HP 9000 Superdome Enterprise Server
using
HP-UX 11.i 64-bit
and
Oracle9i Database Enterprise Edition

TPC Benchmark® C
Full Disclosure Report

Third Edition

Submitted for Review
June 12, 2002

Third Edition - June 12, 2002

Hewlett-Packard Company believes that the information in this document is accurate as of the publication date. The information in this document is subject to change without notice. Hewlett-Packard Company assumes no responsibility for any errors that may appear in this document.
The pricing information in this document is believed to accurately reflect the current prices as of the publication date. However, Hewlett-Packard Company provides no warranty of the pricing information in this document.
Benchmark results are highly dependent upon workload, specific application requirements, and system design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, TPC Benchmark® C should not be used as a substitute for a specific customer application benchmark when critical capacity planning and/or product evaluation decisions are contemplated.
All performance data contained in this report was obtained in a rigorously controlled environment. Results obtained in other operating environments may vary significantly. Hewlett-Packard Company does not warrant or represent that a user can or will achieve similar performance expressed in transactions per minute (tpmC®) or normalized price/performance ($/tpmC®). No warranty of system performance or price/performance is expressed or implied in this report.

Printed in U.S.A., June 12, 2002.

Abstract

Overview

This report documents the methodology and results of the TPC Benchmark® C test conducted on the HP 9000 Superdome Enterprise Server in a client/server configuration, using Oracle9i Database Enterprise Edition and the TUXEDO 6.4 transaction monitor. The operating system used for the benchmark was Hewlett-Packard's HP-UX 11.i 64-bit. The application was written in C and compiled using HP C/ANSI C/HP-UX.

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 required by the benchmark specification.

Standard and Executive Summary Statements

Page *iii* contains the standard system summary and pages ***iv-vi*** contain the executive summary of the benchmark results for the HP 9000 Superdome Enterprise Server.

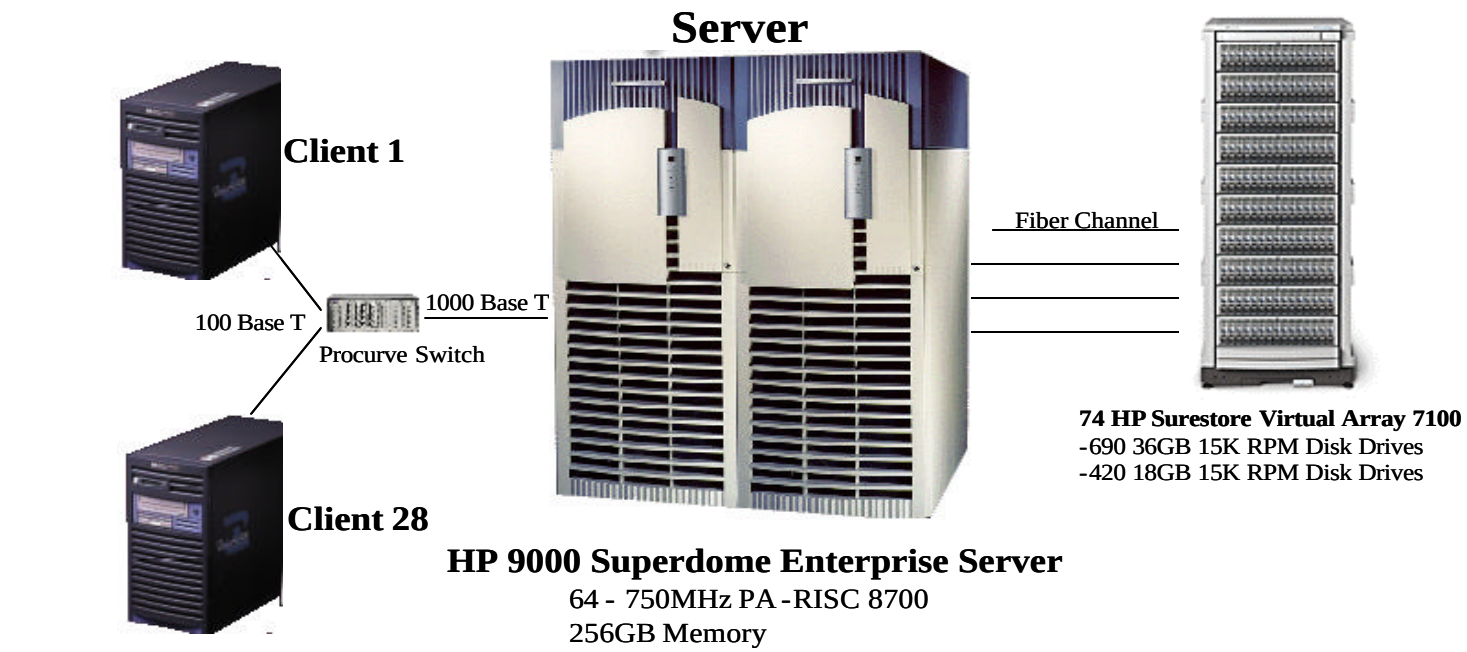
Auditor

The benchmark configuration, environment and methodology used to produce and validate the test results, and the pricing model used to calculate the price/performance, were audited by Tom Sawyer for Performance Metrics, Inc. to verify compliance with the relevant TPC specifications.

Standard System Summary

Company Name	**System Name**	**Database Software**	**Operating System Software**
Hewlett-Packard Company	HP 9000 Superdome Enterprise Server	Oracle9i Database Enterprise Edition	HP-UX 11.i 64-bit
HP H/W Availability Date - Now S/W Availability Date - Now			
Total System Cost	**TPC-C® Throughput**	**Price/Performance**	
Hardware Software 3-year maintenance	Sustained maximum throughput of System running TPC-C® expressed in transactions per minute	Total system cost/tpmC ($6,388,889/389434.4)	
$6,388,889	**389,434.40 tpmC**	**$16.41 per tpmC**	

invent ORACLE	**HP 9000 Superdome Enterprise Server**			TPC-C Revision 5 Report Date: June 12, 2002
Total System Cost	TPC Throughput	Price/Performance		Availability Date
$6,388,889	**389,434.40 tpmC**	**$16.41/tpmC**		**Now**
Processors	Database Manager	Operating System	Other Software	Number of Users
64 PA-RISC 8700 750MHz	**Oracle9i Database Enterprise Edition**	**HP-UX 11.i 64-bit**	**TUXEDO 6.4**	**308,000**



System Components	*Server (Superdome)*		*each Client (28 C3700s)*	
	Qty	*Type*	*Qty*	*Type*
Processors	64	750MHz PA-RISC 8700	1	750MHz PA-RISC 8700
Cache Memory	each	.75 MB I-cache, 1.5MB D-cache	each	.75MB I-cache/1.5 MB D-cache
Memory	256	GB	1	8 GB
Disk Controllers	25	PCI Fibre Channel 2X	1	Ultra2 SCSI LVD
Disk Drives	74	Surestore Virtual Array 7100 with 420 18GB and 690 36GB 15K RPM drives	1	18 GB disk
Total Storage	14607	GB		
Tape Drives	1	DVD ROM		
Terminals	1	Console Terminal	1	Console Terminal

HP 9000 Superdome Enterprise Server

TPC-C Rev 5

Report Date: June 12, 2002

Description	Part Number	Brand	Price Key	US List Price	Qty	Price	3Year Main.Price
Server Hardware							
Super Dome left chassis	A5201A, Opt. 101		1	317,429	1	317,429	147,264
Super Dome right chassis	A5202A, Opt. 101		1	235,655	1	235.655	
Memory module - 2 GB	A5198A, Opt. 0D1		1	14,000	128	1.792.000	
I/O enclosures	A4856A, Opt. 0D1		1	14,805	4	59.220	
PDCA Redundant Power Source	A5800A, Opt. 0D1		1	578	2	1.156	
Cell Board with 4 PA-8700 750MHz Processors	A6445A, Opt. 0D1		1	10,080	16	161,280	
ICOD right to use processor	A6441A Opt. 104		1	23,806	64	1.523.584	351.744
Super Dome PCI Core I/O card	A5210A, Opt. 0D1		1	1,045	1	1.045	
PCI 1000BT Lan Adapter	A4926A, Opt. 0D1		1	2,135	1	2.135	
PCI Fibre Channel 2X	A5158A, Opt 0D1		1	2,240	25	56.000	
Rack Installation Kit	A5170A, Opt. 0D1		1	410	1	410	
DVD-ROM (includes SCSI-2 card, cables, enclosures)	C4318SZ		1	3,738	1	3.738	
HP9000 A500 Support Management Station (includes memory,CPU,lan card,disk,OS, etc)	A5570B		1	11,780	1	11.780	
.5m 68pin SCSI Cable	Opt. 001		1	99	1	99	
WSE 68pin SCSI Teminator	Opt. 835		1	46	1	46	
HP-UX 11.i Sys Media, CD-ROM	B3920EA, Opt. AAF		1	520	1	520	
PowerTrust 12kVA UPS 230VUPS	A6585A		1	10,999	6	65.994	
PowerTrust 3.0VA UPS 230VUPS	A1356A		1	3,199	2	6.398	
PowerTrust 2.0kVA UPS 230VUPS	A1353A		1	2,199	1	2,199	
(Support Option for UPS)			1	0	1	0	
Surestore VA 7100 w/dual controllers 512MB cache	A6262A		1	44,250	74	3.274.500	271.876
18GB 15K RPM FC HDD	A6191A, Opt. 0D1		1	914	420	383.880	
36GB 15K RPM FC HDD.	A6193A, Opt 0D1		1	1,349	690	930.810	
2 meter Fibre Optic Cable	A3583A		1	175	74	12,950	
16 meter Fibre Optic Cable	A3531A		1	200	25	5.000	
10 Port Short Wave Fibre Channel Hub	A3724A		1	9,690	25	242.250	
HP9000 Std. Rack System E41	A4902A		1	1,910	7	13.370	
Modular Power Dist.	A5137AZ		1	145	26	3.770	
200-240 Volts	A5137AZ, Opt AW4		1	94	26	2,444	
				Subtotal		**9,109,661**	**770,884**
Server Software							
Oracle9i Database Enterprise Edition (9.2.0.1.0) for HPUX 11i	Runtime	Oracle	2	988.800	1	988.800	
Server Support Package: Database		Oracle	2				6,000
				Subtotal		**988,800**	**6,000**
Client Hardware							
hp workstation c3700 with 2GB Memory	A6057B		1	12,545	28	351.260	108.668
1 GB Memory Moudle	A6016A, Opt. OD1		1	1,395	168	234.360	
700/96 Console	C1064GX		1	550	1	550	
18 GB LVD 10K RPM Disk	A4998A, Opt. OD1		1	595	28	16,660	
				Subtotal		**602,830**	**108,668**
Client Software							
HP C/ANSI C Compiler	B3901BA, Option AH0		1	1600	1	1.600	170
BEA Tuxedo 6.4		Bea Sys.	3	2,850	28	79,800	57,960
				Subtotal		**81,400**	**58,130**
User Connectivity							
HP ProCurve Switch 4000M	J4121A		1	1999	1	1.999	588
HP ProCurve Switch Gigabit-SX Module	J4113A		1	1189	1	1,189	
				Subtotal		**3,188**	**588**
HP's Large configuration Discount and Support Prepayment*						($5,037,840)	($303,421)
				Total		**5,748,040**	**640,850**

*All discounts are based on US list prices and for similar quantities and configurations

1=HP, 2= Oracle, 3=BEA Systems

Three Year Cost of Ownership:	**$6,388,889**
tpmC Rating:	**389,434**
$/tpmC:	**$16.41**

Audited by Tom Sawyer for Performance Metrics, Inc.

Prices used in TPC benchmarks reflect 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.

Numerical Quantities Summary for HP 9000 Superdome Enterprise Server

MQTH, Computed Maximum Qualified Throughput **389,434.40 tpmC**

Response Times (in seconds)

	90th %-ile	Maximum	Average
New-Order	0.46s	30.22s	0.29s
Payment	0.42s	29.69s	0.24s
Order-Status	0.41s	28.99s	0.25s
Delivery (interactive portion)	0.09s	20.60s	0.08s
Delivery (deferred portion)	0.50s	28.30s	0.29s
Stock-Level	0.39s	28.44s	0.23s
Menu	0.10s	0.57s	0.001s

Transaction Mix, in percent of total transactions

New-Order	44.81%
Payment	43.04%
Order-Status	4.05%
Delivery	4.04%
Stock-Level	4.05%

Keying/Think Times

	Keying Time			Think Time		
	Min	Avg	Max	Min	Avg	Max
New-Order	18.02s	18.03s	18.2s	0.01s	12.02s	188.22s
Payment	3.01s	3.02s	3.19s	0.01s	12.02s	222.72s
Order-Status	2.01s	2.02s	2.19s	0.01s	10.03s	165.36s
Delivery (interactive)	2.01s	2.02s	2.19s	0.01s	5.03s	82.24s
Stock-Level	2.01s	2.02s	2.18s	0.01s	5.02s	75.62s

Test Duration

Ramp up time	73 minutes
Measurement interval	143 minutes
Transactions during measurement interval	124,266,353
Ramp down time	4.795 minutes

Checkpointing

Number of checkpoints in measurement interval	5
Checkpoint Interval	28.6 minutes

TPC Benchmark C Overview

This is the full disclosure report for a benchmark test of the HP 9000 Superdome Enterprise Server using Oracle9i Database Enterprise Edition . It meets the requirements of the TPC Benchmark® C Standard Specification, Revision 5 dated March, 2001.

TPC Benchmark® C was developed by the **T**ransaction Processing **P**erformance **C**ouncil (TPC). It is the intent of this group to develop a suite of benchmarks to measure the performance of computer systems executing a wide range of applications. Hewlett-Packard Company Oracle Corporation are active participants in the TPC.

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

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

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

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

Hewlett-Packard Company does not warrant or represent that a user can or will achieve performance similar to the benchmark results contained in this report. No warranty of system performance or price/performance is expressed or implied by this report.

1 General Items

1.1 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 the HP C/ANSI C/HP-UX application code used in this TPC-C® test.

1.2 Test Sponsor

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

The High Performance Systems Division of Hewlett-Packard Company and Oracle Corporation are the test sponsors of this TPC Benchmark® C.

1.3 Parameter Settings

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

- Database options
- Recover/commit options
- Consistency/locking options
- Operating system and application configuration parameter
- Compilation and linkage options and run-time optimizations used to create/install applications, OS, and/or databases

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

The intent of the above clause is that anyone attempting to recreate the benchmark environment has sufficient information to compile, link, optimize, and execute all software used to produce the disclosed benchmark result.

Appendix A contains the application "make" files. Appendix C contains the HP-UX operating system parameters used to generate the kernel for the configuration used in this benchmark. Also included are all of the Oracle9i Database Enterprise Edition database parameters and the TUXEDO 6.4 transaction monitor parameters used.

1.4 Configuration Diagrams

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

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

The server System Under Test, an HP 9000 Superdome Enterprise Server depicted in Figure 1.1, consisted of:

- 64 750MHz PA-RISC 8700 System Processors

- 256 GB of memory
- 25 PCI Fibre Channel 2X Adapters
- 74 Surestore Virtual Array 7100 (with 420 18GB and 690 36GB 15K RPM disk)
- Two LAN interfaces

As indicated in Figure 1.2, this benchmark configuration used Remote Terminal Emulator (RTE) programs that executed on 7 N4000 Enterprise Server drivers to emulate TPC-C user sessions. The emulated users on the driver systems were directly connected to the client systems under test via 28 separate 100 Base-T local area network (LAN) and communicated using TCP/IP. The clients were connected to the SUT via a procurve 4000M switch.

The priced configuration for the HP 9000 Superdome Enterprise Server is shown in Figure 1.1. In the priced configuration, the RTE shown in the benchmark configuration is replaced by the appropriate number of workstations (emulating ANSI terminals) connected to hubs.

Figure 1.1: HP Superdome Enterprise Server Priced Configuration

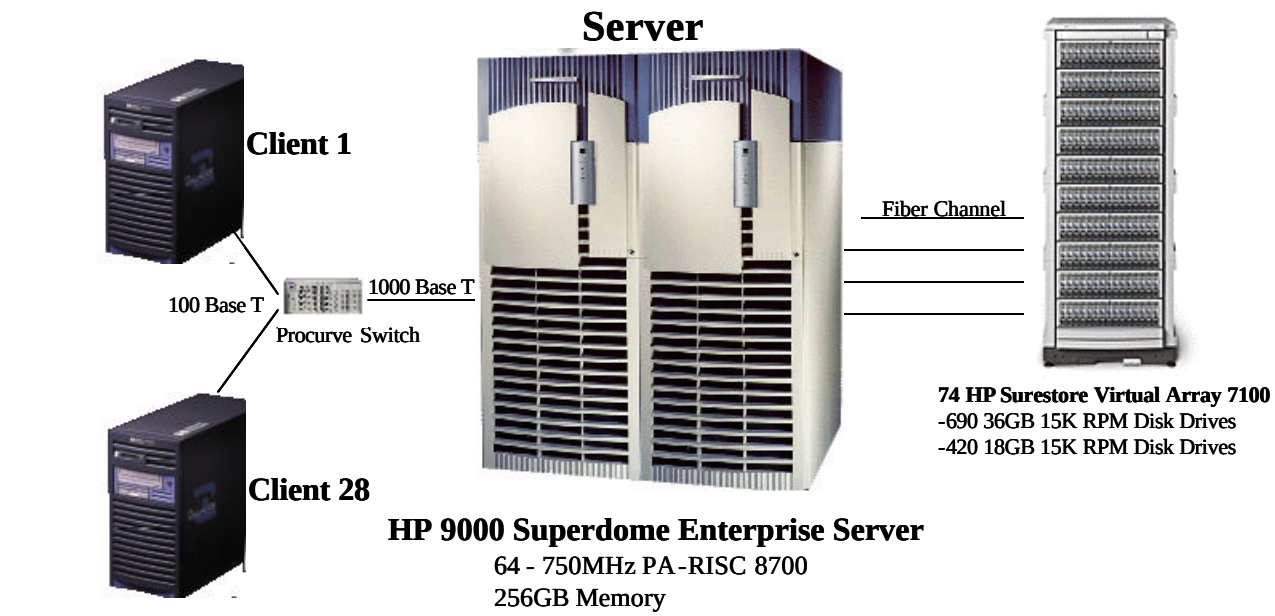
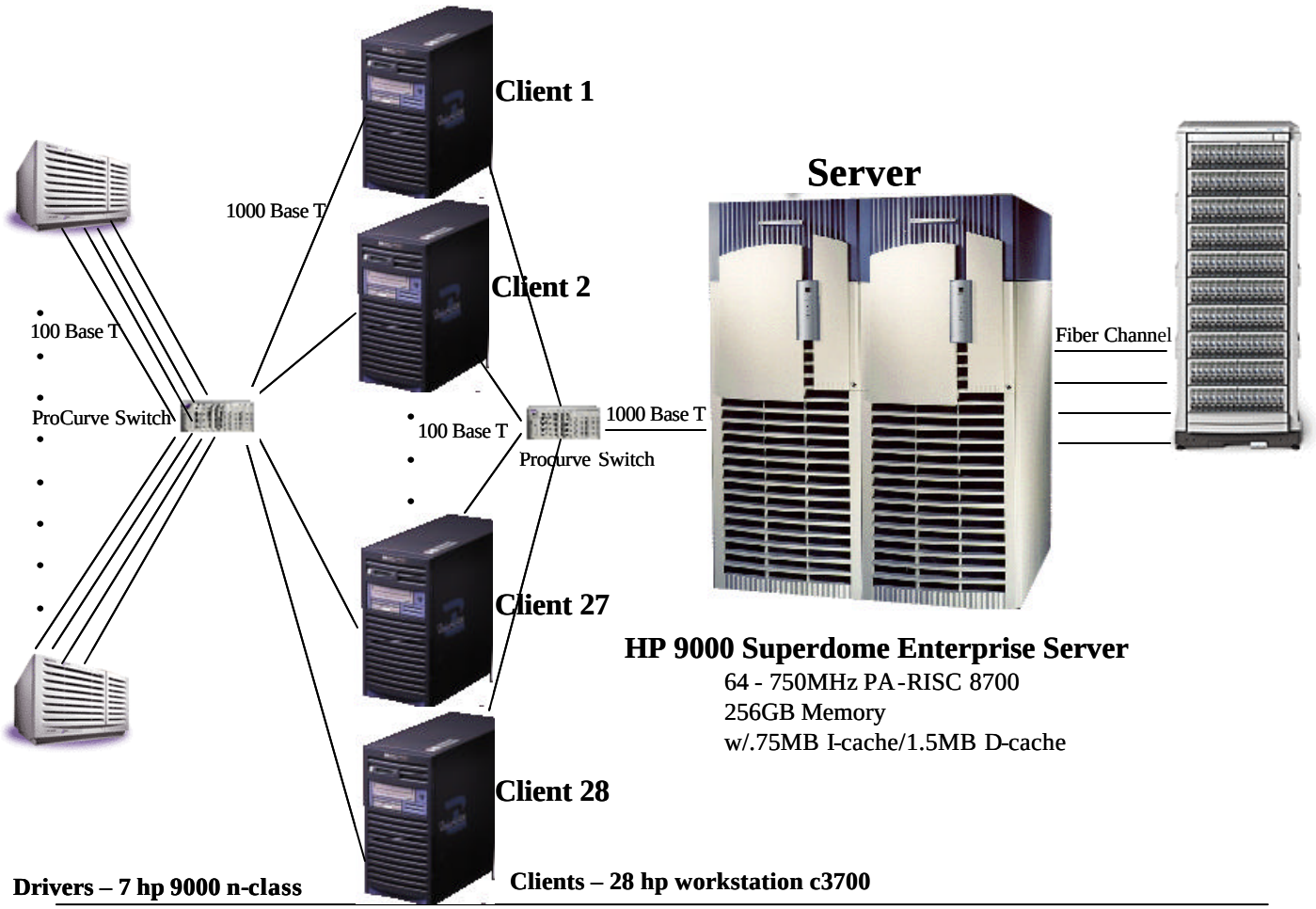


Figure 1.2: HP Superdome Enterprise Server Benchmark Configuration



2 Clause 1 Related Items

2.1 Table Definitions

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

Appendix B describes the programs that define, create, and populate the Oracle9i Database Enterprise Edition database for TPC-C® testing.

2.2 Physical Organization of Database

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

Space was allocated to Oracle9i Database Enterprise Edition according to the data in section 5.2. The size of the database table space on each disk drive was calculated to provide even distribution of load across the disk drives.

2.3 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 maximum key value for these new rows.

There were no restrictions on insert and delete operations to any tables.

2.4 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. 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.

Partitioning, replication, and additional or duplicated attributes were not used in this implementation.

3 Clause 2 Related Items

3.1 Random Number Generation

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

The library routine SRAND48 (3C) was used to seed the library routine DRAND48 (3C) which generated pseudo-random numbers using the well-known linear congruential algorithm and 48-bit integer arithmetic. Further information on SRAND48 (3C) and DRAND48 (3C) can be found in the HP-UX Reference Manual Vol. 3.

3.2 Input/Output Screen Layout

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

The screen layouts corresponded exactly to those in Clauses 2.4.3, 2.5.3, 2.6.3, 2.7.3, and 2.8.3 of the TPC-C® Standard Specification.

3.3 Priced Terminal Feature Verification

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

The terminal features were verified by manually exercising each specification on an HP 712/80 workstation running an ANSI terminal emulator.

3.4 Presentation Manager or Intelligent Terminal

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

Application code running on the client implemented the TPC-C user interface. A listing of this code is included in Appendix A. Used capabilities of the terminal beyond basic ASCII entry and display were restricted to cursor positioning.

A presentation manager was not used.

Table 3.1: Transaction Statistics

Type	Item	Value
New Order	Home warehouse items	99.00%
	Remote warehouse items	1.00%
	Rolled back transactions	1.00%
	Average items per order	10.00
Payment	Home warehouse	85.00%
	Remote warehouse	15.00%
	Non primary key access	60.00%
Order Status	Non primary key access	60.06%
Delivery	Skipped transactions	0
Transaction Mix	New Order	44.81%
	Payment	43.04%
	Order Status	4.05%
	Delivery	4.04%
	Stock Level	4.05%

3.5 Transaction Statistics

Table 3.1 lists the numerical quantities that Clauses 8.1.3.5 to 8.1.3.11 require.

3.6 Queuing Mechanism

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

Delivery transactions were submitted to servers using the same TUXEDO mechanism that other transactions used. The only difference was that the call was asynchronous, i.e., control would return to the client process immediately and the deferred delivery part would complete asynchronously.

4 Clause 3 Related Items

4.1 Transaction System Properties (ACID)

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

The TPC Benchmark® C Standard Specification defines a set of transaction processing system properties that a system under test (SUT) must support during the execution of the benchmark. Those properties are Atomicity, Consistency, Isolation, and Durability (ACID). This section quotes the specification definition of each of these properties and describes the tests done as specified and monitored by the auditor to demonstrate compliance.

4.2 Atomicity

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

4.2.1 Completed Transaction

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

The values of w_ytd, d_ytd, c_balance, c_ytd_payment, and c_payment_cnt of a randomly selected warehouse, district, and customer were retrieved. The Payment transaction was executed on the same warehouse, district, and customer. The transaction was committed. The values w_ytd, d_ytd, c_balance, c_ytd_payment, and c_payment_cnt were retrieved again. It was verified that all values had been changed appropriately.

4.2.2 Aborted Transaction

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number as specified in Clause 2.5.1.2) and substitute a ROLLBACK of the transaction for the COMMIT of the transaction. Verify that the records in the CUSTOMER, WAREHOUSE, and DISTRICT tables have NOT been changed

The values of w_ytd, d_ytd, c_balance, c_ytd_payment and c_payment_cnt of a randomly selected warehouse, district, and customer were retrieved. The Payment transaction was executed on the same warehouse, district, and customer. The transaction was rolled back. The values of w_ytd, d_ytd, c_balance, c_ytd_payment, c_payment_cnt were retrieved again. It was verified that none of the values had changed.

4.3 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 the database is initially in a consistent state.

The TPC Benchmark C standard requires the System Under Test to meet the following 12 consistency conditions (c.f. *TPC Standard Specification, Clauses 3.3.2.1 to 3.3.2.12*):

1. the sum of the district balances in a warehouse is equal to the warehouse balance;
2. for each district, the next order-id minus one is equal to maximum order-id in the ORDER table and equal to the maximum new-order-id in the NEW-ORDER table;

3. for each district, the maximum order-id minus minimum order-id in the ORDER table plus one equals the number of rows in the NEW-ORDER table for that district;
4. for each district, the sum of the order-line counts equals the number of rows in the ORDER-LINE table for that district;
5. for each row in the ORDER table, the carrier-id is set to a null value only if there is a corresponding row in the NEW-ORDER table;
6. for each row in the ORDER table, the order-line count must equal the number of rows in the ORDER-LINE table for that order;
7. for any row in the ORDER-LINE table, the delivery date/time is set to a null value only if the corresponding row in the ORDER table has the carrier-id set to a null value;
8. for each warehouse, the year-to-date amount must equal the sum of the amounts in the HISTORY table for that warehouse;
9. for each district, the year-to-date amount must equal the sum of the amounts in the HISTORY table for that district;
10. for each customer, the balance must equal the sum of the order-line amount minus the sum of the history amount for that customer;
11. for each district, the total orders minus the total new-orders must equal the sum of the customer delivery count;
12. for any randomly selected customer, the balance plus the year-to-date payment must equal the sum of the order-line amount.

The TPC Benchmark C Standard Specification requires explicit demonstration that the conditions are satisfied for the first four conditions only.

To demonstrate that consistency is maintained, conditions 1-4 were verified for a sample of warehouses before and after the durability tests.

4.4 Isolation

Operations of concurrent transactions must yield results which are indistinguishable from the results which would be obtained by forcing each transaction to be serially executed to completion in some order.

*This property is commonly called **serializability**. Sufficient conditions must be enabled at either the system or application level to ensure serializability of transactions under any arbitrary mix of TPC-C transactions, unless otherwise specified by the transaction profile. The system or application must have full serializability enabled (i.e., repeated reads of the same rows within any committed transaction must return identical data when run concurrently with any arbitrary mix of TPC-C transactions), except in the case of Stock-Level transaction. For the Stock-Level transaction, the isolation requirement is relaxed to simply require that the transaction see only committed data.*

The TPC Benchmark C Standard (Revision 3.5) defines nine required tests to be performed to demonstrate that the required levels of transaction isolation are met.

For conventional locking schemes, isolation should be tested as described below. Systems that implement other isolation schemes may require different validation techniques. It is the responsibility of the test sponsor to disclose those techniques and the tests for them. If isolation schemes other than conventional locking are used, it is permissible to implement these tests differently provided full details are disclosed. (Examples of different validation techniques are shown in Isolation Test 7, Clause 3.4.2.7).

4.4.1 Isolation Test 1

This test demonstrates isolation for read-write conflicts of Order-Status and New-Order transactions.

The execution of the above test proceeded as follows:

1. An Order-Status transaction T0 was executed for a randomly selected customer, and the order returned was noted. T0 was committed
2. A New-Order transaction T1 was started for the same customer used in T0. T1 was stopped prior to COMMIT.
3. An Order-Status transaction T2 was started for the same customer used in T1. T2 completed and was committed without being blocked by T1. T2 returned the same order that T0 had returned.
4. T1 was allowed to complete and was committed.
5. An Order-Status transaction T3 was started for the same customer used in T1. T3 returned the order inserted by T1.

This outcome demonstrates serialization of T2 before T1. It has equivalent validity to the outcome specified in the Standard which supposes T1 to be serialized before T2.

4.4.2 Isolation Test 2

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

The execution of the above test proceeded as follows:

1. An Order-Status transaction T0 was executed for a randomly selected customer and the order returned was noted. T0 was committed.
2. A New-Order transaction T1 with an invalid item number, was started for the same customer used in T0. T1 was stopped immediately prior to ROLLBACK.
3. An Order-Status transaction T2 was started for the same customer used in T1. T2 completed and was committed without being blocked by T1. T2 returned the same order that T0 had returned.
4. T1 was allowed to ROLLBACK.
5. An Order-Status transaction T3 was started for the same customer used in T1. T3 returned the same order that T0 had returned.

4.4.3 Isolation Test 3

This test demonstrates isolation for write-write conflicts of two New-Order transactions.

The execution of the above test proceeded as follows:

1. The D_NEXT_O_ID of a randomly selected district was retrieved.
2. A New-Order transaction T1 was started for a randomly selected customer within the district used in step 1. T1 was stopped immediately prior to COMMIT.
3. Another New-Order transaction T2 was started for the same customer used in T1. T2 waited.
4. T1 was allowed to complete. T2 completed and was committed.

5. The order number returned by T1 was the same as the D_NEXT_O_ID retrieved in step 1. The order number returned by T2 was one greater than the order number returned by T1.
6. The D_NEXT_O_ID of the same district was retrieved again. It had been incremented by two (i.e. it was one greater than the order number returned by T2).

4.4.4 Isolation Test 4

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

The execution of the above test proceeded as follows:

1. The D_NEXT_O_ID of a randomly selected district was retrieved.
2. A New-Order transaction T1, with an invalid item number, was started for a randomly selected customer within the district used in step 1. T1 was stopped immediately prior to ROLLBACK.
3. Another New-Order transaction T2 was started for the same customer used in T1. T2 waited.
4. T1 was allowed to roll back, and T2 completed and was committed.
5. The order number returned by T2 was the same as the D_NEXT_O_ID retrieved in step 1.
6. The D_NEXT_O_ID of the same district was retrieved again. It had been incremented by one (i.e. one greater than the order number returned by T2).

4.4.5 Isolation Test 5

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

The execution of the above test proceeded as follows:

1. A query was executed to find out the customer who would be updated by the next delivery transaction for a randomly selected warehouse and district.
2. The C_BALANCE of the customer found in step 1 was retrieved.
3. A Delivery business transaction T1 was started for the same warehouse used in step 1. T1 was stopped immediately prior to the COMMIT of the database transaction corresponding to the district used in step 1.
4. A Payment transaction T2 was started for the same customer found in step 1. T2 waited.
5. T1 was allowed to complete. T2 completed and was committed.
6. The C_BALANCE of the customer found in step 1 was retrieved again. The C_BALANCE reflected the results of both T1 and T2.

4.4.6 Isolation Test 6

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

The execution of the above test proceeded as follows:

1. A query was executed to find out the customer who would be updated by the next delivery transaction for a randomly selected warehouse and district.
2. The C_BALANCE of the customer found in step 1 was retrieved.

3. A Delivery business transaction T1 was started for the same warehouse used in step 1. T1 was stopped immediately prior to the ROLLBACK of the database transaction corresponding to the district used in step 1.
4. A Payment transaction T2 was started for the same customer found in step 1. T2 waited.
5. T1 was allowed to ROLLBACK. T2 completed and was committed.

The C_BALANCE of the customer found in step 1 was retrieved again. The C_BALANCE reflected the results of only T2.

4.4.7 Isolation Test 7

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

The execution of the above test proceeded as follows:

1. The I_PRICE of two randomly selected items X and Y were retrieved.
2. A New-Order transaction T2 with a group of items including items X and Y was started. T2 was stopped immediately after retrieving the prices of all items. The prices of items X and Y retrieved matched those retrieved in step 1.
3. A transaction T3 was started to increase the price of items X and Y by 10%.
4. T3 did not stall and no transaction was rolled back. T3 was committed.
5. T2 was resumed, and the prices of all items were retrieved again within T2. The prices of items X and Y matched those retrieved in step 1.
6. T2 was committed.
7. The prices of items X and Y were retrieved again. The values matched the values set by T3.

- Execution followed *Case D* of *Clause 3.4.2.7*.

4.4.8 Isolation Test 8

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

The execution of the above test proceeded as follows:

1. An Order-Status transaction T1 was started for a randomly selected customer.
2. T1 was stopped immediately after reading the order table for the selected customer. The most recent order for that customer was found.
3. A New-Order transaction T2 was started for the same customer. T2 completed and was committed without being blocked by T1.
4. T1 was resumed and the ORDER table was read again to determine the most recent order for the same customer. The order found was the same as the one found in step 2.
5. T1 completed and was committed.

4.4.9 Isolation Test 9

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

The execution of the above test proceeded as follows:

1. The NO_D_ID of all new_ORDER rows for a randomly selected warehouse and district was changed to 11. The changes were committed.
2. A Delivery transaction T1 was started for the selected warehouse.
3. T1 was stopped immediately after reading the new_ORDER table for the selected warehouse and district. No qualifying row was found.
4. A New-Order transaction T2 was started for the same warehouse and district. T2 completed and was committed without being blocked by T1.
5. T1 was resumed and the new_ORDER table was read again. No qualifying row was found.
6. T1 completed and was committed.
7. The NO_D_ID of all new_ORDER rows for the selected warehouse and district was restored to the original value. The changes were committed.

4.5 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.

List of single failures:

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

Specified durability tests were executed to demonstrate satisfaction of the durability requirements for this implementation of TPC Benchmark C. One durability test, described below, covering the following failure situations was performed under the auditor's supervision:

- *Permanent irrecoverable failure of any single durable medium containing TPC-C database tables or recovery log data (Clause 3.5.3.1).*

Since all data and log files are similarly, and uniformly, striped across all 74 arrays, the tests for the loss of data and log durable media were combined. A test was performed under a load of 308,000 users on the full-scale database for the loss of recovery log and loss of data tests. Another durability test, described below, combining the following failure situations was performed under the auditor's supervision:

- *instantaneous interruption which requires system reboot [of processors] to recover. (Clause 3.5.3.2*)
- *failure of all or part of memory. (Clause 3.5.3.3).*

This test was performed under the full performance-measurement load of 308,000 users on the full-scale database built for 308,000 users.

4.5.1 Loss of Log and Data Disks

Because the log and data devices are Redundant Disk Arrays which each function independently of the rest of the system in ensuring data integrity under loss and/or replacement of any individual disk drive (and other failures as well), integrity under such failure and replacement does not entail any interruption in processing. The test below validates the durability by demonstrating persistence of the results of transactions processed both before and during these failures, validating the durability upon database recovery (in this instance, forced) of transactions which completed before the failure and the non-effect of transactions which did not complete.

1. The D_NEXT_O_ID fields for all rows in the DISTRICT table were summed up to determine the initial count of the total number of orders (count1).
2. A test was initiated with 308,000 terminals. On the driver system, completed/rolled-back transactions (including New-Orders) were recorded in a "success" file.
3. After running at steady state throughput levels for 5 minutes, one of the individual disks containing recovery log and data was unplugged from its array.
4. Because of the built-in redundancy in the disk array, the test continued normally.
5. On the system log files, messages appeared indicating that a disk was missing.
6. The test was finished on the driver.
7. The contents of the "success" file on the driver and the ORDER table were spot-compared to verify that records in the "success" file for completed New-Order transactions had corresponding records in the ORDER table.
8. Step 1 was repeated to determine the total number of orders (count2). Count2-count1 matched exactly the number of records for successful New Orders in the RTE "success" file.
9. Consistency test 3 was run on the database and the results were verified.
10. The disk was installed back in its slot. Messages appeared in the system log files indicating redundancy was restored. The disk array automatically copied the mirrored-pair mate of the missing disk onto the new disk.

4.5.2 Instantaneous Interruption and Loss of Memory

Instantaneous interruption and loss of memory tests were combined because the loss of power erases the contents of memory. This failure was induced while the benchmark was running by turning off the power supplies to the server.

1. The D_NEXT_O_ID fields for all rows in district table were summed up to determine the initial count of the total number of orders (count1).
2. Transactions were started at full load. On the driver system, completed/rolled-back transactions (including New-Orders) were recorded in a "success" file.
3. After six minute, the benchmark throughput reached the steady state level and the server systems were de-powered.
4. The test was aborted on the driver.
5. The server system was restarted.
6. The database was restarted and a recovery performed using the transaction log.

7. The contents of the "success" file on the driver and the ORDERs table were spot-compared to verify that records in the "success" file for completed New-Order transactions had corresponding records in the ORDERs table.

8. Step 1 was repeated to determine the current total number of orders (count2). Count2-count1 (=1,429,543) was 49 more than the number of records for successful New Orders in the RTE "success" file (=1,444,076 - 14,582 rolled-back). *This difference would be due only to transactions which were committed on the system under test but for which the output data was not displayed on the [emulated] input/output screen before the failure.*

9. Consistency test 3 was run on the database and the results were verified.

5 Clause 4 Related Items

5.1 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 overscaled 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.

The TPC-C database for this test was configured with 32,000 warehouses.

Table	Occurrences
Warehouse	32,000
District	320,000
Customer	960,000,000
History	960,000,000
Orders	960,000,000
New Orders	288,000,000
Order Line	9,599,323,450
Item	100,000
Stock	320,000,000

During the measurement only 30,800 warehouses and their associated data were accessed. This was confirmed using D_NEXT_O_1D and W_YTD as described in *Clause 4.2.2 Comment (2)*.

5.2 Database and Growth Layout

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

Table 5.2 indicates the distribution of the database tables over the disks of the tested and priced systems.

Table 5.2: Disk Usage in Tested System

I) root, swap, file systems:
==========

Use	Device	Size (GB)	Device Model
root+swap+file systems	/dev/dsk/c0t0d0	16	HP C5447A
file system	/dev/dsk/c0t0d1	16	HP C5447A
swap	/dev/dsk/c0t0d4	20	HP C5447A
swap	LVM -striped	64	XP 512
swap	LVM -striped	64	XP 512
swap	LVM -striped	64	XP 512
swap	LVM -striped	62	XP 512

II) Database files:
==========

We use 74 arrays. Every array is attached to the system via a Fibre Channel link.

Each array contains 15 18-GB or 36-GB disks drives, allocated to 4 RAID1 LUNs. After formatting and mirroring, the available capacity of the arrays with 18GB drives is 121.73GB, or 30.43GB per LUN. The capacity of the arrays with 36-GB drives is 243.46GB, or 60.87GB per LUN.

All LUNs numbered 0 are grouped together in a volume group called vgcassini1. Likewise, LUN 1s are in vgcassini2, Lun 2s in vgcassini3, and LUN 3s in vgcassini4. All the Oracle files, including the logs, are logical vloumes that are divided among these 4 volume groups. Each logical volume is striped 74-way across all the arrays.

Most of the space (about 4.28TB) in the volume groups on the tested system was unused during the performance tests, but is available to satisfy the 8-hour log and 60-day storage requirements.

The 74 arrays and their 4 luns are accessed via the following paths:

/dev/dsk/c57t0[d0|d1|d2|d3]
/dev/dsk/c58t0[d0|d1|d2|d3]
/dev/dsk/c59t0[d0|d1|d2|d3]
/dev/dsk/c164t0[d0|d1|d2|d3]
/dev/dsk/c192t0[d0|d1|d2|d3]
/dev/dsk/c165t0[d0|d1|d2|d3]
/dev/dsk/c156t0[d0|d1|d2|d3]
/dev/dsk/c157t0[d0|d1|d2|d3]
/dev/dsk/c158t0[d0|d1|d2|d3]
/dev/dsk/c66t0[d0|d1|d2|d3]
/dev/dsk/c142t0[d0|d1|d2|d3]
/dev/dsk/c65t0[d0|d1|d2|d3]
/dev/dsk/c160t0[d0|d1|d2|d3]
/dev/dsk/c161t0[d0|d1|d2|d3]
/dev/dsk/c162t0[d0|d1|d2|d3]

/dev/dsk/c152t0[d0|d1|d2|d3]
/dev/dsk/c153t0[d0|d1|d2|d3]
/dev/dsk/c154t0[d0|d1|d2|d3]
/dev/dsk/c116t0[d0|d1|d2|d3]
/dev/dsk/c117t0[d0|d1|d2|d3]
/dev/dsk/c118t0[d0|d1|d2|d3]
/dev/dsk/c203t0[d0|d1|d2|d3]
/dev/dsk/c17t0[d0|d1|d2|d3]
/dev/dsk/c18t0[d0|d1|d2|d3]
/dev/dsk/c41t0[d0|d1|d2|d3]
/dev/dsk/c42t0[d0|d1|d2|d3]
/dev/dsk/c43t0[d0|d1|d2|d3]
/dev/dsk/c179t0[d0|d1|d2|d3]
/dev/dsk/c180t0[d0|d1|d2|d3]
/dev/dsk/c181t0[d0|d1|d2|d3]
/dev/dsk/c183t0[d0|d1|d2|d3]
/dev/dsk/c184t0[d0|d1|d2|d3]
/dev/dsk/c185t0[d0|d1|d2|d3]
/dev/dsk/c175t0[d0|d1|d2|d3]
/dev/dsk/c176t0[d0|d1|d2|d3]
/dev/dsk/c177t0[d0|d1|d2|d3]
/dev/dsk/c135t0[d0|d1|d2|d3]
/dev/dsk/c170t0[d0|d1|d2|d3]
/dev/dsk/c134t0[d0|d1|d2|d3]
/dev/dsk/c168t0[d0|d1|d2|d3]
/dev/dsk/c191t0[d0|d1|d2|d3]
/dev/dsk/c167t0[d0|d1|d2|d3]
/dev/dsk/c27t0[d0|d1|d2|d3]
/dev/dsk/c26t0[d0|d1|d2|d3]
/dev/dsk/c79t0[d0|d1|d2|d3]
/dev/dsk/c202t0[d0|d1|d2|d3]
/dev/dsk/c193t0[d0|d1|d2|d3]
/dev/dsk/c201t0[d0|d1|d2|d3]
/dev/dsk/c200t0[d0|d1|d2|d3]
/dev/dsk/c189t0[d0|d1|d2|d3]
/dev/dsk/c198t0[d0|d1|d2|d3]
/dev/dsk/c188t0[d0|d1|d2|d3]
/dev/dsk/c187t0[d0|d1|d2|d3]
/dev/dsk/c136t0[d0|d1|d2|d3]
/dev/dsk/c49t0[d0|d1|d2|d3]
/dev/dsk/c195t0[d0|d1|d2|d3]
/dev/dsk/c173t0[d0|d1|d2|d3]
/dev/dsk/c172t0[d0|d1|d2|d3]
/dev/dsk/c171t0[d0|d1|d2|d3]
/dev/dsk/c50t0[d0|d1|d2|d3]
/dev/dsk/c196t0[d0|d1|d2|d3]
/dev/dsk/c197t0[d0|d1|d2|d3]
/dev/dsk/c214t0[d0|d1|d2|d3]
/dev/dsk/c213t0[d0|d1|d2|d3]
/dev/dsk/c212t0[d0|d1|d2|d3]
/dev/dsk/c206t0[d0|d1|d2|d3]
/dev/dsk/c205t0[d0|d1|d2|d3]
/dev/dsk/c204t0[d0|d1|d2|d3]
/dev/dsk/c218t0[d0|d1|d2|d3]
/dev/dsk/c217t0[d0|d1|d2|d3]
/dev/dsk/c216t0[d0|d1|d2|d3]

/dev/dsk/c210t0[d0|d1|d2|d3]
/dev/dsk/c209t0[d0|d1|d2|d3]
/dev/dsk/c208t0[d0|d1|d2|d3]

4) List of all Oracle datafiles and the corresponding device: (sorted by name)

Datafile	File #	Size	Tablespace	Logical volume
log01		160001		/dev/vgcassini3/rlog01
log02		160001		/dev/vgcassini4/rlog02
control01				/dev/vgcassini1/rcontrol01
control02				/dev/vgcassini1/rcontrol02
cust001	9	8001	CUST	/dev/vgcassini3/rcust001
cust002	306	8001	CUST	/dev/vgcassini4/rcust002
cust003	307	8001	CUST	/dev/vgcassini1/rcust003
cust004	293	8001	CUST	/dev/vgcassini2/rcust004
cust005	194	8001	CUST	/dev/vgcassini3/rcust005
cust006	82	8001	CUST	/dev/vgcassini4/rcust006
cust007	80	8001	CUST	/dev/vgcassini1/rcust007
cust008	309	8001	CUST	/dev/vgcassini2/rcust008
cust009	266	8001	CUST	/dev/vgcassini3/rcust009
cust010	44	8001	CUST	/dev/vgcassini4/rcust010
cust011	305	8001	CUST	/dev/vgcassini1/rcust011
cust012	137	8001	CUST	/dev/vgcassini2/rcust012
cust013	88	8001	CUST	/dev/vgcassini3/rcust013
cust014	263	8001	CUST	/dev/vgcassini4/rcust014
cust015	312	8001	CUST	/dev/vgcassini1/rcust015
cust016	36	8001	CUST	/dev/vgcassini2/rcust016
cust017	322	8001	CUST	/dev/vgcassini3/rcust017
cust018	35	8001	CUST	/dev/vgcassini4/rcust018
cust019	219	8001	CUST	/dev/vgcassini1/rcust019
cust020	253	8001	CUST	/dev/vgcassini2/rcust020
cust021	316	8001	CUST	/dev/vgcassini3/rcust021
cust022	119	8001	CUST	/dev/vgcassini4/rcust022
cust023	178	8001	CUST	/dev/vgcassini1/rcust023
cust024	117	8001	CUST	/dev/vgcassini2/rcust024
cust025	321	8001	CUST	/dev/vgcassini3/rcust025
cust026	67	8001	CUST	/dev/vgcassini4/rcust026
cust027	182	8001	CUST	/dev/vgcassini1/rcust027
cust028	212	8001	CUST	/dev/vgcassini2/rcust028
cust029	52	8001	CUST	/dev/vgcassini3/rcust029
cust030	210	8001	CUST	/dev/vgcassini4/rcust030
cust031	265	8001	CUST	/dev/vgcassini1/rcust031
cust032	240	8001	CUST	/dev/vgcassini2/rcust032
cust033	161	8001	CUST	/dev/vgcassini3/rcust033
cust034	261	8001	CUST	/dev/vgcassini4/rcust034
cust035	73	8001	CUST	/dev/vgcassini1/rcust035
cust036	115	8001	CUST	/dev/vgcassini2/rcust036
cust037	153	8001	CUST	/dev/vgcassini3/rcust037
cust038	228	8001	CUST	/dev/vgcassini4/rcust038
cust039	250	8001	CUST	/dev/vgcassini1/rcust039
cust040	124	8001	CUST	/dev/vgcassini2/rcust040
cust041	221	8001	CUST	/dev/vgcassini3/rcust041

```
cust042   320   8001   CUST    /dev/vgcassini4/rcust042
cust043    98   8001   CUST    /dev/vgcassini1/rcust043
cust044    86   8001   CUST    /dev/vgcassini2/rcust044
cust045   125   8001   CUST    /dev/vgcassini3/rcust045
cust046   141   8001   CUST    /dev/vgcassini4/rcust046
cust047   174   8001   CUST    /dev/vgcassini1/rcust047
cust048    77   8001   CUST    /dev/vgcassini2/rcust048
cust049   118   8001   CUST    /dev/vgcassini3/rcust049
cust050    56   8001   CUST    /dev/vgcassini4/rcust050
cust051   267   8001   CUST    /dev/vgcassini1/rcust051
cust052    70   8001   CUST    /dev/vgcassini2/rcust052
cust053   170   8001   CUST    /dev/vgcassini3/rcust053
cust054   159   8001   CUST    /dev/vgcassini4/rcust054
cust055    57   8001   CUST    /dev/vgcassini1/rcust055
cust056    65   8001   CUST    /dev/vgcassini2/rcust056
cust057   131   8001   CUST    /dev/vgcassini3/rcust057
cust058   236   8001   CUST    /dev/vgcassini4/rcust058
cust059   155   8001   CUST    /dev/vgcassini1/rcust059
cust060   132   8001   CUST    /dev/vgcassini2/rcust060
cust061    72   8001   CUST    /dev/vgcassini3/rcust061
cust062    81   8001   CUST    /dev/vgcassini4/rcust062
cust063    96   8001   CUST    /dev/vgcassini1/rcust063
cust064   328   8001   CUST    /dev/vgcassini2/rcust064
cust065   231   8001   CUST    /dev/vgcassini3/rcust065
cust066    29   8001   CUST    /dev/vgcassini4/rcust066
cust067   203   8001   CUST    /dev/vgcassini1/rcust067
cust068   252   8001   CUST    /dev/vgcassini2/rcust068
cust069   129   8001   CUST    /dev/vgcassini3/rcust069
cust070   190   8001   CUST    /dev/vgcassini4/rcust070
cust071   123   8001   CUST    /dev/vgcassini1/rcust071
cust072   243   8001   CUST    /dev/vgcassini2/rcust072
cust073   114   8001   CUST    /dev/vgcassini3/rcust073
cust074    40   8001   CUST    /dev/vgcassini4/rcust074
cust075   154   8001   CUST    /dev/vgcassini1/rcust075
cust076   195   8001   CUST    /dev/vgcassini2/rcust076
cust077    58   8001   CUST    /dev/vgcassini3/rcust077
cust078   136   8001   CUST    /dev/vgcassini4/rcust078
cust079   327   8001   CUST    /dev/vgcassini1/rcust079
cust080   280   8001   CUST    /dev/vgcassini2/rcust080
cust081   276   8001   CUST    /dev/vgcassini3/rcust081
cust082   233   8001   CUST    /dev/vgcassini4/rcust082
cust083   211   8001   CUST    /dev/vgcassini1/rcust083
cust084   324   8001   CUST    /dev/vgcassini2/rcust084
cust085   100   8001   CUST    /dev/vgcassini3/rcust085
cust086   208   8001   CUST    /dev/vgcassini4/rcust086
cust087   104   8001   CUST    /dev/vgcassini1/rcust087
cust088   158   8001   CUST    /dev/vgcassini2/rcust088
cust089   319   8001   CUST    /dev/vgcassini3/rcust089
cust090    30   8001   CUST    /dev/vgcassini4/rcust090
cust091   146   8001   CUST    /dev/vgcassini1/rcust091
cust092   172   8001   CUST    /dev/vgcassini2/rcust092
cust093    61   8001   CUST    /dev/vgcassini3/rcust093
cust094    83   8001   CUST    /dev/vgcassini4/rcust094
cust095   260   8001   CUST    /dev/vgcassini1/rcust095
cust096    79   8001   CUST    /dev/vgcassini2/rcust096
cust097   215   8001   CUST    /dev/vgcassini3/rcust097
```

```
cust098      102    8001   CUST       /dev/vgcassini4/rcust098
cust099      130    8001   CUST       /dev/vgcassini1/rcust099
cust100      108    8001   CUST       /dev/vgcassini2/rcust100
cust101       63    8001   CUST       /dev/vgcassini3/rcust101
cust102       76    8001   CUST       /dev/vgcassini4/rcust102
cust103      204    8001   CUST       /dev/vgcassini1/rcust103
cust104      192    8001   CUST       /dev/vgcassini2/rcust104
cust105      325    8001   CUST       /dev/vgcassini3/rcust105
cust106      220    8001   CUST       /dev/vgcassini4/rcust106
cust107      167    8001   CUST       /dev/vgcassini1/rcust107
cust108      171    8001   CUST       /dev/vgcassini2/rcust108
cust109      216    8001   CUST       /dev/vgcassini3/rcust109
cust110      103    8001   CUST       /dev/vgcassini4/rcust110
cust111      180    8001   CUST       /dev/vgcassini1/rcust111
cust112      315    8001   CUST       /dev/vgcassini2/rcust112
cust113       26    8001   CUST       /dev/vgcassini3/rcust113
cust114       20    8001   CUST       /dev/vgcassini4/rcust114
cust115      148    8001   CUST       /dev/vgcassini1/rcust115
cust116      107    8001   CUST       /dev/vgcassini2/rcust116
cust117      113    8001   CUST       /dev/vgcassini3/rcust117
cust118      176    8001   CUST       /dev/vgcassini4/rcust118
cust119      232    8001   CUST       /dev/vgcassini1/rcust119
cust120      128    8001   CUST       /dev/vgcassini2/rcust120
cust121      183    8001   CUST       /dev/vgcassini3/rcust121
cust122       62    8001   CUST       /dev/vgcassini4/rcust122
cust123      197    8001   CUST       /dev/vgcassini1/rcust123
cust124      121    8001   CUST       /dev/vgcassini2/rcust124
hist01        13   26001   HIST       /dev/vgcassini2/rhist01
hist02       336   26001   HIST       /dev/vgcassini3/rhist02
hist03       358   26001   HIST       /dev/vgcassini4/rhist03
hist04       359   26001   HIST       /dev/vgcassini1/rhist04
icust101       5    8001   ICUST1     /dev/vgcassini2/ricust101
icust102     202    8001   ICUST1     /dev/vgcassini3/ricust102
icust103     298    8001   ICUST1     /dev/vgcassini4/ricust103
icust104     295    8001   ICUST1     /dev/vgcassini1/ricust104
icust105     294    8001   ICUST1     /dev/vgcassini2/ricust105
icust106     308    8001   ICUST1     /dev/vgcassini3/ricust106
icust107      17    8001   ICUST1     /dev/vgcassini4/ricust107
icust201       7    8001   ICUST2     /dev/vgcassini1/ricust201
icust202     292    8001   ICUST2     /dev/vgcassini2/ricust202
icust203     279    8001   ICUST2     /dev/vgcassini3/ricust203
icust204      16    8001   ICUST2     /dev/vgcassini4/ricust204
icust205      31    8001   ICUST2     /dev/vgcassini1/ricust205
icust206     181    8001   ICUST2     /dev/vgcassini2/ricust206
icust207     139    8001   ICUST2     /dev/vgcassini3/ricust207
icust208     286    8001   ICUST2     /dev/vgcassini4/ricust208
icust209     301    8001   ICUST2     /dev/vgcassini1/ricust209
icust210     299    8001   ICUST2     /dev/vgcassini2/ricust210
icust211      51    8001   ICUST2     /dev/vgcassini3/ricust211
iord101       11   25001   IORD1      /dev/vgcassini4/riord101
iord102      355   25001   IORD1      /dev/vgcassini1/riord102
iord201       14   30001   IORD2      /dev/vgcassini4/riord201
iord202      365   30001   IORD2      /dev/vgcassini1/riord202
iord203      353   30001   IORD2      /dev/vgcassini2/riord203
istk01         6    8001   ISTK       /dev/vgcassini3/ristk01
istk02       303    8001   ISTK       /dev/vgcassini4/ristk02
```

```
istk03          302   8001  ISTK        /dev/vgcassini1/ristk03
istk04          290   8001  ISTK        /dev/vgcassini2/ristk04
istk05          177   8001  ISTK        /dev/vgcassini3/ristk05
istk06          289   8001  ISTK        /dev/vgcassini4/ristk06
istk07          300   8001  ISTK        /dev/vgcassini1/ristk07
istk08          310   8001  ISTK        /dev/vgcassini2/ristk08
istk09          206   8001  ISTK        /dev/vgcassini3/ristk09
istk10          296   8001  ISTK        /dev/vgcassini4/ristk10
istk11          304   8001  ISTK        /dev/vgcassini1/ristk11
istk12           66   8001  ISTK        /dev/vgcassini2/ristk12
istk13          248   8001  ISTK        /dev/vgcassini3/ristk13
istk14          109   8001  ISTK        /dev/vgcassini4/ristk14
istk15           85   8001  ISTK        /dev/vgcassini1/ristk15
istk16          254   8001  ISTK        /dev/vgcassini2/ristk16
misc01            4   4001  MISC        /dev/vgcassini1/rmisc01
nord01            8   9001  NORD        /dev/vgcassini4/rnord01
nord02          378   9001  NORD        /dev/vgcassini1/rnord02
oldroll01       375   8001  ROLL_0      /dev/vgcassini4/roldroll01
oldroll02       377   2001  LARGE_RBS_TS /dev/vgcassini1/roldroll02
ordl01           15  32001  ORDL        /dev/vgcassini1/rordl01
ordl02          368  32001  ORDL        /dev/vgcassini2/rordl02
ordl03          369  32001  ORDL        /dev/vgcassini3/rordl03
ordl04          367  32001  ORDL        /dev/vgcassini4/rordl04
ordl05          366  32001  ORDL        /dev/vgcassini1/rordl05
ordl06          371  32001  ORDL        /dev/vgcassini2/rordl06
ordl07          374  32001  ORDL        /dev/vgcassini3/rordl07
ordl08          338  32001  ORDL        /dev/vgcassini4/rordl08
ordl09          362  32001  ORDL        /dev/vgcassini1/rordl09
ordl10          339  32001  ORDL        /dev/vgcassini2/rordl10
ordl11          363  32001  ORDL        /dev/vgcassini3/rordl11
ordl12          360  32001  ORDL        /dev/vgcassini4/rordl12
ordl13          373  32001  ORDL        /dev/vgcassini1/rordl13
ordl14          361  32001  ORDL        /dev/vgcassini2/rordl14
ordl15          340  32001  ORDL        /dev/vgcassini3/rordl15
ordl16          347  32001  ORDL        /dev/vgcassini4/rordl16
ordl17          364  32001  ORDL        /dev/vgcassini1/rordl17
ordl18          337  32001  ORDL        /dev/vgcassini2/rordl18
ordl19          342  32001  ORDL        /dev/vgcassini3/rordl19
ordl20          346  32001  ORDL        /dev/vgcassini4/rordl20
ordl21          334  32001  ORDL        /dev/vgcassini1/rordl21
ordl22          343  32001  ORDL        /dev/vgcassini2/rordl22
ordl23          370  32001  ORDL        /dev/vgcassini3/rordl23
ordl24          350  32001  ORDL        /dev/vgcassini4/rordl24
ordl25          345  32001  ORDL        /dev/vgcassini1/rordl25
ordl26          357  32001  ORDL        /dev/vgcassini2/rordl26
ordl27          341  32001  ORDL        /dev/vgcassini3/rordl27
ordl28          344  32001  ORDL        /dev/vgcassini4/rordl28
ordl29          348  32001  ORDL        /dev/vgcassini1/rordl29
ordl30          351  32001  ORDL        /dev/vgcassini2/rordl30
ordl31          372  32001  ORDL        /dev/vgcassini3/rordl31
ordl32          356  32001  ORDL        /dev/vgcassini4/rordl32
ordl33          354  32001  ORDL        /dev/vgcassini1/rordl33
ordl34          349  32001  ORDL        /dev/vgcassini2/rordl34
ordl35          352  32001  ORDL        /dev/vgcassini3/rordl35
ordr01           12  25001  ORDR        /dev/vgcassini2/rordr01
ordr02          335  25001  ORDR        /dev/vgcassini3/rordr02
```

ordr03	379	25001	ORDR	/dev/vgcassini4/rordr03
roll01	2	2001	UNDO_TS	/dev/vgcassini1/rroll01
roll02	3	2001	UNDO_TS	/dev/vgcassini1/rroll02
stock001	10	8001	STOK	/dev/vgcassini3/rstock001
stock002	329	8001	STOK	/dev/vgcassini4/rstock002
stock003	262	8001	STOK	/dev/vgcassini1/rstock003
stock004	330	8001	STOK	/dev/vgcassini2/rstock004
stock005	314	8001	STOK	/dev/vgcassini3/rstock005
stock006	111	8001	STOK	/dev/vgcassini4/rstock006
stock007	333	8001	STOK	/dev/vgcassini1/rstock007
stock008	269	8001	STOK	/dev/vgcassini2/rstock008
stock009	270	8001	STOK	/dev/vgcassini3/rstock009
stock010	120	8001	STOK	/dev/vgcassini4/rstock010
stock011	126	8001	STOK	/dev/vgcassini1/rstock011
stock012	122	8001	STOK	/dev/vgcassini2/rstock012
stock013	230	8001	STOK	/dev/vgcassini3/rstock013
stock014	214	8001	STOK	/dev/vgcassini4/rstock014
stock015	187	8001	STOK	/dev/vgcassini1/rstock015
stock016	95	8001	STOK	/dev/vgcassini2/rstock016
stock017	47	8001	STOK	/dev/vgcassini3/rstock017
stock018	24	8001	STOK	/dev/vgcassini4/rstock018
stock019	93	8001	STOK	/dev/vgcassini1/rstock019
stock020	42	8001	STOK	/dev/vgcassini2/rstock020
stock021	196	8001	STOK	/dev/vgcassini3/rstock021
stock022	22	8001	STOK	/dev/vgcassini4/rstock022
stock023	78	8001	STOK	/dev/vgcassini1/rstock023
stock024	89	8001	STOK	/dev/vgcassini2/rstock024
stock025	186	8001	STOK	/dev/vgcassini3/rstock025
stock026	185	8001	STOK	/dev/vgcassini4/rstock026
stock027	34	8001	STOK	/dev/vgcassini1/rstock027
stock028	71	8001	STOK	/dev/vgcassini2/rstock028
stock029	223	8001	STOK	/dev/vgcassini3/rstock029
stock030	209	8001	STOK	/dev/vgcassini4/rstock030
stock031	54	8001	STOK	/dev/vgcassini1/rstock031
stock032	84	8001	STOK	/dev/vgcassini2/rstock032
stock033	55	8001	STOK	/dev/vgcassini3/rstock033
stock034	184	8001	STOK	/dev/vgcassini4/rstock034
stock035	69	8001	STOK	/dev/vgcassini1/rstock035
stock036	134	8001	STOK	/dev/vgcassini2/rstock036
stock037	277	8001	STOK	/dev/vgcassini3/rstock037
stock038	97	8001	STOK	/dev/vgcassini4/rstock038
stock039	332	8001	STOK	/dev/vgcassini1/rstock039
stock040	18	8001	STOK	/dev/vgcassini2/rstock040
stock041	39	8001	STOK	/dev/vgcassini3/rstock041
stock042	162	8001	STOK	/dev/vgcassini4/rstock042
stock043	166	8001	STOK	/dev/vgcassini1/rstock043
stock044	222	8001	STOK	/dev/vgcassini2/rstock044
stock045	60	8001	STOK	/dev/vgcassini3/rstock045
stock046	165	8001	STOK	/dev/vgcassini4/rstock046
stock047	106	8001	STOK	/dev/vgcassini1/rstock047
stock048	37	8001	STOK	/dev/vgcassini2/rstock048
stock049	207	8001	STOK	/dev/vgcassini3/rstock049
stock050	175	8001	STOK	/dev/vgcassini4/rstock050
stock051	168	8001	STOK	/dev/vgcassini1/rstock051
stock052	273	8001	STOK	/dev/vgcassini2/rstock052
stock053	200	8001	STOK	/dev/vgcassini3/rstock053

stock054	285	8001	STOK	/dev/vgcassini4/rstock054
stock055	91	8001	STOK	/dev/vgcassini1/rstock055
stock056	242	8001	STOK	/dev/vgcassini2/rstock056
stock057	271	8001	STOK	/dev/vgcassini3/rstock057
stock058	218	8001	STOK	/dev/vgcassini4/rstock058
stock059	199	8001	STOK	/dev/vgcassini1/rstock059
stock060	160	8001	STOK	/dev/vgcassini2/rstock060
stock061	311	8001	STOK	/dev/vgcassini3/rstock061
stock062	238	8001	STOK	/dev/vgcassini4/rstock062
stock063	151	8001	STOK	/dev/vgcassini1/rstock063
stock064	53	8001	STOK	/dev/vgcassini2/rstock064
stock065	257	8001	STOK	/dev/vgcassini3/rstock065
stock066	87	8001	STOK	/dev/vgcassini4/rstock066
stock067	179	8001	STOK	/dev/vgcassini1/rstock067
stock068	49	8001	STOK	/dev/vgcassini2/rstock068
stock069	28	8001	STOK	/dev/vgcassini3/rstock069
stock070	251	8001	STOK	/dev/vgcassini4/rstock070
stock071	59	8001	STOK	/dev/vgcassini1/rstock071
stock072	41	8001	STOK	/dev/vgcassini2/rstock072
stock073	258	8001	STOK	/dev/vgcassini3/rstock073
stock074	90	8001	STOK	/dev/vgcassini4/rstock074
stock075	25	8001	STOK	/dev/vgcassini1/rstock075
stock076	19	8001	STOK	/dev/vgcassini2/rstock076
stock077	21	8001	STOK	/dev/vgcassini3/rstock077
stock078	45	8001	STOK	/dev/vgcassini4/rstock078
stock079	272	8001	STOK	/dev/vgcassini1/rstock079
stock080	164	8001	STOK	/dev/vgcassini2/rstock080
stock081	318	8001	STOK	/dev/vgcassini3/rstock081
stock082	237	8001	STOK	/dev/vgcassini4/rstock082
stock083	99	8001	STOK	/dev/vgcassini1/rstock083
stock084	235	8001	STOK	/dev/vgcassini2/rstock084
stock085	278	8001	STOK	/dev/vgcassini3/rstock085
stock086	245	8001	STOK	/dev/vgcassini4/rstock086
stock087	145	8001	STOK	/dev/vgcassini1/rstock087
stock088	281	8001	STOK	/dev/vgcassini2/rstock088
stock089	217	8001	STOK	/dev/vgcassini3/rstock089
stock090	297	8001	STOK	/dev/vgcassini4/rstock090
stock091	224	8001	STOK	/dev/vgcassini1/rstock091
stock092	282	8001	STOK	/dev/vgcassini2/rstock092
stock093	157	8001	STOK	/dev/vgcassini3/rstock093
stock094	284	8001	STOK	/dev/vgcassini4/rstock094
stock095	227	8001	STOK	/dev/vgcassini1/rstock095
stock096	264	8001	STOK	/dev/vgcassini2/rstock096
stock097	259	8001	STOK	/dev/vgcassini3/rstock097
stock098	291	8001	STOK	/dev/vgcassini4/rstock098
stock099	138	8001	STOK	/dev/vgcassini1/rstock099
stock100	213	8001	STOK	/dev/vgcassini2/rstock100
stock101	133	8001	STOK	/dev/vgcassini3/rstock101
stock102	288	8001	STOK	/dev/vgcassini4/rstock102
stock103	256	8001	STOK	/dev/vgcassini1/rstock103
stock104	101	8001	STOK	/dev/vgcassini2/rstock104
stock105	110	8001	STOK	/dev/vgcassini3/rstock105
stock106	313	8001	STOK	/dev/vgcassini4/rstock106
stock107	229	8001	STOK	/dev/vgcassini1/rstock107
stock108	234	8001	STOK	/dev/vgcassini2/rstock108
stock109	23	8001	STOK	/dev/vgcassini3/rstock109

stock110	27	8001	STOK	/dev/vgcassini4/rstock110
stock111	50	8001	STOK	/dev/vgcassini1/rstock111
stock112	105	8001	STOK	/dev/vgcassini2/rstock112
stock113	48	8001	STOK	/dev/vgcassini3/rstock113
stock114	33	8001	STOK	/dev/vgcassini4/rstock114
stock115	38	8001	STOK	/dev/vgcassini1/rstock115
stock116	169	8001	STOK	/dev/vgcassini2/rstock116
stock117	116	8001	STOK	/dev/vgcassini3/rstock117
stock118	68	8001	STOK	/dev/vgcassini4/rstock118
stock119	43	8001	STOK	/dev/vgcassini1/rstock119
stock120	191	8001	STOK	/dev/vgcassini2/rstock120
stock121	140	8001	STOK	/dev/vgcassini3/rstock121
stock122	32	8001	STOK	/dev/vgcassini4/rstock122
stock123	152	8001	STOK	/dev/vgcassini1/rstock123
stock124	225	8001	STOK	/dev/vgcassini2/rstock124
stock125	135	8001	STOK	/dev/vgcassini3/rstock125
stock126	94	8001	STOK	/dev/vgcassini4/rstock126
stock127	74	8001	STOK	/dev/vgcassini1/rstock127
stock128	163	8001	STOK	/dev/vgcassini2/rstock128
stock129	92	8001	STOK	/dev/vgcassini3/rstock129
stock130	244	8001	STOK	/dev/vgcassini4/rstock130
stock131	239	8001	STOK	/dev/vgcassini1/rstock131
stock132	64	8001	STOK	/dev/vgcassini2/rstock132
stock133	275	8001	STOK	/dev/vgcassini3/rstock133
stock134	283	8001	STOK	/dev/vgcassini4/rstock134
stock135	46	8001	STOK	/dev/vgcassini1/rstock135
stock136	201	8001	STOK	/dev/vgcassini2/rstock136
stock137	127	8001	STOK	/dev/vgcassini3/rstock137
stock138	193	8001	STOK	/dev/vgcassini4/rstock138
stock139	112	8001	STOK	/dev/vgcassini1/rstock139
stock140	173	8001	STOK	/dev/vgcassini2/rstock140
stock141	246	8001	STOK	/dev/vgcassini3/rstock141
stock142	247	8001	STOK	/dev/vgcassini4/rstock142
stock143	331	8001	STOK	/dev/vgcassini1/rstock143
stock144	75	8001	STOK	/dev/vgcassini2/rstock144
stock145	326	8001	STOK	/dev/vgcassini3/rstock145
stock146	268	8001	STOK	/dev/vgcassini4/rstock146
stock147	143	8001	STOK	/dev/vgcassini1/rstock147
stock148	149	8001	STOK	/dev/vgcassini2/rstock148
stock149	189	8001	STOK	/dev/vgcassini3/rstock149
stock150	188	8001	STOK	/dev/vgcassini4/rstock150
stock151	323	8001	STOK	/dev/vgcassini1/rstock151
stock152	147	8001	STOK	/dev/vgcassini2/rstock152
stock153	142	8001	STOK	/dev/vgcassini3/rstock153
stock154	144	8001	STOK	/dev/vgcassini4/rstock154
stock155	205	8001	STOK	/dev/vgcassini1/rstock155
stock156	287	8001	STOK	/dev/vgcassini2/rstock156
stock157	249	8001	STOK	/dev/vgcassini3/rstock157
stock158	156	8001	STOK	/dev/vgcassini4/rstock158
stock159	274	8001	STOK	/dev/vgcassini1/rstock159
stock160	198	8001	STOK	/dev/vgcassini2/rstock160
stock161	241	8001	STOK	/dev/vgcassini3/rstock161
stock162	255	8001	STOK	/dev/vgcassini4/rstock162
stock163	317	8001	STOK	/dev/vgcassini1/rstock163
stock164	150	8001	STOK	/dev/vgcassini2/rstock164
stock165	226	8001	STOK	/dev/vgcassini3/rstock165

```
sys01          1    801  SYSTEM     /dev/vgcassini1/rsys01
tools01      376  20001  TOOLS      /dev/vgcassini3/rtools01
```

The distribution of the database tables over the disk arrays of the priced system is an extension of the distribution described in Table 5.2; some ancillary details are mentioned in Appendix E. 60-day storage growth requirements are met with the unused space of this configuration. Figure 1.2 shows the configuration of the priced-system disks.

5.3 Data Model & Interfaces

A statement must be provided that describes:

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

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

Oracle9i Database Enterprise Edition is a relational DBMS. SQL stored procedures were used, invoked through the Oracle Call Interface (OCI); the application code appears in Appendix A.

5.4 Partitions/Replications

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

No partitioning or replication was used.

5.5 Growth Requirements

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

See Appendix E.

6 Clause 5 Related Items

6.1 Throughput

Measured tpmC must be reported.

Table 6.1: Measured tpmC

tpmC®	389,434.40

6.2 Response Time

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

Table 6.2: Response Times

Response Times	Average	90th %-ile	Maximum
New-Order	0.29s	0.46s	30.22s
Payment	0.24s	0.42s	29.69s
Order-Status	0.25s	0.41s	28.99s
Delivery (interactive portion)	0.08s	0.09s	20.60s
Delivery (deferred portion)	0.29s	0.50s	28.30s
Stock-Level	0.23s	0.39s	28.44s
Menu	0.001s	0.10s	0.57s

6.3 Keying and Think Times

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

Table 6.3: Keying Times

Keying Times	Minimum	Average	Maximum
New Order	18.02s	18.03s	18.2s
Payment	3.01s	3.02s	3.19s
Order Status	2.01s	2.02s	2.19s
Interactive Delivery	2.01s	2.02s	2.19s
Stock Level	2.01s	2.02s	2.18s

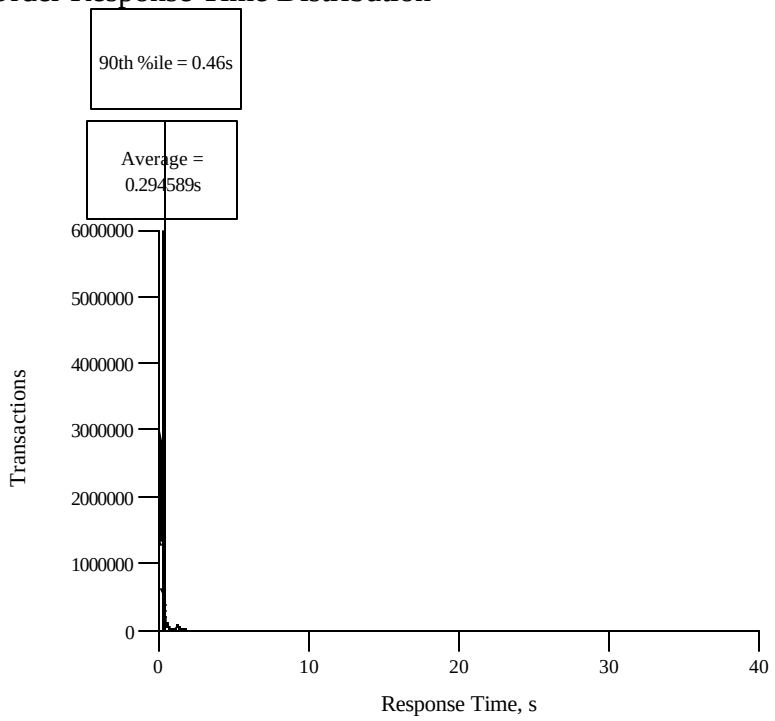
Table 6.4: Think Times

Think Times	Minimum	Average	Maximum
New Order	0.01s	12.02s	188.22s
Payment	0.01s	12.02s	222.72s
Order Status	0.01s	10.03s	165.36s
Interactive Delivery	0.01s	5.03s	82.24s
Stock Level	0.01s	5.02s	75.62s

6.4 Response Time Frequency Distribution Curves and Other Graphs

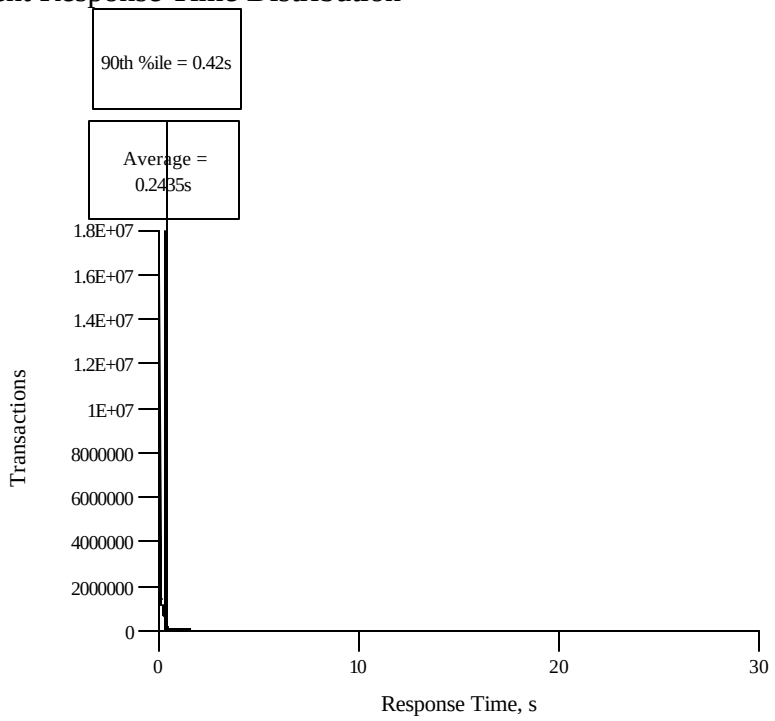
Response Time frequency distribution curves (see Clause 5.6.1) must be reported for each transaction type. The performance curve for response times versus throughput (see Clause 5.6.2) must be reported for the New-Order transaction. The Think Time frequency distribution curve (see Clause 5.6.3) must be reported for the New-Order transaction. A graph of throughput versus elapsed time (see Clause 5.6.5) must be reported for the New-Order transaction, and the measurement interval indicated.

Figure 6.1: New Order Response Time Distribution



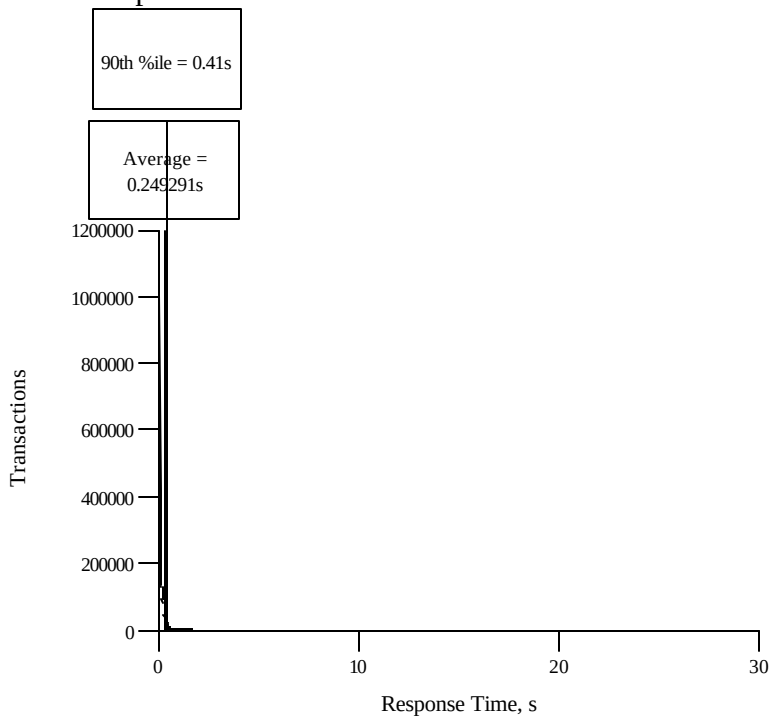
Response time frequency distribution for New Order transaction

Figure 6.2: Payment Response Time Distribution



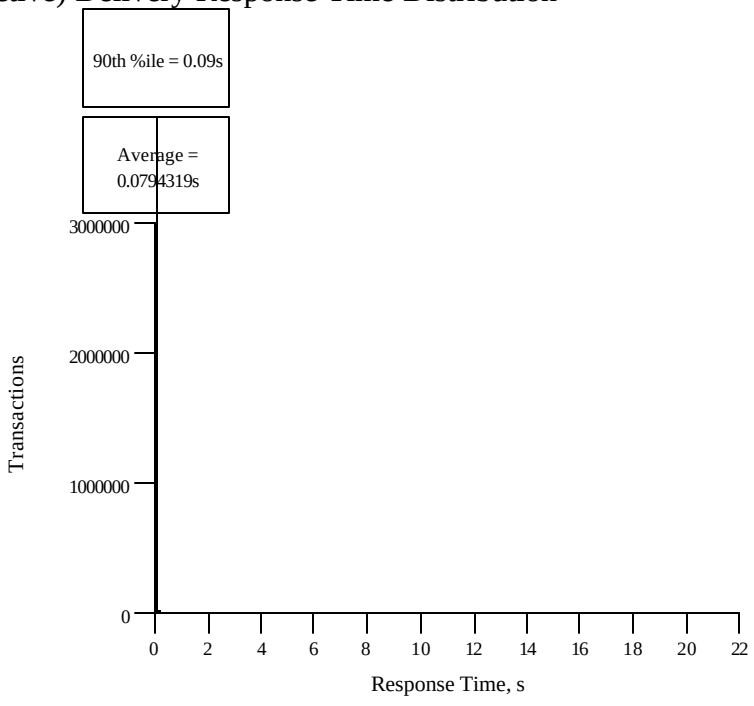
Response time frequency distribution for Payment transaction

Figure 6.3: Order Status Response Time Distribution



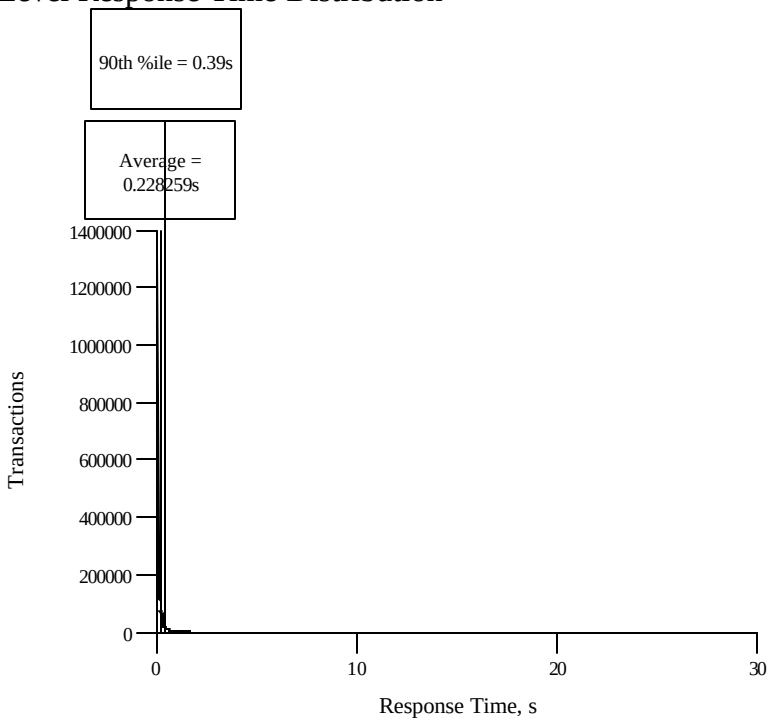
Response time frequency distribution for Order Status transaction

Figure 6.4: (Interactive) Delivery Response Time Distribution



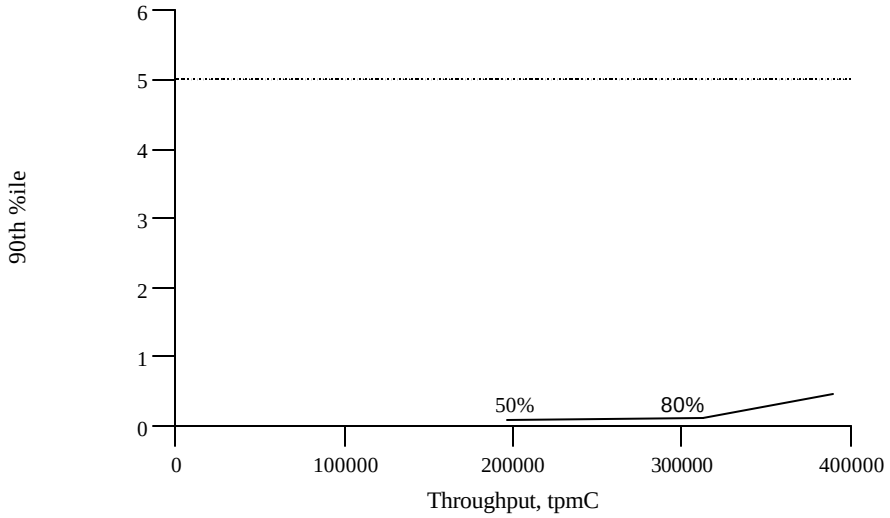
Response time frequency distribution for Delivery transaction

Figure 6.5: Stock Level Response Time Distribution



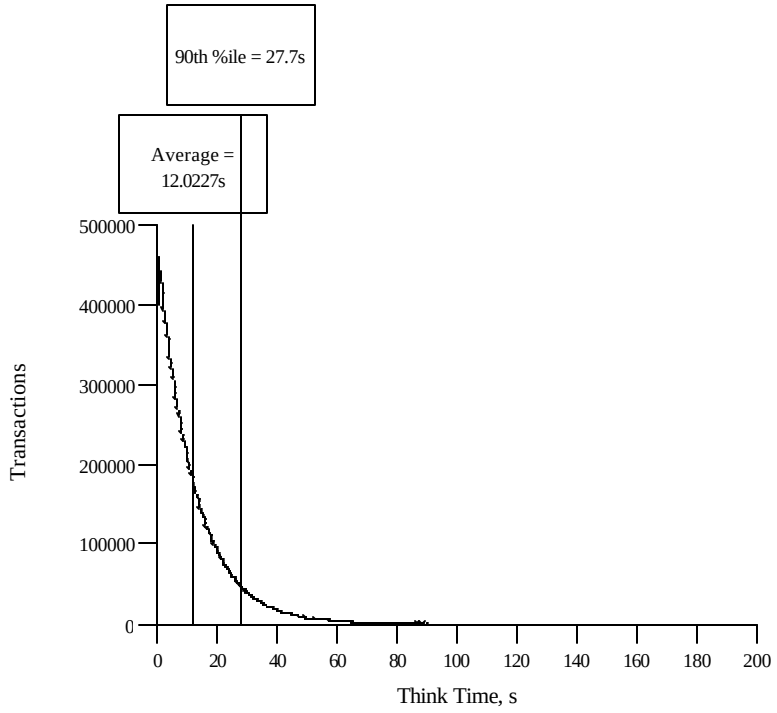
Response time frequency distribution for Stock Level transaction

Figure 6.6: Response Time Versus Throughput



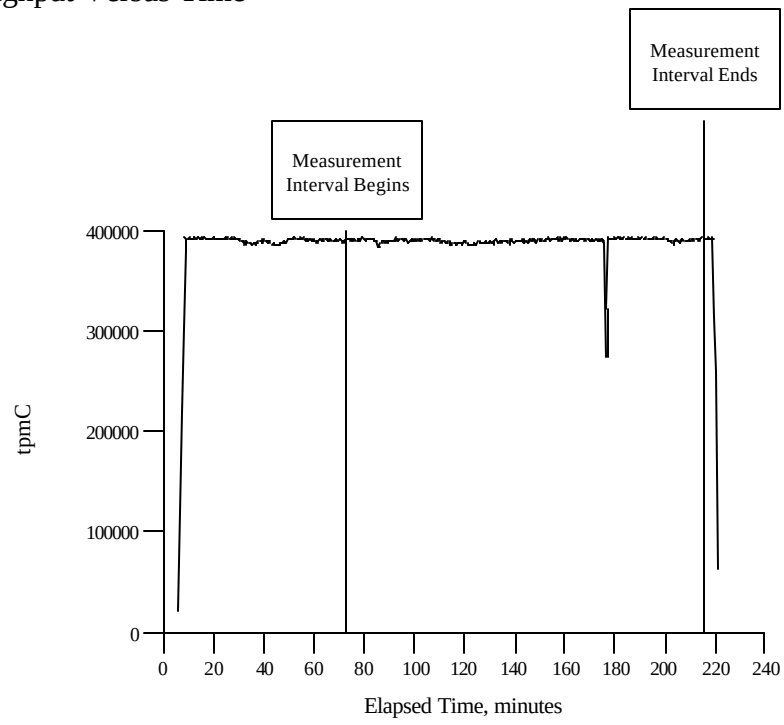
New Order response time versus Throughput

Figure 6.7: New Order Think Time Distribution



Think time frequency distribution for New Order transaction

Figure 6.8: Throughput Versus Time



Throughput of the New-Order transaction versus elapsed time

6.5 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.

Synchronization techniques employed in the benchmark process ensure that all emulated users are logged into the client and have opened the application before submitting transactions. Once all users are connected, each pauses a random amount of time before submitting transactions. The pause time distribution is controlled by a benchmark input parameter. The ramp-up interval is discernible in the graph of throughput over time. The data reduction also tracks the user load and indicates the point in time at which all users have submitted at least one transaction. The throughput is observed to be steady within the systematic and statistical variability of the measurement after all users are submitting transactions. A checkpoint is initiated upon the end of ramp-up.

6.6 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.

Modified database buffers migrated to disk on a least-recently-used basis independent of transaction commits. In addition, every block modification was protected by redo log records. These redo log records were written to the redo log buffer (in memory) and were flushed to a redo log file on disk either when the transaction committed or when the redo log buffer became full. However, due to the rapid commit during this benchmark, the redo log buffer was always flushed by a commit before it became full. Also, because many transactions were committing in a short period of time, a single flush of the redo log buffer resulted in many transactions' redo log data being written to disk. This is called group commit.

6.6.1 Checkpoint

During an Oracle9i Database Enterprise Edition checkpoint, all modified blocks in the shared buffer cache which had not been written to disk since the last checkpoint are written to disk.

6.6.2 Checkpoint Conditions

Oracle9i Database Enterprise Edition performs a checkpoint for the following conditions:

1. A redo log switch occurs.
2. The amount of data written to a redo log reaches the log_checkpoint_interval
3. The amount of time since the last checkpoint reaches the log_checkpoint_timeout.

6.6.3 Checkpoint Implementation

The first method listed above, i.e., a log switch when the redo log file filled up, was used to cause checkpoints. After the initial checkpoint, a log switch was performed every 28.6 minutes in average. All checkpoint intervals were within 2% of 28.6 minutes.

6.6.4 Serializable Transactions

Oracle supports serializable transaction isolation in full compliance with the SQL92 and TPC-C requirements. This is implemented by extending the multiversity concurrency control mechanism long supported by Oracle.

Oracle queries take no read locks and see only data committed as of the beginning of the query's execution. This means that readers and writers coexist without blocking one another, providing a high degree of concurrency and

consistency. While this mode does prevent reading dirty data, Oracle's default isolation level also permits a transaction that issues a query twice to see non-repeatable reads and phantoms, as defined in SQL92 and TPC-C.
Beginning with Oracle7 release 7.3, a transaction may request a high degree of isolation with the command SET TRANSACTION ISOLATION LEVEL SERIALIZABLE, as defined in SQL92. This transaction mode prevents read/write and write/write conflicts that would cause serializability failures. A session can establish this mode as its default mode, so the SET TRANSACTION command need not be issued in each transaction.

Oracle implements SERIALIZABLE mode by extending of the scope of read consistency from the individual query to the entire transaction. Instead of limiting a query to data committed at the time a query begins, in a serializable transaction all queries see data as of the beginning of the transaction. Thus, a serializable transaction sees a fixed snapshot of the database, established as of the beginning of the transaction.

All reads within a serializable transaction see only committed data as of the start of that transaction, plus new updates done by the transaction itself. All reads by a serializable transaction are therefore repeatable, as the transaction will access prior versions of data changed (or deleted) by other transactions after the start of the serializable transaction. This behavior also results in phantom protection since new rows created by other transactions will be invisible to the serializable transaction.

To ensure proper isolation, a serializable transaction cannot modify rows that were changed by other transactions after the beginning of the serializable transaction. If a serializable transaction attempts to update (or delete) a row previously changed by another transaction (serializable or not) since the beginning of the serializable transaction, the update (or delete) statement will fail with error ORA-08177: "Can't serialize access", and the statement will rollback.

SET TRANSACTION ISOLATION

LEVEL SERIALIZABLE;

SELECT …

SELECT…

UPDATE…

IF "Can't serialize access"

THEN ROLLBACK; LOOP and retry

ELSE COMMIT;

When a serializable transaction fails with this error, the application may either commit the work executed to that point, execute additional (but different) statements, or rollback the entire transaction. Repeated attempts to execute the same statement will always fail with the error "Can't serialize access", unless the other transaction has rolled back and released its lock. This error and these recovery options are similar to the treatment of deadlocks in systems that use read locks to ensure serializable execution. In both cases, conflicts between transactions cannot be resolved unless one of the transactions rolls back and restarts or commits without re-executing the statement receiving the error.

6.7 Measurement Period Duration

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

The measurement interval was 143 minutes.

6.8 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 weighted selection method of *Clause 5.2.4.1* was used. The weights were not adjusted during the run.

6.9 Transaction Mix

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

Table 6.5: Transaction Mix

Type	Percentage
New Order	44.81%
Payment	43.04%
Order Status	4.05%
Delivery	4.04%
Stock Level	4.05%

6.10 Transaction Statistics

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 trasaction must be disclosed. The percentage of remote order-lines entered per New-Order transaction must be disclosed. The percentage of remote Payment transactions must be disclosed. The percentage of customer selections by customer last name in the Payment and Order-Status transactions must be disclosed. The percentage of Delivery transactions skipped due to there being fewer than necessary orders in the New-Order table must be disclosed.

See Table 3.1

6.11 Checkpoint Count and Location

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

The number of completed checkpoints in the interval was 4. The time of seconds from the start of the measurement interval to the first checkpoint was 938 seconds. The Checkpoint Interval is 28.6 minutes.

The start time and duration for the checkpoints during the run were:

Start time	Duration
15:31:00	26:28
15:59:45	26:27
16:28:29	26:11
16:56:52	25:55
17:25:04	26:37
17:53:59	26:20

7 Clause 6 Related Items

7.1 RTE Description

If the RTE is commercially available, then its inputs must be specified. Otherwise, a description must be supplied of what inputs (e.g., scripts) to the RTE had been used. The RTE input parameters, code fragments, functions, et cetera used to generate each transaction input field must be disclosed. Comment: The intent is to demonstrate the RTE was configured to generate transaction input data as specified in Clause 2.

The RTE (Remote Terminal Emulator) on the driver system was developed at Hewlett-Packard and is not commercially available. Appendix D lists RTE input parameters and code fragments used to generate each transaction input field.

For this instance of the TPC-C benchmark, 28 drivers and 28 clients were used. The drivers emulated users logged in to the clients. An overview of the benchmark software on the drivers, clients and server is shown in Figure 7.1.

The benchmark is started with the **run** command on the driver system. **Run** controls the overall execution of the benchmark. After reading a configuration file, **run** starts TUXEDO on the client, collects pre-benchmark audit information and inserts a timestamp into a database audit table. When all the initial steps are completed, **run** invokes another program, **driver,** to start the benchmark. As the benchmark completes, **run** shuts down TUXEDO and collects the benchmark results into a single location.

Driver is the heart of the benchmark software. It simulates users as they log in, execute transactions and view results. **Driver** collects response times for each transaction and saves them in a file for future analysis.

Qualify is the post-processing analysis program. It produces the numerical summaries and histograms needed for the disclosure report.

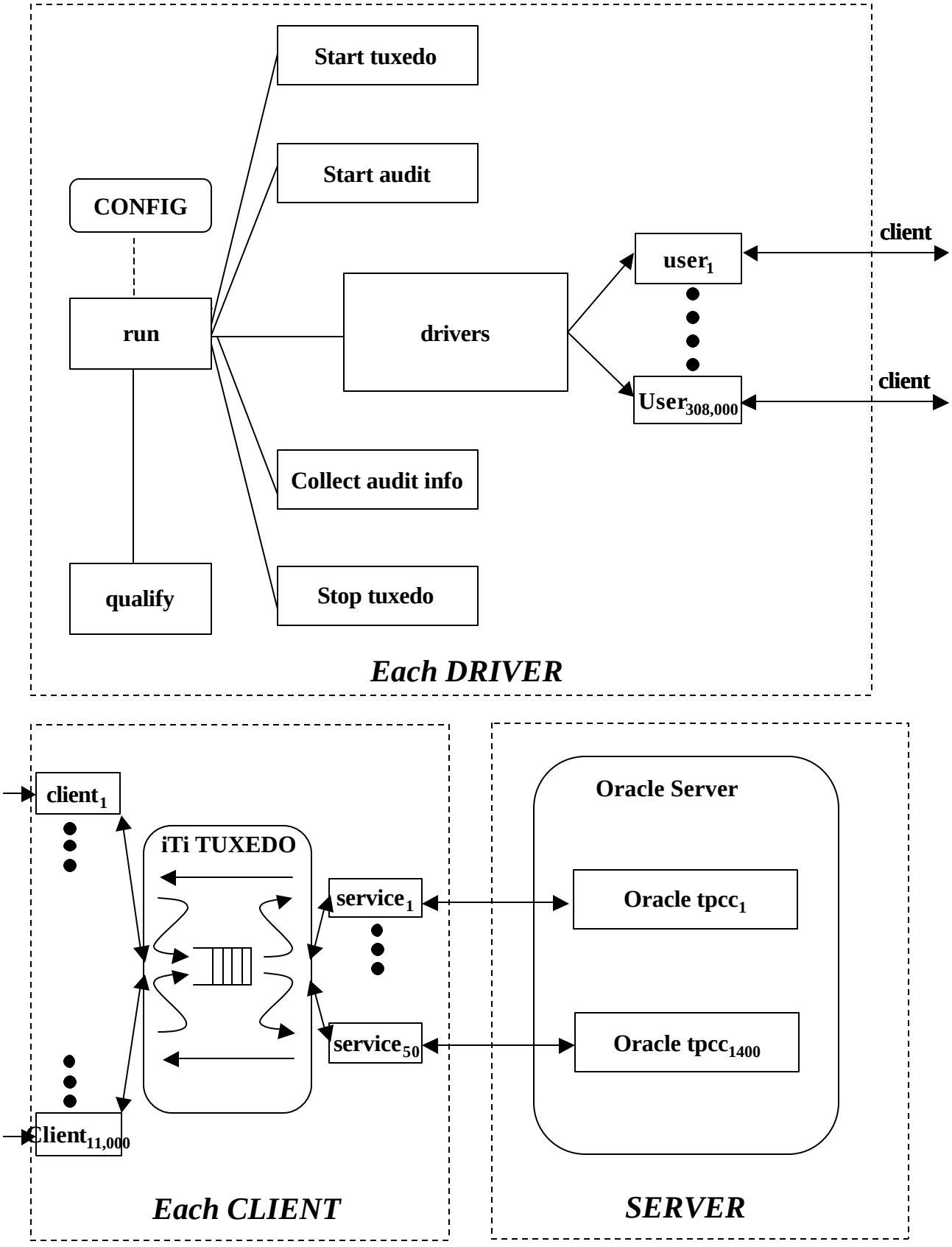


Figure 7.1: Benchmark Software

7.2 Lost Connections

No terminal connections were lost during the measurement interval**.**

7.3 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.

In the benchmark configuration, the 308,000 simulated workstations connected to the clients over 28 100BT lans through a single hp procurve switch In the priced configuration, the 308,000 worksations would connect to the clients via a combination of hubs and switches which eventually mutipexed down to 28 100BT lans.

7.4 Functional Diagrams

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

Figures 1.1 and 1.2 (in Chapter 1) show functional diagrams of the benchmark and configured systems. A description of the RTE and benchmark software is provided above.

7.5 Networks

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

Figures 1.1 and 1.2 (in Chapter 1) diagram the network configurations of the benchmark and configured systems, and represent the Driver connected via LAN replacing the workstations and HUBs connected via LANs. The clients are connected via 100 Base-T to an 100BT/1000BT-Ethernet switch which in turn is connected via 1000BT-Ethernet to the SUT.

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

Ethernet and 100 Base-T local area networks (LAN) with a bandwidth of 100 megabits per second are used in the tested/priced configurations. The 1000BT used has a bandwidth of 1000 megabits per second.

7.6 Client Substitution

Control Set Client Results

client1:(control set)	New_Order	Payment	Order-Status	Delivery	Stock-Level
average response time	0.25	0.21	0.21	0.09	0.19
90th precentile rsp.	0.33	0.28	0.3	0.1	0.28
transaction mix in %	44.79	43.06	4.05	4.05	4.04
(count# out of 4444613)	1990927	1913924	180099	179998	179665
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.03	2.03/10.06	2.02/5.05	2.02/5.04
client2: (control set)					
average response time	0.25	0.2	0.2	0.07	0.19
90th precentile rsp.	0.36	0.31	0.31	0.08	0.3
transaction mix in %	44.81	43.04	4.05	4.05	4.05
(count# out of 4448784)	1993636	1914665	180393	179982	180108
Menu response time	0	0	0	0	0
keying/think times	18.03/12.01	3.02/12.02	2.02/10.01	2.02/5.04	2.02/5.02
client3: (control set)					
average response time	0.25	0.2	0.21	0.08	0.19
90th precentile rsp.	0.35	0.3	0.3	0.08	0.29
transaction mix in %	44.84	43.02	4.04	4.04	4.06
(count# out of 4444267)	1992689	1912113	179565	179534	180366
Menu response time	0	0	0	0	0
keying/think times	18.03/12.04	3.02/12.03	2.02/10.03	2.02/5.00	2.02/5.05
client4: (control set)					
average response time	0.25	0.2	0.2	0.08	0.19
90th precentile rsp.	0.35	0.3	0.31	0.08	0.29
transaction mix in %	44.81	43.06	4.04	4.04	4.05
(count# out of 4446416)	1992521	1914710	179543	179634	180008
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.03	2.02/10.07	2.02/5.04	2.02/5.01
Control Set Aggregate:					
average response time	0.25	0.2	0.21	0.08	0.19
90th precentile rsp.	0.35	0.3	0.3	0.09	0.29
transaction mix in %	44.81	43.05	4.05	4.04	4.05
(count# out of 17784080)	7969773	7655412	719600	719148	720147
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.03	2.02/10.04	2.02/5.03	2.02/5.03

Non-Priced Client Results

	Non-Priced New_Order	Client Payment	Results Order-Status	Delivery	Stock-Level
client5:(non-priced)					
average response time	0.29	0.25	0.25	0.08	0.23
90th precentile rsp.	0.48	0.44	0.42	0.08	0.4
transaction mix in %	44.79	43.06	4.06	4.04	4.04
(count# out of 4437778)	1987829	1911093	180168	179495	179193
Menu response time	0	0	0	0	0
keying/think times	18.03/12.04	3.02/12.03	2.02/9.99	2.02/5.01	2.02/5.01
client6:(non-priced)					
average response time	0.29	0.24	0.24	0.08	0.23
90th precentile rsp.	0.46	0.42	0.4	0.08	0.38
transaction mix in %	44.81	43.03	4.04	4.06	4.05
(count# out of 4439160)	1989122	1910192	179538	180399	179909
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.03	2.02/10.04	2.02/5.03	2.02/5.00
client7:(non-priced)					
average response time	0.31	0.25	0.26	0.08	0.24
90th precentile rsp.	0.47	0.43	0.42	0.08	0.4
transaction mix in %	44.83	43.04	4.05	4.05	4.03
(count# out of 4434196)	1988059	1908446	179405	179392	178894
Menu response time	0	0	0	0	0
keying/think times	18.03/12.04	3.02/12.02	2.02/10.06	2.02/5.01	2.02/5.02
client8:(non-priced)					
average response time	0.3	0.25	0.26	0.08	0.24
90th precentile rsp.	0.47	0.43	0.42	0.09	0.4
transaction mix in %	44.82	43.06	4.04	4.04	4.05
(count# out of 4434274)	1987393	1909338	179022	178968	179553
Menu response time	0	0	0	0	0
keying/think times	18.03/12.03	3.02/12.03	2.02/10.03	2.02/5.06	2.02/5.03
client9:(non-priced)					
average response time	0.3	0.25	0.25	0.08	0.23
90th precentile rsp.	0.46	0.42	0.41	0.09	0.39
transaction mix in %	44.81	43.05	4.05	4.03	4.06
(count# out of 4437842)	1988635	1910299	179659	179032	180217
Menu response time	0	0	0	0	0
keying/think times	18.03/12.03	3.02/12.02	2.02/9.99	2.02/5.04	2.02/5.02
client10:(non-priced)					
average response time	0.31	0.26	0.26	0.08	0.24
90th precentile rsp.	0.52	0.48	0.45	0.08	0.43
transaction mix in %	44.85	43.02	4.04	4.05	4.03
(count# out of 4432659)	1988144	1907014	179168	179686	178647
Menu response time	0	0	0	0	0
keying/think times	18.03/12.03	3.02/12.03	2.02/10.02	2.02/5.03	2.02/5.02
client11:(non-priced)					
average response time	0.3	0.25	0.25	0.08	0.23

90th precentile rsp.	0.48	0.44	0.42	0.08	0.4
transaction mix in %	44.8	43.03	4.06	4.05	4.06
(count# out of4438082)	1988259	1909747	180089	179901	180086
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.03	2.02/10.01	2.02/5.02	2.02/5.04
client12:(non-priced)					
average response time	0.3	0.25	0.26	0.08	0.23
90th precentile rsp.	0.46	0.43	0.42	0.08	0.39
transaction mix in %	44.82	43.03	4.05	4.04	4.06
(count# out of 4437454)	1988898	1909617	179699	179189	180051
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.03	2.02/10.03	2.02/5.04	2.02/5.00
client13:(non-priced)					
average response time	0.3	0.25	0.26	0.08	0.24
90th precentile rsp.	0.48	0.45	0.43	0.09	0.41
transaction mix in %	44.81	43.06	4.04	4.05	4.03
(count# out of 4435725)	1987808	1909940	179242	179762	178973
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.03	2.02/ 10.06	2.02/5.03	2.02/5.00
client14:(non-priced)					
average response time	0.32	0.27	0.28	0.09	0.25
90th precentile rsp.	0.53	0.5	0.48	0.09	0.45
transaction mix in %	44.82	43.04	4.05	4.04	4.06
(count# out of 4431934)	1986354	1907372	179321	178979	179908
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.02	2.02/10.03	2.02/5.03	2.02/5.02
client15:(non-priced)					
average response time	0.3	0.24	0.25	0.08	0.23
90th precentile rsp.	0.44	0.4	0.39	0.08	0.37
transaction mix in %	44.8	43.08	4.05	4.04	4.04
(count# out of 4437863)	1988155	1911864	179649	179094	179101
Menu response time	0	0	0	0	0
keying/think times	18.03/12.03	3.02/12.02	2.02/10.01	2.02/5.03	2.02/5.02
client16:(non-priced)					
average response time	0.3	0.24	0.25	0.08	0.23
90th precentile rsp.	0.46	0.42	0.41	0.08	0.39
transaction mix in %	%	44.79	43.06	4.05	4.05
(count# out of 4431934)	4437784)	1987664	1910892	179545	179780
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.04	2.02/10.03	2.02/5.02	2.02/5.03
client17:(non-priced)					
average response time	0.3	0.25	0.25	0.08	0.24
90th precentile rsp.	0.48	0.44	0.43	0.09	0.41
transaction mix in %	44.79	43.06	4.06	4.05	4.04
(count# out of4437934)	1987670	1910900	180188	179843	179333
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.02	2.02/10.00	2.02/5.02	2.02/5.01
client18:(non-priced)					

average response time	0.3	0.25	0.26	0.08	0.24
90th precentile rsp.	0.49	0.45	0.44	0.08	0.41
transaction mix in %	44.8	43.06	4.03	4.05	4.05
(count# out of 4436705)	1987762	1910553	179006	179538	179846
Menu response time	0	0	0	0	0
keying/think times	18.03/12.03	3.02/12.02	2.02/10.06	2.02/5.05	2.02/5.04
client19:(non-priced)					
average response time	0.3	0.25	0.26	0.08	0.24
90th precentile rsp.	0.5	0.47	0.45	0.08	0.42
transaction mix in %	44.83	43.02	4.04	4.05	4.06
(count# out of 4437855)	1989269	1909223	179488	179726	180149
Menu response time	0	0	0	0	0
keying/think times	18.03/12.01	3.02/12.02	2.02/10.04	2.02/5.00	2.02/5.02
client20:(non-priced)					
average response time	0.29	0.25	0.25	0.08	0.23
90th precentile rsp.	0.48	0.44	0.43	0.08	0.4
transaction mix in %	44.84	43.02	4.05	4.04	4.05
(count# out of 4437245)	1989656	1908719	179704	179423	179743
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.04	2.02/9.98	2.02/5.04	2.02/5.03
client21:(non-priced)					
average response time	0.3	0.24	0.25	0.08	0.23
90th precentile rsp.	0.47	0.43	0.42	0.08	0.39
transaction mix in %	44.8	43.03	4.06	4.04	4.07
(count# out of 4440932)	1989720	1911013	180147	179263	180789
Menu response time	0	0	0	0	0
keying/think times	18.03/12.01	3.02/12.01	2.02/10.03	2.02/5.02	2.02/5.03
client22:(non-priced)					
average response time	0.32	0.26	0.27	0.08	0.25
90th precentile rsp.	0.54	0.5	0.48	0.08	0.45
transaction mix in %	44.84	43.02	4.05	4.04	4.05
(count# out of 4433406)	1987722	1907406	179341	179204	179733
Menu response time	0	0	0	0	0
keying/think times	18.03/12.01	3.02/12.02	2.02/10.06	2.02/5.02	2.02/5.05
client23:(non-priced)					
average response time	0.3	0.24	0.25	0.08	0.23
90th precentile rsp.	0.45	0.4	0.4	0.08	0.38
transaction mix in %	44.82	43.06	4.05	4.04	4.04
(count# out of 4438777)	1989266	1911413	179769	179213	179116
Menu response time	0	0	0	0	0
keying/think times	18.03/12.01	3.02/12.03	2.02/10.05	2.02/5.04	2.02/5.01
client24:(non-priced)					
average response time	0.31	0.26	0.27	0.09	0.24
90th precentile rsp.	0.49	0.46	0.44	0.09	0.42
transaction mix in %	44.82	43.04	4.04	4.05	4.05
(count# out of 431438)	1986204	1907271	179248	179311	179404
Menu response time	0	0	0	0	0
keying/think times	18.03/12.03	3.02/12.04	2.02/10.02	2.02/5.05	2.02/5.01

client25:(non-priced)					
average response time	0.3	0.25	0.26	0.08	0.24
90th precentile rsp.	0.5	0.47	0.45	0.08	0.42
transaction mix in %	44.84	43.03	4.05	4.05	4.03
(count# out of 4436678)	1989556	1909112	179594	179535	178881
Menu response time	0	0	0	0	0
keying/think times	18.03/12.01	3.02/12.02	2.02/10.01	2.02/5.03	2.02/5.02
client26:(non-priced)					
average response time	0.28	0.24	0.24	0.08	0.22
90th precentile rsp.	0.44	0.4	0.4	0.08	0.37
transaction mix in %	44.8	43.05	4.04	4.05	4.06
(count# out of 4442546)	1990125	1912365	179624	180032	180400
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.01	2.02/10.03	2.02/5.03	2.02/5.02
client27:(non-priced)					
average response time	0.31	0.25	0.26	0.08	0.24
90th precentile rsp.	0.49	0.45	0.44	0.08	0.41
transaction mix in %	44.81	43.05	4.05	4.03	4.05
(count# out of 4437958)	1988748	1910670	179884	179007	179649
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.01	2.02/10.04	2.02/5.00	2.02/5.02
client28:(non-priced)					
average response time	0.3	0.25	0.26	0.08	0.24
90th precentile rsp.	0.51	0.48	0.46	0.08	0.43
transaction mix in %	44.8	43.05	4.06	4.04	4.05
(count# out of 4436048)	1987328	1909910	179974	179365	179471
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.03	2.02/10.03	2.02/5.02	2.02/5.02

Non-Priced Aggregate:

average response time	0.3	0.25	0.26	0.08	0.23
90th precentile rsp.	0.48	0.44	0.43	0.09	0.4
transaction mix in %	44.81	43.04	4.05	4.04	4.05
47719346	45834369	4310472	4307137	4310949	
Menu response time	0	0	0	0	0
keying/think times	18.03/12.02	3.02/12.02	2.02/10.03	2.02/5.03	2.02/5.02

	New_Orders	**Users**	**New_Orders/User**	**non-priced**	**deviation**
client1(controlset)	1990927	11000	180.993		
client2:(controlset)	1993636	11000	181.24		
client3:(controlset)	1992689	11000	181.154		
client4:(controlset)	1992521	11000	181.138		
Control set:	7969773	44000	181.131		
client5:(non-priced)	1987829	11000	180.712	-0.419	(-0.23%)
client6:(non-priced)	1989122	11000	180.829	-0.302	(-0.17%)
client7:(non-priced)	1988059	11000	180.733	-0.398	(-0.22%)
client8:(non-priced)	1987393	11000	180.672	-0.459	(-0.25%)
client9:(non-priced)	1988635	11000	180.785	-0.346	(-0.19%)
client10:(non-priced)	1988144	11000	180.74	-0.391	(-0.22%)
client11:(non-priced)	1988259	11000	180.751	-0.38	(-0.21%)
client12:(non-priced)	1988898	11000	180.809	-0.322	(-0.18%)
client13:(non-priced)	1987808	11000	180.71	-0.421	(-0.23%)
client14:(non-priced)	1986354	11000	180.578	-0.553	(-0.31%)
client15:(non-priced)	1988155	11000	180.741	-0.39	(-0.22%)
client16:(non-priced)	1987664	11000	180.697	-0.434	(-0.24%)
client17:(non-priced)	1987670	11000	180.697	-0.434	(-0.24%)
client18:(non-priced)	1987762	11000	180.706	-0.425	(-0.23%)
client19:(non-priced)	1989269	11000	180.843	-0.288	(-0.16%)
client20:(non-priced)	1989656	11000	180.878	-0.253	(-0.14%)
client21:(non-priced)	1989720	11000	180.884	-0.247	(-0.14%)
client22:(non-priced)	1987722	11000	180.702	-0.429	(-0.24%)
client23:(non-priced)	1989266	11000	180.842	-0.289	(-0.16%)
client24:(non-priced)	1986204	11000	180.564	-0.567	(-0.31%)
client25:(non-priced)	1989556	11000	180.869	-0.262	(-0.14%)
client26:(non-priced)	1990125	11000	180.92	-0.211	(-0.12%)
client27:(non-priced)	1988748	11000	180.795	-0.336	(-0.19%)
client28:(non-priced)	1987328	11000	180.666	-0.465	(-0.26%)
Non-priced:	47719346	264000	180.755	-0.376	(-0.21%)

8 Clause 7 Related Items

8.1 System Pricing

A detailed list of hardware and software used in the priced system must be reported. Each item must have vendor part number, description, and release/revision level, and either general availability status or committed delivery data. If package-pricing is used contents of the package must be disclosed. Pricing source(s) 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.

Each priced configuration consists of an integrated system package, additional options, and components. Prices for all Hewlett-Packard products that are not provided by a third party quote are HP's US list prices. A one (1) year warranty is standard with all Hewlett-Packard products.

8.2 Support Pricing

The three year support pricing for Hewlett-Packard products is based on twenty-four (24) months of monthly support costs; thirty-six (36) months minus the twelve month warranty period. The Oracle Corporation support pricing is based on thirty-six (36) months of monthly support costs. The following support products were priced in the benchmark:

- HP four-hour on-site repair hardware support,
- HP telephone support for software and updates
- Oracle Corporation Standard Technical Support and,
- BEA TUXEDO Standard Technical Support

8.2.1 HP Hardware Support

HP's on-site support for hardware provides service 24 hour, seven day support.

8.2.2 HP Software Support

HP Software Support provides the following:

- Access to the HP Response Centers for fault isolation and problem solving assistance,
- Guaranteed two (2) hour call return, immediate response for critical calls,
- Electronic access to product and support information,
- Electronic access to software patches,
- Right-to-use and copy software updates.

8.3 Oracle Corporation Standard Technical Support

Oracle Corporation Standard Technical Support includes:

Product updates,

- A regular technical publication,
- Unlimited, toll-free telephone service to assist in product installation, syntax, and usage that is available 24 hours, seven days a week.

8.4 Availability

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

see below

8.5 Priced System Configuration

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

8.6 Throughput, Price/Performance, and Availability Date

A statement of the measured tpmC® *as well as the respective calculations for the 3-year pricing, price/performance (price/tpmC®).*

For Throughput and Price/Performance, please see page iv and v. The Price/Performance calculation spreadsheet appears on page v.

All hardware components in this test of the HP 9000 Superdome Enterprise Server system are currently available. HP-UX 11.i 64-bit incorporating is available Now. Oracle9i Database Enterprise Edition is be available Now.

9 Clause 9 Related Items

9.1 Auditor's Report

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

If audited, the auditor's attestation letter must be made readily available to the public as part of the Full Disclosure Report, but a detailed report from the auditor is not required.

This implementation of the TPC Benchmark® C on the HP 9000 Superdome Enterprise Server was audited by Tom Sawyer for Performance Metrics, Inc..

Tom Sawyer
Performance Metrics, Inc.
137 Yankton Street, Suite 101
Folsom, CA 95630
U.S.A.
Phone: 916 985-1131
Fax: 916 985-1185

The attestation letter is shown on the following pages.

November 16, 2001

Mr. Andreas Hotea
Business Critical Computing Unit
Hewlett-Packard Company
19111 Pruneridge Avenue
Cupertino, CA 95014

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

Platform:	Hewlett-Packard 9000 SuperDome Enterprise Server
Database Manager:	Oracle9i Database Enterprise Edition
Operating System:	HP-UX 11i
Transaction Manager:	Tuxedo version 6.4

Server: Hewlett-Packard 9000 SuperDome Enterprise Server				
CPU's	**Memory**	**Disks**	**90% Response**	**tpmC**
64 PA-8700 @ 750 MHz	**Main: 256 GB iCache: 0.75MB each dCache: 1.5MB each**	**1,080 18GB 30 36GB 36 9GB**	**0.46 sec.**	**389,434.40**

28 Clients: of which 4 (configured below) were used for pricing Hewlett-Packard Models C3700		
CPU	**Memory**	**Disks**
PA-8700@ 750 MHz	**Main: 8.0 GB iCache: 0.75MB dCache: 1.5MB**	**1 @ 18GB**

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

- The transactions were correctly implemented.
- The database was properly sized and populated.
- The database was properly scaled with 32,000 warehouses of which 30,800 were used in the measurement. I verified that d_next_o_id and w_ytd contained initial values for the unused warehouses.
- The ACID properties were met.
- The durability data loss and log loss tests were performed on the measured system. Both log and data were stripped across all disks and all disks were mirrored. One drive was removed and the system continued to run until shut down after 5 minutes.
- Input data was generated according to the specified percentages.
- Eight hours of mirrored log space was configured on the measured system.
- Eight hours of dynamic table growth space was configured on the measured system.
- The 60-day space calculation was verified; 660 36GB disks were substituted for 18GB disks – see Auditor's Notes..
- Measurement cycle times had no delays as Telenet was used as the connection protocol.
- There were 308,000 user contexts present on the system.
- Each emulated user started with a different random number seed.
- The NURand constants used for database load and at run time were 1 and 86.
- The steady state portion of the test was 2 hours and 23 minutes.
- One checkpoint was taken before the measured interval.
- The average checkpoint interval was 28:34. Four checkpoints were taken during the measured interval. Checkpoints were in progress at the beginning and end of the interval. The measurement interval of 143 minutes is 10 seconds longer than 5-times the average checkpoint cycle.
- The system pricing was checked for major components and maintenance**.**

Auditor Notes:

A mix of client machines was used, 4 of which were the type priced. I examined results for the individual clients, and aggregates of the priced and unpriced clients. The data satisfied the requirements of TAB-ID 334.

Groups of 15 36GB disks were substituted for 18GB disks to satisfy the 60-day requirement. I examined SAR data which showed the 36GB disks to be faster.

The 36GB drives had copies of the operating system, swap and Oracle. SAR data showed that these disks were not active during the measurement. They were replaced in the priced system by space on the 74 disk arrays.

Sincerely,

Tom Sawyer
Auditor

10 Report Availability

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

Transaction Processing
Performance Council
c/o Shanley Public Relations
650 N. Winchester Blvd.
Suite 1
San Jose, CA 95128

or your local Hewlett-Packard sales office.

Appendix A Client/Server Source

This appendix contains the source and makefiles for all client and server programs.

A.1 Client Front-End

client/client.c

```
/*************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 10:53:26 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*************************************************************************/
/*********************************************************************
History
941101 JVM Fixed login screen to detect broken connection (used to loop)
941013 JVM Added audit strings to the login form
941013 VM modified the getfield procedure to add digit and char check
       according to the field type.
941014 VM added the status_msg routine to display transaction results.
941015 VM added zip routine to format zip codes and phone routine
       to format phone numbers.
*********************************************************************/


#include "iobuf.h"
#include "tpcc.h"
#include <signal.h>

#define until(c)  while(!(c))


/* a generic transaction variable. */
generic_trans generic_transaction;
generic_trans *trans=&generic_transaction;

/* global variables set up during initialization */
int user;
ID warehouse;
ID district;

main(argc, argv)
   int argc;
   char **argv;
   {
   int key;

   /* setup the transactions */
   key = setup(argc, argv);

   /* repeat until done */
   while (key != '9' && key != EOF)
     {

     /* get the menu choice */
     key = menu_read();

     /* process according to the choice */
     switch(key)
        {
        case '1': key = neworder(&trans->neworder); break;
        case '2': key = payment(&trans->payment); break;
        case '3': key = ordstat(&trans->ordstat); break;
        case '4': key = delivery(&trans->delivery); break;
        case '5': key = stocklev(&trans->stocklev); break;
        case EOF: break;
        case '9': break;
        default:    msgline("Please enter a valid menu choice");
        }
     }

   /* done */
   cleanup();
   }



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

Neworder form processing

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

define_iobuf(neworder_form, 900);

int neworder(trans)
   neworder_trans *trans;
   {
   int key;
   display(neworder_form);
   key = neworder_read(trans);
   if (key != ENTER) return key;
   neworder_transaction(trans);
   neworder_write(trans);
   return key;
   }


int neworder_read(trans)
   neworder_trans *trans;
   {
   int i;
   int field;
   int key;
   int ol;

   /* Our warehouse number is fixed */
   trans->W_ID = warehouse;
   trans->D_ID = EMPTY_NUM;

   /* assume nothing set yet */
   trans->C_ID = EMPTY_NUM;
   for (i=0; i<15; i++)
      {
      trans->item[i].OL_I_ID = EMPTY_NUM;
      trans->item[i].OL_QUANTITY = EMPTY_NUM;
      trans->item[i].OL_SUPPLY_W_ID = EMPTY_NUM;
      }

   /* Process fields until done */
   for (field = 1; field > 0; field = next_field(field, key, 47))
      retry: switch (field)
         {

         case 1: key = read_number(4, 29, &trans->D_ID, 2);
                 break;

         case 2: key = read_number(5, 12, &trans->C_ID, 4);
                 break;

         case 3: case 6: case 9: case 12: case 15:
         case 18: case 21: case 24: case 27: case 30:
         case 33: case 36: case 39: case 42: case 45:
            ol = (field - 3) / 3;
            key = read_number(9+ol, 2, &trans->item[ol].OL_SUPPLY_W_ID,6);
            break;

         case 4: case 7: case 10: case 13: case 16:
         case 19: case 22: case 25: case 28: case 31:
         case 34: case 37: case 40: case 43: case 46:
            ol = (field - 3) / 3;
            key = read_number(9+ol,10, &trans->item[ol].OL_I_ID, 6);
            break;

         case 5: case 8: case 11: case 14: case 17:
         case 20: case 23: case 26: case 29: case 32:
         case 35: case 38: case 41: case 44: case 47:
            ol = (field - 3) / 3;
            key = read_number(9+ol, 45, &trans->item[ol].OL_QUANTITY, 2);
            break;
         }

   /* abort the screen if requested */
   if (key != ENTER)
      return key;

   /* calculate how many items were entered */
   for (i=15; i>0; i--)
      if ((trans->item[i-1].OL_I_ID != EMPTY_NUM) ||
                  (trans->item[i-1].OL_SUPPLY_W_ID != EMPTY_NUM) ||
                  (trans->item[i-1].OL_QUANTITY != EMPTY_NUM)) break;
   trans->O_OL_CNT = i;

   /* make sure all necessary fields are filled in */
   if (trans->D_ID == EMPTY_NUM)
      {field=1; msgline("Please specify district"); goto retry;}
   if (trans->C_ID == EMPTY_NUM)
      {field=2; msgline("Please specify customer id"); goto retry;}
   if (trans->O_OL_CNT == 0)
      {field=3; msgline("Please enter at least one orderline"); goto retry;}
   for (i=0; i<trans->O_OL_CNT; i++)
      {
      if (trans->item[i].OL_SUPPLY_W_ID == EMPTY_NUM)
```

```
        {field=i*3+3; msgline("Please enter supply warehouse"); goto retry;}
      if (trans->item[i].OL_I_ID == EMPTY_NUM)
        {field=i*3+4; msgline("Please enter Item id"); goto retry;}
      if (trans->item[i].OL_QUANTITY == EMPTY_NUM
       || trans->item[i].OL_QUANTITY <= 0)
        {field=i*3+5; msgline("Please enter quantity > 0"); goto retry;}
      }

    /* decide if they were all local */
    for (i=0; i<trans->O_OL_CNT; i++)
      if (trans->item[i].OL_SUPPLY_W_ID != trans->W_ID ) break;
    trans->all_local = (i == trans->O_OL_CNT);

    /* display number of order lines */
    number(6, 42, trans->O_OL_CNT, 2);

    msgline("");
    flush();
    return key;
    }



neworder_write(t)
    neworder_trans *t;
    {
    int i;
    MONEY amount, total_amount, cost;

    /* Rev. 3.3 error checking: both of the following branches are
     * skipped.  We'll go to status and print an error message.
     */

    /* CASE: invalid item, display only these values */
    if (t->status == E_INVALID_ITEM)
        {
        text(5, 25, t->C_LAST);
        text(5,52,  t->C_CREDIT);
        number(6, 15, t->O_ID, 8);
        }

    /* CASE: everything OK, display everything */
    else if (t->status == OK)
        {
        text(5, 25, t->C_LAST);
        text(5,52,  t->C_CREDIT);
        number(6, 15, t->O_ID, 8);
        date(4, 61, t->O_ENTRY_D);
        real(5, 64, t->C_DISCOUNT * 100, 5, 2);
        real(6, 59, t->W_TAX*100, 5, 2);
        real(6, 74, t->D_TAX*100, 5, 2);

        total_amount = 0;
        for (i=0; i < t->O_OL_CNT; i++)
            {

            /* keep track of amount of each line and total */
            amount = t->item[i].I_PRICE * t->item[i].OL_QUANTITY;
            total_amount += amount;

            /* display the item line */
            text(9+i, 19, t->item[i].I_NAME);
            number(9+i, 51, t->item[i].S_QUANTITY, 3);
            position(9+i, 58);   pushc(t->item[i].brand_generic);
            money(9+i, 62, t->item[i].I_PRICE, 7);
            money(9+i, 71, amount, 8);
            }

        /* Clear the screen of any empty input fields */
        clear_screen();

        /* display the total cost */
        text(24, 63, "Total:");
        cost = total_amount * (1 - t->C_DISCOUNT) * (1 + t->W_TAX + t->D_TAX);
        money(24, 71, cost, 9);
        }

    /* display the status message */
    status(24, 1, t->status);
    }


neworder_setup()
    {
    int item;
    iobuf *old;

    /* start with an empty form */
    reset(neworder_form);

    /* redirect the data to a special menu buffer */
    old = out_buf;  out_buf = neworder_form;

    /* clear the iobuf below the menu */
    position(3,1);
    clear_screen();

    /* set up all the field labels */
    text(3, 36, "New Order");
    text(4, 1, "Warehouse:");
    number(4, 12, warehouse, 6);
    text(4, 19, "District:");
    empty(4, 29, 2);
    text(4, 55, "Date:");
    text(5, 1, "Customer:");
    empty(5, 12, 4);
    text(5, 19, "Name:");
    text(5, 44, "Credit:");
    text(5, 57, "Disc.:");
    text(6, 1, "Order Number:");
    text(6, 25, "Number of Lines:");
    text(6, 52, "W_Tax:");
    text(6, 67, "D_Tax:");
    text(8, 2, "Supp_W Item_Num   Item_Name");
    text(8, 45, "Qty Stock   B/G     Price  Amount");

    /* display blank fields for each item */
    for (item = 1; item <= 15; item++)
        {
        empty(8+item, 2, 6);
        empty(8+item, 10, 6);
        empty(8+item, 45, 2);
        }

    trigger();

    /* restore to the previous I/O buffer */
    out_buf = old;
    }



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

Payment form processing

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

define_iobuf(payment_form, 400);


int payment(trans)
    payment_trans *trans;
    {
    int key;
    display(payment_form);
    key = payment_read(trans);
    if (key != ENTER)  return key;
    payment_transaction(trans);
    payment_write(trans);
    return key;
    }


payment_setup()
    {
    int item;
    iobuf *old;

    /* start with an empty form */
    reset(payment_form);

    /* redirect the data to a special menu buffer */
    old = out_buf;  out_buf = payment_form;

    /* clear the iobuf below the menu */
    position(3,1);
    clear_screen();

    /* set up all the field labels */
    text(3, 38, "Payment");
    text(4, 1, "Date:");
    text(6, 1, "Warehouse:");
    number(6, 12, warehouse, 6);
    text(6, 42, "District:");
    empty(6, 52, 2);
    text(11, 1, "Customer:");
    empty(11, 11, 4);
    text(11, 17, "Cust-Warehouse:");
    empty(11, 33, 6);
    text(11, 40, "Cust-District:");
    empty(11, 54, 2);
    text(12, 1, "Name:");
    empty(12, 29, 16);
    text(12, 50, "Since:");
    text(13, 50, "Credit:");
```

```
    text(14, 50, "%Disc:");
    text(15, 50, "Phone:");
    text(17, 1, "Amount Paid:");
    empty(17, 23, 8);
    text(17, 37, "New Cust-Balance:");
    text(18, 1, "Credit Limit:");
    text(20, 1, "Cust-Data:");
    trigger();

    out_buf = old;
    }


int payment_read(t)
    payment_trans *t;
    {
    int i;
    int field;
    int key;

    /* Our warehouse number is fixed */
    t->W_ID = warehouse;
    t->C_ID = EMP TY_NUM;
    t->D_ID = EMPTY_NUM;
    t->C_W_ID = EMPTY_NUM;
    t->C_D_ID = EMPTY_NUM;
    t->H_AMOUNT = EMPTY_FLT;
    t->C_LAST[0] = ' \0';

    /* Process fields until done */
    for (field = 1; field > 0; field = next_field(field, key, 6))
        retry: switch (field)
            {

        case 1: key = read_number(6, 52, &t->D_ID, 2);
                break;

         case 2:
            /* if last name specified, skip this field */
            if (t->C_LAST[0] !=' \0')
                break;

            /* read in the customer id */
            key = read_number(11, 11, &t->C_ID, 4);

            /* if specified, don't allow last name to be entered */
            if (t->C_ID != EMPTY_NUM)
                {
                blanks(12, 29, 16);
                t->C_LAST[0] = ' \0';
                }

            /* refresh the C_LAST underlines, if possibly needed */
            else if (t->C_LAST[0] == '\0')
                empty(12, 29, 16);
            break;

        case 3: key = read_number(11, 33,  &t->C_W_ID, 6);
                break;

        case 4: key = read_number(11, 55, &t->C_D_ID, 2);
                break;

         case 5:
            /* skip this field if C_ID was already specified */
            if (t->C_ID != EMPTY_NUM)
                break;

            /* read in the customer last name */
            key = read_text(12, 29, t->C_LAST, 16);

            /* if specified, don't allow c_id to be entered */
            if (t->C_LAST[0] !=' \0')
                {
                blanks(11, 11, 4);
                t->C_ID = EMPTY_NUM;
                }

            /* refresh the C_ID underlines, if possibly needed */
            else if (t->C_ID == EMPTY_NUM)
                empty(11, 11, 4);
            break;

        case 6: key = read_money(17, 23, &t->H_AMOUNT, 8);
                break;
         }

    /* if Aborted, then done */
    if (key != ENTER)
        return key;


    /* Make sure all the fields were entered */
    if (t->D_ID == EMPTY_NUM)
        {field=1; msgline("Please enter district id"); goto retry;}
    if (t->C_ID == EMPTY_NUM && t->C_LAST[0] == '\0')
        {field=2; msgline("C_ID or C_LAST must be entered"); goto retry;}
    if (t->C_W_ID == EMPTY_NUM)
        {field=3; msgline("Please enter customer's warehouse"); goto retry;}
    if (t->C_D_ID == EMPTY_NUM)
        {field=4; msgline("please enter customer's district"); goto retry;}
    if (t->H_AMOUNT == EMPTY_FLT)
        {field=6; msgline("Please enter payment amount"); goto retry;}
    if (t->H_AMOUNT <= 0)
                {field=6; msgline("Please enter a positive payment"); goto retry;}


    t->byname = (t->C_ID == EMPTY_NUM);
    msgline("");
    flush();
    return key;
    }


payment_write(t)
    payment_trans *t;
    {

    /* if errors, display a message and quit */
    if (t->status != OK)
        {
        status(24, 1, t->status);
        return;
        }

    /* display the screen */
    date(4, 7, t->H_DATE);
    text(7, 1, t->W_STREET_1);
    text(7, 42, t->D_STREET_1);
    text(8, 1, t->W_STREET_2);
    text(8, 42, t->D_STREET_2);
    text(9, 1, t->W_CITY);
    text(9, 22, t->W_STATE);
    zip(9, 25, t->W_ZIP);
    text(9, 42, t->D_CITY);
    text(9, 63, t->D_STATE);
    zip(9, 66, t->D_ZIP);
    number(11, 11, t->C_ID, 4);
    text(12, 9, t->C_FIRST);
    text(12, 26, t->C_MIDDLE);
    text(12, 29, t->C_LAST);
    date_only(12, 58, t->C_SINCE);
    text(13, 9, t->C_STREET_1);
    text(13, 58, t->C_CREDIT);
    text(14, 9, t->C_STREET_2);
    real(14, 58, t->C_DISCOUNT*100, 5, 2);  /* percentage or fraction? */
    text(15, 9, t->C_CITY);
    text(15, 30, t->C_STATE);
    zip(15, 33, t->C_ZIP);
    phone(15, 58, t->C_PHONE);
    money(17, 17, t->H_AMOUNT,14);
    money(17, 55, t->C_BALANCE, 15);
    money(18, 17, t->C_CREDIT_LIM, 14);

    /* Display cust data if bad credit. */
    if (t->C_CREDIT[0] == 'B' && t->C_CREDIT[1] == 'C')
        long_text(20, 12, t->C_DATA, 50);
    }



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

ORDSTAT form processing

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

define_iobuf(ordstat_form, 300);


int ordstat(t)
    ordstat_trans *t;
    {
    int key;
    display(ordstat_form);
    key = ordstat_read(trans);
    if (key != ENTER)  return key;
    ordstat_transaction(trans);
    ordstat_write(trans);
    return key;
    }


ordstat_setup()
```

```
    {
    int item;
    iobuf *old;

    /* start with an empty form */
    reset(ordstat_form);

    /* redirect the data to a special menu buffer */
    old = out_buf;  out_buf = ordstat_form;

    /* clear the iobuf below the menu */
    position(3,1);
    clear_screen();

    /* set up all the field labels */
    text(3, 35, "Order-Status");
    text(4, 1, "Warehouse:");
    number(4, 12, warehouse, 6);
    text(4, 19, "District:");
    empty(4, 29, 2);
    text(5, 1, "Customer:");
    empty(5, 11, 4);
    text(5, 18, "Name:");
    empty(5, 44, 16);
    text(6, 1, "Cust-Balance:");
    text(8, 1, "Order-Number");
    text(8, 26, "Entry-Date:");
    text(8, 60, "Carrier-Number:");
    text(9, 1, "Supply-W");
    text(9, 14, "Item-Num");
    text(9, 25, "Qty");
    text(9, 33,"Amount");
    text(9, 45, "Delivery-Date");

    trigger();

    /* done */
    out_buf = old;
    }



int ordstat_read(t)
    ordstat_trans *t;
    {
    int i;
    int field;
    int key;

    /* Our warehouse number is fixed */
    t->W_ID = warehouse;
    t->C_ID = EMPTY_NUM;
    t->D_ID = EMPTY_NUM;
    t->C_LAST[0] = '\0';

    /* Process fields until done */
    for (field = 1; field > 0; field = next_field(field, key, 3))
        retry: switch (field)
            {

            case 1: key = read_number(4, 29, &t->D_ID, 2);
                    break;

            case 2:
                /* if last name specified, skip this field */
                if (t->C_LAST[0] != '\0')
                    break;

                /* read in the customer id */
                key = read_number(5, 11, &t->C_ID, 4);

                /* if specified, don't allow last name to be entered */
                if (t->C_ID != EMPTY_NUM)
                    {
                    blanks(5, 44, 16);
                    t->C_LAST[0] = '\0';
                    }

                /* refresh the C_LAST underlines, if possibly needed */
                else if (t->C_LAST[0] == '\0')
                    empty(5, 44, 16);
                break;

            case 3:
                /* skip this field if C_ID was already specified */
                if (t->C_ID != EMPTY_NUM)
                    break;

                /* read in the customer last name */
                key = read_text(5, 44, t->C_LAST, 16);

                /* if specified, don't allow c_id to be entered */
                if (t->C_LAST[0] != '\0')
                    {
                    blanks(5, 11, 4);
                    t->C_ID = EMPTY_NUM;
                    }

                /* refresh the C_ID underlines, if possibly needed */
                else if (t->C_ID == EMPTY_NUM)
                    empty(5, 11, 4);
                break;
            }

        /* if Aborted, then done */
        if (key != ENTER)
            return key;


        /* ensure all the necessary fields were entered */
        if (t->D_ID == EMPTY_NUM)
            {field=1; msgline("Please enter district id"); goto retry;}
        if (t->C_ID == EMPTY_NUM && t->C_LAST[0] == '\0')
            {field=2;  msgline("C_ID or C_LAST must be entered"); goto retry;}


        t->byname = (t->C_ID == EMPTY_NUM);
        msgline("");
        flush();
        return key;
        }


ordstat_write(t)
    ordstat_trans *t;
    {
    int i;

    /* if errors, display a status message and quit */
    if (t->status != OK)
        {
        status(24, 1, t->status);
        return;
        }

    /* display the results */
    number(5, 11, t->C_ID, 4);
    text(5, 24, t->C_FIRST);
    text(5, 41, t->C_MIDDLE);
    text(5, 44, t->C_LAST);
    money(6, 15, t->C_BALANCE, 10);
    number(8, 15, t->O_ID, 8);
    date(8, 38, t->O_ENTRY_DATE);
    if (t->O_CARRIER_ID > 0)
        number(8, 76, t->O_CARRIER_ID, 2);

    for (i=0; i< t->ol_cnt; i++)
        {
        number(i+10, 3, t->item[i].OL_SUPPLY_W_ID, 6);
        number(i+10, 14, t->item[i].OL_I_ID, 6);
        number(i+10, 25, t->item[i].OL_QUANTITY, 2);
        money(i+10, 32, t->item[i].OL_AMOUNT, 9);
        date_only(i+10, 47, t->item[i].OL_DELIVERY_DATE);
        }
    }




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

delivery form processing

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

define_iobuf(delivery_form, 300);



int delivery(t)
    delivery_trans *t;
    {
    int key;
    display(delivery_form);
    key = delivery_read(trans);
    if (key != ENTER)  return key;
    delivery_enque(trans);
    delivery_write(trans);
    return key;
    }


delivery_setup()
    {
    int item;
```

```
    iobuf *old;

    /* start with an empty form */
    reset(delivery_form);

    /* redirect the data  to a special menu buffer */
    old = out_buf;  out_buf = delivery_form;

    /* clear the iobuf below the menu */
    position(3,1);
    clear_screen();

    /* set up all the field labels */
    text(3, 38, "Delivery");
    text(4, 1, "Warehouse:");
    number(4, 12, warehouse, 6);
    text(6, 1, "Carrier Number:");
    empty(6, 17, 2);

    trigger();

    /* done */
    out_buf = old;
    }



int delivery_read(t)
    delivery_trans *t;
    {
    int i;
    int field;
    int key;

    /* Our warehouse number is fixed */
    t->W_ID = warehouse;
    t->O_CARRIER_ID = EMPTY_NUM;

    /* Process fields until done */
    for (field = 1; field > 0; field = next_field(field, key, 1))
        retry: switch (field)
            {
            case 1: key = read_number(6, 17, &t->O_CARRIER_ID, 2);
                    break;
            }

    /* if Aborted, then done */
    if (key != ENTER)
        return key;

    /* Must enter the carrier id */
    if ((t->O_CARRIER_ID == EMPTY_NUM) ||
                    (t->O_CARRIER_ID < 1) ||
                    (t->O_CARRIER_ID > 10))
        {field=1; msgline("Please enter a Carrier Number within 1 and 10"); goto retry; }

    /* clear the message line */
    msgline("");
    flush();
    return key;
    }


delivery_write(t)
    delivery_trans *t;
    {
    if (t->status == OK)
        text(8, 1, "Execution Status: Delivery has been queued");
    else
        status(8, 1, t->status);
    }



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

stocklev form processing

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

define_iobuf(stocklev_form, 300);


int stocklev(t)
    stocklev_trans *t;
    {
    int key;
    display(stocklev_form);
    key = stocklev_read(trans);
    if (key != ENTER)  return key;
    stocklev_transaction(trans);
    stocklev_write(trans);
    return key;
    }


stocklev_setup()
    {
    int item;
    iobuf *old;

    /* start with an empty form */
    reset(stocklev_form);

    /* redirect the data to a special menu buffer */
    old = out_buf;  out_buf = stocklev_form;

    /* clear the iobuf below the menu */
    position(3,1);
    clear_screen();

    /* set up all the field labels */
    text(3, 35, "Stock-Level");
    text(4, 1, "Warehouse:");
    number(4, 12, warehouse, 6);
    text(4, 19, "District:");
    number(4, 29, district, 2);
    text(6, 1, "Stock Level Threshold:");
    empty(6, 24, 2);
    text(8, 1, "low stock");

    trigger();

    /* done */
    out_buf = old;
    }



int stocklev_read(t)
    stocklev_trans *t;
    {
    int field;
    int key;

    t->W_ID = warehouse;
    t->D_ID = district;
    t->threshold = EMPTY_NUM;

    /* Process fields until done */
    for (field = 1; field > 0; field = next_field(field, key, 1))
        retry: switch (field)
            {
            case 1: key = read_number(6, 24, &t->threshold, 2);
                    break;
            }

    /* if Aborted, then done */
    if (key != ENTER)
        return key;


    /* make sure the necessary fields were entered */
    if ((t->threshold == EMPTY_NUM) ||
        (t->threshold < 10) ||
                    (t->threshold > 20))

        {field=1; msgline("Please enter a threshold within 10 and 20"); goto retry; }

    /* clear the message line */
    msgline("");
    flush();
    return key;
    }


stocklev_write(t)
    stocklev_trans *t;
    {
    if (t->status == OK)
        number(8, 12, t->low_stock, 3);
    else
      status(10, 1, t->status);
    }



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

login form processing

******************************************************************
******************************************************************/
```

```
int login()
   {
   int field;
   int key;
   char auditstr[21];
   int w_id, d_id;

   /* assume the default values */
   w_id = warehouse;
   d_id = district;
   auditstr[0] = ' \0';

   /* display the login menu */
   position(1,1);  clear_screen();
   text(3, 30, "Please login.");
   text(5,5,"Warehouse:");
   number(5, 16, w_id, 6);
   text(5, 24, "District:");
   number(5, 34, d_id, 2);
   text(15, 5, "Audit String:");
   text(15, 19, CLIENT_AUDIT_STRING);
   empty(16, 19, 20);
   trigger();

   /* Get values until done */
   for (field = 1; field > 0;  field = next_field(field, key, 3))
      retry: switch (field)
         {
         case 1:
            key = read_number(5, 16, &w_id, 6, Num);
            break;

         case 2:
            key = read_number(5, 34, &d_id, 2, Num);
            break;

         case 3:
            key = read_text(16, 19, auditstr, 20);
            break;
         }

   if (key != ENTER)
      return EOF;

   if (w_id == EMPTY_NUM && warehouse == EMPTY_NUM)
      {
      msgline("You must enter a warehouse id");
      field =1;
      goto retry;
      }

   if (d_id == EMPTY_NUM && district == EMPTY_NUM)
      {
      msgline("You must enter a district id");
      field = 2;
      goto retry;
      }

   if (w_id != EMPTY_NUM)
      warehouse = w_id;
   if (d_id != EMPTY_NUM)
      district = d_id;

   /* done */
   flush();
   return key;
   }



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

menu form processing

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

menu_setup()
   {

   /* display the menu on the iobuf -- never erased */
   position(1, 1);
   clear_screen();
   string("(1)New-Order (2)Payment (3)Order-Status ");
   string("(4)Delivery (5)StockLevel (9)Exit");
   }

int menu_read()
   {
   position(1, 1);
   trigger();
   return getkey();
   }



int next_field(current, key, max)
   int current;
   int key;
   int max;
   {

   if (key == BACKTAB)
      if (current == 1)    return max;
      else                 return current-1;
   else if (key == TAB)
      if (current == max)  return 1;
      else                 return current+1;
   else
      return 0;
   }



msgline(str)
   char *str;
   {
   position(24, 1);
   clear_screen();
   string(str);
   flush();  /* Needed? */
   }



int setup(argc, argv)
   int argc;
   char **argv;
   {
   int key;

   /* Ignore SIGPIPE, since they occur normally */
   signal(SIGPIPE, SIG_IGN);

   /* get the user, warehouse and district numbers */
   warehouse = EMPTY_NUM;
   district = EMPTY_NUM;
   key = login();
   user = warehouse*DIST_PER_WARE + district + 1;

   /* set up the forms */
   menu_setup();
   neworder_setup();
   payment_setup();
   ordstat_setup();
   delivery_setup();
   stocklev_setup();

   /* connect to the delivery queue */
   delivery_init(user);

   /* connect to the transaction processor */
   transaction_begin(user);

   return key;
   }



cleanup()
   {

   /* detach from transaction engine */
   transaction_done();

   /* detach from the delivery queue */
   delivery_done();

   /* clear the screen */
   position(1, 1);
   clear_screen();
   flush();
   }


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

Screen Output Routines
```

```
*********************************************************************
*********************************************************************/


number(row, col, n, width)
   int row;
   int col;
   int n;
   int width;
   {
   char str[81];
   fmt_num(str, n, width);
   text(row, col, str);
   }



real(row, col, x, width, dec)
   int row;
   int col;
   double x;
   int width;
   int dec;
   {
   char str[81];
   fmt_flt(str, x, width, dec);
   text(row, col, str);
   }

date(row, col, date_str)
   int row;
   int col;
   char *date_str;
   {
   text(row, col, date_str);
   }


date_only(row, col, date_str)
   int row;
   int col;
   char *date_str;
   {
   date_str[10] = '\0';
   text(row, col, date_str);
   }

money(row, col, x, width)
   int row;
   int col;
   double x;
   int width;
   {
   char str[81];
   fmt_money(str, x, width);
   text(row, col, str);
   }


long_text(row, col, str, width)
   int row, col, width;
   char *str;
   {
   int pos;

   /* repeat until the entire string is written out */
   for (pos = width; *str != '\0';  str++, pos++)
      {

      /* if at end of line, position the cursor to next line */
      if (pos >= width)
         {
         position(row, col);
         pos = 0;
         row++;
         }

      /* output the next character */
      pushc(*str);
      }
   }


text(row, col, str)
   int row;
   int col;
   char str[];
   {
   position(row, col);
   string(str);
   }


phone(row, col,str)
     int row;
     int col;
     char *str;
     {

      char temp[30];

      fmt_phone(temp,str);
      text(row,col,temp);

      }


zip(row, col, str)
   int row;
   int col;
   char *str;
   {
   char temp[30];

   fmt_zip(temp,str);
   text(row,col,temp);
   }

empty(row, col, len)
   int row;
   int col;
   int len;
   {
   position(row, col);
   while (len-- > 0)
      pushc('_');
   }


blanks(row, col, len)
   int row, col, len;
   {
   position(row, col);
   while (len-- > 0)
      pushc(' ');
   }


status(row, col, status)
/*********************************************************************
status displays the transaction status
  Note: must correspond to 'get_status' in driver/keystroke.c
*********************************************************************/
  int row, col;
  int status;
  {
  text(row, col, "Execution Status: ");

  if (status == OK)
     string("Transaction Committed");
  else if (status == E_INVALID_ITEM)
     string("Item number is not valid");
  /* Do the rev. 3.3 error checking here.       */
  else if (status == E_INVALID_INPUT)
     string("Invalid input, transaction not executed");
  else
     {
     string("Rollback -- ");
     number(row, col+30, status, 5);
     }
  }

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

ASCII terminal control

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


trigger()
/*********************************************************************
trigger sends a turnaround sequence to let the driver know to send input
*********************************************************************/
   {
   pushc(TRIGGER);
   }


position(row, col)
/*********************************************************************
position positions the cursor at the given row and column
*********************************************************************/
   int row;
   int col;
   {
   pushc(ESCAPE);
   pushc('[');
   if (row >= 10)
```

```
    pushc('0' + row/10);
  pushc('0'+ row%10);
  pushc(';');
  if (col >= 10)
    pushc('0' + col/10);
  pushc('0' + col%10);
  pushc('H');
  }

clear_screen()
/*************************************************************************
clear_screen clears the iobuf from cursor position to end of iobuf
*************************************************************************/
  {
  pushc(ESCAPE);
  pushc('[');
  pushc('J');
  }



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

Screen Input Routines

*************************************************************************
*************************************************************************/
#define funny(key)  (key != ENTER && key !=TAB && key != BACKTAB)


read_number(row, col, n, width)
/*************************************************************************
read_number reads an integer field
*************************************************************************/
  int row;
  int col;
  int *n;
  int width;
  {
  char temp[81];
  int key;
  int err;
  debug("read_number: row=%d  col=%d  width=%d n=%d \n",row, col,width,*n);

  /* generate the current characters */
  fmt_num(temp, *n, width);
  err = NO;

  /* repeat until a valid number or a funny key is pressed */
  for (;;)
    {
    /* Let the user edit the field */
    key = getfield(row, col, temp, width, Num);
        if (funny(key)) return key;

    /* convert the field to a number */
    *n = cvt_num(temp);
    if (*n != INVALID_NUM) break;

    msgline("Invalid digit entered");
        pushc(BELL);
    err = YES;
    }

  /* display the new number */
  number(row, col, *n, width);
  if (err) msgline("");
  debug("read_number: n=%d  key=%d\n", *n, key);
  return key;
  }


int read_money(row, col, m, width)
  int row;
  int col;
  double *m;
  int width;
  {
  char temp[81];
  int key;
  int err;

  err = NO;
  fmt_money(temp, *m, width);

  /* repeat until a valid number or a funny key is pressed */
  for (;;)
    {
    key = getfield(row, col, temp, width, Money);
        if (funny(key)) return key;

    *m = cvt_money(temp);
    if (*m != INVALID_FLT)  break;

    msgline("Please enter amount $99999.99");
    pushc(BELL);
    err = YES;
        }

  money(row, col, *m, width);
  if (err) msgline("");
  return key;
  }



int read_real(row, col, x, width, dec)
  int row, col, width;
  double *x;
  {
  char temp[81];
  int key;
  int err;

  /* generate the current characters */
  fmt_flt(temp, *x, width, dec);
  err = NO;

  /* repeat until a valid number or a funny key is pressed */
  for (;;)
        {
    key = getfield(row, col, temp, width);
        if (funny(key)) return key;

    /* convert the field to a number */
    *x = cvt_flt(temp);
        if (*x != INVALID_FLT) break;

    msgline("Please enter a valid floating pt number");
    pushc(BELL);
    err = YES;
    }

  /* display the new number */
  real(row, col, *x, width, dec);
  if (err) msgline("");

  return key;
  }



int read_text(row, col, s, width)
  int row, col, width;
  char *s;
  {
  char temp[81];
  int key;
  int i;

  /* generate the current characters */
  fmt_text(temp, s, width);

  /* let the user edit the field */
  key = getfield(row, col, temp, width, Text);
  if (funny(key)) return key;

  /* Strip off leading and trailing space characters */
  cvt_text(temp, s);

  /* redisplay the current text */
  fmt_text(temp, s, width);
  text(row, col, temp);

  return key;
  }



int getfield(row, col, buf, width,ftype)
  int row, col, width;
  char buf[];
  FIELD_TYPE ftype;

  {
  int pos, key;

  debug("getfield: width=%d  buf=%*s\n", width, width, buf);

  /* go to the beginning of the field */
  position(row, col);
  pos = 0;

  /* repeat until a special control character is pressed */
  for (;;)
    {
```

```
    /* get the next character */
    key = getkey();

    /* CASE: Add to buf if it fits and Is it a valid character ? */
    if (pos < width  && valid_char(key, ftype))
        {
        buf[pos] = key;
        pos++;
        pushc(key);
        }

    /* CASE: char is BACKSPACE.  Erase last character. */
    else if (key == BACKSPACE && pos > 0)
        {
        pos--;
        buf[pos] = '_';
        pushc(BACKSPACE);
        pushc('_');
        pushc(BACKSPACE);
        }

    /* CASE: enter, tab, backtab, ^c.  Exit loop */
    else if (key==ENTER || key==TAB || key==BACKTAB || key==CNTRLC
            || key == EOF)
        break;

    else if (key=='\031')   /* for debugging, let ^X == ENTER */
        {key=ENTER; break;}

    /* Otherwise, ignore the character and beep */
    else
        pushc(BELL);
    }

  debug("getfield: final key: %d  buf=%*s\n", key, width, buf);
  return key;
  }
```

```
int valid_char(key, ftype)
/*************************************************************************
valid_char is true if the key is valid for this type of field
*************************************************************************/
  int key;
  FIELD_TYPE ftype;
  {
  int valid;
  switch(ftype)
                {
                case Num : valid = (isdigit(key) || key == '-' || key == ' ');
                                break;

    case Text : valid = (isprint(key) || key == ' ');
                                break;

    case Money : valid = (isdigit(key) || key == '-' || key == '.'
                                                || key == '$' || key == ' ');
                        break;

    default     : valid = NO;
                                break;
    }

  return valid;
  }
```

client/tux_transaction.c

```
/*************************************************************************
 @(#) Version: A.10.10 $Date: 2001/08/29 16:24:31 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*************************************************************************/

#include <varargs.h>
#include <errno.h>


#include "tpcc.h"
#include "atmi.h"
#include "Uunix.h"

#ifdef USE_DRAND48
#define RandomNumber(min,max)  \
                ((int)(drand48() * ((int)(max) - (int)(min) + 1)) + (int)(min))
#else
#define RandomNumber(min,max)  \
                ((int)(randy() * ((int)(max) - (int)(min) + 1)) + (int)(min))
#endif

extern int userid;


neworder_trans *neworder_ptr;
payment_trans  *payment_ptr;
ordstat_trans *ordstat_ptr;
stocklev_trans *stocklev_ptr;
delivery_trans *delivery_ptr;

long result;

static tux_error(format, va_alist)
  char *format;
  va_dcl
  {
  va_list argptr;

  va_start(argptr);
  vmessage(format, argptr);

  message("Tuxedo error %d\n", tperrno);

  errno = Uunixerr;
  if (tperrno == TPEOS)
    syserror("Tuxedo encountered O/S error\n");

  if (tperrno == TPESVCERR || tperrno == TPETIME)      {
                message("Retrying transation\n");
                if (tperrno == TPETIME)
                  sleep (RandomNumber(1,30));
  }
  else
                error("EXITING !!!\n");

  }


transaction_begin(u)
  int u;
  {

  /* keep track of which user we are (for error messages only) */
  userid = u;

  /* attach to Tuxedo */
  if (tpinit( (TPINIT *)NULL) == -1)
    tux_error("Failed to attach to Tuxedo\n");

  /* allocate structures for each transaction */
  neworder_ptr = (neworder_trans *)tpalloc("CARRAY", NULL, sizeof(neworder_trans));
  payment_ptr = (payment_trans *)tpalloc("CARRAY", NULL, sizeof(payment_trans));
  ordstat_ptr = (ordstat_trans *)tpalloc("CARRAY", NULL, sizeof(ordstat_trans));
  stocklev_ptr = (stocklev_trans *)tpalloc("CARRAY", NULL, sizeof(stocklev_trans));
  delivery_ptr = (delivery_trans *)tpalloc("CARRAY", NULL, sizeof(delivery_trans));
  if (neworder_ptr == NULL || payment_ptr == NULL || ordstat_ptr == NULL
    || stocklev_ptr == NULL || delivery_ptr == NULL)
    tux_error("Unable to allocate Tuxedo memory \n");
  }



transaction_done()
  {
  if (tpterm() == -1)
    tux_error("Unable to detach from Tuxedo\n");
  }



void neworder_transaction(t)
  neworder_trans *t;
  {
  *neworder_ptr = *t;
  while (tpcall("NEWO_SVC", (char *)neworder_ptr, sizeof(neworder_trans),
        (char **)(&neworder_ptr), &result, TPSIGRSTRT|TPNOTIME) == -1) {
    tux_error("Tuxedo failed for neworder transaction\n");
                *neworder_ptr = *t;
                }
  *t = *neworder_ptr;
  }



void paym ent_transaction(t)
  payment_trans *t;
  {
  *payment_ptr = *t;
  while (tpcall("PMT_SVC", (char *)payment_ptr, sizeof(payment_trans),
                (char **)(&payment_ptr), &result, TPSIGRSTRT|TPNOTIME) == -1) {
    tux_error("Tuxedo failed for payment transaction\n");
                *payment_ptr = *t;
                }
  *t = *payment_ptr;
```

```
    }


void ordstat_transaction(t)
    ordstat_trans *t;
    {
    *ordstat_ptr = *t;
    while (tpcall("ORDS_SVC", (char *)ordstat_ptr, sizeof(ordstat_trans),
                    (char **)(&ordstat_ptr), &result, TPSIGRSTRT|TPNOTIME) == -1) {
                    tux_error("Tuxedo failed for ordstat transaction\n");
                    *ordstat_ptr = *t;
                    }
    *t = *ordstat_ptr;
    }



stocklev_transaction(t)
    stocklev_trans *t;
    {
    *stocklev_ptr = *t;
    while (tpcall("STKL_SVC", (char *)stocklev_ptr, sizeof(stocklev_trans),
                    (char **)(&stocklev_ptr), &result, TPSIGRSTRT|TPNOTIME) == -1) {
        tux_error("Tuxedo failed for stocklev transaction\n");
                    *stocklev_ptr = *t;
                    }
    *t = *stocklev_ptr;
    }


delivery_init(u)
    int u;
    {
    }


delivery_enque(t)
    delivery_trans *t;
    {
    gettimeofday(&t->enque[0], NULL);
    t->status = OK;

    *delivery_ptr = *t;
    while (tpacall("DVRY_SVC", (char *)delivery_ptr, sizeof(delivery_trans),
            TPNOREPLY) == -1) {
        tux_error("Tuxedo failed enqueing delivery transaction\n");
                    *delivery_ptr = *t;
                    }
    }


delivery_done()
    {
    }
```

client/Makefile

```
#*****************************************************************************
*
#@(#) Version: A.10.10 $Date: 98/02/03 08:13:40 $
#
#(c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
#*****************************************************************************
*

include ../buildenv.mk

#
# Makefile for compiling the client, batch-tpcc, and service code
#

OH      = ${ORACLE_HOME}
P       = ${WORK_DIR}/src
I       = $(P)/lib
L       = $(P)/lib
D       = $(P)/driver
Q       = $(P)/que
S       = $(P)/client

include env_rdbms.mk

SH_OPT     = -Wl,-a,shared
OPT        =  -Wl,-a,archive_shared
LDOPTS     = -ldld -a archive_shared

ORA_CFLAGS = -DORA_BLOCK_PATH=\"/project/tpcc/blocks/\" \
                    -DORA_NULL_DATE=\"01-01-1811\" \
                    -D_HPUX_SOURCE -DSS_64BIT_SERVER -DHPPA64 -DSLS8NATIVE \
                    -DSLU8NATIVE -DSLTS_ENABLE -DHPUX_KTHREAD -
DSLXMX_ENABLE \
                    -DSLMXMX_ENABLE

ORA_INCLUDE=-I. -I${S}/oracle \
   -I$(ORACLE_HOME)/rdbms/demo \
   -I$(ORACLE_HOME)/rdbms/public -I$(ORACLE_HOME)/rdbms/include \
   -I$(ORACLE_HOME)/plsql/public \
   -I$(ORACLE_HOME)/network/public
SYB_INCLUDE= -I${SYBASE}/include
VIS_INCLUDE= -I ${VISIGENIC}/include
TUX_INCLUDE= -I${ROOTDIR}/include
INCLUDE    = -I. -I$L

SH_CFLAGS  = $(BUILDFLAGS) ${SH_OPT} ${INCLUDE} ${TUX_INCLUDE}
CFLAGS          = $(BUILDFLAGS) ${OPT} ${INCLUDE} ${TUX_INCLUDE}
CFLAGS_SYB = $(BUILDFLAGS) ${OPT} ${INCLUDE} ${TUX_INCLUDE}
${SYB_INCLUDE}
CFLAGS_ORA = $(BUILDFLAGS) ${ORA_CFLAGS} ${OPT} ${INCLUDE}
${ORA_INCLUDE} \
                    ${TUX_INCLUDE}
CFLAGS_SQL = -Dunix -D_HPUX_SOURCE -DVG_UNIX ${OPT} ${INCLUDE} \
                    ${TUX_INCLUDE} ${SQL_INCLUDE} ${VIS_INCLUDE}



LDFLAGS_ORA= ${OPT} $(CFLAGS_ORA) ${L}/tpc_lib.a
LDFLAGS_SYB= ${OPT} ${L}/tpc_lib.a -L${SYBASE}/lib -lsybdb -lm
LDFLAGS_SQL= ${OPT} ${L}/tpc_lib.a -L/opt/odbc/lib -lodbc -lm

PROGRAMS    = client service startup client_batch msg_server raw

all: ${PROGRAMS}

all_ora: others_oracle service_oracle tpcc_client

tpcc_client: client
                    $(MV) client ${WORK_DIR}/bin/
others_sybase: raw startup client_batch_syb msg_server_syb
                    $(MV) raw startup client_batch msg_server ${WORK_DIR}/bin
others_oracle: raw startup client_batch_ora
                    $(MV) raw startup ${WORK_DIR}/bin
                    $(MV) client_batch_ora ${WORK_DIR}/bin/client_batch
others_sqlserver: raw startup client_batch_sql msg_server_sql
                    $(MV) raw startup client_batch msg_server ${WORK_DIR}/bin
service_oracle: service_ora
                    $(MV) service_ora ${WORK_DIR}/bin/service
service_sybase: service_syb
                    $(MV) service ${WORK_DIR}/bin/
service_sqlserver: service_sql
                    $(MV) service ${WORK_DIR}/bin/

${S}/oracle/transaction.o: ${S}/oracle/transaction.c
                    $(CC) ${CFLAGS_ORA} $(L)/tpc_lib.a -c ${S}/oracle/transaction.c;
${S}/sybase/transaction.o: ${S}/sybase/transaction.c
                    $(CC) ${CFLAGS_SYB} $(L)/tpc_lib.a -c ${S}/sybase/transaction.c;
${S}/sqlserver/transactionb.o: ${S}/sqlserver/transactionb.c
                    $(CC) ${CFLAGS_SQL} $(L)/tpc_lib.a -c ${S}/sqlserver/transactionb.c;

ORA_OBJS=plnew.o plord.o plpay.o pldel.o plsto.o tpccpl.o
plnew.o: ${S}/oracle/plnew.c
                    $(CC) ${CFLAGS_ORA} $(L)/tpc_lib.a -c ${S}/oracle/plnew.c;
plord.o: ${S}/oracle/plord.c
                    $(CC) ${CFLAGS_ORA} $(L)/tpc_lib.a -c ${S}/oracle/plord.c;
plpay.o: ${S}/oracle/plpay.c
                    $(CC) ${CFLAGS_ORA} $(L)/tpc_lib.a -c ${S}/oracle/plpay.c;
pldel.o: ${S}/oracle/pldel.c
                    $(CC) ${CFLAGS_ORA} $(L)/tpc_lib.a -c ${S}/oracle/pldel.c;
plsto.o: ${S}/oracle/plsto.c
                    $(CC) ${CFLAGS_ORA} $(L)/tpc_lib.a -c ${S}/oracle/plsto.c;
tpccpl.o: ${S}/oracle/tpccpl.c
                    $(CC) ${CFLAGS_ORA} $(L)/tpc_lib.a -c ${S}/oracle/tpccpl.c;

raw: raw.o
                    cc ${CFLAGS} raw.o $(L)/tpc_lib.a -o raw

startup: startup.o $(L)/tpc_lib.a
                    cc ${SH_CFLAGS} startup.o $(L)/tpc_lib.a -o startup
                    chmod a+rw startup

# Warning: can't use +pd because kernel will use this size to extend
# DATA pregion when break/sbreak is called.
# Force shared libc to avoid to both libc data from the archived and shared version.
client: client.o  tux_transaction.o $(L)/tpc_lib.a
                    ${ROOTDIR}/bin/buildclient -v -o client \
        -f "$(BUILDFLAGS) $(OPT) -Wl,+pi 256K -Wl,-R 0x100000 \
            client.o tux_transaction.o $(L)/tpc_lib.a" \
                    -l "-lnsl  -lm -Wl,-ashared -lc"

service_ora: service.o ${S}/oracle/transaction.o $(ORA_OBJS) $(L)/tpc_lib.a
                    if [ -f /project/iti/lib/libgp.a ] ; then \
            mv /project/iti/lib/libgp.a /project/iti/lib/libgp.a.cc_service ; \
        fi
                    ${ROOTDIR}/bin/buildserver -v -b shm \
                -s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s DVRY_SVC \
```

```
	-o service_ora \
	-f "$(BUILDFLAGS) $(OPT) \
	   -Wl,+pi 4M -Wl,+pd 256K \
	   -Wl,-R 0x400000 -Wl,-D 0x40100000 \
	                service.o transaction.o $(ORA_OBJS) $L/tpc_lib.a \
	          ${LDFLAGS_ORA} $(LDFLAGS32) $(SSABED) $(DEF_OPT)
$(TTLIBS)" \
	-l "-lnsl "
	    if [ -f /project/iti/lib/libgp.a.cc_service ] ; then \
	       cp /project/iti/lib/libgp.a.cc_service /project/iti/lib/libgp.a ; \
	    fi

service_syb: service.o ${S}/sybase/transaction.o $(L)/tpc_lib.a
	     ${ROOTDIR}/bin/buildserver -v -b shm \
	-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s DVRY_SVC \
	-o service \
	-f "service.o transaction.o $L/tpc_lib.a \
	   ${SYBASE}/lib/libsybdb.a -lm";

service_sql: service.o ${S}/sqlserver/transactionb.o $(L)/tpc_lib.a
	     ${ROOTDIR}/bin/buildserver -v -b shm \
	-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s DVRY_SVC \
	-o service \
	-f "service.o transactionb.o $L/tpc_lib.a \
	  /vsbuild/v1.10/build/com/obj/inst/libodbc.sl"

client_batch_ora: $(D)/driver.o $(D)/generate.o ${S}/oracle/transaction.o \
	          $(ORA_OBJS) $(Q)/dummy_que.o $(L)/tpc_lib.a  \
	          $(L)/server_default.o
	    $(CC) $(BUILDFLAGS) $(OPT) \
	$(D)/driver.o $(D)/generate.o transaction.o $(ORA_OBJS) \
	     $(Q)/dummy_que.o $(L)/server_default.o $(L)/tpc_lib.a ${LDFLAGS_ORA} \
	     $(LDFLAGS32) $(SSABED) $(DEF_OPT) $(TTLIBS) -o client_batch_ora;

client_batch_syb: $(D)/driver.o $(D)/generate.o ${S}/sybase/transaction.o \
	          $(Q)/dummy_que.o $(L)/tpc_lib.a $(L)/server_default.o
	    $(CC) $(D)/driver.o $(D)/generate.o transaction.o \
	     $(Q)/dummy_que.o $(L)/server_default.o $(L)/tpc_lib.a \
	     ${LDFLAGS_SYB} -o client_batch;

client_batch_sql: $(D)/driver.o $(D)/generate.o ${S}/sqlserver/transactionb.o \
	          $(Q)/dummy_que.o $(L)/tpc_lib.a $(L)/server_default.o
	                $(CC) $(D)/driver.o $(D)/generate.o transactionb.o \
	          $(Q)/dummy_que.o $(L)/server_default.o $(L)/tpc_lib.a \
	          ${LDFLAGS_SQL} -o client_batch;

msg_server_ora: $(Q)/msg_server.o ${S}/oracle/transaction.o $(ORA_OBJS) \
	         $(L)/tpc_lib.a
	    $(CC) $(Q)/msg_server.o transaction.o $(ORA_OBJS) ${LDFLAGS_ORA} \
	     $(LDFLAGS32) $(SSABED) $(DEF_OPT) $(TTLIBS) -o msg_server;
msg_server_syb: $(Q)/msg_server.o  ${S}/sybase/transaction.o $(L)/tpc_lib.a
	    $(CC) $(Q)/msg_server.o transaction.o ${LDFLAGS_SYB} -o msg_server;

msg_server_sql: $(Q)/msg_server.o  ${S}/sqlserver/transactionb.o $(L)/tpc_lib.a
	    $(CC) $(Q)/msg_server.o transactionb.o ${LDFLAGS_SQL} -o msg_server;

clean:
	    rm -f *.o

clobber: clean
	    rm -f ${PROGRAMS}

install: ${PROGRAMS}
	    cp ${PROGRAMS} ${WORK_DIR}/bin
```

A.2 Tpc_lib Source

lib/tpcc.h

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 14:01:49 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/
#ifndef TPCC_INCLUDED
#define TPCC_INCLUDED
#include <values.h>


/* The auditor can define these 20 char strings to be anything */
#define DRIVER_AUDIT_STRING  "driver audit string"
#define CLIENT_AUDIT_STRING  "client audit string"

#ifdef DEBUG
#define debug printf
#else
#define debug (void)
#endif
```

```
#include <stdio.h>

typedef int ID;          /* All id's */
typedef double MONEY;    /* Large integer number of cents */
typedef char TEXT;       /* Add an extra byte for null terminator */
typedef double TIME;     /* Elapsed seconds from start of run (float?) */
typedef int COUNT;       /* integer numbers of things */
typedef double REAL;     /* real numbers */
typedef int LOGICAL;     /* YES or NO */
typedef struct {         /* days and seconds since Jan 1, 1900 */
   int day;            /* NULL represented by negative day */
   int sec;
   } DATE;

/* Macro to convert time of day to TIME */
#include <time.h>
extern struct timeval start_time;
#define elapsed_time(t) ( ((t)->tv_sec - start_time.tv_sec) +  \
                  ((t)->tv_usec - start_time.tv_usec) / 1000000.0 )

typedef enum {Num,Money,Text,Time,Real,Date} FIELD_TYPE;   /* screen field types */


/* Various TPCC constants */
#define W_ID_LEN       4
#define D_ID_LEN       2
#define C_ID_LEN       4
#define I_ID_LEN       6
#define OL_QTY_LEN     2
#define PMT_LEN        7
#define C_ID_LEN       4
#define C_LAST_LEN     16
#define CARRIER_LEN    2
#define THRESHOLD_LEN  2
#define DIST_PER_WARE  10
#define CUST_PER_DIST  3000
#define ORD_PER_DIST   3000
#define MAXITEMS       100000
#define MAX_DIGITS     3      /* # of digits of the NURand number selected
                                 to generate the customer last name */
#define MAXWAREHOUSE   2000   /* maximum # of warehouses - scaling factor */
#define LOADSEED       42 /* # of digits of the NURand number selected

/*************************************************************************/
/* database identifiers and populations                                  */
/*************************************************************************/

  int no_warehouse;                  /*  scaling factor    */
  int no_item;                      /*  100000            */
  int no_dist_pw;                    /*    10              */
  int no_cust_pd;                    /*   3000             */
  int no_ord_pd;                     /*   3000             */
  int no_new_pd;                     /*    900             */
  int tpcc_load_seed;                /*    900             */

/* fields to add to each transaction for acid testing */
#define ACID_STUFF     \
   char acid_txn[2]; \
    int  acid_timing; \
    int  acid_action; \
    FILE *acid_res

typedef struct {
   ID OL_SUPPLY_W_ID;
   ID OL_I_ID;
   TEXT I_NAME[24+1];
   COUNT OL_QUANTITY;
   COUNT S_QUANTITY;
   MONEY I_PRICE;
   char brand_generic;
   } neworder_item;

typedef struct {
   int status;
   LOGICAL all_local;
   ID W_ID;
   ID D_ID;
   ID C_ID;
   TEXT C_LAST[C_LAST_LEN+1];
   TEXT C_CREDIT[2+1];
   REAL C_DISCOUNT;
   COUNT O_OL_CNT;
   ID O_ID;
   TEXT O_ENTRY_D[20];  /* dates as text fields */
   REAL W_TAX;
   REAL D_TAX;
   neworder_item item[15];
   ACID_STUFF;
   } neworder_trans;



typedef struct {
   int status;
   LOGICAL byname;
```

```
    ID W_ID;
    ID D_ID;
    ID C_ID;
    ID C_D_ID;
    ID C_W_ID;
    MONEY H_AMOUNT;
    TEXT H_DATE[20];  /* date as text field */
    TEXT W_STREET_1[20+1];
    TEXT W_STREET_2[20+1];
    TEXT W_CITY[20+1];
    TEXT W_STATE[2+1];
    TEXT W_ZIP[9+1];
    TEXT D_STREET_1[20+1];
    TEXT D_STREET_2[20+1];
    TEXT D_CITY[20+1];
    TEXT D_STATE[2+1];
    TEXT D_ZIP[9+1];
    TEXT C_FIRST[16+1];
    TEXT C_MIDDLE[2+1];
    TEXT C_LAST[16+1];
    TEXT C_STREET_1[20+1];
    TEXT C_STREET_2[20+1];
    TEXT C_CITY[20+1];
    TEXT C_STATE[2+1];
    TEXT C_ZIP[9+1];
    TEXT C_PHONE[16+1];
    TEXT C_SINCE[20]; /* date as text field */
    TEXT C_CREDIT[2+1];
    MONEY C_CREDIT_LIM;
    REAL C_DISCOUNT;
    REAL C_BALANCE;
    TEXT  C_DATA[200+1];
    ACID_STUFF;
    } payment_trans;


typedef struct {
    int status;
    LOGICAL byname;
    ID W_ID;
    ID D_ID;
    ID C_ID;
    TEXT C_FIRST[16+1];
    TEXT C_MIDDLE[2+1];
    TEXT C_LAST[16+1];
    MONEY C_BALANCE;
    ID O_ID;
    TEXT O_ENTRY_DATE[20];   /* date as text field */
    ID O_CARRIER_ID;
    COUNT ol_cnt;
    struct {
        ID OL_SUPPLY_W_ID;
        ID OL_I_ID;
        COUNT OL_QUANTITY;
        MONEY OL_AMOUNT;
        TEXT OL_DELIVERY_DATE[20]; /* date as text field */
        } item[15];
    ACID_STUFF;
    } ordstat_trans;


typedef struct {
    int status;
    ID W_ID;
    ID D_ID;
    COUNT threshold;
    COUNT low_stock;
    ACID_STUFF;
    } stocklev_trans;

typedef struct {
    int status;
    ID W_ID;
    ID O_CARRIER_ID;
    struct {
        ID O_ID;
                int status;
        } order[10];
    struct timeval enque[1];
    struct timeval deque[1];
    struct timeval complete[1];
    ACID_STUFF;
    } delivery_trans;


typedef union {
    neworder_trans neworder;
    payment_trans payment;
    ordstat_trans ordstat;
    delivery_trans delivery;
    stocklev_trans stocklev;
    int status;
    } generic_trans;
```

```
/*************************************************
Record formats for results
*************************************************/

#ifdef NOTYET
typedef struct
    {
    float t1, t2, t3, t4, t5;
    int status          :8;
    unsigned int type    :3;
    unsigned int ol_cnt  :4;
    unsigned int remote_ol_cnt :4;
    unsigned int byname :1;
    unsigned int remote  :1;
    unsigned int skipped  :4;
    } success_t;
#endif

typedef struct
    {
    TIME t1, t2, t3, t4, t5;
    int status;
    unsigned int type    :3;
    unsigned int ol_cnt  :4;
    unsigned int remote_ol_cnt :4;
    unsigned int byname :1;
    unsigned int remote  :1;
    unsigned int skipped  :4;
    } success_t;

typedef struct
    {
    struct timeval start_time;
    } success_header_t;


/*************************************************************************
Record formats for loading routines.  (DB's have own internal formats
*************************************************************************/
typedef struct
    {
    ID W_ID;
    TEXT W_NAME[10+1];
    TEXT W_STREET_1[20+1];
    TEXT W_STREET_2[20+1];
    TEXT W_CITY[20+1];
    TEXT W_STATE[2+1];
    TEXT W_ZIP[9+1];
    REAL W_TAX;
    MONEY W_YTD;
    } warehouse_row;

typedef struct
    {
    ID D_ID;
    ID D_W_ID;
    TEXT D_NAME[10+1];
    TEXT D_STREET_1[20+1];
    TEXT D_STREET_2[20+1];
    TEXT D_CITY[20+1];
    TEXT D_STATE[2+1];
    TEXT D_ZIP[9+1];
    REAL D_TAX;
    MONEY D_YTD;
    ID D_NEXT_O_ID;
    } district_row;


typedef struct
    {
    ID C_ID;
    ID C_D_ID;
    ID C_W_ID;
    TEXT C_FIRST[16+1];
    TEXT C_MIDDLE[2+1];
    TEXT C_LAST[16+1];
    TEXT C_STREET_1[20+1];
    TEXT C_STREET_2[20+1];
    TEXT C_CITY[20+1];
    TEXT C_STATE[2+1];
    TEXT C_ZIP[9+1];
    TEXT C_PHONE[16+1];
    DATE C_SINCE;
    TEXT C_CREDIT[2+1];
    MONEY C_CREDIT_LIM;
    REAL C_DISCOUNT;
    MONEY C_BALANCE;
    MONEY C_YTD_PAYMENT;
    COUNT C_PAYMENT_CNT;
    COUNT C_DELIVERY_CNT;
    TEXT  C_DATA[500+1];
    } customer_row;

typedef struct
    {
```

```
    ID H_C_ID;
    ID H_C_D_ID;
    ID H_C_W_ID;
    ID H_D_ID;
    ID H_W_ID;
    DATE H_DATE;
    MONEY H_AMOUNT;
    TEXT H_DATA[24+1];
    } history_row;

typedef struct
    {
    ID NO_O_ID;
    ID NO_D_ID;
    ID NO_W_ID;
    } neworder_row;

typedef struct
    {
    ID O_ID;
    ID O_D_ID;
    ID O_W_ID;
    ID O_C_ID;
    DATE O_ENTRY_D;
    ID O_CARRIER_ID;
    COUNT O_OL_CNT;
    LOGICAL O_ALL_LOCAL;
    } order_row;

typedef struct
    {
    ID OL_O_ID;
    ID OL_D_ID;
    ID OL_W_ID;
    ID OL_NUMBER;
    ID OL_I_ID;
    ID OL_SUPPLY_W_ID;
    DATE OL_DELIVERY_D;
    COUNT OL_QUANTITY;
    MONEY OL_AMOUNT;
    TEXT OL_DIST_INFO[24+1];
    } orderline_row;

typedef struct
    {
    ID I_ID;
    ID I_IM_ID;
    TEXT I_NAME[24+1];
    MONEY I_PRICE;
    TEXT I_DATA[50+1];
    } item_row;

typedef struct
    {
    ID S_I_ID;
    ID S_W_ID;
    COUNT S_QUANTITY;
    TEXT S_DIST_01[24+1];
    TEXT S_DIST_02[24+1];
    TEXT S_DIST_03[24+1];
    TEXT S_DIST_04[24+1];
    TEXT S_DIST_05[24+1];
    TEXT S_DIST_06[24+1];
    TEXT S_DIST_07[24+1];
    TEXT S_DIST_08[24+1];
    TEXT S_DIST_09[24+1];
    TEXT S_DIST_10[24+1];
    COUNT S_YTD;
    COUNT S_ORDER_CNT;
    COUNT S_REMOTE_CNT;
    TEXT S_DATA[50+1];
    } stock_row;


/* Empty field values */
#define EMPTY_NUM (MAXINT-1)
#define INVALID_NUM (MAXINT)
#define EMPTY_FLT   (MAXDOUBLE)
#define INVALID_FLT (MINDOUBLE)

/* Status conditions */
#define OK 0
#define E 1
#define E_INVALID_ITEM 2
#define E_NOT_ENOUGH_ORDERS 3
#define E_DB_ERROR 4
#define E_INVALID_INPUT 5

/* Error message strings */
static char *e_mesg[]={"Transaction complete.","Error","Invalid item number.",
                       "Not enough orders.", "Database ERROR !!!!"};



#define YES 1
#define NO 0



double cvt_flt();
double cvt_money();
TIME getclock();
TIME getlocalclock();

#define TPC_MSG_QUE  150


/***************************************
Transaction specific stuff
***************************************/

/* types of transactions */
#define NEWORDER 1
#define PAYMENT  2
#define ORDSTAT  3
#define DELIVERY 4
#define STOCKLEV 5
#define DEFERRED 6   /* deferred portion of delivery */

/* the name of each transaction */
static char *transaction_name[] =
    {"", "New_Order", "Payment", "Order-Status",
                    "Delivery", "Stock-Level", "Deferred-Delivery"};

/* size of each transaction record */
static int transaction_size[] = {0,
     sizeof(neworder_trans),
     sizeof(payment_trans),
     sizeof(ordstat_trans),
     sizeof(delivery_trans),
     sizeof(stocklev_trans),
     sizeof(delivery_trans),
     0};

/* valid response time for each transaction */
static TIME valid_response[] = {0, 5, 5, 5, 5, 20};


#endif /* TPCC_INCLUDED */
```

lib/date.c

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 10:56:52 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/
#include "tpcc.h"
#include <time.h>

/* macro to get starting day of a particular year (1901 thru 2100) */
#define YEAR(yr)  ( (yr-1900)*365 + (yr-1900-1)/4 )

CurrentDate(date)
/*************************************************************************
CurrentDate fetches the current date and time
*************************************************************************/
    DATE *date;
    {
    struct timeval time;
    struct timezone tz;

    /* get the current time of day */
    if (gettimeofday(&time, &tz) < 0)
                syserror("Can't get time of day \n");

    /* adjust the time of day by the timezone */
    time.tv_sec -= tz.tz_minuteswest * 60;

    /* convert seconds and days since EPOCH (Jan 1, 1970) */
    date->day = time.tv_sec / (24*60*60);
    date->sec = time.tv_sec - date->day * (24*60*60);

    /* convert to days since Jan 1, 1900 */
    date->day += YEAR(1970);
    }


EmptyDate(date)
/*******************************************************
Get a NULL date and time
*******************************************************/
    DATE *date;
    {
    date->day = 0;   /* Use EMPTYNUM instead */
    date->sec = 0;
```

```
    }

int IsEmptyDate(date)
    DATE *date;
    {
    return (date->day == 0 & date ->sec == 0);
    }


#define Feb29 (31+29-1)

fmt_date(str, date)
/*************************************************************************
fmt_date formats the DATE into a string  MM-DD-YY HH-MM-SS
*************************************************************************/
    char str[20];
    DATE *date;
    {
    /* Note: should probably do date and time separately */

    int quad, year, month, day;
    int hour, minute, sec;

    static int dur[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    static int first = YES;

    day = date ->day;
    sec = date->sec;

    /* if NULL date, then return empty string */
    if (day == EMPTY_NUM || sec == EMPTY_NUM)
                {str[0] = ' \0';  return;}

    /* 2100, 1900 are NOT leap years.  If we are Feb 29 or later, add a day */
    if (day >= Feb29 + YEAR(2100))  day++;
    if (day >=  Feb29)              day++;

    /* figure out which quad and day within quad we are in */
    quad = day / (4*365+1);
    day = day- quad * (4*365+1);

    /* get our year within quad and day within the year */
    if      (day < 1*365+1)    {year = 0;}
    else if (day < 2*365+1)    {year = 1; day -= 1*365+1;}
    else if (day < 3*365+1)    {year = 2; day -= 2*365+1;}
    else                       {year = 3; day -= 3*365+1;}

    /* if this is a leap year, february has 29 days */
    if (year == 0)          dur[1] = 29;
    else                    dur[1] = 28;

    /* decide which day and month we are */
    for (month = 0; day >= dur[month]; month++)
        day -= dur[month];

    /* decide what time of day it is */
    minute = sec / 60;
    sec = se c - minute * 60;
    hour = minute / 60;
    minute = minute - hour * 60;

    /* format the date and time */
    fmtint(str+0,  day+1, 2, ' ');
    str[2]='-';
    fmtint(str+3, month+1, 2, '0');
    str[5]='-';
    fmtint(str+6, 1900+quad*4+year, 4, '0');
    str[10] = ' ';
    fmtint(str+11, hour, 2, ' ');
    str[13] = ':';
    fmtint(str+14, minute, 2, '0');
    str[16] = ':';
    fmtint(str+17, sec, 2, '0');
    str[19] = ' \0';
    }
```

lib/errlog.c

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 10:56:52 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/
#include <stdio.h>
#include <varargs.h>
#include <unistd.h>
#include <errno.h>
#include <stdlib.h>
#include <fcntl.h>

int userid;
int msgfile_fd = -10000;

#define MSG_BUF_SIZE 3*1024

static msg_buf();

error(format, va_alist)
/*************************************************************************
error formats a message and outputs it to a standard location (stderr for now)
*************************************************************************/
    char *format;
                va_dcl
    {
    va_list argptr;

    msg_buf("error \n", strlen("error \n"));

    /* point to the list of arguments */
    va_start(argptr);

    /* format and print to stderr */
    vmessage(format, argptr);

    /* done */
    va_end(argptr);

    /* take an error exit */
    exit(1);
    }


syserror( format, va_alist )
/*************************************************************************
syserror logs a message with the system error code
*************************************************************************/
    char *format;
                va_dcl
    {
    va_list argptr;
    int save_errno = errno;

    msg_buf("syserror \n", strlen("syserror \n"));
    /* point to the list of arguments */
    va_start(argptr);

    /* format and print to stderr */
    vmessage(format, argptr);

    /* done */
    va_end(argptr);

    /* display the system error message */
    message("    System error message: %d %s\n", save_errno, strerror(save_errno));

    /* take an error exit */
    exit(1);
    }



message(format, va_alist)
/*************************************************************************
message formats a message and outputs it to a standard location (stderr for now)
*************************************************************************/
    char *format;
                va_dcl
{
    va_list argptr;

    msg_buf("message \n", strlen("message \n"));
    /* point to the list of arguments */
    va_start(argptr);

    /* format and print to stderr */
    vmessage(format, argptr);

    /* done */
    va_end(argptr);
    }



vmessage(format, argptr)
/*************************************************

*************************************************/
    char *format;
    va_list argptr;
{
  char buf[MSG_BUF_SIZE];

  /* format a message id */
  sprintf(buf, "Host %-8s User %-6d Pid %-6d ", getenv("HOST_NAME"), userid, getpid());
```

```
  /* format the string and print it */
  vsprintf(buf+strlen(buf), format, argptr);
  if (getenv("NO_ERROR_LOG") == NULL)
      msg_buf(buf, strlen(buf));
  if (getenv("NO_STDERR") == NULL)
      write(2, buf, strlen(buf));
}



static msg_buf(buf, size)
  char *buf;
  int size;
  {
  char  *fname;
  time_t tepoch = time(NULL);
  char writebuf[MSG_BUF_SIZE+66];
  int ltimestamp;

  ltimestamp = strftime(writebuf, 64, "%m/%d %T ", localtime(&tepoch));

  /* get the file name to use */
  fname = getenv("ERROR_LOG");
  if (fname == NULL)
    fname = "/tmp/ERROR_LOG";

  /* get exclusive access to the error log file */
  if (msgfile_fd == -10000)    {
      msgfile_fd= open(fname, O_WRONLY | O_CREAT | O_APPEND, 0666);
      if (msgfile_fd < 0)
                  console_error("Can't open tpc error log file 'ERROR_LOG' \n");
  }
  strncpy(writebuf+ltimestamp, buf, size);
  write(msgfile_fd, writebuf, ltimestamp + size);
  }



console_error(str)
  char *str;
  {
  int fd = open("/dev/tty", O_WRONLY);
  write(fd, str, strlen(str));
  close(fd);
  exit(1);
  }
```

lib/fmt.c

```
/**************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 10:56:52 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
**************************************************************************/
#include "tpcc.h"
#include "iobuf.h"
#include <math.h>    /* needed for ceil (VM) */
#include <strings.h>

/* formatting routines.  */

/* Note: Currently use integer routines to format and convert.  Need to
   modify the code for cases when integers don't work.  */

fmt_money(str, m, width)
    char *str;
    MONEY m;
    int width;
    {

    if (m == EMPTY_FLT)
        {
        memset(str, '_', width);
        str[width] = '\0';
        return;
        }

    /* format it as a number with a leading blank */
    *str = ' ';
    fmt_flt(str+1, m/100, width-1, 2);

    /* fill in a leading dollar */
    while (*(str+1) == ' ')
        str++;
    *str = '$';
    }


double cvt_money(str)
    char *str;
    {
    char temp[81], *t, *s;
    double cvt_flt(), f;

    /* skip leading and trailing blanks */
    cvt_text(str, temp);

    /* remove leading $ */
    if (*temp == '$')   t = temp + 1;
    else                t = temp;

    /* start scan at current character */
    s = t;

    /* allow leading minus sign */
    if (*s == '-')
                s++;

    /* allow leading digits */
    while (isdigit(*s))
                s++;

    /* allow decimal pt and two decimal digits */
    if (*s == '.')    s++;
    if (isdigit(*s))  s++;
    if (isdigit(*s))  s++;

    /* There should be no more characters */
    if (*s != ' \0') return INVALID_FLT;

    /* convert the floating pt number */
    f = cvt_flt(t);
    if (f == EMPTY_FLT)        return EMPTY_FLT;
    else if (f == INVALID_FLT) return INVALID_FLT;
    else                       return rint(f*100);
    }


fmt_num(str, n, width)
    char str[];
    int n;
    int width;
    {
    /* mark the end of the string */
    str[width] = '\0';

    /* if empty number, return the empty field */
    if (n == EMPTY_NUM)
        memset(str, '_', width);

    /* otherwise, convert the integer */
    else
        fmtint(str, n, width, ' ');

    debug("fmt_num:  n=%d  str=%s\n", n, str);
    }



cvt_num(str)
    char str[];
    {
    char text[81];
    cvt_text(str, text);
    if (*text == ' \0')
        return EMPTY_NUM;
    else
        return cvtint(text);
    }



fmt_flt(str, x, width, dec)
/**************************************************************************
fmt_flt converts a floating pt number to a string   "999999.9999"
**************************************************************************/
    char *str;
    double x;
    int width;
    int dec;
    {
    int negative;
    int integer, fract;
    double absolute;

    static double pow10[] =
    {1., 10., 100., 1000., 10000., 100000., 1000000., 10000000., 100000000.};

    /* mark the end of string */
    str[width] = '\0';

    /* if empty value, make it be an empty field */
    if (x == EMPTY_FLT)
        {
        memset(str, '_', width);
        return;
```

```
    }

  absolute = (x < 0)? -x: x;

  /* separate into integer and fractional parts */
  integer = (int) absolute;
  fract   = (absolute - integer) * pow10[dec] + .5;

  /* let the integer portion contain the sign */
  if (x < 0) integer = -integer;

  /* Format integer and fraction separately */
  fmtint(str, integer, width-dec-1, ' ');
  str[width-dec-1] = '.';
  fmtint(str+width-dec, fract, dec, '0');
  }



double cvt_flt(str)
  char str[];
  {
  char text[81];
  char *t;
  double value;
  int div;
  int fract;
  int negative;
  int i;

  /* normalize the text */
  cvt_text(str, text);
  if (*text == ' \0')
    return EMPTY_FLT;

  negative = NO;
  fract = NO;
  value = 0;
  div = 1.0;

  negative = (text[0] == ' -');
  if (negative)    t = text+1;
  else             t = text;

  for (;  *t !=  ' \0' ; t++)
    {

    if (*t == '.')
                  if (fract)  return INVALID_FLT;
      else        fract = YES;

    else if (isdigit(*t))
      {
      value = value*10 + (int)*t - (int)'0';
      if (fract) div *= 10;
      }

    else
      return INVALID_FLT;
    }

  if (fract)
    value /= div;

  if (negative)
    value = -value;

  return value;
  }



fmt_text(s, text, width)
  char *s, *text;
  int width;
  {

  /* if an empty string, then all underscores */
  if (*text == ' \0')
    for (; width > 0; width--)
      *s++  = '_';

  /* otherwise, blank fill it */
  else
    {

    /* copy the text into the new buffer */
    for ( ; *text != '\0';  width–)
      *s++ = *text++;

    /* fill in the rest with blanks */
    for (;  width > 0;  width--)
      *s++ = ' ';
    }

  /* and finally, te rminate the string */
  *s = '\0';
  }



cvt_text(s, text)
  char *s;
  char *text;
  {
  char *lastnb;

  /* skip leading blanks and underscores */
  for (; *s == ' ' || *s == '_'; s++)
    ;

  /* copy the characters, keeping track of last blank or underscore */
  lastnb = text-1;
  for (; *s != ' \0'; *text++ = *s++)
    if (*s != ' ' && *s != '_')
      lastnb = text;

  /* truncate the text string to last nonblank character */
  *(lastnb+1) = '\0';
  }



fmtint(field, value, size, fill)
/*************************************************************************
fmtint formats an integer value into a character field to make the integer
right-justified within the character field, padded with leading fill
characters (e.g. leading blanks if a blank is passed in for the fill argument
*************************************************************************/
  int value;
  char *field;
  int size;
  char fill;
  {
  int negative;
  int dividend;
  int remainder;
  char *p;

  /* create characters from right to left */
  p = field + size - 1;

  /* make note if this is a negative number */
  negative = value < 0;
  if (negative)
    value = -value;

  /* Case: Null field.  Can't do anything */
  if (p < field)
    ;

  /* Case: value is zero.  Print a leading '0' */
  else if (value == 0)
    *p-- = '0';

  /* Otherwise, convert each digit in turn */
  else do
    {

    dividend = value / 10;
    remainder = value - dividend * 10;
    value = dividend;

    *p-- = (char) (  (int)'0'  +  remainder  );

    } while (p >= field && value > 0);

  /* insert a minus sign if appropriate */
  if (negative && p >= field)
    *p-- = '-';

  /* fill in leading characters */
  while (p >= field)
    *p-- = fill;
  }


int cvtint(str)
/*********************************************************************
getint extracts an integer value from the given character field
(ex: turns the string "123" into the integer 123)
*********************************************************************/
  char *str;
  {
  int value;
  char c;
  int negative;
  debug("cvtint: str=%s\n", str);
```

```
    negative = (*str == '-');
    if (negative)  str++;

    /* convert the integer */
    for (value = 0; isdigit(*str); str++)
        value = value*10 + (int)(*str) - (int)'0';

    /* if any non-digit characters, error */
    if (*str != ' \0')
                    return INVALID_NUM;

    /* make negative if there was a minus sign */
    if (negative)
        value = -value;

    debug("cvtint: value=%d\n", value);
    return value;
    }



fmt_phone(str, phone)
    char str[20];
    char *phone;

    {
    /* copy phone number and insert dashes  999999-999-999-9999 */
    str[0] = phone[0];  str[1] = phone[1]; str[2] = phone[2];
    str[3] = phone[3];  str[4] = phone[4]; str[5] = phone[5];
    str[6] = ' -';
    str[7] = phone[6];  str[8] = phone[7]; str[9] = phone[8];
    str[10] = ' -';
    str[11] = phone[9]; str[12] = phone[10]; str[13] = phone[11];
    str[14] = ' -';
    str[15] = phone[12]; str[16] = phone[13]; str[17] = phone[14];
    str[18] = phone[15];
    str[19] = ' \0';
    }


fmt_zip(str,zip)
    char str[20];
    char *zip;
    {
    /* copy zip code and insert dashes 99999-9999 */
    str[0] = zip[0];  str[1] = zip[1];  str[2] = zip[2];
    str[3] = zip[3];  str[4] = zip[4];
    str[5] = ' -';
    str[6] = zip[5]; str[7] = zip[6]; str[8] = zip[7];  str[9] = zip[8];
    str[10] = ' \0';
    }
```

lib/iobuf.h

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 10:56:52 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/

/*********************************************************************
History
941220 LAN Added definition and initialization of the line_col[] array.
       This was needed for modifications made of client program to do
       block I/O using a WYSE terminal.
*********************************************************************/

/* structure for screen emulation */
typedef struct
    {
    int row;
    int col;
    char buf[25][81];
    } screen_t;


typedef struct {
    char *beg;
    char *end;   /* for output buffers */
    char *max;
    char *cur;   /* for input buffers */
    } iobuf;

/* Macro do define an I/O buffer of x characters, initialized to empty */
#define define_iobuf(name, size)                    \
char name##_data[size];                              \
iobuf name[1] = {{name##_data, name##_data,      \
              name##_data+size, name##_data}}

#define reset(buf)  if (1) { \
    (buf)->cur = (buf)->end = (buf)->beg;               \
    *(buf)->beg = '\0';                                 \
    } else (void)0

#define flush()  if(1) { \
        display(out_buf);    \
        reset(out_buf);     \
     } else (void)0


/* Standard I/O to and from in_buf and out_buf */
#ifdef DECLARE_IO_BUFFERS
define_iobuf(output_stuff, 4*1024);
define_iobuf(input_stuff, 1024);
iobuf *in_buf = input_stuff;
iobuf *out_buf = output_stuff;
#else
iobuf *in_buf;
iobuf *out_buf;
#endif

#define pushc(c) if (1) { \
    if (out_buf->end >= out_buf->max)   \
        error("out_buf overflow:  beg=0x%x end=%d  max=%d\n",  \
          out_buf->beg, out_buf->end-out_buf->beg,out_buf->max-out_buf->beg); \
    *(out_buf->end++) = (c);  \
    *(out_buf->end) = '\0';  /* debug */ \
    } else (void)0

#define popc()  \
    (*in_buf->cur++)


/* Standard characters used for screen control */
#define ENTER '\015'
#define TAB   '\t'
#define BACKTAB '\02'  /*  ^B  */
#define CNTRLC '\03'
#define BACKSPACE '\010'
#define BELL '\07'
#define BLANK ' '
#define UNDERLINE '_'
#define ESCAPE '\033'
/*#define EOF     ((char)-1) */
#define TRIGGER '\021'  /* dc1 */
```

lib/iobuf.c

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 10:56:52 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/
#define DECLARE_IO_BUFFERS
#include "iobuf.h"
#undef DECLARE_IO_BUFFERS
#include "tpcc.h"
#include <errno.h>



string(str)
    char str[];
    {
    for (; *str != '\0'; str++)
        pushc(*str);
    }


push(str, len)
    char *str;
    int len;
    {
    for (; len > 0; len--)
        pushc(*str++);
    }


display(scr)
    iobuf *scr;
    {
/* Note: if problems doing output, let the input routine detect it */
    char *p;
    int len;
    for (p = scr->beg; p < scr->end; p+=len)
        {
```

```
        len = write(1, p, scr->end - p);
        if (len <= 0) break;
        }
    }


input(scr)
    iobuf *scr;
    {
    int len;

    /* read in as many characters as are available */
    len = read(0, scr->end, scr->max - scr->end);

    /* if end of input, then pretend we read an END character */
    if (len == 0 || (len == -1 && errno == ECONNRESET))
        {
        *scr->end = EOF;
        len = 1;
        }

    /* Check for errors */
    else if (len == -1)
        syserror("input(scr): unable to read stdin\n");

    /* update the pointers to reflect the new data */
    scr->end += len;
    *scr->end='\0';  /* for debugging */
    }



getkey()
    {
    if (in_buf->cur == in_buf->end)
        {
        flush();
        reset(in_buf);
        input(in_buf);
        }

    return popc();
    }
```

lib/random.c

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 2001/08/27 13:18:41 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/
#include "tpcc.h"
#include "string.h"
#include "random.h"

double drand48();

char lastNames[1000][16];
char customerData1[10][301];
char customerData2[10][201];
char stockData1[10][27];
char stockData2[10][25];
char historyData1[10][13];
char historyData2[10][13];
char citystreetData1[10][11];
char citystreetData2[10][11];
char firstNameData1[10][9];
char firstNameData2[10][9];
char StockDistrict[10][25];
char phoneData[10][17];

static long RandySeedIter = 7;

void GenerateLastNames()
{
    int     i;
    char *name;
    static char *n[] = {"BAR", "OUGHT", "ABLE", "PRI", "PRES",
                        "ESE", "ANTI", "CALLY", "ATION", "EING"};

    for(i = 0; i < 1000; i++) {
        name = lastNames[i];
        strcpy(name, n[(i/100)%10]);
        strcat(name, n[(i/10) %10]);
        strcat(name, n[(i/1)  %10]);
    }
}

int MakeNumberString(min, max, num)
    int min;
    int max;
    TEXT num[];
    {
    static char digit[]="0123456789";
    int length;
    int i;

    length = RandomNumber(min, max);

    for (i=0;  i<length;  i++)
        num[i] = digit[RandomNumber(0,9)];
    num[length] = '\0';

    return length;
    }

ID RandomWarehouse(local, scale, percent)
    ID local;
    ID scale;
    int percent;   /* percent of remote transactions */
    {
    ID w_id;

    /* For the given percent of the time, pick the local warehouse */
    if (RandomNumber(1, 100) > percent || scale == 1)
        w_id = local;

    /* Otherwise, pick a non-local warehouse */
    else
        {
        w_id = RandomNumber(2, scale);
        if (w_id == local)
            w_id = 1;
        }
    return w_id;
    }


/* Initialize a table of Random strings for the stock-district
   field in the stock table.  We can use a table of 10 elements
   and select randomly from this table via rule 4.3.2.2 in
   the TPC-C spec */
void InitRandomStrings()
{
    int i;

    for (i=0; i < 10; i++) {
                    MakeAlphaString(24,24,&StockDistrict[i]);

                    MakeAlphaString(300,300,&customerData1[i]);
                    MakeAlphaString(0,200,&customerData2[i]);

                    MakeAlphaString(26,26,&stockData1[i]);
                    MakeAlphaString(0,24,&stockData2[i]);

                    MakeAlphaString(12,12,&historyData1[i]);
                    MakeAlphaString(0,12, &historyData2[i]);

                    MakeAlphaString(10,10,&citystreetData1[i]);
                    MakeAlphaString(0 ,10,&citystreetData2[i]);

                    MakeAlphaString(8,8,&firstNameData1[i]);
                    MakeAlphaString(0,8,&firstNameData2[i]);

                    MakeNumberString(16,16,&phoneData[i]);
                }
                GenerateLastNames();
}

int MakeAlphaString(min, max, str)
    int min;
    int max;
    TEXT str[];
    {
    static char character[] =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
    int length;
    int i;

    length = RandomNumber(min, max);

    for (i=0;  i<length;  i++) {
        /* NOTE:  we use sizeof(character)-2 because of the following:
           subtract 1 because we are numbering from 0 instead of 1 and
           subtract 1 because the sizeof(character) is 1 greater than
           the data in character because of the inivisible C string
           terminator at the end. */
        str[i] = character[RandomNumber(0, sizeof(character)-2)];
    }
    str[length] = '\0';

    return length;
    }

void RandomPermutation(perm, n)
```

```
    int perm[];
    int n;
    {
    int i, r, t;

    /* generate the identity permutation to start with */
    for (i=1; i<=n; i++)
        perm[i] = i;

    /* randomly shuffle the permutation */
    for (i=1; i<=n; i++)
        {
        r = RandomNumber(i, n);
        t = perm[i]; perm[i] = perm[r]; perm[r] = t;
        }
    }


void RandomDelay(mean, adjust)
/*************************************************************************
random_sleep sleeps according to the TPC specification
*************************************************************************/
    double mean;
    double adjust;
    {
    double secs;
    double exponential();

    secs = exponential(mean);

    delay(secs+adjust);
    }

double exponential(mean)
/*************************************************************************
exponential generates a reverse exponential distribution
*************************************************************************/
    double mean;
    {
    double x;
    double log();

#ifdef USE_DRAND48
    x = -log(1.0-drand48()) * mean;
#else
    x = -log(1.0-randy()) * mean;
#endif

    return x;
    }

void SetRandomSeed(val)
long val;
    {
#ifdef USE_DRAND48
    srand48(val);
#else
    RandySeedIter = val;
    randy();
#endif
    }

void Randomize()
    {
    SetRandomSeed(time(0)+getpid());
    }

/* Random number generator from Proceeding of the ACM */
#define RANDY_A_VAL 16807
/* 2^31 - 1 */
#define RANDY_M_VAL 2147483647
/* m / a */
#define RANDY_Q_VAL 127773
/* m % a */
#define RANDY_R_VAL 2836

double randy()
{
  long    hi, lo, test;

  hi = RandySeedIter / RANDY_Q_VAL;
  lo = RandySeedIter % RANDY_Q_VAL;

  test = (RANDY_A_VAL * lo) - (RANDY_R_VAL * hi);
  RandySeedIter = (test > 0) ? test : test + RANDY_M_VAL;

  return( (double)RandySeedIter / (double)RANDY_M_VAL );
}  /* end of fn randy  */
```

lib/Makefile

```
#************************************************************************
*
#@(#) Version: A.10.10 $Date: 97/12/15 14:02:02 $
#
#(c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
#************************************************************************
*

include ../buildenv.mk

CFLAGS= $(BUILDFLAGS) -Wl,-a,archive_shared

utils=iobuf.o delay.o errlog.o fmt.o random.o tas.o null_key.o null_select.o results_file.o  date.o
prepare_socket.o shm.o spinlock.o

all: tpc_lib.a  server_default.o

tpc_lib.a: ${utils}
                rm -f tpc_lib.a
                ar -r tpc_lib.a ${utils}

clean:
                rm -f *.o
                rm -f *.a

clobber: clean

.s.o:
                cc -c $*.s
```

A.3 Transaction Source

client/service.c

```
/************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 10:53:26 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
************************************************************************/

#include <unistd.h>
#include <sys/types.h>
#include "tpcc.h"
#include "atmi.h"

extern int userid;
char *cmd          = NULL;

int tpsvrinit(argc, argv)
int    argc;
char **argv;
{
   char c;
   int ret;
   time_t t;

   t = time((time_t *) NULL);
   userlog("checking time: %s", ctime(&t));
   /*
    * search for the options
    *    "-n" server number
    *    "-S" server program
    * purpose: to get svr_id & progname for DVRY_LOG files
    *
    */
   while ((c = getopt(argc, argv, "n:S:h:")) != EOF) {
      switch(c) {
      case 'n':
         userid = atoi(optarg);
         break;
      case 'S':
         cmd  = optarg;
         break;
      }
   }

   ret = transaction_begin(userid);
   results_open(userid);

   return 0;
}


void NEWO_SVC(svcinfo)
TPSVCINFO *svcinfo;
{
   neworder_transaction((neworder_trans *)svcinfo->data);
   tpreturn(TPSUCCESS, 0, svcinfo->data, svcinfo->len, 0);
```

```
}


void PMT_SVC(svcinfo)
TPSVCINFO *svcinfo;
{

    payment_transaction((payment_trans *)svcinfo->data);
    tpreturn(TPSUCCESS, 0, svcinfo->data, svcinfo->len, 0);
}

void ORDS_SVC(svcinfo)
TPSVCINFO *svcinfo;
{
    ordstat_transaction((ordstat_trans *)svcinfo->data);
    tpreturn(TPSUCCESS, 0, svcinfo->data, svcinfo->len, 0);
}

void STKL_SVC(svcinfo)
TPSVCINFO *svcinfo;
{
    stocklev_transaction((stocklev_trans *)svcinfo->data);
    tpreturn(TPSUCCESS, 0, svcinfo->data, svcinfo->len, 0);
}

void DVRY_SVC(svcinfo)
TPSVCINFO *svcinfo;
{
    delivery_trans *t = (delivery_trans *)svcinfo->data;
    gettimeofday(t->deque, NULL);
    delivery_transaction(t);
    gettimeofday(t->com plete, NULL);
    results(t);

    /* Why do we return things ? */
    tpreturn(TPSUCCESS, 0, svcinfo->data, svcinfo->len, 0);
}

/*****************************************************************
tpsrvdone cleans up after the TPC transaction service
*****************************************************************/
void tpsvrdone()
{
    transaction_done();
    results_close();

    /* Log a message saying we are done */
    userlog("TUXEDO service %s has shutdown\n", cmd);
}
```

client/oracle/transaction.c

```
#include "ora_tpcc.h"
#include <time.h>
#include "tpcc.h"

#ifdef __STDC__
#include "ociapr.h"
#else
#include "ocikpr.h"
#endif

extern TPCinit(int, char*, char*);

int     numtrans = 0;

transaction_done()
{
   fprintf(stderr, "About to call TPCexit\n");fflush(stderr);
   TPCexit();
   fprintf(stderr, "TPCexit after %d transcations \n", numtrans);fflush(stderr);
}

/* void  */
transaction_begin(id)
int id;
{
   int ret;

   if ((ret=TPCinit(id, "tpcc", "tpcc")) == -1)
   {
                fprintf(stderr, "TPCinit failure!\n");fflush(stderr);
      /* Error */
   }
   numtrans = 0;
   return ret;
}

void neworder_transaction(str)
neworder_trans *str;
{
   int i;
   struct newstruct ora_str;

   ora_str.newin.w_id = str->W_ID;
   ora_str.newin.d_id = str->D_ID;
   ora_str.newin.c_id = str->C_ID;
   for (i = 0; i < str->O_OL_CNT; i++) {
      ora_str.newin.ol_i_id[i] = str->item[i].OL_I_ID;
      ora_str.newin.ol_supply_w_id[i] =  str->item[i].OL_SUPPLY_W_ID;
      ora_str.newin.ol_quantity[i] = str->item[i].OL_QUANTITY;
   }
   for (i = str->O_OL_CNT; i < 15; i++) {
      ora_str.newin.ol_i_id[i] = 0;
      ora_str.newin.ol_supply_w_id[i] = 0;
      ora_str.newin.ol_quantity[i] = 0;
   }

   numtrans++;
   if (TPCnew(&ora_str) == -1) {
      str->status = E_DB_ERROR;
      return;
   } else {
      str->status = OK;
   }

   str->O_ID = ora_str.newout.o_id;
   str->O_OL_CNT = ora_str.newout.o_ol_cnt;
   strncpy (str->C_LAST, ora_str.newout.c_last, 17);
   strncpy (str->C_CREDIT, ora_str.newout.c_credit, 3);
   str->C_DISCOUNT = (REAL) ora_str.newout.c_discount;
   str->W_TAX = (REAL) ora_str.newout.w_tax;
   str->D_TAX = (REAL) ora_str.newout.d_tax;
   strncpy (str->O_ENTRY_D, ora_str.newout.o_entry_d, 20);
   for (i = 0; i < ora_str.newout.o_ol_cnt; i++) {
      strncpy (str->item[i].I_NAME, ora_str.newout.i_name[i], 25);
      str->item[i].S_QUANTITY = ora_str.newout.s_quantity[i];
      str->item[i].brand_generic = ora_str.newout.brand_generic[i];
      str->item[i].I_PRICE = ora_str.newout.i_price[i]*100.0;  /* needs to be in cents */
   }
   str->status = ((ora_str.newout.status[0] != ' \0') ? E_INVALID_ITEM : OK);
}

/*****************************************************************************
 * Payment Query
 *****************************************************************************/

void
payment_transaction(str)
payment_trans *str;
{
   int i;

   struct paystruct ora_str;

   ora_str.payin.w_id = str->W_ID;
   ora_str.payin.d_id = str->D_ID;
   ora_str.payin.c_w_id = str->C_W_ID;
   ora_str.payin.c_d_id = str->C_D_ID;
   ora_str.payin.h_amount = str->H_AMOUNT; /* Amount in cents */
   ora_str.payin.bylastname = str->byname;
   if (ora_str.payin.bylastname) {
      ora_str.payin.c_id = 0;
      strncpy (ora_str.payin.c_last, str->C_LAST, 17);
      ora_str.payin.c_last[16] = ' \0';
      for (i = 15; ((i >= 0) && (ora_str.payin.c_last[i] == ' ')); i--)
         ora_str.payin.c_last[i] = ' \0';
   }
   else {
      ora_str.payin.c_id = str->C_ID;
      strcpy (ora_str.payin.c_last, " ");
   }
   retries = 0;

   numtrans++;
   if (TPCpay (&ora_str)) {
      str->status = E_DB_ERROR;
      return;
   } else {
      str->status = OK;
   }

   strncpy (str->W_STREET_1, ora_str.payout.w_street_1, 21);
   strncpy (str->W_STREET_2, ora_str.payout.w_street_2, 21);
   strncpy (str->W_CITY, ora_str.payout.w_city, 21);
   strncpy (str->W_STATE, ora_str.payout.w_state, 3);
   strncpy (str->W_ZIP, ora_str.payout.w_zip, 10);
   strncpy (str->D_STREET_1, ora_str.payout.d_street_1, 21);
   strncpy (str->D_STREET_2, ora_str.payout.d_street_2, 21);
   strncpy (str->D_CITY, ora_str.payout.d_city, 21);
   strncpy (str->D_STATE, ora_str.payout.d_state, 3);
   strncpy (str->D_ZIP, ora_str.payout.d_zip, 10);
   str->C_ID = ora_str.payout.c_id;
   strncpy (str->C_FIRST, ora_str.payout.c_first, 17);
   strncpy (str->C_MIDDLE, ora_str.payout.c_middle, 3);
```

```
  strncpy (str->C_LAST, ora_str.payout.c_last, 17);
  strncpy (str->C_STREET_1, ora_str.payout.c_street_1, 21);
  strncpy (str->C_STREET_2, ora_str.payout.c_street_2, 21);
  strncpy (str->C_CITY, ora_str.payout.c_city, 21);
  strncpy (str->C_STATE, ora_str.payout.c_state, 3);
  strncpy (str->C_ZIP, ora_str.payout.c_zip, 10);
  strncpy (str->C_PHONE, ora_str.payout.c_phone, 17);
  strncpy (str->C_SINCE, ora_str.payout.c_since, 11);

  strncpy (str->C_CREDIT, ora_str.payout.c_credit, 3);
  str->C_CREDIT_LIM = (MONEY) ora_str.payout.c_credit_lim*100.0; /* needs to be in cents
*/
  str->C_DISCOUNT = (REAL) ora_str.payout.c_discount;
  str->C_BALANCE = (REAL) ora_str.payout.c_balance*100.0;  /* needs to be in cents */
  /* Oracle passes 201 characters, we copy 200 and terminate on 201.       */
  strncpy (str->C_DATA, ora_str.payout.c_data, 200);
  str->C_DATA[200] = '\0';
  strncpy (str->H_DATE, ora_str.payout.h_date, 20);
}

void
ordstat_transaction(str)
ordstat_trans *str;
{
  int i;

  struct ordstruct ora_str;

  ora_str.ordin.w_id = str->W_ID;
  ora_str.ordin.d_id = str->D_ID;
  ora_str.ordin.by lastname = str->byname;
  if (ora_str.ordin.bylastname) {
    ora_str.ordin.c_id = 0;
    strncpy (ora_str.ordin.c_last, str->C_LAST, 17);
    ora_str.ordin.c_last[16] = '\0';
    for (i = 15; ((i >= 0) && (ora_str.ordin.c_last[i] == ' ')); i--)
      ora_str.ordin.c_last[i] = '\0';
  }
  else {
    ora_str.ordin.c_id = str->C_ID;
    strcpy (ora_str.ordin.c_last, " ");
  }
  retries = 0;

  numtrans++;
  if (TPCord (&ora_str)) {
    str->status = E_DB_ERROR;
    return;
  } else {
    str->status = OK;
  }

  str->C_ID = ora_str.ordout.c_id;
  strncpy (str->C_LAST, ora_str.ordout.c_last, 17);
  strncpy (str->C_FIRST, ora_str.ordout.c_first, 17);
  strncpy (str->C_MIDDLE, ora_str.ordout.c_middle, 3);
  str->C_BALANCE = (MONEY) ora_str.ordout.c_balance*100.0; /* needs to be in cents */
  str->O_ID = ora_str.ordout.o_id;
  strncpy (str->O_ENTRY_DATE, ora_str.ordout.o_entry_d, 20);
  str->O_CARRIER_ID = ora_str.ordout.o_carrier_id;
  str->ol_cnt = ora_str.ordout.o_ol_cnt;
  for (i = 0; i < ora_str.ordout.o_ol_cnt; i++) {
    str->item[i].OL_SUPPLY_W_ID = ora_str.ordout.ol_supply_w_id[i];
    str->item[i].OL_I_ID = ora_str.ordout.ol_i_id[i];
    str->item[i].OL_QUANTITY = ora_str.ordout.ol_quantity[i];
    str->item[i].OL_AMOUNT = (MONEY) ora_str.ordout.ol_amount[i]*100.0;  /* needs to be in
cents */
    strncpy (str->item[i].OL_DELIVERY_DATE, ora_str.ordout.ol_delivery_d[i], 11);
  }
}

/*****************************************************************************
 * Delivery Query
 *****************************************************************************/

void delivery_transaction(str)
delivery_trans *str;
{
  double tr_end;
  int i;

  struct delstruct ora_str;

  ora_str.delin.w_id        = str->W_ID;
  ora_str.delin.o_carrier_id = str->O_CARRIER_ID;
  retries      = 0;

  numtrans++;
  if (TPCdel (&ora_str)) {
    str->status = E_DB_ERROR;
    return;
  } else {
    str->status = OK;
  }
  for (i = 0; i < 10; i++) {
    if (del_o_id[i] <= 0) {
                str->order[i].status = E_NOT_ENOUGH_ORDERS;
    } else {
                str->order[i].status = OK;
                str->order[i].O_ID = del_o_id[i];
    }
  }
}

/*****************************************************************************
 * Stock Level Query
 *****************************************************************************/

void stocklev_transaction(str)
stocklev_trans *str;
{

  struct stostruct ora_str;
  ora_str.stoin.w_id = str->W_ID;
  ora_str.stoin.d_id = str->D_ID;
  ora_str.stoin.threshold = str->threshold;
  retries = 0;

  numtrans++;
  if (TPCsto (&ora_str)) {
    str->status = E_DB_ERROR;
    return;
  } else {
    str->status = OK;
  }
  str->low_stock = ora_str.stoout.low_stock;
}
```

client/oracle/tpccpl.c

```
#ifdef RCSID
static char *RCSid =
  "$Header: tpccpl.c 7030100.2 96/04/02 17:51:34 plai Generic<base> $ Copyr (c) 1994 Oracle";
#endif /* RCSID */


/*=========================== ===================================+
|      Copyright (c) 1994  Oracle Corp, Redwood Shores, CA      |
|              OPEN SYSTEMS PERFORMANCE GROUP                   |
|                 All Rights Reserved                           |
 +========== ===================================================+
| FILENAME
|    tpccpl.c
| DESCRIPTION
|    TPC-C transactions in PL/SQL.
|
| Ordered rows of update of stock_item in new_order by item id to prevent
| enqueue deadlocks - 11/18/99 - wbattist
|
| Fix bug in TPCnew for ordering rows of stock_item if number of items is
| NITEMS - 03/9/00 - wbattist
 +=============================================================*/

#include <stdio.h>
#include <time.h>
#include "ora_tpcc.h"
#ifdef TUX
#include <userlog.h>
#else
#include <stdarg.h>
#endif

#define SQLTXT "alter session set isolation_level = serializable"
#define SQLTXTTRC "alter session set sql_trace = true"
#define SQLTXTTIM "alter session set timed_statistics = true"

FILE *lfp;
FILE *fopen ();
#ifdef ORA_NT
#undef boolean
#include "dpbcore.h"
#define gettime dpbtimef
#else
extern double gettime ();
#endif
int proc_no = 0;
int logon = 0;
int new_init = 0;
int pay_init = 0;
int ord_init = 0;
int del_init = 0;
int sto_init = 0;
int res_init = 0;

int execstatus;
int errcode;
```

```
OCIEnv *tpcenv;
OCIServer *tpcsrv;
OCIError *errhp;
OCISvcCtx *tpcsvc;
OCISession *tpcusr;
OCIStmt *curi;


/* for stock-level transaction */

int w_id;
int d_id;
int c_id;
int threshold;
int low_stock;

/* for delivery transaction */

int del_o_id[10];
int retries;

/* for order-status transaction */

int bylastname;
char c_last[17];
char c_first[17];
char c_middle[3];
double c_balance;
int o_id;
text o_entry_d[20];
ub4  datelen;
int o_carrier_id;
int o_ol_cnt;
int ol_supply_w_id[15];
int ol_i_id[15];
int ol_quantity[15];
int ol_amount[15];
ub4  ol_del_len[15];
text ol_delivery_d[15][11];

/* for payment transaction */

int c_w_id;
int c_d_id;
int h_amount;
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_street_1[21];
char c_street_2[21];
char c_city[21];
char c_state[3];
char c_zip[10];
char c_phone[17];
ub4  sincelen;
text c_since_d[11];
float c_discount;
char c_credit[3];
int c_credit_lim;
char c_data[201];
ub4 hlen;
text h_date[20];

/* for new order transaction */

int nol_i_id[15];
int nol_supply_w_id[15];
int nol_quantity[15];
int nol_quanti10[15];
int nol_quanti91[15];
int nol_ytdqty[15];
int nol_amount[15];
int o_all_local;
float w_tax;
float d_tax;
float total_amount;
char i_name[15][25];
int s_quantity[15];
char brand_gen[15];
int i_price[15];
char brand_generic[15][1];
int status;
int tracelevel = 0;

OCIDate cr_date;
OCIDate c_since;
OCIDate o_entry_d_base;
OCIDate ol_d_base[15];
dvoid *xmem;

#ifdef AVOID_DEADLOCK
int indx[15], ordl_cnt;
void swap(struct newstruct *str, int i, int j);
void q_sort(int *arr, struct newstruct *str, int left, int right);
/* void disitems(int *itm, int wid, int did, int cid, int cnt); */
#endif

/*
extern char oracle_home[256];
*/

/* NewOrder Binding stuff */



#ifndef TUX
void userlog (char* fmtp, ...)
{
 va_list va;
 va_start(va,fmtp);
 vfprintf(stderr,fmtp,va);
 va_end(va);
}
#endif

/* vmm313 void ocierror(fname, lineno, errhp, status)  */
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);
/* vmm313   TPCexit(1); */
/* vmm313   exit(1); */
 case OCI_INVALID_HANDLE:
  fprintf(stderr,"Module %s Line %d\n", fname, lineno);
  fprintf(stderr,"Error - OCI_INVALID_HANDLE\n");
  TPCexit(1);
  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);
}
```

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

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

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

int sqlfile(fnam,linebuf)
char   *fnam;
text   *linebuf;
{
FILE *fd;
int nulpt = 0;
char realfile[512];

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

/*
    sprintf(realfile,"%s/bench/tpc/tpcc/blocks/%s",oracle_home,fnam);
*/
    sprintf(realfile,"%s",fnam);
    fd = vopen(realfile,"r");
    while (fgets((char *)linebuf+nulpt, SQL_BUF_SIZE,fd))
    {
       nulpt = strlen((char *)linebuf);
    }
    return(nulpt);
}

#ifdef NOT
void vgetdate (unsigned char *oradt)
{
 struct tm *loctime;
 time_t  int_time;

 struct ORADATE {
 unsigned charcentury;
 unsigned charyear;
 unsigned charmonth;
 unsigned charday;
 unsigned charhour;
 unsigned charminute;
 unsigned charsecond;
 } Date;
int century;
int cnvrtOK;

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

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

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

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

Date.century = (unsigned char)(century + 100);
if (Date.century < 119 || Date.century > 120) cnvrtOK = 0;
Date.year  = (unsigned char)(loctime->tm_year+100);
if (Date.year < 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);

      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,m onth,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 = Da te.second - 1;

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

      return;
}
#endif


void TPCexit (void)

{

  if (new_init) {
    tkvcndone();
    new_init = 0;
  }
  if (pay_init) {
    tkvcpdone();
    pay_init = 0;
  }
  if (ord_init) {
    tkvcodone();
    ord_init = 0;
  }
  if (del_init) {
    tkvcddone();
    del_init = 0;
  }
  if (sto_init) {
    tkvcsdone();
    sto_init = 0;
  }


  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);

  if (lfp) {
    fclose (lfp);
```

```
    lfp = NULL;
  }

}



TPCinit (id, uid, pwd)

int id;
char *uid;
char *pwd;

{

  char filename[40];
  text stmbuf[100];

  proc_no = id;
  sprintf (filename, "tpcc_%d.del", proc_no);
  if ((lfp = fopen (filename, "w")) == NULL) {
#ifdef TUX
    userlog ("Error in TPC-C server %d: Failed to open %s\n",
          proc_no, filename);
#else
    fprintf (stderr, "Error in TPC-C server %d: Failed to open %s\n",
          proc_no, filename);
#endif
    return (-1);
  }

  OCIInitialize(OCI_DEFAULT|OCI_OBJECT,(dvoid *)0,0,0,0);
  OCIEnvInit(&tpcenv, OCI_DEFAULT, 0, (dvoid **)0);
  OCIERROR(errhp, OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&tpcsrv,
OCI_HTYPE_SERVER, 0 , (dvoid **)0));
  OCIERROR(errhp, OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&errhp,
OCI_HTYPE_ERROR, 0 , (dvoid **)0));
  OCIERROR(errhp, 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);

  /* run all transaction in serializable mode */

  OCIHandleAlloc(tpcenv, (dvoid **)&curi, OCI_HTYPE_STMT, 0, (dvoid**)0);
  sprintf ((char *) stmbuf, SQLTXT);
  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);


/*
This is done in cvdrv.c
  if (tracelevel == 2) {
    OCIHandleAlloc(tpcenv, (dvoid **)&curi, OCI_HTYPE_STMT, 0, (dvoid**)0);
    memset(stmbuf,0,100);
    sprintf ((char *) stmbuf, SQLTXTTRC);
    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((dvoid *)curi, OCI_HTYPE_STMT);
  }
*/
  if (tracelevel == 3) {
    OCIHandleAlloc(tpcenv, (dvoid **)&curi, OCI_HTYPE_STMT, 0, (dvoid**)0);
    memset(stmbuf,0,100);
    sprintf ((char *) stmbuf, SQLTXTTIM);
    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((dvoid *)curi, OCI_HTYPE_STMT);
  }

  logon = 1;

  OCIERROR(errhp,OCIDateSysDate(errhp,&cr_date));

  if (tkvcninit ()) {  /* new order */
    TPCexit ();
    return (-1);
  }
  else
    new_init = 1;

  if (tkvcpinit ()) {   /* payment */
    TPCexit ();
    return (-1);
  }
  else
    pay_init = 1;

  if (tkvcoinit ()) {   /* order status */
    TPCexit ();
    return (-1);
  }
  else
    ord_init = 1;


  if (tkvcdinit ()) {   /* delivery */
    TPCexit ();
    return (-1);
  }
  else
    del_init = 1;

  if (tkvcsinit ()) {   /* stock level */
    TPCexit ();
    return (-1);
  }
  else
    sto_init = 1;

  return (0);

}



TPCnew (str)

struct newstruct *str;

{

  int i;

  w_id = str->newin.w_id;
  d_id = str->newin.d_id;
  c_id = str->newin.c_id;
  for (i = 0; i < 15; i++) {
    nol_i_id[i] = str->newin.ol_i_id[i];
    nol_supply_w_id[i] = str->newin.ol_supply_w_id[i];
    nol_quantity[i] = str->newin.ol_quantity[i];
  }
  retries = 0;

#ifdef AVOID_DEADLOCK

  ordl_cnt = NITEMS;

  for (i = 0; i < NITEMS; i++) {
    if (nol_i_id[i] == 0) {
      ordl_cnt = i;
      break;
    }
  }

  for(i=0;i<15;i++)
    indx[i] = i;

  q_sort(nol_i_id,str,0,ordl_cnt-1);

/*   disitems(nol_i_id, w_id, d_id, c_id, ordl_cnt); */
#endif

/*
  vgetdate(cr_date); */

  OCIERROR(errhp,OCIDateSysDate(errhp,&cr_date));

  if (str->newout.terror = tkvcn ()) {
    if (str->newout.terror != RECOVERR)
      str->newout.terror = IRRECERR;
    return (-1);
  }

  /* fill in date for o_entry_d from time in beginning of txn*/
/*
  cvtdmyhms(cr_date,o_entry_d);
*/
   datelen = sizeof(o_entry_d);
   OCIERROR(errhp,
      OCIDateToText(errhp,&cr_date,(text*)FULLDATE,SIZ(FULLDATE),(text*)0,0,
                &datele n,o_entry_d));

  str->newout.terror = NOERR;
  str->newout.o_id = o_id;
```

```
  str->newout.o_ol_cnt = o_ol_cnt;
  strncpy (str->newout.c_last, c_last, 17);
  strncpy (str->newout.c_credit, c_credit, 3);
  str->newout.c_discount = c_discount;
  str->newout.w_tax = (float)(w_tax);
  str->newout.d_tax = (float)(d_tax);
  strncpy (str->newout.o_entry_d, (char*)o_entry_d, 20);
  str->newout.total_amount = total_amount;
  for (i = 0; i < o_ol_cnt; i++) {
    strncpy (str->newout.i_name[i], i_name[i], 25);
    str->newout.s_quantity[i] = s_quantity[i];
    str->newout.brand_generic[i] = brand_generic[i][0];
    str->newout.i_price[i] = (float)(i_price[i])/100;
    str->newout.ol_amount[i] = (float)(nol_amount[i])/100;
  }
#ifdef AVOID_DEADLOCK
   q_sort(indx,str,0,ordl_cnt-1);
#endif

  if (status)
    strcpy (str->newout.status, "Item number is not valid");
  else
    str->newout.status[0] = ' \0';
  str->newout.retry = retries;
#if defined(TOP)  || defined(TUX)    /* changed mjb 17 feb for  tuxedo */
  return(1);
#else
  return (0);
#endif

}



TPCpay (str)

struct paystruct *str;

{

  w_id = str->payin.w_id;
  d_id = str->payin.d_id;
  c_w_id = str->payin.c_w_id;
  c_d_id = str->payin.c_d_id;
  h_amount = str->payin.h_amount;
  bylastname = str->payin.bylastname;

/*
  vgetdate(cr_date);  */
  OCIERROR(errhp,OCIDateSysDate(errhp,&cr_date));

  if (bylastname) {
    c_id = 0;
    strncpy (c_last, str->payin.c_last, 17);
  }
  else {
    c_id = str->payin.c_id;
    strcpy (c_last, " ");
  }
  retries = 0;

  if (str->payout.terror = tkvcp ()) {
    if (str->payout.terror != RECOVERR)
      str->payout.terror = IRRECERR;
    return (-1);
  }

/*
  cvtdmyhms(cr_date,h_date);
*/
      hlen=SIZ(h_date);
      OCIERROR(errhp,OCIDateToText(errhp,&cr_date,
            (text*)FULLDATE,strlen(FULLDATE),(text*)0,0,&hlen,h_date));

/*
  cvtdmy(c_since,c_since_d);
*/
      sincelen=SIZ(c_since_d);
      OCIERROR(errhp,OCIDateToText(errhp,&c_since,
        (text*)SHORTDATE,strlen(SHORTDATE),(text*)0,0,&sincelen,c_since_d));



  str->payout.terror = NOERR;
  strncpy (str->payout.w_street_1, w_street_1, 21);
  strncpy (str->payout.w_street_2, w_street_2, 21);
  strncpy (str->payout.w_city, w_city, 21);
  strncpy (str->payout.w_state, w_state, 3);
  strncpy (str->payout.w_zip, w_zip, 10);
  strncpy (str->payout.d_street_1, d_street_1, 21);
  strncpy (str->payout.d_street_2, d_street_2, 21);
  strncpy (str->payout.d_city, d_city, 21);
  strncpy (str->payout.d_state, d_state, 3);
  strncpy (str->payout.d_zip, d_zip, 10);
  str->payout.c_id = c_id;
  strncpy (str->payout.c_first, c_first, 17);
  strncpy (str->payout.c_middle, c_middle, 3);
  strncpy (str->payout.c_last, c_last, 17);
  strncpy ( str->payout.c_street_1, c_street_1, 21);
  strncpy (str->payout.c_street_2, c_street_2, 21);
  strncpy (str->payout.c_city, c_city, 21);
  strncpy (str->payout.c_state, c_state, 3);
  strncpy (str->payout.c_zip, c_zip, 10);
  strncpy (str->payout.c_phone, c_phone, 17);
  strncpy (str->payout.c_since, (char*)c_since_d, 11);
  strncpy (str->payout.c_credit, c_credit, 3);
  str->payout.c_credit_lim = (float)(c_credit_lim)/100;
  str->payout.c_discount = c_discount;
  str->payout.c_balance = (float)(c_balance)/100;
  strncpy (str->payout.c_data, c_data, 201);
  strncpy (str->payout.h_date, (char*)h_date, 20);
  str->payout.retry = retries;
#if defined(TOP)  || defined(TUX)   /* changed mjb 17 Feb */
  return(1);
#else
  return (0);
#endif

}



TPCord (str)

struct ordstruct *str;

{

  int i;
  w_id = str->ordin.w_id;
  d_id = str->ordin.d_id;
  bylastname = str->ordin.bylastname;
  if (bylastname) {
    c_id = 0;
    strncpy (c_last, str->ordin.c_last, 17);
  }
  else {
    c_id = str->ordin.c_id;
    strcpy (c_last, " ");
  }
  retries = 0;

  if (str->ordout.terror = tkvco ()) {
    if (str->ordout.terror != RECOVERR)
      str->ordout.terror = IRRECERR;
    return (-1);
  }

  datelen = sizeof(o_entry_d);
  OCIERROR( errhp,
      OCIDateToText(errhp,&o_entry_d_base,(text*)FULLDATE,SIZ(FULLDATE),(text*)0,0,
                &datelen,o_entry_d));

  str->ordout.terror = NOERR;
  str->ordout.c_id = c_id;
  strncpy (str->ordout.c_last, c_last, 17);
  strncpy (str->ordout.c_first, c_first, 17);
  strncpy (str->ordout.c_middle, c_middle, 3);
  str->ordout.c_balance = c_balance/100;
  str->ordout.o_id = o_id;
  strncpy (str->ordout.o_entry_d, (char*)o_entry_d, 20);
  if ( o_carrier_id == 11 )
     str->ordout.o_carrier_id = 0;
  else
     str->ordout.o_carrier_id = o_carrier_id;
  str->ordout.o_ol_cnt = o_ol_cnt;
  for (i = 0; i < o_ol_cnt; i++) {
    ol_delivery_d[i][10] = ' \0';
    if ( !strcmp((char*)ol_delivery_d[i],"15-09-1911") )
                strncpy((char*)ol_delivery_d[i],"NOT DELIVR",10);
    str->ordout.ol_supply_w_id[i] = ol_supply_w_id[i];
    str->ordout.ol_i_id[i] = ol_i_id[i];
    str->ordout.ol_quantity[i] = ol_quantity[i];
    str->ordout.ol_amount[i] = (float)(ol_amount[i])/100;
    strncpy (str->ordout.ol_delivery_d[i], (char*)ol_delivery_d[i], 11);
  }
  str->ordout.retry = retries;
#if defined(TOP)  || defined(TUX)
  return(1);
#else
  return (0);
#endif

}



TPCdel (str)

struct delstruct *str;
```

```
{

  double tr_end;
  int i;

  w_id = str->delin.w_id;
  o_carrier_id = str->delin.o_carrier_id;
  retries = 0;
/*
  vgetdate(cr_date);  */
  OCIERROR(errhp,OCIDateSysDate(errhp,&cr_date));

  if (str->delout.terror = tkvcd ()) {
    if(str->delout.terror == DEL_ERROR)
      return DEL_ERROR;
    if (str->delout.terror != RECOVERR)
      str->delout.terror = IRRECERR;
    return (-1);
  }

  /* Comment out for the HP kit.
  tr_end = gettime ();
  fprintf (lfp, "%d %d %f %f %d %d", str->delin.in_timing_int,
          (tr_end - str->delin.qtime) <= DELRT ? 1 : 0,
          str->delin.qtime, tr_end, w_id, o_carrier_id);
  for (i = 0; i < 10; i++) {
    fprintf (lfp, " %d %d", i + 1, del_o_id[i]);
    if (del_o_id[i] <= 0) {
#ifdef TUX
      userlog ("DELIVERY: no new order for w_id: %d, d_id %d\n",
            w_id, i + 1);
#else
      fprintf (stderr, "DELIVERY: no new order for w_id: %d, d_id %d\n",
            w_id, i + 1);
#endif
    }
  }
  fprintf (lfp, " %d\n", retries);
  */

  str->delout.terror = NOERR;
  str->delout.retry = retries;
#if defined(TOP) || defined(TUX)   /* changed mjb 17 feb */
  return(1);
#else
  return (0);
#endif

}



TPCsto (str)

struct stostruct *str;

{

  w_id = str->stoin.w_id;
  d_id = str->stoin.d_id;
  threshold = str->stoin.threshold;
  retries = 0;

  if (str->stoout.terror = tkvcs ()) {
    if (str->stoout.terror != RECOVERR)
      str->stoout.terror = IRRECERR;
    return (-1);
  }

  str->stoout.terror = NOERR;
  str->stoout.low_stock = low_stock;
  str->stoout.retry = retries;
#if defined(TOP) || defined(TUX) /* changed mjb 17 feb */
  return(1);
#else
  return (0);
#endif

}

#ifdef AVOID_DEADLOCK

void q_sort(int *arr,struct newstruct *str,int left, int right)
{
 int i, last;

 if(left >= right)
   return;
 swap(str,left,(left+right)/2);
 last = left;
 for(i=left+1;i<=right;i++)
  if(arr[i] < arr[left])
    swap(str,last,i);
 swap(str,left,last);
 q_sort(arr,str,left,last-1);
 q_sort(arr,str,last+1,right);
}

void swap(struct newstruct *str, int i, int j)
{
 int temp;
 float tempf;
 char tmpstr[25];
 char tmpch;

 temp = indx[i];
 indx[i] = indx[j];
 indx[j] = temp;

 temp = nol_i_id[i];
 nol_i_id[i] = nol_i_id[j];
 nol_i_id[j] = temp;

 temp = nol_supply_w_id[i];
 nol_supply_w_id[i] = nol_supply_w_id[j];
 nol_supply_w_id[j] = temp;

 temp = nol_quantity[i];
 nol_quantity[i] = nol_quantity[j];
 nol_quantity[j] = temp;

 strcpy(tmpstr,str->newout.i_name[i]);
 strcpy(str->newout.i_name[i],str->newout.i_name[j]);
 strcpy(str->newout.i_name[j],tmpstr);

 temp = str->newout.s_quantity[i];
 str->newout.s_quantity[i] = str->newout.s_quantity[j];
 str->newout.s_quantity[j] = temp;

 tmpch = str->newout.brand_generic[i];
 str->newout.brand_generic[i] = str->newout.brand_generic[j];
 str->newout.brand_generic[j] = tmpch;

 tempf = str->newout.i_price[i];
 str->newout.i_price[i] = str->newout.i_price[j];
 str->newout.i_price[j] = tempf;

 tempf = str->newout.ol_amount[i];
 str->newout.ol_amount[i] = str->newout.ol_amount[j];
 str->newout.ol_amount[j] = tempf;

}
/*
void disitems(itm, wid, did, cid, cnt)
int *itm;
int  wid;
int  did;
int  cid;
int  cnt;
{
 int i;
 int ordered = TRUE;

 for (i=1; i<15, itm[i]; i++)
   if (itm[i-1] > itm[i])
   {
     ordered = FALSE;
     break;
   }

 if (ordered)
   return;

 printf("w=%d, d=%d, c=%d, cnt=%d\n", wid, did, cid, cnt);
 for (i=0; i<15, itm[i] ; i++)
    printf("%d ", itm[i]);

 printf("\n");
}
*/
#endif
```

client/oracle/plnew.c

```
        #ifdef RCSID
static char *RCSid =
  "$Header: tkvcnew.c 21-apr-98.18:32:59 rdecker Exp $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=================================================================+
|      Copyright (c) 1996 , 1997, 1998 Oracle Corp, Redwood Shores, CA      |
|                OPEN SYSTEMS PERFORMANCE GROUP                |
|                    All Rights Reserved                       |
+=================================================================+
| FILENAME
|   plnew.c
```

```
| DESCRIPTION
|    OCI version (using PL/SQL stored procedure) of
|    NEW ORDER transaction in TPC-C benchmark.
 +================================================================*/

#include "ora_tpcc.h"
#ifdef TUX
#include <userlog.h>
#endif
#include "tpccflags.h"

extern void userlog();

#ifdef PLSQLNO
#define SQLTXT2 "BEGIN initnew.new_init(:idx1arr); END;"
#else
#define SQLTXT2 "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 \
   WHERE rowid = :s_rowid"

#define SQLTXT3 "\
SELECT 0,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :10 AND s_w_id = :30 AND s_i_id = i_id UNION ALL \
SELECT 1,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :11 AND s_w_id = :31 AND s_i_id = i_id UNION ALL \
SELECT 2,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :12 AND s_w_id = :32 AND s_i_id = i_id UNION ALL \
SELECT 3,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :13 AND s_w_id = :33 AND s_i_id = i_id UNION ALL \
SELECT 4,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :14 AND s_w_id = :34 AND s_i_id = i_id UNION ALL \
SELECT 5,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :15 AND s_w_id = :35 AND s_i_id = i_id UNION ALL \
SELECT 6,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :16 AND s_w_id = :36 AND s_i_id = i_id UNION ALL \
SELECT 7,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :17 AND s_w_id = :37 AND s_i_id = i_id UNION ALL \
SELECT 8,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :18 AND s_w_id = :38 AND s_i_id = i_id UNION ALL \
SELECT 9,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :19 AND s_w_id = :39 AND s_i_id = i_id UNION ALL \
SELECT 10,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :20 AND s_w_id = :40 AND s_i_id = i_id UNION ALL \
SELECT 11,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :21 AND s_w_id = :41 AND s_i_id = i_id UNION ALL \
SELECT 12,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :22 AND s_w_id = :42 AND s_i_id = i_id UNION ALL \
SELECT 13,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :23 AND s_w_id = :43 AND s_i_id = i_id UNION ALL \
SELECT 14,stok.rowid,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stok WHERE i_id = :24 AND s_w_id = :44 AND s_i_id = i_id"


#define SQLTXT4 "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, :null_date, :ol_i_id, :ol_supply_w_id, :ol_quantity, \
   :ol_amount, :ol_dist_info)"
#endif  /* PLSQLNO */

#define NITEMS 15
#define ROWIDLEN 20
#define OCIROWLEN 20

sb4 no_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
            dvoid **bufpp, ub4 *alenp, ub1 *piecep,
            dvoid **indpp)
{
  *bufpp = (dvoid*)0;
  *alenp =0;
  *indpp = (dvoid*)0;
  *piecep=OCI_ONE_PIECE;
  return (OCI_CONTINUE);
}
struct newctx {
  sb2 nol_i_id_ind[NITEMS];
  sb2 nol_supply_w_id_ind[NITEMS];
  sb2 nol_quantity_ind[NITEMS];
  sb2 nol_amount_ind[NITEMS];
  sb2 i_name_ind[NITEMS];
  sb2 s_quantity_ind[NITEMS];
  sb2 i_price_ind[NITEMS];
  sb2 ol_w_id_ind[NITEMS];
  sb2 ol_d_id_ind[NITEMS];
  sb2 ol_o_id_ind[NITEMS];
  sb2 ol_number_ind[NITEMS];
  sb2 cons_ind[NITEMS];
  sb2 s_rowid_ind[NITEMS];
  sb2 s_remote_ind[NITEMS];
  sb2 s_quant_ind[NITEMS];
  sb2 i_data_ind[NITEMS];
  sb2 s_data_ind[NITEMS];
  sb2 s_dist_info_ind[NITEMS];
  sb2 ol_dist_info_ind[NITEMS];
  sb2 null_date_ind[NITEMS];
#ifdef PLSQLNO
  sb2 s_bg_ind[NITEMS];
#endif

  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 i_data_len[NITEMS];
  ub2 s_dist_info_len[NITEMS];
  ub2 s_data_len[NITEMS];
  ub2 ol_w_id_len[NITEMS];
  ub2 ol_d_id_len[NITEMS];
  ub2 ol_o_id_len[NITEMS];
  ub2 ol_number_len[NITEMS];
  ub2 cons_len[NITEMS];
  ub2 s_rowid_len[NITEMS];
  ub2 s_remote_len[NITEMS];
  ub2 s_quant_len[NITEMS];
  ub2 ol_dist_info_len[NITEMS];
  ub2 null_date_len[NITEMS];
#ifdef PLSQLNO
  ub2 s_bg_len[NITEMS];
#endif

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

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

  OCIRowid  *s_rowid_ptr[NITEMS];

  int s_remote[NITEMS];
  char i_data[NITEMS][51];
  char s_data[NITEMS][51];
  char s_dist_info[NITEMS][25];
  OCIDate null_date[NITEMS]; /* base date for null date entry */
  OCIStmt *curn1;
#ifdef PLSQLNO
  OCIBind *ol_i_id_bp;
  OCIBind *ol_supply_w_id_bp;
  OCIBind *i_price_bp;
  OCIBind *i_name_bp;
  OCIBind *s_bg_bp;
  OCIBind *s_data_bp;
  OCIBind *i_data_bp;
  ub4 nol_i_count;
  ub4 nol_s_count;
  ub4 nol_q_count;
  ub4 nol_item_count;
  ub4 nol_name_count;
  ub4 nol_qty_count;
  ub4 nol_bg_count;
  ub4 nol_am_count;
  ub4 s_remote_count;
  ub4 s_data_count;
  ub4 i_data_count;
#endif
  OCIStmt *curn2;
  OCIStmt *curn3[10];
  OCIBind *ol_i_id_bp4;
  OCIBind *ol_supply_w_id_bp4;
  OCIBind *ol_quantity_bp;
  OCIBind *ol_quantity_bp4;
  OCIBind *s_remote_bp;
```

```
  OCIBind *s_quantity_bp;
  OCIStmt *curn4;
  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 *s_rowid_bp;
  OCIBind *id_bp[10][15];
  OCIBind *sd_bp[10][15];
  OCIDefine *Dcons[10];
  OCIDefine *Ds_rowid[10];
  OCIDefine *Di_price[10];
  OCIDefine *Di_data[10];
  OCIDefine *Ds_dist_info[10];
  OCIDefine *Ds_data[10];
  OCIDefine *Ds_quantity[10];
  OCIDefine *Di_name[10];
  OCIBind *ol_o_id_bp;
  OCIBind *ol_d_id_bp;
  OCIBind *ol_w_id_bp;
  OCIBind *ol_number_bp;
  OCIBind *ol_amount_bp;
  OCIBind *ol_dist_info_bp;
  OCIBind *null_date_bp;

  sb2 w_id_ind;
  ub2 w_id_len;
  ub2 w_id_rc;

  sb2 d_id_ind;
  ub2 d_id_len;
  ub2 d_id_rc;

  sb2 c_id_ind;
  ub2 c_id_len;
  ub2 c_id_rc;

  sb2 o_all_local_ind;
  ub2 o_all_local_len;
  ub2 o_all_local_rc;

  sb2 o_ol_cnt_ind;
  ub2 o_ol_cnt_len;
  ub2 o_ol_cnt_rc;

  sb2 w_tax_ind;
  ub2 w_tax_len;
  ub2 w_tax_rc;

  sb2 d_tax_ind;
  ub2 d_tax_len;
  ub2 d_tax_rc;

  sb2 o_id_ind;
  ub2 o_id_len;
  ub2 o_id_rc;

  sb2 c_discount_ind;
  ub2 c_discount_len;
  ub2 c_discount_rc;

  sb2 c_credit_ind;
  ub2 c_credit_len;
  ub2 c_credit_rc;

  sb2 c_last_ind;
  ub2 c_last_len;
  ub2 c_last_rc;

  sb2 retries_ind;
  ub2 retries_len;
  ub2 retries_rc;

  sb2 cr_date_ind;
  ub2 cr_date_len;
  ub2 cr_date_rc;

  int cs;
  int norow;

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

typedef struct newctx newctx;

newctx *nctx;

tkvcninit ()
{

  int i;
  text stmbuf[16*1024];
#ifndef PLSQLNO
  char sd[4];
  char id[4];
  int j;
# endif /* !PLSQLNO */

  nctx = (newctx *) malloc (sizeof(newctx));
  memset(nctx,(char)0,sizeof(newctx));
  nctx->cs = 1;
  nctx->norow=0;
  for(i=0;i<NITEMS;i++) {
    OCIERROR(errhp, OCIDescriptorAlloc(tpcenv,(dvoid**)&nctx->s_rowid_ptr[i],
      OCI_DTYPE_ROWID,0,(dvoid**)0));
  }
  nctx->w_id_ind = TRUE;
  nctx->w_id_len = sizeof(w_id);
  nctx->d_id_ind = TRUE;
  nctx->d_id_len = sizeof(d_id);
  nctx->c_id_ind = TRUE;
  nctx->c_id_len = sizeof(c_id);
  nctx->o_all_local_ind = TRUE;
  nctx->o_all_local_len = sizeof(o_all_local);
  nctx->o_ol_cnt_ind = TRUE;
  nctx->o_ol_cnt_len = sizeof(o_ol_cnt);
  nctx->w_tax_ind = TRUE;
  nctx->w_tax_len = 0;
  nctx->d_tax_ind = TRUE;
  nctx->d_tax_len = 0;
  nctx->o_id_ind = TRUE;
  nctx->o_id_len = sizeof(o_id);
  nctx->c_discount_ind = TRUE;
  nctx->c_discount_len = 0;
  nctx->c_credit_ind = TRUE;
  nctx->c_credit_len = 0;
  nctx->c_last_ind = TRUE;
  nctx->c_last_len = 0;
  nctx->retries_ind = TRUE;
  nctx->retries_len = sizeof(retries);
  nctx->cr_date_ind = TRUE;
  nctx->cr_date_len = sizeof(cr_date);

  /* open first cursor */
  OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid **)(&nctx->curn1),
        OCI_HTYPE_STMT, 0, (dvoid**)0));
#ifdef PLSQLNO
  sqlfile("/project/tpcc/blocks/tkvcpnew.sql",stmbuf);
#else
  sqlfile("/project/tpcc/blocks/tkvcbnew.sql",stmbuf);
#endif
  OCIERROR(errhp,OCIStmtPrepare(nctx->curn1, errhp, stmbuf, strlen((char *)stmbuf),
                                                    OCI_NTV_SYNTAX,
OCI_DEFAULT));

  /* bind variables */

  OCIBNDR(nctx->curn1, nctx->w_id_bp, errhp, ":w_id",ADR(w_id),SIZ(w_id),
        SQLT_INT, &nctx->w_id_ind, &nctx->w_id_len,&nctx->w_id_rc);
  OCIBNDR(nctx->curn1, nctx->d_id_bp, errhp, ":d_id",ADR(d_id),SIZ(d_id),
        SQLT_INT, &nctx->d_id_ind,&nctx->d_id_len,&nctx->d_id_rc);
  OCIBNDR(nctx->curn1, nctx->c_id_bp, errhp, ":c_id",ADR(c_id),SIZ(c_id),
        SQLT_INT, &nctx->c_id_ind, &nctx->c_id_len, &nctx->c_id_rc);
  OCIBNDR(nctx->curn1, nctx->o_all_local_bp, errhp, ":o_all_local",
        ADR(o_all_local), SIZ(o_all_local),SQLT_INT, &nctx->o_all_local_ind,
        &nctx->o_all_local_len,&nctx->o_all_local_rc);
  OCIBNDR(nctx->curn1, nctx->o_all_cnt_bp, errhp, ":o_ol_cnt",ADR(o_ol_cnt),
        SIZ(o_ol_cnt),SQLT_INT,&nctx->o_ol_cnt_ind,&nctx->o_ol_cnt_len,
        &nctx->o_ol_cnt_rc);
  OCIBNDR(nctx->curn1, nctx->w_tax_bp, errhp, ":w_tax",ADR(w_tax),SIZ(w_tax),
        SQLT_FLT, &nctx->w_tax_ind,&nctx->w_tax_len,&nctx->w_tax_rc);
  OCIBNDR(nctx->curn1, nctx->d_tax_bp, errhp, ":d_tax",ADR(d_tax),SIZ(d_tax),
        SQLT_FLT, &nctx->d_tax_ind, &nctx->d_tax_len,&nctx->d_tax_rc);
  OCIBNDR(nctx->curn1, nctx->o_id_bp, errhp, ":o_id",ADR(o_id),SIZ(o_id),
        SQLT_INT, &nctx->o_id_ind,&nctx->o_id_len,&nctx->o_id_rc);
  OCIBNDR(nctx->curn1, nctx->c_discount_bp, errhp, ":c_discount",
        ADR(c_discount), SIZ(c_discount),SQLT_FLT,
      &nctx->c_discount_ind,&nctx->c_discount_len,&nctx->c_discount_rc);
  OCIBNDR(nctx->curn1, nctx->c_credit_bp, errhp, ":c_credit",c_credit,
        SIZ(c_credit),SQLT_CHR,
      &nctx->c_credit_ind,&nctx->c_credit_len,&nctx->c_credit_rc);
  OCIBNDR(nctx->curn1, nctx->c_last_bp, errhp, ":c_last",c_last,SIZ(c_last),
        SQLT_STR, &nctx->c_last_ind, &nctx->c_last_len, &nctx->c_last_rc);
  OCIBNDR(nctx->curn1, nctx->retries_bp, errhp, ":retry",ADR(retries),
        SIZ(retries),SQLT_INT,
```

```
    &nctx->retries_ind, &nctx->retries_len, &nctx->retries_rc);
 OCIBNDR(nctx->curn1, nctx->cr_date_bp, errhp, ":cr_date",&cr_date,SIZ(OCIDate),
SQLT_ODT, &nctx->cr_date_ind, &nctx->cr_date_len, &nctx->cr_date_rc);

#ifdef PLSQLNO
 OCIBNDRAA(nctx->curn1, nctx->ol_i_id_bp,errhp,":ol_i_id",nol_i_id,
        SIZ(int), SQLT_INT, nctx->nol_i_id_ind,nctx->nol_i_id_len,
        nctx->nol_i_id_rcode,NITEMS,&nctx->nol_i_count);
 OCIBNDRAA(nctx->curn1, nctx->ol_supply_w_id_bp, errhp, ":ol_supply_w_id",
        nol_supply_w_id,SIZ(int),SQLT_INT, nctx->nol_supply_w_id_ind,
        nctx->nol_supply_w_id_len, nctx->nol_supply_w_id_rcode,
        NITEMS, &nctx->nol_s_count);
 OCIBNDRAA(nctx->curn1, nctx->ol_quantity_bp,errhp,":ol_quantity",nol_quantity,
        SIZ(int),SQLT_INT,nctx->nol_quantity_ind,nctx->nol_quantity_len,
        nctx->nol_quantity_rcode,NITEMS,&nctx->nol_q_count);
 OCIBNDRAA(nctx->curn1, nctx->i_price_bp,errhp,":i_price",i_price,SIZ(int),
         SQLT_INT, nctx->i_price_ind,nctx->i_price_len,nctx->i_price_rcode,
         NITEMS, &nctx->nol_item_count);
 OCIBNDRAA(nctx->curn1, nctx->i_name_bp,errhp,":i_name",i_name,
          SIZ(i_name[0]),SQLT_STR, nctx->i_name_ind,nctx->i_name_len,
          nctx->i_name_rcode,NITEMS,&nctx->nol_name_count);
 OCIBNDRAA(nctx->curn1, nctx->s_quantity_bp,errhp,":s_quantity",s_quantity,
        SIZ(int), SQLT_INT,nctx->s_quant_ind,nctx->s_quant_len,
        nctx->s_quant_rcode,NITEMS,&nctx->nol_qty_count);
 OCIBNDRAA(nctx->curn1, nctx->s_bg_bp,errhp,":brand_generic",brand_generic,
        SIZ(char), SQLT_CHR,nctx->s_bg_ind,nctx->s_bg_len,
        nctx->s_bg_rcode,NITEMS,&nctx->nol_bg_count);
 OCIBNDRAA(nctx->curn1, nctx->ol_amount_bp,errhp,":ol_amount",nol_amount,
        SIZ(int),SQLT_INT, nctx->nol_amount_ind,nctx->nol_amount_len,
        nctx->nol_amount_rcode,NITEMS,&nctx->nol_am_count);
 OCIBNDRAA(nctx->curn1, nctx->s_remote_bp,errhp,":s_remote",nctx->s_remote,
        SIZ(int),SQLT_INT, nctx->s_remote_ind,nctx->s_remote_len,
        nctx->s_remote_rcode,NITEMS,&nctx->s_remote_count);

 /* open second cursor */
 OCIERROR(errhp,OCIHandleAlloc(tpcenv, (dvoid **)(&nctx->curn2), OCI_HTYPE_STMT,
                                                0, (dvoid**)0));
 sprintf ((char *) stmbuf, SQLTXT2);
 OCIERROR(errhp,OCIStmtPrepare(nctx->curn2, errhp, stmbuf,
        strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));


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

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


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

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

 /* open second cursor */
 OCIERROR(errhp,OCIHandleAlloc(tpcenv, (dvoid **)(&nctx->curn2), OCI_HTYPE_STMT,
                                                0, (dvoid**)0));
 sprintf ((char *) stmbuf, SQLTXT2);
 OCIERROR(errhp,OCIStmtPrepare(nctx->curn2, errhp, stmbuf,
        strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));



 /* bind variables */

 OCIBNDRA(nctx->curn2, nctx->s_quantity_bp,errhp,":s_quantity",s_quantity,
        SIZ(int), SQLT_INT,nctx->s_quant_ind,nctx->s_quant_len,
        nctx->s_quant_rcode);
 OCIBNDRA(nctx->curn2, nctx->s_rowid_bp, errhp, ":s_rowid",nctx->s_rowid_ptr,
        sizeof(nctx->s_rowid_ptr[0]),SQLT_RDD,nctx->s_rowid_ind,
                nctx->s_rowid_len,nctx->s_rowid_rcode);
 OCIBNDRA(nctx->curn2, nctx->ol_quantity_bp,errhp,":ol_quantity",nol_quantity,
                SIZ(int),SQLT_INT,nctx->nol_quantity_ind,nctx->nol_quantity_len,
                nctx->nol_quantity_rcode);
 OCIBNDRA(nctx->curn2, nctx->s_remote_bp, errhp, ":s_remote",nctx->s_remote,
        SIZ(int), SQLT_INT,nctx->s_remote_ind,nctx->s_remote_len,
                nctx->s_remote_rcode);

 /* open third cursor and bind variables */

 for (i = 0; i < 10; i++)
 {
  j = i + 1;
  OCIERROR(errhp,OCIHandleAlloc(tpcenv, (dvoid **)&((nctx->curn3)[i]),
                                                OCI_HTYPE_STMT, 0,
(dvoid**)0));

  sprintf ((char *) stmbuf, SQLTXT3, j, j, j, j, j, j, j, j, j, j, j,
                j, j, j);

  OCIERROR(errhp,OCIStmtPrepare((nctx->curn3)[i], errhp, stmbuf,
                strlen((char *)stmbuf),OCI_NTV_SYNTAX,
                                                OCI_DEFAULT));
  OCIERROR(errhp,
        OCIAttrSet(nctx->curn3[i],OCI_HTYPE_STMT,(dvoid*)&nctx->norow,0,
            OCI_ATTR_PREFETCH_ROWS,errhp));
  for (j = 0; j < NITEMS; j++)
  {
   sprintf (id, ":%d", j + 10);
   sprintf (sd, ":%d", j + 30);
   OCIBNDRA((nctx->curn3)[i],(nctx->id_bp)[i][j],errhp,id,ADR(nol_i_id[j]),
                SIZ(int),SQLT_INT,
                &nctx->nol_i_id_ind[j],&nctx->nol_i_id_len[j],
                &nctx->nol_i_id_rcode[j]);
   OCIBNDRA((nctx->curn3)[i],(nctx->sd_bp)[i][j],errhp,sd,
                ADR(nol_supply_w_id[j]),SIZ(int),SQLT_INT,
                &nctx->nol_supply_w_id_ind[j],&nctx->nol_supply_w_id_len[j],
                &nctx->nol_supply_w_id_rcode[j]);
   nctx->nol_i_id_ind[j] = NA;
   nctx->nol_supply_w_id_ind[j] = NA;
   nctx->nol_i_id_len[j] = sizeof(int);
   nctx->nol_supply_w_id_len[j] = sizeof(int);
  }

  OCIDEF((nctx->curn3)[i],(nctx->Dcons)[i],errhp,1,&(nctx->cons[0]),
                SIZ(nctx->cons[0]),SQLT_INT);
  OCIDEF((nctx->curn3)[i], (nctx->Ds_rowid)[i],errhp,2,
        nctx->s_rowid_ptr, sizeof(nctx->s_rowid_ptr[0]), SQLT_RDD);
  OCIDEF((nctx->curn3)[i], (nctx->Di_price)[i],errhp,3,i_price,SIZ(int),
        SQLT_INT);

  OCIDFNRA((nctx->curn3)[i], (nctx->Di_name)[i],errhp,4,i_name,
        SIZ(i_name[0]),SQLT_STR, nctx->i_name_ind,nctx->i_name_len,
        nctx->i_name_rcode);
  OCIDFNRA((nctx->curn3)[i], (nctx->Di_data)[i],errhp,5,nctx->i_data,
                SIZ(nctx->i_data[0]), SQLT_STR, NULL,nctx->i_data_len,NULL);
  OCIDFNRA((nctx->curn3)[i], (nctx->Ds_dist_info)[i],errhp,6,
        nctx->s_dist_info, SIZ(nctx->s_dist_info[0]),SQLT_STR,
        NULL,nctx->s_dist_info_len, NULL);
  OCIDFNRA((nctx->curn3)[i],(nctx->Ds_data)[i],errhp,7,nctx->s_data,
                SIZ(nctx->s_data[0]),SQLT_STR,NULL,nctx->s_data_len,NULL);
  OCIDEF((nctx->curn3)[i],(nctx->Ds_quantity)[i],errhp,8,s_quantity,
                SIZ(int),SQLT_INT);

 }

 /* open fourth cursor */
 OCIHandleAlloc(tpcenv, (dvoid **)(&nctx->curn4), OCI_HTYPE_STMT, 0,
          (dvoid**)0);
 sprintf ((char *) stmbuf, SQLTXT4);
 OCIStmtPrepare(nctx->curn4, errhp, stmbuf, strlen((char *)stmbuf),
                        OCI_NTV_SYNTAX, OCI_DEFAULT);
 /* bind variables */

 OCIBNDRA(nctx->curn4, nctx->ol_o_id_bp,errhp,":ol_o_id",nctx->ol_o_id,
        SIZ(int),SQLT_INT, NULL,nctx->ol_o_id_len,
        NULL);
 OCIBNDRA(nctx->curn4, nctx->ol_d_id_bp,errhp,":ol_d_id",nctx->ol_d_id,
        SIZ(int),SQLT_INT, NULL,nctx->ol_d_id_len,
        NULL);
 OCIBNDRA(nctx->curn4, nctx->ol_w_id_bp,errhp,":ol_w_id",nctx->ol_w_id,
        SIZ(int),SQLT_INT, NULL,nctx->ol_w_id_len,
        NULL);
 OCIBNDRA(nctx->curn4, nctx->ol_number_bp,errhp,":ol_number",nctx->ol_number,
        SIZ(int),SQLT_INT, NULL,nctx->ol_number_len,
        NULL);
 OCIBNDRA(nctx->curn4, nctx->ol_i_id_bp4,errhp,":ol_i_id",nol_i_id,SIZ(int),
        SQLT_INT, NULL,nctx->nol_i_id_len, NULL);
 OCIBNDRA(nctx->curn4, nctx->ol_supply_w_id_bp4,errhp,":ol_supply_w_id",
        nol_supply_w_id,SIZ(int),SQLT_INT, NULL,
        nctx->nol_supply_w_id_len, NULL);
 OCIBNDRA(nctx->curn4, nctx->ol_quantity_bp4,errhp,":ol_quantity",nol_quantity,
        SIZ(int),SQLT_INT, NULL,nctx->nol_quantity_len,
                NULL);
 OCIBNDRA(nctx->curn4, nctx->ol_amount_bp,errhp,":ol_amount",nol_amount,
        SIZ(int),SQLT_INT, NULL,nctx->nol_amount_len,
        NULL);
 OCIBNDRA(nctx->curn4, nctx->ol_dist_info_bp,errhp,":ol_dist_info",
        nctx->s_dist_info, SIZ(nctx->s_dist_info[0]),SQLT_AFC,
```

```
        NULL, nctx->ol_dist_info_len,
        NULL);
  OCIBNDRA(nctx->curn4, nctx->null_date_bp,errhp,":null_date",nctx->null_date,
        SIZ(OCIDate), SQLT_ODT,NULL,
        nctx->null_date_len, NULL);

  /* set up the null date Null date is 15-sep-11 */
  for (i=0;i<NITEMS;i++)
  {
    OCIDateSetDate(&nctx->null_date[i],(sb2)1811,(ub1)9,(ub1)15);
  }
#endif

  return (0);

}



tkvcn ()
{

   int i;
   int rcount;
# ifndef PLSQLNO
   ub4 flags;
   int rowoff,rpc,rpc3,iters,j,k;
# endif /* !PLSQLNO */
   int failed = 0;

retry:

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

   /* get number of order lines, and check if all are local */

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

   nctx->w_id_ind = TRUE;
   nctx->w_id_len = sizeof(w_id);
   nctx->d_id_ind = TRUE;
   nctx->d_id_len = sizeof(d_id);
   nctx->c_id_ind = TRUE;
   nctx->c_id_len = sizeof(c_id);
   nctx->o_all_local_ind = TRUE;
   nctx->o_all_local_len = sizeof(o_all_local);
   nctx->o_ol_cnt_ind = TRUE;
   nctx->o_ol_cnt_len = sizeof(o_ol_cnt);
   nctx->w_tax_ind = TRUE;
   nctx->w_tax_len = 0;
   nctx->d_tax_ind = TRUE;
   nctx->d_tax_len = 0;
   nctx->o_id_ind = TRUE;
   nctx->o_id_len = sizeof(o_id);
   nctx->c_discount_ind = TRUE;
   nctx->c_discount_len = 0;
   nctx->c_credit_ind = TRUE;
   nctx->c_credit_len = 0;
   nctx->c_last_ind = TRUE;
   nctx->c_last_len = 0;
   nctx->retries_ind = TRUE;
   nctx->retries_len = sizeof(retries);
   nctx->cr_date_ind = TRUE;
   nctx->cr_date_len = sizeof(cr_date);
#ifdef PLSQLNO
   /* this is the row count */
   rcount = o_ol_cnt;
   nctx->nol_i_count = o_ol_cnt;
   nctx->nol_q_count = o_ol_cnt;
   nctx->nol_s_count = o_ol_cnt;
   nctx->s_remote_count = o_ol_cnt;

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

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

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

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

#else
   execstatus = OCIStmtExecute(tpcsvc,nctx->curn1,errhp,1,0,0,0,OCI_DEFAULT);
#endif


   if(execstatus != OCI_SUCCESS) {
     OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
     errcode = OCIERROR(errhp,execstatus);
     if(errcode == NOT_SERIALIZABLE) {
                retries++;
                goto retry;
     } else if (errcode == RECOVERR) {
```

```
                retries++;
                goto retry;
    } else if (errcode == SNAPSHOT_TOO_OLD) {
                retries++;
                goto retry;
    } else {
                return -1;
    }
   }

#ifdef PLSQLNO
   /* did the txn succeed ? */
   if (rcount != o_ol_cnt)
   {
    status = rcount - o_ol_cnt;
    o_ol_cnt = rcount;
   }
#endif

#ifdef DEBUG
   printf("w_id = %d, d_id = %d, c_id = %d\n",w_id, d_id, c_id);
#endif

#ifndef PLSQLNO
   /* initialization for array operations */

   for (i = 0; i < o_ol_cnt; i++) {
     nctx->ol_w_id[i] = w_id;
     nctx->ol_d_id[i] = d_id;
     nctx->ol_number[i] = i + 1;
     nctx->null_date_ind[i]= TRUE;
     nctx->nol_i_id_ind[i] = TRUE;
     nctx->nol_supply_w_id_ind[i] = TRUE;
     nctx->nol_quantity_ind[i] = TRUE;
     nctx->nol_amount_ind[i] = TRUE;
     nctx->ol_w_id_ind[i] = TRUE;
     nctx->ol_d_id_ind[i] = TRUE;
     nctx->ol_o_id_ind[i] = TRUE;
     nctx->ol_number_ind[i] = TRUE;
     nctx->ol_dist_info_ind[i] = TRUE;
     nctx->s_remote_ind[i] = TRUE;
     nctx->s_quant_ind[i] = TRUE;
     nctx->cons_ind[i] = TRUE;
     nctx->s_rowid_ind[i] = TRUE;

     nctx->nol_i_id_len[i] = sizeof(int);
     nctx->nol_supply_w_id_len[i] = sizeof(int);
     nctx->nol_quantity_len[i] = sizeof(int);
     nctx->nol_amount_len[i] = sizeof(int);
     nctx->ol_w_id_len[i] = sizeof(int);
     nctx->ol_d_id_len[i] = sizeof(int);
     nctx->ol_o_id_len[i] = sizeof(int);
     nctx->ol_number_len[i] = sizeof(int);
     nctx->ol_dist_info_len[i] = nctx->s_dist_info_len[i];
     nctx->null_date_len[i]=sizeof(OCIDate);
     nctx->s_remote_len[i] = sizeof(int);
     nctx->s_quant_len[i] = sizeof(int);
     nctx->s_rowid_len[i] = sizeof(nctx->s_rowid_ptr[0]);
     nctx->cons_len[i] = sizeof(int);
   }
   for (i = o_ol_cnt; i < NITEMS; i++) {
     nctx->nol_i_id_ind[i] = NA;
     nctx->nol_supply_w_id_ind[i] = NA;
     nctx->nol_quantity_ind[i] = NA;
     nctx->nol_amount_ind[i] = NA;
     nctx->ol_w_id_ind[i] = NA;
     nctx->ol_d_id_ind[i] = NA;
     nctx->ol_o_id_ind[i] = NA;
     nctx->ol_number_ind[i] = NA;
     nctx->ol_dist_info_ind[i] = NA;
     nctx->null_date_ind[i]= NA;
     nctx->s_remote_ind[i] = NA;
     nctx->s_quant_ind[i] = NA;
     nctx->cons_ind[i] = NA;
     nctx->s_rowid_ind[i] = NA;

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

   rpc3 = SelItemStk ();
   if (rpc3 == -2)
     goto retry;
   else if (rpc3 == -1)
     return (-1);

   /* compute order line amounts, total amount and stock quantities */

   total_amount = 0.0;
   for (i = 0; i < o_ol_cnt; i++)
    {
     nctx->ol_o_id[i] = o_id;
     if (nctx->nol_i_id_ind[i] != NA) {
       s_quantity[i] -= nol_quantity[i];
       if (s_quantity[i] < 10)
         s_quantity[i] += 91;
       nol_amount[i] = (nol_quantity[i] * i_price[i]);
       total_amount += nol_amount[i];
       if (strstr (nctx->i_data[i], "ORIGINAL") &&
           strstr (nctx->s_data[i], "ORIGINAL"))
         brand_gen[i] = 'B';
       else
         brand_gen[i] = 'G';
     }

    }
   total_amount *= ((float)(1 - c_discount)) * (1.0 + ((float)d_tax) + ((float)w_tax));
   total_amount = total_amount/100;

   rpc = UpdStk2 ();
   if (rpc == -2)
     goto retry;
   else if (rpc == -1)
     return (-1);

   /* error processing - will keep it separated for readablity */
   /* number of items selected != number of stock updated */

   if (rpc3 != rpc) {
#ifdef TUX
      userlog ("Error in TPC-C server %d: %d rows of item read, ",
               proc_no, rpc3);
      userlog ("                    but %d rows of stock updated\n", rpc);
#else
      fprintf (stderr, "Error in TPC-C server %d: %d rows of item read, ",
               proc_no, rpc3);
      fprintf (stderr, "                    but %d rows of stock update \n", rpc);
#endif
      /* rollback */
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      return (-1);
   }


/* common code for insert into order_line */
   for (i=0; i< o_ol_cnt; i++)  /* move district info in place */
   {
    nctx->ol_dist_info_len[i]=nctx->s_dist_info_len[i];
   }

   /* array insert into order line table */
   flags= (status ? OCI_DEFAULT : (OCI_DEFAULT|OCI_COMMIT_ON_SUCCESS));
   if ((o_ol_cnt - status) > 0)
   {
     execstatus = OCIStmtExecute(tpcsvc,nctx->curn4,errhp,o_ol_cnt - status,
                                 0,0,0,flags);
     if(execstatus != OCI_SUCCESS) {
       OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
       errcode = OCIERROR(errhp,execstatus);
       if(errcode == NOT_SERIALIZABLE) {
         retries++;
         goto retry;
       } else if (errcode == RECOVERR) {
         retries++;
         goto retry;
       } else if (errcode == SNAPSHOT_TOO_OLD) {
         retries++;
         goto retry;
       } else {
         return -1;
       }
     }
     OCIAttrGet(nctx->curn4,OCI_HTYPE_STMT,&rcount,NULL,
                                   OCI_ATTR_ROW_COUNT, errhp);
     if (rcount != (o_ol_cnt - status))
     {
#ifdef TUX
       userlog ("Error in TPC-C server %d: array insert failed\n",
                                   proc_no);
#else
       fprintf (stderr, "Error in TPC-C server %d: array insert failed\n",
                proc_no);
#endif
       /* rollback */
       OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
       return (-1);
     }
   }
```

```
  /* commit if no invalid item */

  if (status) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
#ifndef TUX
    fflush(stdout);
#endif
  }

#endif
  total_amount = 0.0;
  for (i = 0; i < o_ol_cnt; i++)
  {
    if (nctx->nol_amount_ind[i] != NA) {
      total_amount += nol_amount[i];
    }

  }
  total_amount *= ((float)(1 - c_discount)) * (float)(1.0 + ((float)(d_tax)) + ((float)(w_tax)));
  total_amount = total_amount/100;

  return (0);

}



void tkvcndone ()

{

  int i;

  if (nctx)
  {
    OCIHandleFree((dvoid *)nctx->curn1,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)nctx->curn2,OCI_HTYPE_STMT);
    for (i = 0; i < 10; i++)
      OCIHandleFree((dvoid *)(nctx->curn3)[i],OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)nctx->curn4,OCI_HTYPE_STMT);
    free (nctx);
  }
}

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

void shiftitemstock (i, j)

int i, j;

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

  /* shift up the order_line values */

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

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

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

void swapitemstock (i, j)

int i, j;

{

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

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

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

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

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

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

  tempsb2 = nctx->s_quantity_ind[i];
  nctx->s_quantity_ind[i] = nctx->s_quantity_ind[j];
  nctx->s_quantity_ind[j] = tempsb2;
  tempub2 = nctx->s_quantity_len[i];
  nctx->s_quantity_len[i] = nctx->s_quantity_len[j];
  nctx->s_quantity_len[j] = tempub2;
  tempub2 = nctx->s_quantity_rcode[i];
  nctx->s_quantity_rcode[i] = nctx->s_quantity_rcode[j];
  nctx->s_quantity_rcode[j] = tempub2;
  tempi = s_quantity[i];
  s_quantity[i] = s_quantity[j];
  s_quantity[j] = tempi;

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

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

}
```

```
SelItemStk ()
{

  int i, j, rpc3,rcount;

  /* array select from item and stock tables */
  execstatus=OCIStmtExecute(tpcsvc,(nctx->curn3)[d_id-1],errhp,o_ol_cnt,
                                    0,0,0,OCI_DEFAULT);
  if((execstatus != OCI_SUCCESS) && (execstatus != OCI_NO_DATA)) {
   errcode = OCIERROR(errhp,execstatus);
   if(errcode == NOT_SERIALIZABLE) {
      retries++;
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      return (-2);
   } else if (errcode == RECOVERR) {
   /* In case of NO_DATA this should NOT return, but simply fall through */
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      retries++;
      return (-2);
   } else if (errcode == SNAPSHOT_TOO_OLD) {
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      retries++;
      return (-2);
   } else {
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      return (-1);
   }
  }
  /* mark invalid items */
  OCIAttrGet((nctx->curn3)[d_id-1], OCI_HTYPE_STMT,&rcount,NULL,
                    OCI_ATTR_ROW_COUNT, errhp);
  rpc3 = rcount;

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

  if ((status = o_ol_cnt - rcount) >1)
  {
#ifdef TUX
     userlog ("TPC-C server %d: more than 1 invalid item?\n", proc_no);
#else
     fprintf (stderr, "TPC-C server %d: more than 1 invalid item?\n", proc_no)
;
#endif
    return (rpc3);
  }
  if (status == 0) return (rpc3);

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

  for (i = 0; i < o_ol_cnt; i++) {
   if (nctx->cons[i] != i) break;   /* this item is invalid */
  }


#ifdef TUX
        userlog ("TPC-C server %d: reordering items and stocks \n",
              proc_no);
#else
/*
        fprintf (stderr, "TPC-C server %d: reordering items and stocks \n",
              proc_no);   */
#endif

  /* not the last item - shift up */

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

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

  return (rpc3);

}



UpdStk2 ()
{

  int rpc, rcount;

  /* array update of stock table */

  execstatus = OCIStmtExecute(tpcsvc,nctx->curn2,errhp,o_ol_cnt-status,0,0,0,
                                                    OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
       retries++;
                return (-2);
    } else if (errcode == RECOVERR) {
       retries++;
                return (-2);
    } else if (errcode == SNAPSHOT_TOO_OLD) {
       retries++;
                return (-2);
    } else {
       return -1;
    }
  }
  OCIAttrGet(nctx->curn2,OCI_HTYPE_STMT,&rcount,NULL, OCI_ATTR_ROW_COUNT,
errhp);
  rpc = rcount;

  if (rpc != (o_ol_cnt - status)) {
#ifdef TUX
     userlog ("Error in TPC-C server %d: array update failed\n",
           proc_no);
#else
     fprintf (stderr, "Error in TPC-C server %d: array update failed\n",
           proc_no);
#endif
     OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
     return (-1);
  }

  return (rpc);

}
```

client/oracle/plpay.c

```
#ifdef RCSID
static char *RCSid =
   "$Header: plpay.c 7030100.1 95/07/19 14:44:59 plai Generic<base> $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*============================================================+
|      Copyright (c) 1995  Oracle Corp, Redwood Shores, CA    |
|              OPEN SYSTEMS PERFORMANCE GROUP                 |
|                 All Rights Reserved                         |
 +============================================================+
| FILENAME
|   plpay.c
| DESCRIPTION
|   OCI version (using PL/SQL stored procedure) of
|   PAYMENT transaction in TPC-C benchmark.
 +=================================== ==========================*/

#include "ora_tpcc.h"
#include "tpccflags.h"

#ifdef TUX
#include <userlog.h>
#endif


#define SQLTXT_INIT "BEGIN initpay.pay_init; END;"
#define SQLTXT_STP  "begin payment.dopayment(:w_id,:d_id,:c_w_id,:c_d_id,     \
                :c_id,:by_lname,:h_amount,:c_last,:w_street_1,:w_street_2,\
                :w_city,:w_state,:w_zip,:d_street_1,:d_street_2,:d_city,  \
                :d_state,:d_zip,:c_first,:c_middle,:c_street_1,           \
                :c_street_2,:c_city,:c_state,:c_zip,:c_phone,:c_since,    \
                :c_credit,:c_credit_lim,:c_discount,:c_balance,:c_data,   \
                :cr_date,:retry); end;"
```

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

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

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

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

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

  OCIBind *by_lname_bp;

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

  OCIBind *c_credit_lim_bp;
  OCIBind *c_credit_lim_bp1;
  sb2 c_credit_lim_ind;
  ub2 c_credit_lim_len;
  ub2 c_credit_lim_rc;

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

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

  OCIBind *c_data_bp;
  OCIBind *c_data_bp1;
```

```
  sb2 c_data_ind;
  ub2 c_data_len;
  ub2 c_data_rc;

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

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

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

  OCIBind *byln_bp;
  sb2 byln_ind;
  ub2 byln_len;
  ub2 byln_rc;
};

typedef struct payctx payctx;



payctx *pctx;

int tkvcpinit (void)

{

   text stmbuf[SQL_BUF_SIZE];
   pctx = (payctx *)malloc(sizeof(payctx));
   memset(pctx,(char)0,sizeof(payctx));

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

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

   /*  build the init statement  and execute it */

   sprintf ((char*)stmbuf, SQLTXT_INIT);
   OCIERROR(errhp,OCIStmtPrepare(pctx->curpi, errhp, stmbuf,
          strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));
   OCIERROR(errhp,
     OCIStmtExecute(tpcsvc,pctx->curpi,errhp,1,0,0,0,OCI_DEFAULT));
#ifdef PLSQLPAY
  /*   prepare the stub for calling plsql stored procedure */
   sprintf ((char*)stmbuf, SQLTXT_STP);
   OCIERROR(errhp,OCIStmtPrepare(pctx->curp0, errhp, stmbuf,
          strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));
#else

   /* customer id !=  0, go by last name */

   sqlfile("/project/tpcc/blocks/pay nz.sql",stmbuf);
   OCIERROR(errhp,OCIStmtPrepare(pctx->curp0, errhp, stmbuf,
          strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));

   /* customer id ==  0, go by last name */

   sqlfile("/project/tpcc/blocks/payz.sql",stmbuf); /* sqlfile opens $O/bench/.../blocks/... */
   OCIERROR(errhp,OCIStmtPrepare(pctx->curp1, errhp, stmbuf,
          strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));

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



   /* bind variables */


  OCIBNDR(pctx->curp0, pctx->w_id_bp, errhp,":w_id",ADR(w_id),SIZ(int),
        SQLT_INT, &pctx->w_id_ind, NULL, NULL);
  OCIBNDR(pctx->curp0, pctx->d_id_bp, errhp,":d_id",ADR(d_id),SIZ(int),
        SQLT_INT, &pctx->d_id_ind, NULL, NULL);
  OCIBND(pctx->curp0, pctx->c_w_id_bp, errhp,":c_w_id",ADR(c_w_id),SIZ(int),
        SQLT_INT);
  OCIBND(pctx->curp0, pctx->c_d_id_bp, errhp,":c_d_id",ADR(c_d_id),SIZ(int),
        SQLT_INT);
  OCIBND(pctx->curp0, pctx->c_id_bp, errhp,":c_id",ADR(c_id),SIZ(int),
        SQLT_INT);
#ifdef PLSQLPAY
  OCIBND(pctx->curp0, pctx->by_lname_bp, errhp,":by_lname",ADR(bylastname),
        SIZ(int), SQLT_INT);
#endif
  OCIBNDR(pctx->curp0, pctx->h_amount_bp, errhp,":h_amount",ADR(h_amount),
        SIZ(int),SQLT_INT, &pctx->h_amount_ind, &pctx->h_amount_len,
        &pctx->h_amount_rc);
  OCIBNDR(pctx->curp0, pctx->c_last_bp, errhp,":c_last",c_last,SIZ(c_last),
        SQLT_STR, &pctx->c_last_ind, &pctx->c_last_len, &pctx->c_last_rc);
  OCIBNDR(pctx->curp0, pctx->w_street_1_bp, errhp,":w_street_1",w_street_1,
        SIZ(w_street_1),SQLT_STR, &pctx->w_street_1_ind,
        &pctx->w_street_1_len,&pctx->w_street_1_rc);
  OCIBNDR(pctx->curp0, pctx->w_street_2_bp, errhp,":w_street_2",w_street_2,
        SIZ(w_street_2),SQLT_STR, &pctx->w_street_2_ind,
        &pctx->w_street_2_len,&pctx->w_street_2_rc);
  OCIBNDR(pctx->curp0, pctx->w_city_bp, errhp,":w_city",w_city,SIZ(w_city),
        SQLT_STR, &pctx->w_city_ind, &pctx->w_city_len,&pctx->w_city_rc);
  OCIBNDR(pctx->curp0, pctx->w_state_bp, errhp,":w_state",w_state,SIZ(w_state),
        SQLT_STR, &pctx->w_state_ind, &pctx->w_state_len,&pctx->w_state_rc);
  OCIBNDR(pctx->curp0, pctx->w_zip_bp, errhp,":w_zip",w_zip,SIZ(w_zip),
        SQLT_STR, &pctx->w_zip_ind, &pctx->w_zip_len,&pctx->w_zip_rc);
  OCIBNDR(pctx->curp0, pctx->d_street_1_bp, errhp,":d_street_1",d_street_1,
        SIZ(d_street_1),SQLT_STR, &pctx->d_street_1_ind,
        &pctx->d_street_1_len, &pctx->d_street_1_rc);
  OCIBNDR(pctx->curp0, pctx->d_street_2_bp, errhp,":d_street_2",d_street_2,
        SIZ(d_street_2),SQLT_STR, &pctx->d_street_2_ind,
        &pctx->d_street_2_len, &pctx->d_street_2_rc);
```

```
  OCIBNDR(pctx->curp0, pctx->d_city_bp, errhp,":d_city",d_city,SIZ(d_city),
        SQLT_STR, &pctx->d_city_ind,&pctx->d_city_len,&pctx->d_city_rc);
  OCIBNDR(pctx->curp0, pctx->d_state_bp, errhp,":d_state",d_state,SIZ(d_state),
        SQLT_STR, &pctx->d_state_ind,&pctx->d_state_len, &pctx->d_state_rc);
  OCIBNDR(pctx->curp0, pctx->d_zip_bp, errhp,":d_zip",d_zip,SIZ(d_zip),
        SQLT_STR, &pctx->d_zip_ind, &pctx->d_zip_len, &pctx->d_zip_rc);
  OCIBNDR(pctx->curp0, pctx->c_first_bp, errhp,":c_first",c_first,SIZ(c_first),
        SQLT_STR, &pctx->c_first_ind, &pctx->c_first_len,&pctx->c_first_rc);
  OCIBNDR(pctx->curp0, pctx->c_middle_bp, errhp,":c_middle",c_middle,2,
        SQLT_AFC, &pctx->c_middle_ind,&pctx->c_middle_len,
        &pctx->c_middle_rc);
  OCIBNDR(pctx->curp0, pctx->c_street_1_bp, errhp,":c_street_1",c_street_1,
        SIZ(c_street_1),SQLT_STR, &pctx->c_street_1_ind,
        &pctx->c_street_1_len,&pctx->c_street_1_rc);
  OCIBNDR(pctx->curp0, pctx->c_street_2_bp, errhp,":c_street_2",c_street_2,
        SIZ(c_street_2),SQLT_STR, &pctx->c_street_2_ind,
        &pctx->c_street_2_len,&pctx->c_street_2_rc);
  OCIBNDR(pctx->curp0, pctx->c_city_bp, errhp,":c_city",c_city,SIZ(c_city),
        SQLT_STR, &pctx->c_city_ind, &pctx->c_city_len,&pctx->c_city_rc);
  OCIBNDR(pctx->curp0, pctx->c_state_bp, errhp,":c_state",c_state,SIZ(c_state),
        SQLT_STR, &pctx->c_state_ind, &pctx->c_state_len,&pctx->c_state_rc);
  OCIBNDR(pctx->curp0, pctx->c_zip_bp, errhp,":c_zip",c_zip,SIZ(c_zip),
        SQLT_STR, &pctx->c_zip_ind, &pctx->c_zip_len, &pctx->c_zip_rc);
  OCIBNDR(pctx->curp0, pctx->c_phone_bp, errhp,":c_phone",c_phone,SIZ(c_phone),
SQLT_STR, &pctx->c_phone_ind,&pctx->c_phone_len, &pctx->c_phone_rc);
  OCIBNDR(pctx->curp0, pctx->c_since_bp, errhp,":c_since",&c_since,
        SIZ(OCIDate), SQLT_ODT, &pctx->c_since_ind,&pctx->c_since_len,
        &pctx->c_since_rc);
  OCIBNDR(pctx->curp0, pctx->c_credit_bp, errhp,":c_credit",c_credit,
        SIZ(c_credit),SQLT_CHR, &pctx->c_credit_ind, &pctx->c_credit_len,
        &pctx->c_credit_rc);
  OCIBNDR(pctx->curp0, pctx->c_credit_lim_bp, errhp,":c_credit_lim",
        ADR(c_credit_lim),SIZ(int), SQLT_INT, &pctx->c_credit_lim_ind,
        &pctx->c_credit_lim_len, &pctx->c_credit_lim_rc);
  OCIBNDR(pctx->curp0, pctx->c_discount_bp, errhp,":c_discount",
        ADR(c_discount),SIZ(c_discount), SQLT_FLT, &pctx->c_discount_ind,
        &pctx->c_discount_len,&pctx->c_discount_rc);
  OCIBNDR(pctx->curp0, pctx->c_balance_bp, errhp,":c_balance",ADR(c_balance),
        SIZ(double),SQLT_FLT, &pctx->c_balance_ind,&pctx->c_balance_len,
        &pctx->c_balance_rc);
  OCIBNDR(pctx->curp0, pctx->c_data_bp, errhp,":c_data",c_data,SIZ(c_data),
        SQLT_STR, &pctx->c_data_ind,&pctx->c_data_len,&pctx->c_data_rc);
/*
  OCIBNDR(pctx->curp0, pctx->h_date_bp, errhp,":h_date",h_date,SIZ(h_date),
        SQLT_STR, &pctx->h_date_ind, &pctx->h_date_len,&pctx->h_date_rc);
*/
  OCIBNDR(pctx->curp0, pctx->retries_bp, errhp,":retry",ADR(retries),SIZ(int),
        SQLT_INT, &pctx->retries_ind, &pctx->retries_len,&pctx->retries_rc);
  OCIBNDR(pctx->curp0, pctx->cr_date_bp, errhp,":cr_date",ADR(cr_date),
        SIZ(OCIDate),SQLT_ODT,&pctx->cr_date_ind, &pctx->cr_date_len,
        &pctx->cr_date_rc);
#ifndef PLSQLPAY

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


  OCIBNDR(pctx->curp1, pctx->w_id_bp1, errhp,":w_id",ADR(w_id),SIZ(int),
        SQLT_INT, &pctx->w_id_ind, &pctx->w_id_len,&pctx->w_id_rc);
  OCIBNDR(pctx->curp1, pctx->d_id_bp1, errhp,":d_id",ADR(d_id),SIZ(int),
        SQLT_INT, &pctx->d_id_ind,&pctx->d_id_len,&pctx->d_id_rc);
  OCIBND(pctx->curp1, pctx->c_w_id_bp1, errhp,":c_w_id",ADR(c_w_id),SIZ(int),
        SQLT_INT);
  OCIBND(pctx->curp1, pctx->c_d_id_bp1, errhp,":c_d_id",ADR(c_d_id),SIZ(int),
        SQLT_INT);
  OCIBNDR(pctx->curp1, pctx->c_id_bp1, errhp,":c_id",ADR(c_id),SIZ(int),
        SQLT_INT, &pctx->c_id_ind, &pctx->c_id_len, &pctx->c_id_rc);
  OCIBNDR(pctx->curp1, pctx->h_amount_bp1, errhp,":h_amount",ADR(h_amount),
        SIZ(int),SQLT_INT, &pctx->h_amount_ind, &pctx->h_amount_len,
        &pctx->h_amount_rc);
  OCIBND(pctx->curp1, pctx->c_last_bp1, errhp,":c_last",c_last,SIZ(c_last),
        SQLT_STR);
  OCIBNDR(pctx->curp1, pctx->w_street_1_bp1, errhp,":w_street_1",w_street_1,
        SIZ(w_street_1),SQLT_STR, &pctx->w_street_1_ind,
        &pctx->w_street_1_len,&pctx->w_street_1_rc);
  OCIBNDR(pctx->curp1, pctx->w_street_2_bp1, errhp,":w_street_2",w_street_2,
        SIZ(w_street_2),SQLT_STR, &pctx->w_street_2_ind,
        &pctx->w_street_2_len,&pctx->w_street_2_rc);
  OCIBNDR(pctx->curp1, pctx->w_city_bp1, errhp,":w_city",w_city,SIZ(w_city),
        SQLT_STR, &pctx->w_city_ind,&pctx->w_city_len,&pctx->w_city_rc);
  OCIBNDR(pctx->curp1, pctx->w_state_bp1, errhp,":w_state",w_state,SIZ(w_state),
        SQLT_STR, &pctx->w_state_ind, &pctx->w_state_len,&pctx->w_state_rc);
  OCIBNDR(pctx->curp1, pctx->w_zip_bp1, errhp,":w_zip",w_zip,SIZ(w_zip),
        SQLT_STR, &pctx->w_zip_ind, &pctx->w_zip_len,&pctx->w_zip_rc);
  OCIBNDR(pctx->curp1, pctx->d_street_1_bp1, errhp,":d_street_1",d_street_1,
        SIZ(d_street_1),SQLT_STR, &pctx->d_street_1_ind,
        &pctx->d_street_1_len, &pctx->d_street_1_rc);
  OCIBNDR(pctx->curp1, pctx->d_street_2_bp1, errhp,":d_street_2",d_street_2,
        SIZ(d_street_2),SQLT_STR, &pctx->d_street_2_ind,
        &pctx->d_street_2_len, &pctx->d_street_2_rc);
  OCIBNDR(pctx->curp1, pctx->d_city_bp1, errhp,":d_city",d_city,SIZ(d_city),
        SQLT_STR, &pctx->d_city_ind,&pctx->d_city_len,&pctx->d_city_rc);
  OCIBNDR(pctx->curp1, pctx->d_state_bp1, errhp,":d_state",d_state,
        SIZ(d_state), SQLT_STR, &pctx->d_state_ind,&pctx->d_state_len,
        &pctx->d_state_rc);
  OCIBNDR(pctx->curp1, pctx->d_zip_bp1, errhp,":d_zip",d_zip,SIZ(d_zip),
        SQLT_STR, &pctx->d_zip_ind, &pctx->d_zip_len, &pctx->d_zip_rc);
  OCIBNDR(pctx->curp1, pctx->c_first_bp1, errhp,":c_first",c_first,
        SIZ(c_first), SQLT_STR, &pctx->c_first_ind, &pctx->c_first_len,
        &pctx->c_first_rc);
  OCIBNDR(pctx->curp1, pctx->c_middle_bp1, errhp,":c_middle",c_middle,2,
        SQLT_AFC, &pctx->c_middle_ind,&pctx->c_middle_len,
        &pctx->c_middle_rc);

  OCIBNDR(pctx->curp1, pctx->c_street_1_bp1, errhp,":c_street_1",c_street_1,
        SIZ(c_street_1),SQLT_STR, &pctx->c_street_1_ind,
        &pctx->c_street_1_len,&pctx->c_street_1_rc);
  OCIBNDR(pctx->curp1, pctx->c_street_2_bp1, errhp,":c_street_2",c_street_2,
        SIZ(c_street_2),SQLT_STR, &pctx->c_street_2_ind,
        &pctx->c_street_2_len,&pctx->c_street_2_rc);
  OCIBNDR(pctx->curp1, pctx->c_city_bp1, errhp,":c_city",c_city,SIZ(c_city),SQLT_STR,
                &pctx->c_city_ind,&pctx->c_city_len,&pctx->c_city_rc);
  OCIBNDR(pctx->curp1, pctx->c_state_bp1, errhp,":c_state",c_state,SIZ(c_state),
SQLT_STR, &pctx->c_state_ind, &pctx->c_state_len,&pctx->c_state_rc);
  OCIBNDR(pctx->curp1, pctx->c_zip_bp1, errhp,":c_zip",c_zip,SIZ(c_zip),
        SQLT_STR, &pctx->c_zip_ind, &pctx->c_zip_len, &pctx->c_zip_rc);
  OCIBNDR(pctx->curp1, pctx->c_phone_bp1, errhp,":c_phone",c_phone,SIZ(c_phone),
        SQLT_STR, &pctx->c_phone_ind,&pctx->c_phone_len, &pctx->c_phone_rc);
  OCIBNDR(pctx->curp1, pctx->c_since_bp1, errhp,":c_since",&c_since,
        SIZ(OCIDate), SQLT_ODT, &pctx->c_since_ind,&pctx->c_since_len,
        &pctx->c_since_rc);
  OCIBNDR(pctx->curp1, pctx->c_credit_bp1, errhp,":c_credit",c_credit,
        SIZ(c_credit),SQLT_CHR, &pctx->c_credit_ind, &pctx->c_credit_len,
        &pctx->c_credit_rc);
  OCIBNDR(pctx->curp1, pctx->c_credit_lim_bp1, errhp,":c_credit_lim",
        ADR(c_credit_lim),SIZ(int), SQLT_INT, &pctx->c_credit_lim_ind,
        &pctx->c_credit_lim_len, &pctx->c_credit_lim_rc);
  OCIBNDR(pctx->curp1, pctx->c_discount_bp1, errhp,":c_discount",
        ADR(c_discount),SIZ(c_discount), SQLT_FLT, &pctx->c_discount_ind,
        &pctx->c_discount_len,&pctx->c_discount_rc);
  OCIBNDR(pctx->curp1, pctx->c_balance_bp1, errhp,":c_balance",ADR(c_balance),
        SIZ(double),SQLT_FLT, &pctx->c_balance_ind,&pctx->c_balance_len,
        &pctx->c_balance_rc);
  OCIBNDR(pctx->curp1, pctx->c_data_bp1, errhp,":c_data",c_data,SIZ(c_data),
        SQLT_STR, &pctx->c_data_ind,&pctx->c_data_len,&pctx->c_data_rc);
/*
  OCIBNDR(pctx->curp1, pctx->h_date_bp1, errhp,":h_date",h_date,SIZ(h_date),
        SQLT_STR, &pctx->h_date_ind, &pctx->h_date_len,&pctx->h_date_rc);
*/
  OCIBNDR(pctx->curp1, pctx->retries_bp1, errhp,":retry",ADR(retries),SIZ(int),
        SQLT_INT, &pctx->retries_ind, &pctx->retries_len,&pctx->retries_rc);
  OCIBNDR(pctx->curp1, pctx->cr_date_bp1, errhp,":cr_date",ADR(cr_date),
        SIZ(OCIDate),SQLT_ODT, &pctx->cr_date_ind, &pctx->cr_date_len,
        &pctx->cr_date_rc);
#endif

  return (0);

}


tkvcp ()

{

retry:

  pctx->w_id_ind = TRUE;
  pctx->w_id_len = SIZ(w_id);
  pctx->d_id_ind = TRUE;
  pctx->d_id_len = SIZ(d_id);
  pctx->c_w_id_ind = TRUE;
  pctx->c_w_id_len = 0;
  pctx->c_d_id_ind = TRUE;
  pctx->c_d_id_len = 0;
  pctx->c_id_ind = TRUE;
  pctx->c_id_len = 0;
  pctx->h_amount_len = SIZ(h_amount);
  pctx->h_amount_ind = TRUE;
  pctx->c_last_ind = TRUE;
  pctx->c_last_len = SIZ(c_last);
  pctx->w_street_1_ind = TRUE;
  pctx->w_street_1_len = 0;
  pctx->w_street_2_ind = TRUE;
  pctx->w_street_2_len = 0;
  pctx->w_city_ind = TRUE;
  pctx->w_city_len = 0;
  pctx->w_state_ind = TRUE;
  pctx->w_state_len = 0;
  pctx->w_zip_ind = TRUE;
  pctx->w_zip_len = 0;
  pctx->d_street_1_ind = TRUE;
  pctx->d_street_1_len = 0;
  pctx->d_street_2_ind = TRUE;
  pctx->d_street_2_len = 0;
  pctx->d_city_ind = TRUE;
  pctx->d_city_len = 0;
  pctx->d_state_ind = TRUE;
  pctx->d_state_len = 0;
  pctx->d_zip_ind = TRUE;
```

```
  pctx->d_zip_len = 0;
  pctx->c_first_ind = TRUE;
  pctx->c_first_len = 0;
  pctx->c_middle_ind = TRUE;
  pctx->c_middle_len = 0;
  pctx->c_street_1_ind = TRUE;
  pctx->c_street_1_len = 0;
  pctx->c_street_2_ind = TRUE;
  pctx->c_street_2_len = 0;
  pctx->c_city_ind = TRUE;
  pctx->c_city_len = 0;
  pctx->c_state_ind = TRUE;
  pctx->c_state_len = 0;
  pctx->c_zip_ind = TRUE;
  pctx->c_zip_len = 0;
  pctx->c_phone_ind = TRUE;
  pctx->c_phone_len = 0;
  pctx->c_since_ind = TRUE;
  pctx->c_since_len = 0;
  pctx->c_credit_ind = TRUE;
  pctx->c_credit_len = 0;
  pctx->c_credit_lim_ind = TRUE;
  pctx->c_credit_lim_len = 0;
  pctx->c_discount_ind = TRUE;
  pctx->c_discount_len = 0;
  pctx->c_balance_ind = TRUE;
  pctx->c_balance_len = sizeof(double);
  pctx->c_data_ind = TRUE;
  pctx->c_data_len = 0;
  pctx->h_date_ind = TRUE;
  pctx->h_date_len = 0;
  pctx->retries_ind = TRUE;
  pctx->retries_len = sizeof(retries);
  pctx->cr_date_ind = TRUE;
  pctx->cr_date_len = 7;

#ifdef PLSQLPAY
    execstatus=OCIStmtExecute(tpcsvc,pctx-
>curp0,errhp,1,0,0,0,OCI_DEFAULT|OCI_COMMIT_ON_SUCCESS);
#else
  if(bylastname) {
    execstatus=OCIStmtExecute(tpcsvc,pctx-
>curp1,errhp,1,0,0,0,OCI_DEFAULT|OCI_COMMIT_ON_SUCCESS);
  } else {
    execstatus=OCIStmtExecute(tpcsvc,pctx-
>curp0,errhp,1,0,0,0,OCI_DEFAULT|OCI_COMMIT_ON_SUCCESS);
  }
#endif


  if(execstatus != OCI_SUCCESS) {
     OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
       retries++;
       goto retry;
    } else if (errcode == RECOVERR) {
       retries++;
       goto retry;
    } else if (errcode == SNAPSHOT_TOO_OLD) {
       retries++;
       goto retry;
    } else {
       return -1;
    }
   }
   return 0;
}



void tkvcpdone ()

{
  if(pctx) {
    free(pctx);
  }
}
```

client/oracle/plord.c

```
#ifdef RCSID
static char *RCSid =
   "$Header: plord.c 7030100.1 95/07/19 14:46:13 plai Generic<base> $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=========================================================================+
|      Copyright (c) 1995  Oracle Corp, Redwood Shores, CA      |
|                OPEN SYSTEMS PERFORMANCE GROUP                 |
|                   All Rights Reserved                         |
+=========================================================================+
| FILENAME
|    plord.c
| DESCRIPTION
|    OCI version (using PL/SQL anynomous block) of
|    ORDER STATUS transaction in TPC-C benchmark.
+=========================================================================*/

#include "ora_tpcc.h"
#include "tpccflags.h"

#ifdef PLSQLORD
#define SQLTXT "BEGIN orderstatus.getstatus (:w_id, :d_id, :c_id, :byln, \
   :c_last, :c_first, :c_middle, :c_balance, :o_id, :o_entry_d, :o_cr_id, \
   :o_ol_cnt, :ol_s_w_id, :ol_i_id, :ol_quantity, :ol_amount, :ol_d_d); END;"
#else

#define SQLCUR0 "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 SQLCUR1 "SELECT c_id, c_balance, c_first, c_middle, c_last, \
              o_id, o_entry_d, o_carrier_id, o_ol_cnt \
              FROM cust, ordr \
              WHERE cust.rowid = :cust_rowid \
              AND   o_d_id = c_d_id AND o_w_id = c_w_id AND o_c_id = c_id \
              ORDER BY o_c_id, o_d_id, o_w_id, o_id DESC"


#define SQLCUR2 "SELECT c_balance, c_first, c_middle, c_last, \
              o_id, o_entry_d, o_carrier_id, o_ol_cnt,c_id \
              FROM cust, ordr \
              WHERE c_id = :c_id AND c_d_id = :d_id AND c_w_id = :w_id \
              AND o_d_id = c_d_id AND o_w_id = c_w_id AND o_c_id = c_id \
              ORDER BY o_c_id, o_d_id, o_w_id, o_id DESC"


#define SQLCUR3 "SELECT ol_i_id,ol_supply_w_id,ol_quantity,ol_amount, \
              ol_delivery_d\
              FROM ordl \
              WHERE ol_d_id = :d_id AND ol_w_id = :w_id AND ol_o_id = :o_id"

#define SQLCUR4 "SELECT count(c_last) FROM cust \
              WHERE c_d_id = :d_id AND c_w_id = :w_id AND c_last = :c_last "
#endif



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

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

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

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

```
  OCIStmt *curo0;
  OCIBind *w_id_bp0;
  OCIBind *d_id_bp0;
  OCIBind *c_id_bp;
  OCIBind *c_last_bp;
#ifdef PLSQLORD
  OCIBind *byln_bp;
  OCIBind *c_first_bp;
  OCIBind *c_middle_bp;
  OCIBind *c_balance_bp;
  OCIBind *o_entry_d_bp;
  OCIBind *o_cr_id_bp;
  OCIBind *o_ol_cnt_bp;
  OCIBind *ol_i_id_bp;
  OCIBind *ol_supply_w_id_bp;
  OCIBind *ol_quantity_bp;
  OCIBind *ol_amount_bp;
  OCIBind *ol_d_base_bp;
  ub4 ol_i_id_cnt;
  ub4 ol_sup_cnt;
  ub4 ol_qty_cnt;
  ub4 ol_amt_cnt;
  ub4 ol_del_d_cnt;
#else
  OCIStmt *curo1;
  OCIStmt *curo2;
  OCIStmt *curo3;
  OCIStmt *curo4;
  OCIBind *w_id_bp2;
  OCIBind *w_id_bp3;
  OCIBind *w_id_bp4;
  OCIBind *d_id_bp2;
  OCIBind *d_id_bp3;
  OCIBind *d_id_bp4;
  OCIBind *c_last_bp4;
  OCIBind *o_id_bp;
  OCIBind *c_rowid_bp;
  OCIDefine *c_rowid_dp;
  OCIDefine *c_last_dp;
  OCIDefine *c_last_dp1;
  OCIDefine *c_id_dp;
  OCIDefine *c_id_dp1;
  OCIDefine *c_first_dp1;
  OCIDefine *c_first_dp2;
  OCIDefine *c_middle_dp1;
  OCIDefine *c_middle_dp2;
  OCIDefine *c_balance_dp1;
  OCIDefine *c_balance_dp2;
  OCIDefine *o_id_dp1;
  OCIDefine *o_id_dp2;
  OCIDefine *o_entry_d_dp1;
  OCIDefine *o_entry_d_dp2;
  OCIDefine *o_cr_id_dp1;
  OCIDefine *o_cr_id_dp2;
  OCIDefine *o_ol_cnt_dp1;
  OCIDefine *o_ol_cnt_dp2;
  OCIDefine *ol_d_d_dp;
  OCIDefine *ol_i_id_dp;
  OCIDefine *ol_supply_w_id_dp;
  OCIDefine *ol_quantity_dp;
  OCIDefine *ol_amount_dp;
  OCIDefine *ol_d_base_dp;
  OCIDefine *c_count_dp;
  OCIRowid *c_rowid_ptr[100];
  OCIRowid *middle_cust;

  int cs;
  int cust_idx;
  int norow;
  int rcount;
  int somerows;
#endif
};

typedef struct ordctx ordctx;

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

ordctx *octx;

defctx cbctx;

#ifndef PLSQLORD
sb4 rid_data(dvoid *ctxp, OCIDefine *dp, ub4 iter,
             dvoid **bufpp, ub4 **alenp, ub1 *piecep,
             dvoid **indpp, ub2 **rcodepp)
{
  ub4 i;
  if (((defctx*)ctxp)->reexec)/* if this is the second execute - use entry 0 */
  {
    i = 0;
    ((defctx*)ctxp)->count--;   /* count down */
  }
  else
    i = iter;
  *bufpp = octx->c_rowid_ptr[i];
  *indpp= &octx->c_rowid_ind[i];
  *alenp= &octx->c_rowid_len[i];
  *rcodepp = &octx->c_rowid_rcode[i];
  *piecep =OCI_ONE_PIECE;
  return (OCI_CONTINUE);
}
#endif

tkvcoinit ()

{

  int i;
  text stmbuf[SQL_BUF_SIZE];

  octx = (ordctx *) malloc (sizeof(ordctx));
  memset(octx,(char)0,sizeof(ordctx));
#ifndef PLSQLORD
  octx->cs = 1;
  octx->norow = 0;
  octx->somerows = 10;
/* get the rowid handles */
  for(i=0;i<100;i++) {
   OCIERROR(errhp, OCIDescriptorAlloc(tpcenv,(dvoid**)&octx->c_rowid_ptr[i],
     OCI_DTYPE_ROWID,0,(dvoid**)0));
  }
#endif


  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx->curo0,OCI_HTYPE_STMT,0,(dvoid**)0));
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx->curo0,OCI_HTYPE_STMT,0,(dvoid**)0));
#ifndef PLSQLORD
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx->curo1,OCI_HTYPE_STMT,0,(dvoid**)0));
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx->curo2,OCI_HTYPE_STMT,0,(dvoid**)0));
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx->curo3,OCI_HTYPE_STMT,0,(dvoid**)0));
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx->curo4,OCI_HTYPE_STMT,0,(dvoid**)0));
#endif

#ifdef PLSQLORD
  sprintf((char *) stmbuf, SQLTXT);
  OCIERROR(errhp,
    OCIStmtPrepare(octx->curo0,errhp,stmbuf,strlen((char *)stmbuf),
           OCI_NTV_SYNTAX,OCI_DEFAULT));
#else
/*  c_id = 0, use find customer by lastname. Get an array or rowid's back*/
  sprintf((char *) stmbuf, SQLCUR0);
  OCIERROR(errhp,
    OCIStmtPrepare(octx->curo0,errhp,stmbuf,strlen((char *)stmbuf),
                                  OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(errhp,
    OCIAttrSet(octx->curo0,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
           OCI_ATTR_PREFETCH_ROWS,errhp));
/* get order/customer info back based on rowid */
  sprintf((char *) stmbuf, SQLCUR1);
  OCIERROR(errhp,
    OCIStmtPrepare(octx->curo1,errhp,stmbuf,strlen((char *)stmbuf),
                                  OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(errhp,
    OCIAttrSet(octx->curo1,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
           OCI_ATTR_PREFETCH_ROWS,errhp));

/*  c_id == 0, use lastname to find customer */
  sprintf((char *) stmbuf, SQLCUR2);
  OCIERROR(errhp,
    OCIStmtPrepare(octx->curo2,errhp,stmbuf,strlen((char *)stmbuf),
                                  OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(errhp,
    OCIAttrSet(octx->curo2,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
           OCI_ATTR_PREFETCH_ROWS,errhp));

  sprintf((char *) stmbuf, SQLCUR3);
  OCIERROR(errhp,
    OCIStmtPrepare(octx->curo3,errhp,stmbuf,strlen((char *)stmbuf),
                                  OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(errhp,
    OCIAttrSet(octx->curo3,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
           OCI_ATTR_PREFETCH_ROWS,errhp));

  sprintf((char *) stmbuf, SQLCUR4);
  OCIERROR(errhp,
    OCIStmtPrepare(octx->curo4,errhp,stmbuf,strlen((char *)stmbuf),
                                  OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(errhp,
```

```
    OCIAttrSet(octx->curo4,OCI_HTYPE_STMT,(dvoid*)&octx->norow,0,
          OCI_ATTR_PREFETCH_ROWS,errhp));
#endif

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

    octx->ol_supply_w_id_len[i] = sizeof(int);
    octx->ol_i_id_len[i] = sizeof(int);
    octx->ol_quantity_len[i] = sizeof(int);
    octx->ol_amount_len[i] = sizeof(int);
    octx->ol_delivery_d_len[i] = sizeof(ol_d_base[0]);
  }
  octx->ol_supply_w_id_csize = NITEMS;
  octx->ol_i_id_csize = NITEMS;
  octx->ol_quantity_csize = NITEMS;
  octx->ol_amount_csize = NITEMS;
  octx->ol_delivery_d_csize = NITEMS;
  octx->ol_w_id_csize = NITEMS;
  octx->ol_o_id_csize = NITEMS;
  octx->ol_d_id_csize = NITEMS;
  octx->ol_w_id_ind = TRUE;
  octx->ol_d_id_ind = TRUE;
  octx->ol_o_id_ind = TRUE;
  octx->ol_w_id_len = sizeof(int);
  octx->ol_d_id_len = sizeof(int);
  octx->ol_o_id_len = sizeof(int);

  /* bind variables */
#ifdef PLSQLORD

  OCIBND(octx->curo0, octx->w_id_bp0, errhp,":w_id",ADR(w_id),
       SIZ(int),SQLT_INT);
  OCIBND(octx->curo0, octx->d_id_bp0, errhp,":d_id",ADR(d_id),
       SIZ(int), SQLT_INT);
  OCIBND(octx->curo0, octx->c_id_bp , errhp,":c_id",ADR(c_id),
       SIZ(c_id),SQLT_INT);
  OCIBND(octx->curo0, octx->byln_bp , errhp,":byln",ADR(bylastname),
       SIZ(int),SQLT_INT);
  OCIBND(octx->curo0, octx->c_last_bp , errhp,":c_last",c_last,
       SIZ(c_last),SQLT_STR);
  OCIBND(octx->curo0, octx->c_first_bp , errhp,":c_first",c_first,
       SIZ(c_first),SQLT_STR);
  OCIBND(octx->curo0, octx->c_middle_bp , errhp,":c_middle",c_middle,
       SIZ(c_middle),SQLT_STR);
  OCIBND(octx->curo0, octx->c_balance_bp , errhp,":c_balance",
       ADR(c_balance),SIZ (float),SQLT_FLT);
  OCIBND(octx->curo0, octx->c_id_bp , errhp,":o_id",ADR(o_id),
       SIZ(int),SQLT_INT);
  OCIBND(octx->curo0, octx->o_entry_d_bp , errhp,":o_entry_d",o_entry_d,
       SIZ(o_entry_d),SQLT_STR);
  OCIBND(octx->curo0, octx->o_cr_id_bp , errhp,":o_cr_id",ADR(o_carrier_id),
       SIZ(int), SQLT_INT);
  OCIBND(octx->curo0, octx->o_ol_cnt_bp , errhp,":o_ol_cnt",ADR(o_ol_cnt),
       SIZ(int),SQLT_INT);

  OCIBNDRAA(octx->curo0, octx->ol_i_id_bp, errhp, ":ol_i_id",
        ol_i_id,SIZ(int),SQLT_INT,
        octx->ol_i_id_ind,octx->ol_i_id_len,
        octx->ol_i_id_rcode,NITEMS,&octx->ol_i_id_cnt);
  OCIBNDRAA(octx->curo0,octx->ol_supply_w_id_bp,errhp,":ol_s_w_id",
        ol_supply_w_id,SIZ(int),SQLT_INT,
        octx->ol_supply_w_id_ind,octx->ol_supply_w_id_len,
        octx->ol_supply_w_id_rcode,NITEMS,&octx->ol_sup_cnt);
  OCIBNDRAA(octx->curo0, octx->ol_quantity_bp,errhp,":ol_quantity",
        ol_quantity,SIZ(int),SQLT_INT,
        octx->ol_quantity_ind,octx->ol_quantity_len,
        octx->ol_quantity_rcode,NITEMS,&octx->ol_qty_cnt);
  OCIBNDRAA(octx->curo0,octx->ol_amount_bp,errhp,":ol_amount",ol_amount,
        SIZ(float),SQLT_FLT,octx->ol_amount_ind,
        octx->ol_amount_len, octx->ol_amount_rcode,NITEMS,
        &octx->ol_amt_cnt);
  OCIBNDRAA(octx->curo0,octx->ol_d_base_bp,errhp,":ol_d_d",ol_d_base,
        SIZ(OCIDate),SQLT_ODT,octx->ol_delivery_d_ind,
        octx->ol_delivery_d_len, octx->ol_delivery_d_rcode,NITEMS,
        &octx->ol_del_d_cnt);

#else

  /* c_id (customer id) is not known */
  OCIBND(octx->curo0,octx->w_id_bp0,errhp,":w_id",ADR(w_id),SIZ(int),SQLT_INT);
  OCIBND(octx->curo0,octx->d_id_bp0,errhp,":d_id",ADR(d_id),SIZ(int),SQLT_INT);
  OCIBND(octx->curo0,octx->c_last_bp,errhp,":c_last",c_last,SIZ(c_last),
       SQLT_STR);
  OCIDFNDYN(octx->curo0,octx->c_rowid_dp,errhp,1,octx->c_rowid_ptr,
        SIZ(OCIRowid*), SQLT_RDD, octx->c_rowid_ind, &cbctx,rid_data);

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

  OCIDEF(octx->curo1,octx->c_id_dp,errhp,1,ADR(c_id),SIZ(int),SQLT_INT);
  OCIDEF(octx->curo1,octx->c_balance_dp1,errhp,2,ADR(c_balance),
       SIZ(double),SQLT_FLT);
  OCIDEF(octx->curo1,octx->c_first_dp1,errhp,3,c_first,SIZ(c_first)-1,
       SQLT_CHR);
  OCIDEF(octx->curo1,octx->c_middle_dp1,errhp,4,c_middle,
       SIZ(c_middle)-1,SQLT_AFC);
  OCIDEF(octx->curo1,octx->c_last_dp1,errhp,5,c_last,SIZ(c_last)-1,
       SQLT_CHR);
  OCIDEF(octx->curo1,octx->o_id_dp1,errhp,6,ADR(o_id),SIZ(int),SQLT_INT);
  OCIDEF(octx->curo1,octx->o_entry_d_dp1,errhp,7,
       &o_entry_d_base,SIZ(OCIDate),SQLT_ODT);
  OCIDEF(octx->curo1,octx->o_cr_id_dp1,errhp,8,ADR(o_carrier_id),
       SIZ(int),SQLT_INT);
  OCIDEF(octx->curo1,octx->o_ol_cnt_dp1,errhp,9,ADR(o_ol_cnt),
       SIZ(int),SQLT_INT);

/* Bind for third cursor , no-zero customer id */
  OCIBND(octx->curo2,octx->w_id_bp2,errhp,":w_id",ADR(w_id),SIZ(int),SQLT_INT);
  OCIBND(octx->curo2,octx->d_id_bp2,errhp,":d_id",ADR(d_id),SIZ(int),SQLT_INT);
  OCIBND(octx->curo2,octx->c_id_bp,errhp,":c_id",ADR(c_id),SIZ(int),SQLT_INT);
  OCIDEF(octx->curo2,octx->c_balance_dp2,errhp,1,ADR(c_balance),
       SIZ(double),SQLT_FLT);
  OCIDEF(octx->curo2,octx->c_first_dp2,errhp,2,c_first,SIZ(c_first)-1,
       SQLT_CHR);
  OCIDEF(octx->curo2,octx->c_middle_dp2,errhp,3,c_middle,
       SIZ(c_middle)-1,SQLT_AFC);
  OCIDEF(octx->curo2,octx->c_last_dp,errhp,4,c_last,SIZ(c_last)-1, SQLT_CHR);
  OCIDEF(octx->curo2,octx->o_id_dp2,errhp,5,ADR(o_id),SIZ(int),SQLT_INT);
  OCIDEF(octx->curo2,octx->o_entry_d_dp2,errhp,6,&o_entry_d_base,
      SIZ(OCIDate),SQLT_ODT);
  OCIDEF(octx->curo2, octx->o_cr_id_dp2,errhp,7,ADR(o_carrier_id),
       SIZ(int), SQLT_INT);
  OCIDEF(octx->curo2,octx->o_ol_cnt_dp2,errhp,8,ADR(o_ol_cnt),
       SIZ(int),SQLT_INT);
  OCIDEF(octx->curo2,octx->c_id_dp1,errhp,9,ADR(c_id),SIZ(int),SQLT_INT);

/* Bind for last cursor */

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

  OCIDFNRA(octx->curo3, octx->ol_i_id_dp, errhp, 1, ol_i_id,SIZ(int),SQLT_INT,
          octx->ol_i_id_ind,octx->ol_i_id_len, octx->ol_i_id_rcode);
  OCIDFNRA(octx->curo3,octx->ol_supply_w_id_dp,errhp,2, ol_supply_w_id,
        SIZ(int),SQLT_INT, octx->ol_supply_w_id_ind,
        octx->ol_supply_w_id_len, octx->ol_supply_w_id_rcode);
  OCIDFNRA(octx->curo3, octx->ol_quantity_dp,errhp,3, ol_quantity,SIZ(int),
        SQLT_INT, octx->ol_quantity_ind,octx->ol_quantity_len,
          octx->ol_quantity_rcode);
  OCIDFNRA(octx->curo3,octx->ol_amount_dp,errhp,4,ol_amount, SIZ(int),
        SQLT_INT,octx->ol_amount_ind, octx->ol_amount_len,
        octx->ol_amount_rcode);
  OCIDFNRA(octx->curo3,octx->ol_d_base_dp,errhp,5,ol_d_base,SIZ(OCIDate),
        SQLT_ODT, octx->ol_delivery_d_ind,octx->ol_delivery_d_len,
          octx->ol_delivery_d_rcode);

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


#endif
  return (0);

}



tkvco ()

{

  int i;
  int rcount;

  for (i = 0; i < NITEMS; i++) {
    octx->ol_supply_w_id_ind[i] = TRUE;
    octx->ol_i_id_ind[i] = TRUE;
    octx->ol_quantity_ind[i] = TRUE;
    octx->ol_amount_ind[i] = TRUE;
    octx->ol_delivery_d_ind[i] = TRUE;
    octx->ol_supply_w_id_len[i] = sizeof(int);
    octx->ol_i_id_len[i] = sizeof(int);
    octx->ol_quantity_len[i] = sizeof(int);
    octx->ol_amount_len[i] = sizeof(int);
    octx->ol_delivery_d_len[i] = sizeof(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;
#ifdef PLSQLORD
  octx->ol_i_id_cnt = 0;
  octx->ol_sup_cnt = 0;
  octx->ol_qty_cnt = 0;
  octx->ol_amt_cnt = 0;
  octx->ol_del_d_cnt = 0;
  OCIERROR(errhp,
    OCIStmtExecute(tpcsvc,octx->curo0,errhp,1,0,0,0,OCI_DEFAULT));
#else
retry:
  if(bylastname)
  {
   cbctx.reexec = FALSE;
   execstatus=OCIStmtExecute(tpcsvc,octx->curo0,errhp,100,0,0,0,OCI_DEFAULT);
   /* will get OCI_NO_DATA if <100 found */
   if ((execstatus != OCI_NO_DATA) && (execstatus != OCI_SUCCESS))
   {
    errcode=OCIERROR(errhp, execstatus);
    if((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
          || (errcode == SNAPSHOT_TOO_OLD))
    {
     OCITransCommit(tpcsvc,errhp,OCI_DEFAULT);
     retries++;
     goto retry;
    } else {
     return -1;
    }
   }
   if (execstatus == OCI_NO_DATA) /* there are no more rows */
   {
    /* get rowcount, find middle one */
     OCIAttrGet(octx-
>curo0,OCI_HTYPE_STMT,&rcount,NULL,OCI_ATTR_ROW_COUNT,errhp);
     if (rcount <1)
     {
      userlog("No Data Found\n");
      return (-1);
     }
     octx->cust_idx=(rcount-1)/2 ;
   }
   else
   {
    /* count the number of rows */
    execstatus=OCIStmtExecute(tpcsvc,octx->curo4,errhp,1,0,0,0,OCI_DEFAULT);
    if ((execstatus != OCI_NO_DATA) && (execstatus != OCI_SUCCESS))
    {
     if ((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
           || (errcode == SNAPSHOT_TOO_OLD))
     {
      OCITransCommit(tpcsvc,errhp,OCI_DEFAULT);
      retries++;
      goto retry;
     } else {
      return -1;
     }
    }
    if (octx->rcount+1 < 200 )
      octx->cust_idx=(octx->rcount-1)/2 ;
    else          /* */
    {
      cbctx.reexec = TRUE;
      cbctx.count = (octx->rcount+1)/2 ;
      execstatus=OCIStmtExecute(tpcsvc,octx->curo0,errhp,cbctx.count,
                        0,0,0,OCI_DEFAULT);
      /* will get OCI_NO_DATA if <100 found */
      if (cbctx.count > 0)
      {
        userlog ("did not get all rows ");
        return (-1);
      }

      if ((execstatus != OCI_NO_DATA) && (execstatus != OCI_SUCCESS))
      {
       errcode=OCIERROR(errhp, execstatus);
       if((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
            || (errcode == SNAPSHOT_TOO_OLD))
       {
        OCITransCommit(tpcsvc,errhp,OCI_DEFAULT);
        retries++;
        goto retry;
        } else {
        return -1;
        }
      }
      octx->cust_idx=0 ;
    }
  }

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

  execstatus = OCIStmtExecute(tpcsvc,octx->curo3,errhp,o_ol_cnt,0,0,0,
              OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
  if (execstatus != OCI_SUCCESS )
  {
   errcode=OCIERROR(errhp,execstatus);
   OCITransCommit(tpcsvc,errhp,OCI_DEFAULT);
   if((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
       || (errcode == SNAPSHOT_TOO_OLD))
   {
    retries++;
    goto retry;
   }
   else
   {
    return -1;
   }

  }
#endif
  /* clean up and convert the delivery dates */
   for (i = 0; i < o_ol_cnt; i++)
   {
    if (octx->ol_delivery_d_ind[i] ==-1) /* null date in field */
                strncpy((char*)ol_delivery_d[i],"01-01-1811",10);
    else
    {
     ol_del_len[i]=sizeof(ol_delivery_d[i]);
     OCIERROR(errhp,OCIDateToText(errhp,&ol_d_base[i],
       (text*)SHORTDATE,strlen(SHORTDATE),(text*)0,0,&ol_del_len[i],ol_delivery_d[i]));
    }
/*
                cvtdmy(ol_d_base[i],ol_delivery_d[i]);
*/
   }

  return (0);

}



void tkvcodone ()

{

  if (octx)
    free (octx);

}
```

client/oracle/plsto.c

```
#ifdef RCSID
static char *RCSid =
  "$Header: plsto.c 7010000.3 95/02/14 12:48:03 plai Generic<base> $ Copyr (c) 1994 Oracle";
#endif /* RCSID */
```

```
/*=============================================================+
|      Copyright (c) 1994  Oracle Corp, Redwood Shores, CA     |
|                OPEN SYSTEMS PERFORMANCE GROUP                |
|                   All Rights Reserved                       |
 +=============================================================+
| FILENAME
|   plsto.c
| DESCRIPTION
|   OCI version of STOCK LEVEL transaction in TPC-C benchmark.
 +=============================================================*/

#include "ora_tpcc.h"
#include "tpccflags.h"

#ifdef PL SQLSTO
#define SQLTXT "BEGIN stocklevel.getstocklevel (:w_id, :d_id, :threshold, \
  :low_stock); END;"
#else
#define SQLTXT "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)"
/* query using functional index */
/*
#define SQLTXT "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_o_id BETWEEN (d_next_o_id - 20) AND (d_next_o_id - 1) AND \
               decode(SIGN(s_quantity -21) , -1, s_w_id*100000 + s_i_id, NULL) \
                                    = ol_w_id*100000 + ol_i_id  AND \
               s_quantity < :threshold;"

*/

#endif

struct stoctx {
  OCIStmt *curs;
  OCIBind *w_id_bp;
  OCIBind *d_id_bp;
  OCIBind *threshold_bp;
#ifdef PLSQLSTO
  OCIBind *low_stock_bp;
#else
  OCIDefine *low_stock_bp;
#endif
  int norow;
};

typedef struct stoctx stoctx;

stoctx *sctx;


tkvcsinit ()

{

  text stmbuf[SQL_BUF_SIZE];
  sctx = (stoctx *)malloc(sizeof(stoctx));
  memset(sctx,(char)0,sizeof(stoctx));

  sctx->norow=0;

  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&sctx->curs,OCI_HTYPE_STMT,0,(dvoid**)0));
  sprintf ((char *) stmbuf, SQLTXT);
  OCIERROR(errhp,OCIStmtPrepare(sctx->curs,errhp,stmbuf,strlen((char *)stmbuf),
                                OCI_NTV_SYNTAX,OCI_DEFAULT));
#ifndef PLSQLSTO
  OCIERROR(errhp,
    OCIAttrSet(sctx->curs,OCI_HTYPE_STMT,(dvoid*)&sctx->norow,0,
          OCI_ATTR_PREFETCH_ROWS,errhp));
#endif

  /* bind variables */

  OCIBND(sctx->curs,sctx->w_id_bp,errhp, ":w_id", ADR(w_id),sizeof(int),
       SQLT_INT);
  OCIBND(sctx->curs,sctx->d_id_bp,errhp, ":d_id", ADR(d_id),sizeof(int),
    SQLT_INT);
  OCIBND(sctx->curs,sctx->threshold_bp,errhp, ":threshold", ADR(threshold),
    sizeof(int),SQLT_INT);
#ifdef PLSQLSTO
  OCIBND(sctx->curs,sctx->low_stock_bp,errhp,":low_stock" , ADR(low_stock),
    sizeof(int), SQLT_INT);
#else
  OCIDEFINE(sctx->curs,sctx->low_stock_bp,errhp, 1, ADR(low_stock),
    sizeof(int), SQLT_INT);
#endif
  return (0);

}



tkvcs ()

{

retry:
   execstatus= OCIStmtExecute(tpcsvc,sctx->curs,errhp,1,0,0,0,
                        OCI_COMMIT_ON_SUCCESS | OCI_DEFAULT);
   if (execstatus != OCI_SUCCESS)
   {
    errcode=OCIERROR(errhp,execstatus);
    OCITransCommit(tpcsvc,errhp,OCI_DEFAULT);
    if((errcode == NOT_SERIALIZABLE) || (errcode == RECOVERR)
       || (errcode == SNAPSHOT_TOO_OLD))
    {
      retries++;
      goto retry;
    } else {
      return -1;
    }
   }

 return (0);
}



void tkvcsdone ()

{
 if(sctx) free(sctx);
}
```

client/oracle/pldel.c

```
#ifdef RCSID
static char *RCSid =
  "$Header: pldel.c 7030100.5 96/06/24 16:26:06 plai Generic<base> $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=============================================================+
|      Copyright (c) 1996  Oracle Corp, Redwood Shores, CA     |
|                OPEN SYSTEMS PERFORMANCE GROUP                |
|                   All Rights Reserved                       |
 +=============================================================+
| FILENAME
|   pldel.c
| DESCRIPTION
|   OCI version of DELIVERY transaction in TPC-C benchmark.
 +=============================================================*/

#include "ora_tpcc.h"
#ifdef TUX
#include <userlog.h>
#endif

/*
extern int userlog();
*/

#include "tpccflags.h"

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

#ifdef PLSQLDEL
#define SQLTXT "BEGIN delivery.deliver (:w_id, :carrier_id, :order_id,\
 :retry); END;"
#else
# ifdef DMLRETDEL
#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 "
# else
#define SQLTXT1A "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 1, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
WHERE no_w_id = :w_id AND no_d_id = 1 AND o_w_id = :w_id AND o_d_id = 1 AND \
  o_id = no_o_id AND rownum <= 1 UNION ALL \
"

#define SQLTXT1B "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 2, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
```

```
WHERE no_w_id = :w_id AND no_d_id = 2 AND o_w_id = :w_id AND o_d_id = 2 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
"

#define SQLTXT1C "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 3, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
WHERE no_w_id = :w_id AND no_d_id = 3 AND o_w_id = :w_id AND o_d_id = 3 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
"

#define SQLTXT1D "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 4, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
WHERE no_w_id = :w_id AND no_d_id = 4 AND o_w_id = :w_id AND o_d_id = 4 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
"

#define SQLTXT1E "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 5, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
WHERE no_w_id = :w_id AND no_d_id = 5 AND o_w_id = :w_id AND o_d_id = 5 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
"

#define SQLTXT1F "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 6, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
WHERE no_w_id = :w_id AND no_d_id = 6 AND o_w_id = :w_id AND o_d_id = 6 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
"

#define SQLTXT1G "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 7, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
WHERE no_w_id = :w_id AND no_d_id = 7 AND o_w_id = :w_id AND o_d_id = 7 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
"

#define SQLTXT1H "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 8, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
WHERE no_w_id = :w_id AND no_d_id = 8 AND o_w_id = :w_id AND o_d_id = 8 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
"

#define SQLTXT1I "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 9, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
WHERE no_w_id = :w_id AND no_d_id = 9 AND o_w_id = :w_id AND o_d_id = 9 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
"

#define SQLTXT1J "\
SELECT /*+ USE_NL(nord ordr) ORDERED */ 10, no_o_id, nord.rowid, o_c_id, ordr.rowid \
FROM nord, ordr \
WHERE no_w_id = :w_id AND no_d_id = 10 AND o_w_id = :w_id AND o_d_id = 10 AND \
   o_id = no_o_id AND rownum <= 1"

#define SQLTXT2 "DELETE FROM nord WHERE rowid = :no_rowid"
#endif

#ifdef DMLRETDEL
#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"
#else
#define SQLTXT3 "UPDATE ordr SET o_carrier_id = :carrier_id \
   WHERE rowid = :o_rowid"
#endif

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

#define SQLTXT5A "\
SELECT :d_id1, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id1 AND ol_o_id = :o_id1 UNION ALL \
SELECT :d_id2, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id2 AND ol_o_id = :o_id2 UNION ALL \
"

#define SQLTXT5B "\
SELECT :d_id3, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id3 AND ol_o_id = :o_id3 UNION ALL \
SELECT :d_id4, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id4 AND ol_o_id = :o_id4 UNION ALL \
"

#define SQLTXT5C "\
SELECT :d_id5, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id5 AND ol_o_id = :o_id5 UNION ALL \
SELECT :d_id6, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id6 AND ol_o_id = :o_id6 UNION ALL \
"

#define SQLTXT5D "\
SELECT :d_id7, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id7 AND ol_o_id = :o_id7 UNION ALL \
SELECT :d_id8, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id8 AND ol_o_id = :o_id8 UNION ALL \
"

#define SQLTXT5E "\
SELECT :d_id9, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id9 AND ol_o_id = :o_id9 UNION ALL \
SELECT :d_id10, SUM(ol_amount) FROM ordl WHERE ol_w_id = :w_id AND \
 ol_d_id = :d_id10 AND ol_o_id = :o_id10"

#endif
#endif /* PLSQLDEL */

#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"

#define NDISTS 10
#define ROWIDLEN 20


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

#ifdef DMLRETDEL
   ub4 del_o_id_len[NDISTS];
   ub4 c_id_len[NDISTS];
   int oid_ctx;
   int cid_ctx;
   OCIBind *olamt_bp;
#else
   ub2 del_o_id_len[NDISTS];
   ub2 c_id_len[NDISTS];
#endif

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

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

   int del_o_id[NDISTS];
   int cons[NDISTS];
   int w_id[NDISTS];
   int d_id[NDISTS];
   int c_id[NDISTS];
   int carrier_id[NDISTS];
   int amt[NDISTS];
   ub4 del_o_id_rcnt;
   int retry;
   OCIRowid *no_rowid_ptr[NDISTS];
   OCIRowid *o_rowid_ptr[NDISTS];
   OCIDate del_date[NDISTS];
```

```
#if defined(ISO) || defined(ISO5) || defined( ISO6) || defined(ISO8)
  char inum[10];
#endif
  OCIStmt *curd0;
  OCIStmt *curd1;
  OCIStmt *curd2;
  OCIStmt *curd3;
  OCIStmt *curd4;
  OCIStmt *curd5;
  OCIStmt *curd6;
  OCIStmt *curdtest;

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

  int norow;
};

typedef struct delctx delctx;

delctx *dctx;

#ifdef DMLRETDEL
struct amtctx {
 int ol_amt[NDISTS][NITEMS];
 sb2 ol_amt_ind[NDISTS][NITEMS];
 ub4 ol_amt_len[NDISTS][NITEMS];
 ub2 ol_amt_rcode[NDISTS][NITEMS];
 int ol_cnt[NDISTS];
};
typedef struct amtctx amtctx;
amtctx *actx;

#endif


#ifdef DMLRETDEL
extern sb4 no_data();

sb4 TPC_oid_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
            dvoid **bufpp, ub4 **alenp, ub1 *piecep,
            dvoid **indpp, ub2 **rcodepp)
{
 *bufpp = &dctx->del_o_id[iter];
 *indpp= &dctx->del_o_id_ind[iter];
 dctx->del_o_id_len[iter]=sizeof(dctx->del_o_id[0]);
 *alenp= &dctx->del_o_id_len[iter];
 *rcodepp = &dctx->del_o_id_rcode[iter];
 *piecep =OCI_ONE_PIECE;
 return (OCI_CONTINUE);
}
sb4 cid_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
            dvoid **bufpp, ub4 **alenp, ub1 *piecep,
            dvoid **indpp, ub2 **rcodepp)
{
 *bufpp = &dctx->c_id[iter];
 *indpp= &dctx->c_id_ind[iter];
 dctx->c_id_len[iter]=sizeof(dctx->c_id[0]);
 *alenp= &dctx->c_id_len[iter];
 *rcodepp = &dctx->c_id_rcode[iter];
 *piecep =OCI_ONE_PIECE;
 return (OCI_CONTINUE);
}


sb4 amt_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
            dvoid **bufpp, ub4 **alenp, ub1 *piecep,
            dvoid **indpp, ub2 **rcodepp)
{
 amtctx *actx;
 actx =(amtctx*)ctxp;
 actx->ol_cnt[iter]= actx->ol_cnt[iter]+1;
 *bufpp = &actx->ol_amt[iter][index];
 *indpp= &actx->ol_amt_ind[iter][index];
 actx->ol_amt_len[iter][index]=sizeof(actx->ol_amt[0][0]);
 *alenp= &actx->ol_amt_len[iter][index];
 *rcodepp = &actx->ol_amt_rcode[iter][index];
 *piecep =OCI_ONE_PIECE;
 return (OCI_CONTINUE);
}

#endif


tkvcdinit ()

{

# ifndef DMLRETDEL
  int i,j;
  char bstr1[10];
  char bstr2[10];
# endif /* !DMLRETDEL */
  text stmbuf[SQL_BUF_SIZE];

  dctx = (delctx *) malloc (sizeof(delctx));
  memset(dctx,(char)0,sizeof(delctx));
  dctx->norow = 0;
#ifdef DMLRETDEL
  actx = (amtctx *) malloc (sizeof(amtctx));
  memset(actx,(char)0,sizeof(amtctx));
#else
  for(i=0;i<NDISTS;i++) {
   OCIERROR(errhp, OCIDescriptorAlloc(tpcenv,(dvoid**)&dctx->o_rowid_ptr[i],
         OCI_DTYPE_ROWID,0,(dvoid**)0));
   OCIERROR(errhp, OCIDescriptorAlloc(tpcenv,(dvoid**)&dctx->no_rowid_ptr[i],
         OCI_DTYPE_ROWID,0,(dvoid**)0));
  }
#endif

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

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

/* If PLSQLDEL and ISO? are both defined, then they both try to use
  curd0! This could cause a problem. Will try to fix later - VMM 12/30/97 */

#ifdef PLSQLDEL
  OCIHandleAlloc(tpcenv, (dvoid **)(&dctx->curd0), OCI_HTYPE_STMT,
            0, (dvoid**)0);
  sprintf ((char *) stmbuf, SQLTXT);
  OCIStmtPrepare(dctx->curd0, errhp, stmbuf, strlen((char *)stmbuf),
            OCI_NTV_SYNTAX, OCI_DEFAULT);
  OCIBND(dctx->curd0, dctx->w_id_bp , errhp,":w_id",ADR(w_id),SIZ(int),
       SQLT_INT);
  OCIBND(dctx->curd0, dctx->carrier_id_bp , errhp,":carrier_id",
       ADR(dctx->carrier_id), SIZ(int), SQLT_INT);

  OCIBNDRAA(dctx->curd0, dctx->o_id_bp, errhp, ":order_id",
         dctx->del_o_id,SIZ(int),SQLT_INT, dctx->del_o_id_ind,
         dctx->del_o_id_len,dctx->del_o_id_rcode,NDISTS,
         &dctx->del_o_id_rcnt);
  OCIBND(dctx->curd0, dctx->retry_bp , errhp,":retry",ADR(dctx->retry),
       SIZ(int),SQLT_INT);
#else
# ifdef DMLRETDEL
  OCIHandleAlloc(tpcenv, (dvoid **)(&dctx->curd1), OCI_HTYPE_STMT, 0, (dvoid**)0);
  sprintf ((char *) stmbuf, "%s",    SQLTXT1);
  OCIStmtPrepare(dctx->curd1, errhp, stmbuf, strlen((char *)stmbuf),OCI_NTV_SYNTAX,
OCI_DEFAULT);

  OCIBND(dctx->curd1, dctx->w_id_bp,errhp,":w_id",dctx->w_id,SIZ(int),
       SQLT_INT);
  OCIBNDRA(dctx->curd1, dctx->d_id_bp,errhp,":d_id",dctx->d_id,SIZ(int),
       SQLT_INT,NULL,NULL,NULL);

  OCIBNDRAD(dctx->curd1, dctx->del_o_id_bp, errhp, ":o_id",
         SIZ(int),SQLT_INT,NULL,
         &dctx->oid_ctx,no_data,TPC_oid_data);
# else

  OCIHandleAlloc(tpcenv, (dvoid **)(&dctx->curd1), OCI_HTYPE_STMT, 0, (dvoid**)0);
  sprintf ((char *) stmbuf, "%s%s%s%s%s%s%s%s%s%s",    SQLTXT1A,

                SQLTXT1B,
```

```
                SQLTXT1C,

                SQLTXT1D,

                SQLTXT1E,

                SQLTXT1F,

                SQLTXT1G,

                SQLTXT1H,

                SQLTXT1I,

                SQLTXT1J
                );
  OCIStmtPrepare(dctx->curd1, errhp, stmbuf, strlen((char *)stmbuf),OCI_NTV_SYNTAX,
OCI_DEFAULT);

  OCIERROR(errhp,
    OCIAttrSet(dctx->curd1,OCI_HTYPE_STMT,(dvoid*)&dctx->norow,0,
          OCI_ATTR_PREFETCH_ROWS,errhp));



  /* bind variables */

  OCIBND(dctx->curd1, dctx->w_id_bp,errhp,":w_id",ADR(w_id),SIZ(int),SQLT_INT);

  OCIDFNRA(dctx->curd1, dctx->d_id_dp,errhp,1,dctx->d_id,SIZ(int),
       SQLT_INT, dctx->d_id_ind,dctx->d_id_len,dctx->d_id_rcode);
  OCIDFNRA(dctx->curd1, dctx->del_o_id_dp,errhp,2,dctx->del_o_id,
       SIZ(int), SQLT_INT,dctx->del_o_id_ind,
       dctx->del_o_id_len, dctx->del_o_id_rcode);
  OCIDFNRA(dctx->curd1, dctx->no_rowid_dp,errhp,3,dctx->no_rowid_ptr,
       SIZ(OCIRowid *), SQLT_RDD,dctx->no_rowid_ind,
       dctx->no_rowid_len, dctx->no_rowid_rcode);
  OCIDFNRA(dctx->curd1, dctx->c_id_dp,errhp,4,dctx->c_id,SIZ(dctx->c_id[0]),
       SQLT_INT, dctx->c_id_ind,dctx->c_id_len,dctx->c_id_rcode);
  OCIDFNRA(dctx->curd1, dctx->o_rowid_dp,errhp,5,dctx->o_rowid_ptr,
       SIZ(OCIRowid *), SQLT_RDD,dctx->o_rowid_ind,
       dctx->o_rowid_len, dctx->o_rowid_rcode);

  /* open second cursor */

  OCIHandleAlloc(tpcenv, (dvoid **)(&dctx->curd2), OCI_HTYPE_STMT, 0, (dvoid**)0);
  sprintf ((char *) stm buf, SQLTXT2);
  OCIStmtPrepare(dctx->curd2, errhp, stmbuf, strlen((char *)stmbuf),
                                OCI_NTV_SYNTAX, OCI_DEFAULT);

  /* bind variables */
  OCIBNDRA(dctx->curd2, dctx->no_rowid_bp,errhp,":no_rowid",&(dctx->no_rowid_ptr[0]),
               SIZ(dctx->no_rowid_ptr[0]),SQLT_RDD,dctx->no_rowid_ind,
               dctx->no_rowid_len,dctx->no_rowid_rcode);

# endif /*DMLRETDEL*/

  /* open third cursor */

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


  /* bind variables */

  OCIBNDRA(dctx->curd3, dctx->carrier_id_bp,errhp,":carrier_id",dctx->carrier_id,
       SIZ(dctx->carrier_id[0]),SQLT_INT,dctx->carrier_id_ind,
       dctx->carrier_id_len,dctx->carrier_id_rcode);

# ifdef DMLRETDEL
  OCIBNDRA(dctx->curd3, dctx->w_id_bp3, errhp, ":w_id", dctx->w_id,SIZ(int),
       SQLT_INT, NULL, NULL, NULL);
  OCIBNDRA(dctx->curd3, dctx->d_id_bp3, errhp, ":d_id", dctx->d_id,SIZ(int),
       SQLT_INT,NULL, NULL, NULL);
  OCIBNDRA(dctx->curd3, dctx->del_o_id_bp3, errhp, ":o_id", dctx->del_o_id,
       SIZ(int), SQLT_INT,NULL,NULL,NULL);
  OCIBNDRAD(dctx->curd3, dctx->c_id_bp3, errhp, ":o_c_id", SIZ(int),
       SQLT_INT,NULL,&dctx->cid_ctx,no_data, cid_data);
# else

  OCIBNDRA(dctx->curd3, dctx->o_rowid_bp,errhp,":o_rowid",&(dctx->o_rowid_ptr[0]),
               SIZ(dctx->o_rowid_ptr[0]),SQLT_RDD,dctx->o_rowid_ind,
               dctx->o_rowid_ptr_len,dctx->o_rowid_rcode);
#endif

  /* open fourth cursor */

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

  /* bind variables */

  OCIBND(dctx->curd4, dctx->w_id_bp4,errhp,":w_id",dctx->w_id,
       SIZ(int), SQLT_INT);
  OCIBND(dctx->curd4, dctx->d_id_bp4,errhp,":d_id",dctx->d_id,
       SIZ(int), SQLT_INT);
  OCIBND(dctx->curd4, dctx->o_id_bp,errhp,":o_id",dctx->del_o_id,
       SIZ(int),SQLT_INT);
  OCIBND(dctx->curd4, dctx->cr_date_bp,errhp,":cr_date", dctx->del_date,
       SIZ(OCIDate), SQLT_ODT);
# ifdef DMLRETDEL
  OCIBNDRAD(dctx->curd4, dctx->olamt_bp, errhp, ":ol_amount",
       SIZ(int), SQLT_INT,NULL, actx,no_data,amt_data);
# else



  /* open fifth cursor */

  OCIHandleAlloc(tpcenv, (dvoid **)(&dctx->curd5), OCI_HTYPE_STMT, 0, (dvoid**)0);
  sprintf ((char *) stmbuf, "%s%s%s%s%s", SQLTXT5A,

                SQLTXT5B,

                SQLTXT5C,

                SQLTXT5D,

                SQLTXT5E
  );
  OCIStmtPrepare(dctx->curd5, errhp, stmbuf, strlen((char *)stmbuf),
                                OCI_NTV_SYNTAX, OCI_DEFAULT);

  OCIERROR(errhp,
    OCIAttrSet(dctx->curd5,OCI_HTYPE_STMT,(dvoid*)&dctx->norow,0,
          OCI_ATTR_PREFETCH_ROWS,errhp));

  /* bind variables */

  OCIBND(dctx->curd5,dctx->w_id_bp,errhp,":w_id",ADR(w_id),SIZ(w_id),SQLT_INT);
  for (i = 0; i < NDISTS; i++) {
    sprintf (bstr1, ":d_id%d", i + 1);
    sprintf (bstr2, ":o_id%d", i + 1);
    OCIBNDRA(dctx->curd5,dctx->bstr1_bp[i],errhp,bstr1,ADR(dctx->d_id[i]),
                SIZ(dctx->d_id[0]),SQLT_INT, &(dctx->d_id_ind[i]),
                &(dctx->d_id_len[i]),&(dctx->d_id_rcode[i]));
    OCIBNDRA(dctx->curd5,dctx->bstr2_bp[i],errhp,bstr2,ADR(dctx->del_o_id[i]),
                SIZ(dctx->del_o_id[0]),SQLT_INT, &(dctx->del_o_id_ind[i]),
                &(dctx->del_o_id_len[i]),&(dctx->del_o_id_rcode[i]));
  }

  OCIDFNRA(dctx->curd5,dctx->cons_dp,errhp,1,dctx->cons,SIZ(dctx->cons[0]),SQLT_INT,
       dctx->cons_ind,dctx->cons_len,dctx->cons_rcode);
  OCIDFNRA(dctx->curd5,dctx->amt_dp,errhp,2,dctx->amt,SIZ(dctx->amt[0]),SQLT_INT,
       dctx->amt_ind,dctx->amt_len,dctx->amt_rcode);
#endif
  /* open sixth cursor */

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

  /* bind variables */


  OCIBND(dctx->curd6,dctx->amt_bp,errhp,":amt",dctx->amt,SIZ(int),
       SQLT_INT);
  OCIBND(dctx->curd6,dctx->w_id_bp6,errhp,":w_id",dctx->w_id,SIZ(int),
       SQLT_INT);
  OCIBND(dctx->curd6,dctx->d_id_bp6,errhp,":d_id",dctx->d_id,SIZ(int),
       SQLT_INT);
  OCIBND(dctx->curd6,dctx->c_id_bp,errhp,":c_id",dctx->c_id,SIZ(int),
       SQLT_INT);
#endif
  return (0);

}


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

```
tkvcd ()

{

  int i, j;
  int rpc,rcount,count;
  int invalid;
# ifndef DMLRETDEL
  int tmp_id,v;
  int tmp_amt;
# endif /*  !DMLRETDEL */

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

  OCIStmtExecute(tpcsvc,dctx->curd0,errhp,1,0,0,0,OCI_DEFAULT);
  sysdate (sdate);
  printf ("Delivery started at %s on node %s\n", sdate, dctx->inum);
#endif
#ifdef PLSQLDEL
  for (i = 0; i < NDISTS; i++)
  {
    dctx->del_o_id_ind[i] = TRUE;
    dctx->del_o_id_len[i] = sizeof(int);
  }

  OCIERROR(errhp,
    OCIStmtExecute(tpcsvc,dctx->curd0,errhp,1,0,0,0,OCI_DEFAULT));

  for (i = 0; i < NDISTS; i++)
  {
    del_o_id[i] = 0;
    if (dctx->del_o_id_ind[i] == 0)
    {
     del_o_id[i] = dctx->del_o_id[i];
    }
  }
#else

retry:

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

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

  invalid = 0;

  /* initialization for array operations */

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

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

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

#ifdef DMLRETDEL /* VMM 1/13/98 */
  memset(actx,(char)0,sizeof(amtctx));
#endif /* DMLRETDEL */
  /* array select from new_order and orders tables */

  execstatus=OCIStmtExecute(tpcsvc,dctx->curd1,errhp,NDISTS,0,0,0,OCI_DEFAULT);
  if((execstatus != OCI_SUCCESS) && (execstatus != OCI_NO_DATA)) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      retries++;
      goto retry;
    } else if (errcode == RECOVERR) {
      retries++;
      goto retry;
    } else if (errcode == SNAPSHOT_TOO_OLD) {
      retries++;
      goto retry;
    } else {
      return -1;
    }
  }
  /* mark districts with no new order */
  OCIAttrGet(dctx->curd1,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,errhp);
  rpc = rcount;
#ifdef DMLRETDEL   /* we have to compress the array here */
  if (rcount != NDISTS )
  {
   int j = 0;
   for (i=0;i < NDISTS; i++)
   {
    if (dctx->del_o_id_ind[j] == 0) /* there is data here */
     j++;
    else
     shiftdata(j);
   }
  }
#else
  invalid = NDISTS - rcount;
  for (i = rpc; i < NDISTS; i++) {
    dctx->del_o_id_ind[i] = NA;
    dctx->w_id_ind[i] = NA;
    dctx->d_id_ind[i] = NA;
    dctx->c_id_ind[i] = NA;
    dctx->carrier_id_ind[i] = NA;
    dctx->no_rowid_ind[i] = NA;
    dctx->o_rowid_ind[i] = NA;
  }
#endif

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

#ifndef DMLRETDEL
  /* array delete of new_order table */
  execstatus=OCIStmtExecute(tpcsvc,dctx->curd2,errhp,rpc,0,0,0,OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      retries++;
      goto retry;
    } else if (errcode == RECOVERR) {
      retries++;
      goto retry;
    } else if (errcode == SNAPSHOT_TOO_OLD) {
      retries++;
      goto retry;
    } else {
      return -1;
    }
  }

  /* mark districts with no new order */
  OCIAttrGet(dctx->curd2,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,errhp);

  if (rcount != rpc) {
#ifdef TUX
    userlog ("Error in TPC-C server %d: %d rows selected, %d rows deleted\n",
          proc_no, rpc, dctx->curd2.rpc);
#else
    fprintf (stderr,
          "Error in TPC-C server %d: %d rows selected, %d rows deleted\n",
```

```
            proc_no, rpc, rcount);
#endif /* TUX */
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      return (DEL_ERROR);
    }
#endif /* DMLRETDEL */

    execstatus=OCIStmtExecute(tpcsvc,dctx->curd3,errhp,rpc,0,0,0,OCI_DEFAULT);
    if(execstatus != OCI_SUCCESS) {
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      errcode = OCIERROR(errhp,execstatus);
      if(errcode == NOT_SERIALIZABLE) {
        retries++;
        goto retry;
      } else if (errcode == RECOVERR) {
        retries++;
        goto retry;
      } else if (errcode == SNAPSHOT_TOO_OLD) {
        retries++;
        goto retry;
      } else {
        return -1;
      }
    }

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

    if (rcount != rpc) {
#ifdef TUX
      userlog ("Error in TPC-C server %d: %d rows selected, %d ords updated\n",
            proc_no, rpc, rcount);
#else
      fprintf (stderr,
            "Error in TPC-C server %d: %d rows selected, %d ords updated\n",
            proc_no, rpc, rcount);
#endif
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      return (-1);
    }

    /* array update of order_line table */
    execstatus=OCIStmtExecute(tpcsvc,dctx->curd4,errhp,rpc,0,0,0,OCI_DEFAULT);
    if(execstatus != OCI_SUCCESS) {
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      errcode = OCIERROR(errhp,execstatus);
      if(errcode == NOT_SERIALIZABLE) {
        retries++;
        goto retry;
      } else if (errcode == RECOVERR) {
        retries++;
        goto retry;
      } else if (errcode == SNAPSHOT_TOO_OLD) {
        retries++;
        goto retry;
      } else {
        return -1;
      }
    }
#ifdef DMLRETDEL
    OCIAttrGet(dctx-
>curd4,OCI_HTYPE_STMT,&rcount,NULL,OCI_ATTR_ROW_COUNT,errhp);
/* add up amounts */
    count=0;
    for (i=0;i<rpc;i++)
    {
      dctx->amt[i]=0;
      for (j=0;j<actx->ol_cnt[i];j++)
        if ( actx->ol_amt_rcode[i][j] == 0)
        {
          dctx->amt[i] = dctx->amt[i] + actx->ol_amt[i][j];
          count = count+1;
        }
    }
    if (rcount > rpc*NITEMS) {
      userlog ("Error in TPC-C server %d: %d ordnrs updated, %d ordl updated\n",
            proc_no, rpc, rcount);
    }
#else
    /* array select from order_line table */
    execstatus=OCIStmtExecute(tpcsvc,dctx->curd5,errhp,rpc,0,0,0,OCI_DEFAULT);
    if((execstatus != OCI_SUCCESS) && (execstatus != OCI_NO_DATA)) {
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      errcode = OCIERROR(errhp,execstatus);
      if(errcode == NOT_SERIALIZABLE) {
        retries++;
        goto retry;
      } else if (errcode == RECOVERR) {
        retries++;
        goto retry;
      } else if (errcode == SNAPSHOT_TOO_OLD) {
        retries++;
        goto retry;
      } else {
        return -1;
      }
    }

    OCIAttrGet(dctx->curd5,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,errhp);
    if (rcount != rpc) {
#ifdef TUX
      userlog ("Error in TPC-C server %d: %d rows selected, %d ordl selected\n",
            proc_no, rpc, rcount);
#else
      fprintf (stderr,
            "Error in TPC-C server %d: %d rows selected, %d ordl selected\n",
            proc_no, rpc, rcount);
#endif
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      return (-1);
    }

    /* reorder amount selected if necessary */

    for (i = 0; i < rpc; i++) {
      if (dctx->cons[i] != dctx->d_id[i]) {
#ifdef TUX
        userlog ("TPC-C server %d: reordering amount\n", proc_no);
#else
        fprintf (stderr, "TPC-C server %d: reordering amount\n", proc_no);
#endif
        for (j = i + 1; j < rpc; j++) {
          if (dctx->cons[j] == dctx->d_id[i]) {
            tmp_id = dctx->cons[i];
            dctx->cons[i] = dctx->cons[j];
            dctx->cons[j] = tmp_id;
            tmp_amt = dctx->amt[i];
            dctx->amt[i] = dctx->amt[j];
            dctx->amt[j] = tmp_amt;
            break;
          }
        }
        if (j >= rpc) {
#ifdef TUX
          userlog ("Error in TPC-C server %d: missing ordl?\n", proc_no);
#else
          fprintf (stderr,
                "Error in TPC-C server %d: missing ordl?\n", proc_no);
#endif
                OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
          return (-1);
        }
      }
    }
#endif
#if defined(ISO5) || defined(ISO6)
    printf ("d_id:amount\n");
    for (i = 0; i < rpc; i++)
      printf ("%d:%.2f ", dctx->d_id[i], (float)dctx->amt[i]/100);
    printf ("\n");
#endif

    /* array update of customer table */
#if defined(ISO5) || defined (ISO6)
    execstatus=OCIStmtExecute(tpcsvc,dctx->curd6,errhp,rpc,0,0,0,
                                                        OCI_DEFAULT);
#else
    execstatus=OCIStmtExecute(tpcsvc,dctx->curd6,errhp,rpc,0,0,0,
                        OCI_COMMIT_ON_SUCCESS | OCI_DEFAULT);
#endif

    if(execstatus != OCI_SUCCESS) {
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      errcode = OCIERROR(errhp,execstatus);
      if(errcode == NOT_SERIALIZABLE) {
        retries++;
        goto retry;
      } else if (errcode == RECOVERR) {
        retries++;
        goto retry;
      } else if (errcode == SNAPSHOT_TOO_OLD) {
        retries++;
        goto retry;
      } else {
        return -1;
      }
    }

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

    if (rcount != rpc) {
#ifdef TUX
      userlog ("Error in TPC-C server %d: %d rows selected, %d cust updated\n",
            proc_no, rpc, rcount);
#else
      fprintf (stderr,
            "Error in TPC-C server %d: %d rows selected, %d cust updated\n",
            proc_no, rpc, rcount);
#endif
      OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
      return (-1);
```

```
  }

#if defined(ISO5) || defined(ISO6)
  sysdate (sdate);
#ifdef ISO5
  printf ("Delivery sleep before commit at %s\n", sdate);
#else
  printf ("Delivery sleep before abort at %s\n", sdate);
#endif
  sleep (60);
  sysdate (sdate);
  printf ("Delivery wake up at %s\n", sdate);
#endif

#ifdef ISO6
  printf("Delivery ISO6 Rolling back.\n");
  OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
#endif

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

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

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

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

  return (0);

}



void tkvcddone ()

{

  if (dctx)
  {
#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
    OCIHandleFree((dvoid *)dctx->curd0,OCI_HTYPE_STMT);
#endif
#ifdef PLSQLDEL
    OCIHandleFree((dvoid *)dctx->curd0,OCI_HTYPE_STMT);
#else
    /* Again the above will cause a problem if both PSQLDEL and ISO are
                defined - VMM 12/30/97 */
    OCIHandleFree((dvoid *)dctx->curd1,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx->curd2,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx->curd3,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx->curd4,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx->curd5,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx->curd6,OCI_HTYPE_STMT);
#endif
    free (dctx);
  }

}
```

client/oracle/ora_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();


/* 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.hh: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];

/* 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 (void *)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));

#define OCIBNDRA(stmp,bndp,errp,sqlvar,progv,progvl,ftype,indp,alen,arcode) \
     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),0,0,OCI_DEFAULT));

#define OCIBNDRAD(stmp,bndp,errp,sqlvar,progvl,ftype,indp,ctxp,cbf_nodata,cbf_data) \
     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)),0,(progvl),(ftype), \
                 indp,0,0,0,0,OCI_DATA_AT_EXEC)); \
     ocierror(__FILE__,__LINE__,(errp), \
        OCIBindDynamic((bndp),(errp),(ctxp),(cbf_nodata),(ctxp),(cbf_data)));



#define OCIBNDR(stmp,bndp,errp,sqlvar,progv,progvl,ftype,indp,alen,arcode) \
     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),0,0,OCI_DEFAULT));

#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;
};

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
```

client/oracle/tpccflags.h

```
#define PLSQLNO
#define DMLRETDEL
```

A.4 Server Stored Procedures

ACID/blocks/new.sql

```
DECLARE   /* new order */
    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
      SELECT c_discount, c_last, c_credit
       INTO :c_discount, :c_last, :c_credit
       FROM cust
       WHERE c_id = :c_id
        AND c_d_id = :d_id
        AND c_w_id = :w_id;

      UPDATE wh_dist SET d_next_o_id = d_next_o_id + 1, d_tax=d_tax+0
       WHERE  d_id = :d_id
        AND  w_id = :w_id
    RETURNING d_tax, d_next_o_id-1, w_tax
       INTO :d_tax, :o_id, :w_tax;

      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_w_id, o_d_id, o_c_id, o_carrier_id,
                    o_ol_cnt, o_all_local,o_entry_d)
          VALUES (:o_id, :w_id, :d_id, :c_id, 11,
                :o_ol_cnt, :o_all_local, :cr_date);
      EXIT;

      EXCEPTION
        WHEN not_serializable OR deadlock OR snapshot_too_old THEN
          ROLLBACK;
          :retry := :retry + 1;
      END;
    END LOOP;
  END;
```

ACID/blocks/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 initpay.ware_name,
             :w_street_1, :w_street_2, :w_city, :w_state, :w_zip;


      SELECT rowid
      BULK COLLECT INTO initpay.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;

     initpay.c_num := sql%rowcount;
     initpay.cust_rowid := initpay.row_id((initpay.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 = initpay.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 = initpay.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 initpay.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, initpay.ware_name || '    ' || initpay.dist_name);

     EXIT;

     EXCEPTION
       WHEN not_serializable OR deadlock OR snapshot_too_old THEN
         ROLLBACK;
         :retry := :retry + 1;
       END;

    END LOOP;
  END;
```

ACID/blocks/paynz.sql

```
DECLARE /* paynz */
--      cust_rowid              ROWID;
--      dist_name               VARCHAR2(11);
--      ware_name               VARCHAR2(11);
      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 initpay.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 initpay.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;
--                              insert into dummy values (rowidtochar(initpay.cust_rowid));

--      :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, '9999.99') || ' | ')
                     || c_data , 1, 500)
          WHERE rowid = initpay.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 initpay.dist_name,:d_street_1,:d_street_2,:d_city,:d_state,
              :d_zip;
       IF SQL%NOTFOUND THEN
        raise NO_D ATA_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, initpay.ware_name || '    ' || initpay.dist_name);
--      COMMIT;
--      :h_date := to_char (:cr_date, 'DD-MM-YYYY.HH24:MI:SS');
      EXIT;

      EXCEPTION
        WHEN not_serializable OR deadlock OR snapshot_too_old THEN
          ROLLBACK;
          :retry := :retry + 1;
      END;

    END LOOP;
  END;
```

startup.c

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 10:53:27 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/

#include <signal.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <fcntl.h>
#include <errno.h>

#undef NULL
#define NULL ((void *)0)

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

   server <port> <a.out> <arg list>

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

void spawn_user();


int main(argc, argv)
   int argc;
   char **argv;
   {
   int server_fd;
   int client_fd;

   /* We don't want zombie children */
   signal(SIGCHLD, SIG_IGN);

   if (rtprio(0, 80) < 0)
      perror("Server can't run real-time");

   /* create a socket to accept new requests */
   server_fd = server_socket(argv[1]);
   if (server_fd < 0)
      syserror("Can't create a listening socket\n");


   /* repeat forever */
   for (;;)
      {
      client_fd = connect_client(server_fd);

                  spawn_user(server_fd, client_fd, argc-2, argv+2);

                  close(client_fd);
      }

   return 0;
   }



void spawn_user(s_fd, c_fd, argc, argv)
   int s_fd;
   int c_fd;
   int argc;
   char **argv;
   {
   int pid;

   pid = fork();
   if (pid < 0)
      syserror("Can't fork off child process\n");

   if (pid > 0)  return;

   /* close the accept fd */
   close(s_fd);

   /* adjust the file descriptors so socket is stdin and stdout */
   close(0); close(1);
   if (dup(c_fd) != 0)
      syserror("Couldn't dup to fd 0\n");
   if (dup(c_fd) != 1)
      syserror("Couldn't dup to fd 1\n");
   close(c_fd);

   /* execute the child process */
   execvp(argv[0], argv);

   syserror("Unable to exec %s\n", argv[0]);
   }



int connect_client(server_fd)
/*****************************************************************************
connect_client connects the clients who are waiting
*****************************************************************************/
   int server_fd;
   {
   int fd, vfd;
   struct sockaddr dummy_addr;
   int dummy_size = sizeof(dummy_addr);

   /* accept a connection to a new client.  Exit if no more */
   fd = accept(server_fd, &dummy_addr , &dummy_size);
   if (fd < 0)
      syserror("Can't accept new client\n");

   /* set the socket parameters */
   if (prepare_socket(fd) < 0)
                  syserror("Can't set socket parameters\n");

   return fd;
   }



int server_socket(service)
/*****************************************************************
server_socket creates a socket for a server with the given name
*****************************************************************/
   char service[];
   {
   int fd;
   struct sockaddr_in address;
   struct servent *server;
   int port;
   char *s;

   /* if the service is all digits, then use that as tcp port number */
   for (s=service; isdigit(*s); s++)
                  ;
```

```
  if (*s == '\0')
                port = atoi(service);

  /* otherwise, get the named service port number */
  else
                {
    server = getservbyname(service, "tcp");
    if (server == NULL)
                  error("Service %s is unknown\n", service);
    port = server->s_port;
                }

  /* create a socket */
  fd = socket(AF_INET, SOCK_STREAM, 0);
  if (fd < 0)
                syserror("Can't create a socket\n");
  if (prepare_socket(fd) < 0)
                syserror("Can't configure the socket\n");

  /* build up an internet style address */
  address.sin_family = AF_INET;
  address.sin_port = port;
  address.sin_addr.s_addr = INADDR_ANY;

  /* set up the socket to listen at the given address */
  if (bind(fd, &address, sizeof(address)) < 0)
    syserror("Can't bind the socket to address\n");
  if (listen(fd, SOMAXCONN) < 0)
    syserror("Can't listen\n");

  return fd;
  }
```

delay.c

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 10:56:52 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/
#include <sys/time.h>
#include <errno.h>
#ifndef HPUX9
#include <time.h>
#endif
#include "tpcc.h"
#include "shm.h"

delay(sec)
/*****************************************************************************
delay sleeps for the specified number of seconds.  (to closest 1/100'th second)
*****************************************************************************/
   double sec;
   {
#ifdef HPUX9
   struct timeval delay;
#else
   struct timespec delay;
#endif

   /* if no delay,  done */
   if (sec <= 0.0) return;

   /* add a portion of a clock tick to keep averages correct */
   sec += 1.0 / CLK_TCK;

   /* convert the delay to seconds and nanoseconds */
   delay.tv_sec = sec;
#ifdef HPUX9
   delay.tv_usec = (sec - delay.tv_sec) * 1000000;
#else
   delay.tv_nsec = (sec - delay.tv_sec) * 1000000000;
#endif

   /* sleep on a select call */
#ifdef HPUX9
   if (select(0, NULL, NULL, NULL, &delay) < 0) {
     syserror("delay:  select() call failed\n");
   }
#else
   if (nanosleep(&delay,NULL) == -1) {
                if (errno != EINTR) {
        syserror("delay:  nanosleep() call failed, errno = %d\n",errno);
                }
   }
#endif
}



struct timeval start_time;

initclock()
   {
   gettimeofday(&start_time, NULL);
   }


TIME getclock()
/*****************************************************************************
getclock returns the current time, expressed in seconds from start of run
*****************************************************************************/
   {
   struct timeval current;
   gettimeofday(&current, NULL);

   return elapsed_time(&current);
   }
```

random.h

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 14:02:00 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/
#ifndef TPCC_RANDOM
#define TPCC_RANDOM

#ifdef USE_DRAND48
double drand48();
#else
double randy();
#endif

extern int    MakeNumberString();
extern ID     RandomWarehouse();
extern int    MakeAlphaString();
extern void   RandomPermutation();
extern void   RandomDelay();
extern double exponential();
extern void   Randomize();
extern void   SetRandomSeed();

extern char lastNames[1000][16];
extern char customerData1[10][301];
extern char customerData2[10][201];
extern char stockData1[10][27];
extern char stockData2[10][25];
extern char historyData1[10][13];
extern char historyData2[10][13];
extern char citystreetData1[10][11];
extern char citystreetData2[10][11];
extern char firstNam eData1[10][9];
extern char firstNameData2[10][9];
extern char StockDistrict[10][25];
extern char phoneData[10][17];

/*****************************************************************************
RandomNumber selects a uniform random number from min to max inclusive
*****************************************************************************/
#ifdef USE_DRAND48
#define RandomNumber(min,max)  \
                ((int)(drand48() * ((int)(max) - (int)(min) + 1)) + (int)(min))
#else
#define RandomNumber(min,max)  \
                ((int)(randy() * ((int)(max) - (int)(min) + 1)) + (int)(min))
#endif

/*****************************************************************************
NURandomNumber selects a non-uniform random number
*****************************************************************************/
#define NURandomNumber(a, min, max, c) \
  ((RandomNumber(0, a) | RandomNumber(min, max)) + (c)) % \
                                ((max) - (min) + 1) + (min)


#define SelectCityStreetData(data) \
{ \
   strcpy(data,citystreetData1[RandomNumber(0,9)]); \
   strcat(data,citystreetData 2[RandomNumber(0,9)]); \
}

#define SelectFirstName(data) \
{ \
   strcpy(data,firstNameData1[RandomNumber(0,9)]); \
   strcat(data,firstNameData2[RandomNumber(0,9)]); \
}

#define SelectHistoryData(data) \
{ \
   strcpy(data,historyData1[RandomNumber(0,9)]); \
   strcat(data,historyData2[RandomNumber(0,9)]); \
}
```

```
#define SelectStockData(data) \
{ \
   strcpy(data,stockData1[RandomNumber(0,9)]); \
   strcat(data,stockData2[RandomNumber(0,9)]); \
}

#define SelectClientData(data) \
{ \
   strcpy(data,customerData1[RandomNumber(0,9)]); \
   strcat(data,customerData2[RandomNumber(0,9)]); \
}

#define SelectPhoneData(data) strcpy(data,phoneData[RandomNumber(0,9)])
#define SelectStockDistrict(data) strcpy(data,StockDistrict[RandomNumber(0,9)])

#define MakeZip(zip) \
{ \
  MakeNumberString(4, 4, zip); \
  zip[4] = '1'; \
  zip[5] = '1'; \
  zip[6] = '1'; \
  zip[7] = '1'; \
  zip[8] = '1'; \
  zip[9] = '\0'; \
}

#define MakeAddress(str1, str2, city, state, zip) \
{ \
   SelectCityStreetData(str1); \
   SelectCityStreetData(str2); \
   SelectCityStreetData(city); \
   MakeAlphaString(2,2,state); \
   MakeZip(zip); \
}

#define LastName(num, name) strcpy(name, lastNames[(num)])

#define Original(str) \
{ \
   int len = strlen(str); \
   if (len >= 8) { \
      int pos = RandomNumber(0,(len-8)); \
      str[pos+0] = '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'; \
   } \
}


#endif
```

results_file.c

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 14:02:01 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include "tpcc.h"



static FILE *rfile;

results_open(id)
   int id;
   {
   char fullname[128];
   char *basename;

   /* get the base file name for the deferred results */
   /*
    * Make it a directory under /tmp so at least we can set it to a
    * symbolic link in case /tmp doesn't have enough room.
    */
   basename = getenv("TPCC_RESULTS_FILE");
   if (basename == NULL)
      basename = "/tmp/TPCC_RESULTS_FILE";

   /* create the full file name */
   sprintf(fullname, "%s.%d", basename, id);

   /* open the file */
   unlink(fullname);
   rfile = fopen(fullname, "wb");
   if (rfile == NULL)
      syserror("Delivery server %d can't open file %s\n", id, fullname);

   /* allocate a larger buffer */
   }



results(t)
   delivery_trans *t;
   {
   if (fwrite(t, sizeof(*t), 1, rfile) != 1)
      syserror("Delivery server: Can't post results\n");
   fflush(rfile);
   }


results_close()
   {
   if (fclose(rfile) < 0)
      syserror("Delivery server can't close file\n");
   }
```

Appendix B Database Design

The source code for the process to define, create and populate the Oracle9i Database Enterprise Edition TPC-C database is included in this appendix.

B.1 Scripts addfile.sh

create_cust.sql

```
spool step11createcust.log;
set echo on;
drop table cust;
drop cluster custcluster including tables;
set timing on;
create cluster custcluster (
c_id    number(5,0)
,c_d_id    number(2,0)
,c_w_id    number(5,0)
 )
 single    table
 hashkeys    960000000
 hash is   (c_id * 320000 + c_w_id * 10 + c_d_id)
 size    850
 initrans    3
 pctfree    0
 storage ( buffer_pool recycle freelists 22 freelist groups 43 )
 tablespace    cust;
create table cust (
       c_id            number(5,0),
       c_d_id          number(2,0),
       c_w_id          number(5,0),
       c_discount      number,
       c_credit        char(2),
       c_last          varchar2(16),
       c_first         varchar2(16),
       c_credit_lim    number,
       c_balance       number,
       c_ytd_payment   number,
       c_payment_cnt   number,
       c_delivery_cnt  number,
       c_street_1      varchar2(20),
       c_street_2      varchar2(20),
       c_city          varchar2(20),
       c_state         char(2),
       c_zip           char(9),
       c_phone         char(16),
       c_since         date,
       c_middle        char(2),
       c_data          varchar2(500)
)
cluster custcluster (c_id
,c_d_id
,c_w_id
);
spool off;
set echo off;
exit sql.sqlcode;
```

create_hist.sql

```
spool step12createhist.log;
set echo on;
drop table hist;
set timing on;
create table hist (
       h_c_id          number,
       h_c_d_id        number,
       h_c_w_id        number,
       h_d_id          number,
       h_w_id          number,
       h_date          date,
       h_amount        number,
       h_data          varchar2(24)
)
 initrans    4
 pctfree    5
 storage ( freelists 22 freelist groups 43 )
 tablespace    hist;
spool off;
set echo off;
exit sql.sqlcode;
```

create_icust1.sql

```
spool step32createicust1.log;
set echo on;
drop index icust1;
set timing on;
create unique index icust1 on cust (c_w_id, c_d_id, c_id)
 initrans    3
 parallel    80
 pctfree    1
 storage ( freelists 22 freelist groups 43 )
 tablespace    icust1;
spool off;
set echo off;
exit sql.sqlcode;
```

create_icust2.sql

```
spool step33createicust2.log;
set echo on;
drop index icust2;
set timing on;
create unique index icust2 on cust (c_last, c_w_id, c_d_id, c_first, c_id)
 initrans    3
 parallel    80
 pctfree    1
 storage ( freelists 22 freelist groups 43 )
 tablespace    icust2;
spool off;
set echo off;
exit sql.sqlcode;
```

create_inord.sql

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step37createinord > junk 2>&1

if test $? -ne 0
then
 exit 1;
else
 exit 0;
fi
wait
```

create_iord1.sql

```
spool step35createiordr1.log;
set echo on;
drop index iordr1;
set timing on;
create unique index iordr1 on ordr (o_w_id, o_d_id, o_id)
 initrans    3
 parallel    32
 pctfree    1
 storage ( freelists 22 freelist groups 43 )
 tablespace    iord1;
spool off;
set echo off;
exit sql.sqlcode;
```

create_iord2.sql

```
spool step36createiordr2.log;
set echo on;
drop index iordr2;
set timing on;
create unique index iordr2 on ordr (o_w_id, o_d_id, o_c_id, o_id)
 initrans    4
 parallel    32
 pctfree    25
 storage ( freelists 22 freelist groups 43 )
 tablespace    iord2;
spool off;
set echo off;
exit sql.sqlcode;
```

create_istok.sql

```
spool step34createistok.log;
set echo on;
drop index istok;
set timing on;
create unique index istok on stok (s_i_id, s_w_id)
  initrans    3
  parallel    100
  pctfree     1
  storage ( freelists 22 freelist groups 43 )
  tablespace    istk;
spool off;
set echo off;
exit sql.sqlcode;
```

create_nord.sql

```
spool step14createnord.log;
set echo on;
drop table nord;
set timing on;
create table nord (
        no_w_id         number,
        no_d_id         number,
        no_o_id         number,
                constraint inord primary key (no_w_id, no_d_id, no_o_id)
)
  organization index
  initrans    4
  pctfree     5
  storage ( freelists 22 freelist groups 43 )
  tablespace    nord;
spool off;
set echo off;
exit sql.sqlcode;
```

create_ordr.sql

```
spool step13createordr.log;
set echo on;
drop table ordr;
set timing on;
create table ordr (
        o_id            number,
        o_w_id          number,
        o_d_id          number,
        o_c_id          number,
        o_carrier_id    number,
        o_ol_cnt        number,
        o_all_local     number,
        o_entry_d       date
)
  initrans    4
  pctfree     5
  storage ( freelists 22 freelist groups 43 )
  tablespace    ordr;
spool off;
set echo off;
exit sql.sqlcode;
```

create_ord1.sql

```
spool step15createordl.log;
set echo on;
drop table ordl;
set timing on;
create table ordl (
        ol_w_id         number,
        ol_d_id         number,
        ol_o_id         number,
        ol_number       number,
        ol_i_id         number,
        ol_delivery_d   date,
        ol_amount       number,
        ol_supply_w_id  number,
        ol_quantity     number,
        ol_dist_info    char(24),
                constraint iordl primary key (ol_w_id, ol_d_id, ol_o_id, ol_number)
)
  organization index
  initrans    4
  pctfree     5
  storage ( freelists 22 freelist groups 43 )
  tablespace    ordl;
spool off;
set echo off;
exit sql.sqlcode;
```

create_stok.sql

```
spool step16createstok.log;
set echo on;
drop table stok;
drop cluster stokcluster including tables;
set timing on;
create cluster stokcluster (
s_i_id    number(6,0)
,s_w_id    number(5,0)
  )
  single    table
  hashkeys    3200000000
  hash is   (s_i_id * 32000 + s_w_id)
  size    350
  initrans    3
  pctfree     0
  storage ( buffer_pool keep freelists 22 freelist groups 43 )
  tablespace    stok;
create table stok (
        s_i_id          number(6,0),
        s_w_id          number(5,0),
        s_quantity      number,
        s_ytd           number,
        s_order_cnt     number,
        s_remote_cnt    number,
        s_data          varchar2(50),
        s_dist_01       char(24),
        s_dist_02       char(24),
        s_dist_03       char(24),
        s_dist_04       char(24),
        s_dist_05       char(24),
        s_dist_06       char(24),
        s_dist_07       char(24),
        s_dist_08       char(24),
        s_dist_09       char(24),
        s_dist_10       char(24)
)
cluster stokcluster (s_i_id
,s_w_id
);
spool off;
set echo off;
exit sql.sqlcode;
```

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;
```

driver.sh

```
#!sh

STEP=1
START=0
END=0
CONTINUE=1
PROGNAME="driver.sh"
STOPFILE="stop"
TRACEFILE="log/trace.log"
JUNKFILE="junk"


SEQLPLUS=sqlplus
export SEQLPLUS
TPCCLOAD=tpccload.exe
export TPCCLOAD
ORACLE_SID=tpcc
export ORACLE_SID
BUILDDIR=${ORACLE_HOME}/bench/tpc/tpcc/scripts/build32k
export SCRIPTS
SCRIPTS=${BUILDDIR}/scripts
export SCRIPTS
SQLDIR=${BUILDDIR}/sql
export SQLDIR
DB_SIZE=32000
export DB_SIZE
PATH=$PATH:${SCRIPTS}


usage()
{
  echo ""
  echo "Usage: $PROGNAME [<startstepno> <stopstepno>] [<startstepno>] [-step <stepno>]"
  echo "       [<startstepno> <stopstepno>] - allows user to run a specified"
  echo "                    range of steps."
  echo "       [<startstepno>]              - runs from step number <start> till"
  echo "                    the end of the script."
  echo "       [-step <stepno>]             - runs only step number <stepno> and"
  echo "                    then stops."
  echo ""
  echo "             STEP    FUNCTION"
  echo "-------------------------------------------------------------------"
  echo "               1     Create Database."
  echo "               2     Create Rollback Segments for Build."
  echo "               3     Shutdown Database."
  echo "               4     Startup Database for Build"
  echo "               5     Create User TPCC."
  echo "               6     Create Tablespaces."
  echo "               7     Assign Temporary Tablespace to user TPCC."
  echo "               8     Create Data Dictonary Views."
  echo "               9     Create Warehouse."
  echo "               10    Create District."
  echo "               11    Create Customer."
  echo "               12    Create History."
  echo "               13    Create Order."
  echo "               14    Create Neworder."
  echo "               15    Create Orderline."
  echo "               16    Create Stock."
  echo "               17    Create Item."
  echo "               18    Load Warehouse."
  echo "               19    Load District."
  echo "               20    Load Item."
  echo "               21    Load History"
  echo "               22    Load Neworder"
  echo "               23    Load Order/Orderline"
  echo "               24    Load Customer"
  echo "               25    Load Stock"
  echo "               26    Create Warehouse Index."
  echo "               27    Create District Index."
  echo "               28    Create Item Index."
  echo "               29    Create Customer1 Index."
  echo "               30    Create Customer2 Index."
  echo "               31    Create Stock Index."
  echo "               32    Create Orders1 Index."
  echo "               33    Create Orders2 Index."
  echo "               34    Create Neworder Index."
  echo "               35    Create Orderline Index."
  echo "               36    Analyze Tables/Clusters/Indexes."
  echo "               37    Create Statistics Tables."
  echo "               38    Load Stored Procedures."
  echo "               39    Create Rollback Segments for Runs."
  echo "               40    Generate Space Report."
  echo "               41    Misc."
  echo "-------------------------------------------------------------------"

  exit 1;
}

torun()
{
  mv -f *.log* log > junk 2>&1
  rm -f $JUNKFILE*

  if test -f $STOPFILE
  then
    echo "An error has occured, please look into the $TRACEFILE file."
    echo "OR you need to remove the 'stop' file found in the current diretory."
    echo "The 'stop' file need to be removed after an error occurs and before resuming."
    exit 1;
  fi
  if test $STEP -ge $START
  then
    if test $STEP -le $END
    then
      STEP=`expr $STEP + 1`
      return 0
    else
      if test $CONTINUE -eq 0
      then
        STEP=`expr $STEP + 1`
        return 0
      fi
    fi
  fi
  STEP=`expr $STEP + 1`
  return 1
}

case $# in
      0)   usage;
           ;;
      1)   case $1 in
                -h)  usage
                     ;;
             [0-9]*)  START=$1
                      CONTINUE=0
                      ;;
                *)  usage
                    ;;
           esac
           ;;
      2)   case $1 in
             -step)  shift
                     case $1 in
                       [0-9]*) ;;
                           *) usage
                              ;;
                     esac
                     START=$1
                     END=$1
                     CONTINUE=1
                     ;;
           [0-9]*)  START=$1
                    shift
                    case $1 in
                      [0-9]*) ;;
                          *) usage
                             ;;
                    esac
                    END=$1
                    CONTINUE=1
                    ;;
               *)  usage
                   ;;
           esac
           ;;
      *)   usage
           ;;
esac

if torun
then
  echo "Creating the Database ..."
  echo "Creating the Database ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step2createdb.sh
  if test $? -ne 0
  then
    echo "Creating the Database failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step2createdb.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Creating the Database failed ..."
  else
    echo "Creating the Database done." `date` "\n" >> $TRACEFILE
    echo "Creating the Database done ..."
  fi
fi

if torun
then
  echo "Creating the Rollback Segments ..."
  echo "Creating the Rollback Segments ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step3createrollback.sh
```

```
  if test $? -ne 0
  then
    echo "Creating the Rollback Segments failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step3createrollback.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Creating the Rollback Segments failed ..."
  else
    echo "Creating the Rollback Segments done." `date` "\n" >> $TRACEFILE
    echo "Creating the Rollback Segments done ..."
  fi
fi

if torun
then
  echo "Shutting down the Database ..."
  echo "Shutting down the Database ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/stepshut.sh
  if test $? -ne 0
  then
    echo "Shutting down the Database failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/stepshut.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Shutting down the Database failed ..."
  else
    echo "Shutting down the Database done." `date` "\n" >> $TRACEFILE
    echo "Shutting down the Database done ..."
  fi
fi

if torun
then
  echo "Start up Database for Build ..."
  echo "Start up Database for Build ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/stepstartb.sh
  if test $? -ne 0
  then
    echo "Start up Database for Build failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/stepstartb.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Start up Database for Build failed ..."
  else
    echo "Start up Database for Build done." `date` "\n" >> $TRACEFILE
    echo "Start up Database for Build done ..."
  fi
fi

if torun
then
  echo "Create User TPCC ..."
  echo "Create User TPCC ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/stepcreateuser.sh
  if test $? -ne 0
  then
    echo "Create User TPCC failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/stepcreateuser.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create User TPCC failed ..."
  else
    echo "Create User TPCC done." `date` "\n" >> $TRACEFILE
    echo "Create User TPCC done ..."
  fi
fi

if torun
then
  echo "Create Tablespaces ..."
  echo "Create Tablespaces ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step5createts.sh
  if test $? -ne 0
  then
    echo "Create Tablespaces failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step5createts*.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Tablespaces failed ..."
  else
    echo "Create Tablespaces done." `date` "\n" >> $TRACEFILE
    echo "Create Tablespaces done ..."
  fi
fi

if torun
then
  echo "Assign Temporary Tablespace to user TPCC ..."
  echo "Assign Temporary Tablespace to user TPCC ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/stepusertemp.sh
  if test $? -ne 0
  then
    echo "Assign Temporary Tablespace to user TPCC failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/stepusertemp.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Assign Temporary Tablespace to user TPCC failed ..."
  else
    echo "Assign Temporary Tablespace to user TPCC done." `date` "\n" >> $TRACEFILE
    echo "Assign Temporary Tablespace to user TPCC done ..."
  fi
fi

if torun
then
  echo "Create Data Dictionary views ..."
  echo "Create Data Dictionary views ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step6createddviews.sh
  if test $? -ne 0
  then
    echo "Create Data  Dictionary views failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step6createddviews.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Data Dictionary views failed ..."
  else
    echo "Create Data Dictionary views done." `date` "\n" >> $TRACEFILE
    echo "Create Data Dictionary views done ..."
  fi
fi

if torun
then
  echo "Create Warehouse ..."
  echo "Create Warehouse ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step9createware.sh
  if test $? -ne 0
  then
    echo "Create Warehouse failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step9createware.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Warehouse failed ..."
  else
    echo "Create Warehouse  done." `date` "\n" >> $TRACEFILE
    echo "Create Warehouse  done ..."
  fi
fi

if torun
then
  echo "Create District ..."
  echo "Create District ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step10createdist.sh
  if test $? -ne 0
  then
    echo "Create District failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step10createdist.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create District failed ..."
  else
    echo "Create District done." `date` "\n" >> $TRACEFILE
    echo "Create District done ..."
  fi
fi

if torun
then
  echo "Create Customer ..."
  echo "Create Customer ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step11createcust.sh
  if test $? -ne 0
  then
    echo "Create Customer failed.\n" `date` "\n" >> $TRACEFILE
    echo "Look at log/step11createcust.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Customer failed ..."
  else
    echo "Create Customer done." `date` "\n" >> $TRACEFILE
    echo "Create Customer done ..."
  fi
fi

if torun
then
  echo "Create History ..."
  echo "Create History ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step12createhist.sh
  if test $? -ne 0
  then
    echo "Create History failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step12createhist.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create History failed ..."
  else
    echo "Create History done." `date` "\n" >> $TRACEFILE
    echo "Create History done ..."
  fi
fi

if torun
then
  echo "Create Order ..."
  echo "Create Order ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step13createordr.sh
  if test $? -ne 0
  then
    echo "Create Order failed." `date` "\n" >> $TRACEFILE
```

```
    echo "Look at log/step13createordr.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Order failed ..."
  else
    echo "Create Order done." `date` "\n" >> $TRACEFILE
    echo "Create Order done ..."
  fi
fi

if torun
then
  echo "Create Neworder ..."
  echo "Create Neworder ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step14createnord.sh
  if test $? -ne 0
  then
    echo "Create Neworder failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step14createnord.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Neworder failed ..."
  else
    echo "Create Neworder done." `date` "\n" >> $TRACEFILE
    echo "Create Neworder done ..."
  fi
fi

if torun
then
  echo "Create Orderline ..."
  echo "Create Orderline ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step15createordl.sh
  if test $? -ne 0
  then
    echo "Create Orderline failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step15createordl.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Orderline failed ..."
  else
    echo "Create Orderline done." `date` "\n" >> $TRACEFILE
    echo "Create Orderline done ..."
  fi
fi

if torun
then
  echo "Create Stock ..."
  echo "Create Stock ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step16createstok.sh
  if test $? -ne 0
  then
    echo "Create Stock failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step16createstok.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Stock failed ..."
  else
    echo "Create Stock done." `date` "\n" >> $TRACEFILE
    echo "Create Stock done ..."
  fi
fi

if torun
then
  echo "Create Item ..."
  echo "Create Item ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step17createitem.sh
  if test $? -ne 0
  then
    echo "Create Item failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step17createitem.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Item failed ..."
  else
    echo "Create Item done." `date` "\n" >> $TRACEFILE
    echo "Create Item done ..."
  fi
fi

if torun
then
  echo "Load Warehouse ..."
  echo "Load Warehouse ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step20loadware.sh
  if test $? -ne 0
  then
    echo "Load Warehouse failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step20loadware.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Warehouse failed ..."
  else
    echo "Load Warehouse done." `date` "\n" >> $TRACEFILE
    echo "Load Warehouse done ..."
  fi
fi

if torun
then
  echo "Load District ..."
  echo "Load District ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step21loaddist.sh
  if test $? -ne 0
  then
    echo "Load District failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step21loaddist.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load District failed ..."
  else
    echo "Load District done." `date` "\n" >> $TRACEFILE
    echo "Load District done ..."
  fi
fi

if torun
then
  echo "Load Item ..."
  echo "Load Item ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step22loaditem.sh
  if test $? -ne 0
  then
    echo "Load Item failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step22loaditem.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Item failed ..."
  else
    echo "Load Item done." `date` "\n" >> $TRACEFILE
    echo "Load Item done ..."
  fi
fi

if torun
then
  echo "Load History ..."
  echo "Load History ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step23loadhist.sh
  if test $? -ne 0
  then
    echo "Load History failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step23loadhist*.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load History failed ..."
  else
    echo "Load History done." `date` "\n" >> $TRACEFILE
    echo "Load History done ..."
  fi
fi

if torun
then
  echo "Load Neworder ..."
  echo "Load Neworder ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step24loadnord.sh
  if test $? -ne 0
  then
    echo "Load Neworder failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step24loadnord*.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Neworder failed ..."
  else
    echo "Load Neworder done." `date` "\n" >> $TRACEFILE
    echo "Load Neworder done ..."
  fi
fi

if torun
then
  echo "Load Order/Orderline ..."
  echo "Load Order/Orderline ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step25loadordrordl.sh
  if test $? -ne 0
  then
    echo "Load Order/Orderline failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step25loadordrordl*.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Order/Orderline failed ..."
  else
    echo "Load Order/Orderline done." `date` "\n" >> $TRACEFILE
    echo "Load Order/Orderline done ..."
  fi
fi

if torun
then
  echo "Load Customer ..."
  echo "Load Customer ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step26loadcust.sh
  if test $? -ne 0
  then
    echo "Load Customer failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step26loadcust*.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Customer failed ..."
```

```
  else
    echo "Load Customer done." `date` "\n" >> $TRACEFILE
    echo "Load Customer done ..."
  fi
fi

if torun
then
  echo "Load Stock ..."
  echo "Load Stock ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step27loadstok.sh
  if test $? -ne 0
  then
    echo "Load Stock failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step27loadstok*.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Stock failed ..."
  else
    echo "Load Stock done." `date` "\n" >> $TRACEFILE
    echo "Load Stock done ..."
  fi
fi

if torun
then
  echo "Create Warehouse Index ..."
  echo "Create Warehouse Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step29createiware.sh
  if test $? -ne 0
  then
    echo "Create Warehouse Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step29createiware.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Warehouse Index failed ..."
  else
    echo "Create Warehouse Index done." `date` "\n" >> $TRACEFILE
    echo "Create Warehouse Index done ..."
  fi
fi

if torun
then
  echo "Create District Index ..."
  echo "Create District Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step30createidist.sh
  if test $? -ne 0
  then
    echo "Create District Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step30createidist.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create District Index failed ..."
  else
    echo "Create District Index done." `date` "\n" >> $TRACEFILE
    echo "Create District Index done ..."
  fi
fi

if torun
then
  echo "Create Item Index ..."
  echo "Create Item Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step31createiitem.sh
  if test $? -ne 0
  then
    echo "Create Item Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step31createiitem.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Item Index failed ..."
  else
    echo "Create Item Index done." `date` "\n" >> $TRACEFILE
    echo "Create Item Index done ..."
  fi
fi

if torun
then
  echo "Create Customer1 Index ..."
  echo "Create Customer1 Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step32createicust1.sh
  if test $? -ne 0
  then
    echo "Create Customer1 Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step32createicust1.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Customer1 Index failed ..."
  else
    echo "Create Customer1 Index done." `date` "\n" >> $TRACEFILE
    echo "Create Customer1 Index done ..."
  fi
fi

if torun
then
  echo "Create Customer2 Index ..."
  echo "Create Customer2 Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step33createicust2.sh
  if test $? -ne 0
  then
    echo "Create Customer2 Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step33createicust2.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Customer2 Index failed ..."
  else
    echo "Create Customer2 Index done." `date` "\n" >> $TRACEFILE
    echo "Create Customer2 Index done ..."
  fi
fi

if torun
then
  echo "Create Stock Index ..."
  echo "Create Stock Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step34createistok.sh
  if test $? -ne 0
  then
    echo "Create Stock Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step34createistok.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Stock Index failed ..."
  else
    echo "Create Stock Index done." `date` "\n" >> $TRACEFILE
    echo "Create Stock Index done ..."
  fi
fi

if torun
then
  echo "Create Order1 Index ..."
  echo "Create Order1 Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step35createiordr1.sh
  if test $? -ne 0
  then
    echo "Create Order1 Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step35createiordr1.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Order1 Index failed ..."
  else
    echo "Create Order1 Index done." `date` "\n" >> $TRACEFILE
    echo "Create Order1 Index done ..."
  fi
fi

if torun
then
  echo "Create Order2 Index ..."
  echo "Create Order2 Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step36createiordr2.sh
  if test $? -ne 0
  then
    echo "Create Order2 Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step36createiordr2.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Order2 Index failed ..."
  else
    echo "Create Order2 Index done." `date` "\n" >> $TRACEFILE
    echo "Create Order2 Index done ..."
  fi
fi

if torun
then
  echo "Create Neworder Index ..."
  echo "Create Neworder Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step37createinord.sh
  if test $? -ne 0
  then
    echo "Create Neworder Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step37createinord.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Neworder Index failed ..."
  else
    echo "Create Neworder Index done." `date` "\n" >> $TRACEFILE
    echo "Create Neworder Index done ..."
  fi
fi

if torun
then
  echo "Create Orderline Index ..."
  echo "Create Orderline Index ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step38createiordl.sh
  if test $? -ne 0
  then
    echo "Create Orderline Index failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step38createiordl.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Orderline Index failed ..."
  else
    echo "Create Orderline Index done." `date` "\n" >> $TRACEFILE
    echo "Create Orderline Index done ..."
```

```
  fi
fi

if torun
then
  echo "Analyze Tables/Clusters/Indexes ..."
  echo "Analyze Tables/Clusters/Indexes ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step39analyze.sh
  if test $? -ne 0
  then
    echo "Analyze Tables/Clusters/Indexes failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step39analyze.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Analyze Tables/Clusters/Indexes failed ..."
  else
    echo "Analyze Tables/Clusters/Indexes done." `date` "\n" >> $TRACEFILE
    echo "Analyze Tables/Clusters/Indexes done ..."
  fi
fi

if torun
then
  echo "Create Statistics Tables ..."
  echo "Create Statistics Tables ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step40createstats.sh
  if test $? -ne 0
  then
    echo "Create Statistics Tables failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step40createstats.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Statistics Tables failed ..."
  else
    echo "Create Statistics Tables done." `date` "\n" >> $TRACEFILE
    echo "Create Statistics Tables done ..."
  fi
fi

if torun
then
  echo "Load Stored Procedures ..."
  echo "Load Stored Procedures ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step41createstoredprocs.sh
  if test $? -ne 0
  then
    echo "Load Stored Procedures failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step41createstoredprocs.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Load Stored Procedures failed ..."
  else
    echo "Load Stored Procedures done." `date` "\n" >> $TRACEFILE
    echo "Load Stored Procedures done ..."
  fi
fi

if torun
then
  echo "Create Rollback Segments for Runs ..."
  echo "Create Rollback Segments for Runs ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step18createrollsegs.sh
  if test $? -ne 0
  then
    echo "Create Rollback Segments for Runs failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step18createrollsegs.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Create Rollback Segments for Runs failed ..."
  else
    echo "Create Rollback Segments for Runs done." `date` "\n" >> $TRACEFILE
    echo "Create Rollback Segments for Runs done ..."
  fi
fi

if torun
then
  echo "Generate Space Reports ..."
  echo "Generate Space Reports ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step42createspacestats.sh
  if test $? -ne 0
  then
    echo "Generate Space Reports failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step42createspacestats.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Generate Space Reports failed ..."
  else
    echo "Generate Space Reports done." `date` "\n" >> $TRACEFILE
    echo "Generate Space Reports done ..."
  fi
fi

if torun
then
  echo "Misc ..."
  echo "Misc ..." `date` "\n" >> $TRACEFILE
  ${SCRIPTS}/step43createmisc.sh
  if test $? -ne 0
  then
    echo "Misc failed." `date` "\n" >> $TRACEFILE
    echo "Look at log/step43createmisc.log for more details." `date` "\n" >> $TRACEFILE
    echo "Stopped" >> $STOPFILE
    echo "Misc failed ..."
  else
    echo "Misc done." `date` "\n" >> $TRACEFILE
    echo "Misc done ..."
  fi
fi

torun
```

runchk.sh

```
#!/bin/sh
#
# $Header: runchk.sh 01-jun-98.19:11:42 skareenh Exp $
#
# runchk.sh
#
#  Copyright (c) Oracle Corporation 1998. All Rights Reserved.
#
#    NAME
#      runchk.sh
#
#    DESCRIPTION
#                Does spot checking after a TPC-C run.
#
#    NOTES
#                runchk.sh [output_file]
#
#                options:
#        -o <output file>        : name of output file
#        -h or -help             : print list of arguments
#        -m <max w_id>           : maximum warehouse id in database
#        -net "list of aliases"  : list of SQL*Net V2 connect string aliases
#        -wids "list of w_id's"  : list of warehouse ids for sampling
#        -bwids "list of w_id's" : list of beginning ids of warehouse ranges
#        -ewids "list of w_id's" : list of ending ids of warehouse ranges
#
#    MODIFIED   (MM/DD/YY)
#    skareenh    06/01/98 - Creation
#

#
# Defaults
#

OFILE="runchkv1"
MULT=
SNET=
WIDS=
BWIDS=
EWIDS=

#
# Parse arguments
#

while [ "$#" != "0" ]
do
  case $1 in
    -o|-ofile)
      shift
      if [ "$1" != "" ]
      then
        OFILE=$1
        shift
      fi
      ;;
    -h|-help)
  echo "Options:"
  echo "  -o <output file>       : name of output file"
  echo "  -h or -help            : print list of arguments"
  echo "  -m <max w_id>          : maximum warehouse id in database"
  echo "  -net \"list of aliases\" : list of SQL*Net V2 connect string aliases"
  echo "  -wids \"list of w_id's\" : list of warehouse ids for sampling"
  echo "  -bwids \"list of w_id's\": list of beginning ids of warehouse ranges"
  echo "  -ewids \"list of w_id's\": list of ending ids of warehouse ranges"
      exit 0
      ;;
    -m|-mult)
      shift
      if [ "$1" != "" ]
      then
        MULT=$1
        shift
      fi
      ;;
    -net)
      shift
      if [ "$1" != "" ]
```

```
        then
          SNET=$1
          shift
        fi
        ;;
      -wids)
        shift
        if [ "$1" != "" ]
        then
          WIDS=$1
          shift
        fi
        ;;
      -bwids)
        shift
        if [ "$1" != "" ]
        then
          BWIDS=$1
          shift
        fi
        ;;
      -ewids)
        shift
        if [ "$1" != "" ]
        then
          EWIDS=$1
          shift
        fi
        ;;
      *)
        echo "Bad argument: $1"
        echo "Use -help option to see correct usage."
        exit 1
        ;;
    esac
done

#
# Check arguments
#

if [ "$OFILE" = "" ]
then
  echo "Error: output file is not specified"
  echo "Use -help option to see correct usage."
  exit 1
fi

NHOSTS=0
for HOST in $SNET
do
  NHOSTS=`expr $NHOSTS + 1`
done

NWIDS=0
for WID in $WIDS
do
  NWIDS=`expr $NWIDS + 1`
done

NBWIDS=0
for BWID in $BWIDS
do
  NBWIDS=`expr $NBWIDS + 1`
done

NEWIDS=0
for EWID in $EWIDS
do
  NEWIDS=`expr $NEWIDS + 1`
done

if [ $NBWIDS != $NEWIDS ]
then
  echo "Error: # of beginning w_id's != # of ending w_id's"
  echo "Use -help option to see correct usage."
  exit 1
fi

if [ $NHOSTS != 0 ]
then
  if [ $NWIDS != 0 -a $NHOSTS != $NWIDS ]
  then
    echo "Error: # of SQL*Net V2 aliases != # of samples"
    echo "Use -help option to see correct usage."
    exit 1
  fi
  if [ $NBWIDS != 0 -a $NHOSTS != $NBWIDS ]
  then
    echo "Error: # of SQL*Net V2 aliases != # of w_id ranges"
    echo "Use -help option to see correct usage."
    exit 1
  fi
else
  if [ $NWIDS != 0 -a $NBWIDS != 0 -a $NWIDS != $NBWIDS ]
  then
    echo "Error: # of samples != # of w_id ranges"
    echo "Use -help option to see correct usage."
    exit 1
  fi
fi

#
# By default, if no SQL*Net V2 string is specified, all checks are
# run on the local node.
#

#
# By default, if no w_id ranges are specified, then ranges will be
# generated automatically based on the total number of warehouse and
# the number of SQL*Net V2 strings or the number of w_id samples.
#
if [ $NBWIDS = 0 ]
then
  if [ "$MULT" = "" ]
  then
    echo "Error: max warehouse id in this database is not specified"
    echo "Use -help option to see correct usage."
    exit 1
  fi

  if [ $NHOSTS != 0 ]
  then
    NSAMPS=$NHOSTS
  elif [ $NWIDS != 0 ]
  then
    NSAMPS=$NWIDS
  elif [ $MULT -ge 4 ]
  then
    NSAMPS=4
  else
    NSAMPS=$MULT
  fi

  EVENW=`expr $MULT % $NSAMPS`
  if [ $EVENW != 0 ]
  then
    echo "Error: cannot evenly divide # of warehouses by # of nodes/samples"
    echo "       so cannot generate w_id ranges automatically"
    echo "Use -help option to see correct usage."
    exit 1
  fi

  WPERN=`expr $MULT / $NSAMPS`
  SW=1
  EW=$WPERN
  BWIDS="$SW"
  EWIDS="$EW"
  I=2
  while [ $I -le $NSAMPS ]
  do
    SW=`expr $SW + $WPERN`
    EW=`expr $EW + $WPERN`
    BWIDS="$BWIDS $SW"
    EWIDS="$EWIDS $EW"
    I=`expr $I + 1`
  done
  NBWIDS=$NSAMPS
  NEWIDS=$NSAMPS
fi

if [ $NWIDS = 0 ]
then
  WIDS=""
  NWIDS=$NBWIDS
  I=1
  while [ $I -le $NWIDS ]
  do
    J=1
    for DUMMY in $BWIDS
    do
      if [ $J = $I ]
      then
        SW=$DUMMY
        break;
      fi
      J=`expr $J + 1`
    done

    J=1
    for DUMMY in $EWIDS
    do
      if [ $J = $I ]
      then
        EW=$DUMMY
        break;
      fi
      J=`expr $J + 1`
    done
```

```
    if [ $I = 1 ]
    then
      WIDS="$SW"
    elif [ $I = $NWIDS ]
    then
      WIDS="$WIDS $EW"
    else
      TID=`expr $SW + $EW`
      TID=`expr $TID / 2`
      WIDS="$WIDS $TID"
    fi
    I=`expr $I + 1`
  done
fi


#
# audit directory
#
if [ "X$SRCHOME" = "X" -a  "X$ORACLE_HOME" != "X" ]
then
   SRCHOME=$ORACLE_HOME
fi
BENCH_HOME=$SRCHOME
TPCC_AUDIT= $BENCH_HOME/audit

#
# output directory
#
ODIR=${TPCC_AUDIT}/output
if [ ! -d $ODIR ]
then
   mkdir $ODIR
fi

#
# shell script directory
#
SDIR=${TPCC_AUDIT}/scripts

#
# SQL script directory
#
SQLDIR=${TPCC_AUDIT}/sql

#
# source directory
#
SRCDIR=${TPCC_AUDIT}/source

#
# run checks
#
date | tee ${ODIR}/${OFILE}
${SDIR}/swt_temp.sh temp

cat >> ${ODIR}/${OFILE} <<!



Number of Active Warehouses
---------------------------

!

sqlplus tpcc/tpcc > ${ODIR}/${OFILE}.0 2>&1 <<!
  set termout on
  set echo on
  set timing on

  select count(*) from ware;

drop table tpcc_audit_tab2;
create table tpcc_audit_tab2 (endtime date);
delete from tpcc_audit_tab2;
insert into tpcc_audit_tab2 (endtime)
 select sysdate from dual;
commit;

   quit;
!

cat ${ODIR}/${OFILE}.0 >> ${ODIR}/${OFILE}
cat ${ODIR}/${OFILE}.0
rm -f ${ODIR}/${OFILE}.0

sqlplus tpcc/tpcc @${SQLDIR}/runchk $MULT > ${ODIR}/${OFILE}.5 &

sqlplus tpcc/tpcc @${SQLDIR}/runchk_stok $MULT > ${ODIR}/${OFILE}.6 &

sqlplus tpcc/tpcc @${SQLDIR}/remote $MULT > ${ODIR}/${OFILE}.0 &

I=1
while [ $I -le $NWIDS ]
do
  NSWID=1
  for DUMSWID in $WIDS
  do
    if [ $NSWID = $I ]
    then
      SWID=$DUMSWID
      break;
    fi
    NSWID=`expr $NSWID + 1`
  done

  J=1
  for DUMMY in $BWIDS
  do
    if [ $J = $I ]
    then
      SW=$DUMMY
      break;
    fi
    J=`expr $J + 1`
  done

  J=1
  for DUMMY in $EWIDS
  do
    if [ $J = $I ]
    then
      EW=$DUMMY
      break;
    fi
    J=`expr $J + 1`
  done

  if [ $NHOSTS = 0 ]
  then
    sqlplus tpcc/tpcc @${SQLDIR}/runchk_parallel $SWID $SW $EW | \
          tee ${ODIR}/${OFILE}.$I &
  else
    NCSTR=1
    for DUMCSTR in $SNET
    do
      if [ $NCSTR = $I ]
      then
        CSTR=$DUMCSTR
        break;
      fi
      NCSTR=`expr $NCSTR + 1`
    done
    sqlplus tpcc/tpcc@$CSTR @${SQLDIR}/runchk_parallel $SWID $SW $EW | \
          tee ${ODIR}/${OFILE}.$I &
  fi
  I=`expr $I + 1`
done

wait

I=1
while [ $I -le $NWIDS ]
do
  cat >> ${ODIR}/${OFILE} <<!



Sample $I of Run Check
----------------------

!

  cat ${ODIR}/${OFILE}.$I >> ${ODIR}/${OFILE}
  rm -f ${ODIR}/${OFILE}.$I
  I=`expr $I + 1`
done

cat >>  ${ODIR}/${OFILE} <<!


Show unused warehouses: w_id > $MUL
-----------------------

!

cat ${ODIR}/${OFILE}.0 >> ${ODIR}/${OFILE}
rm -f ${ODIR}/${OFILE}.0

cat ${ODIR}/${OFILE}.5 >> ${ODIR}/${OFILE}
rm -f ${ODIR}/${OFILE}.5

cat ${ODIR}/${OFILE}.6 >> ${ODIR}/${OFILE}
rm -f ${ODIR}/${OFILE}.6

${SDIR}/swt_temp.sh system
date | tee -a ${ODIR}/${OFILE}
```

tpccenv.sh

```
#!sh
SQLDBA=svrmgrl
export SQLDBA
SEQLPLUS=sqlplus
export SEQLPLUS
# cp benchrun/bin/tpccload.exe .
TPCCLOAD=tpccload.exe
export TPCCLOAD
ORACLE_SID=tpcc
export ORACLE_SID
BUILDDIR=${ORACLE_HOME}/bench/tpc/tpcc/scripts/build32k
export SCRIPTS
SCRIPTS=${BUILDDIR}/scripts
export SCRIPTS
SQLDIR=${BUILDDIR}/sql
export SQLDIR
DB_SIZE=32000
export DB_SIZE
PATH=$PATH:${SCRIPTS}
```

p_build.ora

```
#
#
#===============================================================+
#       Copyright (c) 1998  Oracle Corp, Redwood Shores, CA      |
#                OPEN SYSTEMS PERFORMANCE GROUP                  |
#                   All Rights Reserved                          |
#===============================================================+
# FILENAME
#     p_build.ora
# DESCRIPTION
#     Oracle parameter file for building TPC-C database.
#================================================================
#

_lgwr_async_io                                          = FALSE
disk_asynch_io                  = TRUE
_discrete_transactions_enabled    = FALSE
log_archive_start               = FALSE
db_file_multiblock_read_count    = 64
distributed_transactions         = 0
hash_join_enabled               = FALSE
lock_sga                        = TRUE
log_checkpoints_to_alert         = TRUE
log_checkpoint_interval          = 1000000000
cpu_count                                       = 64
db_writer_processes             = 10
_db_writer_chunk_writes          = 2000
_db_writer_max_writes            = 2000
_disable_incremental_checkpoints  = TRUE
compatible                                      = 9.0.1.0.0
control_files                                   = (?/dbs/tpcc_disks/control01,
?/dbs/tpcc_disks/control02)
sort_area_size                  = 30000000
parallel_max_servers             = 1000
recovery_parallelism             = 100
db_name                         = tpcc
db_files                        = 400
db_block_size                                           = 2048
db_cache_size                                           = 1000M
db_8k_cache_size                        = 1200M
enqueue_resources               = 60000
dml_locks                       = 2500
log_buffer                      = 33554432
_log_simultaneous_copies         = 128
_log_archive_buffer_size         = 32
max_rollback_segments            = 1100
open_cursors                    = 3000
processes                       = 2000
sessions                        = 3000
transactions                    = 10000
shared_pool_size                = 750M
shared_pool_reserved_size        = 350M
cursor_space_for_time            = TRUE
transaction_auditing             = FALSE
replication_dependency_tracking   = FALSE
_db_block_cache_protect          = FALSE
db_block_checking                = FALSE
rollback_segments               = (t1,t2,t3,t4,t5,t6,t7,t8,t9,
t10,t11,t12,t13,t14,t15,t16,t17,t18,t19,t20,t21,t22,t23,t24,t25,t26,t27,t28,t29,
t30,t31,t32,t33,t34,t35,t36,t37,t38,t39,t40,t41,t42,t43,t44,t45,t46,t47,t48,t49,
t50,t51,t52,t53,t54,t55,t56,t57,t58,t59,t60,t61,t62,t63,t64,t65,t66,t67,t68,t69,
t70,t71,t72,t73,t74,t75,t76,t77,t78,t79,t80,t81,t82,t83,t84,t85,t86,t87,t88,t89,
t90,t91,t92,t93,t94,t95,t96,t97,t98,t99,t100)
```

p_create.ora

```
compatible = 9.0.1.0.0
db_name = tpcc
control_files                                   = (?/dbs/tpcc_disks/control01,
?/dbs/tpcc_disks/control02)
db_files = 400
dml_locks = 500
log_buffer = 1048576
processes =  1471
db_block_size = 2048
db_cache_size                = 100M
db_8k_cache_size               = 100M
disk_asynch_io               = FALSE
undo_management                 = manual
```

addft.sh

```
#!/bin/ksh

TNAME=$1
FNAME=$2
SIZE=$3
NFILE=`expr $4 + 1`

I=2
while [ $I -le $NFILE ]
   do
   if [ $NFILE -gt 99 ]
                then
                N=`printf "%03d" $I`
   else
                N=`printf "%02d" $I`
   fi
   addfile.sh $TNAME ${FNAME}$N $SIZE   &
   I=`expr $I + 1`
   done
wait
```

addtempfile.sh

```
#!sh

date > step5addfile_$1.log

$SEQLPLUS tpcc/tpcc <<!
   spool step5addfile_$1.log
   set echo on
   alter tablespace $1 add tempfile '$2' size $3 reuse;
   set echo off
   spool off
   exit ;
!

date >> step5addfile_$1.log
```

addtempft.sh

```
#!/bin/ksh

TNAME=$1
FNAME=$2
SIZE=$3
NFILE=`expr $4 + 1`

I=2
while [ $I -le $NFILE ]
do
N=`printf "%02d" $I`
addtempfile.sh $TNAME ${FNAME}$N $SIZE &
I=`expr $I + 1`
done
wait
```

addtemfts.sh

```
#!/bin/ksh

TNAME=$1
FNAME=$2
SIZE=$3
NFILE=`expr $4 + 1`
```

```
I=2
while [ $I -le $NFILE ]
do
N=`printf "%02d" $I`
addtempfile.sh $TNAME ${FNAME}$N $SIZE &
I=`expr $I + 1`
done
wait
```

addts.sh

```
#!sh

date > step5creates_$1.log

$SEQLPLUS tpcc/tpcc <<!
  spool step5creates_$1.log
  set echo on
  drop tablespace $1 including contents;
  create tablespace $1 datafile '$2' size $3 reuse extent management local uniform size $4 segment
space management auto nologging ;
  set echo off
  spool off
  exit ;
!

date >> step5creates_$1.log
```

loadware.sh

```
#!sh
$TPCCLOAD -M ${DB_SIZE} -w > step20loadware.log 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

loaddist.sh

```
#!sh
$TPCCLOAD -M ${DB_SIZE} -d > step21loaddist.log 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

loaditem.sh

```
#!sh
$TPCCLOAD -M ${DB_SIZE} -i > step22loaditem.log 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

loadhist.sh

```
#!sh
I=1
SW=1
EW=1280
INC=1280
while [ $I -le 25 ]
do
  $TPCCLOAD -M ${DB_SIZE} -h -b $SW -e $EW > step23loadhist.log${I} 2>&1 &
  I=`expr $I + 1`
  SW=`expr $SW + $INC`
  EW=`expr $EW + $INC`
done
wait
```

loadnord.sh

```
#!sh
I=1
SW=1
EW=1280
INC=1280
while [ $I -le 25 ]
do
  $TPCCLOAD -M ${DB_SIZE} -n -b $SW -e $EW >step24loadnord.log${I} 2>&1 &
  I=`expr $I + 1`
  SW=`expr $SW + $INC`
  EW=`expr $EW + $INC`
done
wait
```

loadordrordl.sh

```
#!sh
I=1
SW=1
EW=400
INC=400
DIFF=`expr $SW + $INC - $EW - 1`
if test $DIFF -ne 0
   then
   echo "$0:Something wrong with your parameters!"
   exit
fi
while [ $I -le 80 ]
do
  $TPCCLOAD -M ${DB_SIZE} -o ?/dbs/tpcc_disks/dummy${I}.dat -b $SW -e $EW
>step25loadordrordl.log${I} 2>&1 &
  I=`expr $I + 1`
  SW=`expr $SW + $INC`
  EW=`expr $EW + $INC`
done
wait
```

loadcust.sh

```
#!sh
I=1
SW=1
EW=256
INC=256
DIFF=`expr $SW + $INC - $EW - 1`
if test $DIFF -ne 0
   then
   echo "$0:Something wrong with your parameters!"
   exit
fi
while [ $I -le 125 ]
do
  $TPCCLOAD -M ${DB_SIZE} -c -b $SW -e $EW >step26loadcust.log${I} 2>&1 &
  I=`expr $I + 1`
  SW=`expr $SW + $INC`
  EW=`expr $EW + $INC`
done
wait
```

loadstok.sh

```
#!sh
I=1
SI=1
EI=500
INC=500
DIFF=`expr $SI + $INC - $EI - 1`
if test $DIFF -ne 0
   then
   echo "$0:Something wrong with your parameters!"
   exit
fi
```

```
while [ $I -le 200 ]
do
  $TPCCLOAD -M ${DB_SIZE} -S -j $SI -k $EI >step27loadstok.log${I} 2>&1 &
  I=`expr $I + 1`
  SI=`expr $SI + $INC`
  EI=`expr $EI + $INC`
done
wait
```

Createts.sh

```
#!sh

addtempts.sh temp   \?/dbs/${ORACLE_SID}_disks/temp01   8001M  1000M &
addts.sh    stok   \?/dbs/${ORACLE_SID}_disks/stock001 8001M  1000M &
addts.sh    cust   \?/dbs/${ORACLE_SID}_disks/cust001  8001M  1000M &
addts_mb.sh ordl   \?/dbs/${ORACLE_SID}_disks/ordl01   32001M 1000M 8K &
addts_mb.sh nord   \?/dbs/${ORACLE_SID}_disks/nord01   9001M  1000M 8K &
addts_mb.sh ordr   \?/dbs/${ORACLE_SID}_disks/ordr01   25001M 1000M 8K &
addts_mb.sh hist   \?/dbs/${ORACLE_SID}_disks/hist01   26001M 1000M 8K &
addts_mb.sh istk   \?/dbs/${ORACLE_SID}_disks/istk01   8001M   500M 8K &
addts_mb.sh icust1 \?/dbs/${ORACLE_SID}_disks/icust101 8001M   500M 8K &
addts_mb.sh icust2 \?/dbs/${ORACLE_SID}_disks/icust201 8001M   500M 8K &
addts_mb.sh iord1  \?/dbs/${ORACLE_SID}_disks/iord101  25001M  500M 8K &
addts_mb.sh iord2  \?/dbs/${ORACLE_SID}_disks/iord201  30001M  500M 8K &
addts.sh    misc   \?/dbs/${ORACLE_SID}_disks/misc01   4001M   500M &
addts_mb.sh tools  \?/dbs/${ORACLE_SID}_disks/tools01  20001M  1M 8K &

wait

addft.sh    stok   \?/dbs/${ORACLE_SID}_disks/stock  8001M 164 &
addft.sh    cust   \?/dbs/${ORACLE_SID}_disks/cust   8001M  123 &
addft.sh    hist   \?/dbs/${ORACLE_SID}_disks/hist   26001M   3 &
addft.sh    istk   \?/dbs/${ORACLE_SID}_disks/istk   8001M  15 &
addft.sh    icust1 \?/dbs/${ORACLE_SID}_disks/icust1 8001M   6 &
addft.sh    icust2 \?/dbs/${ORACLE_SID}_disks/icust2 8001M  10 &
addft.sh    ordr   \?/dbs/${ORACLE_SID}_disks/ordr   25001M   2 &
addft.sh    iord1  \?/dbs/${ORACLE_SID}_disks/iord1  25001M   1 &
addft.sh    iord2  \?/dbs/${ORACLE_SID}_disks/iord2  30001M   2 &
addft.sh    ordl   \?/dbs/${ORACLE_SID}_disks/ordl   32001M  34 &
addft.sh    nord   \?/dbs/${ORACLE_SID}_disks/nord   9001M   1 &
addtempft.sh temp   \?/dbs/${ORACLE_SID}_disks/temp   8001M  59 &

wait
```

Chkpt.sh

```
#!sh

$SEQLPLUS 'sys/change_on_install as sysdba' @${SQLDIR}/stepchkpt
wait
```

Create_user.sh

```
#!sh

$SEQLPLUS  'sys/change_on_install as sysdba' @${SQLDIR}/stepcreateuser > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

create_cache_views.sql

```
spool step6createddviews.log
@$ORACLE_HOME/rdbms/admin/catalog
@$ORACLE_HOME/rdbms/admin/catproc
connect system/manager
@$ORACLE_HOME/sqlplus/admin/pupbld
spool off
exit sql.sqlcode;
```

extent.sql

```
REM=============================================================
+
REM          Copyright (c) 1994  Oracle Corp, Belmont, CA        |
REM               OPEN SYSTEMS PERFORMANCE GROUP                 |
REM                  All Rights Reserved                         |
REM=============================================================
+
REM FILENAME
REM      extent.sql
REM DESCRIPTION
REM      List all extents in all the TPCC tablespaces.
REM
REM  Usage: sqlplus '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;
```

Initnew.sql

```
-- The initnew package for storing variables used in the
-- New Order anonymous block
CREATE OR REPLACE PACKAGE initnew
AS
 TYPE intarray IS TABLE OF INTEGER index by binary_integer;
 TYPE distarray IS TABLE OF VARCHAR(24) index by binary_integer;
 nulldate       DATE;
 s_dist                                    distarray;
 idx1arr                        intarray;
 s_remote                       intarray;
 PROCEDURE new_init(idxarr intarray);
END initnew;
/
show errors;
CREATE OR REPLACE PACKAGE BODY initnew AS
  PROCEDURE new_init (idxarr  intarray)
  IS
  BEGIN
       -- initialize null date
    nulldate := TO_DATE('01-01-1811', 'MM-DD-YYYY');
    idx1arr := idxarr;
  END new_init;
END initnew;
/
show errors
```

Initpay.sql

```
CREATE OR REPLACE PACKAGE initpay
AS
 TYPE rowidarray IS TABLE OF ROWID INDEX BY BINARY_INTEGER;
 row_id              rowidarray;
 cust_rowid            ROWID;
 dist_name             VARCHAR2(11);
 ware_name              VARCHAR2(11);
 c_num               BINARY_INTEGER;
```

```
 PROCEDURE pay_init;
END initpay;
/
show errors;
CREATE OR REPLACE PACKAGE BODY initpay AS
  PROCEDURE pay_init IS
  BEGIN
    NULL;
  END pay_init;
END initpay;
/
show errors;
```

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;
```

Space_get.sql

```
REM============================================================
+
REM        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA     |
REM                OPEN SYSTEMS PERFORMANCE GROUP                  |
REM                    All Rights Reserved                         |
REM============================================================
+
REM FILENAME
REM       space_get.sql
REM DESCRIPTION
REM       Get sizes of tables, indexes and tablespaces.
REM  Usage: sqlplus internal/internal @space_get [<tpm> <# of warehouses>]
REM============================================================
*/
  set echo on;
  delete from tpcc_data;
  delete from tpcc_space;
  delete from tpcc_totspace;
  insert into tpcc_data
  select substr(segment_name,1,11), substr(segment_type,1,15),
         sum(blocks),
         round(sum(blocks) * 0.05), 0,
         sum(blocks) + round(sum(blocks) * 0.05)
  from   dba_extents
  where  owner = 'TPCC' AND ( segment_type = 'INDEX' OR
         segment_type = 'INDEX PARTITION' OR segment_type = 'CLUSTER'
         OR segment_type = 'TABLE' OR segment_type = 'TABLE PARTITION')
         AND tablespace_name <> 'SYSTEM'
  group by segment_name, segment_type;
  insert into tpcc_data
    select 'SYSTEM', 'SYS', sum(blocks), 0, 0, sum(blocks)
    from   dba_data_files
    where  tablespace_name = 'SYSTEM';
  insert into tpcc_data
    select 'ROLL_SEG', 'SYS',
           sum(blocks), 0, 0, sum(blocks)
    from   dba_data_files
    where  tablespace_name like 'ROLL%'
    group by tablespace_name;
  update tpcc_data
    set five_pct = 0,
       daily_grow = round(blocks * &&1 / 62.5 / &&2),
       total = blocks + round(blocks * &&1 / 62.5 / &&2)
    where segment = 'HIST' OR segment = 'ORDR' OR
          segment = 'IORDL';
  insert into tpcc_space
  select substr(ex$.name,1,11), sum(sp$.sz_blocks), 0, 0, 0, 0
  from
    (select tablespace_name , sum(blocks) sz_blocks
     from dba_data_files
     where tablespace_name <> 'SYSTEM'
     group by tablespace_name
    ) sp$,
    (select distinct tablespace_name, segment_name name
     from dba_extents
     where owner = 'TPCC'
       and (segment_type = 'CLUSTER' or segment_type = 'TABLE'
         or segment_type = 'TABLE PARTITION' or segment_type = 'INDEX'
         or segment_type = 'INDEX PARTITION')
       and tablespace_name <> 'SYSTEM'
    ) ex$
  where sp$.tablespace_name = ex$.tablespace_name
  group by ex$.name;
  insert into tpcc_space
  select substr(tablespace_name,1,11), sum(blocks), 0, 0, 0, 0
  from dba_data_files
  where tablespace_name = 'SYSTEM'
  group by tablespace_name;
  update tpcc_space
    set required =
    (
      select sum(total)
      from   tpcc_data
      where  tpcc_data.segment = tpcc_space.segment
    )
    where segment in
    (
      select segment from tpcc_data
    );
  update tpcc_space
    set static =
    (
      select sum(total)
      from   tpcc_data
      where  tpcc_data.segment = tpcc_space.segment
    )
    where segment in
    (
      select segment from tpcc_data
    );
  update tpcc_space
    set static = 0,
```

```
    dynamic =
  (
    select sum(blocks)
    from   tpcc_data
    where  tpcc_data.segment = tpcc_space.segment
  )
  where segment in ('HIST', 'ORDR', 'IORDL');
update tpcc_space
  set oversize = blocks - required;
insert into tpcc_totspace
  select &&1, &&2, sum(static), sum(dynamic), sum(oversize), 0, 0, 0
  from   tpcc_space;
update tpcc_totspace
  set daily_grow =
  (
    select sum(daily_grow)
    from   tpcc_data
  );
update tpcc_totspace
  set space180 = static + 180 * daily_grow;
set echo off;
```

Space_init.sql

```
REM=============================================================
+
REM FILENAME
REM      space_init.sql
REM DESCRIPTION
REM      Create tables for space calculations.
REM  Usage: sqlplus internal/internal @space_init.sql
REM=============================================================
*/
  set echo on;
  drop table tpcc_data;
  drop table tpcc_space;
  drop table tpcc_totspace;
  create table tpcc_data (
    segment      varchar2(11),
    type         varchar2(15),
    blocks       number,
    five_pct     number,
    daily_grow   number,
    total        number
  );
  create table tpcc_space (
    segment      varchar2(11),
    blocks       number,
    required     number,
    static       number,
    dynamic      number,
    oversize     number
  );
  create table tpc c_totspace (
    tpm          number,
    nware        number,
    static       number,
    dynamic      number,
    oversize     number,
    daily_grow   number,
    daily_spre   number,
    space180     number
  );
  create unique index itpcc_data on tpcc_data (segment);
  create unique index itpcc_space on tpcc_space (segment);
  set echo off;
```

Space_rpt.sql

```
REM=============================================================
+
REM        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA      |
REM                  OPEN SYSTEMS PERFORMANCE GROUP                 |
REM                      All Rights Reserved                        |
REM=============================================================
+
REM FILENAME
REM      space_rpt.sql
REM DESCRIPTION
REM      Generate space report and save it in space.rpt
REM  Usage: sqlplus internal/internal @space_rpt.sql
REM=============================================================
*/
  set space 2
  set pagesize 2000
  set echo off
  set termout off
  set verify off
  set feedback off
  spool space.rpt
  select tpm, nware from tpcc_totspace;
  select * from tpcc_data order by segment;
  select * from tpcc_space order by segment;
  select static, dynamic, oversize, daily_grow, daily_spre, space180
    from tpcc_totspace;
  spool off;
```

Create_dist.sql

```
spool step10createdist.log;
set echo on;
drop table dist;
drop cluster distcluster including tables;
set timing on;
create cluster distcluster (
                                d_w_id          number(5,0),
                                d_id            number(2,0)
                )
                single                          table
                hashkeys        320000
                hash is                         (d_w_id - 1) * 10 + d_id
                size                            1536
                initrans        3
                pctfree                         0
        tablespace                      misc;
create table dist (
        d_id            number(2,0),
                d_w_id          number(5,0),
        d_ytd           number,
        d_tax           number,
        d_next_o_id     number,
        d_name          varchar2(10),
        d_street_1      varchar2(20),
        d_street_2      varchar2(20),
        d_city          varchar2(20),
        d_state         char(2),
        d_zip           char(9)
)
                cluster distcluster (d_w_id, d_id);
spool off;
set echo off;
exit sql.sqlcode;
```

Create_item.sql

```
spool step17createitem.log;
set echo on;
drop table item;
drop cluster itemcluster including tables;
set timing on;
create cluster itemcluster (
i_id    number(6,0)
  )
  single    table
  hashkeys    100000
  hash is    (i_id + 1)
  size    120
  initrans    3
  pctfree    0
  storage ( buffer_pool keep freelists 22 freelist groups 43 )
  tablespace    misc;
create table item (
        i_id            number(6,0),
        i_name          varchar2(24),
        i_price         number,
        i_data          varchar2(50),
        i_im_id         number
)
cluster itemcluster (i_id
);
spool off;
set echo off;
exit sql.sqlcode;
```

Create_iware.sql

```
spool step29createiware.log;
set echo on;
drop index iware;
set timing on;
create unique index iware on ware (w_id)
  initrans    3
```

```
  pctfree    1
  storage ( freelists 22 freelist groups 43 )
  tablespace    misc;
spool off;
set echo off;
exit sql.sqlcode;
```

Create_db.sql

```
spool step2createdb.log
set echo on
startup pfile=admin/p_create.ora nomount
create database tpcc
  controlfile reuse
  maxdatafiles 500
  maxinstances 1
  datafile '?/dbs/tpcc_disks/sys01' size 801M reuse
  logfile '?/dbs/tpcc_disks/log01' size 160000M reuse,
        '?/dbs/tpcc_disks/log02' size 160000M reuse;

@$ORACLE_HOME/rdbms/admin/catalog
@$ORACLE_HOME/rdbms/admin/catproc

spool off
set echo off
exit sql.sqlcode
```

Create_idist.sql

```
spool step30createidist.log;
set echo on;
drop index idist;
set timing on;
create unique index idist on dist (d_w_id, d_id)
  initrans    3
  pctfree    5
  storage ( freelists 22 freelist groups 43 )
  tablespace    misc;
spool off;
set echo off;
exit sql.sqlcode;
```

Create_item.sql

```
spool step17createitem.log;
set echo on;
drop table item;
drop cluster itemcluster including tables;
set timing on;
create cluster itemcluster (
i_id    number(6,0)
  )
  single    table
  hashkeys    100000
  hash is    (i_id + 1)
  size    120
  initrans    3
  pctfree    0
  storage ( buffer_pool keep freelists 22 freelist groups 43 )
  tablespace    misc;
create table item (
        i_id            number(6,0),
        i_name          varchar2(24),
        i_price         number,
        i_data          varchar2(50),
        i_im_id         number
)
cluster itemcluster (i_id
);
spool off;
set echo off;
exit sql.sqlcode;
```

Analyze.sql

```
spool step39analyze.log;
set echo on;
analyze cluster warecluster estimate statistics;
analyze table ware compute statistics;
analyze cluster distcluster estimate statistics;
analyze table dist compute statistics;
analyze cluster stokcluster estimate statistics;
analyze table stok estimate statistics;
analyze index istok estimate statistics;
analyze cluster itemcluster estimate statistics;
analyze table item estimate statistics;
analyze index iordl estimate statistics;
analyze table ordl estimate statistics;
analyze index iordr1 estimate statistics;
analyze index iordr2 estimate statistics;
analyze index inord estimate statistics;
analyze table nord estimate statistics;
analyze table ordr estimate statistics;
analyze table hist estimate statistics;
analyze index iitem estimate statistics;
analyze index iware compute statistics;
analyze index icust1 estimate statistics;
analyze index icust2 estimate statistics;
analyze index idist compute statistics;
analyze cluster custcluster estimate statistics;
analyze table cust estimate statistics;
set echo off;
spool off;
exit sql.sqlcode;
```

Create_rollback.sql

```
#!sh

rb=1
while [ $rb -le 100 ]
do
$SEQLPLUS 'sys/change_on_install as sysdba' << !! >> junk 2>&1
  set echo on
  drop public rollback se gment t$rb;
  alter rollback segment t$rb offline;
  create public rollback segment t$rb
      storage (initial 200K minextents 2 next 200K);
  set echo off
  exit sql.sqlcode
!!
rb=`expr $rb + 1`
done

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

Create_storedprocs.sql

```
spool step41createstoredprocs.log
@sql/initpay
@sql/initnew
spool off
exit sql.sqlcode;
```

Create_spacesats.sql

```
spool step42createspacestats.log
@sql/space_init
@sql/space_get 395000 32000
@sql/space_rpt
spool off
exit sql.sqlcode;
```

Create_misc.sql

```
spool step43createmisc.log
set echo on;
alter user tpcc temporary tablespace system;
grant execute on dbms_lock to public;
grant execute on dbms_pipe to public;
grant select on v_$parameter to public;
@${SQLDIR}/plsql_mon
@${SQLDIR}/cre_tab
@${SQLDIR}/create_cache_views
@${SQLDIR}/views
@${SQLDIR}/dml
@${SQLDIR}/extent
```

```
@${SQLDIR}/freeext
spool off
exit sql.sqlcode;
```

Create_ddviews.sql

```
spool step6createddviews.log
@$ORACLE_HOME/rdbms/admin/catalog
@$ORACLE_HOME/rdbms/admin/catproc
connect system/manager
@$ORACLE_HOME/sqlplus/admin/pupbld
spool off
exit sql.sqlcode;
```

Create_wa re.sql

```
spool step9createware.log;
set echo on;
drop table ware;
drop cluster warecluster including tables;
set timing on;
create cluster warecluster (
                        w_id                    number(5,0)
        )
        single                                  table
        hashkeys                32000
        hash is                                 w_id
        size                                    1536
        initrans                3
        pctfree                                 0
    tablespace          misc;
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);
spool off;
set echo off;
exit sql.sqlcode;
```

Chkpt.sql

```
spool stepchkpt.log;
set echo on;
alter system switch logfile;
alter system switch logfile;
set echo off;
spool off;
exit ;
```

Create_user.sql

```
spool stepcreateuser.log;
set echo on;
create user tpcc identified by tpcc;
grant dba to tpcc;
set echo off;
spool off;
exit ;
```

Shut.sql

```
spool stepshut.log;
set echo on;
alter system switch logfile;
alter system switch logfile;
shutdown immediate;
set echo off;
spool off;
exit ;
```

Startb.sql

```
spool stepstartb.log;
set echo on;
startup pfile=admin/p_build.ora open;
set echo off;
spool off;
exit ;
```

Usertemp.sql

```
spool stepusertemp.log;
set echo on;
alter user tpcc temporary tablespace temp;
set echo off;
spool off;
exit ;
```

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 <unistd.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. disctrict id            */
#define CUSTFAC 3000                    /* max. cust id*/
#define STOCFAC 100000                  /* max. stok id*/
#define ITEMFAC 100000                  /* max. item id                 */
```

```
#define HISTFAC 30000          /* hist / ware  */
#define ORDEFAC 3000           /* order / dist     */
#define NEWOFAC 900            /* new order / dist */

#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 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)"

ldadef tpclda;
csrdef curw, curd, curc, curh, curs, curi, curo1, curo2, curol1, curol2, curno;
unsigned long tpchda[256];

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();


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, "\n");
  exit(1);

}



void errrpt (lda, cur)

csrdef *lda;
csrdef *cur;

{

  text msg[2048];

  if (cur->rc) {
    oerhms (lda, cur->rc, msg, 2048);
    fprintf (stderr, "TPC-C load error: %s\n", msg);
  }

}



void quit ()

{

  if (oclose (&curw))
    errrpt (&tpclda, &curw);

  if (oclose (&curd))
    errrpt (&tpclda, &curd);

  if (oclose (&curc))
    errrpt (&tpclda, &curc);

  if (oclose (&curh))
    errrpt (&tpclda, &curh);

  if (oclose (&curs))
    errrpt (&tpclda, &curs);

  if (oclose (&curi))
    errrpt (&tpclda, &curi);

  if (oclose (&curo1))
    errrpt (&tpclda, &curo1);

  if (oclose (&curo2))
    errrpt (&tpclda, &curo2);

  if (oclose (&curol1))
    errrpt (&tpclda, &curol1);

  if (oclose (&curol2))
    errrpt (&tpclda, &curol2);

  if (oclose (&curno))
    errrpt (&tpclda, &curno);

  if (ologof (&tpclda))
    fprintf (stderr, "TPC-C load error: Error in logging off\n");

}
```

```
void main (argc, argv)

int argc;
char *argv[];

{

  char *uid="tpcc/tpcc";
  text sqlbuf[1024];
  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[15];
  int ol_d_id[15];
  int ol_w_id[15];
  int ol_number[15];
  int ol_i_id[15];
  int ol_supply_w_id[15];
  int ol_amount[15];
  char ol_dist_info[15][24];

  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];
#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 stok 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 ((bware  < 1) || (bware > scale)) {
    fprintf (stderr, "Invalid beginning ware number: '%d' \n", bware);
    myusage ();
  }

  if ((eware  < bware) || (eware > scale)) {
    fprintf (stderr, "Invalid ending ware 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 */

    if (orlon (&tpclda, (ub1 *) tpchda, (text *) uid, -1, (text *) 0, -1, 0)) {
      fprintf (stderr, "TPC-C load error: Error in logging on\n");
      errrpt (&tpclda, &tpclda);
      exit (1);
    }

    fprintf (stderr, "\nConnected to Oracle userid '%s'.\n", uid);

    /* turn off auto-commit */

    if (ocof (&tpclda)) {
      errrpt (&tpclda, &tpclda);
      ologof (&tpclda);
      exit (1);
    }

    /* open cursors */

    if (oopen (&curw, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
      errrpt (&tpclda, &curw);
      ologof (&tpclda);
      exit (1);
    }

    if (oopen (&curd, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
      errrpt (&tpclda, &curd);
      oclose (&curw);
      ologof (&tpclda);
      exit (1);
    }

    if (oopen (&curc, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
      errrpt (&tpclda, &curc);
      oclose (&curw);
      oclose (&curd);
      ologof (&tpclda);
      exit (1);
    }

    if (oopen (&curh, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
      errrpt (&tpclda, &curh);
      oclose (&curw);
      oclose (&curd);
      oclose (&curc);
      ologof (&tpclda);
      exit (1);
    }

    if (oopen (&curs, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
      errrpt (&tpclda, &curs);
      oclose (&curw);
      oclose (&curd);
```

```
    oclose (&curc);
    oclose (&curh);
    ologof (&tpclda);
    exit (1);
  }

  if (oopen (&curi, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
    errrpt (&tpclda, &curi);
    oclose (&curw);
    oclose (&curd);
    oclose (&curc);
    oclose (&curh);
    oclose (&curs);
    ologof (&tpclda);
    exit (1);
  }

  if (oopen (&curo1, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
    errrpt (&tpclda, &curo1);
    oclose (&curw);
    oclose (&curd);
    oclose (&curc);
    oclose (&curh);
    oclose (&curs);
    oclose (&curi);
    ologof (&tpclda);
    exit (1);
  }

  if (oopen (&curo2, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
    errrpt (&tpclda, &curo2);
    oclose (&curw);
    oclose (&curd);
    oclose (&curc);
    oclose (&curh);
    oclose (&curs);
    oclose (&curi);
    oclose (&curo1);
    ologof (&tpclda);
    exit (1);
  }

  if (oopen (&curol1, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
    errrpt (&tpclda, &curol1);
    oclose (&curw);
    oclose (&curd);
    oclose (&curc);
    oclose (&curh);
    oclose (&curs);
    oclose (&curi);
    oclose (&curo1);
    oclose (&curo2);
    ologof (&tpclda);
    exit (1);
  }

  if (oopen (&curol2, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
    errrpt (&tpclda, &curol2);
    oclose (&curw);
    oclose (&curd);
    oclose (&curc);
    oclose (&curh);
    oclose (&curs);
    oclose (&curi);
    oclose (&curo1);
    oclose (&curo2);
    oclose (&curol1);
    ologof (&tpclda);
    exit (1);
  }

  if (oopen (&curno, &tpclda, (text *) 0, -1, -1, (text *) uid, -1)) {
    errrpt (&tpclda, &curno);
    oclose (&curw);
    oclose (&curd);
    oclose (&curc);
    oclose (&curh);
    oclose (&curs);
    oclose (&curi);
    oclose (&curo1);
    oclose (&curo2);
    oclose (&curol1);
    oclose (&curol2);
    ologof (&tpclda);
    exit (1);
  }

  /* parse statements */

  sprintf ((char *) sqlbuf, SQLTXTW);
  if (oparse (&curw, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curw);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTD);
  if (oparse (&curd, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curd);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTC);
  if (oparse (&curc, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curc);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTH);
  if (oparse (&curh, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curh);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTS);
  if (oparse (&curs, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curs);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTI);
  if (oparse (&curi, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curi);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTO1);
  if (oparse (&curo1, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curo1);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTO2);
  if (oparse (&curo2, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curo2);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTOL1);
  if (oparse (&curol1, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curol1);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTOL2);
  if (oparse (&curol2, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curol2);
    quit ();
    exit (1);
  }

  sprintf ((char *) sqlbuf, SQLTXTNO);
  if (oparse (&curno, sqlbuf, -1, 0, 1)) {
    errrpt (&tpclda, &curno);
    quit ();
    exit (1);
  }

  /* bind variables */

  /* ware */

  if (obndrv (&curw, (text *) ":w_id", -1, (ub1 *) &w_id, sizeof (w_id),
          SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errrpt (&tpclda, &curw);
    quit ();
    exit (1);
  }

  if (obndrv (&curw, (text *) ":w_name", -1, (ub1 *) w_name, 11,
          SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errrpt (&tpclda, &curw);
    quit ();
    exit (1);
  }

  if (obndrv (&curw, (text *) ":w_street_1", -1, (ub1 *) w_street_1, 21,
          SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errrpt (&tpclda, &curw);
    quit ();
    exit (1);
```

```
}

if (obndrv (&curw, (text *) ":w_street_2", -1, (ub1 *) w_street_2, 21,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curw);
  quit ();
  exit (1);
}

if (obndrv (&curw, (text *) ":w_city", -1, (ub1 *) w_city, 21,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curw);
  quit ();
  exit (1);
}

if (obndrv (&curw, (text *) ":w_state", -1, (ub1 *) w_state, 2,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curw);
  quit ();
  exit (1);
}

if (obndrv (&curw, (text *) ":w_zip", -1, (ub1 *) w_zip, 9,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curw);
  quit ();
  exit (1);
}

if (obndrv (&curw, (text *) ":w_tax", -1, (ub1 *) &w_tax, sizeof (w_tax),
        SQLT_FLT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curw);
  quit ();
  exit (1);
}

/* dist */

if (obndrv (&curd, (text *) ":d_id", -1, (ub1 *) d_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curd);
  quit ();
  exit (1);
}

if (obndrv (&curd, (text *) ":d_w_id", -1, (ub1 *) d_w_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curd);
  quit ();
  exit (1);
}

if (obndrv (&curd, (text *) ":d_name", -1, (ub1 *) d_name, 11,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curd);
  quit ();
  exit (1);
}

if (obndrv (&curd, (text *) ":d_street_1", -1, (ub1 *) d_street_1, 21,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curd);
  quit ();
  exit (1);
}

if (obndrv (&curd, (text *) ":d_street_2", -1, (ub1 *) d_street_2, 21,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curd);
  quit ();
  exit (1);
}

if (obndrv (&curd, (text *) ":d_city", -1, (ub1 *) d_city, 21,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curd);
  quit ();
  exit (1);
}

if (obndrv (&curd, (text *) ":d_state", -1, (ub1 *) d_state, 2,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curd);
  quit ();
  exit (1);
}

if (obndrv (&curd, (text *) ":d_zip", -1, (ub1 *) d_zip, 9,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curd);
  quit ();
  exit (1);
}

if (obndrv (&curd, (text *) ":d_tax", -1, (ub1 *) d_tax, sizeof (float),
        SQLT_FLT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curd);
  quit ();
  exit (1);
}

/* cust */

if (obndrv (&curc, (text *) ":c_id", -1, (ub1 *) c_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt ( &tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_d_id", -1, (ub1 *) c_d_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_w_id", -1, (ub1 *) c_w_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_first", -1, (ub1 *) c_first, 17,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_last", -1, (ub1 *) c_last, 17,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_street_1", -1, (ub1 *) c_street_1, 21,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_street_2", -1, (ub1 *) c_street_2, 21,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_city", -1, (ub1 *) c_city, 21,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_state", -1, (ub1 *) c_state, 2,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_zip", -1, (ub1 *) c_zip, 9,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_phone", -1, (ub1 *) c_phone, 16,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_credit", -1, (ub1 *) c_credit, 2,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
  quit ();
  exit (1);
}

if (obndrv (&curc, (text *) ":c_discount", -1, (ub1 *) c_discount,
        sizeof (float), SQLT_FLT, -1, (sb2 *) 0, (text *) 0, -1,
```

```
        -1)) {
  errrpt (&tpclda, &curc);
   quit ();
   exit (1);
}

if (obndrv (&curc, (text *) ":c_data", -1, (ub1 *) c_data, 501,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curc);
   quit ();
   exit (1);
}

/* item */

if (obndrv (&curi, (text *) ":i_id", -1, (ub1 *) i_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curi);
   quit ();
   exit (1);
}

if (obndrv (&curi, (text *) ":i_im_id", -1, (ub1 *) i_im_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curi);
   quit ();
   exit (1);
}

if (obndrv (&curi, (text *) ":i_name", -1, (ub1 *) i_name, 25,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curi);
   quit ();
   exit (1);
}

if (obndrv (&curi, (text *) ":i_price", -1, (ub1 *) i_price,
        sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1,
        -1)) {
  errrpt (&tpclda, &curi);
   quit ();
   exit (1);
}

if (obndrv (&curi, (text *) ":i_data", -1, (ub1 *) i_data, 51,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curi);
   quit ();
   exit (1);
}

/* stok */

if (obndrv (&curs, (text *) ":s_i_id", -1, (ub1 *) s_i_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_w_id", -1, (ub1 *) s_w_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_quantity", -1, (ub1 *) s_quantity,
        sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_01", -1, (ub1 *) s_dist_01, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_02", -1, (ub1 *) s_dist_02, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_03", -1, (ub1 *) s_dist_03, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_04", -1, (ub1 *) s_dist_04, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_05", -1, (ub1 *) s_dist_05, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_06", -1, (ub1 *) s_dist_06, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_07", -1, (ub1 *) s_dist_07, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_08", -1, (ub1 *) s_dist_08, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_09", -1, (ub1 *) s_dist_09, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_dist_10", -1, (ub1 *) s_dist_10, 24,
        SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

if (obndrv (&curs, (text *) ":s_data", -1, (ub1 *) s_data, 51,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curs);
   quit ();
   exit (1);
}

/* hist */

if (obndrv (&curh, (text *) ":h_c_id", -1, (ub1 *) h_c_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curh);
   quit ();
   exit (1);
}

if (obndrv (&curh, (text *) ":h_c_d_id", -1, (ub1 *) h_d_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curh);
   quit ();
   exit (1);
}

if (obndrv (&curh, (text *) ":h_c_w_id", -1, (ub1 *) h_w_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curh);
   quit ();
   exit (1);
}

if (obndrv (&curh, (text *) ":h_d_id", -1, (ub1 *) h_d_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curh);
   quit ();
   exit (1);
}

if (obndrv (&curh, (text *) ":h_w_id", -1, (ub1 *) h_w_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curh);
   quit ();
   exit (1);
}

if (obndrv (&curh, (text *) ":h_data", -1, (ub1 *) h_data, 25,
        SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
```

```
  errrpt (&tpclda, &curh);
  quit ();
  exit (1);
}

/* ordl (delivered) */

if (obndrv (&curol1, (text *) ":ol_o_id", -1, (ub1 *) ol_o_id,
        sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol1);
  quit ();
  exit (1);
}

if (obndrv (&curol1, (text *) ":ol_d_id", -1, (ub1 *) ol_d_id,
        sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol1);
  quit ();
  exit (1);
}

if (obndrv (&curol1, (text *) ":ol_w_id", -1, (ub1 *) ol_w_id,
        sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol1);
  quit ();
  exit (1);
}

if (obndrv (&curol1, (text *) ":ol_number", -1, (ub1 *) ol_number,
    sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol1);
  quit ();
  exit (1);
}

if (obndrv (&curol1, (text *) ":ol_i_id", -1, (ub1 *) ol_i_id,
    sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol1);
  quit ();
  exit (1);
}

if (obndrv (&curol1, (text *) ":ol_supply_w_id", -1,
        (ub1 *) ol_supply_w_id, sizeof (int), SQLT_INT, -1,
        (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol1);
  quit ();
  exit (1);
}

if (obndrv (&curol1, (text *) ":ol_dist_info", -1, (ub1 *) ol_dist_info,
        24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol1);
  quit ();
  exit (1);
}

/* ordl (not delivered) */

if (obndrv (&curol2, (text *) ":ol_o_id", -1, (ub1 *) ol_o_id,
        sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol2);
  quit ();
  exit (1);
}

if (obndrv (&curol2, (text *) ":ol_d_id", -1, (ub1 *) ol_d_id,
        sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol2);
  quit ();
  exit (1);
}

if (obndrv (&curol2, (text *) ":ol_w_id", -1, (ub1 *) ol_w_id,
        sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol2);
  quit ();
  exit (1);
}

if (obndrv (&curol2, (text *) ":ol_number", -1, (ub1 *) ol_number,
    sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol2);
  quit ();
  exit (1);
}

if (obndrv (&curol2, (text *) ":ol_i_id", -1, (ub1 *) ol_i_id,
    sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol2);
  quit ();
  exit (1);
}

if (obndrv (&curol2, (text *) ":ol_supply_w_id", -1,
        (ub1 *) ol_supply_w_id, sizeof (int), SQLT_INT, -1,
        (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol2);
  quit ();
  exit (1);
}

if (obndrv (&curol2, (text *) ":ol_amount", -1, (ub1 *) ol_amount,
    sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol2);
  quit ();
  exit (1);
}

if (obndrv (&curol2, (text *) ":ol_dist_info", -1, (ub1 *) ol_dist_info,
        24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curol2);
  quit ();
  exit (1);
}

/* ordr (delivered) */

if (obndrv (&curo1, (text *) ":o_id", -1, (ub1 *) o_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo1);
  quit ();
  exit (1);
}

if (obndrv (&curo1, (text *) ":o_d_id", -1, (ub1 *) o_d_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo1);
  quit ();
  exit (1);
}

if (obndrv (&curo1, (text *) ":o_w_id", -1, (ub1 *) o_w_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo1);
  quit ();
  exit (1);
}

if (obndrv (&curo1, (text *) ":o_c_id", -1, (ub1 *) o_c_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo1);
  quit ();
  exit (1);
}

if (obndrv (&curo1, (text *) ":o_carrier_id", -1, (ub1 *) o_carrier_id,
    sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo1);
  quit ();
  exit (1);
}

if (obndrv (&curo1, (text *) ":o_ol_cnt", -1, (ub1 *) o_ol_cnt,
    sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo1);
  quit ();
  exit (1);
}

/* ordr (not delivered) */

if (obndrv (&curo2, (text *) ":o_id", -1, (ub1 *) o_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo2);
  quit ();
  exit (1);
}

if (obndrv (&curo2, (text *) ":o_d_id", -1, (ub1 *) o_d_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo2);
  quit ();
  exit (1);
}

if (obndrv (&curo2, (text *) ":o_w_id", -1, (ub1 *) o_w_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo2);
  quit ();
  exit (1);
}

if (obndrv (&curo2, (text *) ":o_c_id", -1, (ub1 *) o_c_id, sizeof (int),
        SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
  errrpt (&tpclda, &curo2);
  quit ();
  exit (1);
}
```

```
    if (obndrv (&curo2, (text *) ":o_ol_cnt", -1, (ub1 *) o_ol_cnt,
            sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
      errrpt (&tpclda, &curo2);
      quit ();
      exit (1);
    }

    /* new order */

    if (obndrv (&curno, (text *) ":no_o_id", -1, (ub1 *) no_o_id,
            sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
      errrpt (&tpclda, &curno);
      quit ();
      exit (1);
    }

    if (obndrv (&curno, (text *) ":no_d_id", -1, (ub1 *) no_d_id,
            sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
      errrpt (&tpclda, &curno);
      quit ();
      exit (1);
    }

    if (obndrv (&curno, (text *) ":no_w_id", -1, (ub1 *) no_w_id,
            sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
      errrpt (&tpclda, &curno);
      quit ();
      exit (1);
    }
  }

/*---------------------------------------------------------------+
 | 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 ware: 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 %f %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);

        if (oexec (&curw)) {
          errrpt (&tpclda, &curw);
          orol (&tpclda);
          fprintf (stderr, "Aborted at ware %d\n", loop);
          quit ();
          exit (1);
        }
        else if (ocom (&tpclda)) {
          errrpt (&tpclda, &tpclda);
          orol (&tpclda);
          fprintf (stderr, "Aborted at ware %d\n", loop);
          quit ();
          exit (1);
        }
      }
    }

    end_time = gettime ();
    end_cpu = getcpu ();
    fprintf (stderr, "Done.  %d rows loaded/generated in %10d sec. (%10d 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 dist: 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 %s %s %s %s %s %s %d 30000.0 3001\n",
                i + 1, dwid, d_name[i], d_street_1[i], d_street_2[i],
                d_city[i], str2, num9, d_tax[i]); */
                                    /* Reordered columns */
          printf ("%d %d 3000000 %f 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 {
        if (oexn (&curd, DISTARR, 0)) {
          errrpt (&tpclda, &curd);
          orol (&tpclda);
          fprintf (stderr, "Aborted at ware %d, dist 1\n", dwid);
          quit ();
          exit (1);
        }
        else if (ocom (&tpclda)) {
          errrpt (&tpclda, &tpclda);
          orol (&tpclda);
          fprintf (stderr, "Aborted at ware %d, dist 1\n", dwid);
          quit ();
          exit (1);
        }
      }
    }

    end_time = gettime ();
    end_cpu = getcpu ();
    fprintf (stderr, "Done.  %d rows loaded/generated in %10d sec. (%10d 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 cust: 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 < 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 %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 {
        if (oexn (&curc, CUSTARR, 0)) {
          errrpt (&tpclda, &curc);
          orol (&tpclda);
          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]);
          quit ();
          exit (1);
        }
        else if (ocom (&tpclda)) {
          errrpt (&tpclda, &tpclda);
          orol (&tpclda);
          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]);
          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 %10d sec. (%10d cpu)\n\n",
             nrows, end_time - begin_time, end_cpu - begin_cpu);
  }


/*---------------------------------------------------------------- +
 | 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 {
        if (oexn (&curi, ITEMARR, 0)) {
          errrpt (&tpclda, &curi);
          orol (&tpclda);
          fprintf (stderr, "Aborted at i_id %d\n", i_id[0]);
          quit ();
          exit (1);
        }
        else if (ocom (&tpclda)) {
          errrpt (&tpclda, &tpclda);
          orol (&tpclda);
          fprintf (stderr, "Aborted at i_id %d\n", i_id[0]);
          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 %10d sec. (%10d 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 stok: 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; ) {
      for (i = 0; i < STOCARR; 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 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 {
        if (oexn (&curs, STOCARR, 0)) {
          errrpt (&tpclda, &curs);
          orol (&tpclda);
          fprintf (stderr, "Aborted at w_id %d, s_i_id %d\n", s_w_id[0],
                s_i_id[0]);
          quit ();
          exit (1);
        }
        else if (ocom (&tpclda)) {
          errrpt (&tpclda, &tpclda);
          orol (&tpclda);
          fprintf (stderr, "Aborted at w_id %d, s_i_id %d\n", s_w_id[0],
                s_i_id[0]);
          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 %10d sec. (%10d 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 stok: 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 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 {
        if (oexn (&curs, STOCARR, 0)) {
          errrpt (&tpclda, &curs);
          orol (&tpclda);
          fprintf (stderr, "Aborted at w_id %d, s_i_id %d\n", s_w_id[0],
                s_i_id[0]);
          quit ();
          exit (1);
        }
        else if (ocom (&tpclda)) {
          errrpt (&tpclda, &tpclda);
          orol (&tpclda);
          fprintf (stderr, "Aborted at w_id %d, s_i_id %d\n", s_w_id[0],
                s_i_id[0]);
          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 %10d sec. (%10d 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 hist: 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 dist cycle */
          if (cdid > DISTFAC) {
            cdid = 1;
            cwid++;                 /* shift ware 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 %s 1000 %s\n", cid, cdid, cwid, cdid,
                cwid, sdate, h_data[i]);
        }
      }

      if (gen) {
        fflush (stdout);
      }
      else {
        if (oexn (&curh, HISTARR, 0)) {
          errrpt (&tpclda, &curh);
          orol (&tpclda);
```

```
            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]);
            quit ();
            exit (1);
          }
          else if (ocom (&tpclda)) {
            errrpt (&tpclda, &tpclda);
            orol (&tpclda);
            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]);
            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 %10d sec. (%10d cpu)\n\n",
            nrows, end_time - begin_time, end_cpu - begin_cpu);
    }

/*-------------------------------------------------------------------+
 | Load the ORDERS and ORDER-LINE table.                             |
 +-------------------------------------------------------------------*/

    if (do_A || do_o) {
      nrows = (eware - bware + 1) * ORDEFAC * DISTFAC;

      fprintf (stderr, "Loading/generating ordr 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; ) {
        for (i = 0; i < ORDEARR; i++, row++) {
          cid++;
          if (cid > ORDEFAC) {       /* cycle cust id */
            cid = 1;                 /* cheap mod */
            cdid++;                  /* shift dist cycle */
            if (cdid > DISTFAC) {
              cdid = 1;
              cwid++;                /* shift ware 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];
          }

          for (j = 0; j < o_ol_cnt[i]; j++) {
            ol_i_id[j] = sid = lrand48 () % 100000 + 1;
            if (cid < 2101)
              ol_amount[j] = 0;
            else
              ol_amount[j] = (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[j], cwid,
                      ol_amount[j], 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[j], cwid,
                      ol_amount[j], str24[j]);
              }
            }
            else {
              ol_o_id[j] = cid;
              ol_d_id[j] = cdid;
              ol_w_id[j] = cwid;
              ol_number[j] = j + 1;
              ol_supply_w_id[j] = cwid;
              strncpy (ol_dist_info[j], str24[j], 24);
            }
          }

          if (gen) {
            fflush (olfp);
          }
          else {
            if (cid < 2101) {
              if (oexn (&curol1, olcnt, 0)) {
                errrpt (&tpclda, &curol1);
                orol (&tpclda);
                fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                      cwid, cdid, cid);
                quit ();
                exit (1);
              }
              else if (ocom (&tpclda)) {
                errrpt (&tpclda, &tpclda);
                orol (&tpclda);
                fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                      cwid, cdid, cid);
                quit ();
                exit (1);
              }
            }
            else {
              if (oexn (&curol2, olcnt, 0)) {
                errrpt (&tpclda, &curol2);
                orol (&tpclda);
                fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                      cwid, cdid, cid);
                quit ();
                exit (1);
              }
              else if (ocom (&tpclda)) {
                errrpt (&tpclda, &tpclda);
                orol (&tpclda);
                fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                      cwid, cdid, cid);
                quit ();
                exit (1);
              }
            }
          }
        }

        if (gen) {
          fflush (stdout);
        }
        else {
          if (cid < 2101) {
            if (oexn (&curo1, ORDEARR, 0)) {
              errrpt (&tpclda, &curo1);
              orol (&tpclda);
              fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n   ",
                    cwid, cdid, cid);
              quit ();
              exit (1);
            }
            else if (ocom (&tpclda)) {
              errrpt (&tpclda, &tpclda);
              orol (&tpclda);
              fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n   ",
                    cwid, cdid, cid);
              quit ();
              exit (1);
            }
          }
          else {
            if (oexn (&curo2, ORDEARR, 0)) {
              errrpt (&tpclda, &curo2);
              orol (&tpclda);
              fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n   ",
                    cwid, cdid, cid);
              quit ();
              exit (1);
            }
            else if (ocom (&tpclda)) {
              errrpt (&tpclda, &tpclda);
              orol (&tpclda);
              fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n   ",
                    cwid, cdid, cid);
              quit ();
              exit (1);
```

```
            }
          }
        }

        if ((++loopcount) % 50)
          fprintf (stderr, ".");
        else
          fprintf (stderr, " %d ordr committed\n   ", row);
      }

      end_time = gettime ();
      end_cpu = getcpu ();
      fprintf (stderr, "Done.  %d ordr loaded/generated in %10d sec. (%10d 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 {
          if (oexn (&curno, NEWOARR, 0)) {
            errrpt (&tpclda, &curno);
            orol (&tpclda);
            fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n   ",
                     cwid, cdid, cid + 2100);
            quit ();
            exit (1);
          }
          else if (ocom (&tpclda)) {
            errrpt (&tpclda, &tpclda);
            orol (&tpclda);
            fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n   ",
                     cwid, cdid, cid + 2100);
            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 %10d sec. (%10d 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);

}
```

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;
```

initpay.sql

```
CREATE OR REPLACE PACKAGE initpay
AS
 TYPE rowidarray IS TABLE OF ROWID INDEX BY BINARY_INTEGER;
 row_id              rowidarray;
 cust_rowid           ROWID;
 dist_name           VARCHAR2(11);
 ware_name            VARCHAR2(11);
 c_num              BINARY_INTEGER;
 PROCEDURE pay_init;
END initpay;
/
show errors;
CREATE OR REPLACE PACKAGE BODY initpay AS
  PROCEDURE pay_init IS
  BEGIN
    NULL;
  END pay_init;
END initpay;
/
show errors;
```

pay.sql

```
rem
rem
================================================================+
rem       Copyright (c) 1996  Oracle Corp, Redwood Shores, CA       |
rem                  OPEN SYSTEMS PERFORMANCE GROUP                 |
rem                     All Rights Reserved                         |
rem
================================================================+
rem  FILENAME
rem       pay.sql
rem  DESCRIPTION
rem       SQL script to create a stored procedure for payment
rem       transactions.
rem ============================================================
rem

CREATE OR REPLACE PACKAGE payment
IS
   PROCEDURE dopayment_z
   (
     ware_id               INTEGER,
     dist_id              INTEGER,
     cust_w_id             INTEGER,
     cust_d_id            INTEGER,
     cust_id          IN OUT INTEGER,
     bylastname            INTEGER,
     hist_amount           INTEGER,
     cust_last        IN OUT VARCHAR2,
     ware_street_1        OUT VARCHAR2,
     ware_street_2        OUT VARCHAR2,
     ware_city           OUT VARCHAR2,
     ware_state          OUT VARCHAR2,
     ware_zip            OUT VARCHAR2,
     dist_street_1        OUT VARCHAR2,
     dist_street_2        OUT VARCHAR2,
     dist_city           OUT VARCHAR2,
     dist_state          OUT VARCHAR2,
     dist_zip            OUT VARCHAR2,
     cust_first          OUT VARCHAR2,
     cust_middle          OUT VARCHAR2,
     cust_street_1        OUT VARCHAR2,
     cust_street_2        OUT VARCHAR2,
     cust_city           OUT VARCHAR2,
     cust_state          OUT VARCHAR2,
     cust_zip            OUT VARCHAR2,
     cust_phone           OUT VARCHAR2,
     cust_since          OUT VARCHAR2,
     cust_credit       IN OUT VARCHAR2,
     cust_credit_lim       OUT NUMBER,
     cust_discount         OUT NUMBER,
     cust_balance      IN OUT NUMBER,
     cust_data           OUT VARCHAR2,
     hist_date           OUT VARCHAR2,
     retry            IN OUT INTEGER,
     cur_date                IN DATE
   );
```

```
PROCEDURE dopayment_nz
  (
    ware_id               INTEGER,
    dist_id               INTEGER,
    cust_w_id             INTEGER,
    cust_d_id             INTEGER,
    cust_id          IN OUT INTEGER,
    bylastname            INTEGER,
    hist_amount           INTEGER,
    cust_last        IN OUT VARCHAR2,
    ware_street_1         OUT VARCHAR2,
    ware_street_2         OUT VARCHAR2,
    ware_city             OUT VARCHAR2,
    ware_state            OUT VARCHAR2,
    ware_zip              OUT VARCHAR2,
    dist_street_1         OUT VARCHAR2,
    dist_street_2         OUT VARCHAR2,
    dist_city             OUT VARCHAR2,
    dist_state            OUT VARCHAR2,
    dist_zip              OUT VARCHAR2,
    cust_first            OUT VARCHAR2,
    cust_middle           OUT VARCHAR2,
    cust_street_1         OUT VARCHAR2,
    cust_street_2         OUT VARCHAR2,
    cust_city             OUT VARCHAR2,
    cust_state            OUT VARCHAR2,
    cust_zip              OUT VARCHAR2,
    cust_phone            OUT VARCHAR2,
    cust_since            OUT VARCHAR2,
    cust_credit      IN OUT VARCHAR2,
    cust_credit_lim       OUT NUMBER,
    cust_discount         OUT NUMBER,
    cust_balance     IN OUT NUMBER,
    cust_data             OUT VARCHAR2,
    hist_date             OUT VARCHAR2,
    retry            IN OUT INTEGER,
    cur_date                 IN DATE
  );
END;
/
show errors;

CREATE OR REPLACE PACKAGE BODY payment
IS
  PROCEDURE dopayment_z
  (
    ware_id               INTEGER,
    dist_id               INTEGER,
    cust_w_id             INTEGER,
    cust_d_id             INTEGER,
    cust_id          IN OUT INTEGER,
    bylastname            INTEGER,
    hist_amount           INTEGER,
    cust_last        IN OUT VARCHAR2,
    ware_street_1         OUT VARCHAR2,
    ware_street_2         OUT VARCHAR2,
    ware_city             OUT VARCHAR2,
    ware_state            OUT VARCHAR2,
    ware_zip              OUT VARCHAR2,
    dist_street_1         OUT VARCHAR2,
    dist_street_2         OUT VARCHAR2,
    dist_city             OUT VARCHAR2,
    dist_state            OUT VARCHAR2,
    dist_zip              OUT VARCHAR2,
    cust_first            OUT VARCHAR2,
    cust_middle           OUT VARCHAR2,
    cust_street_1         OUT VARCHAR2,
    cust_street_2         OUT VARCHAR2,
    cust_city             OUT VARCHAR2,
    cust_state            OUT VARCHAR2,
    cust_zip              OUT VARCHAR2,
    cust_phone            OUT VARCHAR2,
    cust_since            OUT VARCHAR2,
    cust_credit      IN OUT VARCHAR2,
    cust_credit_lim       OUT NUMBER,
    cust_discount         OUT NUMBER,
    cust_balance     IN OUT NUMBER,
    cust_data             OUT VARCHAR2,
    hist_date             OUT VARCHAR2,
    retry            IN OUT INTEGER,
    cur_date                 IN DATE
  )
  IS
    TYPE rowidarray IS TABLE OF ROWID INDEX BY BINARY_INTEGER;
    cust_rowid            ROWID;
    ware_rowid            ROWID;
    dist_ytd                 NUMBER(12);
    dist_name             VARCHAR2(11);
    ware_ytd                 NUMBER(12);
    ware_name             VARCHAR2(11);
    history_date          DATE;
    c_num                 BINARY_INTEGER;
    row_id                rowidarray;
    cust_payments            PLS_INTEGER;
    cust_ytd                 NUMBER(12);
    cust_data_temp           VARCHAR2(500);
    node_num                 VARCHAR2(512);
    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);
    CURSOR c_cur IS
                  SELECT rowid
      FROM cust
      WHERE c_d_id = cust_d_id AND c_w_id = cust_w_id AND c_last = cust_last
      ORDER BY c_w_id, c_d_id, c_last, c_first;
  BEGIN
    SELECT substr(value,1,5)
      INTO node_num
      FROM v$parameter
      WHERE name = 'instance_number';

    plsql_mon_pack.print ('Payment started at ' ||
                    to_char (sysdate, 'HH24:MI:SS') || ' on node ' ||
                    node_num);
    LOOP BEGIN
      SELECT rowid, c_first, c_middle, c_last, c_street_1, c_street_2,
          c_city, c_state, c_zip, c_phone,
          to_char (c_since, 'DD-MM-YYYY'), c_credit, c_credit_lim,
          c_discount, c_balance - hist_amount, c_payment_cnt ,
                                  c_ytd_payment + hist_amount, decode (c_credit, 'BC', c_data, ' ')
        INTO cust_rowid, cust_first, cust_middle, cust_last,
          cust_street_1, cust_street_2, cust_city, cust_state,
          cust_zip, cust_phone, cust_since, cust_credit,
          cust_credit_lim, cust_discount, cust_balance, cust_payments,
                                  cust_ytd, cust_data_temp
        FROM cust
        WHERE c_id = cust_id AND c_d_id = cust_d_id AND
            c_w_id = cust_w_id;
      cust_payments := cust_payments + 1;
      IF cust_credit = 'BC' THEN
                  cust_data_temp := substr ((to_char (cust_id) || ' ' ||
                        to_char (cust_d_id) || ' ' ||
                        to_char (cust_w_id) || ' ' ||
                        to_char (dist_id) || ' ' ||
                        to_char (ware_id) || ' ' ||
                        to_char (hist_amount, '9999.99') || ' | ')
                        || cust_data_temp, 1, 500);

        UPDATE cust
          SET c_balance = cust_balance,
            c_ytd_payment = cust_ytd,
            c_payment_cnt = cust_payments,
                                  c_data = cust_data_temp
          WHERE rowid = cust_rowid;

                  cust_data := substr(cust_data_temp,1, 200);

      ELSE

        UPDATE cust
          SET c_balance = cust_balance,
            c_ytd_payment = cust_ytd,
            c_payment_cnt = cust_payments
          WHERE rowid = cust_rowid;

                  cust_data := cust_data_temp;
      END IF;

      SELECT dist.rowid, d_name, d_street_1, d_street_2, d_city,
                    d_state, d_zip, d_ytd + hist_amount,
          ware.rowid, w_name, w_street_1, w_street_2, w_city,
          w_state, w_zip, w_ytd + hist_amount
        INTO cust_rowid, dist_name, dist_street_1, dist_street_2, dist_city,
          dist_state, dist_zip, dist_ytd,
          ware_rowid, ware_name, ware_street_1, ware_street_2, ware_city,
          ware_state, ware_zip, ware_ytd
        FROM dist, ware
        WHERE d_id = dist_id
          AND d_w_id = ware_id
          AND w_id = ware_id;

      UPDATE dist
        SET d_ytd = dist_ytd
        WHERE rowid = cust_rowid;

      UPDATE ware
        SET w_ytd = ware_ytd
        WHERE rowid = ware_rowid;

      history_date := cur_date;

      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
        (cust_id, cust_d_id, cust_w_id, dist_id, ware_id, history_date,
        hist_amount, ware_name || '    ' || dist_name);
      COMMIT;
      hist_date := to_char (history_date,'DD-MM-YYYY.HH24:MI:SS');
```

```
    EXIT;

    EXCEPTION
      WHEN not_serializable OR deadlock OR snapshot_too_old THEN
        ROLLBACK;
        retry := retry + 1;
    END;

  END LOOP;
END;


PROCEDURE dopayment_nz
(
  ware_id                INTEGER,
  dist_id                INTEGER,
  cust_w_id              INTEGER,
  cust_d_id              INTEGER,
  cust_id          IN OUT INTEGER,
  bylastname             INTEGER,
  hist_amount            INTEGER,
  cust_last        IN OUT VARCHAR2,
  ware_street_1       OUT VARCHAR2,
  ware_street_2       OUT VARCHAR2,
  ware_city           OUT VARCHAR2,
  ware_state          OUT VARCHAR2,
  ware_zip            OUT VARCHAR2,
  dist_street_1       OUT VARCHAR2,
  dist_street_2       OUT VARCHAR2,
  dist_city           OUT VARCHAR2,
  dist_state          OUT VARCHAR2,
  dist_zip            OUT VARCHAR2,
  cust_first          OUT VARCHAR2,
  cust_middle         OUT VARCHAR2,
  cust_street_1       OUT VARCHAR2,
  cust_street_2       OUT VARCHAR2,
  cust_city           OUT VARCHAR2,
  cust_state          OUT VARCHAR2,
  cust_zip            OUT VARCHAR2,
  cust_phone          OUT VARCHAR2,
  cust_since          OUT VARCHAR2,
  cust_credit      IN OUT VARCHAR2,
  cust_credit_lim     OUT NUMBER,
  cust_discount       OUT NUMBER,
  cust_balance     IN OUT NUMBER,
  cust_data           OUT VARCHAR2,
  hist_date           OUT VARCHAR2,
  retry            IN OUT INTEGER,
  cur_date                IN DATE
)
IS
  TYPE rowidarray IS TABLE OF ROWID INDEX BY BINARY_INTEGER;
  cust_rowid             ROWID;
  ware_rowid             ROWID;
  dist_ytd                      NUMBER(12);
  dist_name              VARCHAR2(11);
  ware_ytd                      NUMBER(12);
  ware_name              VARCHAR2(11);
  history_date           DATE;
  c_num                  BINARY_INTEGER;
  row_id                 rowidarray;
  cust_payments                 PLS_INTEGER;
  cust_ytd                      NUMBER(12);
  cust_data_temp                VARCHAR2(500);
  node_num                      VARCHAR2(512);
  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);
  CURSOR c_cur IS
    SELECT rowid
    FROM cust
    WHERE c_d_id = cust_d_id AND c_w_id = cust_w_id AND c_last = cust_last
    ORDER BY c_w_id, c_d_id, c_last, c_first;
BEGIN
  SELECT substr(value,1,5)
    INTO node_num
    FROM v$parameter
    WHERE name = 'instance_number';

  plsql_mon_pack.print ('Payment started at ' ||
                  to_char (sysdate, 'HH24:MI:SS') || ' on node ' ||
                  node_num);
  LOOP BEGIN

    c_num := 0;
    FOR c_id_rec IN c_cur LOOP
      c_num := c_num + 1;
      row_id(c_num) := c_id_rec.rowid;
    END LOOP;
    cust_rowid := row_id ((c_num + 1) / 2);    -- use row_id.count ?

    SELECT c_id, c_first, c_middle, c_last, c_street_1, c_street_2,
      c_city, c_state, c_zip, c_phone,
      to_char (c_since, 'DD-MM-YYYY'), c_credit, c_credit_lim,
      c_discount, c_balance - hist_amount, c_payment_cnt,
                        c_ytd_payment + hist_amount, decode (c_credit, 'BC', c_data, ' ')
    INTO cust_id, cust_first, cust_middle, cust_last,
      cust_street_1, cust_street_2, cust_city, cust_state,
      cust_zip, cust_phone, cust_since, cust_credit,
      cust_credit_lim, cust_discount, cust_balance, cust_payments,
                        cust_ytd, cust_data_temp
    FROM cust
    WHERE rowid = cust_rowid;
  cust_payments := cust_payments + 1;
  IF cust_credit = 'BC' THEN
              cust_data_temp := substr ((to_char (cust_id) || ' ' ||
                    to_char (cust_d_id) || ' ' ||
                    to_char (cust_w_id) || ' ' ||
                    to_char (dist_id) || ' ' ||
                    to_char (ware_id) || ' ' ||
                    to_char (hist_amount/100, '9999.99') || ' | ')
                    || cust_data_temp, 1, 500);

    UPDATE cust
      SET c_balance = cust_balance,
        c_ytd_payment = cust_ytd,
        c_payment_cnt = cust_payments,
                        c_data = cust_data_temp
      WHERE rowid = cust_rowid;

              cust_data := substr(cust_data_temp,1, 200);

  ELSE

    UPDATE cust
      SET c_balance = cust_balance,
        c_ytd_payment = cust_ytd,
        c_payment_cnt = cust_payments
      WHERE rowid = cust_rowid;

              cust_data := cust_data_temp;

  END IF;

  SELECT dist.rowid, d_name, d_street_1, d_street_2, d_city,
              d_state, d_zip, d_ytd + hist_amount,
      ware.rowid, w_name, w_street_1, w_street_2, w_city,
      w_state, w_zip, w_ytd + hist_amount
    INTO cust_rowid, dist_name, dist_street_1, dist_street_2, dist_city,
      dist_state, dist_zip, dist_ytd,
      ware_rowid, ware_name, ware_street_1, ware_street_2, ware_city,
      ware_state, ware_zip, ware_ytd
    FROM dist, ware
    WHERE d_id = dist_id
      AND d_w_id = ware_id
      AND w_id = ware_id;

  UPDATE dist
    SET d_ytd = dist_ytd
    WHERE rowid = cust_rowid;

  UPDATE ware
    SET w_ytd = ware_ytd
    WHERE rowid = ware_rowid;

  history_date := cur_date;

  INSERT INTO hist VALUES
    (cust_id, cust_d_id, cust_w_id, dist_id, ware_id, history_date,
    hist_amount, ware_name || '    ' || dist_name);
  COMMIT;
  hist_date := to_char (history_date, 'DD-MM-YYYY.HH24:MI:SS');
  EXIT;

  EXCEPTION
    WHEN not_serializable OR deadlock OR snapshot_too_old THEN
      ROLLBACK;
      retry := retry + 1;
  END;

    END LOOP;
  END;
END;
/
show errors;

quit;
```

p_create.ora

```
compatible = 9.0.1.0.0
db_name = tpcc
control_files                          = (?/dbs/tpcc_disks/control01,
?/dbs/tpcc_disks/control02)
```

```
db_files = 400
dml_locks = 500
log_buffer = 1048576
processes =  1471
db_block_size = 2048
db_cache_size                = 100M
db_8k_cache_size             = 100M
disk_asynch_io               = FALSE
undo_management              = manual
```

addfile.sh

```
#!sh

date > step5addfile_$1.log

$SEQLPLUS tpcc/tpcc <<!
  spool step5addfile_$1.log
  set echo on
  alter tablespace $1 add datafile '$2' size $3 reuse;
  set echo off
  spool off
  exit ;
!

date >> step5addfile_$1.log
```

addts_mb.sh

```
#!sh

date > step5creates_$1.log

$SEQLPLUS tpcc/tpcc <<!
  spool step5creates_$1.log
  set echo on
  drop tablespace $1 including contents;
  create tablespace $1 datafile '$2' size $3 reuse extent management local uniform size $4 segment
space management auto blocksize $5 nologging ;
  set echo off
  spool off
  exit ;
!

date >> step5creates_$1.log
```

analyze.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step39analyze > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

create_stat.sh

```
#!sh

addts_mb.sh  tools  \?/dbs/${ORACLE_SID}_disks/tools01    20001M   1M 8K
addtempts.sh temp   \?/dbs/${ORACLE_SID}_disks/stat_temp01     8001M  10M
```

createcust.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step11createcust > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createdb.sh

```
#!sh
$SEQLPLUS 'sys/change_on_install as sysdba' @${SQLDIR}/step2createdb > junk 2 >& 1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createddviews.sh

```
#!sh
$SEQLPLUS 'sys/change_on_install as sysdba' > junk 2>&1 <<!
@${SQLDIR}/step6createddviews
!

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createdist.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step10createdist > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createhist.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step12createhist > junk  2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createicust1.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step32createicust1 > junk  2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createicust2.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step33createicust2 > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
```

```
  exit 0;
fi
wait
```

createidist.sh

```
 #!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step30createidist > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createiitem.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step31createiitem > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createinord.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step37createinord > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createiordl.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step35createiordr1 > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createiordr1.sh

```
   #!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step35createiordr1 > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createiordr2.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step36createiordr2 > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createistok.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step34createistok > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createitem.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step31createiitem > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createmisc.sh

```
#!sh
$SEQLPLUS 'sys/change_on_install as sysdba' @${SQLDIR}/step43createmisc > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createnord.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step14createnord > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createordl.sh

```
  #!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step15createordl > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createordr.sh

```
  #!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step13createordr > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createrollback.sh

```
#!sh

rb=1
while [ $rb -le 100 ]
do
$SEQLPLUS 'sys/change_on_install as sysdba' << !! >> junk 2>&1
  set echo on
  drop public rollback segment t$rb;
  alter rollback segment t$rb offline;
  create public rollback segment t$rb
      storage (initial 200K minextents 2 next 200K);
  set echo off
  exit sql.sqlcode
!!
rb=`expr $rb + 1`
done

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

createrollsegs.sh

```
  #!sh

#
# Drop all but one rollback segments used for build
#
rb=2
while [ $rb -le 100 ]
do
$SEQLPLUS 'sys/change_on_install as sysdba' << !! >> junk 2>&1
  set echo on
  alter rollback segment t$rb offline;
  drop rollback segment t$rb;
  set echo off
  exit sql.sqlcode
!!
rb=`expr $rb + 1`
done

#
# Create the tablespace for rollback segments
#

$SEQLPLUS 'sys/change_on_install as sysdba' << !! >> junk 2>&1
  drop tablespace roll_0 including contents;
  create tablespace roll_0 datafile '?/dbs/tpcc_disks/roll01' size 8001M reuse
      extent management local uniform size 40K nologging ;
  exit;
!!

#
# Create all rollback segments used for run
#
rb=1
while [ $rb -le 1600 ]
do
$SEQLPLUS 'sys/change_on_install as sysdba' << !! >> junk 2>&1
  set echo on
  alter rollback segment s$rb offline;
  drop rollback segment s$rb;
  create rollback segment s$rb tablespace roll_0;
  alter rollback segment s$rb online;
  set echo off
  exit sql.sqlcode
!!
rb=`expr $rb + 1`
done

rb=1
while [ $rb -le 150 ]
do
$SEQLPLUS 'sys/change_on_install as sysdba' << !! >> junk 2>&1
  set echo on
  alter rollback segment n$rb offline;
  drop rollback segment n$rb;
  create rollback segment n$rb tablespac e roll_0;
  alter rollback segment n$rb online;
  set echo off
  exit sql.sqlcode
!!
rb=`expr $rb + 1`
done

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
```

createspacestats.sh

```
#!sh
$SEQLPLUS 'sys/change_on_install as sysdba' @${SQLDIR}/step42createspacestats > junk
2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createstats.sh

```
#!sh

addts_mb.sh  tools  \?/dbs/${ORACLE_SID}_disks/tools01    20001M   1M 8K
addtempts.sh temp   \?/dbs/${ORACLE_SID}_disks/stat_temp01     8001M  10M
```

createstok.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step16createstok > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createstoredprocs.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step41createstoredprocs > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createts.sh

```
#!sh

addtempts.sh temp   \?/dbs/${ORACLE_SID}_disks/temp01   8001M  1000M &
addts.sh     stok   \?/dbs/${ORACLE_SID}_disks/stock001 8001M  1000M &
addts.sh     cust   \?/dbs/${ORACLE_SID}_disks/cust001  8001M  1000M &
addts_mb.sh  ordl   \?/dbs/${ORACLE_SID}_disks/ordl01   32001M 1000M 8K &
addts_mb.sh  nord   \?/dbs/${ORACLE_SID}_disks/nord01   9001M  1000M 8K &
addts_mb.sh  ordr   \?/dbs/${ORACLE_SID}_disks/ordr01   25001M 1000M 8K &
addts_mb.sh  hist   \?/dbs/${ORACLE_SID}_disks/hist01   26001M 1000M 8K &
addts_mb.sh  istk   \?/dbs/${ORACLE_SID}_disks/istk01   8001M  500M 8K &
addts_mb.sh  icust1 \?/dbs/${ORACLE_SID}_disks/icust101 8001M  500M 8K &
addts_mb.sh  icust2 \?/dbs/${ORACLE_SID}_disks/icust201 8001M  500M 8K &
addts_mb.sh  iord1  \?/dbs/${ORACLE_SID}_disks/iord101  25001M 500M 8K &
addts_mb.sh  iord2  \?/dbs/${ORACLE_SID}_disks/iord201  30001M 500M 8K &
addts.sh     misc   \?/dbs/${ORACLE_SID}_disks/misc01   4001M  500M &
addts_mb.sh  tools  \?/dbs/${ORACLE_SID}_disks/tools01  20001M  1M 8K &

wait

addft.sh     stok   \?/dbs/${ORACLE_SID}_disks/stock  8001M 164 &
addft.sh     cust   \?/dbs/${ORACLE_SID}_disks/cust   8001M  123 &
addft.sh     hist   \?/dbs/${ORACLE_SID}_disks/hist   26001M   3 &
addft.sh     istk   \?/dbs/${ORACLE_SID}_disks/istk   8001M  15 &
addft.sh     icust1 \?/dbs/${ORACLE_SID}_disks/icust1 8001M   6 &
addft.sh     icust2 \?/dbs/${ORACLE_SID}_disks/icust2 8001M  10 &
addft.sh     ordr   \?/dbs/${ORACLE_SID}_disks/ordr   25001M   2 &
addft.sh     iord1  \?/dbs/${ORACLE_SID}_disks/iord1  25001M   1 &
addft.sh     iord2  \?/dbs/${ORACLE_SID}_disks/iord2  30001M   2 &
addft.sh     ordl   \?/dbs/${ORACLE_SID}_disks/ordl   32001M  34 &
addft.sh     nord   \?/dbs/${ORACLE_SID}_disks/nord   9001M   1 &
addtempft.sh temp   \?/dbs/${ORACLE_SID}_disks/temp   8001M  59 &

wait
```

createuser.sh

```
#!sh

$SEQLPLUS  'sys/change_on_install as sysdba' @${SQLDIR}/stepcreateuser > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

createware.sh

```
#!sh
$SEQLPLUS tpcc/tpcc @${SQLDIR}/step9cre ateware > junk 2>&1

if test $? -ne 0
then
  exit 1;
else
  exit 0;
fi
wait
```

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;
```

createcust.sql

```
   spool step11createcust.log;
set echo on;
drop table cust;
drop cluster custcluster including tables;
set timing on;
create cluster custcluster (
c_id    number(5,0)
,c_d_id    number(2,0)
,c_w_id    number(5,0)
  )
  single    table
  hashkeys    960000000
  hash is   (c_id * 320000 + c_w_id * 10 + c_d_id)
  size    850
  initrans    3
  pctfree    0
  storage ( buffer_pool recycle freelists 22 freelist groups 43 )
  tablespace    cust;
create table cust (
        c_id            number(5,0),
        c_d_id          number(2,0),
        c_w_id          number(5,0),
        c_discount      number,
        c_credit        char(2),
        c_last          varchar2(16),
        c_first         varchar2(16),
        c_credit_lim    number,
        c_balance       number,
        c_ytd_payme nt  number,
        c_payment_cnt   number,
        c_delivery_cnt  number,
        c_street_1      varchar2(20),
        c_street_2      varchar2(20),
        c_city          varchar2(20),
        c_state         char(2),
        c_zip           char(9),
        c_phone         char(16),
        c_since         date,
        c_middle        char(2),
        c_data          varchar2(500)
)
cluster custcluster (c_id
,c_d_id
,c_w_id
);
spool off;
set echo off;
exit sql.sqlcode;
```

createlargerollseg.sql

```
    set echo on
drop tablespace large_rbs_ts including contents;
```

```
create tablespace large_rbs_ts datafile
'?/dbs/tpcc_disks/oldroll02' size 2001M reuse extent
management local uniform size 200M nologging ;
create public rollback segment large_rbs tablespace
large_rbs_ts storage (initial 200M next 200M minextents
2);
alter rollback segment large_rbs online;
set echo off
exit sql.sqlcode
```

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;
```

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;
```

Appendix C Tunable Parameters

The HP-UX operating system tunable parameters employed to generate the kernel for the HP 9000 Superdome Enterprise Server and the 28 HP 9000 Model C3700 clients are listed below. Included as well are the Oracle9i Database Enterprise Edition and TUXEDO 6.4 parameters.

C.1 HP-UX Configuration - Clients

Config/Client2/ostune.ver

```
****************************************
* $Source: /usr/local/kcs/sys.ROSE_800/filesets.info/CORE-KRN/RCS/generic,v $
* $Revision: 1.3.106.2 $        $Author: kcs $
* $State: Exp $                 $Locker: CRT $
* $Date: 97/07/12 21:51:58 $
*
****************************************
* Additional drivers required in every machine-type to create a complete
* system file during cold install.  This list is every driver that the
* master.d/ files do not force on the system or is not identifiable by
* ioscan.
* Other CPU-type specific files can exist for their special cases.
* see create_sysfile (1m).
****************************************
*
* Drivers/Subsystems
sba
lba
*btlan6
*btlan3
audio
side
sdisk
sctl
superio
asio0
SCentIf
hcd
hub
c720
graph3
cdfs
nfs_core
*nfs_client
*nfs_server
dlpi
inet
uipc
tun
telm
tels
netdiag1
nms
*fcgsc_lan
*fc_arp
*fcT1_fcp
*fcpmux
vxbase
lvm
lv
hpstreams
clone
strlog
sad
echo
sc
timod
tirdwr
pipedev
pipemod
ffs
ldterm
ptem
pts
ptm
pckt
diag0
diag2
dmem
dev_config
*cifs
sapic
diag1
usbd
hid
beep
maclan
token_arp
autofsc
*nfsm
rpcmod
stape

btlan
dump lvol

STRMSGSZ       65535
bufpages       8192
dnlc_hash_locks  512
create_fastlinks  1
dbc_min_pct       0
dbc_max_pct       0
default_disk_ir   1
fs_async          1
maxfiles       2048
maxfiles_lim   2048
maxdsiz        0x80000000
maxssiz        0X10000000
maxswapchunks  16384
maxuprc        16000
nproc          (300+MAXUPRC)

msgmni         (NPROC)
msgtql         (NPROC)
msgseg         (MSGMNI*2)
msgssz         512
msgmap         (MSGSEG)
msgmax         32768
msgmnb         (MSGMAX*2)

nfile          (NPROC*5)
ninode         (NPROC*5)
nflocks        4000
npty           128
nstrpty        200

semmni         32
semmns         NPROC
semmnu         (SEMMNS)
semume         4
semvmx         40960

shmmax         0X40000000
shmmni         16
shmseg         16

swapmem_on     0
unlockable_mem  1
```

C.2 HP-UX Configurtion – Server

Config/Server/ostune.ver

```
****************************************
* Source: /ux/core/kern/filesets.info/CORE-KRN/generic
* @(#)B.11.11_LR
*
****************************************
* Additional drivers required in every machine-type to create a complete
* system file during cold install.  This list is every driver that the
* master.d/ files do not force on the system or is not identifiable by
* ioscan.
* Other CPU-type specific files can exist for their special cases.
* see create_sysfile (1m).
****************************************
*
* Drivers/Subsystems
cell
sba
lba
asio0
btlan
c720
sdisk
sctl
td
```

```
stape
gelan
cdfs
cxperf
diag0
diag1
diag2
dmem
dev_config
iomem
nfs_core
nfs_client
nfs_server
maclan
dlpi
token_arp
inet
uipc
tun
telm
tels
netdiag1
nms
hpstreams
clone
strlog
sad
echo
sc
timod
tirdwr
pipedev
pipemod
ffs
ldterm
ptem
pts
ptm
pckt
vxfs
vxportal
lvm
lv
nfsm
rpcmod
autofsc
cachefsc
cifs
fddi4
GSCtoPCI
iop_drv
bs_osm
hd_fabric
tape2
olar_psm
olar_psm_if
dev_olar
STRMSGSZ                65535
nstrpty                 60
dump lvol

asyncdsk
* coke
*
maxfiles          2048
maxfiles_lim      2048
nflocks           2048
fs_async          0
bufpages          25000
eqmemsize         10000
maxusers          1024
maxuprc           3084
max_asy nc_ports  3200
nproc             3200
nhtbl_scale       1
maxswapchunks     16384
swchunk           16384
nfile             600000
ninode            30000
npty              10
shmmni            104
semmni            10240
semmns            20480
semmnu            2548
semvmx            32768
shmmax            0x4000000000
shmseg            16
maxssiz           0x10000000
maxdsiz           0x30000000
maxssiz_64bit     0x300000000
maxdsiz_64bit     0x300000000
timezone          480
maxvgs            128
msgmax            32768
msgmnb            32768
msgmni            50
msgseg            7168
msgssz            8
msgtql            256
unlockable_mem    1
swapmem_on        0
```

C.3 Oracle9i Database Enterprise Edition Parameters

Config/Server/dbtune.ver

```
_log_simultaneous_copies        = 64
_redo_buffer_strands            = 4
_disable_incremental_checkpoints  = true
control_files                   = (?/dbs/tpcc_disks/control01, ?/dbs/tpcc_disks/control02)
db_block_checksum               = false
_lgwr_async_io                  = false
timed_statistics                = false
db_writer_processes             = 10
parallel_max_servers            = 700
cpu_count                       = 64
disk_asynch_io                  = TRUE
lock_sga                        = false
compatible                      = 9.0.1.0.0
db_name                         = tpcc
instance_name                   = tpcc
db_files                        = 390
db_block_size                   = 2048
db_8k_cache_size                = 66000M
db_cache_size                   = 3000M
db_keep_cache_size              = 166000M
db_recycle_cache_size           = 2000M
_db_block_max_dirty_target      = 61000000
_db_writer_chunk_writes         = 1000
_db_writer_max_writes           = 1000
dml_locks                       = 500
enqueue_resources               = 60000
hash_join_enabled               = FALSE
log_archive_start               = FALSE
log_checkpoint_timeout          = 0
log_checkpoint_interval         = 1000000000
log_checkpoints_to_alert        = TRUE
log_buffer                      = 33554432
open_cursors                    = 2000
processes                       = 2000
sessions                        = 2000
transactions                    = 6000
distributed_transactions        = 0
shared_pool_size                = 700M
cursor_space_for_time           = TRUE
transaction_auditing            = FALSE
replication_dependency_tracking   = FALSE
db_block_checking               = FALSE
max_dump_file_size              = 10K
transactions_per_rollback_segment = 1
max_rollback_segments           = 2000
rollback_segments               = (large_rbs,s1,s2,s3,s4,s5,s6,s7,s8,s9,
s10,s11,s12,s13,s14,s15,s16,s17,s18,s19,
s20,s21,s22,s23,s24,s25,s26,s27,s28,s29,
s30,s31,s32,s33,s34,s35,s36,s37,s38,s39,
s40,s41,s42,s43,s44,s45,s46,s47,s48,s49,
s50,s51,s52,s53,s54,s55,s56,s57,s58,s59,
s60,s61,s62,s63,s64,s65,s66,s67,s68,s69,
s70,s71,s72,s73,s74,s75,s76,s77,s78,s79,
s80,s81,s82,s83,s84,s85,s86,s87,s88,s89,
s90,s91,s92,s93,s94,s95,s96,s97,s98,s99,
s100,s101,s102,s103,s104,s105,s106,s107,s108,s109,
s110,s111,s112,s113,s114,s115,s116,s117,s118,s119,
s120,s121,s122,s123,s124,s125,s126,s127,s128,s129,
s130,s131,s132,s133,s134,s135,s136,s137,s138,s139,
s140,s141,s142,s143,s144,s145,s146,s147,s148,s149,
s150,s151,s152,s153,s154,s155,s156,s157,s158,s159,
s160,s161,s162,s163,s164,s165,s166,s167,s168,s169,
s170,s171,s172,s173,s174,s175,s176,s177,s178,s179,
s180,s181,s182,s183,s184,s185,s186,s187,s188,s189,
s190,s191,s192,s193,s194,s195,s196,s197,s198,s199,
s200)
rollback_segments          = (s201,s202,s203,s204,s205,s206,s207,s208,s209,
s210,s211,s212,s213,s214,s215,s216,s217,s218,s219,
s220,s221,s222,s223,s224,s225,s226,s227,s228,s229,
s230,s231,s232,s233,s234,s235,s236,s237,s238,s239,
s240,s241,s242,s243,s244,s245,s246,s247,s248,s249,
s250,s251,s252,s253,s254,s255,s256,s257,s258,s259,
s260,s261,s262,s263,s264,s265,s266,s267,s268,s269,
s270,s271,s272,s273,s274,s275,s276,s277,s278,s279,
s280,s281,s282,s283,s284,s285,s286,s287,s288,s289,
```

```
s290,s291,s292,s293,s294,s295,s296,s297,s298,s299,
s300,s301,s302,s303,s304,s305,s306,s307,s308,s309,
s310,s311,s312,s313,s314,s315,s316,s317,s318,s319,
s320,s321,s322,s323,s324,s325,s326,s327,s328,s329,
s330,s331,s332,s333,s334,s335,s336,s337,s338,s339,
s340,s341,s342,s343,s344,s345,s346,s347,s348,s349,
s350,s351,s352,s353,s354,s355,s356,s357,s358,s359,
s360,s361,s362,s363,s364,s365,s366,s367,s368,s369,
s370,s371,s372,s373,s374,s375,s376,s377,s378,s379,
s380,s381,s382,s383,s384,s385,s386,s387,s388,s389,
s390,s391,s392,s393,s394,s395,s396,s397,s398,s399,
s400)
rollback_segments       = (s401,s402,s403,s404,s405,s406,s407,s408,s409,
s410,s411,s412,s413,s414,s415,s416,s417,s418,s419,
s420,s421,s422,s423,s424,s425,s426,s427,s428,s429,
s430,s431,s432,s433,s434,s435,s436,s437,s438,s439,
s440,s441,s442,s443,s444,s445,s446,s447,s448,s449,
s450,s451,s452,s453,s454,s455,s456,s457,s458,s459,
s460,s461,s462,s463,s464,s465,s466,s467,s468,s469,
s470,s471,s472,s473,s474,s475,s476,s477,s478,s479,
s480,s481,s482,s483,s484,s485,s486,s487,s488,s489,
s490,s491,s492,s493,s494,s495,s496,s497,s498,s499,
s500,s501,s502,s503,s504,s505,s506,s507,s508,s509,
s510,s511,s512,s513,s514,s515,s516,s517,s518,s519,
s520,s521,s522,s523,s524,s525,s526,s527,s528,s529,
s530,s531,s532,s533,s534,s535,s536,s537,s538,s539,
s540,s541,s542,s543,s544,s545,s546,s547,s548,s549,
s550,s551,s552,s553,s554,s555,s556,s557,s558,s559,
s560,s561,s562,s563,s564,s565,s566,s567,s568,s569,
s570,s571,s572,s573,s574,s575,s576,s577,s578,s579,
s580,s581,s582,s583,s584,s585,s586,s587,s588,s589,
s590,s591,s592,s593,s594,s595,s596,s597,s598,s599,
s600)
rollback_segments       = (s601,s602,s603,s604,s605,s606,s607,s608,s609,
s610,s611,s612,s613,s614,s615,s616,s617,s618,s619,
s620,s621,s622,s623,s624,s625,s626,s627,s628,s629,
s630,s631,s632,s633,s634,s635,s636,s637,s638,s639,
s640,s641,s642,s643,s644,s645,s646,s647,s648,s649,
s650,s651,s652,s653,s654,s655,s656,s657,s658,s659,
s660,s661,s662,s663,s664,s665,s666,s667,s668,s669,
s670,s671,s672,s673,s674,s675,s676,s677,s678,s679,
s680,s681,s682,s683,s684,s685,s686,s687,s688,s689,
s690,s691,s692,s693,s694,s695,s696,s697,s698,s699,
s700,s701,s702,s703,s704,s705,s706,s707,s708,s709,
s710,s711,s712,s713,s714,s715,s716,s717,s718,s719,
s720,s721,s722,s723,s724,s725,s726,s727,s728,s729,
s730,s731,s732,s733,s734,s735,s736,s737,s738,s739,
s740,s741,s742,s743,s744,s745,s746,s747,s748,s749,
s750,s751,s752,s753,s754,s755,s756,s757,s758,s759,
s760,s761,s762,s763,s764,s765,s766,s767,s768,s769,
s770,s771,s772,s773,s774,s775,s776,s777,s778,s779,
s780,s781,s782,s783,s784,s785,s786,s787,s788,s789,
s790,s791,s792,s793,s794,s795,s796,s797,s798,s799,
s800)
rollback_segments       = (s801,s802,s803,s804,s805,s806,s807,s808,s809,
s810,s811,s812,s813,s814,s815,s816,s817,s818,s819,
s820,s821,s822,s823,s824,s825,s826,s827,s828,s829,
s830,s831,s832,s833,s834,s835,s836,s837,s838,s839,
s840,s841,s842,s843,s844,s845,s846,s847,s848,s849,
s850,s851,s852,s853,s854,s855,s856,s857,s858,s859,
s860,s861,s862,s863,s864,s865,s866,s867,s868,s869,
s870,s871,s872,s873,s874,s875,s876,s877,s878,s879,
s880,s881,s882,s883,s884,s885,s886,s887,s888,s889,
s890,s891,s892,s893,s894,s895,s896,s897,s898,s899,
s900,s901,s902,s903,s904,s905,s906,s907,s908,s909,
s910,s911,s912,s913,s914,s915,s916,s917,s918,s919,
s920,s921,s922,s923,s924,s925,s926,s927,s928,s929,
s930,s931,s932,s933,s934,s935,s936,s937,s938,s939,
s940,s941,s942,s943,s944,s945,s946,s947,s948,s949,
s950,s951,s952,s953,s954,s955,s956,s957,s958,s959,
s960,s961,s962,s963,s964,s965,s966,s967,s968,s969,
s970,s971,s972,s973,s974,s975,s976,s977,s978,s979,
s980,s981,s982,s983,s984,s985,s986,s987,s988,s989,
s990,s991,s992,s993,s994,s995,s996,s997,s998,s999,s1000)
rollback_segments       = (s1001,s1002,s1003,s1004,s1005,s1006,s1007,s1008,s1009,
s1010,s1011,s1012,s1013,s1014,s1015,s1016,s1017,s1018,s1019,
s1020,s1021,s1022,s1023,s1024,s1025,s1026,s1027,s1028,s1029,
s1030,s1031,s1032,s1033,s1034,s1035,s1036,s1037,s1038,s1039,
s1040,s1041,s1042,s1043,s1044,s1045,s1046,s1047,s1048,s1049,
s1050,s1051,s1052,s1053,s1054,s1055,s1056,s1057,s1058,s1059,
s1060,s1061,s1062,s1063,s1064,s1065,s1066,s1067,s1068,s1069,
s1070,s1071,s1072,s1073,s1074,s1075,s1076,s1077,s1078,s1079,
s1080,s1081,s1082,s1083,s1084,s1085,s1086,s1087,s1088,s1089,
s1090,s1091,s1092,s1093,s1094,s1095,s1096,s1097,s1098,s1099,
s1100,s1101,s1102,s1103,s1104,s1105,s1106,s1107,s1108,s1109,
s1110,s1111,s1112,s1113,s1114,s1115,s1116,s1117,s1118,s1119,
s1120,s1121,s1122,s1123,s1124,s1125,s1126,s1127,s1128,s1129,
s1130,s1131,s1132,s1133,s1134,s1135,s1136,s1137,s1138,s1139,
s1140,s1141,s1142,s1143,s1144,s1145,s1146,s1147,s1148,s1149,
s1150,s1151,s1152,s1153,s1154,s1155,s1156,s1157,s1158,s1159,
s1160,s1161,s1162,s1163,s1164,s1165,s1166,s1167,s1168,s1169,
s1170,s1171,s1172,s1173,s1174,s1175,s1176,s1177,s1178,s1179,
s1180,s1181,s1182,s1183,s1184,s1185,s1186,s1187,s1188,s1189,
s1190,s1191,s1192,s1193,s1194,s1195,s1196,s1197,s1198,s1199,
s1200)
rollback_segments       = (s1201,s1202,s1203,s1204,s1205,s1206,s1207,s1208,s1209,
s1210,s1211,s1212,s1213,s1214,s1215,s1216,s1217,s1218,s1219,
s1220,s1221,s1222,s1223,s1224,s1225,s1226,s1227,s1228,s1229,
s1230,s1231,s1232,s1233,s1234,s1235,s1236,s1237,s1238,s1239,
s1240,s1241,s1242,s1243,s1244,s1245,s1246,s1247,s1248,s1249,
s1250,s1251,s1252,s1253,s1254,s1255,s1256,s1257,s1258,s1259,
s1260,s1261,s1262,s1263,s1264,s1265,s1266,s1267,s1268,s1269,
s1270,s1271,s1272,s1273,s1274,s1275,s1276,s1277,s1278,s1279,
s1280,s1281,s1282,s1283,s1284,s1285,s1286,s1287,s1288,s1289,
s1290,s1291,s1292,s1293,s1294,s1295,s1296,s1297,s1298,s1299,
s1300,s1301,s1302,s1303,s1304,s1305,s1306,s1307,s1308,s1309,
s1310,s1311,s1312,s1313,s1314,s1315,s1316,s1317,s1318,s1319,
s1320,s1321,s1322,s1323,s1324,s1325,s1326,s1327,s1328,s1329,
s1330,s1331,s1332,s1333,s1334,s1335,s1336,s1337,s1338,s1339,
s1340,s1341,s1342,s1343,s1344,s1345,s1346,s1347,s1348,s1349,
s1350,s1351,s1352,s1353,s1354,s1355,s1356,s1357,s1358,s1359,
s1360,s1361,s1362,s1363,s1364,s1365,s1366,s1367,s1368,s1369,
s1370,s1371,s1372,s1373,s1374,s1375,s1376,s1377,s1378,s1379,
s1380,s1381,s1382,s1383,s1384,s1385,s1386,s1387,s1388,s1389,
s1390,s1391,s1392,s1393,s1394,s1395,s1396,s1397,s1398,s1399,
s1400)
rollback_segments       = (s1401,s1402,s1403,s1404,s1405,s1406,s1407,s1408,s1409,
s1410,s1411,s1412,s1413,s1414,s1415,s1416,s1417,s1418,s1419,
s1420,s1421,s1422,s1423,s1424,s1425,s1426,s1427,s1428,s1429,
s1430,s1431,s1432,s1433,s1434,s1435,s1436,s1437,s1438,s1439,
s1440,s1441,s1442,s1443,s1444,s1445,s1446,s1447,s1448,s1449,
s1450,s1451,s1452,s1453,s1454,s1455,s1456,s1457,s1458,s1459,
s1460,s1461,s1462,s1463,s1464,s1465,s1466,s1467,s1468,s1469,
s1470,s1471,s1472,s1473,s1474,s1475,s1476,s1477,s1478,s1479,
s1480,s1481,s1482,s1483,s1484,s1485,s1486,s1487,s1488,s1489,
s1490,s1491,s1492,s1493,s1494,s1495,s1496,s1497,s1498,s1499,
s1500,s1501,s1502,s1503,s1504,s1505,s1506,s1507,s1508,s1509,
s1510,s1511,s1512,s1513,s1514,s1515,s1516,s1517,s1518,s1519,
s1520,s1521,s1522,s1523,s1524,s1525,s1526,s1527,s1528,s1529,
s1530,s1531,s1532,s1533,s1534,s1535,s1536,s1537,s1538,s1539,
s1540,s1541,s1542,s1543,s1544,s1545,s1546,s1547,s1548,s1549,
s1550,s1551,s1552,s1553,s1554,s1555,s1556,s1557,s1558,s1559,
s1560,s1561,s1562,s1563,s1564,s1565,s1566,s1567,s1568,s1569,
s1570,s1571,s1572,s1573,s1574,s1575,s1576,s1577,s1578,s1579,
s1580,s1581,s1582,s1583,s1584,s1585,s1586,s1587,s1588,s1589,
s1590,s1591,s1592,s1593,s1594,s1595,s1596,s1597,s1598,s1599,
s1600)
rollback_segments       = ( n1,n2,n3,n4,n5,n6,n7,n8,n9,
n10,n11,n12,n13,n14,n15,n16,n17,n18,n19,
n20,n21,n22,n23,n24,n25,n26,n27,n28,n29,
n30,n31,n32,n33,n34,n35,n36,n37,n38,n39,
n40,n41,n42,n43,n44,n45,n46,n47,n48,n49,
n50,n51,n52,n53,n54,n55,n56,n57,n58,n59,
n60,n61,n62,n63,n64,n65,n66,n67,n68,n69,
n70,n71,n72,n73,n74,n75,n76,n77,n78,n79,
n80,n81,n82,n83,n84,n85,n86,n87,n88,n89,
n90,n91,n92,n93,n94,n95,n96,n97,n98,n99,
n100,n101,n102,n103,n104,n105,n106,n107,n108,n109,
n110,n111,n112,n113,n114,n115,n116,n117,n118,n119,
n120,n121,n122,n123,n124,n125,n126,n127,n128,n129,
n130,n131,n132,n133,n134,n135,n136,n137,n138,n139,
n140,n141,n142,n143,n144,n145,n146,n147,n148,n149,
n150)
```

C.4 Tuxedo UBBconfig

Config/Client2/tmcfg.ver

```
# This is a UBBconfig for a client1-server configuration.
#
# This UBBconfig requires settings for:
# SERVER_NAME CLIENT_NAME MASTER_NAME SERVER_ADDR CLIENT_ADDR
NODE_NAMES
# TLISTEN_PORT TBRIDGE_PORT
# In addition, it requires setting the things all UBBconfig.gens need:
#               IPCKEY                  some decent IPCKEY, should be different for each
config
#               ROOTDIR
#               TUXCONFIG
#               APPDIR
#               ULOGDIR
#

#------------------------------------------------------------------
*RESOURCES
#------------------------------------------------------------------
                IPCKEY      40001
                PERM        0666
                MASTER      client2

  MAXACCESSERS          11075         # 1024 or more
  MAXGTT    1024
  MAXSERVERS            55
  MAXSERVICES           260   # MAXSERVERS * #-of-services-each-server + 10( for BBL)
```

```
  MODEL     SHM
  LDBAL     Y

# During benchmark, don't want to scan too often.  In particular, while
# the client1s are stabilizing in virtual memory, we don't want to sanity
# scan; and if we do sanity scan, we want large timeouts, since the BRIDGE
# the BBL, the DBBL, and the client1s aren't getting much CPU time during that
# period.  Current settings:
#               *  scan servers every 5 minutes (maximum allowed by TUXEDO);
#               *  wait 1 minute for sanity  responses (maximum allowed by TUXEDO);
#               *  scan all the BBLs from DBBL every 30 minutes (want one scan in the
#                  audited results);
#               *  timeout a blocking call after 5 minutes (the maximum).
                SCANUNIT   60
                SANITYSCAN                  5
                DBBLWAIT   1
                BBLQUERY   30
                BLOCKTIME 5
#
#--------------------------------------------------------------------
*MACHINES
#--------------------------------------------------------------------
DEFAULT:
        TUXCONFIG="/project/iti/confs/TUXconfig.client2"
                        ROOTDIR="/project/iti"
                        APPDIR="/project/tpcc/bin"
                        ULOGPFX="/tmp/TUXEDO_LOG"

# for debugging, put both into the same log on the same machine
#                       ULOGPFX="/home/iti/confs/tpcc/ULOG"
# but for a big run, need some space, and want them local to the
# machine rather than across the net.

# Leave TUXCONFIG alone on the MASTER machine; over-ride for each
# other machine?
client2         LMID=client2
    TUXCONFIG="/project/iti/confs/TUXconfig.client2"
#--------------------------------------------------------------------
*GROUPS
#--------------------------------------------------------------------
group1          LMID=client2
            GRPNO=1
group2          LMID=client2
            GRPNO=2
group3          LMID=client2
            GRPNO=3
group4          LMID=client2
            GRPNO=4
group5          LMID=client2
            GRPNO=5
group6          LMID=client2
            GRPNO=6
group7          LMID=client2
            GRPNO=7
group8          LMID=client2
            GRPNO=8
group9          LMID=client2
            GRPNO=9
group10         LMID=client2
            GRPNO=10

#--------------------------------------------------------------------
#--------------------------------------------------------------------

#--------------------------------------------------------------------
*SERVERS
#--------------------------------------------------------------------
#
# "--" is application-specific arguments to be passed to server
# "-n" is designed to specify server-id

service SRVGRP=group1
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n1"
    RQADDR=tpcc_1 SRVID=1

service SRVGRP=group1
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n2"
    RQADDR=tpcc_2 SRVID=2

service SRVGRP=group1
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n3"
    RQADDR=tpcc_3 SRVID=3

service SRVGRP=group1
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n4"
    RQADDR=tpcc_4 SRVID=4

service SRVGRP=group1
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n5"
    RQADDR=tpcc_5 SRVID=5

service SRVGRP=group2
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n6"
    RQADDR=tpcc_6 SRVID=6

service SRVGRP=group2
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n7"
    RQADDR=tpcc_7 SRVID=7

service SRVGRP=group2
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n8"
    RQADDR=tpcc_8 SRVID=8

service SRVGRP=group2
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n9"
    RQADDR=tpcc_9 SRVID=9

service SRVGRP=group2
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n10"
    RQADDR=tpcc_10 SRVID=10

service SRVGRP=group3
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n11"
    RQADDR=tpcc_11 SRVID=11

service SRVGRP=group3
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n12"
    RQADDR=tpcc_12 SRVID=12

service SRVGRP=group3
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n13"
    RQADDR=tpcc_13 SRVID=13

service SRVGRP=group3
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n14"
    RQADDR=tpcc_14 SRVID=14

service SRVGRP=group3
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n15"
    RQADDR=tpcc_15 SRVID=15

service SRVGRP=group4
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n16"
    RQADDR=tpcc_16 SRVID=16

service SRVGRP=group4
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n17"
    RQADDR=tpcc_17 SRVID=17

service SRVGRP=group4
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n18"
    RQADDR=tpcc_18 SRVID=18

service SRVGRP=group4
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n19"
    RQADDR=tpcc_19 SRVID=19

service SRVGRP=group4
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n20"
    RQADDR=tpcc_20 SRVID=20

service SRVGRP=group5
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n21"
    RQADDR=tpcc_21 SRVID=21

service SRVGRP=group5
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n22"
    RQADDR=tpcc_22 SRVID=22

service SRVGRP=group5
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n23"
    RQADDR=tpcc_23 SRVID=23

service SRVGRP=group5
                CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n24"
    RQADDR=tpcc_24 SRVID=24
```

```
service SRVGRP=group5
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n25"
    RQADDR=tpcc_25 SRVID=25

service SRVGRP=group6
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n26"
    RQADDR=tpcc_26 SRVID=26

service SRVGRP=group6
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n27"
    RQADDR=tpcc_27 SRVID=27

service SRVGRP=group6
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n28"
    RQADDR=tpcc_28 SRVID=28

service SRVGRP=group6
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n29"
    RQADDR=tpcc_29 SRVID=29

service SRVGRP=group6
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n30"
    RQADDR=tpcc_30 SRVID=30

service SRVGRP=group7
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n31"
    RQADDR=tpcc_31 SRVID=31

service SRVGRP=group7
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n32"
    RQADDR=tpcc_32 SRVID=32

service SRVGRP=group7
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n33"
    RQADDR=tpcc_33 SRVID=33

service SRVGRP=group7
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n34"
    RQADDR=tpcc_34 SRVID=34

service SRVGRP=group7
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n35"
    RQADDR=tpcc_35 SRVID=35

service SRVGRP=group8
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n36"
    RQADDR=tpcc_36 SRVID=36

service SRVGRP=group8
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n37"
    RQADDR=tpcc_37 SRVID=37

service SRVGRP=group8
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n38"
    RQADDR=tpcc_38 SRVID=38

service SRVGRP=group8
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n39"
    RQADDR=tpcc_39 SRVID=39

service SRVGRP=group8
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n40"
    RQADDR=tpcc_40 SRVID=40

service SRVGRP=group9
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n41"
    RQADDR=tpcc_41 SRVID=41

service SRVGRP=group9
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n42"
    RQADDR=tpcc_42 SRVID=42

service SRVGRP=group9
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n43"
    RQADDR=tpcc_43 SRVID=43

service SRVGRP=group9
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n44"
    RQADDR=tpcc_44 SRVID=44

service SRVGRP=group9
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n45"
    RQADDR=tpcc_45 SRVID=45

service SRVGRP=group10
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n46"
    RQADDR=tpcc_46 SRVID=46

service SRVGRP=group10
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n47"
    RQADDR=tpcc_47 SRVID=47

service SRVGRP=group10
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n48"
    RQADDR=tpcc_48 SRVID=48

service SRVGRP=group10
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n49"
    RQADDR=tpcc_49 SRVID=49

service SRVGRP=group10
        CLOPT="-s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC -s
DVRY_SVC -- -n50"
    RQADDR=tpcc_50 SRVID=50
#------------------------------------------------------------------------
*SERVICES
#------------------------------------------------------------------------
*ROUTING
#------------------------------------------------------------------------
```

Appendix D RTE Configuration

This appendix lists RTE input parameters and code fragments used to generate each transaction input file, to demonstrate the RTE was configured to generate transaction input data as specified in *Clause 2* of the specification.

D.1 Field Value Generation

Source/src/driver/generate.c

```
/*****************************************************************************
 @(#) Version: A.10.10 $Date: 97/12/15 13:53:51 $

 (c) Copyright 1996, Hewlett-Packard Company, all rights reserved.
*****************************************************************************/

#include <stdio.h>
#include <values.h>
#include <unistd.h>
#include <time.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <fcntl.h>
#include <signal.h>
#include <math.h>

#include "shm_lookup.h"
#include "random.h"

#include <time.h>

int CLAST_CONST_C = 208;
int CID_CONST_C =  37;
int IID_CONST_C = 75;

int trans_type = 0;   /* type of transaction 0 == all */

extern ID warehouse;
extern ID district;

extern int no_warehouse;
extern int no_item;
extern int no_dist_pw;
extern int no_cust_pd;
extern int no_ord_pd;
extern int no_new_pd;
extern int tpcc_load_seed;

neworder_gen(t)
   neworder_trans *t;
   {
   int i;

   t->W_ID = warehouse;

   t->D_ID = RandomNumber(1, no_dist_pw);
   t->C_ID = NURandomNumber( 1023, 1, no_cust_pd, CID_CONST_C);

   t->O_OL_CNT = RandomNumber(5, 15);

   for (i=0; i<t->O_OL_CNT; i++)
      {
      t->item[i].OL_I_ID = NURandomNumber(8191, 1, no_item, IID_CONST_C);
      t->item[i].OL_SUPPLY_W_ID = RandomWarehouse(warehouse, scale, 1);
      t->item[i].OL_QUANTITY = RandomNumber(1, 10);
      }

   /* 1% of transactions roll back. Give the last order line a bad item */
   if (RandomNumber(1, 100) == 1)
      t->item[t->O_OL_CNT - 1].OL_I_ID = -1;
   }

payment_gen(t)
   payment_trans *t;
   {

   /* home warehouse is fixed */
   t->W_ID = warehouse;

   /* Random district */
   t->D_ID = RandomNumber(1, no_dist_pw);

   /* Customer is from remote warehouse and district 15% of the time */
   t->C_W_ID = RandomWarehouse(warehouse, scale, 15);
   if (t->C_W_ID == t->W_ID)
      t->C_D_ID = t->D_ID;
   else
      t->C_D_ID = RandomNumber(1, no_dist_pw);

   /* by name 60% of the time */
   t->byname = RandomNumber(1, 100) <= 60;
   if (t->byname)
      LastName(NURandomNumber(255, 0, no_cust_pd/3 - 1, CLAST_CONST_C),
                                    t->C_LAST);
   else
      t->C_ID = NURandomNumber( 1023, 1, no_cust_pd, CID_CONST_C);

   /* amount is random from [1.00..5,000.00] */
   t->H_AMOUNT = RandomNumber(100, 500000);

   }

ordstat_gen(t)
   ordstat_trans *t;
   {

   /* home warehouse is fixed */
   t->W_ID = warehouse;

   /* district is randomly selected from warehouse */
   t->D_ID = RandomNumber(1, no_dist_pw);

   /* by name 60% of the time */
   t->byname = RandomNumber(1, 100) <= 60;
   if (t->byname)
      LastName(NURandomNumber(255, 0, no_cust_pd/3 - 1, CLAST_CONST_C),
                                    t->C_LAST);
   else
      t->C_ID = NURandomNumber(1023, 1, no_cust_pd, CID_CONST_C);
   }

delivery_gen(t)
   delivery_trans *t;
   {
   t->W_ID = warehouse;
   t->O_CARRIER_ID = RandomNumber(1,10);
   }

stocklev_gen(t)
   stocklev_trans *t;
   {
   t->W_ID = warehouse;
   t->D_ID = district;
   t->threshold = RandomNumber(10, 20);
   }

int get_trans_type()
/*****************************************************************************
 * get_trans_type selects a transaction according to the weighted average
 * For TPC-C rev 3.0 and less and TPC-C rev 3.2 this is:
 *       new-order :  ???
 *       payment   : 43.0%
 *       order stat:  4.0%
 *       delivery  :  4.0%
 *       stock     :  4.0%
 *****************************************************************************/
   {
   static double weight[] = { 0.0,  0.0, .4305, .0405, .0405, .0405};
   double drand48();
   int type;
   double r;

   /* choose a random number between 0.0 and 1.0 */
   if (trans_type == 0) {
#ifdef USE_DRAND48
      r = drand48();
#else
      r = randy();
#endif

      /*
       * select one of STOCKLEV, DELIVERY, ORDSTAT and PAYMENT
       * based on weight
       */
      for (type = STOCKLEV;  type > NEWORDER;  type--) {
         r -= weight[type];
         if (r < 0) break;
      }
   } else {
            /* user wants only a certain type (say all stocklevel) so do that
         instead */
            type = trans_type;
   }
```

```
/* return the value of the selected card, or NEWORDER if none selected */
return type;
}
```

Appendix E Disk Storage

The calculations used to determine the storage requirements for the 8 hours logical log and the 60-day space calculations are contained in this appendix.

The calculations for the 8 hours recovery log were based on how often the oracle redo log files were filling up and needed to be switched. The database took a checkpoint, and switched from the "current" log file to the other, "active" log file, every 28.6 minutes. Each log file is 160GB. So, to run for 8 hours, we need:

((8 * 60) / 28.6) * 160 = 2,685.31GB (must be mirrored)

On the 74 disk arrays, there is an additional 4,387GB of space available to be used for 8 hours of log

TPC-C 60-Day Space Requirements

TPM	389434.4
Warehouses	32000

SEGMENT	TYPE	TSPACE	BLOCKS	FIVE_PCT	DAILY_GROWTH	BLOCK SIZE	TOTAL in MB
CUSTCLUSTER	CLUSTER	CUST	480,768,000	24,038,400	0	2,048	985,950
DISTCLUSTER	CLUSTER	MISC	512,000	25,600	0	2,048	1,050
HIST	TABLE	HIST	7,040,000	0	1,370,809	8,192	65,709
ICUST1	INDEX	ICUST1	5,120,000	256,000	0	8,192	42,000
ICUST2	INDEX	ICUST2	10,240,000	512,000	0	8,192	84,000
IDIST	INDEX	MISC	256,000	12,800	0	2,048	525
IITEM	INDEX	MISC	256,000	12,800	0	2,048	525
INORD	INDEX	NORD	640,000	32,000	0	8,192	5,250
IORDL	INDEX	ORDL	88,192,000	0	17,172,499	8,192	823,160
IORDR1	INDEX	IORDR1	3,776,000	188,800	0	8,192	30,975
IORDR2	INDEX	IORDR2	5,248,000	262,400	0	8,192	43,050
ISTOK	INDEX	ISTOK	12,800,000	640,000	0	8,192	105,000
ITEMCLUSTER	CLUSTER	MISC	256,000	12,800	0	2,048	525
IWARE	INDEX	MISC	256,000	12,800	0	2,048	525
ORDR	TABLE	ORDR	4,864,000	0	947,104	8,192	45,399
ROLL_SEG	SYS	ROLL_0	4,096,512	0	0	2,048	8,001
STOKCLUSTER	CLUSTER	STOK	641,024,000	32,051,200	0	2,048	1,314,600
SYSTEM	SYS	SYS	410,112	0	0	2,048	801
WARECLUSTER	CLUSTER	MISC	256,000	12,800	0	2,048	525
Total							3,557,571

Dynamic space	782,000	Initial MB for (History+Orders+Order_Line)
Static space	2,623,302	Initial Blocks + 5% - Dynamic
Daily Growth	152,269	Total Dynamic [(calc. as (Initial Blocks)*tpmC/(WHS*62.5)]
Daily Spread	0	Oracle may be configured so that daily spread is 0
60-day space (MB)	11,759,433	Static + 60*(Daily Growth+Daily Spread)
60-day (GB)	11,483.82	Excludes OS, Paging and RDBMS Logs
8-hour log (GB)	2,792.73	RDBMS Logs
Server swap (GB)	256.00	OS: Paging
Server OS (GB)	16.00	OS: UNIX File System
Total Space Needed	14,548.55	GB

Priced-Sytem Configuration	**Size in MB after RAID 1 redundency**	**Quantity**	**Total (GB)**
VA7100 with 15 18.2 GB disk drives in RAID 1 mode	124,652	28	3,408.45
VA7100 with 15 36.4-GB disk drives in RAID 1 mode	249,304	46	11,199.20
Total Storage in Priced System (GB)			14,607.66

Appendix F Price Quotes

The following pages contain the price quotes for the hardware included in this FDR.

Andreas Hotea
HP
Cupertino, CA 95014

HP Unix Sales Development
19111 Pruneridge Aveune
Cupertino, CA 95014
(408) 447-2320

June 12, 2002

HP 9000 Superdome Enterprise Server

TPC-C Rev 5

Report Date: June 12, 2002

Description	Part Number	Brand	Price Key	US List Price	Qty	Price	3Year Main.Price
Server Hardware							
Super Dome left chassis	A5201A, Opt. 101		1	317,429	1	317,429	147,264
Super Dome right chassis	A5202A, Opt. 101		1	235,655	1	235,655	
Memory module - 2 GB	A5198A, Opt. 0D1		1	14,000	128	1,792,000	
I/O enclosures	A4856A, Opt. 0D1		1	14,805	4	59,220	
PDCA Redundant Power Source	A5800A, Opt. 0D1		1	578	2	1,156	
Cell Board with 4 PA-8700 750MHz Processors	A6445A, Opt. 0D1		1	10,080	16	161,280	
ICOD right to use processor	A6441A Opt. 104		1	23,806	64	1,523,584	351,744
Super Dome PCI Core I/O card	A5210A, Opt. 0D1		1	1,045	1	1,045	
PCI 1000BT Lan Adapter	A4926A, Opt. 0D1		1	2,135	1	2,135	
PCI Fibre Channel 2X	A5158A, Opt 0D1		1	2,240	25	56,000	
Rack Installation Kit	A5170A, Opt. 0D1		1	410	1	410	
DVD-ROM	C4318SZ		1	3,738	1	3,738	
(includes SCSI-2 card, cables, enclosures)							
HP9000 A500 Support Management Station	A5570B		1	11,780	1	11,780	
(includes memory,CPU,lan card,disk,OS, etc)							
.5m 68pin SCSI Cable	Opt. 001		1	99	1	99	
WSE 68pin SCSI Teminator	Opt. 835		1	46	1	46	
HP-UX 11.i Sys Media, CD-ROM	B3920EA, Opt. AAF		1	520	1	520	
PowerTrust 12kVA UPS 230VUPS	A6585A		1	10,999	6	65,994	
PowerTrust 3.0VA UPS 230VUPS	A1356A		1	3,199	2	6,398	
PowerTrust 2.0kVA UPS 230VUPS	A1353A		1	2,199	1	2,199	
(Support Option for UPS)			1	0	1	0	
Surestore VA 7100 w/dual controllers 512MB cache	A6262A		1	44,250	74	3,274,500	271,876
18GB 15K RPM FC HDD	A6191A, Opt. 0D1		1	914	420	383,880	
36GB 15K RPM FC HDD.	A6193A, Opt 0D1		1	1,349	690	930,810	
2 meter Fibre Optic Cable	A3583A		1	175	74	12,950	
16 meter Fibre Optic Cable	A3531A		1	200	25	5,000	
10 Port Short Wave Fibre Channel Hub	A3724A		1	9,690	25	242,250	
HP9000 Std. Rack System E41	A4902A		1	1,910	7	13,370	
Modular Power Dist.	A5137AZ		1	145	26	3,770	
200-240 Volts	A5137AZ, Opt AW4		1	94	26	2,444	
				Subtotal		**9,109,661**	**770,884**
Client Hardware							
hp workstation c3700 with 2GB Memory	A6057B		1	12,545	28	351,260	108,668
1 GB Memory Moudle	A6016A, Opt. OD1		1	1,395	168	234,360	
700/96 Console	C1064GX		1	550	1	550	
18 GB LVD 10K RPM Disk	A4998A, Opt. OD1		1	595	28	16,660	
				Subtotal		**602,830**	**108,668**
Client Software							
HP C/ANSI C Compiler	B3901BA, Option AH0		1	1,600	1	1,600	170
				Subtotal		**1,600**	**170**
User Connectivity							
HP ProCurve Switch 4000M	J4121A		1	1,999	1	1,999	588
HP ProCurve Switch Gigabit-SX Module	J4113A		1	1,189	1	1,189	
				Subtotal		**3,188**	**588**
HP's Large configuration Discount and Support Prepayment*						(5,037,840)	(303,421)
*All discounts are based on US list prices and for similar quantities and configurations				**Total**		**9,717,279**	**880,310**

All the components in the price list are currently available. Maintenance support price is for 24 hours, 7 days with 4 hour response time.

The eCommerce Transaction Platform

November 16, 2001

Ms. Lucille Boushey
TPC-C Performance Project Manager
Hewlett Packard
408 447 7364
408 447 5958 FAX

Dear Ms. Boushey:

Per your request I am enclosing the pricing information regarding TUXEDO 6.4 that you requested. This pricing applies to Tuxedo 6.4, 6.5, 7.1, and 8.0. Please note that Tuxedo 8.0 is our most recent version of Tuxedo. Core functionality services pricing is appropriate for your activities. As per the table below HP PA-RISC systems are classified as either a Tier 1, 2, 3, 4 or 5 systems depending on the performance and CPU capacity of the system. This quote is valid for 60 days from the date of this letter.

10.1.1 Tuxedo Core Functionality Services (CFS) Program Product Pricing and Description

TUX-CFS provides a basic level of middleware support for distributed computing, and is best used by organizations with substantial resources and knowledge for advanced distributed computing implementations.

TUX-CFS prices are server only and are based on the overall performance characteristics of the server and uses the same five tier computer classification as TUXEDO 6.4, 6.5, 7.1, and 8.0. Prices range from $3,000 for Tier 1 to $250,000 for Tier 5. Under this pricing option EVERY system running TUX-CFS at the user site must have a TUXEDO license installed and pay the appropriate per server license fees. Note that a 5% discount will apply to total list price license purchases less than $100,000 (for instance 30 Tier 1 servers – 30* 3,000 = $90,000 - would be eligible for a 5% discount). Support is not discountable.

Very Truly Yours,

Robert J. Gieringer

Rob Gieringer,
Worldwide Pricing Manager

10.1.1.1 BEA Tux/CFS Unlimited User License Fees Per Server

Unlimited User License fees per server	Number of Users	Dollar Amount	Maintenance (5 x 8) per year	Maintenance (7 x 24) per year
Tier 1 -- PC Servers with 1 or 2 CPUs, entry level RISC Uni-processor workstations and servers	Unlimited	$3,000.00	$480.00	$690.00
Tier 2 - PC Servers with 3 or 4 CPUs, Midrange RISC Uni-processor servers and workstations with up to 2 CPUs	Unlimited	$12,000.00	$1,920.00	$2,760.00
Tier 3 - Midrange Multiprocessors, up to 8 CPUs per system capacity	Unlimited	$30,000.00	$4,800.00	$6,900.00
Tier 4 - Large (more than 8, less than 32 CPUs)	Unlimited	$100,000.00	$16,000.00	$23,000.00
Tier 5 - Massively Parallel Systems, > 32 processors	Unlimited	$250,000.00	$40,000.00	$57,500.00

	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)	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, T520,T600 1-16 CPUs S-Class	9000/V series all models X-Class 9000 Series - Superdome