#for these I use average row length, calculated from multi-blocksize stats.
#we figure out how many new rows we will gain in a run (in createtablesspaces.sh)
#and add that much to the base tablespace size.
tpcc_hist_growth=51
tpcc_ordr_growth=35
tpcc_nord_growth=13
#tpcc_ordl_growth=660
tpcc_ordl_growth=900

#i started indices at 1/10th... need an exact figure
tpcc_iordr1_growth=20
tpcc_iordr2_growth=20
tpcc_iordl_growth=66
tpcc_inord_growth=2

tpcc_item_growth=0
tpcc_iitem_growth=0
tpcc_temp_growth=0

tpcc_cust_growth=regular
tpcc_icust1_growth=regular
tpcc_icust2_growth=regular

tpcc_stok_growth=regular
tpcc_istok_growth=regular

tpcc_ware_growth=regular
tpcc_iware_growth=regular

tpcc_dist_growth=regular
tpcc_idist_growth=regular

# minimum size of temp tablespace
tpcc_tempts_min=10240

# for Linux, set appropriate tablespace heuristics
# to set high io tables to have 64 files, and minimize
# others.
if expr $tpcc_os = linux > /dev/null; then
  for table in $tpcc_table_list $tpcc_index_list temp; do
    eval "tpcc_${table}_tsfileinc=1"
  done
  tpcc_os=unix

  tpcc_stok_tsfileinc=64
  tpcc_cust_tsfileinc=64
  tpcc_iordl2_tsfileinc=16
  tpcc_icust2_tsfileinc=16
  tpcc_iordl_tsfileinc=16
else
#in case someone changes out of linux, and the shell is stuck
  for table in $tpcc_table_list $tpcc_index_list temp; do
    eval "tpcc_${table}_tsfileinc="
  done
  tpcc_stok_tsfileinc=
  tpcc_cust_tsfileinc=
  tpcc_iordl2_tsfileinc=
  tpcc_icust2_tsfileinc=
  tpcc_iordl_tsfileinc=
fi


# import local options
. ${tpcc_bench}/localoptions.sh

if expr `echo x$tpcc_no_options` = xt > /dev/null; then
  echo Please modify ${tpcc_bench}/localoptions.sh to configure the generator.
  exit 1
fi
```

```
tpcc_fixordrordl=${tpcc_genscripts_dir}/loadfixordrordl.sh
tpcc_updateordrordl=${tpcc_scripts}/updateordrordl.sh

#tp- get table param.  (that is, $tpcc_tablename_tableparam)
tp(){
  eval echo \""\$tpcc_$1_$2"\"
}

# automatically generated variables
if expr `echo $tpcc_version | cut -b1` = t > /dev/null; then
  tpcc_auto_undo=t
else
  tpcc_auto_undo=f
fi
if expr `echo $tpcc_version | cut -b2` = t > /dev/null; then
  tpcc_autospace_avail=t
else
  tpcc_autospace_avail=f
fi
if expr `echo $tpcc_version | cut -b3` = t > /dev/null; then
  tpcc_queue_avail=t
  tpcc_use_sysaux=t
else
  tpcc_queue_avail=f
  tpcc_use_sysaux=f
fi

# for NT, ORACLE does not like $variables in sql scripts, so we must
# hardcode these things for it.
if test x$tpcc_os = xnt; then
  tpcc_hardcode=t
else
  tpcc_hardcode=f
fi

# if this is unset we need to make sure it's something anyway
if test x$tpcc_defbs = x; then
  tpcc_defbs=2
fi

# used for loading program
if test x$tpcc_hash_overflow = xt; then
  tpcc_hash_overflow=t
else
  unset tpcc_hash_overflow
fi
if test x$tpcc_overflow = xt; then
  tpcc_hash_overflow=t
else
  unset tpcc_hash_overflow
fi

tpcc_create_steps="buildcreatets buildcreatedb \
buildcreatetable-ware buildcreatetable-cust buildcreatetable-dist buildcreatetable-hist buildcreatetable-
stok buildcreatetable-item buildcreatetable-ordr buildcreatetable-ordl buildcreatetable-nord \
buildloadware buildloaddist buildloaditem buildloadhist buildloadnord buildloadordrordl buildloadcust
buildloadstok buildfixoo \
buildcreateindex-iware buildcreateindex-icust1 buildcreateindex-icust2 buildcreateindex-idist
buildcreateindex-istok buildcreateindex-iitem buildcreateindex-iordr1 buildcreateindex-iordr2
buildcreateindex-iordl buildcreateindex-inord \
listfiles
"

tpcc_steps="runsqllocal-createdb shutdowndb startupdb-p_build createuser runscript-createts_temp
assigntemp ddview \
 runscript-createts_ware runsql-createtable_ware runscript-loadware runsql-createindex_iware \
 runscript-createts_cust runsql-createtable_cust runscript-loadcust runsql-createindex_icust1 runsql-
createindex_icust2 \
 runscript-createts_dist runsql-createtable_dist runscript-loaddist runsql-createindex_idist \
 runscript-createts_hist runsql-createtable_hist runscript-loadhist \
 runscript-createts_stok runsql-createtable_stok runscript-loadstok runsql-createindex_istok \
 runscript-createts_item runsql-createtable_item runscript-loaditem runsql-createindex_iitem \
 runscript-createts_ordr runsql-createtable_ordr runsql-createtable_ordl \
 runscript-loadordrordl runsql-createindex_iordr1 runsql-createindex_iordr2 runsql-createindex_iordl
\
 runscript-createts_nord runsql-createtable_nord runscript-loadnord runsql-createindex_inord \
analyze runscript-loadfixordrordl createstats createstoredprocs createspacestats createmisc"

# no longer automatically exports env variables
set +a

# check for problems with configuration
badconf=
for table in $tpcc_table_list; do
  if expr `tp $table imp` = queue > /dev/null; then
    if expr $tpcc_queue_avail = f > /dev/null; then
     echo Table $table may not be a queue, since queues are
     echo are unavailable in the selected Oracle version.
     badconf=t
    fi
  fi
  if expr $tpcc_autospace_avail = f \& `tp $table autospace` = t > /dev/null; then
    echo Table $table may not use bitmapped space management
    echo since it is not available in the selected Oracle version.
    badconf=t
  fi
done

if test -n "$badconf"; then
  exit 1
fi

# make sure we have everything
if $tpcc_require ORACLE_SID \
 tpcc_tokilobytes tpcc_createts tpcc_lcm\
 tpcc_sqlplus tpcc_internal_connect\
 tpcc_np tpcc_cpu tpcc_os tpcc_runlen tpcc_ldrive tpcc_scale tpcc_disks_location tpcc_auto_undo
tpcc_tempts_min\
 tpcc_system_size tpcc_logfile_size\
 tpcc_undo_size tpcc_undo_bs\
 oracle_dba oracle_dba_password tpcc_dba_user_pass
then exit 1; fi


-------------------------------------------------
                 tkvcinin.sql
-------------------------------------------------

-- The initnew package for storing variables used in the
-- New Order anonymous block

CREATE OR REPLACE PACKAGE inittpcc
AS
 TYPE intarray IS TABLE OF INTEGER INDEX BY BINARY_INTEGER;
 TYPE distarray IS TABLE OF VARCHAR(24) INDEX BY BINARY_INTEGER;
 nulldate       DATE;
 TYPE rowidarray IS TABLE OF ROWID INDEX BY PLS_INTEGER;
 s_dist             distarray;
 idx1arr       intarray;
 s_remote        intarray;
 dist                   intarray;
 row_id                 rowidarray;
 cust_rowid             rowid;
 dist_name              VARCHAR2(11);
 ware_name              VARCHAR2(11);
 c_num                  PLS_INTEGER;

 PROCEDURE init_no(idxarr intarray);
 PROCEDURE init_del;
 PROCEDURE init_pay;
END inittpcc;
/
show errors;

CREATE OR REPLACE PACKAGE BODY inittpcc AS
  PROCEDURE init_no (idxarr  intarray)
  IS
  BEGIN
        -- initialize null date
   nulldate := TO_DATE('01-01-1811', 'MM-DD-YYYY');
    idx1arr := idxarr;
  END init_no;

  PROCEDURE init_del
  IS
  BEGIN
    FOR i IN 1 .. 10 LOOP
      dist(i) := i;
    END LOOP;
  END init_del;

  PROCEDURE init_pay IS
  BEGIN
    NULL;
  END init_pay;

END inittpcc;
/
show errors
exit


-------------------------------------------------
                 tpcc.h
-------------------------------------------------


/*
 * $Header: tpcc.h 7030100.1 95/07/19 15:10:55 plai Generic<base> $ Copyr (c) 1993 Oracle
 */
/*==================================================================+
 |        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA      |
|                  OPEN SYSTEMS PERFORMANCE GROUP                |
|                      All Rights Reserved                       |
+==================================================================+
| FILENAME
|    tpcc.h
| DESCRIPTION
|    Include file for TPC-C benchmark programs.
+==================================================================*/

#ifndef TPCC_H
#define TPCC_H

#ifndef FALSE
# define FALSE 0
#endif

#ifndef TRUE
# define TRUE 1
#endif

#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

#ifndef boolean
#define boolean int
#endif

#include <oratypes.h>
#include <oci.h>
#include <ocidfn.h>
/*
#ifdef __STDC__
#include "ociapr.h"
#else
#include "ocikpr.h"
#endif
*/

typedef struct cda_def csrdef;
typedef struct cda_def ldadef;


/* TPC-C transaction functions */

extern int TPCinit ();
extern int TPCnew ();
extern int TPCpay ();
extern int TPCord ();
extern int TPCdel ();
extern int TPCsto ();
extern void TPCexit ();
extern int TPCdumpinit ();
extern void TPCdumpnew ();
extern void TPCdumppay ();
extern void TPCdumpord ();
extern void TPCdumpdel ();
extern void TPCdumpsto ();
extern void TPCdumpexit ();
extern void userlog(char* fmtp, ...);


/* Error codes */

#define RECOVERR -10
#define IRRECERR -20
#define NOERR    111
#define DEL_ERROR -666
#define DEL_DATE_LEN 7
#define NDISTS 10
#define NITEMS 15
#define SQL_BUF_SIZE 8192

#define FULLDATE "dd-mon-yy.hh24:mi:ss"
#define SHORTDATE "dd-mm-yyyy"



#define DELRT 80.0

extern int tkvcninit ();
extern int tkvcpinit ();
extern int tkvcoinit ();
extern int tkvcdinit ();
extern int tkvcsinit ();

extern int tkvcn ();
extern int tkvcp ();
extern int tkvco ();
extern int tkvcd ();
extern int tkvcs ();

extern void tkvcndone ();
extern void tkvcpdone ();
extern void tkvcodone ();
extern void tkvcddone ();
extern void tkvcsdone ();

extern int tkvcss (); /* for alter session to get memory size and trace */
extern boolean multitranx;
extern int ord_init;


extern void errrpt ();
extern int ocierror(char *fname, int lineno,OCIError *errhp, sword status);
extern int sqlfile(char *fname, text *linebuf);

extern FILE *lfp;
extern FILE *fopen ();
extern int proc_no;
extern int doid[];

extern int execstatus;
extern int errcode;

extern OCIEnv *tpcenv;
extern OCIServer *tpcsrv;
extern OCIError *errhp;
extern OCISvcCtx *tpcsvc;
extern OCISession *tpcusr;
extern OCIStmt *curntest;
/* The bind and define handles for each transaction are
   included in their respective header files. */



/* for stock-level transaction */

extern int w_id;
extern int d_id;
extern int c_id;
extern int threshold;
extern int low_stock;

/* for delivery transaction */

extern int del_o_id[10];
extern int carrier_id;
extern int retries;

/* for order-status transaction */

extern int bylastname;
extern char c_last[17];
extern char c_first[17];
extern char c_middle[3];
extern double c_balance;
extern int o_id;
extern text o_entry_d[20];
extern int o_carrier_id;
extern int o_ol_cnt;
extern int ol_supply_w_id[15];
extern int ol_i_id[15];
extern int ol_quantity[15];
extern int ol_amount[15];
ub4    ol_del_len[15];
extern text ol_delivery_d[15][11];
/* xnie - begin */
extern OCIRowid *o_rowid;
/* xnie - end */

/* for payment transaction */

extern int c_w_id;
extern int c_d_id;
extern int h_amount;
extern char w_street_1[21];
extern char w_street_2[21];
extern char w_city[21];
extern char w_state[3];
extern char w_zip[10];
extern char d_street_1[21];
extern char d_street_2[21];
extern char d_city[21];
extern char d_state[3];
extern char d_zip[10];
extern char c_street_1[21];
extern char c_street_2[21];
extern char c_city[21];
extern char c_state[3];
extern char c_zip[10];
extern char c_phone[17];
extern text c_since_d[11];
extern char c_credit[3];
extern int c_credit_lim;
extern float c_discount;
extern char c_data[201];
extern text h_date[20];

/* for new order transaction */

extern int nol_i_id[15];
extern int nol_supply_w_id[15];
extern int nol_quantity[15];
extern int nol_quanti10[15];
extern int nol_quanti91[15];
extern int nol_ytdqty[15];
extern int nol_amount[15];
extern int o_all_local;
extern float w_tax;
extern float d_tax;
extern float total_amount;
extern char i_name[15][25];
extern int i_name_strlen[15];
extern ub2 i_name_strlen_len[15];
extern ub2 i_name_strlen_rcode[15];
extern ub4 i_name_strlen_csize;
extern int s_quantity[15];
extern char brand_gen[15];
extern ub2 brand_gen_len[15];
extern ub2 brand_gen_rcode[15];
extern ub4 brand_gen_csize;
extern int i_price[15];
extern char brand_generic[15][1];
extern int status;
extern int tracelevel;

/* Miscellaneous */
extern OCIDate cr_date;
extern OCIDate c_since;
extern OCIDate o_entry_d_base;
extern OCIDate ol_d_base[15];

#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 */

#ifndef NULLP
# define NULLP(x)  (x  *)NULL
#endif /* NULLP */

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

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

#define min(x,y)  (((x) < (y)) ? (x) : (y))

#define OCIERROR(errp,function)\
        ocierror(__FILE__,__LINE__,(errp),(function));

#define OCIBND(stmp, bndp, errp, sqlvar, progv, progvl, ftype)\
       ocierror(__FILE__,__LINE__,(errp), \
            OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_BIND,0,(dvoid**)0)); \
       ocierror(__FILE__,__LINE__, (errp), \
            OCIBindByName((stmp), &(bndp), (errp), \
             (text *)(sqlvar), strlen((sqlvar)),\
             (progv), (progvl), (ftype),0,0,0,0,0,OCI_DEFAULT));



/* bind arrays for sql */
#define OCIBNDRA(stmp,bndp,errp,sqlvar,progv,progvl,ftype,indp,alen,arcode) \
       DISCARD ocierror(__FILE__,__LINE__,(errp), \
        OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_BIND,0,(dvoid**)0)); \
       DISCARD ocierror(__FILE__,__LINE__,(errp), \
         OCIBindByName((stmp),&(bndp),(errp),(text *)(sqlvar),strlen((sqlvar)),\
                    (progv),(progvl),(ftype),(indp),(alen),(arcode),0,0,OCI_DEFAULT));

/* use with callback data */
#define OCIBNDRAD(stmp,bndp,errp,sqlvar,progvl,ftype,indp,ctxp,\
                cbf_nodata,cbf_data) \
      DISCARD ocierror(__FILE__,__LINE__,(errp), \
       OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_BIND,0,(dvoid**)0)); \
      DISCARD ocierror(__FILE__,__LINE__,(errp), \
          OCIBindByName((stmp),&(bndp),(errp),(text *)(sqlvar), \
                     strlen((sqlvar)),0,(progvl),(ftype), \
                     indp,0,0,0,0,OCI_DATA_AT_EXEC)); \
      DISCARD ocierror(__FILE__,__LINE__,(errp), \
        OCIBindDynamic((bndp),(errp),(ctxp),(cbf_nodata),(ctxp),(cbf_data)));



/* bind in/out for plsql without indicator and rcode */
#define OCIBNDPL(stmp,bndp,errp,sqlvar,progv,progvl,ftype,alen) \
   DISCARD ocierror(__FILE__,__LINE__,(errp), \
     OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_BIND,0,(dvoid**)0)); \
   DISCARD ocierror(__FILE__,__LINE__,(errp), \
     OCIBindByName((stmp),&(bndp),(errp),(CONST text *)(sqlvar), \
      (sb4)strlen((CONST char *)(sqlvar)), (dvoid*)(progv),(progvl),(ftype),\
          NULLP(dvoid),(alen), NULLP(ub2), 0,NULLP(ub4),OCI_DEFAULT));

/* bind  in values for plsql with indicator and rcode */
#define OCIBNDR(stmp,bndp,errp,sqlvar,progv,progvl,ftype,indp,alen,arcode) \
      DISCARD ocierror(__FILE__,__LINE__,(errp), \
       OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_BIND,0,(dvoid**)0)); \
      DISCARD ocierror(__FILE__,__LINE__,(errp), \
       OCIBindByName((stmp),&(bndp),(errp),(text *)(sqlvar),strlen((sqlvar)),\
          (progv),(progvl),(ftype),(indp),(alen),(arcode),0,0, \
               OCI_DEFAULT));

/* bind in/out for plsql arrays witout indicator and rcode */
#define OCIBNDPLA(stmp,bndp,errp,sqlvar,progv,progvl,ftype,alen,ms,cu) \
      DISCARD ocierror(__FILE__,__LINE__, (errp), \
        OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_BIND,0,(dvoid**)0));\
      DISCARD ocierror(__FILE__,__LINE__,(errp),\
        OCIBindByName((stmp),&(bndp),(errp),(CONST text *)(sqlvar), \
          (sb4)strlen((CONST char *) (sqlvar)),(void *)(progv), \
          (progvl),(ftype),NULL,(alen),NULL,(ms),(cu),OCI_DEFAULT));

/* bind in/out values for plsql with indicator and rcode */
#define OCIBNDRAA(stmp,bndp,errp,sqlvar,progv,progvl,ftype,indp,alen,arcode,\
              ms,cu) \
      ocierror(__FILE__,__LINE__, (errp), \
        OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_BIND,0,(dvoid**)0));\
      ocierror(__FILE__,__LINE__,(errp),\
        OCIBindByName((stmp),&(bndp),(errp),(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) \
      OCIHandleAlloc((stmp),(dvoid**)&(dfnp),OCI_HTYPE_DEFINE,0,\
                  (dvoid**)0);\
      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) \
      OCIHandleAlloc((stmp),(dvoid**)&(dfnp),OCI_HTYPE_DEFINE,0,\
                  (dvoid**)0);\
      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)));



/* New order */

struct newinstruct {
   int w_id;
   int d_id;
   int c_id;
   int ol_i_id[15];
   int ol_supply_w_id[15];
   int ol_quantity[15];
};

struct newoutstruct {
   int terror;
   int o_id;
   int o_ol_cnt;
   char c_last[17];
   char c_credit[3];
   float c_discount;
   float w_tax;
   float d_tax;
   char o_entry_d[20];
   float total_amount;
   char i_name[15][25];
   int s_quantity[15];
   char brand_generic[15];
   float i_price[15];
   float ol_amount[15];
   char status[26];
   int retry;
};

struct newstruct {
   struct newinstruct newin;
   struct newoutstruct newout;
};


/* Payment */

struct payinstruct {
   int w_id;
   int d_id;
   int c_w_id;
   int c_d_id;
   int c_id;
   int bylastname;
   int h_amount;
   char c_last[17];
};

struct payoutstruct {
   int terror;
   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];
   int  c_id;
   char c_first[17];
   char c_middle[3];
   char c_last[17];
   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];
   char c_since[11];
   char c_credit[3];
   double c_credit_lim;
   float  c_discount;
   double c_balance;
   char c_data[201];
   char h_date[20];
   int retry;
};

struct paystruct {
   struct payinstruct payin;
   struct payoutstruct payout;
};


/* Order status */

struct ordinstruct {
   int w_id;
   int d_id;
   int c_id;
   int bylastname;
   char c_last[17];
};

struct ordoutstruct {
   int terror;
   int c_id;
   char c_last[17];
   char c_first[17];
   char c_middle[3];
   double c_balance;
   int o_id;
   char o_entry_d[20];
   int o_carrier_id;
   int o_ol_cnt;
   int ol_supply_w_id[15];
   int ol_i_id[15];
   int ol_quantity[15];
   float ol_amount[15];
   char ol_delivery_d[15][11];
   int retry;
};

struct ordstruct {
   struct ordinstruct ordin;
   struct ordoutstruct ordout;
};


/* Delivery */

struct delinstruct {
   int w_id;
   int o_carrier_id;
   double qtime;
   int in_timing_int;
   int plsqlflag;
};

struct deloutstruct {
   int terror;
   int retry;
};

struct delstruct {
   struct delinstruct delin;
   struct deloutstruct delout;
};


/* Stock level */

struct stoinstruct {
   int w_id;
   int d_id;
   int threshold;
};

struct stooutstruct {
   int terror;
   int low_stock;
   int retry;
};

struct stostruct {
   struct stoinstruct stoin;
   struct stooutstruct stoout;
};

#endif


--------------------------------------------------
                tpccload.c
--------------------------------------------------

#ifdef RCSID
static char *RCSid =
   "$Header: tpccload.c 7030100.1 96/05/13 16:20:36 plai Generic<base> $ Copyr (c) 1993 Oracle";
#endif /* RCSID */

/*==================================================================+
 |       Copyright (c) 1994  Oracle Corp, Redwood Shores, CA       |
 |                 OPEN SYSTEMS PERFORMANCE GROUP                  |
 |                      All Rights Reserved                        |
 +==================================================================+
 | FILENAME
 |    tpccload.c
 | DESCRIPTION
 |    Load or generate TPC-C database tables.
 |    Usage: tpccload -M <# of wares> [options]
 |           options: -A  load all tables
 |                    -w  load ware table
 |                    -d  load dist table
 |                    -c  load cust table
|                 -i  load item table
|                 -s  load stok table (cluster around s_w_id)
|                 -S  load stok table (cluster around s_i_id)
|                 -h  load hist table
|                 -n  load new-order table
|                 -o <oline file> load order and order-line table
|                 -b <ware#>  beginning ware number
|                 -e <ware#>  ending ware number
|                 -j <item#>  beginning item number (with -S)
|                 -k <item#>  ending item number (with -S)
|                 -g  generate rows to standard output
+==================================================================*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <sys/types.h>
#include "tpcc.h"

#ifdef ORA_NT
#undef boolean
#include <process.h>
#include "dpbcore.h"
# define gettime dpbtimef
# define getcpu  dpbcpu
#define lrand48() ((long)rand() <<15 | rand())
#ifdef __STDC__
#  define PROTO(args)    args
#else
#  define PROTO(args)    ()
#endif
#endif

#define DISTARR 10        /* dist insert array size   */
#define CUSTARR 100     /* cust insert array size  */
#define STOCARR 100     /* stok insert array size  */
#define ITEMARR 100     /* item insert array size  */
#define HISTARR 100             /* hist insert array size    */
#define ORDEARR 100             /* order insert array size      */
#define NEWOARR 100             /* new order insert array size  */

#define DISTFAC 10         /* max. dist id  */
#define CUSTFAC 3000        /* max. cust id */
#define STOCFAC 100000         /* max. stok id*/
#define ITEMFAC 100000         /* max. item id     */
#define HISTFAC 30000           /* history / warehouse  */
#define ORDEFAC 3000            /* order / district      */
#define NEWOFAC 900             /* new order / district */

#define C       0               /* constant in non-uniform dist. eqt. */
#define CNUM1   1               /* first constant in non-uniform dist. eqt. */
#define CNUM2   2               /* second constant in non-uniform dist. eqt. */
#define CNUM3   3               /* third constant in non-uniform dist. eqt. */

#define SEED    2               /* seed for random functions */

#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 SQLTXTW "INSERT INTO ware (w_id, w_ytd, w_tax, w_name, w_street_1, w_street_2,
w_city, w_state, w_zip) VALUES (:w_id, 30000000, :w_tax, :w_name, :w_street_1, \
   :w_street_2, :w_city, :w_state, :w_zip)"

#define SQLTXTD "INSERT INTO dist (d_id, d_w_id, d_ytd, d_tax, d_next_o_id, d_name,
d_street_1, d_street_2, d_city, d_state, d_zip) VALUES (:d_id, :d_w_id,3000000, :d_tax, \
    3001, :d_name, :d_street_1, :d_street_2, :d_city, :d_state, :d_zip)"

#define SQLTXTC "INSERT INTO cust (C_ID, C_D_ID, C_W_ID, C_FIRST, C_MIDDLE,
C_LAST, 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_YTD_PAYMENT,
C_PAYMENT_CNT, C_DELIVERY_CNT, C_DATA) VALUES (:c_id, :c_d_id, :c_w_id, \
   :c_first, 'OE', :c_last, :c_street_1, :c_street_2, :c_city, :c_state, \
   :c_zip, :c_phone, SYSDATE, :c_credit, 5000000, :c_discount, -1000, 1000, 1, \
   0, :c_data)"

#define SQLTXTH "INSERT INTO hist (h_c_id, h_c_d_id, h_c_w_id, h_d_id, h_w_id, h_date,
h_amount, h_data) VALUES (:h_c_id, :h_c_d_id, :h_c_w_id, \
   :h_d_id, :h_w_id, SYSDATE, 1000, :h_data)"

#define SQLTXTS "INSERT INTO stok (s_i_id, s_w_id, s_quantity,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, s_ytd, s_order_cnt,
s_remote_cnt, s_data) \
VALUES (:s_i_id, :s_w_id, :s_quantity, \
   :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, 0, 0, 0, :s_data)" \

#define SQLTXTI "INSERT INTO item (I_ID,I_IM_ID,I_NAME,I_PRICE,I_DATA) VALUES
(:i_id, :i_im_id, :i_name, :i_price, \
   :i_data)"

#define SQLTXTO1 "INSERT INTO ordr (O_ID,
O_D_ID,O_W_ID,O_C_ID,O_ENTRY_D,O_CARRIER_ID,O_OL_CNT,O_ALL_LOCAL) \
   VALUES (:o_id, :o_d_id, :o_w_id, :o_c_id, \
   SYSDATE, :o_carrier_id, :o_ol_cnt, 1)"

#define SQLTXTO2 "INSERT INTO ordr (O_ID,
O_D_ID,O_W_ID,O_C_ID,O_ENTRY_D,O_CARRIER_ID,O_OL_CNT,O_ALL_LOCAL) \
   VALUES (:o_id, :o_d_id, :o_w_id, :o_c_id, \
   SYSDATE, 11, :o_ol_cnt, 1)"

#define SQLTXTOL1 "INSERT INTO ordl (OL_O_ID, OL_D_ID, OL_W_ID, OL_NUMBER,
OL_DELIVERY_D, OL_I_ID, OL_SUPPLY_W_ID, OL_QUANTITY, OL_AMOUNT,
OL_DIST_INFO) \
   VALUES (:ol_o_id, :ol_d_id, \
   :ol_w_id, :ol_number, SYSDATE, :ol_i_id, :ol_supply_w_id, 5, 0, \
   :ol_dist_info)"

#define SQLTXTOL2 "INSERT INTO ordl (OL_O_ID, OL_D_ID, OL_W_ID, OL_NUMBER,
OL_DELIVERY_D, OL_I_ID, OL_SUPPLY_W_ID, OL_QUANTITY, OL_AMOUNT,
OL_DIST_INFO) \
   VALUES (:ol_o_id, :ol_d_id, \
   :ol_w_id, :ol_number, to_date('01-Jan-1811'), :ol_i_id, :ol_supply_w_id, 5, :ol_amount, \
   :ol_dist_info)"

#define SQLTXTNO "INSERT INTO nord (no_o_id, no_d_id, no_w_id) VALUES (:no_o_id,
:no_d_id, :no_w_id)"

#define SQLTXTENHA "alter session set \"_enable_hash_overflow\"=true"
#define SQLTXTDIHA "alter session set \"_enable_hash_overflow\"=false"

static char *lastname[] = {
   "BAR",
   "OUGHT",
   "ABLE",
   "PRI",
   "PRES",
   "ESE",
   "ANTI",
   "CALLY",
   "ATION",
   "EING"
};

char num9[10];
char num16[17];
char str2[3];
char str24[15][25];
int randperm3000[3000];

void initperm();
void randstr();
void randdatastr();
void randnum();
void randlastname (char*, int);
int NURand();
void sysdate();

OCIEnv *tpcenv;
OCIServer *tpcsrv;
OCIError *errhp;
OCISvcCtx *tpcsvc;
OCISession *tpcusr;

OCIStmt *curw;
OCIStmt *curd;
OCIStmt *curc;
OCIStmt *curh;
OCIStmt *curs;
OCIStmt *curi;
OCIStmt *curo1;
OCIStmt *curo2;
OCIStmt *curol1;
OCIStmt *curol2;
OCIStmt *curno;

OCIBind *w_id_bp = (OCIBind *) 0;
OCIBind *w_name_bp = (OCIBind *) 0;
OCIBind *w_street1_bp = (OCIBind *) 0;
OCIBind *w_street2_bp = (OCIBind *) 0;
OCIBind *w_city_bp = (OCIBind *) 0;
OCIBind *w_state_bp = (OCIBind *) 0;
OCIBind *w_zip_bp = (OCIBind *) 0;
OCIBind *w_tax_bp = (OCIBind *) 0;

OCIBind *d_id_bp = (OCIBind *) 0;
OCIBind *d_w_id_bp = (OCIBind *) 0;
OCIBind *d_name_bp = (OCIBind *) 0;
OCIBind *d_street1_bp = (OCIBind *) 0;
OCIBind *d_street2_bp = (OCIBind *) 0;
OCIBind *d_city_bp = (OCIBind *) 0;
OCIBind *d_state_bp = (OCIBind *) 0;
OCIBind *d_zip_bp = (OCIBind *) 0;
OCIBind *d_tax_bp = (OCIBind *) 0;

OCIBind *c_id_bp = (OCIBind *) 0;
OCIBind *c_d_id_bp = (OCIBind *) 0;
OCIBind *c_w_id_bp = (OCIBind *) 0;
OCIBind *c_first_bp = (OCIBind *) 0;
OCIBind *c_last_bp = (OCIBind *) 0;
OCIBind *c_street1_bp = (OCIBind *) 0;
OCIBind *c_street2_bp = (OCIBind *) 0;
OCIBind *c_city_bp = (OCIBind *) 0;
OCIBind *c_state_bp = (OCIBind *) 0;
OCIBind *c_zip_bp = (OCIBind *) 0;
OCIBind *c_phone_bp = (OCIBind *) 0;
OCIBind *c_discount_bp = (OCIBind *) 0;
OCIBind *c_credit_bp = (OCIBind *) 0;
OCIBind *c_data_bp = (OCIBind *) 0;

OCIBind *i_id_bp = (OCIBind *) 0;
OCIBind *i_im_id_bp = (OCIBind *) 0;
OCIBind *i_name_bp = (OCIBind *) 0;
OCIBind *i_price_bp = (OCIBind *) 0;
OCIBind *i_data_bp = (OCIBind *) 0;

OCIBind *s_i_id_bp = (OCIBind *) 0;
OCIBind *s_w_id_bp = (OCIBind *) 0;
OCIBind *s_quantity_bp = (OCIBind *) 0;
OCIBind *s_dist_01_bp = (OCIBind *) 0;
OCIBind *s_dist_02_bp = (OCIBind *) 0;
OCIBind *s_dist_03_bp = (OCIBind *) 0;
OCIBind *s_dist_04_bp = (OCIBind *) 0;
OCIBind *s_dist_05_bp = (OCIBind *) 0;
OCIBind *s_dist_06_bp = (OCIBind *) 0;
OCIBind *s_dist_07_bp = (OCIBind *) 0;
OCIBind *s_dist_08_bp = (OCIBind *) 0;
OCIBind *s_dist_09_bp = (OCIBind *) 0;
OCIBind *s_dist_10_bp = (OCIBind *) 0;
OCIBind *s_data_bp = (OCIBind *) 0;

OCIBind *h_c_id_bp = (OCIBind *) 0;
OCIBind *h_c_d_id_bp = (OCIBind *) 0;
OCIBind *h_c_w_id_bp = (OCIBind *) 0;
OCIBind *h_d_id_bp = (OCIBind *) 0;
OCIBind *h_w_id_bp = (OCIBind *) 0;
OCIBind *h_data_bp = (OCIBind *) 0;

OCIBind *ol_o_id_bp = (OCIBind *) 0;
OCIBind *ol_d_id_bp = (OCIBind *) 0;
OCIBind *ol_w_id_bp = (OCIBind *) 0;
OCIBind *ol_i_id_bp = (OCIBind *) 0;
OCIBind *ol_number_bp = (OCIBind *) 0;
OCIBind *ol_supply_w_id_bp = (OCIBind *) 0;
OCIBind *ol_dist_info_bp = (OCIBind *) 0;
OCIBind *ol_amount_bp = (OCIBind *) 0;

OCIBind *o_id_bp = (OCIBind *) 0;
OCIBind *o_d_id_bp = (OCIBind *) 0;
OCIBind *o_w_id_bp = (OCIBind *) 0;
OCIBind *o_c_id_bp = (OCIBind *) 0;
OCIBind *o_carrier_id_bp = (OCIBind *) 0;
OCIBind *o_ol_cnt_bp  = (OCIBind *) 0;
OCIBind *o_ocnt_bp = (OCIBind *) 0;
OCIBind *o_olcnt_bp  = (OCIBind *) 0;


OCIBind *no_o_id_bp  = (OCIBind *) 0;
OCIBind *no_d_id_bp  = (OCIBind *) 0;
OCIBind *no_w_id_bp  = (OCIBind *) 0;

void myusage()
{
   fprintf (stderr, "\n");
   fprintf (stderr, "Usage:\ttpccload -M <multiplier> [options]\n");
   fprintf (stderr, "options:\n");
   fprintf (stderr, "\t-A :\tload all tables\n");
   fprintf (stderr, "\t-w :\tload ware table\n");
   fprintf (stderr, "\t-d :\tload dist table\n");
   fprintf (stderr, "\t-c :\tload cust table\n");
   fprintf (stderr, "\t-i :\tload item table\n");
   fprintf (stderr, "\t-s :\tload stok table (cluster around s_w_id)\n");
   fprintf (stderr, "\t-S :\tload stok table (cluster around s_i_id)\n");
   fprintf (stderr, "\t-h :\tload hist table\n");
   fprintf (stderr, "\t-n :\tload new-order table\n");
   fprintf (stderr, "\t-o <oline file> :\tload order and order-line table\n");
   fprintf (stderr, "\t-b <ware#> :\tbeginning ware number\n");
   fprintf (stderr, "\t-e <ware#> :\tending ware number\n");
   fprintf (stderr, "\t-j <item#> :\tbeginning item number (with -S)\n");
   fprintf (stderr, "\t-k <item#> :\tending item number (with -S)\n");
   fprintf (stderr, "\t-g :\tgenerate rows to standard output\n");
   fprintf (stderr,"\t   $tpcc_bench must be set to the location of the kit\n");
   fprintf (stderr, "\n");
   exit(1);
}

int sqlfile(fnam,linebuf)
char   *fnam;
text   *linebuf;
{
    FILE *fd;
    int nulpt = 0;
    char realfile[512];

    sprintf(realfile,"%s",fnam);
    fd = fopen(realfile,"r");
    if (!fd)
    {
      return (0);
    }
    while (fgets((char *)linebuf+nulpt, SQL_BUF_SIZE, fd))
    {
        nulpt = strlen((char *)linebuf);
    }
    return(nulpt);
}

void quit()
{
   OCIERROR(errhp,OCISessionEnd ( tpcsvc,errhp, tpcusr, OCI_DEFAULT));
   OCIERROR(errhp,OCIServerDetach ( tpcsrv, errhp, OCI_DEFAULT));
   OCIHandleFree((dvoid *)tpcusr, OCI_HTYPE_SESSION);
   OCIHandleFree((dvoid *)tpcsvc, OCI_HTYPE_SVCCTX);
   OCIHandleFree((dvoid *)errhp, OCI_HTYPE_ERROR);
   OCIHandleFree((dvoid *)tpcsrv, OCI_HTYPE_SERVER);
   OCIHandleFree((dvoid *)tpcenv, OCI_HTYPE_ENV);
}

void main (argc, argv)
int argc;
char *argv[];
{
   char *uid="tpcc";
   char *pwd="tpcc";
   int scale=0;
   int i, j;
   int loop;
   int loopcount;
   int cid;
   int dwid;
   int cdid;
   int cwid;
   int sid;
   int swid;
   int olcnt;
   int nrows;
   int row;

   int w_id;
   char w_name[11];
   char w_street_1[21];
   char w_street_2[21];
   char w_city[21];
   char w_state[2];
   char w_zip[9];
   float w_tax;

   int d_id[10];
   int d_w_id[10];
   char d_name[10][11];
   char d_street_1[10][21];
   char d_street_2[10][21];
   char d_city[10][21];
   char d_state[10][2];
   char d_zip[10][9];
   float d_tax[10];

   int c_id[100];
   int c_d_id[100];
   int c_w_id[100];
   char c_first[100][17];
   char c_last[100][17];
   char c_street_1[100][21];
   char c_street_2[100][21];
   char c_city[100][21];
   char c_state[100][2];
   char c_zip[100][9];
   char c_phone[100][16];
   char c_credit[100][2];
   float c_discount[100];
   char c_data[100][501];
   int i_id[100];
   int i_im_id[100];
   int i_price[100];
   char i_name[100][25];
   char i_data[100][51];

   int s_i_id[100];
   int s_w_id[100];
   int s_quantity[100];
   char s_dist_01[100][24];
   char s_dist_02[100][24];
   char s_dist_03[100][24];
   char s_dist_04[100][24];
   char s_dist_05[100][24];
   char s_dist_06[100][24];
   char s_dist_07[100][24];
   char s_dist_08[100][24];
   char s_dist_09[100][24];
   char s_dist_10[100][24];
   char s_data[100][51];

   int h_w_id[100];
   int h_d_id[100];
   int h_c_id[100];
   char h_data[100][25];

   int o_id[100];
   int o_d_id[100];
   int o_w_id[100];
   int o_c_id[100];
   int o_carrier_id[100];
   int o_ol_cnt[100];

   int ol_o_id[1500];
   int ol_d_id[1500];
   int ol_w_id[1500];
   int ol_number[1500];
   int ol_i_id[1500];
   int ol_supply_w_id[1500];
   int ol_amount[1500];
   char ol_dist_info[1500][24];
   int o_cnt;
   int ol_cnt;

   ub2 ol_o_id_len[1500];
   ub2 ol_d_id_len[1500];
   ub2 ol_w_id_len[1500];
   ub2 ol_number_len[1500];
   ub2 ol_i_id_len[1500];
   ub2 ol_supply_w_id_len[1500];
   ub2 ol_dist_info_len[1500];
   ub2 ol_amount_len[1500];

   ub4 ol_o_id_clen;
   ub4 ol_d_id_clen;
   ub4 ol_w_id_clen;
   ub4 ol_number_clen;
   ub4 ol_i_id_clen;
   ub4 ol_supply_w_id_clen;
   ub4 ol_dist_info_clen;
   ub4 ol_amount_clen;

   ub2 o_id_len[100];
   ub2 o_d_id_len[100];
   ub2 o_w_id_len[100];
   ub2 o_c_id_len[100];
   ub2 o_carrier_id_len[100];
   ub2 o_ol_cnt_len[100];

   ub4 o_id_clen;
   ub4 o_d_id_clen;
   ub4 o_w_id_clen;
   ub4 o_c_id_clen;
   ub4 o_carrier_id_clen;
   ub4 o_ol_cnt_clen;

   text stmbuf[16*1024];

   int no_o_id[100];
   int no_d_id[100];
   int no_w_id[100];

   char sdate[30];

#ifdef ORA_NT
   clock_t begin_time, end_time;
   clock_t begin_cpu, end_cpu;

   char *arg_ptr,  **end_args;
#else
   double begin_time, end_time;
   double begin_cpu, end_cpu;
   double gettime(), getcpu();

   extern int getopt();
   extern char *optarg;
   extern int optind, opterr;
   int opt;
#endif

   char     *argstr="M:AwdcisShno:b:e:j:k:g";
   int do_A=0;
   int do_w=0;
   int do_d=0;
   int do_i=0;
   int do_c=0;
   int do_s=0;
   int do_S=0;
   int do_h=0;
   int do_o=0;
   int do_n=0;
   int gen=0;
   int bware=1;
   int eware=0;
   int bitem=1;
   int eitem=0;

   FILE *olfp=NULL;
   char olfname[100];
   char* basename;
   int status;
#ifdef ORA_NT
   char fname[100];
   FILE *logfile;
#endif /* ORA_NT */

/*---------------------------------------------------------------+
 | Parse command line -- look for scale factor.             |
 +---------------------------------------------------------------*/

   if (argc == 1) {
      myusage ();
   }

#ifdef ORA_NT
   end_args = argv + argc;
   for (++argv; argv < end_args; )
   {
      arg_ptr = *argv++;

         if (*arg_ptr != '-')
         {
            myusage ();
         } else
         {
        switch (arg_ptr[1]) {
         case '?': myusage ();
                   break;
         case 'M': scale = atoi (*argv++);
                   break;
         case 'A': do_A = 1;
                   break;
         case 'w': do_w = 1;
                   break;
         case 'd': do_d = 1;
                   break;
         case 'c': do_c = 1;
                   break;
         case 'i': do_i = 1;
                   break;
         case 's': do_s = 1;
                   break;
         case 'S': do_S = 1;
                   break;
         case 'h': do_h = 1;
                   break;
         case 'n': do_n = 1;
                   break;
         case 'o': do_o = 1;
                   strcpy (olfname, *argv++);
                   break;
         case 'b': bware = atoi (*argv++);
                   break;
         case 'e': eware = atoi (*argv++);
                   break;
         case 'j': bitem = atoi (*argv++);
                   break;
         case 'k': eitem = atoi (*argv++);
                   break;
         case 'g': gen = 1;
                   strcpy (fname, *argv++);
                   break;
        case 'l': logfile=fopen(*argv++,"w");
                  break;
        default: fprintf (stderr, "THIS SHOULD NEVER HAPPEN!!!\n");
                 fprintf (stderr, "(reached default case in getopt ())\n");
                 myusage ();
      }
     }
  }


#else

  while ((opt = getopt (argc, argv, argstr)) != -1) {
    switch (opt) {
      case '?': myusage ();
                break;
      case 'M': scale = atoi (optarg);
                break;
      case 'A': do_A = 1;
                break;
      case 'w': do_w = 1;
                break;
      case 'd': do_d = 1;
                break;
      case 'c': do_c = 1;
                break;
      case 'i': do_i = 1;
                break;
      case 's': do_s = 1;
                break;
      case 'S': do_S = 1;
                break;
      case 'h': do_h = 1;
                break;
      case 'n': do_n = 1;
                break;
      case 'o': do_o = 1;
                strcpy (olfname, optarg);
                break;
      case 'b': bware = atoi (optarg);
                break;
      case 'e': eware = atoi (optarg);
                break;
      case 'j': bitem = atoi (optarg);
                break;
      case 'k': eitem = atoi (optarg);
                break;
      case 'g': gen = 1;
                break;
      default: fprintf (stderr, "THIS SHOULD NEVER HAPPEN!!!\n");
               fprintf (stderr, "(reached default case in getopt ())\n");
               myusage ();
    }
  }

# endif /* ORA_NT */

/*-------------------------------------------------------------*|
|    Rudimentary error checking                  |
|*-------------------------------------------------------------*/

  if (scale  < 1) {
    fprintf (stderr, "Invalid scale factor: '%d'\n", scale);
    myusage ();
  }

  if (!(do_A || do_w || do_d || do_c || do_i || do_s || do_S || do_h || do_o ||
      do_n)) {
    fprintf (stderr, "What should I load???\n");
    myusage ();
  }

  if (gen && (do_A || (do_w + do_d + do_c + do_i + do_s + do_S + do_h + do_o +
                       do_n > 1))) {
    fprintf (stderr, "Can only generate table one at a time\n");
    myusage ();
  }

  if (do_S && (do_A || do_s)) {
    fprintf (stderr, "Cluster stock table around s_w_id or s_i_id?\n");
    myusage ();
  }

  if (eware <= 0)
    eware = scale;
  if (eitem <= 0)
    eitem = STOCFAC;

  if (do_S) {
    if ((bitem  < 1) || (bitem > STOCFAC)) {
      fprintf (stderr, "Invalid beginning item number: '%d'\n", bitem);
      myusage ();
    }

    if ((eitem  < bitem) || (eitem > STOCFAC)) {
      fprintf (stderr, "Invalid ending item number: '%d'\n", eitem);
      myusage ();
    }
  }
  if (do_o) {
    if ((basename = getenv ("tpcc_bench")) == NULL)
    {
      fprintf (stderr, "$tpcc_bench is not set");
      myusage ();
    }
  }

  if ((bware  < 1) || (bware > scale)) {
    fprintf (stderr, "Invalid beginning warehouse number: '%d'\n", bware);
    myusage ();
  }

  if ((eware  < bware) || (eware > scale)) {
    fprintf (stderr, "Invalid ending warehouse number: '%d'\n", eware);
    myusage ();
  }

  if (gen && do_o) {
    if ((olfp = fopen (olfname, "w")) == NULL) {
      fprintf (stderr, "Can't open '%s' for writing order lines\n", olfname);
      myusage ();
    }


  }

/*-------------------------------------------------------------+
 | Prepare to insert into database.              |
 +-------------------------------------------------------------*/

  sysdate (sdate);
  if (!gen) {

    /* log on to Oracle */

    OCIInitialize(OCI_DEFAULT|OCI_OBJECT,(dvoid *)0,0,0,0);
    OCIEnvInit(&tpcenv, OCI_DEFAULT, 0, (dvoid **)0);
    OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&tpcsrv, OCI_HTYPE_SERVER, 0 , (dvoid **)0);
    OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&errhp, OCI_HTYPE_ERROR, 0 , (dvoid **)0);
    OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&tpcsvc, OCI_HTYPE_SVCCTX, 0 , (dvoid **)0);
    OCIServerAttach(tpcsrv, errhp, (text *)0,0,OCI_DEFAULT);
    OCIAttrSet((dvoid *)tpcsvc, OCI_HTYPE_SVCCTX, (dvoid *)tpcsrv,
            (ub4)0,OCI_ATTR_SERVER, errhp);
    OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&tpcusr, OCI_HTYPE_SESSION, 0 , (dvoid **)0);
    OCIAttrSet((dvoid *)tpcusr, OCI_HTYPE_SESSION, (dvoid *)uid,
            (ub4)strlen(uid),OCI_ATTR_USERNAME, errhp);
    OCIAttrSet((dvoid *)tpcusr, OCI_HTYPE_SESSION, (dvoid *)pwd, (ub4)strlen(pwd),
         OCI_ATTR_PASSWORD, errhp);
    OCIERROR(errhp, OCISessionBegin(tpcsvc, errhp, tpcusr, OCI_CRED_RDBMS,
OCI_DEFAULT));

    OCIAttrSet(tpcsvc, OCI_HTYPE_SVCCTX, tpcusr, 0, OCI_ATTR_SESSION, errhp);

    fprintf (stderr, "\nConnected to Oracle userid '%s/%s'.\n", uid, pwd);

    /* open cursors and parse statement */
    if (do_A || do_w)  {
      OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid **)(&curw), OCI_HTYPE_STMT, 0,
(dvoid**)0));
      OCIERROR(errhp,OCIStmtPrepare(curw, errhp, (text *)SQLTXTW,
              strlen((char *)SQLTXTW), (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT));
    }

    if (do_A || do_d) {
      OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid **)(&curd), OCI_HTYPE_STMT, 0,
(dvoid**)0));
      OCIERROR(errhp,OCIStmtPrepare(curd, errhp, (text *)SQLTXTD,
              strlen((char *)SQLTXTD), (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT));
    }

    if (do_A || do_c) {
      OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid **)(&curc), OCI_HTYPE_STMT, 0,
(dvoid**)0));
      OCIERROR(errhp,OCIStmtPrepare(curc, errhp, (text *)SQLTXTC,
              strlen((char *)SQLTXTC), (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT));
    }

    if (do_A || do_h) {
      OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid **)(&curh), OCI_HTYPE_STMT, 0,
(dvoid**)0));
      OCIERROR(errhp,OCIStmtPrepare(curh, errhp, (text *)SQLTXTH,
              strlen((char *)SQLTXTH), (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT));
    }

    if (do_A || do_s || do_S) {
      OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid **)(&curs), OCI_HTYPE_STMT, 0,
(dvoid**)0));
      OCIERROR(errhp,OCIStmtPrepare(curs, errhp, (text *)SQLTXTS,
                strlen((char *)SQLTXTS), (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT));
    }

    if (do_A || do_i) {
      OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid **)(&curi), OCI_HTYPE_STMT, 0,
(dvoid**)0));
      OCIERROR(errhp,OCIStmtPrepare(curi, errhp, (text *)SQLTXTI,
                strlen((char *)SQLTXTI), (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT));
    }

    if (do_A || do_o) {
      int stat;
      char fname[160];
      OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid **)(&curo1), OCI_HTYPE_STMT, 0,
(dvoid**)0));
      DISCARD strcpy(fname,basename);
      DISCARD strcat(fname, "/");
      DISCARD strcat(fname, "benchrun/blocks/load_ordordl.sql");
      stat = sqlfile(fname, stmbuf);
      if (!stat)
      {
        fprintf (stderr, "unable to open %s \n",fname);
        quit();
        exit(1);
      }
      OCIERROR(errhp,OCIStmtPrepare(curo1, errhp, stmbuf,
                strlen((char *)stmbuf), (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT));
    }

    if (do_A || do_n) {
      OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid **)(&curno), OCI_HTYPE_STMT, 0,
(dvoid**)0));
      OCIERROR(errhp,OCIStmtPrepare(curno, errhp, (text *)SQLTXTNO,
                strlen((char *)SQLTXTNO), (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT));
    }

    /* bind variables */

    /* warehouse */

    if (do_A || do_w) {
      OCIERROR(errhp, OCIBindByName(curw, &w_id_bp, errhp, (text *)(":w_id"),
strlen((":w_id")),
                 (ub1 *)&(w_id), sizeof(w_id), SQLT_INT, (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curw, &w_name_bp, errhp,(text *)":w_name",
strlen(":w_name"),
                 (ub1 *)w_name, 11, SQLT_STR, (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curw, &w_street1_bp, errhp, (text *)":w_street_1",
           strlen(":w_street_1"), (ub1 *)w_street_1, 21, SQLT_STR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curw, &w_street2_bp, errhp, (text *)":w_street_2",
           strlen(":w_street_2"), (ub1 *)w_street_2, 21, SQLT_STR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curw, &w_city_bp, errhp, (text *)":w_city",
           strlen(":w_city"), (ub1 *)w_city, 21, SQLT_STR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curw, &w_state_bp, errhp, (text *)":w_state",
           strlen(":w_state"), (ub1 *)w_state, 2, SQLT_CHR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curw, &w_zip_bp, errhp, (text *)":w_zip",
           strlen(":w_zip"), (ub1 *)w_zip, 9, SQLT_CHR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curw, &w_tax_bp, errhp, (text *)":w_tax",
           strlen(":w_tax"), (ub1 *) & w_tax, sizeof(w_tax), SQLT_FLT,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));
    }

    /* district */

    if (do_A || do_d) {
      OCIERROR(errhp, OCIBindByName(curd, &d_id_bp, errhp, (text *)":d_id",
           strlen(":d_id"), (ub1 *)d_id, sizeof(int), SQLT_INT,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curd, &d_w_id_bp, errhp, (text *)":d_w_id",
           strlen(":d_w_id"), (ub1 *)d_w_id, sizeof(int), SQLT_INT,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curd, &d_name_bp, errhp, (text *)":d_name",
           strlen(":d_name"), (ub1 *)d_name, 11, SQLT_STR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curd, &d_street1_bp, errhp, (text *)":d_street_1",
           strlen(":d_street_1"), (ub1 *)d_street_1, 21, SQLT_STR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curd, &d_street2_bp, errhp, (text *)":d_street_2",
           strlen(":d_street_2"), (ub1 *)d_street_2, 21, SQLT_STR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curd, &d_city_bp, errhp, (text *)":d_city",
           strlen(":d_city"), (ub1 *)d_city, 21, SQLT_STR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curd, &d_state_bp, errhp, (text *)":d_state",
                 strlen(":d_state"), (ub1 *)d_state, 2, SQLT_CHR,
                 (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curd, &d_zip_bp, errhp, (text *)":d_zip",
                 strlen(":d_zip"), (ub1 *)d_zip, 9, SQLT_CHR,
                 (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curd, &d_tax_bp, errhp, (text *)":d_tax",
                 strlen(":d_tax"), (ub1 *)d_tax, sizeof(float), SQLT_FLT,
                 (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));
    }

    /* customer */

    if (do_A || do_c) {
      OCIERROR(errhp, OCIBindByName(curc, &c_id_bp, errhp, (text *)":c_id",
                 strlen(":c_id"), (ub1 *)c_id, sizeof(int), SQLT_INT,
                 (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curc, &c_d_id_bp, errhp, (text *)":c_d_id",
                 strlen(":c_d_id"), (ub1 *)c_d_id, sizeof(int), SQLT_INT,
                 (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curc, &c_w_id_bp, errhp, (text *)":c_w_id",
                 strlen(":c_w_id"), (ub1 *)c_w_id, sizeof(int), SQLT_INT,
                 (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curc, &c_first_bp, errhp, (text *)":c_first",
           strlen(":c_first"), (ub1 *)c_first, 17, SQLT_STR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curc, &c_last_bp, errhp, (text *)":c_last",
           strlen(":c_last"), (ub1 *)c_last, 17, SQLT_STR,
           (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curc, &c_street1_bp, errhp, (text *)":c_street_1",
                 strlen(":c_street_1"), (ub1 *)c_street_1, 21, SQLT_STR,
                 (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curc, &c_street2_bp, errhp, (text *)":c_street_2",
                 strlen(":c_street_2"), (ub1 *)c_street_2, 21, SQLT_STR,
                 (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curc, &c_city_bp, errhp, (text *)":c_city",
                 strlen(":c_city"), (ub1 *)c_city, 21, SQLT_STR,
                 (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                 (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

      OCIERROR(errhp, OCIBindByName(curc, &c_state_bp, errhp, (text *)":c_state",
                 strlen(":c_state"), (ub1 *)c_state, 2, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curc, &c_zip_bp, errhp, (text *)":c_zip",
                strlen(":c_zip"), (ub1 *)c_zip, 9, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curc, &c_phone_bp, errhp, (text *)":c_phone",
                strlen(":c_phone"), (ub1 *)c_phone, 16, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curc, &c_credit_bp, errhp, (text *)":c_credit",
                strlen(":c_credit"), (ub1 *)c_credit, 2, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curc, &c_discount_bp, errhp, (text *)":c_discount",
                strlen(":c_discount"), (ub1 *)c_discount, sizeof(float), SQLT_FLT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curc, &c_data_bp, errhp, (text *)":c_data",
                strlen(":c_data"), (ub1 *)c_data, 501, SQLT_STR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));
  }

  /* item */

  if (do_A || do_i) {
    OCIERROR(errhp, OCIBindByName(curi, &i_id_bp, errhp, (text *)":i_id",
                strlen(":i_id"), (ub1 *)i_id, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curi, &i_im_id_bp, errhp, (text *)":i_im_id",
                strlen(":i_im_id"), (ub1 *)i_im_id, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curi, &i_name_bp, errhp, (text *)":i_name",
                strlen(":i_name"), (ub1 *)i_name, 25, SQLT_STR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curi, &i_price_bp, errhp, (text *)":i_price",
                strlen(":i_price"), (ub1 *)i_price, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curi, &i_data_bp, errhp, (text *)":i_data",
                strlen(":i_data"), (ub1 *)i_data, 51, SQLT_STR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));
  }

  /* stock */

  if (do_A || do_s || do_S) {
    OCIERROR(errhp, OCIBindByName(curs, &s_i_id_bp, errhp, (text *)":s_i_id",
                strlen(":s_i_id"), (ub1 *)s_i_id, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curs, &s_w_id_bp, errhp, (text *)":s_w_id",
                strlen(":s_w_id"), (ub1 *)s_w_id, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curs, &s_quantity_bp, errhp, (text *)":s_quantity",
                strlen(":s_quantity"), (ub1 *)s_quantity, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curs, &s_dist_01_bp, errhp, (text *)":s_dist_01",
                strlen(":s_dist_01"), (ub1 *)s_dist_01, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));


    OCIERROR(errhp, OCIBindByName(curs, &s_dist_02_bp, errhp, (text *)":s_dist_02",
                strlen(":s_dist_02"), (ub1 *)s_dist_02, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));


    OCIERROR(errhp, OCIBindByName(curs, &s_dist_03_bp, errhp, (text *)":s_dist_03",
                strlen(":s_dist_03"), (ub1 *)s_dist_03, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curs, &s_dist_04_bp, errhp, (text *)":s_dist_04",
                strlen(":s_dist_04"), (ub1 *)s_dist_04, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));


    OCIERROR(errhp, OCIBindByName(curs, &s_dist_05_bp, errhp, (text *)":s_dist_05",
                strlen(":s_dist_05"), (ub1 *)s_dist_05, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));


    OCIERROR(errhp, OCIBindByName(curs, &s_dist_06_bp, errhp, (text *)":s_dist_06",
                strlen(":s_dist_06"), (ub1 *)s_dist_06, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));


    OCIERROR(errhp, OCIBindByName(curs, &s_dist_07_bp, errhp, (text *)":s_dist_07",
                strlen(":s_dist_07"), (ub1 *)s_dist_07, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));


    OCIERROR(errhp, OCIBindByName(curs, &s_dist_08_bp, errhp, (text *)":s_dist_08",
                strlen(":s_dist_08"), (ub1 *)s_dist_08, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));


    OCIERROR(errhp, OCIBindByName(curs, &s_dist_09_bp, errhp, (text *)":s_dist_09",
                strlen(":s_dist_09"), (ub1 *)s_dist_09, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));


    OCIERROR(errhp, OCIBindByName(curs, &s_dist_10_bp, errhp, (text *)":s_dist_10",
                strlen(":s_dist_10"), (ub1 *)s_dist_10, 24, SQLT_CHR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curs, &s_data_bp, errhp, (text *)":s_data",
                strlen(":s_data"), (ub1 *)s_data, 51, SQLT_STR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));
  }

  /* history */

  if (do_A || do_h) {
    OCIERROR(errhp, OCIBindByName(curh, &h_c_id_bp, errhp, (text *)":h_c_id",
                strlen(":h_c_id"), (ub1 *)h_c_id, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curh, &h_c_d_id_bp, errhp, (text *)":h_c_d_id",
                strlen(":h_c_d_id"), (ub1 *)h_d_id, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curh, &h_c_w_id_bp, errhp, (text *)":h_c_w_id",
                strlen(":h_c_w_id"), (ub1 *)h_w_id, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curh, &h_d_id_bp, errhp, (text *)":h_d_id",
                strlen(":h_d_id"), (ub1 *)h_d_id, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curh, &h_w_id_bp, errhp, (text *)":h_w_id",
                strlen(":h_w_id"), (ub1 *)h_w_id, sizeof(int), SQLT_INT,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

    OCIERROR(errhp, OCIBindByName(curh, &h_data_bp, errhp, (text *)":h_data",
                strlen(":h_data"), (ub1 *)h_data, 25, SQLT_STR,
                (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

  }

  /* order and order_line (delivered) */

  if (do_A || do_o) {

    for (i = 0; i < ORDEARR; i++ ) {
      o_id_len[i] = sizeof(int);
```

```
            o_d_id_len[i] = sizeof(int);
            o_w_id_len[i] = sizeof(int);
            o_c_id_len[i] = sizeof(int);
            o_carrier_id_len[i] = sizeof(int);
            o_ol_cnt_len[i] = sizeof(int);
        }

        OCIERROR(errhp, OCIBindByName(curo1, &ol_o_id_bp, errhp, (text *)":ol_o_id",
                    strlen(":ol_o_id"), (ub1 *)ol_o_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)ol_o_id_len, (ub2 *)0,
                    (ub4) 15*ORDEARR, (ub4 *)&ol_o_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &ol_d_id_bp, errhp, (text *)":ol_d_id",
                    strlen(":ol_d_id"), (ub1 *)ol_d_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)ol_d_id_len, (ub2 *)0,
                    (ub4) 15*ORDEARR, (ub4 *) &ol_d_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &ol_w_id_bp, errhp, (text *)":ol_w_id",
                    strlen(":ol_w_id"), (ub1 *)ol_w_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)ol_w_id_len, (ub2 *)0,
                    (ub4) 15*ORDEARR, (ub4 *) &ol_w_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &ol_number_bp, errhp, (text *)":ol_number",
                    strlen(":ol_number"), (ub1 *)ol_number, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)ol_number_len, (ub2 *)0,
                    (ub4) 15*ORDEARR, (ub4 *) &ol_number_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &ol_i_id_bp, errhp, (text *)":ol_i_id",
                    strlen(":ol_i_id"), (ub1 *)ol_i_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)ol_i_id_len, (ub2 *)0,
                    (ub4) 15*ORDEARR, (ub4 *) &ol_i_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &ol_supply_w_id_bp, errhp, (text
*)":ol_supply_w_id",
                    strlen(":ol_supply_w_id"), (ub1 *)ol_supply_w_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)ol_supply_w_id_len, (ub2 *)0,
                    (ub4) 15*ORDEARR, (ub4 *) &ol_supply_w_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &ol_dist_info_bp, errhp, (text *)":ol_dist_info",
                    strlen(":ol_dist_info"), (ub1 *)ol_dist_info, 24, SQLT_CHR,
                    (dvoid *) 0, (ub2 *)ol_dist_info_len, (ub2 *)0,
                    (ub4) 15*ORDEARR, (ub4 *) &ol_dist_info_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &ol_amount_bp, errhp, (text *)":ol_amount",
                    strlen(":ol_amount"), (ub1 *)ol_amount, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)ol_amount_len, (ub2 *)0,
                    (ub4) 15*ORDEARR, (ub4 *) &ol_amount_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &o_id_bp, errhp, (text *)":o_id",
                    strlen(":o_id"), (ub1 *)o_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)o_id_len, (ub2 *)0,
                    (ub4) ORDEARR, (ub4 *) &o_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &o_d_id_bp, errhp, (text *)":o_d_id",
                    strlen(":o_d_id"), (ub1 *)o_d_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)o_d_id_len, (ub2 *)0,
                    (ub4) ORDEARR, (ub4 *) &o_d_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &o_w_id_bp, errhp, (text *)":o_w_id",
                    strlen(":o_w_id"), (ub1 *)o_w_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)o_w_id_len, (ub2 *)0,
                    (ub4) ORDEARR, (ub4 *) &o_w_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &o_c_id_bp, errhp, (text *)":o_c_id",
                    strlen(":o_c_id"), (ub1 *)o_c_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)o_c_id_len, (ub2 *)0,
                    (ub4) ORDEARR, (ub4 *) &o_c_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &o_carrier_id_bp, errhp, (text *)":o_carrier_id",
                    strlen(":o_carrier_id"), (ub1 *)o_carrier_id, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)o_carrier_id_len, (ub2 *)0,
                    (ub4) ORDEARR, (ub4 *) &o_carrier_id_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &o_ol_cnt_bp, errhp, (text *)":o_ol_cnt",
                    strlen(":o_ol_cnt"), (ub1 *)o_ol_cnt, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)o_ol_cnt_len, (ub2 *)0,
                    (ub4) ORDEARR, (ub4 *) &o_ol_cnt_clen, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &o_ocnt_bp, errhp, (text *)":order_rows",
                    strlen(":order_rows"), (ub1 *)&o_cnt, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                    (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

        OCIERROR(errhp, OCIBindByName(curo1, &o_olcnt_bp, errhp, (text *)":ordl_rows",
                    strlen(":ordl_rows"), (ub1 *)&ol_cnt, sizeof(int), SQLT_INT,
                    (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                    (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));
    }

    /* new order */

        if (do_A || do_n) {
            OCIERROR(errhp, OCIBindByName(curno, &no_o_id_bp, errhp, (text *)":no_o_id",
                        strlen(":no_o_id"), (ub1 *)no_o_id, sizeof(int), SQLT_INT,
                        (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                        (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

            OCIERROR(errhp, OCIBindByName(curno, &no_d_id_bp, errhp, (text *)":no_d_id",
                        strlen(":no_d_id"), (ub1 *)no_d_id, sizeof(int), SQLT_INT,
                        (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                        (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));

            OCIERROR(errhp, OCIBindByName(curno, &no_w_id_bp, errhp, (text *)":no_w_id",
                        strlen(":no_w_id"), (ub1 *)no_w_id, sizeof(int), SQLT_INT,
                        (dvoid *) 0, (ub2 *)0, (ub2 *)0,
                        (ub4) 0, (ub4 *) 0, (ub4) OCI_DEFAULT));
        }
    }

/*---------------------------------------------------------------+
 | Initialize random number generator                  |
 +---------------------------------------------------------------*/

    srand (SEED);
#ifndef ORA_NT
    srand48 (SEED);
#endif
    initperm ();


/*---------------------------------------------------------------+
 | Load the WAREHOUSE table.                        |
 +---------------------------------------------------------------*/

    if (do_A || do_w) {
        nrows = eware - bware + 1;

        fprintf (stderr, "Loading/generating warehouse: w%d - w%d (%d rows)\n",
                 bware, eware, nrows);

        begin_time = gettime ();
        begin_cpu = getcpu ();

        for (loop = bware; loop <= eware; loop++) {

            w_tax = (float) ((lrand48 () % 2001) * 0.0001);
            randstr (w_name, 6, 10);
            randstr (w_street_1, 10, 20);
            randstr (w_street_2, 10, 20);
            randstr (w_city, 10, 20);
            randstr (str2, 2, 2);
            randnum (num9, 9);
            num9[4] = num9[5] = num9[6] = num9[7] = num9[8] = '1';

            if (gen) {
                printf ("%d 30000000 %6.4f %s %s %s %s %s %s\n", loop, w_tax,
                        w_name, w_street_1, w_street_2, w_city, str2, num9);
                fflush (stdout);
            }
            else {
                w_id = loop;
                strncpy (w_state, str2, 2);
                strncpy (w_zip, num9, 9);

            status = OCIStmtExecute(tpcsvc, curw, errhp, (ub4) 1, (ub4) 0,
                         (CONST OCISnapshot*) 0, (OCISnapshot*) 0,
                         (ub4) OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
            if (status != OCI_SUCCESS) {
                fprintf (stderr, "Error at ware %d\n", loop);
                OCIERROR(errhp, status);
                quit ();
                    exit (1);
                }
            }
        }

        end_time = gettime ();
        end_cpu = getcpu ();
        fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec. (%10.2f cpu)\n\n",
                 nrows, end_time - begin_time, end_cpu - begin_cpu);
    }

/*---------------------------------------------------------------+
 | Load the DISTRICT table.                   |
 +---------------------------------------------------------------*/

    if (do_A || do_d) {
        nrows = (eware - bware + 1) * DISTFAC;

        fprintf (stderr, "Loading/generating district: w%d - w%d (%d rows)\n",
                 bware, eware, nrows);
    begin_time = gettime ();
    begin_cpu = getcpu ();

    dwid = bware - 1;

    for (row = 0; row < nrows; ) {
      dwid++;

      for (i = 0; i < DISTARR; i++, row++) {
        d_tax[i] = (float) ((lrand48 () % 2001) * 0.0001);
        randstr (d_name[i], 6, 10);
        randstr (d_street_1[i], 10, 20);
        randstr (d_street_2[i], 10, 20);
        randstr (d_city[i], 10, 20);
        randstr (str2, 2, 2);
        randnum (num9, 9);
        num9[4] = num9[5] = num9[6] = num9[7] = num9[8] = '1';

        if (gen) {
          printf ("%d %d 3000000 %6.4f 3001 %s %s %s %s %s %s\n",
                  i + 1, dwid, d_tax[i], d_name[i], d_street_1[i],
             d_street_2[i], d_city[i], str2, num9 );
        }
        else {
          d_id[i] = i + 1;
          d_w_id[i] = dwid;
          strncpy (d_state[i], str2, 2);
          strncpy (d_zip[i], num9, 9);
        }
      }

      if (gen) {
        fflush (stdout);
      }
      else {
        status = OCIStmtExecute(tpcsvc, curd, errhp, (ub4) DISTARR, (ub4) 0,
                    (CONST OCISnapshot*) 0, (OCISnapshot*) 0,
                    (ub4) OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
        if (status != OCI_SUCCESS) {
       fprintf (stderr, "Aborted at ware %d, dist 1\n", dwid);
             OCIERROR(errhp, status);
             quit ();
             exit (1);
        }
      }
    }

    end_time = gettime ();
    end_cpu = getcpu ();
    fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec. (%10.2f cpu)\n\n",
          nrows, end_time - begin_time, end_cpu - begin_cpu);
  }

/*--------------------------------------------------------------+
 | Load the CUSTOMER table.                                     |
 +--------------------------------------------------------------*/

  if (do_A || do_c) {

    nrows = (eware - bware + 1) * CUSTFAC * DISTFAC;

    fprintf (stderr, "Loading/generating customer: w%d - w%d (%d rows)\n    ",
          bware, eware, nrows);

    if (getenv("tpcc_hash_overflow")) {
      fprintf(stderr,"Hash overflow is enabled\n");
      OCIHandleAlloc(tpcenv, (dvoid **)&curi, OCI_HTYPE_STMT, 0, (dvoid**)0);
      sprintf ((char *) stmbuf, SQLTXTENHA);
      OCIStmtPrepare(curi, errhp, stmbuf, strlen((char *)stmbuf),
                  OCI_NTV_SYNTAX, OCI_DEFAULT);
      OCIERROR(errhp,OCIStmtExecute(tpcsvc, curi, errhp,1,0,0,0,OCI_DEFAULT));
      OCIHandleFree(curi, OCI_HTYPE_STMT);
      fprintf (stderr,"Customer loaded for horizontal partitioning\n");
    }
    else
    {
      fprintf (stderr,"Customer not loaded for horizontal partitioning\n");
    }
    begin_time = gettime ();
    begin_cpu = getcpu ();

    cid = 0;
    cdid = 1;
    cwid = bware;
    loopcount = 0;

    for (row = 0; row < nrows; ) {
      for (i = 0; i < CUSTARR; i++, row++) {
        cid++;
        if (cid > CUSTFAC) {        /* cycle cust id */
          cid = 1;                  /* cheap mod */
          cdid++;                   /* shift dist cycle */
          if (cdid > DISTFAC) {
            cdid = 1;
            cwid++;                 /* shift ware cycle */
          }
        }
        c_id[i] = cid;
        c_d_id[i] = cdid;
        c_w_id[i] = cwid;
        if (cid <= 1000)
          randlastname (c_last[i], cid - 1);
        else
          randlastname (c_last[i], NURand (255, 0, 999, CNUM1));
        c_credit[i][1] = 'C';
        if (lrand48 () % 10)
          c_credit[i][0] = 'G';
        else
          c_credit[i][0] = 'B';
        c_discount[i] = (float)((lrand48 () % 5001) * 0.0001);
        randstr (c_first[i], 8, 16);
        randstr (c_street_1[i], 10, 20);
        randstr (c_street_2[i], 10, 20);
        randstr (c_city[i], 10, 20);
        randstr (str2, 2, 2);
        randnum (num9, 9);
        num9[4] = num9[5] = num9[6] = num9[7] = num9[8] = '1';
        randnum (num16, 16);
        randstr (c_data[i], 300, 500);

        if (gen) {
          printf ("%d %d %d %s OE %s %s %s %s %s %s %s %s %cC 5000000 %6.4f -1000 1000
1 0 %s\n",
                  cid, cdid, cwid, c_first[i], c_last[i],
                  c_street_1[i], c_street_2[i], c_city[i], str2, num9,
                  num16, sdate, c_credit[i][0], c_discount[i], c_data[i]);
        }
        else {
          strncpy (c_state[i], str2, 2);
          strncpy (c_zip[i], num9, 9);
          strncpy (c_phone[i], num16, 16);
        }
      }

      if (gen) {
        fflush (stdout);
      }
      else {
        status = OCIStmtExecute(tpcsvc, curc, errhp, (ub4) CUSTARR, (ub4) 0,
                    (CONST OCISnapshot*) 0, (OCISnapshot*) 0,
                    (ub4) OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);

        if (status != OCI_SUCCESS) {
       fprintf (stderr, "Aborted at w_id %d, d_id %d, c_id %d\n",
                    c_w_id[0], c_d_id[0], c_id[0]);
             OCIERROR(errhp, status);
             quit ();
             exit (1);
        }
      }

      if ((++loopcount) % 50)
        fprintf (stderr, ".");
      else
        fprintf (stderr, " %d rows committed\n    ", row);
    }

    end_time = gettime ();
    end_cpu = getcpu ();
    fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec. (%10.2f cpu)\n\n",
          nrows, end_time - begin_time, end_cpu - begin_cpu);
    if (getenv("tpcc_hash_overflow")) {
      fprintf(stderr,"Hash overflow is disabled\n");
      OCIHandleAlloc(tpcenv, (dvoid **)&curi, OCI_HTYPE_STMT, 0, (dvoid**)0);
      sprintf ((char *) stmbuf, SQLTXTDIHA);
      OCIStmtPrepare(curi, errhp, stmbuf, strlen((char *)stmbuf),
                  OCI_NTV_SYNTAX, OCI_DEFAULT);
      OCIERROR(errhp,OCIStmtExecute(tpcsvc, curi, errhp,1,0,0,0,OCI_DEFAULT));
      OCIHandleFree(curi, OCI_HTYPE_STMT);
    }
  }


/*--------------------------------------------------------------+
 | Load the ITEM table.                                         |
 +--------------------------------------------------------------*/

  if (do_A || do_i) {
    nrows = ITEMFAC;

    fprintf (stderr, "Loading/generating item: (%d rows)\n    ", nrows);
    begin_time = gettime ();
    begin_cpu = getcpu ();

    loopcount = 0;

    for (row = 0; row < nrows; ) {
      for (i = 0; i < ITEMARR; i++, row++) {
        i_im_id[i] = (lrand48 () % 10000) + 1;
        i_price[i] = ((lrand48 () % 9901) + 100);
        randstr (i_name[i], 14, 24);
        randdatastr (i_data[i], 26, 50);

        if (gen) {
          printf ("%d %d %s %d %s\n", row + 1, i_im_id[i], i_name[i],
                  i_price[i], i_data[i]);
        }
        else {
          i_id[i] = row + 1;
        }
      }

      if (gen) {
        fflush (stdout);
      }
      else {
        status = OCIStmtExecute(tpcsvc, curi, errhp, (ub4) ITEMARR, (ub4) 0,
                  (CONST OCISnapshot*) 0, (OCISnapshot*) 0,
                  (ub4) OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
        if (status != OCI_SUCCESS) {
        fprintf (stderr, "Aborted at i_id %d\n", i_id[0]);
            OCIERROR(errhp, status);
            quit ();
            exit (1);
        }
      }

      if ((++loopcount) % 50)
        fprintf (stderr, ".");
      else
        fprintf (stderr, " %d rows committed\n    ", row);
    }

    end_time = gettime ();
    end_cpu = getcpu ();
    fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec. (%10.2f cpu)\n\n",
           nrows, end_time - begin_time, end_cpu - begin_cpu);
  }


/*------------------------------------------------------------------+
 | Load the STOCK table.                    |
 +------------------------------------------------------------------*/

  if (do_A || do_s) {

    nrows = (eware - bware + 1) * STOCFAC;

    fprintf (stderr, "Loading/generating stock: w%d - w%d (%d rows)\n    ",
           bware, eware, nrows);

    begin_time = gettime ();
    begin_cpu = getcpu ();

    sid = 0;
    swid = bware;
    loopcount = 0;

    for (row = 0; row < nrows; ) {
        /* added row < nrows condition on next line - alex.ni */
     for (i = 0; (i < STOCARR) && (row < nrows); i++, row++) {
        if (++sid > STOCFAC) {         /* cheap mod */
          sid = 1;
          swid++;
        }
        s_quantity[i] = (lrand48 () % 91) + 10;
        randstr (str24[0], 24, 24);
        randstr (str24[1], 24, 24);
        randstr (str24[2], 24, 24);
        randstr (str24[3], 24, 24);
        randstr (str24[4], 24, 24);
        randstr (str24[5], 24, 24);
        randstr (str24[6], 24, 24);
        randstr (str24[7], 24, 24);
        randstr (str24[8], 24, 24);
        randstr (str24[9], 24, 24);
        randdatastr (s_data[i], 26, 50);

        if (gen) {
          printf ("%d %d %d %s %s %s %s %s %s %s %s %s %s 0 0 0 %s\n",
                  sid, swid, s_quantity[i], str24[0], str24[1], str24[2],
                  str24[3], str24[4], str24[5], str24[6], str24[7],
                  str24[8], str24[9], s_data[i]);
        }
        else {
          s_i_id[i] = sid;
          s_w_id[i] = swid;
          strncpy (s_dist_01[i], str24[0], 24);
          strncpy (s_dist_02[i], str24[1], 24);
          strncpy (s_dist_03[i], str24[2], 24);
          strncpy (s_dist_04[i], str24[3], 24);
          strncpy (s_dist_05[i], str24[4], 24);
          strncpy (s_dist_06[i], str24[5], 24);
          strncpy (s_dist_07[i], str24[6], 24);
          strncpy (s_dist_08[i], str24[7], 24);
          strncpy (s_dist_09[i], str24[8], 24);
          strncpy (s_dist_10[i], str24[9], 24);
        }
      }

      if (gen) {
        fflush (stdout);
      }
      else {
     /* Changed to STOCKARR to i - alex.ni */
        status = OCIStmtExecute(tpcsvc, curs, errhp, (ub4) i, (ub4) 0,
                  (CONST OCISnapshot*) 0, (OCISnapshot*) 0,
                  (ub4) OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
        if (status != OCI_SUCCESS) {
          fprintf (stderr, "Aborted at w_id %d, s_i_id %d\n", s_w_id[0], s_i_id[0]);
            OCIERROR(errhp, status);
            quit ();
            exit (1);
        }
      }

      if ((++loopcount) % 50)
        fprintf (stderr, ".");
      else
        fprintf (stderr, " %d rows committed\n    ", row);
    }

    end_time = gettime ();
    end_cpu = getcpu ();
    fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec. (%10.2f cpu)\n\n",
           nrows, end_time - begin_time, end_cpu - begin_cpu);
  }

/*------------------------------------------------------------------+
 | Load the STOCK table (cluster around s_i_id).         |
 +------------------------------------------------------------------*/

  if (do_S) {

    nrows = (eitem - bitem + 1) * (eware - bware + 1);

    fprintf (stderr, "Loading/generating stock: i%d - i%d, w%d - w%d (%d rows)\n    ",
           bitem, eitem, bware, eware, nrows);

    begin_time = gettime ();
    begin_cpu = getcpu ();

    sid = bitem;
    swid = bware - 1;
    loopcount = 0;

    for (row = 0; row < nrows; ) {
      for (i = 0; i < STOCARR; i++, row++) {
        if (++swid > eware) {          /* cheap mod */
          swid = bware;
          sid++;
        }
        s_quantity[i] = (lrand48 () % 91) + 10;
        randstr (str24[0], 24, 24);
        randstr (str24[1], 24, 24);
        randstr (str24[2], 24, 24);
        randstr (str24[3], 24, 24);
        randstr (str24[4], 24, 24);
        randstr (str24[5], 24, 24);
        randstr (str24[6], 24, 24);
        randstr (str24[7], 24, 24);
        randstr (str24[8], 24, 24);
        randstr (str24[9], 24, 24);
        randdatastr (s_data[i], 26, 50);

        if (gen) {
          printf ("%d %d %d %s %s %s %s %s %s %s %s %s %s 0 0 0 %s\n",
                  sid, swid, s_quantity[i], str24[0], str24[1], str24[2],
                  str24[3], str24[4], str24[5], str24[6], str24[7],
                  str24[8], str24[9], s_data[i]);
        }
        else {
          s_i_id[i] = sid;
```

```
            s_w_id[i] = swid;
            strncpy (s_dist_01[i], str24[0], 24);
            strncpy (s_dist_02[i], str24[1], 24);
            strncpy (s_dist_03[i], str24[2], 24);
            strncpy (s_dist_04[i], str24[3], 24);
            strncpy (s_dist_05[i], str24[4], 24);
            strncpy (s_dist_06[i], str24[5], 24);
            strncpy (s_dist_07[i], str24[6], 24);
            strncpy (s_dist_08[i], str24[7], 24);
            strncpy (s_dist_09[i], str24[8], 24);
            strncpy (s_dist_10[i], str24[9], 24);
          }
        }

        if (gen) {
          fflush (stdout);
        }
        else {
          status = OCIStmtExecute(tpcsvc, curs, errhp, (ub4) STOCARR, (ub4) 0,
                    (CONST OCISnapshot*) 0, (OCISnapshot*) 0,
                    (ub4) OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
          if (status != OCI_SUCCESS) {
              fprintf (stderr, "Aborted at w_id %d, s_i_id %d\n", s_w_id[0], s_i_id[0]);
              OCIERROR(errhp, status);
              quit ();
              exit (1);
          }
        }

        if ((++loopcount) % 50)
          fprintf (stderr, ".");
        else
          fprintf (stderr, " %d rows committed\n   ", row);
      }

      end_time = gettime ();
      end_cpu = getcpu ();
      fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec. (%10.2f cpu)\n\n",
            nrows, end_time - begin_time, end_cpu - begin_cpu);
    }

  /*-------------------------------------------------------------+
   | Load the HISTORY table.                    |
   +-------------------------------------------------------------*/

    if (do_A || do_h) {
      nrows = (eware - bware + 1) * HISTFAC;

      fprintf (stderr, "Loading/generating history: w%d - w%d (%d rows)\n   ",
            bware, eware, nrows);

      begin_time = gettime ();
      begin_cpu = getcpu ();

      cid = 0;
      cdid = 1;
      cwid = bware;
      loopcount = 0;

      for (row = 0; row < nrows; ) {
        for (i = 0; i < HISTARR; i++, row++) {
          cid++;
          if (cid > CUSTFAC) {        /* cycle cust id */
            cid = 1;                  /* cheap mod */
            cdid++;                   /* shift district cycle */
            if (cdid > DISTFAC) {
              cdid = 1;
              cwid++;                 /* shift warehouse cycle */
            }
          }
          h_c_id[i] = cid;
          h_d_id[i] = cdid;
          h_w_id[i] = cwid;
          randstr (h_data[i], 12, 24);
          if (gen) {
            printf ("%d %d %d %d %d %s 1000 %s\n", cid, cdid, cwid, cdid,
                  cwid, sdate, h_data[i]);
          }
        }

        if (gen) {
          fflush (stdout);
        }
        else {
          status = OCIStmtExecute(tpcsvc, curh, errhp, (ub4) HISTARR, (ub4) 0,
                    (CONST OCISnapshot*) 0, (OCISnapshot*) 0,
                    (ub4) OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
          if (status != OCI_SUCCESS) {
           fprintf (stderr, "Aborted at w_id %d, d_id %d, c_id %d\n",
                    h_w_id[0], h_d_id[0], h_c_id[0]);
              OCIERROR(errhp, status);
              quit ();
              exit (1);
          }
        }

        if ((++loopcount) % 50)
          fprintf (stderr, ".");
        else
          fprintf (stderr, " %d rows committed\n   ", row);
      }

      end_time = gettime ();
      end_cpu = getcpu ();
      fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec. (%10.2f cpu)\n\n",
            nrows, end_time - begin_time, end_cpu - begin_cpu);
    }

  /*-------------------------------------------------------------+
   | Load the ORDERS and ORDER-LINE table.          |
   +-------------------------------------------------------------*/

    if (do_A || do_o) {

      int batch_olcnt;

      nrows = (eware - bware + 1) * ORDEFAC * DISTFAC;

      fprintf (stderr, "Loading/generating orders and order-line: w%d - w%d (%d ord, ~%d ordl)\n   ",
            bware, eware, nrows, nrows * 10);

      begin_time = gettime ();
      begin_cpu = getcpu ();

      cid = 0;
      cdid = 1;
      cwid = bware;
      loopcount = 0;

      for (row = 0; row < nrows; ) {

        batch_olcnt = 0;

        for (i = 0; i < ORDEARR; i++, row++) {
          cid++;
          if (cid > ORDEFAC) {        /* cycle cust id */
            cid = 1;                  /* cheap mod */
            cdid++;                   /* shift district cycle */
            if (cdid > DISTFAC) {
              cdid = 1;
              cwid++;                 /* shift warehouse cycle */
            }
          }
          o_carrier_id[i] = lrand48 () % 10 + 1;
          o_ol_cnt[i] = olcnt = lrand48 () % 11 + 5;

          if (gen) {
            if (cid < 2101) {
              printf ("%d %d %d %d %s %d %d 1\n", cid, cdid, cwid,
                    randperm3000[cid - 1], sdate,o_carrier_id[i],
                    o_ol_cnt[i]);
            }
            else {
            /* set carrierid to 11 instead of null */
              printf ("%d %d %d %d %s 11 %d 1\n", cid, cdid, cwid,
                    randperm3000[cid - 1], sdate, o_ol_cnt[i]);
            }
          }
          else {
            o_id[i] = cid;
            o_d_id[i] = cdid;
            o_w_id[i] = cwid;
            o_c_id[i] = randperm3000[cid - 1];
            if (cid >= 2101 ) {
               o_carrier_id[i] = 11;
            }
          }

          for (j = 0; j < o_ol_cnt[i]; j++, batch_olcnt++ ) {
            ol_i_id[batch_olcnt] = sid = lrand48 () % 100000 + 1;
            if (cid < 2101)
              ol_amount[batch_olcnt] = 0;
            else
              ol_amount[batch_olcnt] = (lrand48 () % 999999 + 1) ;
            randstr (str24[j], 24, 24);

            if (gen) {
              if (cid < 2101) {
                fprintf (olfp, "%d %d %d %d %s %d %d 5 %ld %s\n", cid,
                      cdid, cwid, j + 1, sdate, ol_i_id[batch_olcnt], cwid,
                      ol_amount[batch_olcnt], str24[j]);
              }
```

```
                else {
                    /* Insert a default date instead of null date */
                    fprintf (olfp, "%d %d %d %d 01-Jan-1811 %d %d 5 %ld %s\n", cid,
                            cdid, cwid, j + 1, ol_i_id[batch_olcnt], cwid,
                            ol_amount[batch_olcnt], str24[j]);
                }
              }
              else {
                ol_o_id[batch_olcnt] = cid;
                ol_d_id[batch_olcnt] = cdid;
                ol_w_id[batch_olcnt] = cwid;
                ol_number[batch_olcnt] = j + 1;
                ol_supply_w_id[batch_olcnt] = cwid;
                strncpy (ol_dist_info[batch_olcnt], str24[j], 24);
              }
            }
            if (gen) {
              fflush (olfp);
            }
          }

          o_cnt =  ORDEARR;
          ol_cnt =  batch_olcnt;

          for (j = 0; j < batch_olcnt; j++) {
            ol_o_id_len[j] = sizeof(int);
            ol_d_id_len[j] = sizeof(int);
            ol_w_id_len[j] = sizeof(int);
            ol_number_len[j] = sizeof(int);
            ol_i_id_len[j] = sizeof(int);
            ol_supply_w_id_len[j] = sizeof(int);
            ol_dist_info_len[j] = 24;
            ol_amount_len[j] = sizeof(int);
          }
          for (j = batch_olcnt; j < 15*ORDEARR; j++) {
            ol_o_id_len[j] = 0;
            ol_d_id_len[j] = 0;
            ol_w_id_len[j] = 0;
            ol_number_len[j] = 0;
            ol_i_id_len[j] = 0;
            ol_supply_w_id_len[j] = 0;
            ol_dist_info_len[j] = 0;
            ol_amount_len[j] = 0;
          }

          o_id_clen = ORDEARR;
          o_d_id_clen = ORDEARR;
          o_w_id_clen = ORDEARR;
          o_c_id_clen = ORDEARR;
          o_carrier_id_clen = ORDEARR;
          o_ol_cnt_clen = ORDEARR;

          ol_o_id_clen = batch_olcnt;
          ol_d_id_clen = batch_olcnt;
          ol_w_id_clen = batch_olcnt;
          ol_number_clen = batch_olcnt;
          ol_i_id_clen = batch_olcnt;
          ol_supply_w_id_clen = batch_olcnt;
          ol_dist_info_clen = batch_olcnt;
          ol_amount_clen = batch_olcnt;

          OCIERROR(errhp, OCIStmtExecute(tpcsvc, curo1, errhp, (ub4) 1, (ub4) 0,
                          (CONST OCISnapshot*) 0, (OCISnapshot*) 0,
                          (ub4) OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS ));

          if ((++loopcount) % 50) {
               fprintf (stderr, ".");
          } else {
               fprintf (stderr, " %d orders committed\n    ", row);
          }
        }

        end_time = gettime ();
        end_cpu = getcpu ();
        fprintf (stderr, "Done.  %d orders loaded/generated in %10.2f sec. (%10.2f cpu)\n\n",
                 nrows, end_time - begin_time, end_cpu - begin_cpu);
    }

/*--------------------------------------------------------------+
 | Load the NEW-ORDER table.                  |
 +--------------------------------------------------------------*/

    if (do_A || do_n) {
        nrows = (eware - bware + 1) * NEWOFAC * DISTFAC;

        fprintf (stderr, "Loading/generating new-order: w%d - w%d (%d rows)\n    ",
                 bware, eware, nrows);

        begin_time = gettime ();
        begin_cpu = getcpu ();

        cid = 0;
        cdid = 1;
        cwid = bware;
        loopcount = 0;

        for (row = 0; row < nrows; ) {
          for (i = 0; i < NEWOARR; i++, row++) {
            cid++;
            if (cid > NEWOFAC) {
              cid = 1;
              cdid++;
              if (cdid > DISTFAC) {
                cdid = 1;
                cwid++;
              }
            }

            if (gen) {
              printf ("%d %d %d\n", cid + 2100, cdid, cwid);
            }
            else {
              no_o_id[i] = cid + 2100;
              no_d_id[i] = cdid;
              no_w_id[i] = cwid;
            }
          }

          if (gen) {
            fflush (stdout);
          }
          else {
            status = OCIStmtExecute(tpcsvc, curno, errhp, (ub4) NEWOARR, (ub4) 0,
                             (CONST OCISnapshot*) 0, (OCISnapshot*) 0,
                             (ub4) OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
            if (status != OCI_SUCCESS) {
          fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n", cwid, cdid, cid + 2100);
                OCIERROR(errhp, status);
                quit ();
                exit (1);
            }
          }

          if ((++loopcount) % 45)
            fprintf (stderr, ".");
          else
            fprintf (stderr, " %d rows committed\n    ", row);
        }

        end_time = gettime ();
        end_cpu = getcpu ();
        fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec. (%10.2f cpu)\n\n",
                 nrows, end_time - begin_time, end_cpu - begin_cpu);
    }

/*--------------------------------------------------------------+
 | clean up and exit.                         |
 +--------------------------------------------------------------*/

    if (olfp)
        fclose (olfp);
    if (!gen)
        quit ();
    exit (0);

}

void initperm ()
{
    int i;
    int pos;
    int temp;

    /* init randperm3000 */

    for (i = 0; i < 3000; i++)
        randperm3000[i] = i + 1;
    for (i = 3000; i > 0; i--) {
        pos = lrand48 () % i;
        temp = randperm3000[i - 1];
        randperm3000[i - 1] = randperm3000[pos];
        randperm3000[pos] = temp;
    }
}

void randstr (str, x, y)
char *str;
int x;
int y;
{
    int i, j;
    int len;
    len = (lrand48 () % (y - x + 1)) + x;
    for (i = 0; i < len; i++) {
        j = lrand48 () % 62;
        if (j < 26)
            str[i] = (char) (j + 'a');
        else if (j < 52)
            str[i] = (char) (j - 26 + 'A');
        else
            str[i] = (char) (j - 52 + '0');
    }
    str[len] = '\0';

}

void randdatastr (str, x, y)
char *str;
int x;
int y;
{
    int i, j;
    int len;
    int pos;

    len = (lrand48 () % (y - x + 1)) + x;
    for (i = 0; i < len; i++) {
        j = lrand48 () % 62;
        if (j < 26)
            str[i] = (char) (j + 'a');
        else if (j < 52)
            str[i] = (char) (j - 26 + 'A');
        else
            str[i] = (char) (j - 52 + '0');
    }
    str[len] = '\0';
    if ((lrand48 () % 10) == 0) {
        pos = (lrand48 () % (len - 8));
        str[pos] = 'O';
        str[pos + 1] = 'R';
        str[pos + 2] = 'I';
        str[pos + 3] = 'G';
        str[pos + 4] = 'I';
        str[pos + 5] = 'N';
        str[pos + 6] = 'A';
        str[pos + 7] = 'L';
    }
}

void randnum (str, len)
char *str;
int len;
{
    int i;

    for (i = 0; i < len; i++)
        str[i] = (char) (lrand48 () % 10 + '0');
    str[len] = '\0';

}

void randlastname (str, id)
char *str;
int id;
{
    id = id % 1000;
    strcpy (str, lastname[id / 100]);
    strcat (str, lastname[(id / 10) % 10]);
    strcat (str, lastname[id % 10]);
}

int NURand (A, x, y, cnum)
int A, x, y, cnum;
{
    int a, b;

    a = lrand48 () % (A + 1);
    b = (lrand48 () % (y - x + 1)) + x;
    return ((((a | b) + cnum) % (y - x + 1)) + x);

}

void sysdate (sdate)
char *sdate;
{
    time_t tp;
    struct tm *tmptr;

    time (&tp);
    tmptr = localtime (&tp);
    strftime (sdate, 29, "%d-%b-%Y", tmptr);
}

int ocierror(fname, lineno, errhp, status)
char *fname;
int lineno;
OCIError *errhp;
sword status;
{
  text errbuf[512];
  sb4 errcode;
  sb4 lstat;
  ub4 recno=2;

  switch (status) {
  case OCI_SUCCESS:
    break;
  case OCI_SUCCESS_WITH_INFO:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_SUCCESS_WITH_INFO\n");
    lstat = OCIErrorGet (errhp, recno++, (text *) NULL, &errcode, errbuf,
                         (ub4) sizeof(errbuf), OCI_HTYPE_ERROR);
    fprintf(stderr,"Error - %s\n", errbuf);
    break;
  case OCI_NEED_DATA:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_NEED_DATA\n");
    return (IRRECERR);
  case OCI_NO_DATA:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_NO_DATA\n");
    return (IRRECERR);
  case OCI_ERROR:
    lstat = OCIErrorGet (errhp, (ub4) 1,
            (text *) NULL, &errcode, errbuf,
             (ub4) sizeof(errbuf), OCI_HTYPE_ERROR);
    if (errcode == NOT_SERIALIZABLE) return (errcode);
    if (errcode == SNAPSHOT_TOO_OLD) return (errcode);
    while (lstat != OCI_NO_DATA)
    {
      fprintf(stderr,"Module %s Line %d\n", fname, lineno);
      fprintf(stderr,"Error - %s\n", errbuf);
      lstat = OCIErrorGet (errhp, recno++, (text *) NULL, &errcode, errbuf,
                           (ub4) sizeof(errbuf), OCI_HTYPE_ERROR);
    }
    return (errcode);
  case OCI_INVALID_HANDLE:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_INVALID_HANDLE\n");
    exit(-1);
  case OCI_STILL_EXECUTING:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_STILL_EXECUTE\n");
    return (IRRECERR);
  case OCI_CONTINUE:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_CONTINUE\n");
    return (IRRECERR);
  default:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Status - %s\n", status);
    return (IRRECERR);
  }
  return (RECOVERR);
}



--------------------------------------------------
                   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;
```

```
------------------------------------------------
                 mkraw_run.sh
------------------------------------------------

raw /home/oracle/dev/raw/stok_0_0 /dev/cciss/c0d0p1
raw /home/oracle/dev/raw/stok_0_1 /dev/cciss/c1d0p1
raw /home/oracle/dev/raw/stok_0_2 /dev/cciss/c2d0p1
raw /home/oracle/dev/raw/stok_0_3 /dev/cciss/c3d0p1
raw /home/oracle/dev/raw/stok_0_4 /dev/cciss/c4d0p1
raw /home/oracle/dev/raw/stok_0_5 /dev/cciss/c5d0p1
raw /home/oracle/dev/raw/stok_0_6 /dev/cciss/c6d0p1
raw /home/oracle/dev/raw/stok_0_7 /dev/cciss/c7d0p1
raw /home/oracle/dev/raw/stok_0_8 /dev/cciss/c0d0p2
raw /home/oracle/dev/raw/stok_0_9 /dev/cciss/c1d0p2
raw /home/oracle/dev/raw/stok_0_10 /dev/cciss/c2d0p2
raw /home/oracle/dev/raw/stok_0_11 /dev/cciss/c3d0p2
raw /home/oracle/dev/raw/stok_0_12 /dev/cciss/c4d0p2
raw /home/oracle/dev/raw/stok_0_13 /dev/cciss/c5d0p2
raw /home/oracle/dev/raw/stok_0_14 /dev/cciss/c6d0p2
raw /home/oracle/dev/raw/stok_0_15 /dev/cciss/c7d0p2
raw /home/oracle/dev/raw/stok_0_16 /dev/cciss/c0d0p3
raw /home/oracle/dev/raw/stok_0_17 /dev/cciss/c1d0p3
raw /home/oracle/dev/raw/stok_0_18 /dev/cciss/c2d0p3
raw /home/oracle/dev/raw/stok_0_19 /dev/cciss/c3d0p3
raw /home/oracle/dev/raw/stok_0_20 /dev/cciss/c4d0p3
raw /home/oracle/dev/raw/stok_0_21 /dev/cciss/c5d0p3
raw /home/oracle/dev/raw/stok_0_22 /dev/cciss/c6d0p3
raw /home/oracle/dev/raw/stok_0_23 /dev/cciss/c7d0p3
raw /home/oracle/dev/raw/stok_0_24 /dev/cciss/c0d0p5
raw /home/oracle/dev/raw/stok_0_25 /dev/cciss/c1d0p5
raw /home/oracle/dev/raw/stok_0_26 /dev/cciss/c2d0p5
raw /home/oracle/dev/raw/stok_0_27 /dev/cciss/c3d0p5
raw /home/oracle/dev/raw/stok_0_28 /dev/cciss/c4d0p5
raw /home/oracle/dev/raw/stok_0_29 /dev/cciss/c5d0p5
raw /home/oracle/dev/raw/stok_0_30 /dev/cciss/c6d0p5
raw /home/oracle/dev/raw/stok_0_31 /dev/cciss/c7d0p5
raw /home/oracle/dev/raw/stok_0_32 /dev/cciss/c0d0p6
raw /home/oracle/dev/raw/stok_0_33 /dev/cciss/c1d0p6
raw /home/oracle/dev/raw/stok_0_34 /dev/cciss/c2d0p6
raw /home/oracle/dev/raw/stok_0_35 /dev/cciss/c3d0p6
raw /home/oracle/dev/raw/stok_0_36 /dev/cciss/c4d0p6
raw /home/oracle/dev/raw/stok_0_37 /dev/cciss/c5d0p6
raw /home/oracle/dev/raw/stok_0_38 /dev/cciss/c6d0p6
raw /home/oracle/dev/raw/stok_0_39 /dev/cciss/c7d0p6
raw /home/oracle/dev/raw/stok_0_40 /dev/cciss/c0d0p7
raw /home/oracle/dev/raw/stok_0_41 /dev/cciss/c1d0p7
raw /home/oracle/dev/raw/stok_0_42 /dev/cciss/c2d0p7
raw /home/oracle/dev/raw/stok_0_43 /dev/cciss/c3d0p7
raw /home/oracle/dev/raw/stok_0_44 /dev/cciss/c4d0p7
raw /home/oracle/dev/raw/stok_0_45 /dev/cciss/c5d0p7
raw /home/oracle/dev/raw/stok_0_46 /dev/cciss/c6d0p7
raw /home/oracle/dev/raw/stok_0_47 /dev/cciss/c7d0p7
raw /home/oracle/dev/raw/stok_0_48 /dev/cciss/c0d0p8
raw /home/oracle/dev/raw/stok_0_49 /dev/cciss/c1d0p8
raw /home/oracle/dev/raw/stok_0_50 /dev/cciss/c2d0p8
raw /home/oracle/dev/raw/stok_0_51 /dev/cciss/c3d0p8
raw /home/oracle/dev/raw/stok_0_52 /dev/cciss/c4d0p8
raw /home/oracle/dev/raw/stok_0_53 /dev/cciss/c5d0p8
raw /home/oracle/dev/raw/stok_0_54 /dev/cciss/c6d0p8
raw /home/oracle/dev/raw/stok_0_55 /dev/cciss/c7d0p8
raw /home/oracle/dev/raw/cust_0_0 /dev/cciss/c0d0p9
raw /home/oracle/dev/raw/cust_0_1 /dev/cciss/c1d0p9
raw /home/oracle/dev/raw/cust_0_2 /dev/cciss/c2d0p9
raw /home/oracle/dev/raw/cust_0_3 /dev/cciss/c3d0p9
raw /home/oracle/dev/raw/cust_0_4 /dev/cciss/c4d0p9
raw /home/oracle/dev/raw/cust_0_5 /dev/cciss/c5d0p9
raw /home/oracle/dev/raw/cust_0_6 /dev/cciss/c6d0p9
raw /home/oracle/dev/raw/cust_0_7 /dev/cciss/c7d0p9
raw /home/oracle/dev/raw/cust_0_8 /dev/cciss/c0d0p10
raw /home/oracle/dev/raw/cust_0_9 /dev/cciss/c1d0p10
raw /home/oracle/dev/raw/cust_0_10 /dev/cciss/c2d0p10
raw /home/oracle/dev/raw/cust_0_11 /dev/cciss/c3d0p10
raw /home/oracle/dev/raw/cust_0_12 /dev/cciss/c4d0p10
raw /home/oracle/dev/raw/cust_0_13 /dev/cciss/c5d0p10
raw /home/oracle/dev/raw/cust_0_14 /dev/cciss/c6d0p10
raw /home/oracle/dev/raw/cust_0_15 /dev/cciss/c7d0p10
raw /home/oracle/dev/raw/cust_0_16 /dev/cciss/c0d0p11
raw /home/oracle/dev/raw/cust_0_17 /dev/cciss/c1d0p11
raw /home/oracle/dev/raw/cust_0_18 /dev/cciss/c2d0p11
raw /home/oracle/dev/raw/cust_0_19 /dev/cciss/c3d0p11
raw /home/oracle/dev/raw/cust_0_20 /dev/cciss/c4d0p11
raw /home/oracle/dev/raw/cust_0_21 /dev/cciss/c5d0p11
raw /home/oracle/dev/raw/cust_0_22 /dev/cciss/c6d0p11
raw /home/oracle/dev/raw/cust_0_23 /dev/cciss/c7d0p11
raw /home/oracle/dev/raw/cust_0_24 /dev/cciss/c0d0p12
raw /home/oracle/dev/raw/cust_0_25 /dev/cciss/c1d0p12
raw /home/oracle/dev/raw/cust_0_26 /dev/cciss/c2d0p12
raw /home/oracle/dev/raw/cust_0_27 /dev/cciss/c3d0p12
raw /home/oracle/dev/raw/cust_0_28 /dev/cciss/c4d0p12
raw /home/oracle/dev/raw/cust_0_29 /dev/cciss/c5d0p12
raw /home/oracle/dev/raw/cust_0_30 /dev/cciss/c6d0p12
raw /home/oracle/dev/raw/cust_0_31 /dev/cciss/c7d0p12
raw /home/oracle/dev/raw/cust_0_32 /dev/cciss/c0d0p13
raw /home/oracle/dev/raw/cust_0_33 /dev/cciss/c1d0p13
raw /home/oracle/dev/raw/cust_0_34 /dev/cciss/c2d0p13
raw /home/oracle/dev/raw/cust_0_35 /dev/cciss/c3d0p13
raw /home/oracle/dev/raw/cust_0_36 /dev/cciss/c4d0p13
raw /home/oracle/dev/raw/cust_0_37 /dev/cciss/c5d0p13
raw /home/oracle/dev/raw/cust_0_38 /dev/cciss/c6d0p13
raw /home/oracle/dev/raw/cust_0_39 /dev/cciss/c7d0p13
raw /home/oracle/dev/raw/log_1 /dev/md2
raw /home/oracle/dev/raw/log_2 /dev/md3
raw /home/oracle/dev/raw/ordr_0_0 /dev/cciss/c0d1p1
raw /home/oracle/dev/raw/ordr_0_1 /dev/cciss/c1d1p1
raw /home/oracle/dev/raw/ordr_0_2 /dev/cciss/c2d1p1
raw /home/oracle/dev/raw/ordr_0_3 /dev/cciss/c3d1p1
raw /home/oracle/dev/raw/ordr_0_4 /dev/cciss/c4d1p1
raw /home/oracle/dev/raw/ordr_0_5 /dev/cciss/c5d1p1
raw /home/oracle/dev/raw/ordr_0_6 /dev/cciss/c6d1p1
raw /home/oracle/dev/raw/ordr_0_7 /dev/cciss/c7d1p1
raw /home/oracle/dev/raw/nord_0_0 /dev/cciss/c0d1p2
raw /home/oracle/dev/raw/nord_0_1 /dev/cciss/c1d1p2
raw /home/oracle/dev/raw/nord_0_2 /dev/cciss/c2d1p2
raw /home/oracle/dev/raw/nord_0_3 /dev/cciss/c3d1p2
raw /home/oracle/dev/raw/hist_0_0 /dev/cciss/c4d1p2
raw /home/oracle/dev/raw/hist_0_1 /dev/cciss/c5d1p2
raw /home/oracle/dev/raw/hist_0_2 /dev/cciss/c6d1p2
raw /home/oracle/dev/raw/hist_0_3 /dev/cciss/c7d1p2
raw /home/oracle/dev/raw/iordr2_0_0 /dev/cciss/c0d1p3
raw /home/oracle/dev/raw/iordr2_0_1 /dev/cciss/c1d1p3
raw /home/oracle/dev/raw/iordr2_0_2 /dev/cciss/c2d1p3
raw /home/oracle/dev/raw/iordr2_0_3 /dev/cciss/c3d1p3
raw /home/oracle/dev/raw/iordr2_0_4 /dev/cciss/c4d1p3
raw /home/oracle/dev/raw/iordr2_0_5 /dev/cciss/c5d1p3
raw /home/oracle/dev/raw/iordr2_0_6 /dev/cciss/c6d1p3
raw /home/oracle/dev/raw/iordr2_0_7 /dev/cciss/c7d1p3
raw /home/oracle/dev/raw/icust2_0_0 /dev/cciss/c0d1p5
raw /home/oracle/dev/raw/icust2_0_1 /dev/cciss/c1d1p5
raw /home/oracle/dev/raw/icust2_0_2 /dev/cciss/c2d1p5
raw /home/oracle/dev/raw/icust2_0_3 /dev/cciss/c3d1p5
raw /home/oracle/dev/raw/icust2_0_4 /dev/cciss/c4d1p5
raw /home/oracle/dev/raw/icust2_0_5 /dev/cciss/c5d1p5
raw /home/oracle/dev/raw/icust2_0_6 /dev/cciss/c6d1p5
raw /home/oracle/dev/raw/icust2_0_7 /dev/cciss/c7d1p5
raw /home/oracle/dev/raw/ware_0_0 /dev/cciss/c0d1p6
raw /home/oracle/dev/raw/dist_0_0 /dev/cciss/c1d1p6
raw /home/oracle/dev/raw/item_0_0 /dev/cciss/c2d1p6
raw /home/oracle/dev/raw/roll01 /dev/cciss/c3d1p6
raw /home/oracle/dev/raw/aux.df /dev/cciss/c4d1p6
raw /home/oracle/dev/raw/system_001 /dev/cciss/c5d1p6
raw /home/oracle/dev/raw/sp_0 /dev/cciss/c6d1p6
raw /home/oracle/dev/raw/control_001 /dev/cciss/c7d1p6
raw /home/oracle/dev/raw/iware_0_0 /dev/cciss/c0d1p7
raw /home/oracle/dev/raw/idist_0_0 /dev/cciss/c1d1p7
raw /home/oracle/dev/raw/icust1_0_0 /dev/cciss/c2d1p7
raw /home/oracle/dev/raw/icust1_0_1 /dev/cciss/c3d1p7
raw /home/oracle/dev/raw/iitem_0_0 /dev/cciss/c4d1p7
raw /home/oracle/dev/raw/istok_0_0 /dev/cciss/c5d1p7
raw /home/oracle/dev/raw/istok_0_1 /dev/cciss/c6d1p7
raw /home/oracle/dev/raw/istok_0_2 /dev/cciss/c7d1p7
raw /home/oracle/dev/raw/istok_0_3 /dev/cciss/c0d1p8
raw /home/oracle/dev/raw/temp_0_0 /dev/cciss/c0d1p9
raw /home/oracle/dev/raw/temp_0_1 /dev/cciss/c1d1p9
raw /home/oracle/dev/raw/temp_0_2 /dev/cciss/c2d1p9
raw /home/oracle/dev/raw/temp_0_3 /dev/cciss/c3d1p9
raw /home/oracle/dev/raw/temp_0_4 /dev/cciss/c4d1p9
raw /home/oracle/dev/raw/temp_0_5 /dev/cciss/c5d1p9
raw /home/oracle/dev/raw/temp_0_6 /dev/cciss/c6d1p9
raw /home/oracle/dev/raw/temp_0_7 /dev/cciss/c7d1p9
raw /home/oracle/dev/raw/temp_0_8 /dev/cciss/c0d1p10
raw /home/oracle/dev/raw/temp_0_9 /dev/cciss/c1d1p10
raw /home/oracle/dev/raw/temp_0_10 /dev/cciss/c2d1p10
raw /home/oracle/dev/raw/temp_0_11 /dev/cciss/c3d1p10
raw /home/oracle/dev/raw/temp_0_12 /dev/cciss/c4d1p10
raw /home/oracle/dev/raw/temp_0_13 /dev/cciss/c5d1p10
raw /home/oracle/dev/raw/temp_0_14 /dev/cciss/c6d1p10
raw /home/oracle/dev/raw/temp_0_15 /dev/cciss/c7d1p10
```

Appendix C: Tunable Parameters

SEQUENCE OF EVENTS FOR PERFORMANCE RUN

```
1.   Boot up systems clients, servers, & RTEs).
2.   Change interrupt delay on cpqarray running cfgcciss 2500.
3.   Bind interrupts to CPU 2 using intr.sh.
4.   Startup the database on the server using linux.ora.
5.   Start apache on the clients using httpd.conf.
6.   Start tuxedo on the clients using ubb_multiq_[clinet#].
7.   Set priority of Oracle processes using setrrpri.sh.
8.   Start the RTE.
9.   Adjust RTE throttle.
```

```
-------------------------------------------------
                  bash_profile (clients)
-------------------------------------------------

# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
  . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin
BASH_ENV=$HOME/.bashrc
USERNAME="root"

ulimit -u 27000

export USERNAME BASH_ENV PATH

. /home/oracle/.bash_profile
. /home/oracle/Env_client
. /home/bea/tuxedo8.1/tux.env

set -o vi
ulimit
```

```
-------------------------------------------------
                  cfgcciss.c
-------------------------------------------------

#include <stdio.h>
#include <fcntl.h>
#include <linux/cciss_ioctl.h>

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

  cciss_coalint_struct cfg_coalint_old;
  cciss_coalint_struct cfg_coalint_new;
  int fd;
  int i, delay;
  char ctrlname[20];

  if (argc<2) {
    printf("useage: %s [interrupt dealy]\n", argv[0]);
    exit(0);
  }

  delay = atoi(argv[1]);
  if (delay < 0) {
    printf("delay need to be >=0\n");
    exit(0);
  }

  for (i=0; i<9; i++) {
    sprintf(ctrlname, "/dev/cciss/c%dd0", i);

    if ((fd = open(ctrlname, O_RDWR)) == -1) {
      continue;
    }

    if (ioctl(fd, CCISS_GETINTINFO, &cfg_coalint_old) != 0) {
      printf("error in reading cciss info");
      continue;
    }

    cfg_coalint_new.delay = delay;
    cfg_coalint_new.count = 1;

    if (ioctl(fd, CCISS_SETINTINFO, &cfg_coalint_new) !=0 ||
        ioctl(fd, CCISS_GETINTINFO, &cfg_coalint_new) !=0 ) {
      printf("error in setting cciss");
      continue;
    }

    printf("ctrl #%d: interrupt delay changed from %d to %d\n",
      i, cfg_coalint_old.delay, cfg_coalint_new.delay);

    close(fd);
  }
}
```

```
-------------------------------------------------
                 chkconfig --list (server)
-------------------------------------------------

keytable        0:off 1:on  2:on  3:off 4:on  5:on  6:off
atd             0:off 1:off 2:off 3:off 4:on  5:on  6:off
syslog          0:off 1:off 2:on  3:on  4:on  5:on  6:off
gpm             0:off 1:off 2:on  3:off 4:on  5:on  6:off
sendmail        0:off 1:off 2:off 3:off 4:off 5:off 6:off
kudzu           0:off 1:off 2:off 3:on  4:on  5:on  6:off
netdump-server  0:off 1:off 2:off 3:off 4:off 5:off 6:off
netfs           0:off 1:off 2:off 3:off 4:off 5:off 6:off
network         0:off 1:off 2:on  3:on  4:on  5:on  6:off
random          0:off 1:off 2:on  3:on  4:on  5:on  6:off
rawdevices      0:off 1:off 2:off 3:on  4:on  5:on  6:off
acpid           0:off 1:off 2:off 3:off 4:on  5:on  6:off
ipchains        0:off 1:off 2:on  3:off 4:on  5:on  6:off
iptables        0:off 1:off 2:on  3:off 4:on  5:on  6:off
crond           0:off 1:off 2:off 3:off 4:off 5:off 6:off
anacron         0:off 1:off 2:off 3:off 4:off 5:off 6:off
lpd             0:off 1:off 2:off 3:off 4:off 5:off 6:off
xfs             0:off 1:off 2:on  3:off 4:on  5:on  6:off
ntpd            0:off 1:off 2:off 3:off 4:off 5:off 6:off
portmap         0:off 1:off 2:off 3:on  4:on  5:on  6:off
xinetd          0:off 1:off 2:off 3:on  4:on  5:on  6:off
autofs          0:off 1:off 2:off 3:off 4:on  5:on  6:off
nfs             0:off 1:off 2:off 3:off 4:off 5:off 6:off
nfslock         0:off 1:off 2:off 3:off 4:on  5:on  6:off
identd          0:off 1:off 2:off 3:off 4:off 5:off 6:off
radvd           0:off 1:off 2:off 3:off 4:off 5:off 6:off
rwhod           0:off 1:off 2:off 3:off 4:off 5:off 6:off
snmpd           0:off 1:off 2:off 3:off 4:off 5:off 6:off
snmptrapd       0:off 1:off 2:off 3:off 4:off 5:off 6:off
smartd          0:off 1:off 2:off 3:off 4:off 5:off 6:off
rhnsd           0:off 1:off 2:off 3:off 4:off 5:off 6:off
isdn            0:off 1:off 2:on  3:off 4:on  5:on  6:off
sshd            0:off 1:off 2:on  3:on  4:on  5:on  6:off
rstatd          0:off 1:off 2:off 3:off 4:off 5:off 6:off
rusersd         0:off 1:off 2:off 3:off 4:off 5:off 6:off
rwalld          0:off 1:off 2:off 3:off 4:off 5:off 6:off
yppasswdd       0:off 1:off 2:off 3:off 4:off 5:off 6:off
ypserv          0:off 1:off 2:off 3:off 4:off 5:off 6:off
ypxfrd          0:off 1:off 2:off 3:off 4:off 5:off 6:off
innd            0:off 1:off 2:off 3:off 4:off 5:off 6:off
winbind         0:off 1:off 2:off 3:off 4:off 5:off 6:off
smb             0:off 1:off 2:off 3:off 4:off 5:off 6:off
postgresql      0:off 1:off 2:off 3:off 4:off 5:off 6:off
httpd           0:off 1:off 2:off 3:off 4:off 5:off 6:off
squid           0:off 1:off 2:off 3:off 4:off 5:off 6:off
ip6tables       0:off 1:off 2:on  3:off 4:on  5:on  6:off
rarpd           0:off 1:off 2:off 3:off 4:off 5:off 6:off
named           0:off 1:off 2:off 3:off 4:off 5:off 6:off
arpwatch        0:off 1:off 2:off 3:off 4:off 5:off 6:off
cluster         0:off 1:off 2:on  3:off 4:on  5:on  6:off
ipvsadm         0:off 1:off 2:off 3:off 4:off 5:off 6:off
netdump         0:off 1:off 2:off 3:off 4:off 5:off 6:off
reconfig        0:off 1:off 2:off 3:on  4:on  5:on  6:off
amd             0:off 1:off 2:off 3:off 4:off 5:off 6:off
bootparamd      0:off 1:off 2:off 3:off 4:off 5:off 6:off
dhcpd           0:off 1:off 2:off 3:off 4:off 5:off 6:off
gated           0:off 1:off 2:off 3:off 4:off 5:off 6:off
irda            0:off 1:off 2:off 3:off 4:off 5:off 6:off
iscsi           0:off 1:off 2:off 3:off 4:off 5:off 6:off
junkbuster      0:off 1:off 2:off 3:off 4:off 5:off 6:off
kadmin          0:off 1:off 2:off 3:off 4:off 5:off 6:off
kprop           0:off 1:off 2:off 3:off 4:off 5:off 6:off
krb524          0:off 1:off 2:off 3:off 4:off 5:off 6:off
krb5kdc         0:off 1:off 2:off 3:off 4:off 5:off 6:off
mars-nwe        0:off 1:off 2:off 3:off 4:off 5:off 6:off
mcserv          0:off 1:off 2:off 3:off 4:off 5:off 6:off
mysqld          0:off 1:off 2:off 3:off 4:off 5:off 6:off
ups             0:off 1:off 2:off 3:off 4:off 5:off 6:off
ldap            0:off 1:off 2:off 3:off 4:off 5:off 6:off
routed          0:off 1:off 2:off 3:off 4:off 5:off 6:off
tux             0:off 1:off 2:off 3:off 4:off 5:off 6:off
bgpd            0:off 1:off 2:off 3:off 4:off 5:off 6:off
ospf6d          0:off 1:off 2:off 3:off 4:off 5:off 6:off
ospfd           0:off 1:off 2:off 3:off 4:off 5:off 6:off
ripd            0:off 1:off 2:off 3:off 4:off 5:off 6:off
ripngd          0:off 1:off 2:off 3:off 4:off 5:off 6:off
zebra           0:off 1:off 2:off 3:off 4:off 5:off 6:off
xinetd based services:
  chargen-udp:  off
  chargen:  off
  daytime-udp:  off
  daytime:  off
  echo-udp: off
  echo: off
  services: off
  servers:  off
  time-udp: off
  time: off
  dbskkd-cdb: off
  sgi_fam:  on
  finger: off
  rexec:  on
  rlogin: on
  rsh:  on
  ntalk:  off
  talk: off
  telnet: off
  wu-ftpd:  off
  rsync:  on
  amanda: off
  comsat: off
  amandaidx:  off
  amidxtape:  off
  imap: off
  imaps:  off
  ipop2:  off
  ipop3:  off
  pop3s:  off
  eklogin:  off
  gssftp: off
  klogin: off
  krb5-telnet:  off
  kshell: off
  swat: off
  tftp: off


-------------------------------------------------
                  httpd.conf (clients)
-------------------------------------------------


ServerTokens OS

ServerRoot "/etc/httpd"

PidFile run/httpd.pid

Timeout 300

KeepAlive On

MaxKeepAliveRequests 15000

KeepAliveTimeout 999

CoreDumpDirectory /etc/httpd

##
## Server-Pool Size Regulation (MPM specific)
##
<IfModule prefork.c>
StartServers       15
MinSpareServers    15
MaxSpareServers   150
MaxClients       150
MaxRequestsPerChild  0
</IfModule>

# worker MPM
# StartServers: initial number of server processes to start
# MaxClients: maximum number of simultaneous client connections
# MinSpareThreads: minimum number of worker threads which are kept
spare
# MaxSpareThreads: maximum number of worker threads which are kept
spare
# ThreadsPerChild: constant number of worker threads in each server
process
# MaxRequestsPerChild: maximum number of requests a server process
serves
ServerLimit 300
ThreadLimit 100
#### max processes
<IfModule worker.c>
StartServers        270
MaxClients          10800
MinSpareThreads     20
MaxSpareThreads     10800
ThreadsPerChild     40
MaxRequestsPerChild  0
</IfModule>

Listen 80

LoadModule tpcc_module /etc/httpd/modules/mod_tpcc.so

User apache
Group apache

#
# ServerAdmin: Your address, where problems with the server should
be
# e-mailed.  This address appears on some server-generated pages,
such
# as error documents.  e.g. admin@your-domain.com
#
ServerAdmin you@your.address

ServerName cl73

UseCanonicalName Off

DocumentRoot "/var/www/html"

<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>

#TypesConfig /etc/mime.types

#
# DefaultType is the default MIME type the server will use for a
document
# if it cannot otherwise determine one, such as from filename
extensions.
# If your server contains mostly text or HTML documents,
"text/plain" is
# a good value.  If most of your content is binary, such as
applications
# or images, you may want to use "application/octet-stream" instead
to
# keep browsers from trying to display binary files as though they
are
# text.
#
DefaultType text/plain

#
# The mod_mime_magic module allows the server to use various hints
from the
# contents of the file itself to determine its type.  The
MIMEMagicFile
# directive tells the module where the hint definitions are
located.
#
<IfModule mod_mime_magic.c>
#   MIMEMagicFile /usr/share/magic.mime
    MIMEMagicFile conf/magic
</IfModule>

#
# HostnameLookups: Log the names of clients or just their IP
addresses
# e.g., www.apache.org (on) or 204.62.129.132 (off).
# The default is off because it'd be overall better for the net if
people
# had to knowingly turn this feature on, since enabling it means
that
# each client request will result in AT LEAST one lookup request to
the
# nameserver.
#
HostnameLookups Off

#
# ErrorLog: The location of the error log file.
# If you do not specify an ErrorLog directive within a
<VirtualHost>
# container, error messages relating to that virtual host will be
# logged here.  If you *do* define an error logfile for a
<VirtualHost>
# container, that host's errors will be logged there and not here.
#
ErrorLog logs/error_log

#
# LogLevel: Control the number of messages logged to the error_log.
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
#
LogLevel warn

#
# The following directives define some format nicknames for use
with
# a CustomLog directive (see below).
#
#LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-
Agent}i\"" combined
#LogFormat "%h %l %u %t \"%r\" %>s %b" common
#LogFormat "%{Referer}i -> %U" referer
#LogFormat "%{User-agent}i" agent
#
#CustomLog logs/access_log combined


<Location /tpcc>
    SetHandler tpcc
</Location>
```

```
------------------------------------------------
                  inittab (server)
------------------------------------------------

#
# inittab       This file describes how the INIT process should set
up
#               the system in a certain run-level.
#
# Author:       Miquel van Smoorenburg,
<miquels@drinkel.nl.mugnet.org>
#               Modified for RHS Linux by Marc Ewing and Donnie
Barnes
#

# Default runlevel. The runlevels used by RHS are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have
networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Things to run in every runlevel.
ud::once:/sbin/update

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# When our UPS tells us power has failed, assume we have a few
minutes
# of power left.  Schedule a shutdown for 2 minutes from now.
# This does, of course, assume you have powerd installed and your
# UPS connected and working correctly.
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System
Shutting Down"

# If power was restored before the shutdown kicked in, cancel it.
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown
Cancelled"


# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
# xdm is now a separate service
#x:5:respawn:/etc/X11/prefdm -nodaemon


------------------------------------------------
                  intr.sh
------------------------------------------------

#!/bin/sh

i=56
while [ $i -le 67 ]
do
  echo 2 > /proc/irq/$i/smp_affinity
  if test $i -eq 59 || test $i -eq 60
  then
    echo 1 > /proc/irq/$i/smp_affinity
  fi
  cat /proc/irq/$i/smp_affinity
  let i=i+1
done

exit


------------------------------------------------
                  linux.ora
------------------------------------------------

control_files       = /home/oracle/dev/raw/control_001
processes       = 220
sessions      = 440
_imu_pools      = 230
db_cache_size     = 10000M
db_keep_cache_size    = 60000M
db_recycle_cache_size   = 12500M
db_16k_cache_size   = 7000M
db_8k_cache_size    = 256M
recovery_parallelism    = 100
db_name       = tpcc
db_files      = 300
compatible      = 10.0.0.0.0
dml_locks     = 500
db_block_size     = 2048
log_buffer      = 20971520
log_checkpoint_interval   = 37500000
log_checkpoint_timeout    = 0
log_checkpoints_to_alert  = true
undo_management     = auto
undo_retention      = 0
undo_tablespace     = undo_ts
cursor_space_for_time   = true
plsql_optimize_level    = 2
_optimizer_cache_stats    = false
_optimizer_cost_model   = io
_cursor_cache_frame_bind_memory = true
replication_dependency_tracking = false
db_file_multiblock_read_count = 32
utl_file_dir      = *
_db_cache_pre_warm    = false
_array_update_vector_read_enabled= true
pga_aggregate_target    = 0
db_block_checking     = false
db_block_checksum     = false
_check_block_after_checksum = false
disk_asynch_io      = true
_lgwr_async_io      = false
shared_pool_size    = 256M
java_pool_size      = 0
_ksmg_granule_size    = 134217728
db_writer_processes     = 1
_db_writer_max_writes   = 512
timed_statistics    = false
statistics_level    = basic
fast_start_mttr_target    = 0
_two_pass     = false
max_dump_file_size             = unlimited

------------------------------------------------
                   rr.c
------------------------------------------------

#include <stdio.h>
#include <unistd.h>
#include <sched.h>
#include <sys/types.h>

main(int argc, char *argv[])
{
  struct sched_param sp;
  int i;

  if (argc < 4) {
    fprintf(stderr, "usage: %s -p <prio> pid...\n", argv[0]);
    exit(-1);
  }

  if (!strcmp("-p", argv[1])) {
    sp.sched_priority = atoi(argv[2]);
  }

  printf("setting priority to: %d\n", sp.sched_priority);
  for (i = 3; i < argc; i++) {
    pid_t pid = atoi(argv[i]);
    if (sched_setscheduler(pid, SCHED_RR, &sp) == -1) {
      perror("sched_setscheduler");
      exit(-1);
    }
  }

  exit(0);
}


------------------------------------------------
                   setrrpri.sh
------------------------------------------------

sleep $1

# Run oracle system processes at sched_rr priority
/home/oracle/config/rr -p 95 $(ps aux | grep ora_ | grep -v grep |
awk '{print $2}')

# Run oracle client processes at sched_rr priority
/home/oracle/config/rr -p 95 $(ps aux | grep oracletp | grep -v
grep | awk '{print $2}')

# Run lgwr at a higher priority
/home/oracle/config/rr -p 96 $(ps aux | grep ora_lgwr | grep -v
grep | awk '{print $2}')




------------------------------------------------
```

```
                  start (clients)
-------------------------------------------------

#!/bin/sh
set -ex

#. sbin/envvars

ulimit -u 27000
ulimit -c 99999
ulimit -s 1536

/usr/sbin/httpd.worker -d /etc/httpd


-------------------------------------------------
                  stop (clients)
-------------------------------------------------

#!/bin/sh
kill `cat /etc/httpd/run/httpd.pid`


-------------------------------------------------
                  tux.env (clients)
-------------------------------------------------

TUXDIR=/home/bea/tuxedo8.1; export TUXDIR
PATH=$TUXDIR/bin:$PATH; export PATH
COBCPY=:$TUXDIR/cobinclude; export COBCPY
COBOPT="-C ANS85 -C ALIGN=8 -C NOIBMCOMP -C TRUNC=ANSI -C
OSEXT=cbl"; export COBOPT
SHLIB_PATH=$TUXDIR/lib:$SHLIB_PATH; export SHLIB_PATH
LIBPATH=$TUXDIR/lib:$LIBPATH; export LIBPATH
LD_LIBRARY_PATH=$TUXDIR/lib:$LD_LIBRARY_PATH; export
LD_LIBRARY_PATH
WEBJAVADIR=$TUXDIR/udataobj/webgui/java
export APPDIR=/home/bea/tuxedo8.1
export TUXCONFIG=$APPDIR/tuxconfig
export FSCONFIG=$TUXDIR
#export TMNOTHREADS=yes


-------------------------------------------------
                  ubb (clients)
-------------------------------------------------

#
# 10i UBBconfig file for 10 clients configuration
#
# Clients systems have indentical configuration except:
# IPCKEY 4000[75-84] on client[75-84]
# MASTER cl[75-84] on Client[75-84]
# LMID cl[75-84] on Client[75-84]
#
#-----------------------------------------------------------------
-----------
*RESOURCES
#-----------------------------------------------------------------
-----------
IPCKEY    40075
MASTER    cl75
MAXACCESSERS  19000 # 1024 or more
MAXGTT 19000
MAXSERVERS  22
MAXSERVICES 120 #MAXSERVERS * #-of-services-each-server + 10 (for
BBL)
#MAXCONV    13099
MODEL   SHM
LDBAL Y
*MACHINES
DEFAULT:
      TUXDIR="/home/bea/tuxedo8.1"
      APPDIR="/home/bea/tuxedo8.1"
      TUXCONFIG="/home/bea/tuxedo8.1/tuxconfig"
      UID=0
      GID=0
      TYPE="LINUX"
cl75  LMID=cl75

*GROUPS
TPCC
  LMID=cl75 GRPNO=1 OPENINFO=NONE
DELI1
LMID=cl75 GRPNO=2 OPENINFO=NONE

*SERVERS
DEFAULT: CLOPT="-A"
tpccora  SRVGRP=TPCC SRVID=10 RQADDR=txnque1 REPLYQ=Y MIN=2 MAX=5
tpccora  SRVGRP=TPCC SRVID=20 RQADDR=txnque2 REPLYQ=Y MIN=2 MAX=5
tpccora  SRVGRP=TPCC SRVID=30 RQADDR=txnque3 REPLYQ=Y MIN=2 MAX=5
tpccora  SRVGRP=TPCC SRVID=40 RQADDR=txnque4 REPLYQ=Y MIN=2 MAX=5
tpccora  SRVGRP=TPCC SRVID=50 RQADDR=txnque5 REPLYQ=Y MIN=2 MAX=5
delioral  SRVGRP=DELI1 SRVID=100 RQADDR=txnque6 REPLYQ=N MIN=2
MAX=3

*SERVICES

DEFAULT:
    LOAD=1
    PRIO=1
    BUFTYPE="CARRAY"
    TRANTIME=900
    AUTOTRAN=N
no_transaction
os_transaction
pt_transaction
sl_transaction
dy_transaction1


-------------------------------------------------
                  rc.local (server)
-------------------------------------------------

#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local

touch /var/lock/subsys/local
echo 5 > /proc/sys/kernel/printk
echo 0x40000000 > /proc/sys/kernel/shmall
echo 0x1880000000 > /proc/sys/kernel/shmmax

# following for aio
echo 1048576 > /proc/sys/fs/aio-max-nr

echo kiobuf 60 10 > /proc/slabinfo

# set correct # for > 32G memory. Each is 256M
echo 90 > /proc/sys/vm/nr_hugepages

# Not sure whether the following is still needed
usermod -G root oracle

# mapping of the raw devices
sh /root/mkraw_run_extrao.sh

#insmod  /lib/modules/2.4.18-
tpc.0.9custom/kernel/drivers/block/cciss.o
#rmmod cciss
#insmod  /usr/src/linux-2.4.18-e.31/drivers/block/cciss.o
#sh /root/mkraw_run.sh

rdate -s timezone
```

Appendix D: Third Party Letters

NETGEAR about NETGEAR products why buy NETGEAR where to buy customer services news/events NE

Home
Basket
Order Tracking
Account
Customer Service

hubs & switches
networking components
networking devices

search: go!

VERIFIED by VISA
learn more

Product

GS508TNA 8PORT 10/100/1000BTX COPPER GIGABIT SWITCH

Part Number : GS508TNA
In Stock : YES
Platform : Not Machine Specific
Media : Peripherals

Price: $586.57

BuyNow

Description:

Features:

- **General Information**
- Marketing Information: Copper Gigabit ports in this Ethernet switcl high-speed connectivity without the cost and hassle of fiber cables backbone for power workgroups, data centers, and server farms w convenient Plug and Play installation NET GEAR's high-performar GS508T Copper Gigabit Switch delivers the power of Copper Gigɛ Ethernet to optimize your small to medium-sized business networl
- CAPABLE & AFFORDABLE:
- Copper Gigabit ports provide high-performance backbone connec between workgroups, data centers, and server farms at a low cost
- ACCESSIBLE:
- Multiple users can simultaneously access the backbone and serve network congestion.
- POWERFUL:
- Delivers up to 11.8 million packets per second as it supports 8,00(devices on the network.
- SAVVY:
- Eight, 10/100/1000 auto-negotiating ports automatically sense the speed and operate at the optimum rate. Self-configuring and easy
- RACK-MOUNTABLE:
- Can be positioned where it's most convenient close to servers, at of your network.
-
- **Memory Information**
- Memory Size: 8MB
- Memory Type: DRAM
-
- **Miscellaneous**

THE ECOMMERCE TRANSACTION PLATFORM

July 29, 2003

Raghunath K. Othayoth
ISS - Solutions and Strategy
Hewlett Packard Company
281-518-2748 tel
281-514-8375 fax

Per your request I am enclosing the pricing information regarding TUXEDO 6.5 that you requested. This pricing applies to Tuxedo 6.4, 6.5, 7.1,8.0 and 8.1. Please note that Tuxedo 8.1 is our most recent version of Tuxedo. Core functionality services (CFS)-R pricing is appropriate for your activities. As per the table below HP/Compaq systems are classified as either a Tier 1, 2, 3, 4 or 5 systems depending on the performance and CPU capacity of the system. The HP/Compaq DL 360 machines are Tier 1 machines – price is $1,200 per server (License), eligible for a 5% discount = $1,140 per server + $252 per server (7x24) for support – support is non discountable. This quote is valid for 60 days from the date of this letter.

Tuxedo Core Functionality Services (CFS-R) Program Product Pricing and Description

TUX-CFS-R provides a basic level of middleware support for distributed computing, and is best used by organizations with substantial resources and knowledge for advanced distributed computing implementations.

TUX-CFS-R prices are server only and are based on the overall performance characteristics of the server and uses the same five tier computer classification as TUXEDO 6.4, 6.5, 7.1,8.0, and 8.1. Prices range from $1,200 for Tier 1 to $100,000 for Tier 5. Under this pricing option EVERY system running TUX-CFS-R at the user site must have a TUXEDO license installed and pay the appropriate per server license fees.

Very Truly Yours,

Robert J. Gieringer

Rob Gieringer,
Worldwide Pricing Manager

BEA Tux/CFS-R Unlimited User License Fees Per Server

Unlimited User License fees per server	**Number of Users**	**Dollar Amount**	**Maintenance (5 x 9) per year**	**Maintenance (7 x 24) per year**
Tier 1 -- PC Servers with 1 or 2 CPUs, entry level RISC Uni-processor workstations and servers	Unlimited	$1,200.00	$216	$252
Tier 2 - PC Servers with 3 or 4 CPUs, Midrange RISC Uni-processor servers and workstations with up to 2 CPUs	Unlimited	$4,800.00	$864	$1,008
Tier 3 - Midrange Multiprocessors, up to 8 CPUs per system capacity	Unlimited	$12,000.00	$2,160	$2,520
Tier 4 - Large (more than 8, less than 32 CPUs)	Unlimited	$40,000.00	$7,200	$8,400
Tier 5 - Massively Parallel Systems, > 32 processors	Unlimited	$100,000.00	$18,000	$21,000

	Tier 1	**Tier 1**	**Tier 2**	**Tier 3**	**Tier 4**	**Tier 5**
Operating System						
HP/UX 9.X;10.X	Uni-processor Workstation B Class - 132/180/2000 C Class (3000/3600/ 3700) 2P Client Machines Compaq DL360	9000/E25 9000/E35 9000/E45 9000/E55 9000/G30 9000/G40 9000/A180 9000/A180C 9000 /A400	9000/G50 9000/G60 Multi-Processor Workstations J Class (J282/J2240/J5600/ J6000/J6700) 9000/R380,390 9000/D200,210 220/30/50/60/80 D310/20/30 D350/60/70/80 9000 /A500 9000 – L1000 9000 – R Class	9000/H20, 30 9000/H40, 50 9000/I30, 40 9000/K1XX 9000 – L2000/L3000 9000/I50,60 9000/H60 9000/G70 9000/H70 9000/I70 9000/K2XX 9000/K3XX 9000/K4XX 9000/K5XX N4xxx Series	9000/T500,T5 20,T600 1-16 CPUs S-Class	9000/V series all models X-Class 9000 Series - Superdome

June 26, 2003

Raghunath K. Othayoth
ISS - Solutions and Strategy
Hewlett-Packard Company
281-518-2748 tel

Per your request for information on pricing for several Red Hat products to be used in conjunction with your TPC-C benchmark testing, I have included the following quote. Note that these products are not yet released and are to be released 6 months from the date of this quote, as such the part numbers and prices are subject to change.

Part Number	***Description***	***Unit Price***	***Quantity***	***Price***
TBD	Red Hat Enterprise Linux AS for the Itanium Processor (version 3 Standard Edition)	$1,992	1	$1,992
TBD	2 Additional Years Subscription to Red Hat Enterprise Linux AS for the Itanium processor (version 3 Standard Edition)	$1,992	2	$3,984
TBD	Red Hat Enterprise Linux ES (version 3 Standard Edition)	$799	8	$6,392
TBD	2 Additional Years Subscription to Red Hat Enterprise Linux ES (version 3 Standard Edition)	$799	16	$12,784
TOTAL				$25,152.00

Products will be orderable through www.redhat.com or Red Hat Sales 1-888-REDHAT-1. If we can be of any further assistance, please contact Mike Ferris at mferris@redhat.com.
*Support and maintenance for software includes minimum annual configuration and installation support and continuous proactive update and upgrade support via Red Hat Network.

Appendix E:
Database Pricing

-----Original Message-----
From: MaryBeth Pierantoni [mailto:mary.beth.pierantoni@oracle.com]
Sent: Wednesday, September 03, 2003 5:47 PM
To: Othayoth, Raghunath
Cc: Buch,Vineet; Brey,Michael
Subject: RE: Oracle 10g 64-bit Standard Edition Pricing

September 5, 2003

The following is good for 30 days:

Product	**Price**	**Quantity**	**Extended Price**
Oracle10*g* Database Standard Edition, Processor 3 year term for 4 processors, Unlimited Users	$7,500	4	$30,000
Oracle Database Server Support Package for **3 years**	$2,000	1	$6,000
Oracle Mandatory E-Business Discount			<$1,800>
Oracle TOTAL			**$34,200**

Contact: MaryBeth Pierantoni, mary.beth.pierantoni@oracle.com, 650-506-2118