UNISYS

TPC Benchmark™ C Full Disclosure Report

Unisys Corporation
and
Oracle Corporation

Unisys U6000/500 Model 80

Using UnixWare 2.1
and
Oracle7, Release 7.3

Submitted for review
July 1996

Unisys Part Number 4487 7637-000

First Edition - July 1996

Unisys Corporation believe that the information in this document is accurate as of the publication date. The information in this document is subject to change without notice. Unisys Corporation assume no responsibility for any errors that may appear in this document.

The pricing information in this document is believed to accurately reflect the current prices as of the publication date. However, Unisys Corporation and Oracle Corporation provide no warranty on the pricing information in this document.

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

All performance data contained in this report was obtained in a rigorously controlled environment, and therefore results obtained in other operating environments may vary significantly. Unisys Corporation and Oracle Corporation do not warrant or represent that a user can or will achieve similar performance expressed in transactions per minute (tpmC) or normalized price/performance ($/tpmC). No warranty of system performance or price/performance is expressed or implied in this report.

Printed in USA, July 1996.

Unisys Corporation Part Number: 4487 7637-000

Page Status

Page	Issue
i through xiii	-000
xiv	Blank
0-1 through 0-3	-000
0-4	Blank
1-1 through 1-1	-000
1-2	Blank
2-1 through 2-2	-000
3-1 through 3-3	-000
3-4	Blank
4-1 through 4-4	-000
5-1 through 5-9	-000
5-10	Blank
6-1 through 6-2	-000
7-1 through 7-3	-000
7-4	Blank
8-1 through 8-1	-000
8-2	Blank
9-1 through 9-3	-000
9-4	Blank
A-1 through A-64	-000
B-1 through B-56	-000
C-1 through C-3	-000
C-4	Blank
D-1 through D-1	-000
D-2	Blank
E-1 through E-1	-000
E-2	Blank
F-1 through F-3	-000
F-4	Blank

Unisys uses an 11-digit document numbering system. The suffix of the document number (1234 5678-xyz) indicates the document level. The first digit of the suffix (x) designates a revision level; the second digit (y) designates an update level. For example, the first release of a document has a suffix of -000. A suffix of -130 designates the third update to revision 1. The third digit (z) is used to indicate an errata for a particular level and is not reflected in the page status summary.

Abstract

Overview

This report documents the methodology and results of the TPC Benchmark C (TPC-C) conducted on the Unisys Corporation U6000/500 Model 80 using Oracle7, release 7.3 and Tuxedo transaction monitor. The operating system used for the benchmark was UnixWare 2.1. The DBMS used was Oracle7, release 7.3.

TPC Benchmark Metrics

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

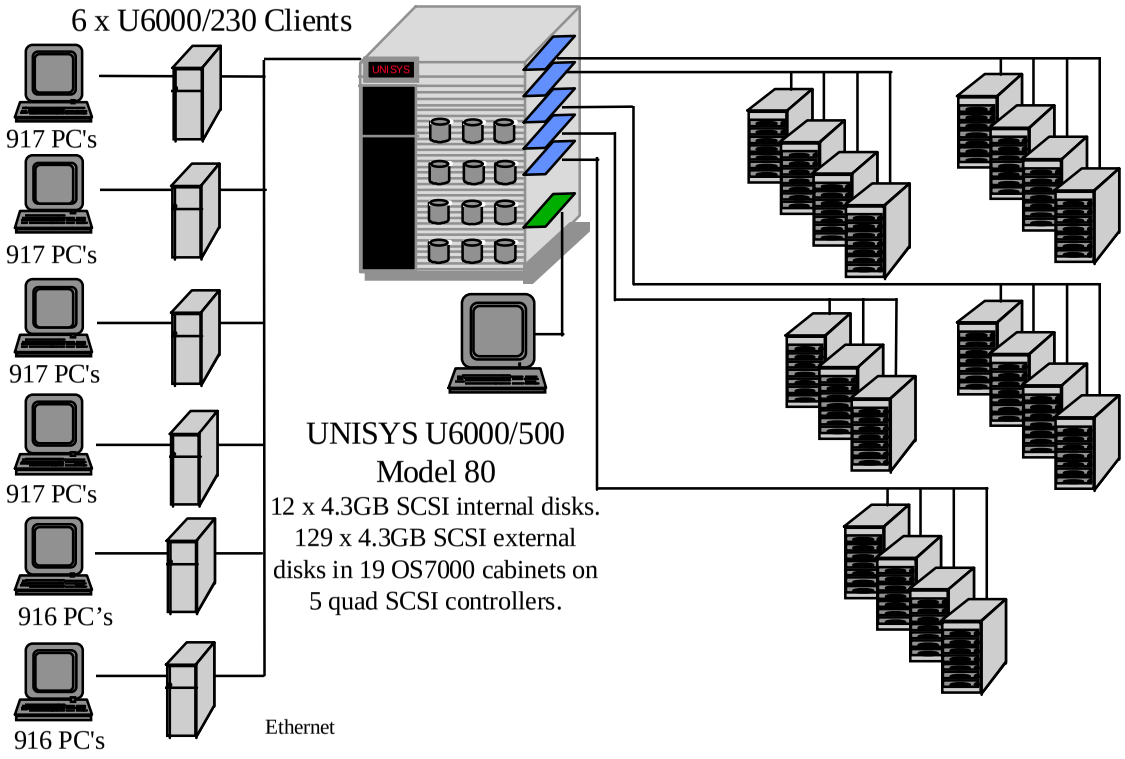
Executive Summary

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

Auditor

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

Unisys Corporation Oracle Corporation	Unisys U6000/500 Model 80 Client/Server			TPC-C Rev. 3.2 Report Date: July 17, 1996
Total System Cost	TPC-C Throughput	Price/Performance		Availability Date
$1,899,743	**6253.32 tpmC**	**$303.80 per tpmC**		**July 17, 1996**
Processors	Database Manager	Operating System	Other Software	Number of Users
10 Pentium 150 MHz	Oracle 7.3.2	UnixWare 2.1	BEA Tuxedo 6.1	**5500**



System Components	Server		Clients	
	Quantity	Type	Quantity	Type
Processors	10	150 MHz Pentium with 2MB Level 2 Cache	6	90MHz Pentium with 512KB Level 2 Cache
Memory	1	2048MB	6	3X64MB SIMM
Disk Controllers Ethernet Controller	5 2	4 channel STIO Controllers Ethernet Controllers	6 6	Ethernet controller Systech Etherplex NIC
Disk Drives	12 129	4.3 GB internal SCSI disk 4.3 GB Disk module on OSS7000	6	1.0 GB internal SCSI disk
Total Storage		606.3 GB		6 GB
CD-ROM / Tape	1	DAT Drive	6	QIC Drive

Unisys Corporation Oracle Corporation	U6000/500 Model 80 Client Server								TPC-C Rev 3.2 7/17/96
Description	Style	Third Party Brand	Pricing	Unit Price	Qty.	Extended Price	Monthly Maint.	Months	5 Years Maint.
Server Hardware									
SYS: U6000/500 Model 80, 2x150MHz, 0MB Mem	UN6580-ZC2			$73,800	1	$73,800	$566	48	$27,168
PROC:2x150MHz Pentium/2MB Cache	UN6580-PC2			$22,000	4	$88,000	$207	48	$39,744
SYS: Model 80 STIO Board	UN6580-SBB			$3,200	1	$3,200	$22	48	$1,056
CTRL:STIO Quad SCSI Board	UN6500-SSB			$6,000	5	$30,000	$31	48	$7,440
CTRL:Enhanced VGA with 2MB	PCV32002-EVG			$390	1	$390	INCL		
INSTL:Disk Install Kit	UN6030-DIK			$75	1	$75	INCL		
TAPE:DAT DDS-2 4 GB	UN6000-DT4			$1,629	1	$1,629	$13	48	$624
DISK:4GB SCSI 3.5 Internal	UN6003-4G3			$3,225	12	$38,700	$25	48	$14,400
MEM:256 MB ECC Memory Board	UN6256-MBD			$24,500	2	$49,000	INCL		
MEM:256 MB Memory Upgrade	UN6256-MUP			$21,500	6	$129,000	INCL		
DISK: Free-standing Enclosure	OSS7000-EXP			$1,850	19	$35,150	$7	48	$6,384
DISK: 4GB Drive	OSD4100-S40			$3,050	129	$393,450	$18	48	$111,456
ETHERNET: 10Mbit	UNE6000-ETH			$323	2	$646	$3	48	$288
DISPLAY:SVG250-COLOR	SVG250-COL			$311	1	$311	INCL		
KEYBOARD:PC 101	PCK101-KBD			$40	1	$40	INCL		
					Subtotal	**$843,391**			**$208,560**
Server Software									
O/S:UnixWare 2.1 Application Server 5- user	UW6021-ASE			$1,295	1	$1,295	$21	60	$1,260
Online Disk Manager for UnixWare 2.1	UW6021-ODM			$2,195	1	$2,195	$35	60	$2,100
UnixWare 2.1 Unlimited User upgrade	UW6000-L21			$12,995	1	$12,995	$206	60	$12,360
UnixWare 2.1 Processor upgrade	UW6021-PRU			$995	8	$7,960	$16	60	$7,680
Oracle 7.3.2 (unlimited) and SQL*Net				$160,000	1	$160,000	$2,215	60	$132,900
					Subtotal	**$184,445**			**$156,300**
Client Hardware									
SYS:U6000/230,1x90MHz Pentium, 0MB Mem	UN6230-MOD			$3,800	6	$22,800	$31	48	$8,928
MEM:ECC Card	UN6310-ECM			$300	6	$1,800	INCL		
MEM:64MB SIMM	UN6310-64M			$2,328	18	$41,904	INCL		
DISK:1GB HH SCSI 3.5	UNA6003-1G3			$750	6	$4,500	$11	48	$3,168
TAPE:QIC 1.0 GB	UN6000-Q10			$810	6	$4,860	$11	48	$3,168
ETHERNET: 10Mbit	PCC500-ETH			$259	6	$1,554	INCL		
DISPLAY:SVG250-COLOR	SVG250-COL			$311	6	$1,866	INCL		
KEYBD:PC 101	PCK101-KBD			$40	6	$240	$4	48	$1,152
					Subtotal	**$79,524**			**$16,416**
Client Software									
O/S:UnixWare 2.1 Application Server 5 user, 2 cpu	UW6021-ASE			$1,295	6	$7,770	$21	60	$7,560
UnixWare 2.1 Unlimited user upgrade	UW6000-L21			$12,995	6	$77,970	$206	60	$74,160
BEA Tuxedo Transaction Monitor, version 6.1		BEA	1	$20,000	6	$120,000	$250	60	$90,000
					Subtotal	**$205,740**			**$171,720**
User Connectivity									
EtherPlex NIC, 2048 users	UCC-2048	SysTech	2	$3,496	6	$20,976	$27	48	$7,740
Slimline 24 port hub +10% spares	DEH1477	Ultra	3	$275	258	$70,950	INCL		
					Subtotal	**$91,926**			**$7,740**
					Total	**$1,405,026**			**$560,736**
HW/SW Volume Discount				0%					
Maintenance Quantity Discount on Unisys Products				15%					**($49,514)**
Maintenance Prepayment Discount on Unisys Products				5%					**($16,505)**
					Final Total	**$1,405,026**			**$494,717**

Notes:
1. BEA Systems
2. SysTech Corp.
3. Data Comm Warehouse
Audited by Richard Gimarc of Performance Metrics, Inc.

Five Year Cost of Ownership	**$1,899,743**
TPC-C Throughput	**6253.32**
$/tpmC	**$303.80**

NUMERICAL QUANTITIES SUMMARY

UNISYS Corporation U6000/500 Model 80
Oracle7 Release 7.3

MQTh, Computed Maximum Qualified Throughput: **6253.32**

% throughput difference, reported & reproducibility runs: 0.86%

Transaction Mix (% of Total Transactions)

Transaction Mix	
New-Order	44.56%
Payment	43.22%
Order-status	4.03%
Delivery	4.12%
Stock-level	4.05%

Response Time (seconds)

Transaction	**Average**	**Maximum**	**90th %ile**
New-Order	2.10	25.49	4.56
Payment	1.27	16.07	2.64
Order-Status	1.66	24.48	3.26
Delivery	0.46	4.45	0.57
Delivery (Deferred)	n/a	n/a	10.93
Stock-Level	5.42	43.73	11.03
Menu	0.45	65.49	0.59

Keying/Think Time Times (seconds)

Transaction	**Minimum**	**Average**	**Maximum**
New-Order	18.00/0.00	18.03/12.38	18.26/118.64
Payment	3.00/0.00	3.01/12.31	3.20/120.35
Order-Status	2.00/0.00	2.01/10.26	2.12/97.76
Delivery	2.00/0.00	2.01/5.12	2.12/42.81
Stock-Level	2.00/0.00	2.01/5.13	2.13/42.58

Test Duration:

Ramp up time	45.5 minutes
Measurement interval (M)	25.0 minutes
Transactions completed during measurement interval	350,771
Ramp-down time	7.5 minutes
Checkpointing:	
Number of checkpoints	1
Checkpoint interval	25 minutes

Table of Contents

Figures

Tables

Preface

Document Structure

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

This report is submitted to satisfy the specification's requirement for full disclosure. It documents the compliance of the benchmark implementation and execution reported for the Unisys Corporation U6000/500 Model 80 using Oracle7.

TPC Benchmark C Overview

The TPC Benchmark™ C Standard Specification Revision 3.2 was developed by the Transaction Processing Council (TPC). It is the intent of the TPC to develop a suite of benchmark to measure the performance of computer systems executing a wide range of applications. Unisys and Oracle Corporations are active participants in the TPC to define and develop such a suite of benchmarks.

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

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

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

Despite the fact that this benchmark offers a rich environment that emulates many OLTP environments, this benchmark does not reflect the entire range of OLTP requirements. In addition, the extent to which a customer can

achieve the results reported by a vendor is highly dependent on how closely TPC-C approximates the customer application. The relative performance of systems derived from this benchmark does not necessarily hold for other workloads or environments. Extrapolations to any other environment are not recommended.

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-C should not be used as a substitute for a specific customer application benchmarking when critical capacity planning and/or product evaluation decisions are contemplated.

0. General Items

0.1. Order and Titles

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

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

0.2. Executive Summary Statement

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

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

0.3. Numerical Quantities Summary

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

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

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

0.4. Application Code Disclosure

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

Appendix A contains the application code used in this TPC-C benchmark.

0.5. Benchmark Sponsor

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

This TPC benchmark C was co-sponsored by Unisys Corporation and Oracle Corporation. The benchmark test was developed by Oracle Corporation, SCO and Unisys. The benchmark was conducted at Unisys, San Jose with support from Oracle Corporation and SCO.

0.6. Parameter Settings

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

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

Appendix C contains the configuration and system parameters used in running these TPC-C tests. It also contains all the client and server OS, Oracle7 and Tuxedo tunable parameters.

0.7. Configuration Diagrams

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

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

The Remote Terminal Emulator (RTE) software used for these TPC-C tests is a product developed by Unisys. The benchmark configuration is illustrated in Figure 0.1. The emulated users on the driver systems were directly connected to the client systems under test via local area network (LAN) and communicate using the TCP/IP protocol. The server contained a total of 135 drives, 57 of 2.1GB capacity, and 78 of 4.3GB capacity. Of these, 13 were used for backup storage. 122 were in use during the run; 1 containing the OS and Oracle7 code, 4 as two mirrored pairs for the audit log, and 117 for the database.

The priced configuration for the Unisys U6000/500 Model 50 server is shown in Figure 0.2. The priced configuration had a total of 141 drives; 120 for the database, 20 for the mirrored audit log and one for the OS and Oracle7 code. All the 2.1GB disks in the benchmarked configuration were replaced with 4.3GB disks to satisfy the 180 day storage and 8-hour audit log requirements.

Each emulated terminal ran a client application which requested TUXEDO services for each transaction. TUXEDO completed the transaction over an Ethernet LAN to the server using Oracle7 SQL*NET v2.1.The server ran Oracle7, release 7.3.

Figure 0.1: Benchmarked Configuration

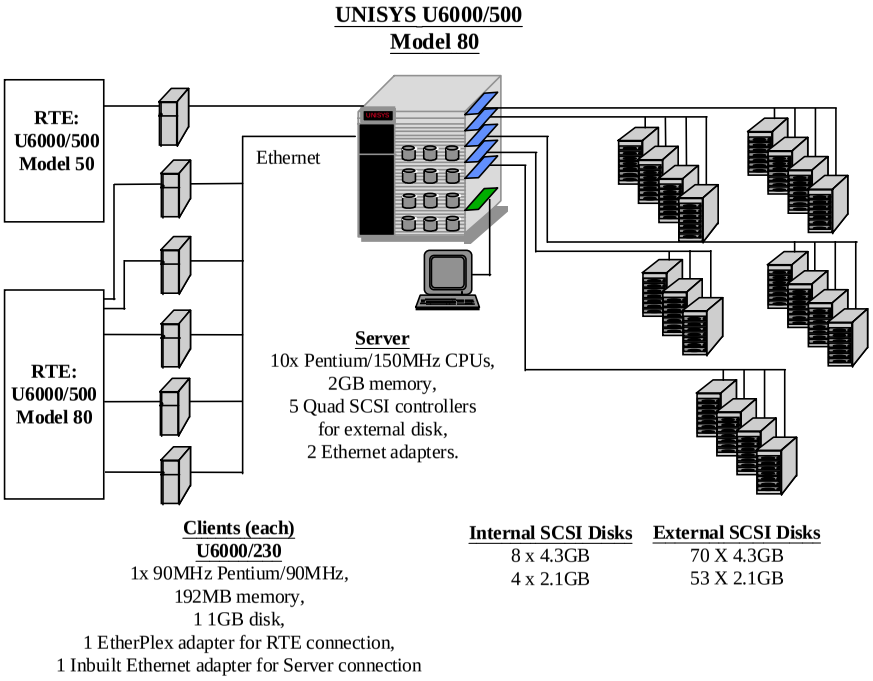
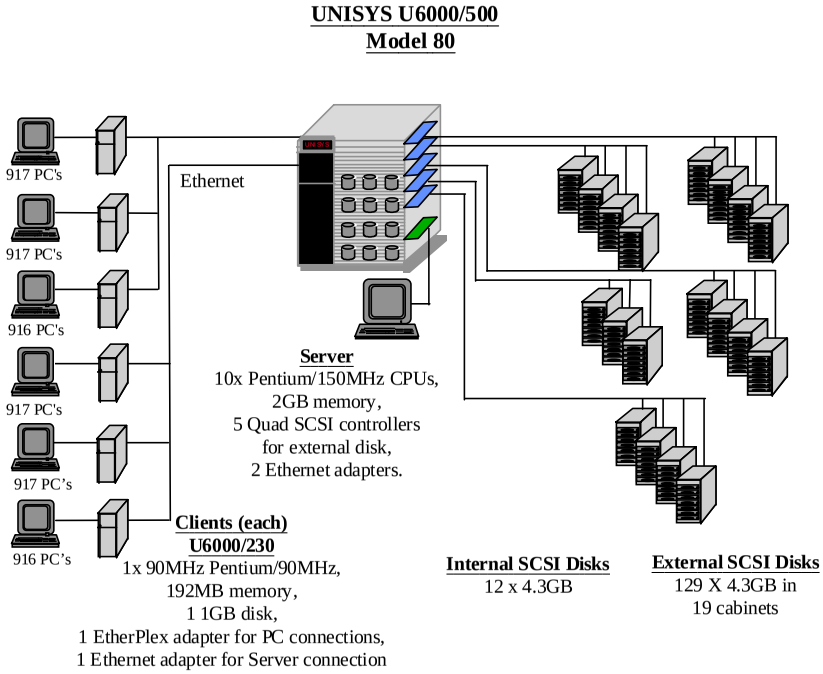


Figure 0.2: Priced Configuration



1. Clause 1: Logical Database Design

1.1. Table Definitions

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

Appendix B contains the definition and description of all the required database tables (structures) and all the programs used to create (load) the database and establish the required initial populations.

1.2. Physical Organization of the Database

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

The disk space was allocated to Oracle7 according to the data in section 4.2 .The database table spaces was laid out with considerations to load distribution. Online Data Manager (ODM) for UnixWare 2.1 was used to stripe data files and mirror redo logs.

1.3. Insert and/or Delete Operations

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

There were no restrictions on insert and/or delete operations to any of the tables. The space required for any additional need was allocated accordingly.

1.4. Partitioning

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

Replication of tables, if used, must be disclosed.

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

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

2. Clause 2: Transaction & Terminal Profiles

2.1. Random Number Generation

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

The RTE software used library routines SRAND48 and DRAND48 to generate pseudo-random numbers using the well known linear congruential algorithm and 48-bit integer arithmetic.

The seeds for each user were captured and verified by the auditor. In addition, the contents of the database were systematically searched, and randomly sampled by the auditor for patterns that would indicated the random number generator had effected and type of discernible pattern, none was found.

2.2. 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 Benchmark C Standard Specification.

2.3. Priced Terminal Feature Verification

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

This was verified by the auditor by a direct experiment during the onsite audit portion of this benchmark.

2.4. Presentation Managers or Intelligent Terminal

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

Application code running on the client implemented the TPC-C user interface. A listing of this code is included in Appendix A. Capabilities of the priced configuration's user display/keyboard device, beyond basic ASCII entry and display, were restricted to cursor positioning. A presentation manager was not used.

2.5. Transaction Statistics

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

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

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

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

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

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

Table 2.1 contains all these statistics.

Table 2.1: Transaction Statistics

Transaction Type	Statistics	Value
New Order	Rolledback transactions	1.03 %
	Home warehouse	99.00%
	Remote warehouse	1.00%
	Average Items per Order	10.00
Payment	Home warehouse	84.88%
	Remote warehouse	15.12%
	Non-primary key access	59.83%
Order Status	Non-primary key access	59.18%
Delivery	Skipped transactions (Interactive)	0
	Skipped transaction counts (Deferred)	0
	Skipped District counts (Deferred)	0
Transaction Mix	New Order Payment Order Status Delivery Stock Level	44.56% 43.22% 4.03% 4.12% 4.05%

2.6. Queuing Mechanism of Delivery

The queuing mechanism used to defer 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.

3. Clause 3: Transaction & System Properties

3.1. Transaction System Properties (ACID)

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

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

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

3.2. Atomicity

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

3.2.1. Completed Transaction

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

The balances from a randomly selected warehouse, district, and customer row was retrieved by customer number. A Payment transaction was submitted. After completion of the Payment transaction, the balances of the selected warehouse, district, and customer were retrieved to verify that the changes had been made.

3.2.2. Aborted Transactions

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

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

3.3. Consistency

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

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

1. The sum of the district balances in a warehouse is equal to the warehouse balance;
2. For each district, the next order id minus one is equal to maximum order id in the ORDER table and equal to the maximum new order id in the NEW ORDER table;
3. For each district, the maximum order id minus minimum order id in the ORDER table plus one equals the number of rows in the NEW-ORDER table for that district;
4. For each district, the sum of the order line counts equals the number of rows in the ORDER-LINE table for that district;

In order to demonstrate the consistency the following steps were taken:

1. Prior to the start of a benchmark run, the consistency of the database was verified by testing successfully conditions 1-4 described above.
2. A 600 user run representing at least 10% of the full user load was executed with a checkpoint during the run.
3. After completion of that test, the consistency of the database was verified by successfully testing the same consistency condition in step 1.

3.4. Isolation

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

The benchmark specification defines seven required tests to be performed to demonstrate that required levels of transaction isolation are met. These tests, described in Clauses 3.4.2.1 - 3.4.2.7, were all performed and verified as required. In Isolation Test 7, Case D was observed.

3.5. Durability

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

Three durability tests were executed to satisfy the requirements of the specification. The test for loss of memory and instantaneous interruption was combined and performed with a fully scaled database with 5,500 emulated users. The loss of log and loss of data tests were performed on a ten warehouse database with 100 emulated users. To the best of our knowledge, these test prove that the fully scaled configuration used for the throughput test would also pass all durability tests.

3.5.1. Loss of Log and Loss of Data Disk

The following steps were taken to demonstrate durability in the case of loss of a data disk:

1. All database volumes on a disk were backed up to a file system.
2. 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 orders.
3. The benchmark was executed with 100 users, On the driver system, the committed and rolled back New-Order transaction were recorded in a “success” file.

4. One of the log disk was powered off and removed from the disk cabinet after five minutes of steady state.
5. The system and volume manager indicated the failure of the log disk and the benchmark run continued as the log disk was mirrored.
6. After another 5 minutes of run, a data disk was powered of and removed from the disk cabinet.
7. Oracle7 recorded errors in its alert file and the database was shutdown (aborted).
8. The test was aborted on the driver and the data disk was powered on.
9. The above data disk was overwritten with meaningless data.
10. The data on that disk was restored from the backup taken before the test.
11. The database was started and the database was recovered using alter database recover database command followed by alter database recover datafile command for all datafiles which were off-line.
12. The content of the “success” file on the driver and the ORDER table were compared to verify that the record in the “success” file for committed New-Order transactions had corresponding records in the ORDER table and no entries existed for rolled back transactions.
13. Step 2 was repeated to determine the total number of orders. This number was subtracted from the count obtained previously in Step 2 and the difference was equal to the number of committed New-Order records in the “success” file.
14. Consistency condition 3 of Clause 3.3.2.3 was verified.

3.5.2. Instantaneous Interruption and Loss of Memory

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

1. The D_NEXT_O_ID fields for all rows in the district table were summed up to determine the initial count of the total number orders (count 1).
2. The benchmark was executed with Full User Count users, On the driver system, the committed and rolled back New-Order transaction were recorded in a “success” file.
3. The system’s primary power was turned off after 8 minutes of full load.
4. The test was aborted on the driver system.
5. Power was restored to the system, the system rebooted, and a database recovery performed. The database recovery restores the database to its state just after the last committed transaction before the failure.
6. Samples of the contents of the “success” file on the driver and the ORDER table were compared to verify that records in the “success” file for committed New-Order transactions had corresponding records in the ORDER table and no entries existed for rolled back transactions.
7. Step 1 was repeated to determine the total number of order (count2). Count2 minus count was not less than the number of committed New-Order records in the “success” file.
8. Consistency Condition of Clause 3.3.2.3 (no gaps in NO_O_ID) was verified.

4. Clause 4: Scaling & Database Population

4.1. Initial Cardinality of Tables

The Cardinality (e.g., the number of rows) of each table, as it existed at the start of the benchmark run, must be disclosed. If the database was over-scaled and inactive rows of the WAREHOUSE table were delete (see Clause 4.2.2) 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 600 warehouses. The benchmark run was conducted with 550 warehouses with the remaining warehouses deleted from the database. The cardinality of each table in the database is listed in Table 4.1

Table 4.1: Initial Cardinality of Database Table

Table	Occurrences
Warehouse	550
District	6,000
Customer	18,000,000
History	18,000,000
Order	18,000,000
New Order	5,400,000
Order Line	179,992,571
Stock	60,000,000
Item	100,000

4.2. Database Layout

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

Table 4.2 lists the distribution of the database over 117 disks and the audit log over 2 mirrored pairs of disks on the benchmark configuration. In addition, there was one disk containing the OS and Oracle7 code. The priced system configured and priced the 2.1GB as 4.3GB disks to satisfy the 180-day space requirement. The growth space on the priced system was accommodated by the free space remaining on the disks containing database tables plus the space on the re-configured disks plus 3 additional disks, making a total of 120 4.3GB database disks.

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

A statement must be provided that describes:

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

2. *The database interface (e.g., embedded, call level) and access language (e.g., SQL, 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.*

Oracle7, release 7.3 is a relational DBMS. PL/SQL stored procedures were used to access the database through the Oracle Call Interface.

4.4. DBMS Partitions/Replications

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

No table partitioning or replication was done.

4.5. DBMS Space Requirements

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

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

Table 4.2: Benchmark Configuration Disk Storage Detail

SCSI adapter	Device	Structure	Size	SCSI adapter	Device	Structure	Size	SCSI adapter	Device	Structure	Size
0	5	iord1	460.00 MB	2	3	ordl8	167.94 MB	3	0	ordl12	167.94 MB
0	5	nord	610.00 MB	2	3	ordl9	167.94 MB	3	0	ordl13	167.94 MB
0	6	iord1	460.00 MB	2	3	ordl10	167.94 MB	3	0	ordl14	167.94 MB
0	6	inord1	620.00 MB	2	3	ordl11	167.94 MB	3	1	ordl1	167.94 MB
1	0	iord1	460.00 MB	2	3	ordl12	167.94 MB	3	1	ordl2	167.94 MB
1	1	iord2	406.00 MB	2	3	ordl13	167.94 MB	3	1	ordl3	167.94 MB
1	2	iord2	406.00 MB	2	3	ordl14	167.94 MB	3	1	ordl4	167.94 MB
1	3	iord2	406.00 MB	2	4	ordl1	167.94 MB	3	1	ordl5	167.94 MB
1	4	iord2	406.00 MB	2	4	ordl2	167.94 MB	3	1	ordl6	167.94 MB
1	5	iord2	406.00 MB	2	4	ordl3	167.94 MB	3	1	ordl7	167.94 MB
1	6	hist1	2015.00 MB	2	4	ordl4	167.94 MB	3	1	ordl8	167.94 MB
2	0	ordl1	167.94 MB	2	4	ordl5	167.94 MB	3	1	ordl9	167.94 MB
2	0	ordl2	167.94 MB	2	4	ordl6	167.94 MB	3	1	ordl10	167.94 MB
2	0	ordl3	167.94 MB	2	4	ordl7	167.94 MB	3	1	ordl11	167.94 MB
2	0	ordl4	167.94 MB	2	4	ordl8	167.94 MB	3	1	ordl12	167.94 MB
2	0	ordl5	167.94 MB	2	4	ordl9	167.94 MB	3	1	ordl13	167.94 MB
2	0	ordl6	167.94 MB	2	4	ordl10	167.94 MB	3	1	ordl14	167.94 MB
2	0	ordl7	167.94 MB	2	4	ordl11	167.94 MB	3	2	ordl1	167.94 MB
2	0	ordl8	167.94 MB	2	4	ordl12	167.94 MB	3	2	ordl2	167.94 MB
2	0	ordl9	167.94 MB	2	4	ordl13	167.94 MB	3	2	ordl3	167.94 MB
2	0	ordl10	167.94 MB	2	4	ordl14	167.94 MB	3	2	ordl4	167.94 MB
2	0	ordl11	167.94 MB	2	5	ordl1	167.94 MB	3	2	ordl5	167.94 MB
2	0	ordl12	167.94 MB	2	5	ordl2	167.94 MB	3	2	ordl6	167.94 MB
2	0	ordl13	167.94 MB	2	5	ordl3	167.94 MB	3	2	ordl7	167.94 MB
2	0	ordl14	167.94 MB	2	5	ordl4	167.94 MB	3	2	ordl8	167.94 MB
2	1	ordl1	167.94 MB	2	5	ordl5	167.94 MB	3	2	ordl9	167.94 MB
2	1	ordl2	167.94 MB	2	5	ordl6	167.94 MB	3	2	ordl10	167.94 MB
2	1	ordl3	167.94 MB	2	5	ordl7	167.94 MB	3	2	ordl11	167.94 MB
2	1	ordl4	167.94 MB	2	5	ordl8	167.94 MB	3	2	ordl12	167.94 MB
2	1	ordl5	167.94 MB	2	5	ordl9	167.94 MB	3	2	ordl13	167.94 MB
2	1	ordl6	167.94 MB	2	5	ordl10	167.94 MB	3	2	ordl14	167.94 MB
2	1	ordl7	167.94 MB	2	5	ordl11	167.94 MB	3	3	ordl1	167.94 MB
2	1	ordl8	167.94 MB	2	5	ordl12	167.94 MB	3	3	ordl2	167.94 MB
2	1	ordl9	167.94 MB	2	5	ordl13	167.94 MB	3	3	ordl3	167.94 MB
2	1	ordl10	167.94 MB	2	5	ordl14	167.94 MB	3	3	ordl4	167.94 MB
2	1	ordl11	167.94 MB	2	6	ordl1	167.94 MB	3	3	ordl5	167.94 MB
2	1	ordl12	167.94 MB	2	6	ordl2	167.94 MB	3	3	ordl6	167.94 MB
2	1	ordl13	167.94 MB	2	6	ordl3	167.94 MB	3	3	ordl7	167.94 MB
2	1	ordl14	167.94 MB	2	6	ordl4	167.94 MB	3	3	ordl8	167.94 MB
2	2	ordl1	167.94 MB	2	6	ordl5	167.94 MB	3	3	ordl9	167.94 MB
2	2	ordl2	167.94 MB	2	6	ordl6	167.94 MB	3	3	ordl10	167.94 MB
2	2	ordl3	167.94 MB	2	6	ordl7	167.94 MB	3	3	ordl11	167.94 MB
2	2	ordl4	167.94 MB	2	6	ordl8	167.94 MB	3	3	ordl12	167.94 MB
2	2	ordl5	167.94 MB	2	6	ordl9	167.94 MB	3	3	ordl13	167.94 MB
2	2	ordl6	167.94 MB	2	6	ordl10	167.94 MB	3	3	ordl14	167.94 MB
2	2	ordl7	167.94 MB	2	6	ordl11	167.94 MB	3	4	ordl1	167.94 MB
2	2	ordl8	167.94 MB	2	6	ordl12	167.94 MB	3	4	ordl2	167.94 MB
2	2	ordl9	167.94 MB	2	6	ordl13	167.94 MB	3	4	ordl3	167.94 MB
2	2	ordl10	167.94 MB	2	6	ordl14	167.94 MB	3	4	ordl4	167.94 MB
2	2	ordl11	167.94 MB	3	0	ordl1	167.94 MB	3	4	ordl5	167.94 MB
2	2	ordl12	167.94 MB	3	0	ordl2	167.94 MB	3	4	ordl6	167.94 MB
2	2	ordl13	167.94 MB	3	0	ordl3	167.94 MB	3	4	ordl7	167.94 MB
2	2	ordl14	167.94 MB	3	0	ordl4	167.94 MB	3	4	ordl8	167.94 MB
2	3	ordl1	167.94 MB	3	0	ordl5	167.94 MB	3	4	ordl9	167.94 MB
2	3	ordl2	167.94 MB	3	0	ordl6	167.94 MB	3	4	ordl10	167.94 MB
2	3	ordl3	167.94 MB	3	0	ordl7	167.94 MB	3	4	ordl11	167.94 MB
2	3	ordl4	167.94 MB	3	0	ordl8	167.94 MB	3	4	ordl12	167.94 MB
2	3	ordl5	167.94 MB	3	0	ordl9	167.94 MB	3	4	ordl13	167.94 MB
2	3	ordl6	167.94 MB	3	0	ordl10	167.94 MB	3	4	ordl14	167.94 MB
2	3	ordl7	167.94 MB	3	0	ordl11	167.94 MB	3	5	iordl1	332.50 MB

SCSI adapter	Device	Structure	Size	SCSI adapter	Device	Structure	Size	SCSI adapter	Device	Structure	Size
3	5	iordl2	332.50 MB	6	1	cust2	917.50 MB	13	6	stk8	563.38 MB
3	5	iordl3	332.50 MB	6	1	icust1	203.38 MB	14	0	stk8	563.38 MB
3	5	iordl4	332.50 MB	6	2	cust2	917.50 MB	14	1	stk8	563.38 MB
3	5	iordl5	332.50 MB	6	2	icust1	203.38 MB	14	2	stk9	563.38 MB
3	5	iordl6	332.50 MB	6	3	cust3	917.50 MB	14	3	stk9	563.38 MB
3	5	iordl7	332.50 MB	6	3	icust1	203.38 MB	14	4	stk9	563.38 MB
3	6	iordl1	332.50 MB	6	4	cust3	917.50 MB	14	5	stk10	563.38 MB
3	6	iordl2	332.50 MB	6	4	icust1	203.38 MB	14	6	stk10	563.38 MB
3	6	iordl3	332.50 MB	6	5	cust4	917.50 MB	15	0	stk10	563.38 MB
3	6	iordl4	332.50 MB	6	5	icust1	203.38 MB	15	1	stk11	563.38 MB
3	6	iordl5	332.50 MB	6	6	cust4	917.50 MB	15	2	stk11	563.38 MB
3	6	iordl6	332.50 MB	7	0	cust5	917.50 MB	15	3	stk11	563.38 MB
3	6	iordl7	332.50 MB	7	1	cust5	917.50 MB	15	4	stk12	563.38 MB
4	0	iordl1	332.50 MB	7	2	cust6	917.50 MB	15	5	stk12	563.38 MB
4	0	iordl2	332.50 MB	7	3	cust6	917.50 MB	15	6	stk12	563.38 MB
4	0	iordl3	332.50 MB	7	4	cust7	917.50 MB	16	0	stk13	563.38 MB
4	0	iordl4	332.50 MB	7	5	cust7	917.50 MB	16	1	stk13	563.38 MB
4	0	iordl5	332.50 MB	7	6	cust8	917.50 MB	16	2	stk13	563.38 MB
4	0	iordl6	332.50 MB	8	0	cust8	917.50 MB	16	3	stk14	563.38 MB
4	0	iordl7	332.50 MB	8	1	cust9	917.50 MB	16	4	stk14	563.38 MB
4	1	iordl1	332.50 MB	8	2	cust9	917.50 MB	16	5	stk14	563.38 MB
4	1	iordl2	332.50 MB	8	3	cust10	917.50 MB	16	6	stk15	563.38 MB
4	1	iordl3	332.50 MB	8	4	cust10	917.50 MB	17	0	stk15	563.38 MB
4	1	iordl4	332.50 MB	8	5	cust11	917.50 MB	17	1	stk15	563.38 MB
4	1	iordl5	332.50 MB	8	6	cust11	917.50 MB	17	2	stk16	563.38 MB
4	1	iordl6	332.50 MB	9	0	cust12	917.50 MB	17	3	stk16	563.38 MB
4	1	iordl7	332.50 MB	9	1	cust12	917.50 MB	17	4	stk16	563.38 MB
4	2	iordl1	332.50 MB	9	2	ord1	1550.00 MB	17	5	stk17	563.38 MB
4	2	iordl2	332.50 MB	9	3	stk18	563.38 MB	17	6	stk17	563.38 MB
4	2	iordl3	332.50 MB	9	4	stk18	563.38 MB	18	1	ware1	9.38 MB
4	2	iordl4	332.50 MB	9	5	stk18	563.38 MB	18	2	ware1	9.38 MB
4	2	iordl5	332.50 MB	9	6	stk1	563.38 MB	18	3	ware1	9.38 MB
4	2	iordl6	332.50 MB	9	6	istk1	255.88 MB	20	0	log1-02	2025.00 MB
4	2	iordl7	332.50 MB	10	0	stk1	563.38 MB	20	1	log2-02	2025.00 MB
4	3	iordl1	332.50 MB	10	0	istk1	255.88 MB	21	0	load1	2045.00 MB
4	3	iordl2	332.50 MB	10	1	stk1	563.38 MB	21	4	stk17	563.38 MB
4	3	iordl3	332.50 MB	10	1	istk1	255.88 MB				
4	3	iordl4	332.50 MB	10	2	stk2	563.38 MB				
4	3	iordl5	332.50 MB	10	2	istk1	255.88 MB				
4	3	iordl6	332.50 MB	10	3	stk2	563.38 MB				
4	3	iordl7	332.50 MB	10	3	istk1	255.88 MB				
4	4	roll1	206.69 MB	10	4	stk2	563.38 MB				
4	4	item1	28.00 MB	10	4	istk1	255.88 MB				
4	5	roll1	206.69 MB	10	5	stk3	563.38 MB				
4	6	log1-01	2025.00 MB	10	5	istk1	255.88 MB				
5	0	roll1	206.69 MB	10	6	stk3	563.38 MB				
5	1	log2-01	2025.00 MB	10	6	istk1	255.88 MB				
5	2	system	610.00 MB	11	0	stk3	563.38 MB				
5	2	cust0	1.00 MB	11	1	stk4	563.38 MB				
5	2	ordl0	1.00 MB	11	2	stk4	563.38 MB				
5	2	iordl0	1.00 MB	11	3	stk4	563.38 MB				
5	2	stk0	1.00 MB	11	4	stk5	563.38 MB				
5	2	temp0	1.00 MB	11	5	stk5	563.38 MB				
5	3	icust2	397.50 MB	11	6	stk5	563.38 MB				
5	4	icust2	397.50 MB	13	0	stk6	563.38 MB				
5	5	icust2	397.50 MB	13	1	stk6	563.38 MB				
5	6	icust2	397.50 MB	13	2	stk6	563.38 MB				
5	6	cust1	917.50 MB	13	3	stk7	563.38 MB				
6	0	cust1	917.50 MB	13	4	stk7	563.38 MB				
6	0	icust1	203.38 MB	13	5	stk7	563.38 MB				

5. Clause 5: Performance Metrics & Response Time

5.1. Measured Throughput (tpmC)

Measured tpmC must be reported.

The measured tpmC was 6253.32.

5.2. Response Times

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

Table 5.1: Response Time Data

Transaction	Average	Maximum	90th %ile
New-Order	2.10	25.49	4.56
Payment	1.27	16.07	2.64
Order-Status	1.66	24.48	3.26
Delivery	0.46	4.45	0.57
Delivery (Deferred)	n/a	n/a	10.93
Stock-Level	5.42	43.73	11.03
Menu	0.45	65.49	0.59

5.3. Keying and Think Times

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

Table 5.2: Keying Times

Transaction	Minimum	Average	Maximum
New-Order	18.00	18.03	18.26
Payment	3.00	3.01	3.20
Order-Status	2.00	2.01	2.12
Delivery	2.00	2.01	2.12
Stock-Level	2.00	2.01	2.13

Table 5.3: Think Times

Transaction	Minimum	Average	Maximum
New-Order	0.00	12.38	118.64
Payment	0.00	12.31	120.35
Order-Status	0.00	10.26	97.76
Delivery	0.00	5.12	42.81
Stock-Level	0.00	5.13	42.58

5.4. Response Time Frequency Distribution Curves

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

Figure 5.1: New Order Response Time Distribution

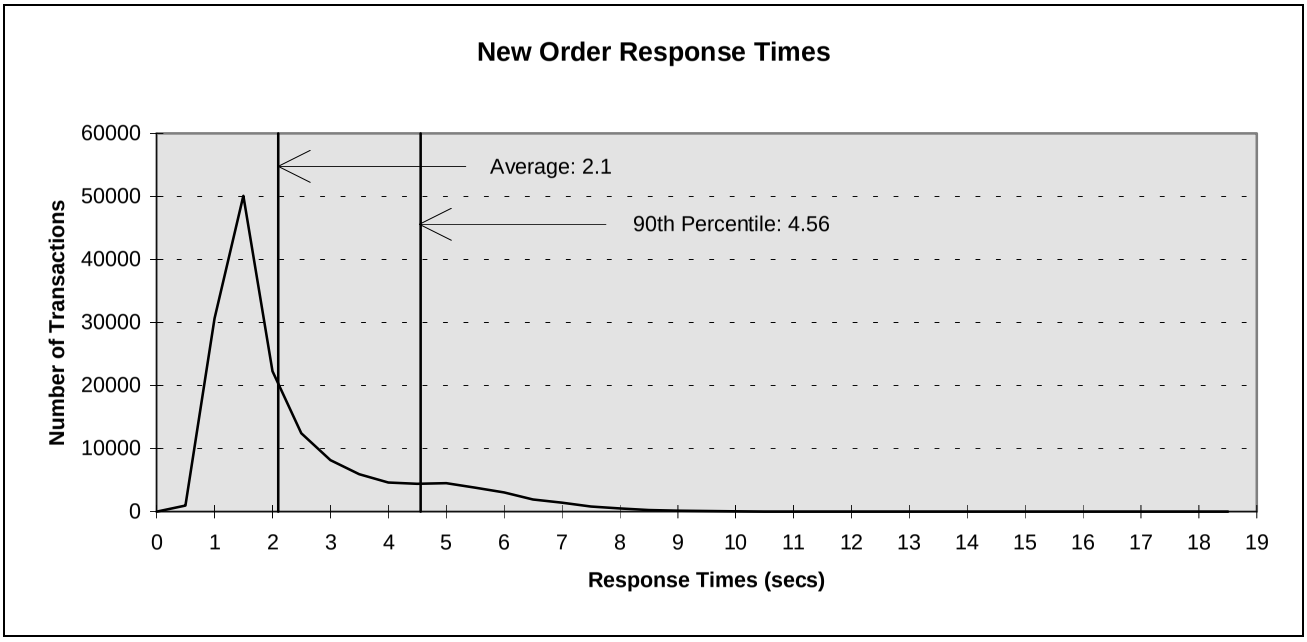


Figure 5.2: Payment Response Time Distribution

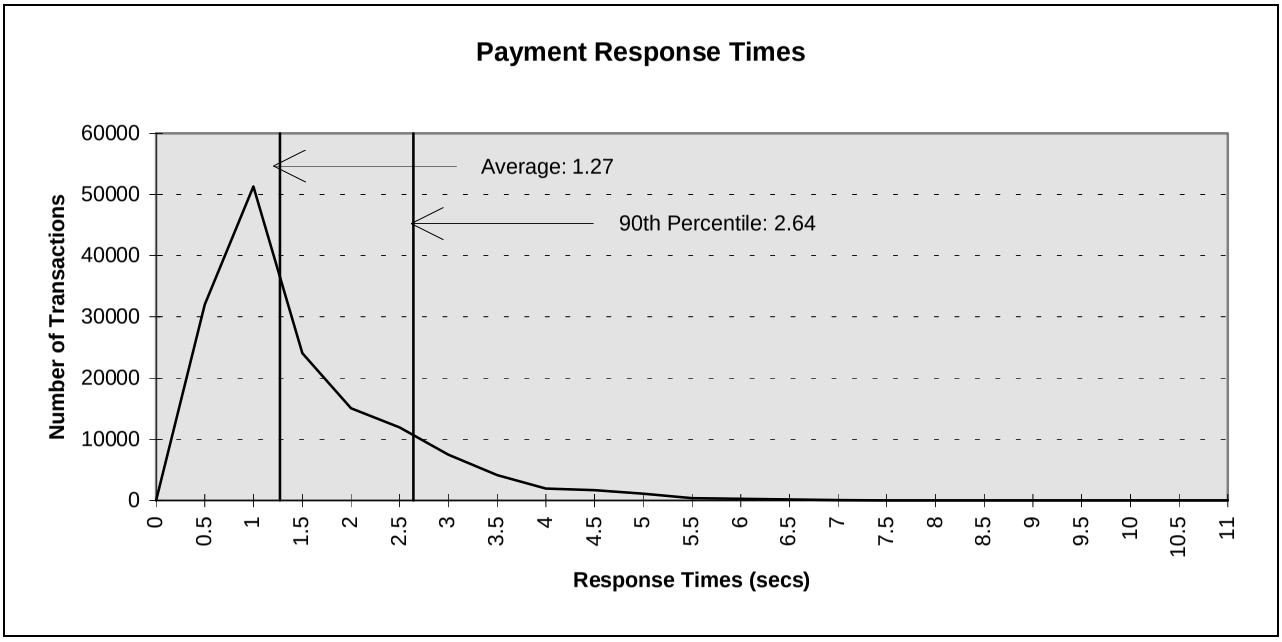


Figure 5.3: Order Status Response Time Distribution

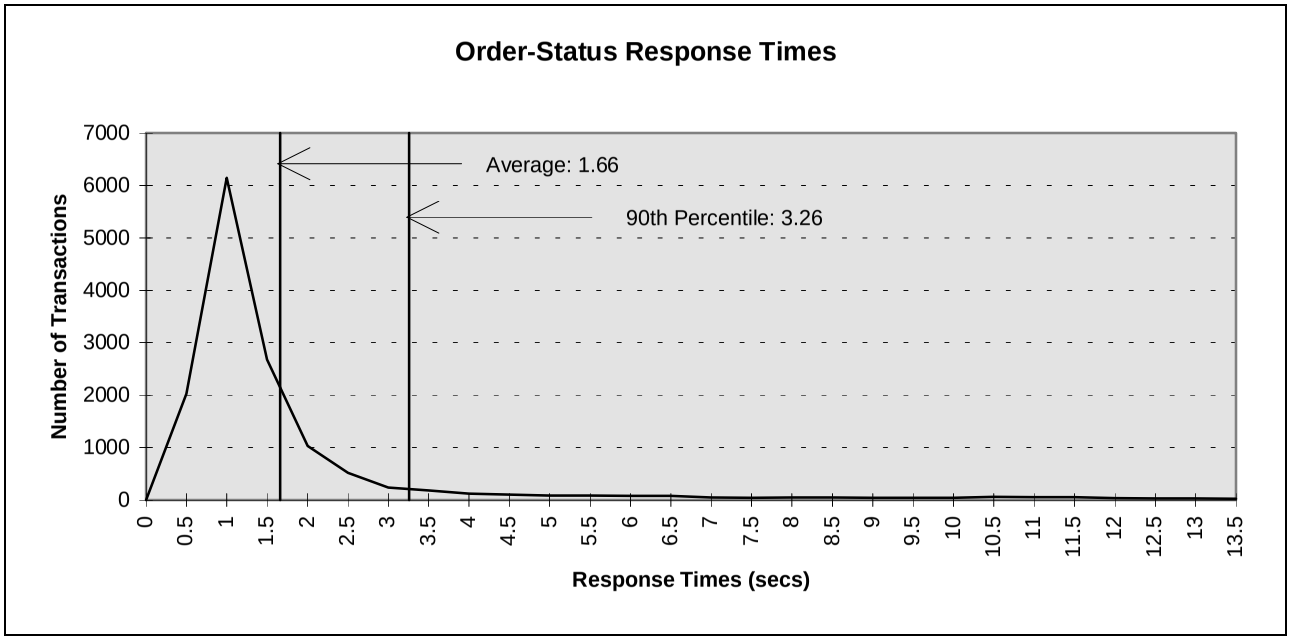


Figure 5.4: Delivery Response Time Distribution

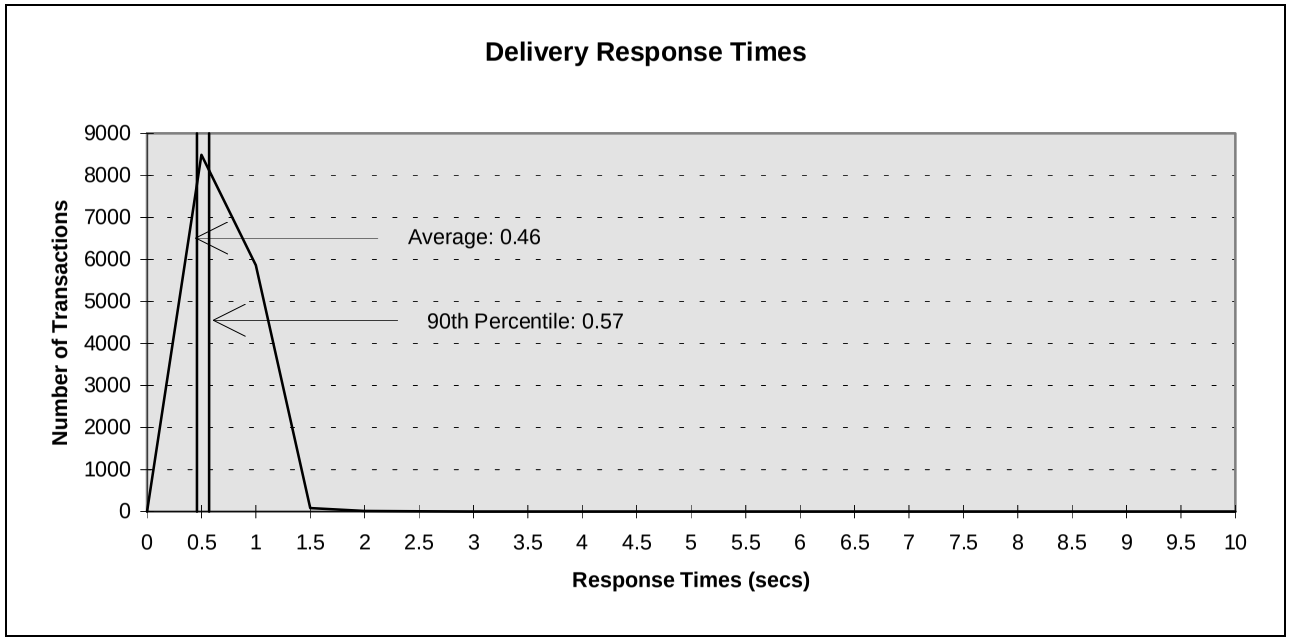
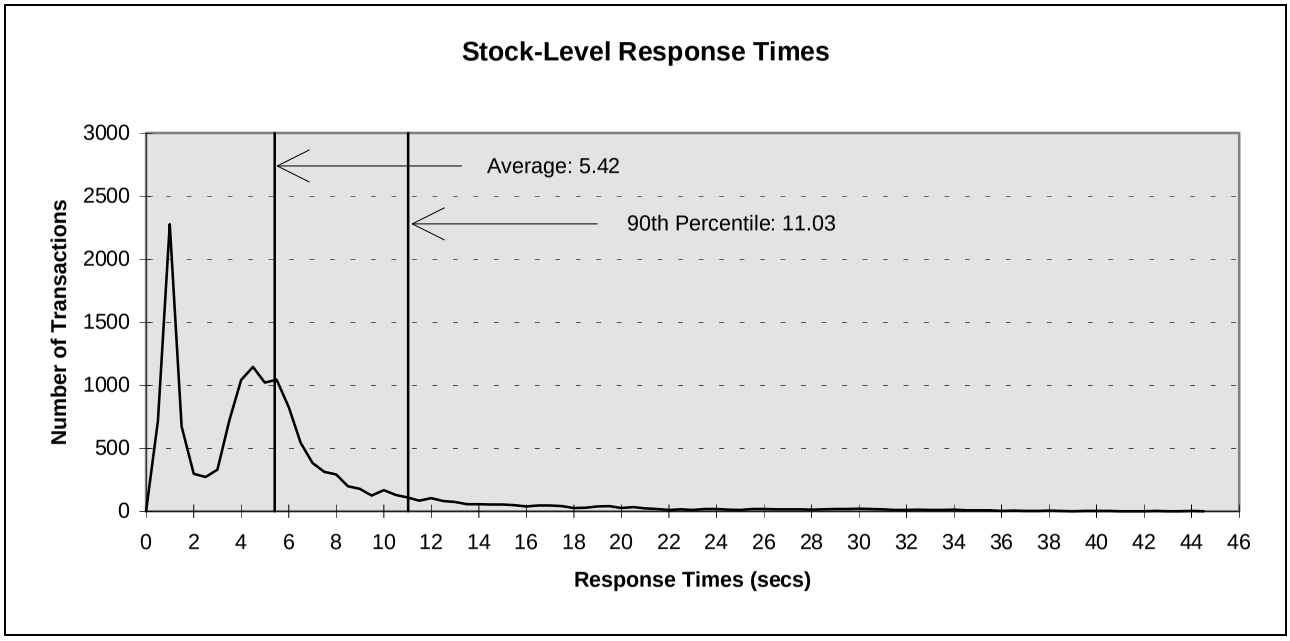


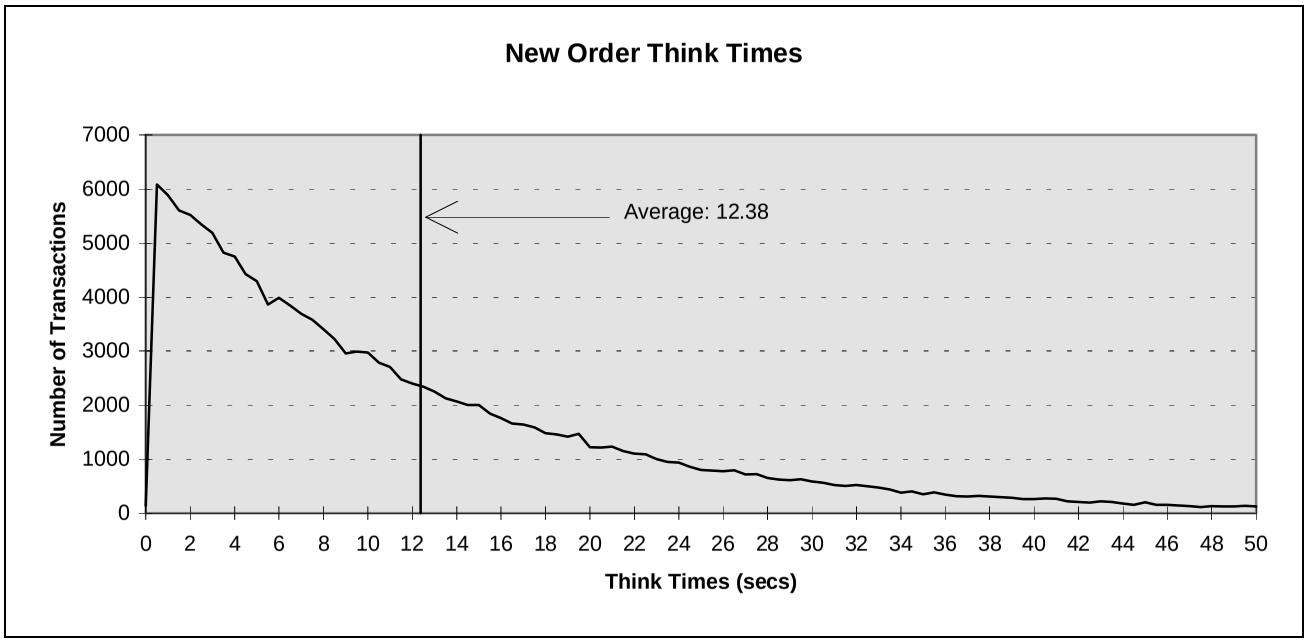
Figure 5.5: Stock Level Response Time Distribution



5.5. New Order Think Time Frequency Distribution Curve

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

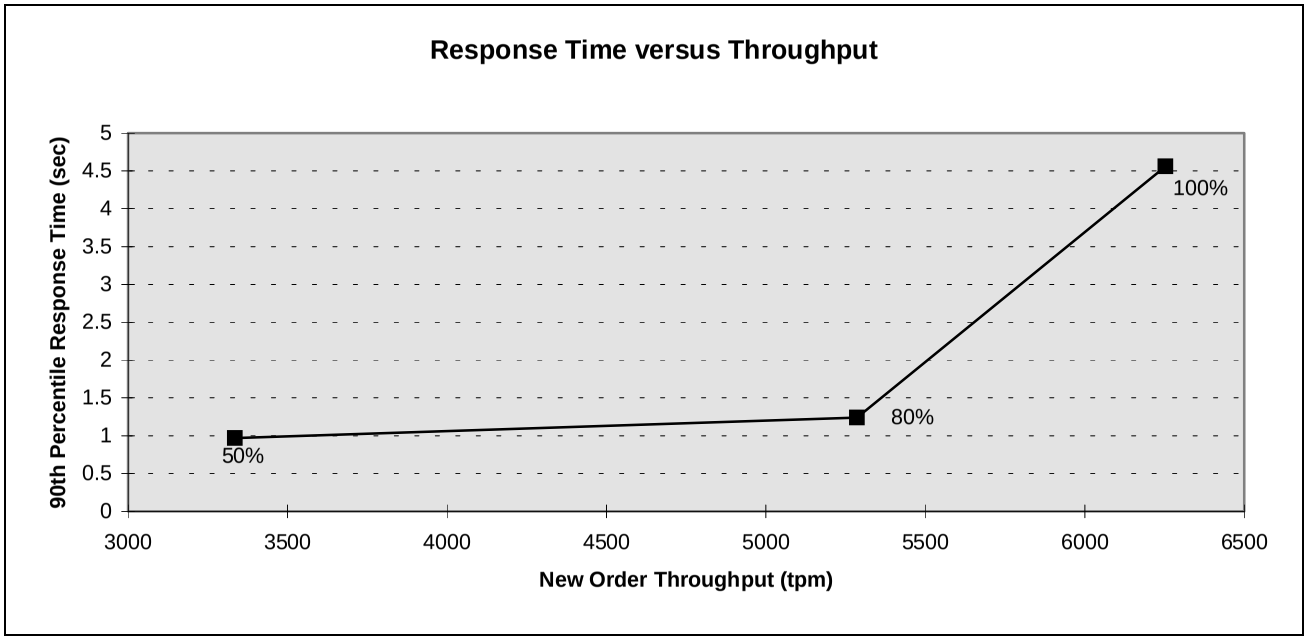
Figure 5.6: New Order Think Time Distribution



5.6. Response Time versus Throughput Performance Curve

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

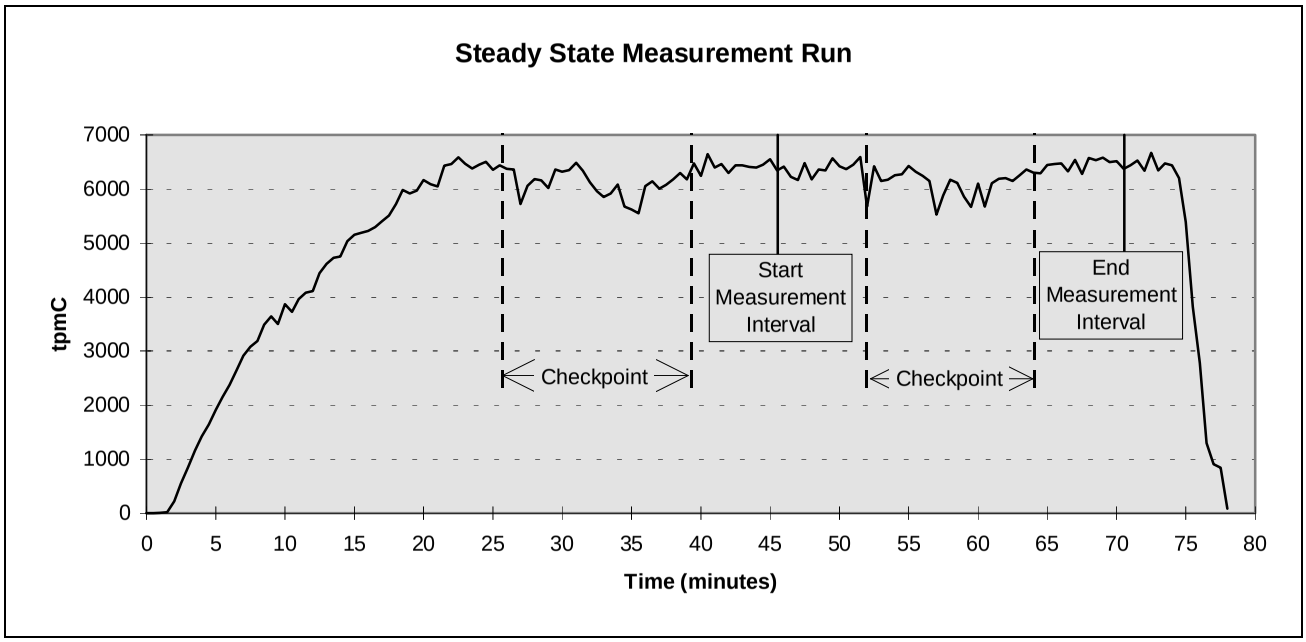
Figure 5.7: Response Time versus Throughput



5.7. New-Order Throughput vs. Time

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

Figure 5.8: Throughput (tpmC) versus Time



5.8. Determination of "Steady State"

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

The transaction throughput rate (tpmC) and response time were relatively constant after the initial 'ramp up' period. The throughput and response time behavior were determined by examining data reported for each 30-second interval over the duration of the benchmark. Ramp-up, steady state, and ramp-down regions are discernible in the graph presented in Figure 6.17.

5.9. Work Performed During Steady State

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

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

The RTE then waited the required think time interval before repeating the process, starting with the selection of a transaction from the menu.

An RTE transmission was sent to an application process running on the client machines through Ethernet LANs. These client application processes handled all screen I/O and passed their requests to the Tuxedo software which handled all requests to the database on the server. Tuxedo was configured with the following number of queues: 3 delivery queues, 19 new order queues, 7 stock level queues, 7 payment queues and 1 order status queue. Tuxedo communicated with the database server over another Ethernet LAN using Oracle7 SQL*NET v2.1.

The RTE submitted deferred delivery requests to the client application process which in turn submitted these delivery transactions to Tuxedo with 2 delivery queues per client machine.

To perform checkpoints at specific intervals, we set Oracle7's checkpoint interval to the maximum allowable value and wrote a script to schedule multiple log switches at specific intervals which forced checkpoints to occur. ORACLE automatically logs all check points to an alert file on the server. The script included a wait time between each check point equal to the measurement interval which was 25 minutes. The check point script was started manually after RTE had all users logged in and sending transactions and a steady and a steady state had been achieved.

At each check point, Oracle7 wrote to disk all buffer pages that had been updated but not yet physically written to the disk. Upon completion of the check point, Oracle7 wrote timestamps to all files indicating when last checkpoint successfully completed. The positioning of the check point was verified to be clear of the guard Zones and is depicted on the graph in figure.

Serializable Transactions : Oracle7 supports serializable transaction isolation in full compliance with the SQL92 and TPC-C requirements. This is implemented by extending 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 the readers and writers coexist without blocking one another, provided 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 command will prevent 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 the scope of read consistency from individual query to the entire transaction itself. ALL reads by serializable transaction are therefore repeatable, as the transaction will access prior versions of data changed (or deleted) by other transactions after the start of serializable transactions. Thus, a serializable transaction sees a fixed snapshot of database, established at the beginning of transaction.

To ensure proper isolation, a serializable transaction cannot modify the rows that were changed by other transactions after the beginning of serializable transaction , update(or delete) statement will fail with error ORA_08177 : "cannot serialize access" and statement will be 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 the point , execute additional statements, or rollback the entire transactions. Repeated attempts to execute the same statement will always fail with the error "Can't serialize access" unless 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, conflict between transactions rollback and restarts or commits without re-executing the statement receiving error.

5.10. Reproducibility

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

In a repeat test, carried out in the same manner as the primaty test, a throughput of 6199.44 tpmC was achieved on the same database during a 25-minute, steady state run. All required transaction statistics were met except for the guard zones at the start and end of the measurement interval. See the Auditor's attestation letter for details.

5.11. Measurement Interval Duration

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

The measurement interval was 25 minutes.

5.12. Regulation of Transaction Mix

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

The RTE used for the benchmark is proprietary and has been developed as per the guidelines in the TPC-C Spec.

The basic skeleton of the RTE is described below:

As a first step, the Tuxedo servers are all booted up on all the client machines. One copy of the program 'csc' is started per client, on the RTE. This program accepts the hostname and the starting Emulated Station number as input and starts the specified number of users on the client. The users start submitting transaction requests once they enroll with the Tuxedo bulletin board and write the 5 timestamps that are needed to measure the Think, Menu, Keyin and Response times to a transaction log. In addition, each New Order transaction also writes an entry in the success log file. The system is said to be fully ramped up once the last user has signed on to the client. The sampling interval is recorded as time stamps in the transaction log file. At the end of the run, the transaction log file is reduced to produce all the statistics necessary for the full disclosure report. Errors are recorded and failed users are restarted, before the ramp up has completed.

The transaction mix is calculated as depicted in the code fragment in Appendix D. The transaction weights are initialized as follows:

NEW_ORDER_PROB	.445
PAYMENT_PROB	.432
ORDER_STATUS_PROB	.041
DELIVER_PROB	.041
STOCK_LEVEL_PROB	.041

No online mix adjustments are made.

5.13. Transaction Statistics

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

The percentage of New-Order transactions rolled back as a result of invalid item number must be disclosed.

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

The percentage of remote order-lines entered per New-Order transaction must be disclosed.

The percentage of remote Payment transactions must be disclosed.

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

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

Table 5.4 shows this information.

Table 5.4: Transaction Statistics

Transaction Type	Statistics	Value
New Order	Rolledback transactions	1.03 %
	Home warehouse	99.00%
	Remote warehouse	1.00%
	Average Items per Order	10.00
Payment	Home warehouse	84.88%
	Remote warehouse	15.12%
	Non-primary key access	59.83%
Order Status	Non-primary key access	59.18%
Delivery	Skipped transactions (Interactive)	0
	Skipped transaction counts (Deferred)	0
	Skipped District counts (Deferred)	0
Transaction Mix	New Order Payment Order Status Delivery Stock Level	44.56% 43.22% 4.03% 4.12% 4.05%

5.14. Checkpoint Statistics

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

There is one checkpoint in the measurement interval, beginning approximately 6.5 minutes from the start of the measurement window and ending approximately 6.5 minutes before the end. The checkpoint interval is 25 minutes (from the start of one to the start of the next) and a checkpoint lasts approximately 12 minutes. In conformance with Clause 5.2.2 there is no checkpoint within a span of 25/4 seconds before or after the beginning or end of the measurement interval.

6. Clause 6: SUT, Driver & Communiations Definition

6.1. Remote Terminal Emulator (RTE) Description

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

The RTE is an internal Unisys tool. Appendix D contains the portions of the RTE that select the transactions and generate the transaction input data.

6.2. Emulated Components

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

There were no emulated components in the benchmark configuration other than the emulated users' workstation.

6.3. Functional Diagrams

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

Figures 0.1 and 0.2 (in Section 0) show functional diagrams of the benchmark and proposed systems. A description of the RTE is provided in Section 5.12 and of the client/server benchmark software in Section 5.9.

6.4. Network Configuration

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

Figures 0.1 and 0.2 (in Section 0) also diagram the network configurations of the benchmark and configured systems and represent the Driver connected via LAN replacing the user PCs that are directly connected via LAN.

6.5. Network Bandwidth

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

Ethernet local area networks (LAN) with a bandwidth of 10 megabits per second are used in the tested/priced configurations.

6.6. Operator Intervention

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

No operator intervention was required.

7. Clause 7: Pricing

7.1. Pricing

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

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

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

7.1.1. System Pricing

Each priced configuration consists of an integrated system package, additional options, and components. Prices for all Unisys Corporation products are US list prices. a one (1) year warranty is standard with all Unisys Corporation products.

7.1.2. Support Pricing

The five year support pricing for Unisys Corporation products is based on forty-eight (48) months of monthly support costs; sixty (60) months minus the twelve month warranty period. The Oracle support pricing is based on sixty (60) months of monthly support costs. The following support products were placed in the benchmark:

- Unisys Corporation four hour onsite repair hardware support,
- Unisys Corporation telephone support for software and updates and
- Oracle Bronze Customer Care.

7.1.2.1.Unisys Hardware Support

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

7.1.2.2.Unisys Software Support

Unisys Corporation software support provides the following:
- Access to Unisys Technical Assistance Center for fault isolation and problem solving assistance
- Two hour call return and immediate response for critical calls
- Electronic access to product support information and software patches.

7.1.3. ORACLE Technical Support

Five years of support is priced for Oracle7.

Oracle Bronze Customer Care includes:
- Real-Time Telephone Technical Assistance from 5:00 a.m. to 6:00 p.m., PST
- Product updates
- Online access to the Real Time Support System (RTSS)
- Online access to Oracle Worldwide Support's newsletter
- Subscription to Oracle Worldwide Support's newsletter
- Access to Oracle Worldwide Support's private forum on CompuServe

7.1.4. Discounts

The following generally available discounts were applied to the priced configurations:
- A Unisys Corporation 15% support quantity discount
- A Unisys Corporation 5% support prepayment discount

7.2. Availability

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

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

All products in the priced system are currently available.

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

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

Unisys Corporation U6000/500 Model 80 with Oracle7, release 7.3, was 6253.32 tpmC at $303.80 per tpmC. Available now.

7.4. Country-Specific Pricing

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

None.

7.5. Usage Pricing

For any usage pricing, the sponsor must disclose:

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

Oracle7 was priced for an unlimited number of Users.

One unlimited User License was priced for the UnixWare 2.1 server.

Six unlimited User Licenses were priced for the UnixWare 2.1 clients.

Six 2048-user licenses were priced for the EtherPlex cards.

8. Clause 8 : Full Disclosure Availablity

8.1. Availability

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

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

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

9. *Clause 9 : Audit*

9.1. Auditor's Report

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

This implementation of the TPC Benchmark C on the Unisys U6000/500 Model 80 was audited by Richard Gimarc, a TPC certified auditor of:

Performance Metrics, Inc.
1 North Santa Cruz Avenue
Suite 203
Los Gatos, CA 95030
Telephone: 408.737.0406
Email: rgimarc@hooked.net

The attestation letter is shown on the next page.

PERFORMANCE METRICS INC.
TPC Certified Auditors

July 15, 1996

Jerrold Buggert
Director, Systems Analysis, Modeling and Measurement
Unisys Corporation
25725 Jeronimo
Mission Viejo, CA 92691

Karl Haas
Director, Open Systems Performance Group
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065

I have verifiedremotelythe TPC Benchmark™ CC/S for the following configuration:

Platform: U6000/500 Model 80
Database Manager: Oracle7 Server Version 7.3.2
Operating System: SCO UnixWare Version 2.1
Transaction Manager:Tuxedo Version 6.1

CPUs	Memory	Disks	New Order Response Time @ 90%	tpmC C/S
Server: U6000/500 Model 80				
10 Pentium @ 150 MHz	2 GB	57 ST32550N @ 2 GB 78 ST15150N @ 4 GB	4.56 sec	6,253.32
6 Clients: U6000/200 Model 30				
1 Pentium @ 90 MHz	192 MB	1 ST31230N@ 1 GB	n.a.	n.a.

In my opinion, these performance results were produced in compliance with the TPC requirements for the benchmark.

The following attributes of the benchmark were given special

- The transactions were correctly implemented
- The databasefiles were properly sized and populated.

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

- Sufficient 4 GB disks were added to the priced configuration to satisfy 8 hours of mirrored log space.
- A 4 GB disk was substituted for each of the 2 GB disks on the server in the priced configuration. Based on collected performance data, we believe this substitution would not degrade performance.
- The data for the 180-day space calculation was verified. To satisfy the 180-day space requirement six 4 GB disks were added to the priced configuration.
- Measurement cycle times did not include delays for emulated components since workstations were priced.
- The steady state portion of the test was 25 minutes.
- One checkpoint was taken during the steady state portion of the test.
- The checkpoint was verified to be clear of the guard zones. See Audit Notes below for deviation on reproducibility run.
- There were 5,500 user contexts present on the system.
- Each emulated user had a unique starting random number seed.
- System pricing was checked for major components and maintenance.

Audit Notes:

- The checkpoint taken during the reproducibility measurement interval took approximately 15 minutes to complete. It started 31 seconds into the opening guard zone and ended almost 2 minutes into the ending guard zone. The steady state graph showed little if any advantage gained, and the result was accepted under ease of benchmarking.

  In my opinion, this deviation had little affect on the result.

Regards,

Richard L. Gimarc
Auditor

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

Appendix A - Client/Server Source

ansi.c

```
/********************** Interface Module for Liberty
********************/
#include <stdio.h>
#include <string.h>
#include <termio.h>
#include <fcntl.h>
#include <time.h>
#include "screens.h"
#include "tpcc_info.h"
#include "utils.h"
#include "do_tpcc.h"
#include "trans_type.h"

#define C (int)0
#define U (int)-1
#define D (int)-2
#define R (int)-3
#define L (int)-4
#define E (int)-5
#define M (int)-6
#define P (int)-7
#define N (int)-8
#define H (int)-9

/* Screen Function Macros */

#define scr_clr write(ofd,cs_str,strlen(cs_str))
/*
Thomas
#define CR write(ofd,&CCright,1)
#define CL write(ofd,&CCleft,1)
*/

#define CR write(ofd,&CCright[0],strlen(CCright))
#define CL write(ofd,&CCleft[0], strlen(CCleft))

#define CCL 8
#define CCR 12
#define CCU 11
#define CCD 22
#define CCH 30
#define RUB 127
#define NC '\0'

/* Forms Buffer Macros */

/* Define Input Fields */


#define dNOf(x,y,label) scurpos(x,y,&NObuf[strlen(NObuf)]);\
                        strcat(NObuf,label);
#define dPAf(x,y,label) scurpos(x,y,&PAbuf[strlen(PAbuf)]);\
                        strcat(PAbuf,label);
#define dOSf(x,y,label) scurpos(x,y,&OSbuf[strlen(OSbuf)]);\
                        strcat(OSbuf,label);
#define dIDf(x,y,label) scurpos(x,y,&IDbuf[strlen(IDbuf)]);\
                        strcat(IDbuf,label);
#define dSLf(x,y,label) scurpos(x,y,&SLbuf[strlen(SLbuf)]);\
                        strcat(SLbuf,label);

/* Format Output Buffer */


#define dNOd(x,y,n) scurpos(x,y,p);\
                        NOndx[i++]=(p+=CPLEN);\
                        *p=' ';\
                        p+=n
#define dPAd(x,y,n) scurpos(x,y,p);\
                        PAndx[i++]=(p+=CPLEN);\
                        *p=' ';\
                        p+=n
#define dOSd(x,y,n) scurpos(x,y,p);\
                        OSndx[i++]=(p+=CPLEN);\
                        *p=' ';\
                        p+=n
#define dIDd(x,y,n) scurpos(x,y,p);\
                        IDndx[i++]=(p+=CPLEN);\
                        *p=' ';\
                        p+=n
#define dSLd(x,y,n) scurpos(x,y,p);\
                        SLndx[i++]=(p+=CPLEN);\
                        *p=' ';\
                        p+=n

/* Stuff for Numeric Formatter */
```

```
#include <math.h>
#define NCHARS 32
static char number[NCHARS];
#define DIGIT(x) {*--p=(char)(x%10+'0'), x/=10;}
#define billion 1000000000.0

#define  MEbufSZ  100
#define  NObufSZ  1024+MEbufSZ
#define  PAbufSZ  512+MEbufSZ
#define  OSbufSZ  512+MEbufSZ
#define  IDbufSZ  512+MEbufSZ
#define  SLbufSZ  128+MEbufSZ

#define  NOoutSZ  2048
#define  PAoutSZ  900
#define  OSoutSZ  1300
#define  IDoutSZ  100
#define  SLoutSZ  64


/* Global vars */

char dummy[3];    /* used by the RTE */
int MEbufsz,NObufsz, PAbufsz, OSbufsz, IDbufsz, SLbufsz;
int NOdmax, PAdmax, OSdmax, IDdmax, SLdmax;
/* Following declarations modfied
char
MEbuf[8192],NObuf[8192],PAbuf[8192],OSbuf[8192],IDbuf[8192],SLbuf[8192];
char NOout[8192],PAout[8192],OSout[8192],IDout[8192],SLout[8192];
*/
char MEbuf[MEbufSZ];
char NObuf[NObufSZ];
char PAbuf[PAbufSZ];
char OSbuf[OSbufSZ];
char IDbuf[IDbufSZ];
char SLbuf[SLbufSZ];

char NOout[NOoutSZ];
char PAout[PAoutSZ];
char OSout[OSoutSZ];
char IDout[IDoutSZ];
char SLout[SLoutSZ];

char *NOndx[150], *PAndx[50], *OSndx[100], *IDndx[10], *SLndx[10];

char cls[18]={CCL,CCL,CCL,CCL,CCL,CCL,CCL,CCL,CCL,CCL,CCL,CCL,CCL,CCL,CCL,
              CCL,CCL,CCL};
/*
Thomas
char CCup=CCU;
char CCdown=CCD;
char CCright =CCR;
char CChome=CCH;

*/

/*
vt100 or ansi specific sequences ,  Thomas
*/
#define MAXSEQLEN     3
char CCup[]="\033[A";
char CCdown[]= "\033[B";
char CCright[] = "\033[C";
char CChome[]= "\033[H";
char CCleft[] = "\010";
/*
char CCleft[] = "\033[D";
*/

/*
Thomas
char cs_str[]="\036\033*";
*/

char cs_str[]="\033[2J";
char cl_str[]="\033T";
char pos_str[]="\033[xx;yyH";
static char get_tr_str[] = "\033[23;1H\033[24;74H\0";
#define GET_TR_STRLEN 16

/*
#define CPLEN 4   */ /* length of pos_str */
#define CPLEN (strlen(pos_str))
#define ID_CPLEN       7
#define ITEM_CPLEN     8

char BELL='\7';
static unsigned char cr, cc;
static struct termio its,ots,tty;
static int res;
static long tflag;
extern int ifd, ofd;

/* extern FILE *errf; abi add */
#ifdef ANSI_DEBUG
static void dump_input();
#endif

extern
void init_forms(w_id, stkl_d_id)
        short int w_id, stkl_d_id;
{
        scurpos(24,1,MEbuf);
        strcat(MEbuf,
        "NewOrder(1) Payment(2) OrderStatus(3) Delivery(4) StockLevel(5)
Quit(6): ");
        dNOf(1,36,"New Order")
        dNOf(2,1,"Warehouse: ")
        sprintf(&NObuf[strlen(NObuf)],"%d",w_id);
        dNOf(2,19,"District: __")
        dNOf(2,55,"Date: ")
        dNOf(3,1,"Customer:  ____")
        dNOf(3,19,"Name: ")
        dNOf(3,44,"Credit: ")
        dNOf(3,57,"Disc: ")
        dNOf(4,1,"Order Number: ")
        dNOf(4,25,"Number of Lines: ")
        dNOf(4,52,"W_tax: ")
        dNOf(4,67,"D_tax: ")
        dNOf(6,2,"Supp_W  Item_ID  Item Name")
        dNOf(6,45,"Qty  Stock  B/G  Price    Amount")
        dNOf(7,3,"____")
```

```
dNOf(7,10,"______")
dNOf(7,45,"__")
dNOf(8,3,"____")
dNOf(8,10,"______")
dNOf(8,45,"__")
dNOf(9,3,"____")
dNOf(9,10,"______")
dNOf(9,45,"__")
dNOf(10,3,"____")
dNOf(10,10,"______")
dNOf(10,45,"__")
dNOf(11,3,"____")
dNOf(11,10,"______")
dNOf(11,45,"__")
dNOf(12,3,"____")
dNOf(12,10,"______")
dNOf(12,45,"__")
dNOf(13,3,"____")
dNOf(13,10,"______")
dNOf(13,45,"__")
dNOf(14,3,"____")
dNOf(14,10,"______")
dNOf(14,45,"__")
dNOf(15,3,"____")
dNOf(15,10,"______")
dNOf(15,45,"__")
dNOf(16,3,"____")
dNOf(16,10,"______")
dNOf(16,45,"__")
dNOf(17,3,"____")
dNOf(17,10,"______")
dNOf(17,45,"__")
dNOf(18,3,"____")
dNOf(18,10,"______")
dNOf(18,45,"__")
dNOf(19,3,"____")
dNOf(19,10,"______")
dNOf(19,45,"__")
dNOf(20,3,"____")
dNOf(20,10,"______")
dNOf(20,45,"__")
dNOf(21,3,"____")
dNOf(21,10,"______")
dNOf(21,45,"__")
dNOf(22,1,"Execution Status: ")
dNOf(22,62,"Total: ");

strcat(PAbuf,cs_str);
dPAf(1,38,"Payment")
dPAf(2,1,"Date: ")
dPAf(4,1,"Warehouse: ")
sprintf(&PAbuf[strlen(PAbuf)],"%d",w_id);
dPAf(4,42,"District: __")
dPAf(9,1,"Customer: ____")
dPAf(9,17,"Cust-Warehouse: ____")
dPAf(9,39,"Cust-District: __")
dPAf(10,1,"Name: ")
dPAf(10,29,"________________")
dPAf(10,50,"Since: ")
dPAf(11,50,"Credit: ")
dPAf(12,50,"% Disc: ")
dPAf(13,50,"Phone: ")
dPAf(15,1,"Amount Paid: ")
dPAf(15,23,"$_______")
dPAf(15,38,"New Cust Balance:")
dPAf(16,1,"Credit Limit: ")
dPAf(18,1,"Cust-Data: ")

strcat(OSbuf,cs_str);
dOSf(1,35,"Order Status")
dOSf(2,1,"Warehouse: ")
sprintf(&OSbuf[strlen(OSbuf)],"%d",w_id);
dOSf(2,19,"District: __")
dOSf(3,1,"Customer: ____   Name: ")
dOSf(3,44,"________________")
dOSf(4,1,"Cust-Balance: ")
dOSf(6,1,"Order-Number: ")
dOSf(6,26,"Entry-Date: ")
dOSf(6,60,"Carrier-Number: ")
dOSf(7,1,"Supply-W ")
dOSf(7,14,"Item-ID ")
dOSf(7,25,"Qty ")
dOSf(7,33,"Amount ")
dOSf(7,45,"Delivery-Date")

strcat(IDbuf,cs_str);
dIDf(1,35,"Delivery")
dIDf(2,1,"Warehouse: ")
sprintf(&IDbuf[strlen(IDbuf)],"%d",w_id);
dIDf(4,1,"Carrier-Number: __")
dIDf(6,1,"Execution Status: ")

strcat(SLbuf,cs_str);
dSLf(1,35,"Stock Level")
dSLf(2,1,"Warehouse: ")
sprintf(&SLbuf[strlen(SLbuf)],"%d",w_id);
dSLf(2,19,"District: ")
sprintf(&SLbuf[strlen(SLbuf)],"%d",stkl_d_id);
dSLf(4,1,"Stock Level Threshold: __")
dSLf(6,1,"Low Stock: ")

/* +++++ */
        strcat(NObuf, MEbuf);
        strcat(PAbuf, MEbuf);
        strcat(OSbuf, MEbuf);
        strcat(IDbuf, MEbuf);
        strcat(SLbuf, MEbuf);
/* +++++ */
        MEbufsz=strlen(MEbuf);
        NObufsz=strlen(NObuf);
        PAbufsz=strlen(PAbuf);
        OSbufsz=strlen(OSbuf);
        IDbufsz=strlen(IDbuf);
        SLbufsz=strlen(SLbuf);
/* ++++++++ */
        NObuf[NObufsz] = NC;
        NObufsz += 1;
        PAbuf[PAbufsz] = NC;
        PAbufsz += 1;
        OSbuf[OSbufsz] = NC;
        OSbufsz += 1;
        IDbuf[IDbufsz] = NC;
```

```
        IDbufsz += 1;
        SLbuf[SLbufsz] = NC;
        SLbufsz += 1;
/* ++++++++ */
#ifdef TEST
        printf ("%d\n",MEbufsz);
        printf ("%d\n",NObufsz);
        printf ("%d\n",PAbufsz);
        printf ("%d\n",OSbufsz);
        printf ("%d\n",IDbufsz);
        printf ("%d\n",SLbufsz);
#endif
        init_outbuf();
}

init_outbuf()
{
        char *p;
        int i;

/*New Order*/
        p=NOout;
        i=0;
        memset(p,' ',NOoutSZ);
        /*Fixed fields...*/
        dNOd( 2,29,2); /* 0:D_ID*/
        dNOd( 3,12,4); /* 1:C_ID*/
        dNOd( 2,61,19); /* 2:DATE*/
        dNOd( 3,25,16); /* 3:C_LAST*/
        dNOd( 3,52, 2); /* 4:C_CREDIT*/
        dNOd( 3,64, 5); /* 5:C_DISCOUNT*/
        dNOd( 4,59, 5); /* 6:W_TAX*/
        dNOd( 4,74, 6); /* 7:D_TAX*/
        dNOd( 4,42, 2); /* 8:O_OL_CNT*/
        dNOd( 4,15, 8); /* 9:O_ID*/
        dNOd(22,70, 9); /*10:O_TOTAL*/
        /*Order Lines...*/
        dNOd( 7,3 , 4); /*11:SUPP_W_1*/
        dNOd( 7,10, 6); /*12:ITEM_ID_1*/
        dNOd( 7,20,24); /*13:I_NAME_1*/
        dNOd( 7,45, 2); /*14:I_QUANTITY_1*/
        dNOd( 7,51, 3); /*15:I_STOCK_1*/
        dNOd( 7,58, 1); /*16:BRAND_GENERIC_1*/
        dNOd( 7,62, 7); /*17:I_PRICE_1*/
        dNOd( 7,71, 8); /*18:OL_AMOUNT_1*/
        dNOd( 8,3 , 4);
        dNOd( 8,10, 6);
        dNOd( 8,20,24);
        dNOd( 8,45, 2);
        dNOd( 8,51, 3);
        dNOd( 8,58, 1);
        dNOd( 8,62, 7);
        dNOd( 8,71, 8);
        dNOd( 9,3 , 4);
        dNOd( 9,10, 6);
        dNOd( 9,20,24);
        dNOd( 9,45, 2);
        dNOd( 9,51, 3);
        dNOd( 9,58, 1);
        dNOd( 9,62, 7);
        dNOd( 9,71, 8);
        dNOd(10,3 , 4);
        dNOd(10,10, 6);
        dNOd(10,20,24);
        dNOd(10,45, 2);
        dNOd(10,51, 3);
        dNOd(10,58, 1);
        dNOd(10,62, 7);
        dNOd(10,71, 8);
        dNOd(11,3 , 4);
        dNOd(11,10, 6);
        dNOd(11,20,24);
        dNOd(11,45, 2);
        dNOd(11,51, 3);
        dNOd(11,58, 1);
        dNOd(11,62, 7);
        dNOd(11,71, 8);
        dNOd(12,3 , 4);
        dNOd(12,10, 6);
        dNOd(12,20,24);
        dNOd(12,45, 2);
        dNOd(12,51, 3);
        dNOd(12,58, 1);
        dNOd(12,62, 7);
        dNOd(12,71, 8);
        dNOd(13,3 , 4);
        dNOd(13,10, 6);
        dNOd(13,20,24);
        dNOd(13,45, 2);
        dNOd(13,51, 3);
        dNOd(13,58, 1);
        dNOd(13,62, 7);
        dNOd(13,71, 8);
        dNOd(14,3 , 4);
        dNOd(14,10, 6);
        dNOd(14,20,24);
        dNOd(14,45, 2);
        dNOd(14,51, 3);
        dNOd(14,58, 1);
        dNOd(14,62, 7);
        dNOd(14,71, 8);
        dNOd(15,3 , 4);
        dNOd(15,10, 6);
        dNOd(15,20,24);
        dNOd(15,45, 2);
        dNOd(15,51, 3);
        dNOd(15,58, 1);
        dNOd(15,62, 7);
        dNOd(15,71, 8);
        dNOd(16,3 , 4);
        dNOd(16,10, 6);
        dNOd(16,20,24);
        dNOd(16,45, 2);
        dNOd(16,51, 3);
        dNOd(16,58, 1);
        dNOd(16,62, 7);
        dNOd(16,71, 8);
        dNOd(17,3 , 4);
        dNOd(17,10, 6);
        dNOd(17,20,24);
        dNOd(17,45, 2);
        dNOd(17,51, 3);
```

```
        dNOd(17,58, 1);
        dNOd(17,62, 7);
        dNOd(17,71, 8);
        dNOd(18,3 , 4);
        dNOd(18,10, 6);
        dNOd(18,20,24);
        dNOd(18,45, 2);
        dNOd(18,51, 3);
        dNOd(18,58, 1);
        dNOd(18,62, 7);
        dNOd(18,71, 8);
        dNOd(19,3 , 4);
        dNOd(19,10, 6);
        dNOd(19,20,24);
        dNOd(19,45, 2);
        dNOd(19,51, 3);
        dNOd(19,58, 1);
        dNOd(19,62, 7);
        dNOd(19,71, 8);
        dNOd(20,3 , 4);
        dNOd(20,10, 6);
        dNOd(20,20,24);
        dNOd(20,45, 2);
        dNOd(20,51, 3);
        dNOd(20,58, 1);
        dNOd(20,62, 7);
        dNOd(20,71, 8);
        dNOd(21,3 , 4);
        dNOd(21,10, 6);
        dNOd(21,20,24);
        dNOd(21,45, 2);
        dNOd(21,51, 3);
        dNOd(21,58, 1);
        dNOd(21,62, 7);
        dNOd(21,71, 8);
        /*Execution status--must be last!*/
        dNOd(22,19,24);
        NOdmax=--i;
        *p=0;
/*Payment*/
pa:     p=PAout;
        i=0;
        memset(p,' ',PAoutSZ);
        /*Fixed fields...*/
        dPAd( 4,52, 2); /* 0:D_ID*/
        dPAd( 9,11, 4); /* 1:C_ID*/
        dPAd( 9,54, 2); /* 2:C_D_ID*/
        dPAd( 9,33, 4); /* 3:C_W_ID*/
        dPAd(15,23, 8); /* 4:H_AMOUNT*/
        dPAd( 2, 7,19); /* 5:DATE*/
        dPAd( 5, 1,20); /* 6:W_STREET_1*/
        dPAd( 6, 1,20); /* 7:W_STREET_2*/
        dPAd( 7, 1,20); /* 8:W_CITY*/
        dPAd( 7,22, 2); /* 9:W_STATE*/
        dPAd( 7,25,10); /*10:W_ZIP*/
        dPAd( 5,42,20); /*11:D_STREET_1*/
        dPAd( 6,42,20); /*12:D_STREET_2*/
        dPAd( 7,42,20); /*13:D_CITY*/
        dPAd( 7,63, 2); /*14:D_STATE*/
        dPAd( 7,66,10); /*15:D_ZIP*/
        dPAd(10, 9,16); /*16:C_FIRST*/
        dPAd(10,26, 2); /*17:C_MIDDLE*/
        dPAd(10,29,16); /*18:C_LAST*/
        dPAd(11, 9,20); /*19:C_STREET_1*/
        dPAd(12, 9,20); /*20:C_STREET_2*/
        dPAd(13, 9,20); /*21:C_CITY*/
        dPAd(13,30, 2); /*22:C_STATE*/
        dPAd(13,33,10); /*23:C_ZIP*/
        dPAd(13,58,19); /*24:C_PHONE*/
        dPAd(10,58,10); /*25:C_SINCE*/
        dPAd(11,58, 2); /*26:C_CREDIT*/
        dPAd(16,17,14); /*27:C_CREDIT_LIM*/
        dPAd(12,58, 5); /*28:C_DISCOUNT*/
        dPAd(15,55,15); /*29:C_BALANCE*/
        dPAd(18,12,50); /*30:DATA_1*/
        dPAd(19,12,50);
        dPAd(20,12,50);
        dPAd(21,12,50);
        PAdmax=--i;
        *p=0;
/*Order Status*/
os:     p=OSout;
        i=0;
        memset(p,' ',OSoutSZ);
        dOSd( 3,11, 4); /* 0:C_ID*/
        dOSd( 3,24,16); /* 1:C_FIRST*/
        dOSd( 3,41, 2); /* 2:C_MIDDLE*/
        dOSd( 3,44,16); /* 3:C_LAST*/
        dOSd( 4,15,15); /* 4:C_BALANCE*/
        dOSd( 6,15, 8); /* 5:O_ID*/
        dOSd( 6,38,19); /* 6:O_ENTRY_D*/
        dOSd( 6,76, 2); /* 7:O_CARRIER_ID*/
/*Order Lines...*/
        dOSd( 8,3 , 4); /* 8:OL_SUPPLY_W_ID_1*/
        dOSd( 8,14, 6); /* 9:OL_I_ID_1*/
        dOSd( 8,25, 2); /*10:OL_QUANTITY_1*/
        dOSd( 8,32, 9); /*11:OL_AMOUNT_1*/
        dOSd( 8,47,10); /*12:OL_DELIVERY_D_1*/
        dOSd( 9,3 , 4);
        dOSd( 9,14, 6);
        dOSd( 9,25, 2);
        dOSd( 9,32, 9);
        dOSd( 9,47,10);
        dOSd(10,3 , 4);
        dOSd(10,14, 6);
        dOSd(10,25, 2);
        dOSd(10,32, 9);
        dOSd(10,47,10);
        dOSd(11,3 , 4);
        dOSd(11,14, 6);
        dOSd(11,25, 2);
        dOSd(11,32, 9);
        dOSd(11,47,10);
        dOSd(12,3 , 4);
        dOSd(12,14, 6);
        dOSd(12,25, 2);
        dOSd(12,32, 9);
        dOSd(12,47,10);
        dOSd(13,3 , 4);
        dOSd(13,14, 6);
        dOSd(13,25, 2);
        dOSd(13,32, 9);
```

```
        dOSd(13,47,10);
        dOSd(14,3 , 4);
        dOSd(14,14, 6);
        dOSd(14,25, 2);
        dOSd(14,32, 9);
        dOSd(14,47,10);
        dOSd(15,3 , 4);
        dOSd(15,14, 6);
        dOSd(15,25, 2);
        dOSd(15,32, 9);
        dOSd(15,47,10);
        dOSd(16,3 , 4);
        dOSd(16,14, 6);
        dOSd(16,25, 2);
        dOSd(16,32, 9);
        dOSd(16,47,10);
        dOSd(17,3 , 4);
        dOSd(17,14, 6);
        dOSd(17,25, 2);
        dOSd(17,32, 9);
        dOSd(17,47,10);
        dOSd(18,3 , 4);
        dOSd(18,14, 6);
        dOSd(18,25, 2);
        dOSd(18,32, 9);
        dOSd(18,47,10);
        dOSd(19,3 , 4);
        dOSd(19,14, 6);
        dOSd(19,25, 2);
        dOSd(19,32, 9);
        dOSd(19,47,10);
        dOSd(20,3 , 4);
        dOSd(20,14, 6);
        dOSd(20,25, 2);
        dOSd(20,32, 9);
        dOSd(20,47,10);
        dOSd(21,3 , 4);
        dOSd(21,14, 6);
        dOSd(21,25, 2);
        dOSd(21,32, 9);
        dOSd(21,47,10);
        dOSd(22,3 , 4);
        dOSd(22,14, 6);
        dOSd(22,25, 2);
        dOSd(22,32, 9);
        dOSd(22,47,10);
        dOSd(22,47,1);
        OSdmax=--i;
        *p=0;
/*Interactive Delivery*/
id:     p=IDout;
        i=0;
        memset(p,' ',IDoutSZ);
        dIDd( 4,17, 2); /* 0:O_CARRIER_ID*/
        dIDd( 6,19,24); /* 1:EXEC_STATUS*/
        IDdmax=--i;
        *p=0;
/*Stock Level*/
sl:     p=SLout;
        SLdmax;
        i=0;
        memset(p,' ',SLoutSZ);
        dSLd( 2,29,2); /* 0:D_ID*/
        dSLd( 6,12,3); /* 1:LOW_STOCK*/
        *p=0;
#ifdef TEST
        printf("NO outbuf len=%d, max index=%d\n", strlen(NOout),NOdmax);
        printf("PA outbuf len=%d, max index=%d\n", strlen(PAout),PAdmax);
        printf("OS outbuf len=%d, max index=%d\n", strlen(OSout),OSdmax);
        printf("ID outbuf len=%d, max index=%d\n", strlen(IDout),IDdmax);
        printf("SL outbuf len=%d, max index=%d\n", strlen(SLout),SLdmax);
#endif
}

extern
int init_term(ifd, ofd)
int ifd;
int ofd;
{
/*Save old config and set up for raw char by char */
        char c;

        /*Input...*/
        ioctl(ifd,TCGETA,&tty);
        memcpy(&its,&tty,sizeof(struct termio));
        tty.c_lflag=0;
        tty.c_iflag=0;
        tty.c_cc[VMIN]=2;
         tty.c_cc[VTIME]=2;
         ioctl(ifd,TCSETA,&tty);
         ioctl(ifd,TCFLSH,2);

        /*Output...*/
        ioctl(ofd,TCGETA,&tty);
        memcpy(&ots,&tty,sizeof(struct termio));
        tty.c_oflag=0;
         ioctl(ofd,TCSETA,&tty);
         ioctl(ofd,TCFLSH,2);
        return 0;
}

extern
void restore_term(ifd,ofd)
int ifd;
int ofd;
{
        ioctl(ifd, TCFLSH, 2 );
        ioctl(ofd, TCFLSH, 2);
        ioctl(ifd, TCSETAW, &its);
        ioctl(ofd, TCSETAW, &ots);
}

/*Place cursor pos escapes in arg buffer*/
scurpos (r,c,s)
        int r,c;
        char *s;
{
        /* Rows, cols start from 1 to match spec */
        /* sprintf(s,"•&a%dy%dC",r-1,c-1);*/
/*
        r--,c--;
        pos_str[2]=r+' ';
```

```
        pos_str[3]=c+' ';
        memcpy(s,pos_str,sizeof(pos_str));
*/
        sprintf(&pos_str[2],"%d;%dH",r,c);/* starts from 1,1 */
        memcpy(s,pos_str,sizeof(pos_str));
}

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

correct_echo(r,c,out)
        int r,c;
        char out;
{
        char outbuf[20]; /* space for 2 cursor postions and 1 char echo */

        sprintf(&pos_str[2], "%d;%dH", r,c); /* cursor postion */
        if (out==' ' || out==NC) out='_';
        strcpy(outbuf, pos_str);
        outbuf[(strlen(pos_str))]=BELL;
        outbuf[(strlen(pos_str)+1)]=out;
        outbuf[(strlen(pos_str)+2)]=pos_str[0];
        outbuf[(strlen(pos_str)+3)]=pos_str[1];
        sprintf(&outbuf[(strlen(pos_str)+4)], "%d;%dH", r,c); /* cursor
pos */
         if((write(ofd,&outbuf,strlen(outbuf)))!=strlen(outbuf))
fprintf(stderr, "Error: correct echo\n"), exit(-1);
}
/*------------*/

/*Do cursor pos*/
curpos (r,c)
        int r,c;
{
        /* Rows, cols start from 1 to match spec */
#ifdef TEST
        printf ("Pos to %d,%d\n",r,c);
#endif
/*
        cr=r; cc=c;
        r--,c--;
        pos_str[2]=r+' ';
         pos_str[3]=c+' ';
        write(ofd, pos_str, CPLEN);
*/
        sprintf(&pos_str[2], "%d;%dH", r,c);
        write(ofd, pos_str, CPLEN);
}

extern
void send_menu()
{
        if ((res=write(ofd,MEbuf,MEbufsz))!=MEbufsz) ERROR ("term write");
}

/*Get a New Order form*/
#define REPEAT_FLD_1 2
get_NO_input(p)
        struct NEWO_DATA *p;
{
        extern short int w_id;
        int fc;
        int fld_value;
        int ol_start, non_zero;
        int o_all_local = TRUE;
        int ol_cnt      = 0;
        if ((res=write (ofd,NObuf,NObufsz))!=NObufsz) ERROR ("term
write");
        fc=get_screen(NO_screen,47);
#ifdef ANSI_DEBUG
        dump_input(NO_screen,47);
#endif
#ifdef TEST
        printf ("%d NO fields read...\n", fc);
#endif
        if (fc==CANCEL) return(fc);

        /*Non-repeating group: d_id, c_id */
        p->d_id = (short int)atoi(NO_screen[0].val);
        p->c_id = (short int)atoi(NO_screen[1].val);

        /*Repeating Group:ol_supply_w_id,ol_i_id,ol_quantity*/
        for (ol_start = REPEAT_FLD_1; ol_start<fc; ol_start+=3) {
           /* 1 trip for each order line */
            non_zero = 0;
           fld_value = atoi(NO_screen[ol_start].val);
           if (fld_value != 0) {
               non_zero++;
               p->ol_table[ol_cnt].ol_supply_w_id = (short int)fld_value;
               if (fld_value != w_id) {
                       o_all_local = FALSE;
               }
           }

           fld_value = atoi(NO_screen[ol_start+1].val);
           if (fld_value != 0) {
               non_zero++;
               p->ol_table[ol_cnt].ol_i_id = fld_value;
           }
           fld_value = atoi(NO_screen[ol_start+2].val);
           if (fld_value != 0) {
               non_zero++;
               if (non_zero==3) p->ol_table[ol_cnt++].ol_quantity =
                               (short int)fld_value;
            }
        /*
         * Outa here: skipped supply_w_id or ol_i_id or ol_quantity
         */
            if (!((non_zero==3) || (non_zero==0)))
              return WRONG_INPUT;
        }
        p->o_ol_cnt    = ol_cnt;
        p->o_all_local = o_all_local;
        return(GOOD_INPUT);
}


/*Get a Payment form*/
get_PA_input(p)
        struct PMT_DATA *p;
{
        int fc;
```

```
        if ((res=write (ofd,PAbuf,PAbufsz))!=PAbufsz) ERROR ("term
write");
        fc=get_screen(PA_screen,6);
#ifdef DEBUG
        dump_input(PA_screen,6);
#endif
#ifdef TEST
        printf ("%d PA fields read...\n", fc);
#endif
        if (fc==CANCEL) return(fc);
        p->d_id = atoi(PA_screen[0].val);
        /*
        * Pay by c_id if the field "c_id" is not blank, otherwise,
        * pay by c_last.
        */
        if (PA_screen[1].val[0]!=NC) {
                /* c_id has the precedence */
                p->c_id   = atoi(PA_screen[1].val);
                p->byname = FALSE;
        } else {
                strcpy (p->c_last, PA_screen[4].val);
                p->byname= TRUE;
        }
        p->c_w_id   = atoi(PA_screen[2].val);
        p->c_d_id   = atoi(PA_screen[3].val);
        p->h_amount = atof(PA_screen[5].val);
        return (GOOD_INPUT);
}

/*Get an Order Status form*/
get_OS_input(p)
        struct ORDS_DATA *p;
{
        int fc;
        if ((res=write (ofd,OSbuf,OSbufsz))!=OSbufsz) ERROR ("term
write");
        fc=get_screen(OS_screen,3);
#ifdef DEBUG
        dump_input(OS_screen,3);
#endif
        if (fc==CANCEL) return(fc);
#ifdef TEST
        printf ("%d OS fields read...\n", fc);
#endif
        p->d_id=atoi(OS_screen[0].val);
        if (OS_screen[1].val[0]!=NC) {
                /* c_id has the precedence */
                p->c_id = atoi(OS_screen[1].val);
                p->byname = FALSE;
        } else {
                strcpy (p->c_last,OS_screen[2].val);
                p->byname = TRUE;
        }
        return GOOD_INPUT;
}

/*Get an Interactive Delivery form*/
get_ID_input(p)
        struct DVRY_DATA *p;
{
        int fc,carrier_id;
        if ((res=write (ofd,IDbuf,IDbufsz))!=IDbufsz) ERROR ("term
write");
        fc=get_screen(ID_screen,1);
#ifdef DEBUG
        dump_input(ID_screen,1);
#endif
        if (fc==CANCEL) return(fc);
#ifdef TEST
        printf ("%d ID fields read...\n", fc);
#endif
        carrier_id=atoi(ID_screen[0].val);
        if (!((carrier_id>=1) && (carrier_id<=10)))
        return(WRONG_INPUT);

        p->carrier_id  = carrier_id;
        p->start_queue = get_time();
        return(GOOD_INPUT);
}

/*Get a Stock Level form*/
get_SL_input(p)
        struct STKL_DATA *p;
{
        int fc, threshold;
        if ((res=write (ofd,SLbuf,SLbufsz))!=SLbufsz) ERROR ("term
write");
        fc=get_screen(SL_screen,1);
#ifdef DEBUG
        dump_input(SL_screen,1);
#endif
        if (fc==CANCEL) return(fc);
#ifdef TEST
        printf ("%d SL fields read...\n", fc);
#endif
        threshold = atoi(SL_screen[0].val);
        if (!((threshold>=10) && (threshold<=20))) return WRONG_INPUT;
        p->threshold=threshold;
        return GOOD_INPUT;
}

/*Terminal handling*/
/*Get a char*/
int gc()
{
        int rc;
        int cc;
        static int c2=0;
        int i;
/*
        static char in[2];
*/
        static char in[MAXSEQLEN]; /* Thomas */

get:    if (c2) {               /*Char in buffer?*/
             rc=in[1];
             for(i=1; i< c2; i++)
                in[i] = in[i+1];
             c2--;
        } else {
             if((cc=read(ifd,&in,MAXSEQLEN))<1) fprintf(stderr, "Error:
get char\n"), exit(-1);
```

```
            if ( in[0] == (char)0x03 ) exit(1); /* exit if interrupt */
              for(i =0; i< MAXSEQLEN; i++)
                      in[i]&=127;
            rc=(in[0]&=127); in[1]&=127;
            if(cc>=2)
              if ((rc=='•') && (in[1] =='['))  /*handle n char escape
sequences*/
                switch (in[2]) {
                   case 'I' : rc=(P); break;
                   default  : pca(BELL); goto get;
                 }
              else c2=cc-1;  /*Just 2 plain old chars close together*/
       }
       /*Single char--Handle 1 char specials here */
       switch (rc) {
              case 9  : rc=N;  break;
              case 10 :
              case 13 : rc=E;  break;
              case 3  : rc=M;  break;
              case CCR: rc=R;  break;
              case CCL: rc=L;  break;
              case CCU: rc=U;  break;
              case CCD: rc=D;  break;
              case CCH: rc=M;  break;
              case RUB:
              case '_': rc=C; break;
       }
#ifdef TEST
       if (rc>=0) printf("Returning char: %d %c\n", rc, rc);
       else printf ("Returning cursor pos: %d\n", rc);
#endif
       return(rc);
}

/*Clear a field from current to end*/
cteof(f,cp,mp)
       struct field *f;
       int cp, mp;
{
       int xp=cp, rv=DP_MASK;
       while (xp<=mp) {
              if (f->val[xp]=='.') rv=0;
              f->val[xp++]=NC; pc(' ');
       }
       while (--xp>=cp) /* abi */ CL;
       return(rv);
}

/*Put a char*/
pc(out)
       char out;
{
       if (out==' ' || out==NC ) { /* <- abi add brackets */ out='_';
        if((write(ofd,&out,1))!=1) fprintf(stderr, "Error: put char\n"),
exit(-1);
       }
}

/*Put a char abi-for insert mode*/
pca(out)
       char out;
{
       if (out==' ' || out==NC) out='_';
        if((write(ofd,&out,1))!=1) fprintf(stderr, "Error: put char\n"),
exit(-1);
}

/*Ring bell*/
bell()
{
       pca(BELL); /* abi change pc to pca */
}

/*Clear screen*/
clr()
{
       scr_clr;
}

/************************Screen Input
Routines*******************************/

/*Input a field*/

gf(f)
       struct field *f;
{
       char *b=f->val;
       byte n=f->sz,t=f->ty,pf=f->fl&DP_MASK;
       int  in;
       byte mp,cp,ck,xp,rv;


       mp=n-1; if(pf) mp+=3; cp=0;
       while(1) {
#ifdef TEST
          printf ("At top of gf loop...\n");
#endif
          if ((in=gc())<=0) {
              switch (in) {
                     case R:    if (cp>=mp) {  /*Cursor right*/
                                   rv=R; goto ret;
                                } else {
                                   if (b[cp]!=NC) {
                                      cp++;
                                      /* CR; abi */
                                   } else goto err;
                                }
                                break;
                     case L:    if (cp<=0) {  /*Cursor left*/
                                   rv=L; goto ret;
                                } else {
                                   cp--;
                                   /* CL abi */ ;
                                }
                                break;
                     case C:    if (b[cp]==NC) goto err;  /*Delete
char*/
                                if (b[cp]=='.') {
                                    cteof(f,cp,mp);
                                    pf=0;
                                    mp=n-1;
```

```
                                        break;
                                    }
                                    xp=cp;
                                    while (xp<=mp) {
                                        if (b[xp]=='.') pf=0;
                                        b[xp]=b[xp+1];
                                        pca(b[xp]);
                                        xp++;
                                        if (b[xp]==NC) break;
                                    }
                                    write (ofd,cls,xp-cp);
                                    break;
                            default:   rv=in; goto ret;
                        }
                    } else { /*Nuttin' special: process as input*/
                        if (t==F_INT && (in<'0' || in>'9')) goto err;
                           else if (t==F_STR &&
                                   (in<'A' || in>'z' || (in>'Z' && in<'a'))) goto
err;
                            else if (t==F_MON && (in!='.' &&
                                     (in <'0' || in>'9'))) goto err;
                        /*It's a legal char, so mebbe put it in buffer*/
                        if (b[cp]=='.') { cteof(f,cp,mp); pf=0;  mp=n-1; }
                         if (cp==mp && (!pf && t==F_MON)) in='.';
                        if (in=='.')
                                if (pf) goto err;
                                else { mp=cp+2; pf=DP_MASK; }
                        if (b[cp]==NC) { /*Concatenate mode*/
                                b[cp]=in; pc(in);
                                /*If at eof and room for more bump ptr else
backspace*/
                                if (cp==mp) CL;
                                else if(cp<mp) cp++; else goto err;
                        } else { /*Insert mode*/
                                if (b[mp]!=NC) goto err; /*Overflow*/
                                xp=mp;
                                while (xp>cp) {b[xp]=b[xp-1]; xp--;}
                                 b[xp]=in;
                                /* abi add to avoid double echo */ xp++;
                                do pca(b[xp++]); while (xp<=mp && b[xp]!=NC);
                                  write (ofd,cls,xp-cp);
                                /* chris */
                                if (cp<mp) {cp++; CR;}
                        } /*else*/
                    } /*else*/
                     continue;
err:                  /* pca(BELL);  abi change pc to pca */ /*abi add */
                    /*  fprintf( errf, "in err got 0x%x \n",in);
                     fflush (errf); */
                     correct_echo (f->r,((f->c)+cp),b[cp]);
            }
ret:        if (pf) f->fl|=DP_MASK; else f->fl&=(~DP_MASK);
            return(rv);
}

/*Input a screen according to field descriptor array*/
get_screen(s,n)
            struct field s[];
            int n;
{
            byte ln, rc, rf, cf=0, nf, cc, fc=0;

            for (cc=0;cc<n;cc++) {
                    memset(s[cc].val,NC,MAXVSZ); s[cc].fl&=(~DP_MASK);
            }
            while (1) {
#ifdef TEST
                    printf ("In get screen--current field is %d\n",cf);
#endif
                    curpos(s[cf].r,s[cf].c);
                    rc=(char)gf(&s[cf]);
                     ln=s[cf].ln;
                    if (s[cf].ln!=cf && s[ln].val[0]!=NC && s[cf].val[0]!=NC)
                        cf=ln;
                    else {
                       switch (rc) {
                            case (char)E: rf=fc=0;
                                    for(cc=0;cc<n && !rf;cc++) {
                                       if (s[cc].val[0]!=NC) fc++;
                                       else rf=(s[cc].fl & REQ_MASK) &&
                                              s[s[cc].ln].val[0]==NC;
                                    }
                                    if (!rf) return(fc);
                                    else {
                                        nf=--cc;
                                        break;
                                    }
                            case (char)M: return(CANCEL);
                            case (char)H: nf=0; break;
                            case (char)U: nf=s[cf].nu; break;
                            case (char)D: nf=s[cf].nd; break;
                            case (char)N:
                            case (char)R: nf=s[cf].nr; break;
                            case (char)L:
                            case (char)P: nf=s[cf].nl; break;
                                    break;
                       }
                    cf=nf;
                    }
            }
}

char badmsg[512];
/* Display routines...*/
/* display the new order data on the screen */
void display_new_order(p)
struct NEWO_DATA *p;
{
    register int   i,k,f,s;
    char new_date[DATETIME_LEN+1];
/*2 cases: good, bad*/
    if (p->items_valid) {
       /*the good...*/
       /*Fixed fields*/
       /* convert date to TPC-C way */
       ifmt (NOndx[0],p->d_id,2);
       ifmt (NOndx[1],p->c_id,4);
       convert_datetime(p->new_date, new_date, YEAR_TO_SECOND);
        sfmt (NOndx[2],p->new_date,19);
       sfmt (NOndx[3],p->c_last,16);
       sfmt (NOndx[4],p->c_credit,2);
       ffmt (NOndx[5],p->c_discount,5,0);
       ffmt (NOndx[6],p->w_tax,5,0);
```

```
        ffmt (NOndx[7],p->d_tax,5,0);
        ifmt (NOndx[8],p->o_ol_cnt,2);
         ifmt (NOndx[9],p->o_id,8);
         ffmt (NOndx[10],p->total,9,1);
        /*Repeating fields: 1 per order line*/
         for(i=0,f=11; i<p->o_ol_cnt; i++) {
              ifmt (NOndx[f++],p->ol_table[i].ol_supply_w_id,4);
              ifmt (NOndx[f++],p->ol_table[i].ol_i_id,6);
               sfmt (NOndx[f++],p->ol_table[i].name_i,16);
              ifmt (NOndx[f++],p->ol_table[i].ol_quantity,2);
              ifmt (NOndx[f++],p->ol_table[i].s_quantity,3);
               *NOndx[f++]=p->ol_table[i].brand_generic;
               ffmt (NOndx[f++],p->ol_table[i].price,7,1);
               ffmt (NOndx[f++],p->ol_table[i].ol_amount,8,1);
         }
        *NOndx[f]=NC;
        if ( (res=write (ofd,NOout,(s=NOndx[f]-NOout))) != s )
             ERROR ("term write");
   } else {
        /*...the bad...*/
        badmsg[0] = 0;
        ifmt (NOndx[0],p->d_id,2);
        ifmt (NOndx[1],p->c_id,4);
        strncat(badmsg,NOndx[0]-ID_CPLEN,ID_CPLEN+ID_CPLEN+2+4);
        sfmt (NOndx[3],p->c_last,16);
        sfmt (NOndx[4],p->c_credit,2);
        strncat(badmsg,NOndx[3]-ID_CPLEN,ID_CPLEN+ID_CPLEN+16+2);
         ifmt (NOndx[9],p->o_id,8);
        strncat(badmsg, NOndx[9]-ID_CPLEN,ID_CPLEN+8);
         sfmt (NOndx[NOdmax],"Item number is not valid",24);
        strncat(badmsg,NOndx[NOdmax]-ITEM_CPLEN,ITEM_CPLEN+24);
        i = strlen(badmsg);
        if (write(ofd,badmsg,i)!=i) ERROR ("term write");

   }
}

/* display the payment data on the screen */
void display_payment(p)
struct PMT_DATA *p;
{
        char date_val[DATETIME_LEN+1];
         int i,s;
        char *xp;

        convert_datetime(p->pay_date, date_val, YEAR_TO_SECOND);
        ifmt (PAndx[ 0],p->d_id,2);
        ifmt (PAndx[ 1],p->c_id,4);
        ifmt (PAndx[ 2],p->c_d_id,2);
        ifmt (PAndx[ 3],p->c_w_id,4);
        ffmt (PAndx[ 4],p->h_amount,8,1);
        convert_datetime(p->pay_date, date_val, YEAR_TO_SECOND);
        sfmt (PAndx[ 5],date_val,19);
        sfmt (PAndx[ 6],p->w_street_1,20);
        sfmt (PAndx[ 7],p->w_street_2,20);
        sfmt (PAndx[ 8],p->w_city,20);
        sfmt (PAndx[ 9],p->w_state,2);
        zipf (PAndx[10],p->w_zip);
        sfmt (PAndx[11],p->d_street_1,20);
        sfmt (PAndx[12],p->d_street_2,20);
        sfmt (PAndx[13],p->d_city,20);
        sfmt (PAndx[14],p->d_state,2);
        zipf (PAndx[15],p->d_zip);
        sfmt (PAndx[16],p->c_first,16);
        sfmt (PAndx[17],p->c_middle,2);
        sfmt (PAndx[18],p->c_last,16);
        sfmt (PAndx[19],p->c_street_1,20);
        sfmt (PAndx[20],p->c_street_2,20);
        sfmt (PAndx[21],p->c_city,20);
        sfmt (PAndx[22],p->c_state,2);
        zipf (PAndx[23],p->c_zip);
        phonef (PAndx[24],p->c_phone);
        convert_datetime(p->c_date, date_val, YEAR_TO_DATE);
        sfmt (PAndx[25],date_val,10);
        sfmt (PAndx[26],p->c_credit,2);
        ffmt (PAndx[27],p->c_credit_lim,14,1);
        ffmt (PAndx[28],p->c_discount,5,0);
        ffmt (PAndx[29],p->c_balance,15,1);
        *PAndx[30]=NC;
        if (p->c_credit[0] == 'B' && p->c_credit[1] == 'C') {
        sfmt (PAndx[30],p->c_data,50);
        sfmt (PAndx[31],&p->c_data[50],50);
        sfmt (PAndx[32],&p->c_data[100],50);
        sfmt (PAndx[33],&p->c_data[150],50);
        *(PAndx[33]+51)=NC;
        }
        if ( (res=write (ofd,PAout,(s=strlen(PAout)))) != s )
             ERROR ("term write");
}

void display_order_status(p)
struct ORDS_DATA *p;
{
        char           date_val[DATETIME_LEN+1];
        register int   i,f,s;

        if (*p->cur_date == NC) date_val[0] = NC;
         else convert_datetime(p->cur_date, date_val, YEAR_TO_SECOND);

        ifmt (OSndx[ 0],p->c_id,4);
        sfmt (OSndx[ 1],p->c_first,16);
        sfmt (OSndx[ 2],p->c_middle,2);
        sfmt (OSndx[ 3],p->c_last,16);
        ffmt (OSndx[ 4],p->c_balance,10,1); /* abi copied from Liberty*/
        ifmt (OSndx[ 5],p->o_id,8);
        sfmt (OSndx[ 6],date_val,16);
        if (p->o_carrier_id != -1) ifmt(OSndx[ 7],p->o_carrier_id,2);
        for (i = 0,f=8; i < p->o_ol_cnt; i++) {
            if (*p->ol_table[i].delivery_date == NC) date_val[0] = NC;
            else convert_datetime(p->ol_table[i].delivery_date,
                                  date_val, YEAR_TO_DATE);
            ifmt (OSndx[f++],p->ol_table[i].ol_supply_w_id,4);
            ifmt (OSndx[f++],p->ol_table[i].ol_i_id,6);
            ifmt (OSndx[f++],p->ol_table[i].ol_quantity,2);
            ffmt (OSndx[f++],p->ol_table[i].ol_amount,9,1);
            sfmt (OSndx[f++],date_val,10);
        }
        *OSndx[f]=NC;
        if ( (res=write (ofd,OSout,(s=strlen(OSout)))) != s )
             ERROR ("term write");
}
```

```
void display_stock_level(p)
struct STKL_DATA *p;
{
        int s;
        ifmt (SLndx[ 0],p->stkl_d_id,2);
         sprintf(SLndx[1], "%3d", p->stock_count);
        if ( (res=write (ofd,SLout,(s=strlen(SLout)))) != s )
             ERROR ("term write");
}

void display_delivery(p)
struct DVRY_DATA *p;
{
        int s;
        ifmt(IDndx[0],p->carrier_id,2);
        strcpy(IDndx[1],"Delivery has been queued");
        if ( (res=write (ofd,IDout,(s=strlen(IDout)))) != s )
             ERROR ("term write");
}

void
display_msg(s)
        char *s;
{
    curpos(23,1);
    write(ofd,"\33T",2);
    write(ofd,s,strlen(s));
}

sfmt(d,s,c)
        char *d,*s;
        int c;
{
        while (c && *s!=NC) {c--; *d++=*s++;}
        while (c-->0) *d++=' ';
}

ifmt(d,n,c) /*Right justifed...*/
        char *d; /*Buffer*/
        int n,c; /*Number to be formatted, field width*/
{
         unsigned short int sign;
         char *p=d+c;

         if(sign=(n<0)) n=-n;

         do  DIGIT(n) while(n>0);

        if (sign) *--p='-';
        while (p>d) *--p=' ';
        if (p!=d) { fprintf (stderr, "iformat botch! %d\n", p-d);
               fprintf(stderr, "ifmt(%x, %u, %u) p=%x\n",d,n,c,p);
               exit(-1);
        }
}

ffmt(d,n,c,f) /*Right justified, 2 decimal places...*/
        char *d;
        int c,f;
        double n;
{
        unsigned int hi9,lo9;
         unsigned short int sign, i;
         char *p=d+c;
        double x100;
#ifdef TEST
        printf ("Trying to format %8.2lf \n",n);
#endif
         x100=n*100.0; /* Scale for 2 decimal digits */
         if(sign=(x100<0.0)) x100=-x100;
         lo9=(int)(fmod(x100+0.5,billion)); /*18 digits will do the trick*/
         hi9=(int) (floor(x100)/billion);
         DIGIT(lo9)
         DIGIT(lo9)
         *--p='.';
         if (hi9) for (i=1;i<=7;i++) DIGIT(lo9)
         else hi9=lo9;
         do DIGIT(hi9) while (hi9>0);
         if(sign) *--p='-';
        if(f) *--p='$';
         while (p>d) *--p=' ';
        if (p!=d) { fprintf (stderr, "fformat botch! %d\n", p-d);
                fprintf(stderr,"ffmt(%x, %8.2f, %u, %u) p=%s\n",d,n,c,f,p);
                exit(-1); }
}

zipf(d,s)
        char *d,*s;
{
        sfmt(d,s,5);
        *(d+5)='-';
        sfmt(d+6,s+5,4);
}

phonef(d,s)
        char *d,*s;
{
        sfmt(d,s,6);
        *(d+6)='-';
        sfmt(d+7,s+6,3);
        *(d+10)='-';
        sfmt(d+11,s+9,3);
        *(d+14)='-';
        sfmt(d+15,s+12,4);
}

/* read a transaction from the screen */
extern
int get_transaction()
{
    int com;

#ifdef NOTDEF
    curpos (23,1);
    curpos (24,74);
         sprintf(dummy, "|");
         write (ofd,  (void *)dummy, strlen(dummy));
#else
        write(ofd, get_tr_str, GET_TR_STRLEN);
#endif
    com=gc();
    clr();
```

```
    return com;
}
/*Get user id*/
extern
int get_user()
{
        clr();
        curpos(24,1);
        write (ofd,"User ID: ",9);
        get_screen(UI_screen,1);
        return(atoi(UI_screen[0].val));
}

/*Display status*/

extern
void display_status(status)
char *status;
{
        curpos(23,1);
        write(ofd, status, strlen(status));
}

#ifdef ANSI_DEBUG
static void dump_input(s,n)
        struct field s[];
        int n;
{

int i;

        for(i=0; i<n;i++){
                sprintf(debug_msg,"field #%d,%s\n",s[i].val);
                append_debug_msg( log700, debug_msg);
        }
}
#endif
```

do_tpcc.c

```
/*=========================================================================
=====*
|    do_tpcc.c - The main program of "do_tpcc".
|
|
|
|       Usage: do_tpcc [User-Id]
|
|
|
|       If the arguments are not shown, the program will request for
User_Id.  |
|
|       If compiled with -DDEBUG flag, it writes debugging messages to the
log |
|        file - logtpcc.
|
|
|
|       "do_tpcc" does the followings:
|
|            - Requests for User ID if needed.
|
|            - Initiating terminal I/O.
|
|            - Setting up terminals which includes defining softkeys and
|
|               defining forms stored in forms cache.
|
|             - Keep reading from terminals and executing transactions until
|
|               the terminal sends a stop signal to terminate the process.
|             - Exit the program.
|
*=========================================================================
===*/
#include <stdio.h>
#include <math.h>
#include <time.h>
#include <fcntl.h>
#include <termio.h>
#include <sys/ipc.h>
#include <unistd.h>

#include "atmi.h"
#include "Uunix.h"
#include "tpcc_info.h"
#include "do_tpcc.h"
#include "term.h"
#include "trans_type.h"

/************************************************************************
 * message formats a message and outputs it to a standard location
 * (stderr for now)

************************************************************************/
#define message(str) userlog("%s",str)

void srand48();
static void exit_program();
static void post_active_client();
void display_new_order();
void display_payment();
void display_order_status();
void display_stock_level();
void display_delivery();
static long get_cltinit();
static int recv_chars();

#ifndef NULL_TRANS
static void do_transactions();
static int do_stock_level();
static int do_delivery();
static int do_order_status();
static int do_payment();
static int do_new_order();
#else
static void do_transactions(user_id);
static int do_stock_level(user_id);
static int do_delivery(user_id);
```

```
static int do_order_status(user_id);
static int do_payment(user_id);
static int do_new_order(user_id);
#endif

#ifdef NULL_TRANS /* Functions to fill data for Loopback -- Roopa */
void fill_newo();
void fill_pmt();
void fill_sl();
void fill_dl();
void fill_os();
#endif

int read_chars();
double atof();

/* The following are global to the application */
char          status[MAX_STR_LEN];
int           ifd = 0;   /* input file descriptor, set to be stdin */
int           ofd = 1;   /* output file descriptor, set to be stdout */
int          tfd = 0;
short int     w_id;
short int     stkl_d_id;
static long   result;
     /* FILE *errf;  abi */

static struct PMT_DATA  *pmt_dataptr;
static struct ORDS_DATA *ords_dataptr;
static struct DVRY_DATA *dvry_dataptr;
static struct STKL_DATA *stkl_dataptr;
static struct NEWO_DATA *newo_dataptr;
static int trans_stat,res;

#ifdef NULL_TRANS
long savemask;
#endif

#ifdef DEBUG
#define PMODE         0644
    int   logtpcc;
    int   log700;
    char  debug_msg[INSIZE+MAX_STR_LEN];
    char  debug_fname[80];
#endif

#define NEWO_TRANS       0
#define PAYMENT_TRANS    1
#define ORDER_STAT_TRANS 2
#define DELIVERY_TRANS   3
#define STOCK_TRANS      4
#define NUM_TRANS        5

static char *trans_name[NUM_TRANS] = {"NEWO_SVC", "PMT_SVC", "ORDS_SVC",
                                      "DVRY_SVC", "STKL_SVC"};

static void SQL_error(sql_code, isam_code)
long sql_code;
long isam_code;
{
    sprintf(status, "Error: SQL_CODE=%d, ISAM_ERR=%d, %s",
       sql_code, isam_code, RETRY_MSG);
}

static void TUXEDO_err(service)
int service;
{
    if (tperrno == TPESVCFAIL ) {
        sprintf(status, "Error: request_service: faild at %s tpcall",
                   trans_name[service]);
        message(status);
       exit_program(1);
    } else if (tperrno == TPEOS) {
        sprintf(status, "Error: request_service: %s TUXEDO OS error =
%d\n",
           trans_name[service], Uunixerr);
    } else {
        sprintf(status, "Error: request_service: %s tperrno = %d\n",
                   trans_name[service], tperrno);
    }
    message(status);
}

/*
 * post_active_client - Whoever finishes executing all the transactions
first
 *             will terminate the monitors running in the background and
 *              create a file "post_act_clt_done" to tell the others not
do the
 *              same thing.
 */
static
void post_active_client()
{
    /*
     * If "post_act_clt_done" is there, do nothing, otherwise, the user
     * must be the first one who finishes all the transactions. He has the
     * responsibility to terminate the monitor since the active window
stops
     */
#ifndef NULL_TRANS
/* Commented out since it doesn't do anything - Roopa */
/*
    if (open("post_act_clt_done", 2 ) == -1 )
       system("post_active_client");
*/
#endif
    return;
}

/*
 * tpurcode is declared in "UBB.h", it contains the application defined
value
 * which was sent as part of tpreturn()
 */

#ifdef NULL_TRANS
static void enroll_client( user_id )
int user_id;
{
    /* w_id and stkl_d_id are constants and are global to the application
*/
    w_id      = (user_id-1)/D_PER_W + 1;
```

```
    stkl_d_id = (user_id-1)%D_PER_W + 1;

    if ((newo_dataptr =
        (struct NEWO_DATA *) malloc(sizeof(struct NEWO_DATA)) ) == NULL )
{
                fprintf( stderr,
            "User-Id: %d failed at buffer allocation request for New-
Order.\n",
                 user_id );
        exit_program(1);
    }
    newo_dataptr->w_id = w_id;
    newo_dataptr->header.dtype = NEW_ORDER;

    if ((pmt_dataptr =
         (struct PMT_DATA *) malloc(sizeof(struct PMT_DATA)) ) == NULL ) {
                fprintf( stderr,
            "User-Id: %d failed at buffer allocation request for
payment.\n",
                 user_id );
        exit_program(1);
    }
    pmt_dataptr->w_id = w_id;
    pmt_dataptr->header.dtype = PAYMENT;

    if ((ords_dataptr =
         (struct ORDS_DATA *) malloc(sizeof(struct ORDS_DATA)) ) == NULL )
{
                fprintf( stderr,
          "User-Id: %d failed at buffer allocation request for Order-
Status.\n",
                 user_id );
        exit_program( 1 );
    }
    ords_dataptr->w_id = w_id;
    ords_dataptr->header.dtype = ORDER_STATUS;

    if ((dvry_dataptr =
         (struct DVRY_DATA *) malloc(sizeof(struct DVRY_DATA)) ) == NULL )
{
        fprintf(stderr,
            "User-Id: %d failed at buffer allocation request for
Delivery.\n",
                 user_id);
        exit_program(1);
    }
    /*
     * Get cltinit time-stamp for delivery transaction to use.
     */
    dvry_dataptr->cltinit  = get_cltinit();
    dvry_dataptr->w_id     = w_id;
    dvry_dataptr->header.dtype  = DELIVERY;

    if ((stkl_dataptr =
         (struct STKL_DATA *) malloc(sizeof(struct STKL_DATA)) ) == NULL )
{
                fprintf( stderr,
           "User-Id: %d failed at buffer allocation request for stock-
level.\n",
                 user_id );
        exit_program(1);
    }
    stkl_dataptr->w_id      = w_id;
    stkl_dataptr->stkl_d_id = stkl_d_id;
    stkl_dataptr->header.dtype = STOCK_LEVEL;
}

#else /* end of if NULL_TRANS */

static void enroll_client( user_id )
int user_id;
{
    /* w_id and stkl_d_id are constants and are global to the application
*/
    w_id      = (user_id-1)/D_PER_W + 1;
    stkl_d_id = (user_id-1)%D_PER_W + 1;

    if (tpinit( (TPINIT *)NULL ) == -1 ) {
        fprintf( stderr,"User-Id: %d failed to enroll client\n", user_id
);
        exit_program(1);
    }
    if ((newo_dataptr =
        (struct NEWO_DATA *) tpalloc("CARRAY", NULL,
                                     sizeof(struct NEWO_DATA)) ) == NULL )
{
        fprintf( stderr,
            "User-Id: %d failed at buffer allocation request for New-
Order.\n",
                 user_id );
        exit_program(1);
    }
    newo_dataptr->w_id = w_id;
    newo_dataptr->header.dtype = NEW_ORDER;

    if ((pmt_dataptr =
         (struct PMT_DATA *) tpalloc("CARRAY", NULL,
                                      sizeof(struct PMT_DATA)) ) == NULL ) {
        fprintf( stderr,
            "User-Id: %d failed at buffer allocation request for
payment.\n",
                 user_id );
        exit_program(1);
    }
    pmt_dataptr->w_id = w_id;
    pmt_dataptr->header.dtype = PAYMENT;

    if ((ords_dataptr =
         (struct ORDS_DATA *) tpalloc("CARRAY", NULL,
                                     sizeof(struct ORDS_DATA)) ) == NULL )
{
        fprintf( stderr,
          "User-Id: %d failed at buffer allocation request for Order-
Status.\n",
                 user_id );
        exit_program( 1 );
    }
    ords_dataptr->w_id = w_id;
    ords_dataptr->header.dtype = ORDER_STATUS;

    if ((dvry_dataptr =
         (struct DVRY_DATA *) tpalloc("CARRAY", NULL,
```

```
                                    sizeof(struct DVRY_DATA)) ) == NULL )
{
        fprintf(stderr,
            "User-Id: %d failed at buffer allocation request for
Delivery.\n",
                user_id);
        exit_program(1);
    }
    /*
     * Get cltinit time-stamp for delivery transaction to use.
     */
    dvry_dataptr->cltinit  = get_cltinit();
    dvry_dataptr->w_id     = w_id;
    dvry_dataptr->header.dtype  = DELIVERY;

    if ((stkl_dataptr =
         (struct STKL_DATA *) tpalloc("CARRAY", NULL,
                                    sizeof(struct STKL_DATA)) ) == NULL )
{
        fprintf( stderr,
          "User-Id: %d failed at buffer allocation request for stock-
level.\n",
                user_id );
        exit_program(1);
    }
    stkl_dataptr->w_id      = w_id;
    stkl_dataptr->stkl_d_id = stkl_d_id;
    stkl_dataptr->header.dtype = STOCK_LEVEL;
}

#endif /* NULL_TRANS*/

/*************************************************************************
*****
 * get_cltinit - fetch the time stamp recorded in the file "CLT_INIT"
which
 *               indicates the time the client system started the session.

*************************************************************************
****/
static
long get_cltinit()
{
        char clt_init[MAX_STR_LEN];
        char buffer[64];
        long cltinit;
        int  clt_init_fd, bytes;

        /* "CLT_INIT" is the file name */
        strcpy( clt_init, "/home/audit/app/CLT_INIT" );
        if ((clt_init_fd = open(clt_init, O_RDONLY)) == -1) {
              fprintf(stderr, "can't open %s\n", clt_init );
              exit(1);
        };
        bytes = read(clt_init_fd, buffer, 64);
        if (bytes == -1) {
              fprintf(stderr, "problems reading file %s\n",clt_init);
              exit(1);
        }
        buffer[bytes] = '\0';
        cltinit = atoi(buffer);
        close(clt_init_fd);
        return cltinit;
}

/*
 * exit_program - restores original terminal attributes before leaving the
 *                program.
 */
static
void exit_program( err )
short int err;
{
    if ( err )
        fprintf( stderr, "exit_program: Error Code = %d\n", err );
    restore_term(ifd,ofd);
    exit( err );
}

/************************ Entry Point
***********************************/
main(argc, argv)
int argc;
char *argv[];
{
    int  user_id;

#ifdef DONT_USE_STDIO
    user_id=10;
    if (argc!=2) ERROR(args wrong);
    if ((tfd=open(argv[1],O_RDWR))<0) ERROR(term open);
    printf("termfd=%d\n",tfd);
    ifd=ofd=tfd;
    if (init_term(ifd,ofd)) exit_program(1); /* Set terminal attributes */

#else
    ifd=0; ofd=1;
    if (init_term(ifd,ofd)) exit_program(1); /* Set terminal attributes */

    if (argc == 2) {                  /*User-Id is provided.*/
        user_id = atoi( argv[1] );
    } else if (argc==1){
                user_id=get_user();
          } else {
                  fprintf( stderr, "Usage: %s [User-Id]\n", argv[0]);
                  exit_program(1);
            }
#endif

/* Originally, init_term was called here abi 12/11 Moved before get_user
call
and corresponsingly into ifdef DONT_USE_STDIO */
    /* if (init_term(ifd,ofd)) exit_program(1); */ /* Set terminal
attributes */

    /* enroll as a client and allocate buffer */
    enroll_client( user_id );

    /* Initialize random number generator */
    srand48(getpid()*time(0)*user_id);

    /* Set up input forms and display buffers */
```

```
    init_forms(w_id, stkl_d_id);

#ifdef DEBUG
/* create the debugging message file: logtpcc, log700 */
        sprintf( debug_fname, "logtpcc_%d", user_id );
        logtpcc = creat( debug_fname, PMODE );
        sprintf( debug_fname, "log700_%d", user_id );
        log700 = creat( debug_fname, PMODE );
#endif

    /*
     * The loop to accept input and execute transactions.
     */
#ifdef NULL_TRANS
    do_transactions(user_id);
#else
    do_transactions();
#endif

    /*
     * Terminate monitors and inform other do_tpcc not to repeat the job
     * again by creating a file named "post_act_clt_done".
     */
    post_active_client();

    if (tpterm() == -1) {
        fprintf(stderr, "User-Id: %d failed to leave application.\n",
                user_id );
        exit_program( 1 );

    }
    exit_program(0);
}
/*
 * do_transactions - the main loop to read input from RTE, then
 *                   executes transactions.
 */
static
void do_transactions(user_id)
{
     short int  com;

     /* Sending menu causes RTE to start sending transactions */
     clr();
     send_menu();
     while (1) {
        status[0] = '\0';
        com = get_transaction();
#ifdef DEBUG
        sprintf(debug_msg,"get_transaction return %c\n", com);
        append_debug_msg( log700, debug_msg);
#endif
        switch (com) {
                case '1': trans_stat=do_new_order(user_id);
                          break;
                case '2': trans_stat=do_payment(user_id);
                          break;
                case '3': trans_stat=do_order_status(user_id);
                          break;
                case '4': trans_stat=do_delivery(user_id);
                          break;
                case '5': trans_stat=do_stock_level(user_id);
                          break;
                case '6': return;

                default : trans_stat=WRONG_INPUT; send_menu();
        } /*switch*/
        if (trans_stat==WRONG_INPUT) sprintf (status,"Error: Bad input
data.");
        if (status[0] == '\0') sprintf (status,"Success!");
        display_status(status);
     } /*while*/
} /*do_transactions*/

static
int do_new_order(user_id)
{
        int rtval;
        int rtcode;

        if ((rtval = get_NO_input(newo_dataptr) != GOOD_INPUT)) {
             return(rtval);
        }
        newo_dataptr->header.returncode = 0;  /* reset the returncode */


#ifdef NULL_TRANS
/* If LOOPBACK client just fill some dummy data and send it -- Roopa */

        rtval = user_id % 20;
        if (rtval == 0) {
            nap(2500);
        } else {
             nap(rtval*100);
        }
        fill_newo();
        display_new_order(newo_dataptr);
#endif

#ifndef NULL_TRANS
#ifdef DEBUG
        sprintf(debug_msg,"new order c_id %d district %d\n",
                 newo_dataptr->c_id,newo_dataptr->d_id);
        append_debug_msg( logtpcc, debug_msg);
#endif

        if (tpcall("NEWO_SVC", (char *) newo_dataptr, sizeof(struct
NEWO_DATA),
                      (char **) &newo_dataptr, &result, 0 ) != -1) {

                rtcode = (int) newo_dataptr->header.returncode;
                if (tpurcode == COMMIT_NEWO || tpurcode == INVALID_NEWO) {
                    display_new_order(newo_dataptr);
                    if (tpurcode == COMMIT_NEWO && rtcode == COMMIT_NEWO) {
                        sprintf(status, "Success: new-order-committed");
                    }
                    else if (rtcode == INVALID_NEWO) {
                        sprintf(status, "Success: invalid item rolled
back");
                    }
                    else
```

```
                        sprintf(status, "NEW_ORDER: tpurcode= %d rtcode= %d
",
                                tpurcode, rtcode);
                } else if (tpurcode == SQL_ERROR) { /* SQL ERROR found */
                        /* return with non-NULL error message */
                    SQL_error(newo_dataptr->header.sql_code,
                                newo_dataptr->header.isam_code);
                } else
                    sprintf(status, "NEW_ORDER: tpurcode= %d ", tpurcode);

        } else {
                TUXEDO_err(NEWO_TRANS);
        }
#endif
        return NEW_ORDER;
}

static
int do_payment(user_id)
{
        int rtval;
        if((rtval = get_PA_input(pmt_dataptr)) != GOOD_INPUT) {
            return rtval;
         }


#ifdef NULL_TRANS

/* If LOOPBACK client just fill some dummy data and send it -- Roopa */

        rtval = user_id % 20;
         if (rtval == 0) {
             nap(2500);
         } else {
             nap(rtval*100);
         }
        fill_pmt();
        display_payment(pmt_dataptr);
#endif

#ifndef NULL_TRANS
#ifdef DEBUG
        sprintf(debug_msg,"payment c_id %d d_id %d\n",
        pmt_dataptr->c_id,pmt_dataptr->d_id);
        append_debug_msg( logtpcc, debug_msg);
#endif
        if (tpcall("PMT_SVC",(char *) pmt_dataptr, sizeof(struct
PMT_DATA),
                (char **) &pmt_dataptr, &result, 0 ) != -1) {
                if (tpurcode == 0) {
                    display_payment(pmt_dataptr);
                } else if ( tpurcode == SQL_ERROR ) {
                    /* return with non-NULL error message */
                    SQL_error(pmt_dataptr->header.sql_code,
                                pmt_dataptr->header.isam_code);
                }
        } else { TUXEDO_err(PAYMENT_TRANS); }
#endif
    return PAYMENT;
}
```

```
int do_stock_level(user_id)
{
        int rtval;
        if ((rtval = get_SL_input(stkl_dataptr)) != GOOD_INPUT) {
            return rtval;
         }


#ifdef NULL_TRANS

/* If LOOPBACK client just fill some dummy data and send it -- Roopa */

        rtval = user_id % 20;
         if (rtval == 0) {
            nap(2500);
         } else {
             nap(rtval*100);
         }
        fill_sl();
         display_stock_level(stkl_dataptr);
#endif

#ifndef NULL_TRANS
#ifdef DEBUG
        sprintf(debug_msg,"Stock threshold %d\n", stkl_dataptr-
>threshold);
        append_debug_msg( logtpcc, debug_msg);
#endif
        if (tpcall("STKL_SVC", (char *) stkl_dataptr, sizeof(struct
STKL_DATA),
                   (char **) &stkl_dataptr, &result, 0 ) != -1) {
                if (tpurcode == 0)
                    display_stock_level(stkl_dataptr);
                else if ( tpurcode == SQL_ERROR ) {
                    /* return with non-NULL error message */
                    SQL_error(stkl_dataptr->header.sql_code,
                                stkl_dataptr->header.isam_code);
                }
        } else {
                TUXEDO_err(STOCK_TRANS);
        }
#endif
        return STOCK_LEVEL;
}

static
int do_delivery(user_id)
{
        int rtval;
        if((rtval = get_ID_input(dvry_dataptr)) != GOOD_INPUT) {
            return rtval;
        }


#ifdef NULL_TRANS

/* If LOOPBACK client just fill some dummy data and send it -- Roopa */

        rtval = user_id % 20;
         if (rtval == 0) {
             nap(2500);
```

```
        } else {
            nap(rtval*100);
        }
        fill_dl();
        display_delivery(dvry_dataptr);
#endif

#ifndef NULL_TRANS
#ifdef DEBUG
        sprintf(debug_msg,"Delivery carrier_id %d\n", dvry_dataptr-
>carrier_id);
        append_debug_msg( logtpcc, debug_msg);
#endif
        if (tpacall("DVRY_SVC", (char *)dvry_dataptr, sizeof(struct
DVRY_DATA),
                    TPNOREPLY) != -1) {
                display_delivery(dvry_dataptr);
        } else {
                TUXEDO_err(DELIVERY_TRANS);
        }
#endif
        return DELIVERY;
}

static
int do_order_status(user_id)
{
        int rtval;
        if ((rtval = get_OS_input(ords_dataptr)) != GOOD_INPUT) {
            return rtval;
        }


#ifdef NULL_TRANS

/* If LOOPBACK client just fill some dummy data and send it -- Roopa */

        rtval = user_id % 20;
        if (rtval == 0) {
            nap(2500);
        } else {
            nap(rtval*100);
        }
        fill_os();
        display_order_status(ords_dataptr);
#endif

#ifndef NULL_TRANS
#ifdef DEBUG
        sprintf(debug_msg,"Order status c_id %d\n", ords_dataptr->c_id);
        append_debug_msg( logtpcc, debug_msg);
#endif
        if (tpcall("ORDS_SVC", (char *)ords_dataptr, sizeof(struct
ORDS_DATA),
                (char **) &ords_dataptr, &result, 0 ) != -1) {
            if (tpurcode == 0)
                display_order_status(ords_dataptr);
            else if ( tpurcode == SQL_ERROR ) {
                /* return with non-NULL error message */
                SQL_error(ords_dataptr->header.sql_code,
                          ords_dataptr->header.isam_code);
            }
        } else {
                TUXEDO_err(ORDER_STAT_TRANS);
        }
#endif
        return ORDER_STATUS;
}



#ifdef NULL_TRANS /* Functions to fill data for Loopback -- Roopa */
/* Fill neworder structure for LoopBack - Roopa */
void fill_newo()
{
        int i = 0;

        newo_dataptr->o_id = ORDER_ID_LB;
        newo_dataptr->w_tax = W_TAX_LB;
        newo_dataptr->d_tax = D_TAX_LB;
        newo_dataptr->total = TOTAL_LB;
        newo_dataptr->c_discount = C_DISCOUNT_LB;
        newo_dataptr->items_valid = ITEMS_VALID_LB;
        strcpy(newo_dataptr->new_date, NEW_DATE_LB );
        strcpy(newo_dataptr->c_last, C_LAST_LB );
        strcpy(newo_dataptr->c_credit, C_CREDIT_LB);
        for(i=0; i<newo_dataptr->o_ol_cnt; i++){
                newo_dataptr->ol_table[i].s_quantity = S_QTY_LB;
                strcpy(newo_dataptr->ol_table[i].name_i, I_NAME_LB);
        /* strcpy(newo_dataptr->ol_table[i].brand_generic,
BRAND_GENERIC_LB); */
        newo_dataptr->ol_table[i].brand_generic =  BRAND_GENERIC_LB;
                newo_dataptr->ol_table[i].price = PRICE_LB;
                newo_dataptr->ol_table[i].ol_amount = OL_AMOUNT_LB;
        }
}
#endif

#ifdef NULL_TRANS /* Functions to fill data for Loopback -- Roopa */
/* Fill payment structure for LoopBack - Roopa */
void fill_pmt()
{
        strcpy(pmt_dataptr->pay_date , PAY_DATE_LB);
        strcpy(pmt_dataptr->w_name , W_NAME_LB);
        strcpy(pmt_dataptr->w_street_1 , W_STREET_1_LB);
        strcpy(pmt_dataptr->w_street_2 , W_STREET_2_LB);
        strcpy(pmt_dataptr->w_city , W_CITY_LB);
        strcpy(pmt_dataptr->w_state , W_STATE_LB);
        strcpy(pmt_dataptr->w_zip , W_ZIP_LB);

        strcpy(pmt_dataptr->d_name , D_NAME_LB);
        strcpy(pmt_dataptr->d_street_1 , D_STREET_1_LB);
        strcpy(pmt_dataptr->d_street_2 , D_STREET_2_LB);
        strcpy(pmt_dataptr->d_city , D_CITY_LB);
        strcpy(pmt_dataptr->d_state , D_STATE_LB);
        strcpy(pmt_dataptr->d_zip , D_ZIP_LB);

        strcpy(pmt_dataptr->c_first , C_FIRST_LB);
        strcpy(pmt_dataptr->c_middle , C_MIDDLE_LB);
        strcpy(pmt_dataptr->c_last , C_LAST_LB);
        strcpy(pmt_dataptr->c_phone , C_PHONE_LB);
        strcpy(pmt_dataptr->c_credit , C_CREDIT_LB);
```

```
        strcpy(pmt_dataptr->c_street_1 , C_STREET_1_LB);
        strcpy(pmt_dataptr->c_street_2 , C_STREET_2_LB);
        strcpy(pmt_dataptr->c_city , C_CITY_LB);
        strcpy(pmt_dataptr->c_state , C_STATE_LB);
        strcpy(pmt_dataptr->c_zip , C_ZIP_LB);

        pmt_dataptr->c_credit_lim = C_CREDIT_LIM_LB;
        pmt_dataptr->c_balance = C_BALANCE_LB;
        pmt_dataptr->c_discount = C_DISCOUNT_LB;
        pmt_dataptr->c_ytd_payment = C_YTD_PAYMENT_LB;
        pmt_dataptr->c_payment_cnt = C_PAYMENT_CNT_LB;

        strcpy(pmt_dataptr->c_date , C_DATE_LB);
        strcpy(pmt_dataptr->c_data , C_DATA_LB);
}
#endif

#ifdef NULL_TRANS /* Functions to fill data for Loopback -- Roopa */
/* Fill order status structure for LoopBack - Roopa */
void fill_os()
{

       int i=0;

        ords_dataptr->o_ol_cnt = 5;
       ords_dataptr->o_id = O_ID_LB;
        ords_dataptr->o_carrier_id = O_CARRIER_ID_LB;
        strcpy(ords_dataptr->c_last, C_LAST_LB);
        strcpy(ords_dataptr->c_first, C_FIRST_LB);
        strcpy(ords_dataptr->c_middle, C_MIDDLE_LB);
        strcpy(ords_dataptr->cur_date, CUR_DATE_LB);
        ords_dataptr->c_balance = C_BALANCE_LB;

       for(i=0; i<ords_dataptr->o_ol_cnt; i++){
              ords_dataptr->ol_table[i].ol_i_id = OL_I_ID_LB;
              ords_dataptr->ol_table[i].ol_supply_w_id =
OL_SUPPLY_W_ID_LB;
              ords_dataptr->ol_table[i].ol_quantity = OL_QUANTITY_LB;
              ords_dataptr->ol_table[i].ol_amount = OL_AMOUNT_LB;
              strcpy(ords_dataptr->ol_table[i].delivery_date,
DELIVERY_DATE_LB );
       }
}
#endif


#ifdef NULL_TRANS /* Functions to fill data for Loopback -- Roopa */
/* Fill stock level structure for LoopBack - Roopa */
void fill_sl()
{

        stkl_dataptr->stock_count = STOCK_COUNT_LB;
}
#endif

#ifdef NULL_TRANS /* Functions to fill data for Loopback -- Roopa */
/* Fill delivery structure for LoopBack - Roopa */
/* Nothing to fill actually- Put a dummy function just for consistency */
void fill_dl()
{
}
#endif

#ifdef DEBUG
append_debug_msg( debug_fd, msg )
int debug_fd;
char *msg;
{

        write_lines( debug_fd, msg );
}

/* The maximum number of chars that vi can handle if all control chars. */
#define LINE_LEN      920
/*
 * write_lines writes the character string "inbuf" to the file fd,
 * It handles the situation where the buffer is over LINE_LEN characters
to
 * avoid the problem of "the line is too long" in the file.
 */
write_lines( fd, inbuf )
int  fd;
char *inbuf;
{
        int line_num, chars_out, nlines, nbytes;
        static int last_write=0;

       nbytes = strlen( inbuf );
        /*
         * write a new-line to avoid the problem of "the line is too long"
         * in the log files if the total length of the first write will be
         * over LINE_LEN chars.
         */
         if ( (last_write + nbytes) > LINE_LEN ) {
               write(fd, "\n", 1 );
               last_write = 0;
        }
        nlines = (nbytes/LINE_LEN);  /* over LINE_LEN characters */
        for(line_num=0;line_num<=nlines; line_num++)  {
           chars_out = LINE_LEN*line_num;

           if ( line_num == nlines )  {
              write(fd, inbuf+chars_out, nbytes-chars_out);
              last_write = ( (inbuf[nbytes-1] == '\n') ? 0 : nbytes-
chars_out );
           }
           else  {  /* if the buffer is over LINE_LEN, write seperate
lines */
              write(fd, inbuf+chars_out, LINE_LEN);
              write(fd, "\n", 1 );
              last_write = 0;
           }
        }
}
#endif
```

utils.c

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

```
#include <sys/time.h>
#include <sys/signal.h>
#ifdef NULL_TRANS
#include "/usr/ucbinclude/sys/signal.h" /* addition, for NULL_TRANS 11/14
*/
#endif
#include "utils.h"
/*
 * Make the time stamp from the format: YYYY-MM-DD hh:mm:ss to the format:
 * DD-MM-YYYY hh:mm:ss. Precision is either YEAR_TO_DATE or
YEAR_TO_SECOND.
 * If the precision is YEAR_TO_DATE, return a string with the format:
 * "DD-MM-YYYY".
 */
#define Y_POS   0
#define M_POS   5
#define D_POS   8

#define NEW_Y_POS   6
#define NEW_M_POS   3
#define NEW_D_POS   0

extern
void convert_datetime(orig_date, new_date, precision)
char *orig_date;
char *new_date;
short int  precision;
{
    /* Convert only if non-NULL  string */
    if (*orig_date != '\0') {
        strcpy(new_date, orig_date);
        /* make the string DD-MM-YYYY */
        *new_date++ = *(orig_date + D_POS);
        *new_date++ = *(orig_date + (D_POS + 1));
        *new_date++ = '-';
        *new_date++ = *(orig_date + M_POS);
        *new_date++ = *(orig_date + (M_POS + 1));
        *new_date++ = '-';
        *new_date++ = *(orig_date + Y_POS);
        *new_date++ = *(orig_date + (Y_POS + 1));
        *new_date++ = *(orig_date + (Y_POS + 2));
        *new_date++ = *(orig_date + (Y_POS + 3));
        if (precision == YEAR_TO_DATE)
             *new_date = '\0';
    } else {
       *new_date = '\0';
    }
}

extern
double get_time()
{
    struct  timeval now;
    struct  timezone tp;

    gettimeofday(&now, &tp);
    return ( (double) (now.tv_sec) +
        (now.tv_usec/1000000.0) );
}

#ifdef NULL_TRANS
extern long savemask;

void sigalarm();
struct  sigvec vec_alarm = {sigalarm,0,0}; /* changed sigvecTOR to sigvec
*/
                                           /* for NULL_TRANS 11/14 */
#define MASK(s) (1L << ((s) - 1))
void init_ms_sleep()
{
    savemask = sigblock(MASK(SIGALRM));
    if (sigvec(SIGALRM, &vec_alarm, 0) == -1) {
         perror("SIGALRM failed");
         exit(1);
    }
}


void sigalarm()
{
        /* do nothing to the alarm signal, need some work later to catch
alarm
           signal and react to it */
}

/*************************************************************************
*****
*  ms_sleep executes setitimer to simulate "sleep", it can sleep
fractional
*  seconds
*************************************************************************
*****/

void ms_sleep( wait )
double wait;
{
     struct  itimerval value;

     /* it_value = 0.0 disables timer */
     if ( wait > 0.0 ) {
          value.it_value.tv_sec = (int) (wait);
          value.it_value.tv_usec = 1000000 * (wait -
value.it_value.tv_sec);
          value.it_interval.tv_sec = 0;
          value.it_interval.tv_usec = 0;
          if (setitimer(ITIMER_REAL,&value,0) == -1) {
              perror("Setitimer failed");
              exit(1);
          }
          sigpause(savemask);
     }
}

#endif
```

do_tpcc.h

```
#ifndef __DO_TPCC_H__
#define __DO_TPCC_H__

#define MAX_OL        15       /* Maximum items in an order */
```

```
#define MIN_OL        5
#define MAX_FLDS       200      /* Maximum fields in a TPC-C form */
#define D_PER_W              10

#define ON     1
#define OFF    0

#define TRUE   1
#define FALSE  0

#define YES    1
#define NO     0

#define INSIZE  1024

#define RETRY_MSG       "Try again..."
#define COMMIT_NEWO  99
#define INVALID_NEWO  1
#define SQL_ERROR    -1
#define OS_ERROR     -2
#define CANCEL       -1

#endif
```

screens.h

```
#define F_INT 1
#define F_STR 2
#define F_MON 3

#define MAXVSZ 17
#define FSTR "                " /* Must be MAXVSZ in length! */

typedef char byte;

#define REQ_MASK 1
#define DP_MASK 2

struct field {
       byte r,c;          /* row, col of 1st input position in field */
       byte nu,nd,nl,nr;   /* Indices of neigbors: up, down, left, right
*/
       byte sz;           /* Number of input postions in field */
       byte ty;           /* type of input value: integer, money, string
*/
       byte fl;           /* flags-- 1="yes"
                                  b0: required field?
                                   b1: contains decimal pt? */
       byte ln;           /* if positive, index of linked field */
       char val[MAXVSZ+1];   /* field buffer */
       };

struct field NO_screen[47]={
       2 ,29,46,1 ,46,1 ,2 ,F_INT,1, 0,FSTR,
       3 ,12,0 ,2 ,0 ,2 ,4 ,F_INT,1, 1,FSTR,
       7 , 3,1 ,5 ,1 ,3 ,4 ,F_INT,1, 2,FSTR,
       7 ,10,1 ,6 ,2 ,4 ,6 ,F_INT,1, 3,FSTR,
       7 ,45,1 ,7 ,3 ,5 ,2 ,F_INT,1, 4,FSTR,
       8 , 3,2 ,8 ,4 ,6 ,4 ,F_INT,1, 5,FSTR,
       8 ,10,3 ,9 ,5 ,7 ,6 ,F_INT,1, 6,FSTR,
       8 ,45,4 ,10,6 ,8 ,2 ,F_INT,1, 7,FSTR,
       9 , 3,5 ,11,7 ,9 ,4 ,F_INT,1, 8,FSTR,
       9 ,10,6 ,12,8 ,10,6 ,F_INT,1, 9,FSTR,
       9 ,45,7 ,13,9 ,11,2 ,F_INT,1,10,FSTR,
       10, 3,8 ,14,10,12,4 ,F_INT,1,11,FSTR,
       10,10,9 ,15,11,13,6 ,F_INT,1,12,FSTR,
       10,45,10,16,12,14,2 ,F_INT,1,13,FSTR,
       11, 3,11,17,13,15,4 ,F_INT,1,14,FSTR,
       11,10,12,18,14,16,6 ,F_INT,1,15,FSTR,
       11,45,13,19,15,17,2 ,F_INT,1,16,FSTR,
       12, 3,14,20,16,18,4 ,F_INT,0,17,FSTR,
       12,10,15,21,17,19,6 ,F_INT,0,18,FSTR,
       12,45,16,22,18,20,2 ,F_INT,0,19,FSTR,
       13, 3,17,23,19,21,4 ,F_INT,0,20,FSTR,
       13,10,18,24,20,22,6 ,F_INT,0,21,FSTR,
       13,45,19,25,21,23,2 ,F_INT,0,22,FSTR,
       14, 3,20,26,22,24,4 ,F_INT,0,23,FSTR,
       14,10,21,27,23,25,6 ,F_INT,0,24,FSTR,
       14,45,22,28,24,26,2 ,F_INT,0,25,FSTR,
       15, 3,23,29,25,27,4 ,F_INT,0,26,FSTR,
       15,10,24,30,26,28,6 ,F_INT,0,27,FSTR,
       15,45,25,31,27,29,2 ,F_INT,0,28,FSTR,
       16, 3,26,32,28,30,4 ,F_INT,0,29,FSTR,
       16,10,27,33,29,31,6 ,F_INT,0,30,FSTR,
       16,45,28,34,30,32,2 ,F_INT,0,31,FSTR,
       17, 3,29,35,31,33,4 ,F_INT,0,32,FSTR,
       17,10,30,36,32,34,6 ,F_INT,0,33,FSTR,
       17,45,31,37,33,35,2 ,F_INT,0,34,FSTR,
       18, 3,32,38,34,36,4 ,F_INT,0,35,FSTR,
       18,10,33,39,35,37,6 ,F_INT,0,36,FSTR,
       18,45,34,40,36,38,2 ,F_INT,0,37,FSTR,
       19, 3,35,41,37,39,4 ,F_INT,0,38,FSTR,
       19,10,36,42,38,40,6 ,F_INT,0,39,FSTR,
       19,45,37,43,39,41,2 ,F_INT,0,40,FSTR,
       20, 3,38,44,40,42,4 ,F_INT,0,41,FSTR,
       20,10,39,45,41,43,6 ,F_INT,0,42,FSTR,
       20,45,40,46,42,44,2 ,F_INT,0,43,FSTR,
       21, 3,41,0 ,43,45,4 ,F_INT,0,44,FSTR,
       21,10,42,0 ,44,46,6 ,F_INT,0,45,FSTR,
       21,45,43,0 ,45, 0,2 ,F_INT,0,46,FSTR,
       };
struct field PA_screen[6]={
       4 ,52,5 ,1 ,5 ,1, 2 ,F_INT,1, 0,FSTR,
       9 ,11,0, 2 ,0 ,2 ,4 ,F_INT,1, 4,FSTR,
       9 ,33,0 ,4 ,1 ,3 ,4 ,F_INT,1, 2,FSTR,
       9 ,54,0 ,4 ,2 ,4 ,2 ,F_INT,1, 3,FSTR,
       10,29,2 ,5 ,3 ,5 ,16,F_STR,1, 1,FSTR,
       15,24,4 ,0 ,4 ,0 ,5 ,F_MON,1, 5,FSTR, /*Not counting 2 decimal
places*/
       };
struct field OS_screen[3]={
       2 ,29,2 ,1 ,2 ,1 ,2 ,F_INT,1, 0,FSTR,
       3 ,11,0 ,2 ,0 ,2 ,4 ,F_INT,1, 2,FSTR,
       3 ,44,1 ,0 ,1 ,0 ,16,F_STR,1, 1,FSTR,
       };
struct field ID_screen[1]={
       4 ,17,0 ,0 ,0 ,0 ,2 ,F_INT,0, 0,FSTR,
       };
struct field SL_screen[1]={
       4 ,24,0 ,0 ,0 ,0 ,2 ,F_INT,0, 0,FSTR,
       };
```

```
struct field UI_screen[1]={
       24,10,0 ,0 ,0 ,0 ,4 ,F_INT,0, 0,FSTR,
       };
```

term.h

```
#ifndef __TPCC_TERM_H__
#define __TPCC_TERM_H__

extern void init_forms();
extern int  init_term();
extern void restore_term();
extern void send_menu();
extern int  get_NO_input();
extern int  get_PA_input();
extern int  get_OS_input();
extern int  get_ID_input();
extern int  get_SL_input();
extern void clr();
extern void bell();
extern void display_new_order();
extern void display_payment();
extern void display_order_status();
extern void display_stock_level();
extern void display_delivery();
extern void display_msg();
extern int  get_transaction();

#endif
```

tpcc_info.h

```
/*
 * $Header: tpcc_info.h 7030100.1 95/07/19 15:11:37 plai Generic<base> $
Copyr (c) 1995 Oracle
 */
/*=================================================================+
 |          Copyright (c) 1995  Oracle Corp, Redwood Shores, CA    |
 |                     OPEN SYSTEMS PERFORMANCE GROUP              |
 |                          All Rights Reserved                    |
 +=================================================================+
 | FILENAME
 |     tpcc_info.h
 | DESCRIPTION
 |     Include file for TPC-C benchmark programs.
 +=================================================================*/

#ifndef TPCC_INFO_H
#define TPCC_INFO_H

#include "utils.h"

/*
* Extracted from databuf.h
*/

#define I_NAME_LEN     24
#define I_DATA         50
#define W_NAME_LEN     10
#define ADDR_LEN       20
#define STATE_LEN      2
#define ZIP_LEN                9
#define DIST_INFO_LEN  24
#define S_DATA_LEN     50
#define D_NAME_LEN     10
#define H_DATA_LEN     24
#define CARRIER_LEN    2

#define C_LAST_LEN     17
#define C_MID_LEN      2
#define PHONE_LEN      16
#define CREDIT_LEN     2

#define C_DATA_LEN  500
#define BC_DATA_LEN  23

#define DATETIME_LEN  19

#define MAX_OL         15
#define C_LAST_LEN     17



#ifdef NULL_TRANS
/* Constants for the LoopBack for neworder -- Roopa */
#define ORDER_ID_LB    100
#define W_TAX_LB       9.99
#define D_TAX_LB       9.99
#define TOTAL_LB       999.9
#define C_DISCOUNT_LB  9.99
#define NEW_DATE_LB    "ddmmyy"
#define C_CREDIT_LB    "xx"
#define I_NAME_LB      "xxxx"
#define BRAND_GENERIC_LB      'x'
#define S_QTY_LB       99
#define PRICE_LB       99.9
#define OL_AMOUNT_LB   999.9
#define ITEMS_VALID_LB 1
/* Constants added for Loop back for 3 tier with Tuxedo for New order
  -- Latha 04/30 */
#define N_O_OL_CNT     9
#define N_C_LAST       "17_XXXXXXXXXXXXXXX"
#define N_C_CREDIT     "3_9"
#define N_C_DISCOUNT   99
#define N_W_TAX                99
#define N_D_TAX                99
#define N_NEW_DATE     "20_12345678901234567"
#define N_TOTAL_AMOUNT 9999
#define N_I_NAME       "25_XXXXXXXXXXXXXXXXXXXXXX"
#define N_S_QUANTITY   99
#define N_BRAND_GEN    '3'
#define N_PRICE        99
#define N_NOL_AMOUNT   99
#define N_ITEMS_VALID  0
#define COMMIT_NEWO    99  /* Added for 3 tier loop backup on seeing
                              ex_trans.c fill_newo func. Latha 04/30 */

/* Constants for the LoopBack for payment -- Roopa */
#define PAY_DATE_LB    "ddmmyy"
#define W_NAME_LB      "xxxx"
```

```
#define W_STREET_1_LB  "xxxx"
#define W_STREET_2_LB  "xxxx"
#define W_CITY_LB      "xxxx"
#define W_STATE_LB     "xxxx"
#define W_ZIP_LB       "xxxx"
#define D_NAME_LB      "xxxx"
#define D_STREET_1_LB  "xxxx"
#define D_STREET_2_LB  "xxxx"
#define D_CITY_LB      "xxxx"
#define D_STATE_LB     "xxxx"
#define D_ZIP_LB       "xxxx"
#define C_MIDDLE_LB    "xxxx"
#define C_PHONE_LB     "xxxx"
#define C_STREET_1_LB  "xxxx"
#define C_STREET_2_LB  "xxxx"
#define C_CITY_LB      "xxxx"
#define C_STATE_LB     "xxxx"
#define C_ZIP_LB       "xxxx"
#define C_CREDIT_LIM_LB       9999.99
#define C_BALANCE_LB  99.99
#define C_DISCOUNT_LB 9.99
#define C_YTD_PAYMENT_LB      99.99
#define C_PAYMENT_CNT_LB      9
#define C_DATE_LB      "ddmmyy"
#define C_DATA_LB      "xxxxxxxxxxxxxxxxxxxxxxxx"
#define C_ID_LB               99

/* Loopback Constants for Order Status -- Roopa */
#define O_ID_LB        99
#define O_CARRIER_ID_LB       9
#define C_LAST_LB      "xxxx"
#define C_FIRST_LB     "xxxx"
#define CUR_DATE_LB    "ddmmyy"
#define OL_I_ID_LB     99
#define OL_SUPPLY_W_ID_LB     99
#define OL_QUANTITY_LB 99
#define DELIVERY_DATE_LB      "ddmmyy"

/* Loopback Constants for Stock Level -- Roopa */
#define STOCK_COUNT_LB 99
#else
#define COMMIT_NEWO    99  /* Added for 3 tier loop backup on seeing
                              ex_trans.c fill_newo func. Latha 04/30 */

#endif  /* NULL_TRANS endif */


struct data_header {
    short int  dtype;
    short int  returncode;
    int        sql_code;
    int        isam_code;
};

struct OL_TABLE {
        short int  ol_supply_w_id;
        short int  ol_quantity;
        short int  s_quantity;
              int  ol_i_id;
        char       name_i[I_NAME_LEN + 1];
        char       brand_generic;
        double     price;
        double     ol_amount;
};

struct OL_TABLE2 {
              int  ol_i_id;
        short int  ol_supply_w_id;
        short int  ol_quantity;
        double     ol_amount;
        char       delivery_date[20];
};

struct NEWO_DATA {
        struct data_header header;
        short int       w_id;
        short int       d_id;
        int       c_id;
        short int       o_ol_cnt;
        short int       o_all_local;
        short int       items_valid; /* true if all valid */
        int             o_id;
        double          w_tax;
        double          d_tax;
        double          total;
        double          c_discount;
        char            new_date[DATETIME_LEN + 1];
        char            c_last[C_LAST_LEN];
        char            c_credit[CREDIT_LEN + 1];
        struct OL_TABLE ol_table[MAX_OL];
};

struct PMT_DATA {
        struct data_header header;
        short int w_id;
        short int d_id;
        int c_id;
        short int c_w_id;
        short int c_d_id;
        short int byname;
        double    h_amount;
        char      pay_date[20];

        char      w_name[11];
        char      w_street_1[21];
        char      w_street_2[21];
        char      w_city[21];
        char      w_state[3];
        char      w_zip[10];

        char      d_name[11];
        char      d_street_1[21];
        char      d_street_2[21];
        char      d_city[21];
        char      d_state[3];
        char      d_zip[10];

        char      c_first[17]; /* was C_LAST_LEN already includes +1 */
        char      c_middle[3];
        char      c_last[17];
        char      c_phone[17];
        char      c_credit[3];
```

```
        char      c_street_1[21];
        char      c_street_2[21];
        char      c_city[21];
        char      c_state[3];
        char      c_zip[10];
        double    c_credit_lim;
        double    c_balance;
        double    c_discount;
        double    c_ytd_payment;
        short int c_payment_cnt;
        char      c_date[20];
        char      c_data[200];
};

struct ORDS_DATA {
       struct data_header header;
        short int w_id;
        short int d_id;
        int c_id;
              int o_id;
        short int o_ol_cnt;
        short int byname;
        short     o_carrier_id;
        char      c_last[17];
        char      c_first[17];
        char      c_middle[3];
        char      cur_date[20];
        double    c_balance;
        struct OL_TABLE2 ol_table[15];
};

struct DVRY_DATA {
       struct data_header header;
       short int w_id;
       short int carrier_id;
       long  cltinit;
       double start_queue;
};

struct STKL_DATA {
       struct data_header header;
       short int w_id;
       short int stkl_d_id;
       short int threshold;
        int       stock_count;
};

#endif
```

trans_type.h

```
#ifndef __TPCC_TRANS_TYPE__
#define __TPCC_TRANS_TYPE__

/*
 * TPC-C Transaction Types
 */
#define GOOD_INPUT      1
#define WRONG_INPUT     0
#define NEW_ORDER       1
#define PAYMENT         2
#define ORDER_STATUS    3
#define DELIVERY        4
#define STOCK_LEVEL     5
#define QUIT            9

#endif
```

utils.h

```
#ifndef __TPCC_UTILS__
#define __TPCC_UTILS__

#define YEAR_TO_DATE   1
#define YEAR_TO_SECOND 2
#define MAX_STR_LEN    128

#define ERROR(x) fprintf(stderr,"Error: %s\n",x),exit()

extern void convert_datetime();
extern double get_time();

#ifdef NULL_TRANS
void init_ms_sleep();
void ms_sleep();
#endif

#endif
```

Makefile

```
#
# DEBUG - open log files for messages to write to log700, logtpcc, and
#         logtrans as requested. (Required for any type of debugging)
#
# DEBUG_{trans_type} - write debugging message for a specific type of
trans.
#
# DEBUG_ALL          - print debugging message for all types of
transaction.
#
# -DNEWO_SQL   - use SQL for new order instead of stored procedures
# -DPMT_SQL    - use SQL for payment instead of stored procedures
# -DORDS_SQL   - use SQL for order status instead of stored procedures
# -DDVRY_SQL   - use SQL for delivery instead of stored procedures
# -DSTCK_SQL   - use SQL for stock instead of stored procedures
#
#########################################################################
######
INFXINCL   = ${INFORMIXDIR}/incl
TUXEDO_INC = -I${ROOTDIR}/include -I${INFXINCL}
#
# NOTE:  compilation options for DA and DS change based on the version
#        of the PA RISC chip you are running on.
#
#OPT       = -Wl,-a,archive +O4 +Oprocelim +Onolimit +Ofastaccess
+Oentrysched +Olibcalls +DA1.1b +DS1.1b
#TRANS_TYPE = -DNEWO_SQL -DORDS_SQL -DSTCK_SQL -DPMT_SQL -DDVRY_SQL
TRANS_TYPE = -DNEWO_SQL -DORDS_SQL -DSTCK_SQL
```

```
# TRANS_TYPE = -DORDS_SQL -DSTCK_SQL
# OPT       = -O -Xt -V -kblended
#OPT       = -O -Xt -DNULL_TRANS
OPT       = -O -Xt
OPT1    = -bI:/usr/lpp/SEMETkex/SEMETkex.exp -bI:/usr/lpp/UPINkex/upin.exp

TUXEDO     = -D_HPUX_SOURCE ${TUXEDO_INC} ${OPT}

DEBUG_ALL  = -DDEBUG_NEWO -DDEBUG_PMT -DDEBUG_ORDS -DDEBUG_DVRY -
DDEBUG_STKL

#For programmers to use
CFLAGS      = ${OPT} ${TRANS_TYPE}
# CFLAGS           = ${OPT} ${TRANS_TYPE}
# CFLAGS           =  ${TRANS_TYPE} -DDEBUG -DDEBUG2

#For auditor to use, specify ISO[1-7] one at a time for each test
#CFLAGS     = -DISO7 ${OPT} ${TRANS_TYPE}

all:            server tpccfs liberty
#all:            server tpccfs liberty batch_tpcc batch_reduce

#
# make the hp700 "smart" terminal client
#
hp700:          do_tpcc.o hp700.o utils.o
              buildclient -v -f 'do_tpcc.o hp700.o utils.o' -o client

#
# make the libery "dumb" terminal client
#
liberty:       do_tpcc.o liberty.o utils.o
              buildclient -v -f 'do_tpcc.o liberty.o utils.o -lm' -o
client

ansi:  do_tpcc.o ansi.o utils.o Makefile
#             buildclient -v -f 'do_tpcc.o ansi.o utils.o -lm' -o client
              ( CFLAGS=" -g -I$(TUXEDO_INC) "; export CFLAGS; \
              buildclient -v -f 'do_tpcc.o ansi.o utils.o \
              -L /usr/lib/libp -lx -lm \
              -L /usr/ucblib -lucb' -o client )
              cp /home/audit/client/client /home/client1

do_tpcc.o:     do_tpcc.c  do_tpcc.h Makefile
              cc $(CFLAGS) $(TUXEDO_INC) -lm -c do_tpcc.c
#             Use the one below for LOOPBACK client - Roopa
#             cc $(CFLAGS) $(TUXEDO_INC)  -L /usr/lib/libp -lx \
#             -lm  -c do_tpcc.c

hp700.o:       hp700.c hp700.h
              cc $(CFLAGS) -c hp700.c

liberty.o:     liberty.c
              cc $(CFLAGS) -c liberty.c

ansi.o: ansi.c Makefile
              cc $(CFLAGS)  -c    ansi.c

utils.o:       utils.h
              cc $(CFLAGS) -c utils.c

ex_trans.o:     ex_trans.ec ex_trans.h
              esql $(CFLAGS) -c ex_trans.ec

batch_reduce:  batch_reduce.o
              cc -o batch_reduce $(CFLAGS) batch_reduce.c

tpccfs.o:       tpccfs.c Makefile
              cc -DDBUSER=iti -DDBPASS="" $(CFLAGS) $(TUXEDO_INC) -c
tpccfs.c

# For Unisys UNIX use this
#tpccfs:         tpccfs.o
#             buildserver -v \
#                           -s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s
STKL_SVC \
#                         -s DVRY_SVC \
#                           -o tpccfs \
#                                -f 'tpccfs.o '
#
# For Unixware use this
tpccfs:         tpccfs.o
              buildserver -v \
                           -s NEWO_SVC -s PMT_SVC -s ORDS_SVC -s STKL_SVC
\
                         -s DVRY_SVC \
                          -o tpccfs \
                                -f 'tpccfs.o -lsocket -lnsl'

server:        ex_trans.o
              esql -o server $(CFLAGS) ex_trans.ec
#             esql -o server $(CFLAGS) ex_trans.ec -lsocket -lnsl

link:
              cp server dvry_server
              cp server newo_server
              cp server pmt_server
              cp server ords_server
              cp server stck_server



msacid_int:    ex_trans.o msacid.o
              esql -o msacid $(CFLAGS) ex_trans.ec msacid.c

msacid:
       @$(MAKE) CFLAGS="$(CFLAGS) -DBATCH_TPCC -DACID_TEST" msacid_int



batch_tpcc_int:        ex_trans.o batch_tpcc.o
              esql -o batch_tpcc $(CFLAGS) ex_trans.ec batch_tpcc.c

batch_tpcc:
       @$(MAKE) CFLAGS="$(CFLAGS) -DBATCH_TPCC" batch_tpcc_int


tar:
       tar -cvf tpcc.tar *.c *.h *.ec Makefile

clean:
```

```
	rm -f *.o client server dvry_server newo_server pmt_server
ords_server \
		stck_server tpccfs ex_trans.c batch_tpcc batch_reduce

install:	liberty tpccfs server
	rm -f /usr2/tpcc/bin/*server
	rm -f /usr2/tpcc/bin/client
	rm -f /usr2/tpcc/bin/tpccfs
	ln server /usr2/tpcc/bin/
	ln /usr2/tpcc/bin/server /usr2/tpcc/bin/newo_server
	ln /usr2/tpcc/bin/server /usr2/tpcc/bin/pmt_server
	ln /usr2/tpcc/bin/server /usr2/tpcc/bin/ords_server
	ln /usr2/tpcc/bin/server /usr2/tpcc/bin/dvry_server
	ln /usr2/tpcc/bin/server /usr2/tpcc/bin/stck_server
	ln client /usr2/tpcc/bin/
	ln tpccfs /usr2/tpcc/bin/
#	ln batch_tpcc /usr2/tpcc/bin/
#	ln batch_reduce /usr2/tpcc/bin/
```

pldel.c

```
#ifdef RCSID
static char *RCSid =
   "$Header: pldel.c 7030100.1 95/07/19 14:47:20 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
 |    pldel.c
 | DESCRIPTION
 |    OCI version (using PL/SQL stored procedure) of
 |    DELIVERY transaction in TPC-C benchmark.
 +=========================================================================*/

#include "tpcc.h"
#include "tpccpl.h"


#ifdef ISO5
#define SQLTXT "BEGIN adelivery.adeliver (:w_id, :cr_id, :o_id, :retry);
END;"
#else
#define SQLTXT "BEGIN delivery.deliver (:w_id, :cr_id, :o_id, :retry);
END;"
#endif

#define NDISTS 10


struct delctx {
   sb2 del_o_id_ind[NDISTS];
   ub2 del_o_id_len[NDISTS];
   ub2 del_o_id_rcode[NDISTS];
   ub4 del_o_id_csize;
};

typedef struct delctx delctx;

delctx *dctx;



pldelinit ()

{

   int i;
   text stmbuf[1024];

   dctx = (delctx *) malloc (sizeof(delctx));

   OOPEN(&tpclda,&curd);

   sprintf ((char *) stmbuf, SQLTXT);
   OPARSE(&tpclda,&curd,stmbuf,NA,FALSE,VER7);

   for (i = 0; i < NDISTS; i++) {
      dctx->del_o_id_ind[i] = TRUE;
      dctx->del_o_id_len[i] = sizeof(int);
   }
   dctx->del_o_id_csize = NDISTS;

   /* bind variables */

   OBNDRV(&tpclda,&curd,":w_id",ADR(w_id),SIZ(int),SQLT_INT);
   OBNDRV(&tpclda,&curd,":cr_id",ADR(o_carrier_id),SIZ(int),SQLT_INT);
   OBNDRV(&tpclda,&curd,":o_id",ADR(o_id),SIZ(int),SQLT_INT);
   OBNDRAA(&tpclda,&curd,":o_id",del_o_id,SIZ(int),SQLT_INT,
           dctx->del_o_id_ind,dctx->del_o_id_len,dctx-
>del_o_id_rcode,NDISTS,
           ADR(dctx->del_o_id_csize));
   OBNDRV(&tpclda,&curd,":retry",ADR(retries),SIZ(int),SQLT_INT);

   return (0);

}



pldel ()

{

   int i;

   for (i = 0; i < NDISTS; i++) {
      dctx->del_o_id_ind[i] = TRUE;
      dctx->del_o_id_len[i] = sizeof(int);
   }
   dctx->del_o_id_csize = NDISTS;

   OEXEC(&tpclda,&curd);

   return (0);

}
```

```
void pldeldone ()

{

   if (dctx)
      free (dctx);

   if (oclose (&curd))
      errrpt (&tpclda, &curd);

}
```

plnew.c

```
#ifdef RCSID
static char *RCSid =
   "$Header: plnew.c 7030100.2 96/04/02 17:49:18 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
 |    plnew.c
 | DESCRIPTION
 |    OCI version (using PL/SQL stored procedure) of
 |    NEW ORDER transaction in TPC-C benchmark.
 +================================================================*/

#include "tpcc.h"
#include "tpccpl.h"
#ifdef TUX
#include <userlog.h>
#endif


#define SQLTXT1 "BEGIN neworder.enterorder (:w_id, :d_id, :c_id,
:o_ol_cnt, \
   :o_all_local, :c_discount, :c_last, :c_credit, :d_tax, :w_tax, :o_id, \
   :o_entry_d, :retry); END;"

#define SQLTXT2 "UPDATE stock SET s_order_cnt = s_order_cnt + 1, \
   s_ytd = s_ytd + :ol_quantity, s_remote_cnt = s_remote_cnt + :s_remote,
\
   s_quantity = s_quantity - :ol_quantity + \
   DECODE (SIGN (s_quantity - :ol_quantity - 10), -1, 91, 0) \
   WHERE s_i_id = :ol_i_id AND s_w_id = :ol_supply_w_id"

#define SQLTXT3 "\
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :10 AND s_w_id = :30 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :11 AND s_w_id = :31 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :12 AND s_w_id = :32 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :13 AND s_w_id = :33 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :14 AND s_w_id = :34 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :15 AND s_w_id = :35 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :16 AND s_w_id = :36 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :17 AND s_w_id = :37 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :18 AND s_w_id = :38 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :19 AND s_w_id = :39 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :20 AND s_w_id = :40 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :21 AND s_w_id = :41 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :22 AND s_w_id = :42 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :23 AND s_w_id = :43 AND s_i_id = i_id UNION
ALL \
SELECT i_id,s_w_id,i_price,i_name,i_data,s_dist_%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :24 AND s_w_id = :44 AND s_i_id = i_id"

#define SQLTXT4 "INSERT INTO order_line VALUES (:ol_o_id, :ol_d_id, \
   :ol_w_id, :ol_number, :ol_i_id, :ol_supply_w_id, NULL, :ol_quantity, \
   :ol_amount, :ol_dist_info)"

#define NITEMS 15


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 i_id_ind[NITEMS];
```

```
   sb2 w_id_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];

   ub2 nol_i_id_len[NITEMS];
   ub2 nol_supply_w_id_len[NITEMS];
   ub2 nol_quantity_len[NITEMS];
   ub2 nol_amount_len[NITEMS];
   ub2 i_name_len[NITEMS];
   ub2 s_quantity_len[NITEMS];
   ub2 i_price_len[NITEMS];
   ub2 ol_w_id_len[NITEMS];
   ub2 ol_d_id_len[NITEMS];
   ub2 ol_o_id_len[NITEMS];
   ub2 ol_number_len[NITEMS];
   ub2 i_id_len[NITEMS];
   ub2 w_id_len[NITEMS];
   ub2 s_remote_len[NITEMS];
   ub2 s_quant_len[NITEMS];
   ub2 i_data_len[NITEMS];
   ub2 s_data_len[NITEMS];
   ub2 s_dist_info_len[NITEMS];
   ub2 ol_dist_info_len[NITEMS];

   ub2 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 i_id_rcode[NITEMS];
   ub2 w_id_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];

   int ol_w_id[NITEMS];
   int ol_d_id[NITEMS];
   int ol_o_id[NITEMS];
   int ol_number[NITEMS];
   int i_id[NITEMS];
   int w_id[NITEMS];
   int s_remote[NITEMS];
   char i_data[NITEMS][51];
   char s_data[NITEMS][51];
   char s_dist_info[NITEMS][25];
};

typedef struct newctx newctx;

newctx *nctx;



plnewinit ()

{

   int i, j;
   text stmbuf[3000];
   char id[4];
   char sd[4];

   nctx = (newctx *) malloc (sizeof(newctx));

   /* open first cursor */

   OOPEN(&tpclda,&curn1);

   sprintf ((char *) stmbuf, SQLTXT1);
   OPARSE(&tpclda,&curn1,stmbuf,NA,FALSE,VER7);

   /* bind variables */

   OBNDRV(&tpclda,&curn1,":w_id",ADR(w_id),SIZ(w_id),SQLT_INT);
   OBNDRV(&tpclda,&curn1,":d_id",ADR(d_id),SIZ(d_id),SQLT_INT);
   OBNDRV(&tpclda,&curn1,":c_id",ADR(c_id),SIZ(c_id),SQLT_INT);
   OBNDRV(&tpclda,&curn1,":o_all_local",ADR(o_all_local),SIZ(o_all_local),
          SQLT_INT);

OBNDRV(&tpclda,&curn1,":o_ol_cnt",ADR(o_ol_cnt),SIZ(o_ol_cnt),SQLT_INT);

   OBNDRV(&tpclda,&curn1,":w_tax",ADR(w_tax),SIZ(w_tax),SQLT_FLT);
   OBNDRV(&tpclda,&curn1,":d_tax",ADR(d_tax),SIZ(d_tax),SQLT_FLT);
   OBNDRV(&tpclda,&curn1,":o_id",ADR(o_id),SIZ(o_id),SQLT_INT);
   OBNDRV(&tpclda,&curn1,":c_discount",ADR(c_discount),SIZ(c_discount),
          SQLT_FLT);
   OBNDRV(&tpclda,&curn1,":c_credit",c_credit,SIZ(c_credit),SQLT_STR);
   OBNDRV(&tpclda,&curn1,":c_last",c_last,SIZ(c_last),SQLT_STR);
   OBNDRV(&tpclda,&curn1,":o_entry_d",o_entry_d,SIZ(o_entry_d),SQLT_STR);
   OBNDRV(&tpclda,&curn1,":retry",ADR(retries),SIZ(retries),SQLT_INT);

   /* open second cursor */

   OOPEN(&tpclda,&curn2);

   sprintf ((char *) stmbuf, SQLTXT2);
   OPARSE(&tpclda,&curn2,stmbuf,NA,FALSE,VER7);

   /* bind variables */

   OBNDRA(&tpclda,&curn2,":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);

OBNDRA(&tpclda,&curn2,":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);
   OBNDRA(&tpclda,&curn2,":ol_quantity",nol_quantity,SIZ(int),SQLT_INT,
          nctx->nol_quantity_ind,nctx->nol_quantity_len,
          nctx->nol_quantity_rcode);
```

```
   OBNDRA(&tpclda,&curn2,":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++) {
      OOPEN(&tpclda,&curn3[i]);
      j = i + 1;
      sprintf ((char *) stmbuf, SQLTXT3, j, j, j, j, j, j, j, j, j, j, j,
j,
               j, j, j);
      OPARSE(&tpclda,&curn3[i],stmbuf,NA,FALSE,VER7);
      for (j = 0; j < NITEMS; j++) {
         sprintf (id, ":%d", j + 10);
         sprintf (sd, ":%d", j + 30);
         OBNDRA(&tpclda,&curn3[i],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]);

OBNDRA(&tpclda,&curn3[i],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] = NULL;
         nctx->nol_supply_w_id_len[j] = NULL;
      }

      ODEFIN(&tpclda,&curn3[i],1,nctx->i_id,SIZ(nctx-
>i_id[0]),SQLT_INT,NA,
             nctx->i_id_ind,NULL,NA,NA,nctx->i_id_len, nctx->i_id_rcode);
      ODEFIN(&tpclda,&curn3[i],2,nctx->w_id,SIZ(nctx-
>w_id[0]),SQLT_INT,NA,
             nctx->w_id_ind,NULL,NA,NA,nctx->w_id_len, nctx->w_id_rcode);
      ODEFIN(&tpclda,&curn3[i],3,i_price,SIZ(float),SQLT_FLT,NA,
             nctx->i_price_ind,NULL,NA,NA,nctx->i_price_len,
             nctx->i_price_rcode);
      ODEFIN(&tpclda,&curn3[i],4,i_name,SIZ(i_name[0]),SQLT_STR,NA,
             nctx->i_name_ind,NULL,NA,NA,nctx->i_name_len,nctx-
>i_name_rcode);
      ODEFIN(&tpclda,&curn3[i],5,nctx->i_data,SIZ(nctx-
>i_data[0]),SQLT_STR,NA,
             nctx->i_data_ind,NULL,NA,NA,nctx->i_data_len, nctx-
>i_data_rcode);
      ODEFIN(&tpclda,&curn3[i],6,nctx->s_dist_info,SIZ(nctx-
>s_dist_info[0]),
             SQLT_STR,NA,nctx->s_dist_info_ind,NULL,NA,NA,
             nctx->s_dist_info_len, nctx->s_dist_info_rcode);
      ODEFIN(&tpclda,&curn3[i],7,nctx->s_data,SIZ(nctx-
>s_data[0]),SQLT_STR,NA,
             nctx->s_data_ind,NULL,NA,NA,nctx->s_data_len, nctx-
>s_data_rcode);
      ODEFIN(&tpclda,&curn3[i],8,s_quantity,SIZ(int),SQLT_INT,NA,
             nctx->s_quantity_ind,NULL,NA,NA,nctx->s_quantity_len,
             nctx->s_quantity_rcode);
   }

   /* open fourth cursor */

   OOPEN(&tpclda,&curn4);

   sprintf ((char *) stmbuf, SQLTXT4);
   OPARSE(&tpclda,&curn4,stmbuf,NA,FALSE,VER7);

   /* bind variables */

   OBNDRA(&tpclda,&curn4,":ol_o_id",nctx->ol_o_id,SIZ(int),SQLT_INT,
          nctx->ol_o_id_ind,nctx->ol_o_id_len,nctx->ol_o_id_rcode);
   OBNDRA(&tpclda,&curn4,":ol_d_id",nctx->ol_d_id,SIZ(int),SQLT_INT,
          nctx->ol_d_id_ind,nctx->ol_d_id_len,nctx->ol_d_id_rcode);
   OBNDRA(&tpclda,&curn4,":ol_w_id",nctx->ol_w_id,SIZ(int),SQLT_INT,
          nctx->ol_w_id_ind,nctx->ol_w_id_len,nctx->ol_w_id_rcode);
   OBNDRA(&tpclda,&curn4,":ol_number",nctx->ol_number,SIZ(int),SQLT_INT,
          nctx->ol_number_ind,nctx->ol_number_len,nctx->ol_number_rcode);
   OBNDRA(&tpclda,&curn4,":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);

OBNDRA(&tpclda,&curn4,":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);
   OBNDRA(&tpclda,&curn4,":ol_quantity",nol_quantity,SIZ(int),SQLT_INT,
          nctx->nol_quantity_ind,nctx->nol_quantity_len,
           nctx->nol_quantity_rcode);
   OBNDRA(&tpclda,&curn4,":ol_amount",nol_amount,SIZ(float),SQLT_FLT,
          nctx->nol_amount_ind,nctx->nol_amount_len,nctx-
>nol_amount_rcode);
   OBNDRA(&tpclda,&curn4,":ol_dist_info",nctx->s_dist_info,
           SIZ(nctx->s_dist_info[0]),SQLT_STR,nctx->ol_dist_info_ind,
           nctx->ol_dist_info_len, nctx->ol_dist_info_rcode);

   return (0);

}



plnew ()

{

   int i, j, k;
   int rpc, rpc3, rowoff, iters;
   int onepass;

#if defined(ISO1) || defined(ISO7)
   int reread;
   char sdate[30];

   sysdate (sdate);
   printf ("New Order started at: %s\n", sdate);
#endif

retry:

#ifdef ISO7
   reread = 1;
#endif

   status = 0;                          /* number of invalid items */
   onepass = 1;
```

```
/* 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;
}

/* execute stored procedure */

if (oexec (&curn1)) {
   if (curn1.rc == NOT_SERIALIZABLE) {
      orol (&tpclda);
      retries++;
      goto retry;
   }
   else if (errrpt (&tpclda, &curn1) == RECOVERR) {
      orol (&tpclda);
      retries++;
      goto retry;
   }
   else {
      orol (&tpclda);
      return (-1);
   }
}

/* 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->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->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(float);
   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] = sizeof(nctx->s_dist_info[0]);
   nctx->s_remote_len[i] = sizeof(int);
   nctx->s_quant_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->s_remote_ind[i] = NA;
   nctx->s_quant_ind[i] = NA;

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

/* array update of stock table */

if (oexn (&curn2, o_ol_cnt, 0)) {
   if (curn2.rc == NOT_SERIALIZABLE) {
      orol (&tpclda);
      retries++;
      goto retry;
   }
   else if (errrpt (&tpclda, &curn2) == RECOVERR) {
      orol (&tpclda);
      retries++;
      goto retry;
   }
   else {
      orol (&tpclda);
      return (-1);
   }
}

/* continue to do array update of stock until whole array is processed
*/

if (curn2.rpc >= (o_ol_cnt - 1)) {
   rpc = curn2.rpc;
}
else {
#ifdef TUX
   userlog ("TPC-C server %d: more than 1 pass of OEXN!\n", proc_no);
#else
```

```
      fprintf (stderr, "TPC-C server %d: more than 1 pass of OEXN!\n",
proc_no);
#endif
      rpc = curn2.rpc;
      rowoff = rpc + 1;
      while (rowoff < o_ol_cnt) {
         if (oexn (&curn2, o_ol_cnt, rowoff)) {
            if (curn2.rc == NOT_SERIALIZABLE) {
               orol (&tpclda);
               retries++;
               goto retry;
            }
            else if (errrpt (&tpclda, &curn2) == RECOVERR) {
               orol (&tpclda);
               retries++;
               goto retry;
            }
            else {
               orol (&tpclda);
               return (-1);
            }
         }
         rpc += curn2.rpc;
         rowoff += curn2.rpc + 1;
      }
   }

#ifdef ISO7
iso7:
#endif

   /* array select from item and stock tables */

   if (oexfet (&curn3[d_id-1], o_ol_cnt, 0, 0)) {
      if (curn3[d_id-1].rc == NOT_SERIALIZABLE) {
         orol (&tpclda);
         retries++;
         goto retry;
      }
      else if (curn3[d_id-1].rc != NO_DATA_FOUND) {
         if (errrpt (&tpclda, &curn3[d_id-1]) == RECOVERR) {
            orol (&tpclda);
            retries++;
            goto retry;
         }
         else {
            orol (&tpclda);
            return (-1);
         }
      }
   }

   /* mark invalid items */

   rpc3 = curn3[d_id-1].rpc;
   if (curn3[d_id-1].rpc != o_ol_cnt)
      for (i = curn3[d_id-1].rpc; i < o_ol_cnt; i++)
         nctx->i_id_ind[i] = NA;

   /* 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
      orol (&tpclda);
      return (-1);
   }

   /* check for invalid items and reorder results if necessary */

   for (i = 0; i < o_ol_cnt; i++) {
      if (nctx->i_id_ind[i] != NA) {
         if ((nctx->i_id[i] != nol_i_id[i]) ||
             (nctx->w_id[i] != nol_supply_w_id[i])) {

            /* this item is invalid or results are out of order */

#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

            for (j = i + 1; j < o_ol_cnt; j++) {

               /* this item is valid, but results are out of order */

               if ((nctx->i_id_ind[j] != NA) &&
                   (nctx->i_id[j] == nol_i_id[i]) &&
                   (nctx->w_id[j] == nol_supply_w_id[i])) {
                  swapitemstock (i, j);
                  break;
               }
            }

            /* this item (not the last one) is invalid */

            if (j >= o_ol_cnt) {
               status++;
               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->s_remote_ind[i] = NA;
               nctx->s_quant_ind[i] = NA;
```

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

            onepass = 0;
            for (j = i + 1; j < o_ol_cnt; j++) {
               if (nctx->i_id_ind[j] == NA) {
                  swapitemstock (i, j);
                  break;
               }
            }
         }
      }
   }
   else {                          /* this item is invalid */
      status++;
      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->s_remote_ind[i] = NA;
      nctx->s_quant_ind[i] = NA;

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

/* more than 1 invalid item!!!  shouldn't happen in TPC-C */

if (status > 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
}
```

```
#ifdef ISO7
   sysdate (sdate);
   printf ("Item table read at: %s\n", sdate);
   for (i = 0; i < o_ol_cnt; i++) {
      if (nctx->nol_i_id_ind[i] != NA)
         printf ("   i_id = %d, i_price = %.2f\n", nol_i_id[i],
i_price[i]);
   }
   if (reread) {
      sleep (30);
      reread = 0;
      status = 0;
      onepass = 1;
      goto iso7;
   }
#endif

   /* 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) {
         nol_amount[i] = (float) (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 *= (1.0 - c_discount) * (1.0 + d_tax + w_tax);

   /* array insert into order line table */

   if (onepass && ((o_ol_cnt - status) > 0)) {
      if (oexn (&curn4, o_ol_cnt - status, 0)) {
         if (curn4.rc == NOT_SERIALIZABLE) {
            orol (&tpclda);
            retries++;
            goto retry;
         }
         else if (errrpt (&tpclda, &curn4) == RECOVERR) {
            orol (&tpclda);
            retries++;
            goto retry;
         }
         else {
            orol (&tpclda);
            return (-1);
         }
      }
      if (curn4.rpc != (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
         orol (&tpclda);
         return (-1);
      }
   }

   /* continue array insert into order line until whole array is processed
*/

   else if ((o_ol_cnt - status) > 0) {
#ifdef TUX
      userlog ("TPC-C server %d: more than 1 pass of OEXN!\n", proc_no);
#else
      fprintf (stderr, "TPC-C server %d: more than 1 pass of OEXN!\n",
proc_no);
#endif
      rpc = 0;
      for (rowoff = 0; rowoff < o_ol_cnt; rowoff++)
         if (nctx->nol_i_id_ind[rowoff] != NA)
            break;
      for (iters = rowoff + 1; iters < o_ol_cnt; iters++)
         if (nctx->nol_i_id_ind[iters] == NA)
            break;
      while ((rpc < (o_ol_cnt - status)) && (iters <= o_ol_cnt)) {
         if (oexn (&curn4, iters, rowoff)) {
            if (curn4.rc == NOT_SERIALIZABLE) {
               orol (&tpclda);
               retries++;
               goto retry;
            }
            else if (errrpt (&tpclda, &curn4) == RECOVERR) {
               orol (&tpclda);
               retries++;
               goto retry;
            }
            else {
               orol (&tpclda);
               return (-1);
            }
         }
         if (curn4.rpc != (iters - rowoff)) {
#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
            orol (&tpclda);
            return (-1);
         }
         rpc += curn4.rpc;
         for (rowoff = iters + 1; rowoff < o_ol_cnt; rowoff++)
            if (nctx->nol_i_id_ind[rowoff] != NA)
               break;
         for (iters = rowoff + 1; iters < o_ol_cnt; iters++)
            if (nctx->nol_i_id_ind[iters] == NA)
               break;
      }
   }

#ifdef ISO1
   sysdate (sdate);
   printf ("Sleep before commit/rollback at: %s\n", sdate);
   sleep (30);
   sysdate (sdate);
   printf ("Wake up after sleep at: %s\n", sdate);
#endif

   /* commit if no invalid item */

   if (status) {
      orol (&tpclda);
   }
   else {
      OCOM(&tpclda,&tpclda);
   }

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

   return (0);

}



void plnewdone ()

{

   int i;

   if (nctx)
      free (nctx);

   if (oclose (&curn1))
      errrpt (&tpclda, &curn1);
   if (oclose (&curn2))
      errrpt (&tpclda, &curn2);
   for (i = 0; i < 10; i++)
      if (oclose (&curn3[i]))
         errrpt (&tpclda, &curn3[i]);
   if (oclose (&curn4))
      errrpt (&tpclda, &curn4);

}



swapitemstock (i, j)

int i, j;

{

   int tempi;
   float tempf;
```

```
char tempstr[52];
ub2 tempub2;
sb2 tempsb2;

tempsb2 = nctx->i_id_ind[i];
nctx->i_id_ind[i] = nctx->i_id_ind[j];
nctx->i_id_ind[j] = tempsb2;
tempub2 = nctx->i_id_len[i];
nctx->i_id_len[i] = nctx->i_id_len[j];
nctx->i_id_len[j] = tempub2;
tempub2 = nctx->i_id_rcode[i];
nctx->i_id_rcode[i] = nctx->i_id_rcode[j];
nctx->i_id_rcode[j] = tempub2;
tempi = nctx->i_id[i];
nctx->i_id[i] = nctx->i_id[j];
nctx->i_id[j] = tempi;

tempsb2 = nctx->w_id_ind[i];
nctx->w_id_ind[i] = nctx->w_id_ind[j];
nctx->w_id_ind[j] = tempsb2;
tempub2 = nctx->w_id_len[i];
nctx->w_id_len[i] = nctx->w_id_len[j];
nctx->w_id_len[j] = tempub2;
tempub2 = nctx->w_id_rcode[i];
nctx->w_id_rcode[i] = nctx->w_id_rcode[j];
nctx->w_id_rcode[j] = tempub2;
tempi = nctx->w_id[i];
nctx->w_id[i] = nctx->w_id[j];
nctx->w_id[j] = tempi;

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);

}
```

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 stored procedure) of
 |    ORDER STATUS transaction in TPC-C benchmark.
 +=====================================================================*/

#include "tpcc.h"
#include "tpccpl.h"


#ifdef ISO8
#define SQLTXT "BEGIN aorderstatus.agetstatus (: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 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;"
#endif

#define NITEMS 15


struct ordctx {
   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];

   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_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];

   ub4 ol_supply_w_id_csize;
   ub4 ol_i_id_csize;
   ub4 ol_quantity_csize;
   ub4 ol_amount_csize;
   ub4 ol_delivery_d_csize;
};

typedef struct ordctx ordctx;

ordctx *octx;


plordinit ()

{

   int i;
   text stmbuf[1024];

   octx = (ordctx *) malloc (sizeof(ordctx));

   OOPEN(&tpclda,&curo);

   sprintf ((char *) stmbuf, SQLTXT);
   OPARSE(&tpclda,&curo,stmbuf,NA,FALSE,VER7);

   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(float);
      octx->ol_delivery_d_len[i] = sizeof(ol_delivery_d[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;

   /* bind variables */

   OBNDRV(&tpclda,&curo,":w_id",ADR(w_id),SIZ(w_id),SQLT_INT);
   OBNDRV(&tpclda,&curo,":d_id",ADR(d_id),SIZ(d_id),SQLT_INT);
   OBNDRV(&tpclda,&curo,":c_id",ADR(c_id),SIZ(c_id),SQLT_INT);
   OBNDRV(&tpclda,&curo,":byln",ADR(bylastname),SIZ(bylastname),SQLT_INT);
   OBNDRV(&tpclda,&curo,":c_last",c_last,SIZ(c_last),SQLT_STR);
   OBNDRV(&tpclda,&curo,":c_first",c_first,SIZ(c_first),SQLT_STR);
   OBNDRV(&tpclda,&curo,":c_middle",c_middle,SIZ(c_middle),SQLT_STR);

OBNDRV(&tpclda,&curo,":c_balance",ADR(c_balance),SIZ(c_balance),SQLT_FLT);
   OBNDRV(&tpclda,&curo,":o_id",ADR(o_id),SIZ(o_id),SQLT_INT);
   OBNDRV(&tpclda,&curo,":o_entry_d",o_entry_d,SIZ(o_entry_d),SQLT_STR);
   OBNDRV(&tpclda,&curo,":o_cr_id",ADR(o_carrier_id),SIZ(o_carrier_id),
          SQLT_INT);
   OBNDRV(&tpclda,&curo,":o_ol_cnt",ADR(o_ol_cnt),SIZ(o_ol_cnt),SQLT_INT);
   OBNDRAA(&tpclda,&curo,":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,ADR(octx-
>ol_supply_w_id_csize));
   OBNDRAA(&tpclda,&curo,":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,
           ADR(octx->ol_i_id_csize));
   OBNDRAA(&tpclda,&curo,":ol_quantity",ol_quantity,SIZ(int),SQLT_INT,
           octx->ol_quantity_ind,octx->ol_quantity_len,octx-
>ol_quantity_rcode,
           NITEMS,ADR(octx->ol_quantity_csize));
```

```
   OBNDRAA(&tpclda,&curo,":ol_amount",ol_amount,SIZ(float),SQLT_FLT,
           octx->ol_amount_ind,octx->ol_amount_len,octx->ol_amount_rcode,
           NITEMS,ADR(octx->ol_amount_csize));

OBNDRAA(&tpclda,&curo,":ol_d_d",ol_delivery_d,SIZ(ol_delivery_d[0]),SQLT_S
TR,
           octx->ol_delivery_d_ind,octx->ol_delivery_d_len,
           octx->ol_delivery_d_rcode,NITEMS,ADR(octx-
>ol_delivery_d_csize));

   return (0);

}



plord ()

{

   int i;

   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(float);
      octx->ol_delivery_d_len[i] = sizeof(ol_delivery_d[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;

   OEXEC(&tpclda,&curo);

   return (0);

}



void plorddone ()

{

   if (octx)
      free (octx);

   if (oclose (&curo))
      errrpt (&tpclda, &curo);

}
```

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 "tpcc.h"
#include "tpccpl.h"


#ifdef ATOMA
#define SQLTXT "BEGIN apayment.adopayment (:w_id, :d_id, :c_w_id, :c_d_id,
\
   :c_id, :byln, \
   :h_amount, :c_last, :w_street_1, :w_street_2, :w_city, :w_state, \
   :w_zip, :d_street_1, :d_street_2, :d_city, :d_state, :d_zip, :c_first,
\
   :c_middle, :c_street_1, :c_street_2, :c_city, :c_state, :c_zip,
:c_phone, \
   :c_since, :c_credit, :c_credit_lim, :c_discount, :c_balance, :c_data, \
   :h_date, :retry); END;"
#else
#define SQLTXT "BEGIN payment.dopayment (:w_id, :d_id, :c_w_id, :c_d_id, \
   :c_id, :byln, \
   :h_amount, :c_last, :w_street_1, :w_street_2, :w_city, :w_state, \
   :w_zip, :d_street_1, :d_street_2, :d_city, :d_state, :d_zip, :c_first,
\
   :c_middle, :c_street_1, :c_street_2, :c_city, :c_state, :c_zip,
:c_phone, \
   :c_since, :c_credit, :c_credit_lim, :c_discount, :c_balance, :c_data, \
   :h_date, :retry); END;"
#endif



plpayinit ()

{

   text stmbuf[1024];

   OOPEN(&tpclda,&curp);

   sprintf ((char *) stmbuf, SQLTXT);
   OPARSE(&tpclda,&curp,stmbuf,NA,FALSE,VER7);
```

```
   /* bind variables */

   OBNDRV(&tpclda,&curp,":w_id",ADR(w_id),SIZ(w_id),SQLT_INT);
   OBNDRV(&tpclda,&curp,":d_id",ADR(d_id),SIZ(d_id),SQLT_INT);
   OBNDRV(&tpclda,&curp,":c_w_id",ADR(c_w_id),SIZ(c_w_id),SQLT_INT);
   OBNDRV(&tpclda,&curp,":c_d_id",ADR(c_d_id),SIZ(c_d_id),SQLT_INT);
   OBNDRV(&tpclda,&curp,":c_id",ADR(c_id),SIZ(c_id),SQLT_INT);
   OBNDRV(&tpclda,&curp,":byln",ADR(bylastname),SIZ(bylastname),SQLT_INT);
   OBNDRV(&tpclda,&curp,":h_amount",ADR(h_amount),SIZ(h_amount),SQLT_FLT);
   OBNDRV(&tpclda,&curp,":c_last",c_last,SIZ(c_last),SQLT_STR);

OBNDRV(&tpclda,&curp,":w_street_1",w_street_1,SIZ(w_street_1),SQLT_STR);

OBNDRV(&tpclda,&curp,":w_street_2",w_street_2,SIZ(w_street_2),SQLT_STR);
   OBNDRV(&tpclda,&curp,":w_city",w_city,SIZ(w_city),SQLT_STR);
   OBNDRV(&tpclda,&curp,":w_state",w_state,SIZ(w_state),SQLT_STR);
   OBNDRV(&tpclda,&curp,":w_zip",w_zip,SIZ(w_zip),SQLT_STR);

OBNDRV(&tpclda,&curp,":d_street_1",d_street_1,SIZ(d_street_1),SQLT_STR);

OBNDRV(&tpclda,&curp,":d_street_2",d_street_2,SIZ(d_street_2),SQLT_STR);
   OBNDRV(&tpclda,&curp,":d_city",d_city,SIZ(d_city),SQLT_STR);
   OBNDRV(&tpclda,&curp,":d_state",d_state,SIZ(d_state),SQLT_STR);
   OBNDRV(&tpclda,&curp,":d_zip",d_zip,SIZ(d_zip),SQLT_STR);
   OBNDRV(&tpclda,&curp,":c_first",c_first,SIZ(c_first),SQLT_STR);
   OBNDRV(&tpclda,&curp,":c_middle",c_middle,SIZ(c_middle),SQLT_STR);

OBNDRV(&tpclda,&curp,":c_street_1",c_street_1,SIZ(c_street_1),SQLT_STR);

OBNDRV(&tpclda,&curp,":c_street_2",c_street_2,SIZ(c_street_2),SQLT_STR);
   OBNDRV(&tpclda,&curp,":c_city",c_city,SIZ(c_city),SQLT_STR);
   OBNDRV(&tpclda,&curp,":c_state",c_state,SIZ(c_state),SQLT_STR);
   OBNDRV(&tpclda,&curp,":c_zip",c_zip,SIZ(c_zip),SQLT_STR);
   OBNDRV(&tpclda,&curp,":c_phone",c_phone,SIZ(c_phone),SQLT_STR);
   OBNDRV(&tpclda,&curp,":c_since",c_since,SIZ(c_since),SQLT_STR);
   OBNDRV(&tpclda,&curp,":c_credit",c_credit,SIZ(c_credit)-1,SQLT_CHR);

OBNDRV(&tpclda,&curp,":c_credit_lim",ADR(c_credit_lim),SIZ(c_credit_lim),
          SQLT_FLT);
   OBNDRV(&tpclda,&curp,":c_discount",ADR(c_discount),SIZ(c_discount),
          SQLT_FLT);

OBNDRV(&tpclda,&curp,":c_balance",ADR(c_balance),SIZ(c_balance),SQLT_FLT);
   OBNDRV(&tpclda,&curp,":c_data",c_data,SIZ(c_data),SQLT_STR);
   OBNDRV(&tpclda,&curp,":h_date",h_date,SIZ(h_date),SQLT_STR);
   OBNDRV(&tpclda,&curp,":retry",ADR(retries),SIZ(retries),SQLT_INT);

   return (0);

}



plpay ()

{

   OEXEC(&tpclda,&curp);

   return (0);

}



void plpaydone ()

{

   if (oclose (&curp))
      errrpt (&tpclda, &curp);

}
```

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 (using PL/SQL stored procedure) of
 |    STOCK LEVEL transaction in TPC-C benchmark.
 +=========================================================================*/

#include "tpcc.h"
#include "tpccpl.h"


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



plstoinit ()

{

   text stmbuf[1024];

   OOPEN(&tpclda,&curs);

   sprintf ((char *) stmbuf, SQLTXT);
   OPARSE(&tpclda,&curs,stmbuf,NA,FALSE,VER7);

   /* bind variables */

   OBNDRV(&tpclda,&curs,":w_id",ADR(w_id),SIZ(w_id),SQLT_INT);
   OBNDRV(&tpclda,&curs,":d_id",ADR(d_id),SIZ(d_id),SQLT_INT);

OBNDRV(&tpclda,&curs,":threshold",ADR(threshold),SIZ(threshold),SQLT_INT);
```

```
OBNDRV(&tpclda,&curs,":low_stock",ADR(low_stock),SIZ(low_stock),SQLT_INT);

   return (0);

}



plsto ()

{

   OEXEC(&tpclda,&curs);

   return (0);

}



void plstodone ()

{

   if (oclose (&curs))
      errrpt (&tpclda, &curs);

}
```

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.
 +=================================================================*/

#include <stdio.h>
#include <unistd.h>
#include <sys/time.h>
#include "tpcc.h"
#include "tpcc_info.h"
#include "tpccpl.h"
#ifdef TUX
#include <userlog.h>
#endif
```

```
#define SQLTXT "alter session set isolation_level = serializable"

FILE *lfp;
FILE *fopen ();
double gettime ();
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;

ldadef tpclda;
csrdef curi;
csrdef curs;
csrdef curd;
csrdef curo;
csrdef curp;
csrdef curn, curn1, curn2, curn3[10], curn4;
unsigned long tpchda[256];

/* 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;
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];

/* for payment transaction */

int c_w_id;
int c_d_id;
float 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];
char c_since[11];
char c_credit[3];
double c_credit_lim;
float c_discount;
char c_data[201];
char h_date[20];

/* for new order transaction */

int nol_i_id[15];
int nol_supply_w_id[15];
int nol_quantity[15];
float 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];
float i_price[15];
int status;
static double getelapsed(long);

#ifdef NULL_TRANS

void fill_newo(struct NEWO_DATA *);
void fill_pmt(struct PMT_DATA *);
void fill_sl(struct STKL_DATA *);
void fill_os(struct ORDS_DATA *);
void fill_dl(struct DVRY_DATA *);
#endif


errrpt (lda, cur)

ldadef *lda;
csrdef *cur;

{

   text msg[2048];

   if (cur->rc) {
      oerhms (lda, cur->rc, msg, 2048);
#ifdef TUX
      userlog ("Error in TPC-C server %d: %s\n", proc_no, msg);
#else
      fprintf (stderr, "Error in TPC-C server %d: %s\n", proc_no, msg);
#endif
   }
   if ((cur->rc == DEADLOCK) || (cur->rc == SNAPSHOT_TOO_OLD))
      return (RECOVERR);
   else
      return (IRRECERR);

}



TPCexit ()

{

#ifndef NULL_TRANS
   if (new_init) {
      plnewdone();
      new_init = 0;
   }
   if (pay_init) {
      plpaydone();
      pay_init = 0;
   }
   if (ord_init) {
      plorddone();
      ord_init = 0;
   }
   if (del_init) {
      pldeldone();
      del_init = 0;
   }
   if (sto_init) {
      plstodone();
      sto_init = 0;
   }

   /* log off */

   if (logon) {
      if (ologof (&tpclda))
#ifdef TUX
         userlog ("Error in TPC-C server %d: Failed to log off\n",
                  proc_no);
#else
         fprintf (stderr, "Error in TPC-C server %d: Failed to log off\n",
                  proc_no);
#endif
      logon = 0;
   }
#endif

   if (lfp) {
      fclose (lfp);
      lfp = NULL;
   }

}
```

```
TPCinit (id, uid)

int id;
char *uid;

{

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

   proc_no = id;
   sprintf (filename, "tpcc_%ld.del", getpid());
   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);
   }
/** NULL_TRANS code added to test the 3 tier system with out using the DB

04/30/96   Latha ***/

#ifdef NULL_TRANS

    new_init = 1;
    pay_init = 1;
    ord_init = 1;
    del_init = 1;
    sto_init = 1;
#else /* actual code * 04/30/96

   /* log on to Oracle */

   if (orlon (&tpclda, (ub1 *) tpchda, (text *) uid, -1, (text *) 0, -1,
0)) {
#ifdef TUX
      userlog ("Error in TPC-C server %d: Failed to log on\n", proc_no);
#else
      fprintf (stderr, "Error in TPC-C server %d: Failed to log on\n",
proc_no);
#endif
      errrpt (&tpclda, &tpclda);
      return (-1);
   }

   /* turn off auto-commit */

   if (ocof (&tpclda)) {
      errrpt (&tpclda, &tpclda);
      ologof (&tpclda);
      return (-1);
   }

   /* run all transaction in serializable mode */
/* comment removed on 03/25/96 */
   if (oopen (&curi, &tpclda, (text *) 0, NA, NA, (text *) 0, NA)) {
      errrpt (&tpclda, &curi);
      ologof (&tpclda);
      return (-1);
   }
   sprintf ((char *) stmbuf, SQLTXT);
   if (oparse (&curi, stmbuf, (sb4) NA, FALSE, (ub4) VER7)) {
      errrpt (&tpclda, &curi);
      oclose (&curi);
      ologof (&tpclda);
      return (-1);
   }
   if (oexec (&curi)) {
      errrpt (&tpclda, &curi);
      orol (&tpclda);
      oclose (&curi);
      ologof (&tpclda);
      return (-1);
   }
   if (oclose (&curi))
      errrpt (&tpclda, &curi);
/* comment removed on 03/35/96 */
   logon = 1;

   if (plnewinit ()) {
      TPCexit ();
      return (-1);
   }
   else
      new_init = 1;

   if (plpayinit ()) {
      TPCexit ();
      return (-1);
   }
   else
      pay_init = 1;

   if (plordinit ()) {
      TPCexit ();
      return (-1);
   }
   else
      ord_init = 1;

   if (pldelinit ()) {
      TPCexit ();
      return (-1);
   }
   else
      del_init = 1;

   if (plstoinit ()) {
      TPCexit ();
      return (-1);
   }
   else
      sto_init = 1;

#endif /* end of else NULL_TRANS */

   return (0);
```

```
}


TPCnew (str)

struct NEWO_DATA *str;

{

   int i;

#ifdef NULL_TRANS
       fill_newo(str);
#else

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

        for(i=str->o_ol_cnt; i<15; i++){
         nol_i_id[i] = 0;
         nol_supply_w_id[i]= 0;
         nol_quantity[i] = 0;
        }
        if(plnew()) {
               str->header.returncode=-1;
               return(-1);
        }else
               str->header.returncode=COMMIT_NEWO; /*Changed from 0 to
                      COMMIT_NEWO since do_tpcc.c expects this. 04/30/96
*/

   str->o_id = o_id;
   str->o_ol_cnt = o_ol_cnt;
   strncpy (str->c_last, c_last, 17);
   strncpy (str->c_credit, c_credit, 3);
   str->c_discount = c_discount;
   str->w_tax = w_tax;
   str->d_tax = d_tax;
   strncpy (str->new_date, o_entry_d, 20);
   str->total = total_amount;
   for (i = 0; i < o_ol_cnt; i++) {
      strncpy (str->ol_table[i].name_i, i_name[i], 25);
      str->ol_table[i].s_quantity = s_quantity[i];
      str->ol_table[i].brand_generic = brand_gen[i];
      str->ol_table[i].price = i_price[i];
      str->ol_table[i].ol_amount = nol_amount[i];
   }
        str->items_valid = (status ? 0:1);
        /* Original Line str->header.returncode = (status ? 1:0); */
        str->header.returncode = (status ? 1:COMMIT_NEWO); /*Changed from
0 to
               COMMIT_NEWO since do_tpcc.c expects this. 04/30/96 */
#endif /*else end for NULL_TRANS */
   return (0);

}



TPCpay (str)

struct PMT_DATA *str;

{

#ifdef NULL_TRANS /* don't do the transaction */

       fill_pmt(str);
       str->header.returncode = 0;
#else /* Do transaction, else  NULL_TRANS */

   w_id = str->w_id;
   d_id = str->d_id;
   c_w_id = str->c_w_id;
   c_d_id = str->c_d_id;
   h_amount = str->h_amount;
   bylastname = str->byname;
   if (bylastname) {
      c_id = 0;
      strncpy (c_last, str->c_last, 17);
   }
   else {
      c_id = str->c_id;
      strcpy (c_last, " ");
   }
   retries = 0;


        if(plpay()) {

               str->header.returncode = -1;
               return(-1);
        }else

               str->header.returncode = 0;

   strncpy (str->w_street_1, w_street_1, 21);
   strncpy (str->w_street_2, w_street_2, 21);
   strncpy (str->w_city, w_city, 21);
   strncpy (str->w_state, w_state, 3);
   strncpy (str->w_zip, w_zip, 10);
   strncpy (str->d_street_1, d_street_1, 21);
   strncpy (str->d_street_2, d_street_2, 21);
   strncpy (str->d_city, d_city, 21);
   strncpy (str->d_state, d_state, 3);
   strncpy (str->d_zip, d_zip, 10);
   str->c_id = c_id;
   strncpy (str->c_first, c_first, 17);
   strncpy (str->c_middle, c_middle, 3);
   strncpy (str->c_last, c_last, 17);
   strncpy (str->c_street_1, c_street_1, 21);
   strncpy (str->c_street_2, c_street_2, 21);
   strncpy (str->c_city, c_city, 21);
```

```
   strncpy (str->c_state, c_state, 3);
   strncpy (str->c_zip, c_zip, 10);
   strncpy (str->c_phone, c_phone, 17);
   strncpy (str->c_date, c_since, 11);
   strncpy (str->c_credit, c_credit, 3);
   str->c_credit_lim = c_credit_lim;
   str->c_discount = c_discount;
   str->c_balance = c_balance;
   strncpy (str->c_data, c_data, 201);
   strncpy (str->pay_date, h_date, 20);
#endif /* end of NULL_TRANS 04/30/96 */

   return (0);

}



TPCord (str)

struct ORDS_DATA *str;

{

   int i;
#ifdef NULL_TRANS /* Don't do Transaction */

        fill_os(str);
        str->header.returncode = 0;
#else /* NULL_TRANS */
   w_id = str->w_id;
   d_id = str->d_id;
   bylastname = str->byname;
   if (bylastname) {
      c_id = 0;
      strncpy (c_last, str->c_last, 17);
               c_last[16]=0;
               for(i=15;((i>=0)&&(c_last[i] ==' '));i++)
                              c_last[i] = '\0';
   }
   else {
      c_id = str->c_id;
      strcpy (c_last, " ");
   }
   retries = 0;

        if(plord()){
               str->header.returncode = -1;
               return(-1);
        }else
               str->header.returncode = 0;

   str->c_id = c_id;
   strncpy (str->c_last, c_last, 17);
   strncpy (str->c_first, c_first, 17);
   strncpy (str->c_middle, c_middle, 3);
   str->c_balance = c_balance;
   str->o_id = o_id;
   strncpy (str->cur_date, o_entry_d, 20);
   str->o_carrier_id = o_carrier_id;
   str->o_ol_cnt = o_ol_cnt;
   for (i = 0; i < o_ol_cnt; i++) {
      ol_delivery_d[i][10] = '\0';
      str->ol_table[i].ol_supply_w_id = ol_supply_w_id[i];
      str->ol_table[i].ol_i_id = ol_i_id[i];
      str->ol_table[i].ol_quantity = ol_quantity[i];
      str->ol_table[i].ol_amount = ol_amount[i];
      strncpy (str->ol_table[i].delivery_date, ol_delivery_d[i], 11);
   }
#endif /* NULL_TRANS end */
   return (0);

}



TPCdel (str)

struct DVRY_DATA *str;

{

   double tr_end;
        int skipped;
   int i;
        double end_process, start_queue, end_queue;

#ifdef NULL_TRANS

        fill_dl(str);
        str->header.returncode = 0;
#else /* NULL_TRANS */
        start_queue = str->start_queue - str->cltinit;
        end_queue = getelapsed(str->cltinit);

   w_id = str->w_id;
   o_carrier_id = str->carrier_id;
   retries = 0;

        if(pldel()) {
               str->header.returncode = -1;
               return(-1);
        }else
               str->header.returncode = 0;

        end_process = getelapsed(str->cltinit);

        skipped = 0;
        for(i=0; i<10; i++){

               if(del_o_id[i] <= 0) {
                      skipped++;
#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, "%4d %3d %.3f\t%.3f\t%.3f\t%4d %2d", (10 - skipped),
skipped,
                        start_queue, end_queue, end_process, w_id,
o_carrier_id);
        for(i = 0; i < 10; i++) {
                fprintf(lfp, "%2d %d", i+1, del_o_id[i]);
        }
    fprintf (lfp, "\n", retries);
        fflush(lfp);
#endif /* NULL_TRANS end */
    return (0);

}



TPCsto (str)

struct STKL_DATA *str;

{

#ifdef NULL_TRANS

        fill_sl(str);
        str->header.returncode = 0;
#else /* NULL_TRANS */

    w_id = str->w_id;
    d_id = str->stkl_d_id;
    threshold = str->threshold;
    retries = 0;

        if(plsto()) {
                str->header.returncode = -1;
                return(-1);
        }else
                str->header.returncode = 0;

        str->stock_count = low_stock;
#endif /*NULL_TRANS */

    return (0);

}

static double getelapsed(cltinit)
long cltinit;
{

        struct timeval now;
        struct timezone tp;

        gettimeofday(&now, &tp);
        return(double)((double)(now.tv_sec - cltinit) +
                        (now.tv_usec/1000000.0));
}

#ifdef NULL_TRANS

void fill_newo(newo_dataptr)
struct NEWO_DATA *newo_dataptr;
{
        int i = 0;

   newo_dataptr->o_id = ORDER_ID_LB;
   newo_dataptr->o_ol_cnt = N_O_OL_CNT;
   strncpy (newo_dataptr->c_last, N_C_LAST, 17);
   strncpy (newo_dataptr->c_credit, N_C_CREDIT, 3);
   newo_dataptr->c_discount = N_C_DISCOUNT;
   newo_dataptr->w_tax = N_W_TAX;
   newo_dataptr->d_tax = N_D_TAX;
   strncpy (newo_dataptr->new_date, N_NEW_DATE, 20);
   newo_dataptr->total = N_TOTAL_AMOUNT;
           for(i=0; i<newo_dataptr->o_ol_cnt; i++){
                   newo_dataptr->ol_table[i].s_quantity = S_QTY_LB;
                   strcpy(newo_dataptr->ol_table[i].name_i, I_NAME_LB);

              newo_dataptr->ol_table[i].brand_generic =  BRAND_GENERIC_LB;
                      newo_dataptr->ol_table[i].price = PRICE_LB;
                   newo_dataptr->ol_table[i].ol_amount = OL_AMOUNT_LB;
           }
       newo_dataptr->items_valid = 1;
       newo_dataptr->header.returncode = COMMIT_NEWO;


}
#endif

#ifdef NULL_TRANS

void fill_pmt(pmt_dataptr)
struct PMT_DATA *pmt_dataptr;
{

        pmt_dataptr->header.returncode = 0;
        strcpy(pmt_dataptr->pay_date , PAY_DATE_LB);
        strcpy(pmt_dataptr->w_name , W_NAME_LB);
        strcpy(pmt_dataptr->w_street_1 , W_STREET_1_LB);
        strcpy(pmt_dataptr->w_street_2 , W_STREET_2_LB);
        strcpy(pmt_dataptr->w_city , W_CITY_LB);
        strcpy(pmt_dataptr->w_state , W_STATE_LB);
        strcpy(pmt_dataptr->w_zip , W_ZIP_LB);

        strcpy(pmt_dataptr->d_name , D_NAME_LB);
        strcpy(pmt_dataptr->d_street_1 , D_STREET_1_LB);
        strcpy(pmt_dataptr->d_street_2 , D_STREET_2_LB);
        strcpy(pmt_dataptr->d_city , D_CITY_LB);
        strcpy(pmt_dataptr->d_state , D_STATE_LB);
        strcpy(pmt_dataptr->d_zip , D_ZIP_LB);

       pmt_dataptr->c_id = C_ID_LB;
        strcpy(pmt_dataptr->c_first , C_FIRST_LB);
        strcpy(pmt_dataptr->c_middle , C_MIDDLE_LB);
        strcpy(pmt_dataptr->c_last , C_LAST_LB);
        strcpy(pmt_dataptr->c_phone , C_PHONE_LB);
        strcpy(pmt_dataptr->c_credit , C_CREDIT_LB);
        strcpy(pmt_dataptr->c_street_1 , C_STREET_1_LB);
        strcpy(pmt_dataptr->c_street_2 , C_STREET_2_LB);
        strcpy(pmt_dataptr->c_city , C_CITY_LB);
        strcpy(pmt_dataptr->c_state , C_STATE_LB);
        strcpy(pmt_dataptr->c_zip , C_ZIP_LB);
```

```
        pmt_dataptr->c_credit_lim = C_CREDIT_LIM_LB;
        pmt_dataptr->c_balance = C_BALANCE_LB;
        pmt_dataptr->c_discount = C_DISCOUNT_LB;
        pmt_dataptr->c_ytd_payment = C_YTD_PAYMENT_LB;
        pmt_dataptr->c_payment_cnt = C_PAYMENT_CNT_LB;

        strcpy(pmt_dataptr->c_date , C_DATE_LB);
        strcpy(pmt_dataptr->c_data , C_DATA_LB);
}
#endif


#ifdef NULL_TRANS

void fill_os(ords_dataptr)
struct ORDS_DATA *ords_dataptr;
{

        int i=0;


        ords_dataptr->header.returncode = 0;

        ords_dataptr->o_ol_cnt = 5;
       ords_dataptr->o_id = O_ID_LB;
        ords_dataptr->o_carrier_id = O_CARRIER_ID_LB;
       ords_dataptr->c_id = 99;
        strcpy(ords_dataptr->c_last, C_LAST_LB);
        strcpy(ords_dataptr->c_first, C_FIRST_LB);
        strcpy(ords_dataptr->c_middle, C_MIDDLE_LB);
        strcpy(ords_dataptr->cur_date, CUR_DATE_LB);
        ords_dataptr->c_balance = C_BALANCE_LB;

        for(i=0; i<ords_dataptr->o_ol_cnt; i++){
               ords_dataptr->ol_table[i].ol_i_id = OL_I_ID_LB;
               ords_dataptr->ol_table[i].ol_supply_w_id =
OL_SUPPLY_W_ID_LB;
               ords_dataptr->ol_table[i].ol_quantity = OL_QUANTITY_LB;
               ords_dataptr->ol_table[i].ol_amount = OL_AMOUNT_LB;
               strcpy(ords_dataptr->ol_table[i].delivery_date,
DELIVERY_DATE_LB );
        }
}
#endif

#ifdef NULL_TRANS

void fill_sl(stkl_dataptr)
struct STKL_DATA *stkl_dataptr;
{

        stkl_dataptr->header.returncode = 0;

        stkl_dataptr->stock_count = STOCK_COUNT_LB;
}
#endif

#ifdef NULL_TRANS

void fill_dl(str)
struct DVRY_DATA * str;
{

}
#endif
```

tpccsvr.c

```
#ifdef RCSID
static char *RCSid =
   "$Header: tpccsvr.c 7030100.1 95/07/19 15:39:28 plai Generic<base> $
Copyr (c) 1995 Oracle";
#endif /* RCSID */

/*+---------------------------------------------------------------------+
  |        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA          |
  |                    OPEN SYSTEMS PERFORMANCE GROUP                   |
  |                         All Rights Reserved                         |
 +---------------------------------------------------------------------+
  | FILENAME
  |    tpccsvr.c
  | DESCRIPTION
  |    Tuxedo server for TPC-C.
 +---------------------------------------------------------------------*/

#include <stdio.h>
#include "tpcc.h"
#include "tpcc_info.h"
#include <atmi.h>
#include <userlog.h>



struct NEWO_DATA *newinfo;
struct PMT_DATA *payinfo;
struct ORDS_DATA *ordinfo;
struct DVRY_DATA *delinfo;
struct STKL_DATA *stoinfo;



tpsvrinit (argc, argv)

int argc;
char *argv[];

{
 /*------- */

/* The following are all defined as macros
    struct servent *servent;*/
    char *uid          = "tpcc/tpcc";
    char  c;
    int   id;

/* Startup... */
    /*
```

```
     * search for the options
     *    "-n" server number
     *    "-S" server program
     *    "-h" host name
     *    These are specified in "CLOPT" which is specified in UBBconfig
     */
    while ((c = getopt(argc, argv, "n:S")) != EOF) {
        switch(c) {
       case 'n':
            id = atoi(optarg);
           break;
        case 'S':
           uid=optarg;
           break;
       }
    }
/*------*/
/*    int id;
   char *uid;

   if (argc >= 2) {
      id = atoi (argv[argc - 2]);
      uid = argv[argc - 1];
added for debugging */
   if (argc >= 2) {
userlog(" user id passed on %d %s \n",id,uid);
      return (TPCinit (id, uid));
   }
   else {
      userlog ("Error: not enough arguments to tpsvrinit\n");
      return (-1);
   }


}



void tpsvrdone ()

{

   TPCexit ();

}



NEWO_SVC (msg)

TPSVCINFO *msg;

{

   newinfo = (struct NEWO_DATA *) msg->data;
   if (TPCnew (newinfo))
      tpreturn (TPFAIL, newinfo->header.returncode,(char *) newinfo,
                sizeof (struct NEWO_DATA), 0);
   else
      tpreturn (TPSUCCESS, newinfo->header.returncode,(char *) newinfo,
                sizeof (struct NEWO_DATA), 0);

}



PMT_SVC (msg)

TPSVCINFO *msg;

{

   payinfo = (struct PMT_DATA *) msg->data;
   if (TPCpay (payinfo))
      tpreturn (TPFAIL, payinfo->header.returncode,(char *) payinfo,
                sizeof (struct PMT_DATA), 0);
   else
      tpreturn (TPSUCCESS, payinfo->header.returncode,(char *) payinfo,
                sizeof (struct PMT_DATA), 0);

}



ORDS_SVC (msg)

TPSVCINFO *msg;

{

   ordinfo = (struct ORDS_DATA *) msg->data;
   if (TPCord (ordinfo))
      tpreturn (TPFAIL, ordinfo->header.returncode,(char *) ordinfo,
                sizeof (struct ORDS_DATA), 0);
   else
      tpreturn (TPSUCCESS, ordinfo->header.returncode,(char *) ordinfo,
               sizeof (struct ORDS_DATA), 0);

}



DVRY_SVC (msg)

TPSVCINFO *msg;

{

   delinfo = (struct DVRY_DATA *) msg->data;
   if (TPCdel (delinfo))
      tpreturn (TPFAIL, delinfo->header.returncode, NULL, 0, 0);
   else
      tpreturn (TPSUCCESS, delinfo->header.returncode, NULL, 0, 0);

}



STKL_SVC (msg)

TPSVCINFO *msg;
```

```
{

   stoinfo = (struct STKL_DATA *) msg->data;
   if (TPCsto (stoinfo))
      tpreturn (TPFAIL, stoinfo->header.returncode, (char *)stoinfo,
                sizeof (struct STKL_DATA), 0);
   else
      tpreturn (TPSUCCESS, stoinfo->header.returncode,(char *) stoinfo,
       sizeof (struct STKL_DATA), 0);

}
```

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

#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

#include <oratypes.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 int TPCexit ();
extern int TPCdumpinit ();
extern int TPCdumpnew ();
extern int TPCdumppay ();
extern int TPCdumpord ();
extern int TPCdumpdel ();
extern int TPCdumpsto ();
extern int TPCdumpexit ();


/* Error codes */

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

#endif
```

tpcc_info.h

```
/*
 * $Header: tpcc_info.h 7030100.1 95/07/19 15:11:37 plai Generic<base> $
Copyr (c) 1995 Oracle
 */
/*=====================================================================+
 |         Copyright (c) 1995  Oracle Corp, Redwood Shores, CA         |
 |                    OPEN SYSTEMS PERFORMANCE GROUP                   |
 |                         All Rights Reserved                         |
 +=====================================================================+
 | FILENAME
 |    tpcc_info.h
 | DESCRIPTION
 |    Include file for TPC-C benchmark programs.
 +=====================================================================*/

#ifndef TPCC_INFO_H
#define TPCC_INFO_H

#include "utils.h"

/*
* Extracted from databuf.h
*/

#define I_NAME_LEN     24
#define I_DATA         50
#define W_NAME_LEN     10
#define ADDR_LEN       20
#define STATE_LEN      2
#define ZIP_LEN                9
#define DIST_INFO_LEN  24
#define S_DATA_LEN     50
#define D_NAME_LEN     10
#define H_DATA_LEN     24
#define CARRIER_LEN    2

#define C_LAST_LEN     17
#define C_MID_LEN      2
#define PHONE_LEN      16
#define CREDIT_LEN     2

#define C_DATA_LEN  500
#define BC_DATA_LEN  23
```

```
#define DATETIME_LEN   19

#define MAX_OL         15
#define C_LAST_LEN     17



#ifdef NULL_TRANS
/* Constants for the LoopBack for neworder -- Roopa */
#define ORDER_ID_LB    100
#define W_TAX_LB       9.99
#define D_TAX_LB       9.99
#define TOTAL_LB       999.9
#define C_DISCOUNT_LB  9.99
#define NEW_DATE_LB    "ddmmyy"
#define C_CREDIT_LB    "xx"
#define I_NAME_LB      "xxxx"
#define BRAND_GENERIC_LB      'x'
#define S_QTY_LB       99
#define PRICE_LB       99.9
#define OL_AMOUNT_LB   999.9
#define ITEMS_VALID_LB 1
/* Constants added for Loop back for 3 tier with Tuxedo for New order
  -- Latha 04/30 */
#define N_O_OL_CNT     9
#define N_C_LAST       "17_XXXXXXXXXXXXXXX"
#define N_C_CREDIT     "3_9"
#define N_C_DISCOUNT   99
#define N_W_TAX                99
#define N_D_TAX                99
#define N_NEW_DATE     "20_12345678901234567"
#define N_TOTAL_AMOUNT 9999
#define N_I_NAME       "25_XXXXXXXXXXXXXXXXXXXXXX"
#define N_S_QUANTITY   99
#define N_BRAND_GEN    '3'
#define N_PRICE        99
#define N_NOL_AMOUNT   99
#define N_ITEMS_VALID  0
#define COMMIT_NEWO    99  /* Added for 3 tier loop backup on seeing
                              ex_trans.c fill_newo func. Latha 04/30 */

/* Constants for the LoopBack for payment -- Roopa */
#define PAY_DATE_LB    "ddmmyy"
#define W_NAME_LB      "xxxx"
#define W_STREET_1_LB  "xxxx"
#define W_STREET_2_LB  "xxxx"
#define W_CITY_LB      "xxxx"
#define W_STATE_LB     "xxxx"
#define W_ZIP_LB       "xxxx"
#define D_NAME_LB      "xxxx"
#define D_STREET_1_LB  "xxxx"
#define D_STREET_2_LB  "xxxx"
#define D_CITY_LB      "xxxx"
#define D_STATE_LB     "xxxx"
#define D_ZIP_LB       "xxxx"
#define C_MIDDLE_LB    "xxxx"
#define C_PHONE_LB     "xxxx"
#define C_STREET_1_LB  "xxxx"
#define C_STREET_2_LB  "xxxx"
#define C_CITY_LB      "xxxx"
#define C_STATE_LB     "xxxx"
#define C_ZIP_LB       "xxxx"
#define C_CREDIT_LIM_LB       9999.99
#define C_BALANCE_LB   99.99
#define C_DISCOUNT_LB  9.99
#define C_YTD_PAYMENT_LB      99.99
#define C_PAYMENT_CNT_LB      9
#define C_DATE_LB      "ddmmyy"
#define C_DATA_LB      "xxxxxxxxxxxxxxxxxxxxxxxxx"
#define C_ID_LB               99

/* Loopback Constants for Order Status -- Roopa */
#define O_ID_LB        99
#define O_CARRIER_ID_LB       9
#define C_LAST_LB      "xxxx"
#define C_FIRST_LB     "xxxx"
#define CUR_DATE_LB    "ddmmyy"
#define OL_I_ID_LB     99
#define OL_SUPPLY_W_ID_LB     99
#define OL_QUANTITY_LB 99
#define DELIVERY_DATE_LB      "ddmmyy"

/* Loopback Constants for Stock Level -- Roopa */
#define STOCK_COUNT_LB 99
#else
#define COMMIT_NEWO    99  /* Added for 3 tier loop backup on seeing
                              ex_trans.c fill_newo func. Latha 04/30 */

#endif  /* NULL_TRANS endif */


struct data_header {
    short int  dtype;
    short int  returncode;
    int        sql_code;
    int        isam_code;
};

struct OL_TABLE {
        short int  ol_supply_w_id;
        short int  ol_quantity;
        short int  s_quantity;
              int  ol_i_id;
        char       name_i[I_NAME_LEN + 1];
        char       brand_generic;
        double     price;
        double     ol_amount;
};

struct OL_TABLE2 {
              int  ol_i_id;
        short int  ol_supply_w_id;
        short int  ol_quantity;
        double     ol_amount;
        char       delivery_date[20];
};

struct NEWO_DATA {
       struct data_header header;
       short int       w_id;
       short int       d_id;
```

```
        int       c_id;
        short int       o_ol_cnt;
        short int       o_all_local;
        short int       items_valid; /* true if all valid */
        int             o_id;
        double          w_tax;
        double          d_tax;
        double          total;
        double          c_discount;
        char            new_date[DATETIME_LEN + 1];
        char            c_last[C_LAST_LEN];
        char            c_credit[CREDIT_LEN + 1];
        struct OL_TABLE ol_table[MAX_OL];
};

struct PMT_DATA {
        struct data_header header;
        short int w_id;
        short int d_id;
        int c_id;
        short int c_w_id;
        short int c_d_id;
        short int byname;
        double    h_amount;
        char      pay_date[20];

        char      w_name[11];
        char      w_street_1[21];
        char      w_street_2[21];
        char      w_city[21];
        char      w_state[3];
        char      w_zip[10];

        char      d_name[11];
        char      d_street_1[21];
        char      d_street_2[21];
        char      d_city[21];
        char      d_state[3];
        char      d_zip[10];

        char      c_first[17]; /* was C_LAST_LEN already includes +1 */
        char      c_middle[3];
        char      c_last[17];
        char      c_phone[17];
        char      c_credit[3];
        char      c_street_1[21];
        char      c_street_2[21];
        char      c_city[21];
        char      c_state[3];
        char      c_zip[10];
        double    c_credit_lim;
        double    c_balance;
        double    c_discount;
        double    c_ytd_payment;
        short int c_payment_cnt;
        char      c_date[20];
        char      c_data[200];
};

struct ORDS_DATA {
        struct data_header header;
        short int w_id;
        short int d_id;
        int c_id;
              int o_id;
        short int o_ol_cnt;
        short int byname;
        short     o_carrier_id;
        char      c_last[17];
        char      c_first[17];
        char      c_middle[3];
        char      cur_date[20];
        double    c_balance;
        struct OL_TABLE2 ol_table[15];
};

struct DVRY_DATA {
        struct data_header header;
        short int w_id;
        short int carrier_id;
        long  cltinit;
        double start_queue;
};

struct STKL_DATA {
        struct data_header header;
        short int w_id;
        short int stkl_d_id;
        short int threshold;
        int       stock_count;
};

#endif
```

tpccpl.h

```
/*
 * $Header: tpccpl.h 7030100.1 96/04/02 18:03:35 plai Generic<base> $
Copyr (c) 1994 Oracle
 */
/*+=================================================================+
 |        Copyright (c) 1994  Oracle Corp, Redwood Shores, CA       |
 |                    OPEN SYSTEMS PERFORMANCE GROUP                |
 |                         All Rights Reserved                      |
 +=================================================================+
 | FILENAME
 |    tpccpl.h
 | DESCRIPTION
 |    Header file for TPC-C transactions in PL/SQL.
 +=================================================================*/

#ifndef TPCCPL_H
#define TPCCPL_H

#include <stdio.h>

#define DELRT 80.0

extern int plnewinit ();
extern int plpayinit ();
extern int plordinit ();
```

```
extern int pldelinit ();
extern int plstoinit ();

extern int plnew ();
extern int plpay ();
extern int plord ();
extern int pldel ();
extern int plsto ();

extern void plnewdone ();
extern void plpaydone ();
extern void plorddone ();
extern void pldeldone ();
extern void plstodone ();

extern errrpt ();

extern FILE *lfp;
extern FILE *fopen ();
extern int proc_no;
extern int doid[];

extern ldadef tpclda;
extern csrdef curs;
extern csrdef curd;
extern csrdef curo;
extern csrdef curp;
extern csrdef curn, curn1, curn2, curn3[10], curn4;
extern unsigned long tpchda[];


/* 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 char 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 float ol_amount[15];
extern char ol_delivery_d[15][11];

/* for payment transaction */

extern int c_w_id;
extern int c_d_id;
extern float 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 char c_since[11];
extern char c_credit[3];
extern double c_credit_lim;
extern float c_discount;
extern char c_data[201];
extern char 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 float 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 float i_price[15];
extern int status;



#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 OBNDRV(lda,cursor,sqlvar,progv,progvl,ftype)\
    if
(obndrv((cursor),(text*)(sqlvar),NA,(ub1*)(progv),(progvl),(ftype),NA,\
       (sb2 *)0, (text *)0, NA, NA))\
       {errrpt(lda,cursor);return(-1);}\
    else\
       DISCARD 0

#define OBNDRA(lda,cursor,sqlvar,progv,progvl,ftype,indp,alen,arcode)\
    if
(obndra((cursor),(text*)(sqlvar),NA,(ub1*)(progv),(progvl),(ftype),NA,\
       (indp),(alen),(arcode),(ub4)0,(ub4*)0,(text*)0,NA,NA))\
       {errrpt(lda,cursor);return(-1);}\
    else\
       DISCARD 0

#define
OBNDRAA(lda,cursor,sqlvar,progv,progvl,ftype,indp,alen,arcode,ms,cs)\
    if
(obndra((cursor),(text*)(sqlvar),NA,(ub1*)(progv),(progvl),(ftype),NA,\
       (indp),(alen),(arcode),(ub4)(ms),(ub4*)(cs),(text*)0,NA,NA))\
       {errrpt(lda,cursor);return(-1);}\
    else\
       DISCARD 0

#define
ODEFIN(lda,cursor,pos,buf,bufl,ftype,scale,indp,fmt,fmtl,fmtt,rlen,rcode)\
    if (odefin((cursor),(pos),(ub1*)(buf),(bufl),(ftype),(scale),(indp),\
       (text*)(fmt),(fmtl),(fmtt),(rlen),(rcode)))\
       {errrpt(lda,cursor);return(-1);}\
    else\
       DISCARD 0

#define OEXFET(lda,cursor,nrows,cancel,exact)\
    if (oexfet((cursor),(nrows),(cancel),(exact)))\
       {if ((cursor)->rc == 1403) DISCARD 0; \
        else if (errrpt(lda,cursor)==RECOVERR) \
           {orol(lda);return(RECOVERR);} \
        else{orol(lda);return(-1);}}\
    else\
       DISCARD 0

#define OOPEN(lda,cursor)\
    if (oopen((cursor),(lda),(text*)0,NA,NA,(text*)0,NA))\
       {errrpt(lda,cursor);return(-1);}\
    else\
       DISCARD 0

#define OPARSE(lda,cursor,sqlstm,sqll,defflg,lngflg)\
    if (oparse((cursor),(sqlstm),(sb4)(sqll),(defflg),(ub4)(lngflg)))\
       {errrpt(lda,cursor);return(-1);}\
    else\
       DISCARD 0

#define OFEN(lda,cursor,nrows)\
    if (ofen((cursor),(nrows)))\
       {if (errrpt(lda,cursor)==RECOVERR) \
           {orol(lda);return(RECOVERR);} \
        else{orol(lda);return(-1);}}\
    else\
       DISCARD 0

#define OEXEC(lda,cursor)\
    if (oexec((cursor)))\
       {if (errrpt(lda,cursor)==RECOVERR) \
           {orol(lda);return(RECOVERR);} \
        else{orol(lda);return(-1);}}\
    else\
       DISCARD 0


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


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

#endif
```

Makefile

```
#
# $Header: prefix.mk 1.2 95/04/12 14:07:16 rdhoopar Osd<unix> $ prefix.mk
#
#
# prefix.mk - generic prefix file for individual Makefiles
#
# This file is combined by "Make.file" with the SOURCE/OBJECT
# file list and "suffix.mk" to create a "Makefile".
#
```

```
.SUFFIXES: .i .pc .ok .h .hok .d .scr .r .j .k

MAKE.FILE= Make.file > Makefile

# The macro definition for SRCHOME is pulled from the environment on most
# ports.  Ports that don't support this feature, need to set SRCHOME here.
# Same for the definitions needed for qa: WORK,HIST,T_COM
#

# for safety, set DIRS to nil         (it should be set by the product
#                                      specific makefiles, e.g.
fortran.mk)
DIRS= ""

# TEST environment setup scripts
TMSETUP=. $(SRCHOME)/test/qaenv.sh

OTSTSETUP=. $(SRCHOME)/test/oratst/qaenv.tst

# TEST environment script for Ora*TST 2.0

ORATSTQAENV= $(SRCHOME)/test/oratst2/qaenv.tst

# Symbol for specifing include directories.
I_SYM= -I

# Flag for entering a symbol as undefined in the symbol table
U_SYM= -u

# CD - Current directory
CD= .

# RCO - Remote check-out
RCO=no_such_checkout

# MAKE - make

MAKE= make $(MKFLAGS)

QAMAKE= $(MAKE)
# Dual universe machines may need to use 'att make' in order for make
# to pick up environment variables.

# MKFLAGS
MKFLAGS=

# AS - Assembler

AS= as

# ASFLAGS - Special flags to pass to assembler
# For the PMAX, ASFLAGS are defined after OPTIMIZE is defined.

ASFLAGS=

# CP0 - C PRE-preprocesser (pass 0)
CP0=cp0

# CPP - C Preprocessor
CPP=/lib/cpp

CPP=/usr/ccs/lib/acpp

#
# Include file paths - be sure to include $(I_SYM)
#

# BSDINCLUDE  - BSD include file path
BSDINCLUDE=

# Wollongong Include Directory

# C Preprocessor flags
#   INCLUDE - include directive for pre-compiler
INCLUDE= $(SRCINCLUDE) $(I_SYM).

# SPFLAGS - Special preprocessor flags
SPFLAGS=

#Our 3b4000 compiler only defines u3b15.

# LPFLAGS - Local preprocessor flags -  should be defined in local .mk
files
LPFLAGS=

# PFLAGS - All preprocessor flags
PFLAGS= $(INCLUDE) $(SPFLAGS) $(LPFLAGS)

# CC - C compiler
CC=cc

# CFLAGS - C Compiler flags:  Automatically used for "cc" options.
# CCFLAGS - flags for cc

CCFLAGS=

# GFLAG is used to turn on "-g" to get the required symbols for the
# debugger.
GFLAG=

CCFLAGS=-Xa -K PIC

OPTIMIZE=-O

# The CFLAGS macro line should look the same for every port.
# Note: the HP9000/800 requires -g to be before -O, otherwise -O will
override
# -g.
# $(QACCFLAGS) is needed by Q/A programs.
QACCFLAGS=

CFLAGS= $(GFLAG) $(OPTIMIZE) $(CDEBUG) $(CCFLAGS) $(QACCFLAGS) $(PFLAGS)
$(SHARED_CFLAG)

# C Compiler commands

MAKERO=
ROFLAGS=-c

PRECOMP=
```

```
MCS= ; mcs -d $*.o
COMPO= $(PRECOMP) $(CC) -c $(CFLAGS) $*.c $(MCS)

COMPD= $(PRECOMP) $(CC) -c $(CFLAGS) -DDEBUG $*.c
COMPS= $(PRECOMP) $(CC) -S $(CFLAGS) $*.c
ASRO= $(AS) $(ASROFLAGS) $*.s $(MAKERO)
COMPRO= $(PRECOMP) $(CC) $(ROFLAGS) $(CFLAGS) $*.c $(MAKERO)
COMPNO= $(PRECOMP) $(CC) -c $(PFLAGS) $(CCFLAGS) $*.c
COMPRONO= $(PRECOMP) $(CC) $(ROFLAGS) $(CDEBUG) $(CCFLAGS) $(PFLAGS) $*.c
$(MAKERONO)

COMPXA= $(PRECOMP) $(CC) -c  $(PFLAGS) -Xa $(OPTIMIZE) $*.c
COMPXANO= $(PRECOMP) $(CC) -c  $(PFLAGS) -Xa $*.c

LCOMPO= $(PRECOMP) $(CC) -c $(CFLAGS) $? ; mv `basename $@` $@
LCOMPD= $(PRECOMP) $(CC) -c $(CFLAGS) -DDEBUG $? ; mv `basename $@` $@
LCOMPS= $(PRECOMP) $(CC) -S $(CFLAGS) $? ; mv `basename $@` $@
LCOMPNO= $(PRECOMP) $(CC) -c $(PFLAGS) $(CCFLAGS) $? ; mv `basename $@` $@

# okpp - OraKit PreProcessor
OKPP=okpp
OKPPINCLUDES= -I.
OKPPFLAGS= -W0 -Oc $(OKPPINCLUDES)
OKPPC= $(OKPP) $(OKPPFLAGS) -F$*.c $*.ok
OKPPH= $(OKPP) $(OKPPFLAGS) -F$*.h $*.hok

# oksc - OraKit Script Compiler
OKSC=oksc
OKSCINCLUDES= -I. -I$(SRCHOME)/orakit/char/include -
I$(SRCHOME)/orakit/char/scripts
OKSCDEFINES=
OKSCFLAGS= $(OKSCINCLUDES) $(OKSCDEFINES)
OKSCR= $(OKSC) $(OKSCFLAGS) -o $*.r $*.scr

# Toolkit2 defines
TK2UIXIL=uixil

# FC - Fortran compiler

FC=f77

FC=lpifortran
FCFLAGS=-lowercase

# LD - Loader
LD= ld

LD=lpild

LIBHOME=$(SRCHOME)/lib

# LDFLAGS - Loader flags
LDPFLAGS= -o $@ -p -L$(LIBHOME)
LDGPFLAGS= -o $@ -pg -L$(LIBHOME)
LDFLAGS= -o $@ -L$(LIBHOME)

PREFCHAR= _

# Loader commands
LDCCOM= $(CC) $(GFLAG) $(CCFLAGS) $(LDFLAGS)
LDPCCOM= $(CC) $(CCFLAGS) $(LDPFLAGS)
LDGPCCOM= $(CC) $(CCFLAGS) $(LDGPFLAGS)
LDFCOM= $(FC) $(LDFLAGS)
LDRCOM= $(LD) -r -x

# Archive/Library commands
# AR - Archive (library) command
AR= ar

# Define ARLOCAL to force AR to use the local directory instead of /tmp.
# This is currently necessary on the PMAX because the libraries are
# huge, and /tmp is very small.

ARLOCAL=

# ARAPPEND - Quickly append file to archive
ARAPPEND= $(AR) q$(ARLOCAL)

# ARDELETE - Delete entry from archive
ARDELETE= $(AR) d$(ARLOCAL)

# ARREPLACE - Replace entry in archive
ARREPLACE= $(AR) r$(ARLOCAL)

# AREXTRACT - extract entry from archive
AREXTRACT= $(AR) x$(ARLOCAL)

# ARCREATE - Create archive
ARCREATE=$(AR) cr$(ARLOCAL)

# RANLIB - Optimize ordering in library (BSD)
RANLIBCMD=
RANLIB=
RANLIBL=

RANLIBCMD=ar ts

# LORDER - Logical ordering for library (SYS V)
LORDER=lorder

# TSORT - Topological sorting for library (SYS V)
TSORT=tsort

# Command (stream) to create archive

DOAR=$(ARCREATE) $@ $? $(RANLIB)

# Command to force objects into library

ARCHIVE=$(ARCREATE) $L $? $(RANLIBL)

# Command to force objects into library
FRESHEN=$(ARCREATE) $L $(OBJS) $(RANLIBL)

# GENSHLIB - Generate shared library from archive file
GENSHLIB=genshlib

# NM - Print symbols from archive/object file
NM= nm
NMUFLAGS= -u
NMDFLAGS= -gn
```

```
# FORMAT - Formatting command used for man pages.
FORMAT=nroff -man

# CLEAN - Define this to be 'cleanup' to remove objects after 'make
compile'.
#         Most 'suffix.mk' have compile: lib$(LIB).a $(CLEAN)
CLEAN=

# When the "make" bug is fixed, redefine CLEAN to be "cleanup" for
# SEQ_PSX.

# TESTX - Test for existence of executables.
TESTX= test -x

# TOUCH - Touch files, modify access date
TOUCH=touch

# Operating System (UNIX) commands
# SHELL - Command intrepreter to use
SHELL=/bin/sh

# OPR - Line printer filter (for backwards compatability)
OPR= lpr

# LP - Line Printer command
LP= lpr -p $(SRCS)

# the NeXT does not have lint. Use -Wall in cc instead

# LINT - Lint program
LINT=lint

# LFLAGS - Lint flags
LFLAGS= -bhpu

# Give the name of the directory that contains lint libraries
LINTDIR=$(SRCHOME)/lib

# Define the flag to use to produce lint output.
LINTOUT=-C

# Describe how to make a lint library from sources.  To the best of
# my knowledge, 'lint' isn't very friendly specifying where the resulting
# library should be placed.  Here, we always put the library in the file
# 'llib-lxx.ln' in the local directory, and move the result into the
target.
DOLINT=$(LINT) $(LFLAGS) $(PFLAGS) $(LINTOUT)xx $?; mv llib-lxx.ln $@

# Describe how to do an inter-module lint.
DOLINTLOAD=$(LINT) $(LFLAGS) $(PFLAGS) $? $(CLINT)

#
# Symbolic directory/library/miscellaneous name definitions for non-ORACLE
# libraries/etc.
#

EXOSLIBS=
LOCALLIBS=

OTHERLIBS=-lsocket -lnsl -lelf -lm

XLIBS=-lXt -lXmu -lX11 -lXaw -lXext

# Set this variable if your port needs special libraries for Trusted
ORACLE.
SECLIBS=

CLIBS= $(LOCALLIBS) $(EXOSLIBS) $(LIBBSD) $(OTHERLIBS) $(SECLIBS)
$(M6LIBS)

# Describe libraries to use when linting.  No '.a' files should appear
here.
CLINT= $(LOCALLIBS) $(EXOSLIBS) $(LIBBSD) $(OTHERLIBS)

CURLIBS= -lcurses

LU62LIBS=-llu62

#
# ORACLE directory macros.
#

NETHOME= $(SRCHOME)/sqlnet
NET2HOME= $(SRCHOME)/network
FORMSHOME= $(SRCHOME)/forms
MENUHOME= $(SRCHOME)/menu
FORMS30HOME= $(SRCHOME)/forms30
MENU5HOME= $(SRCHOME)/menu5
ORATERMHOME= $(SRCHOME)/oraterm
OKTHOME= $(SRCHOME)/orakit
OKTCHOME= $(SRCHOME)/orakit/char
OKTXHOME= $(SRCHOME)/orakit/x
OKTMHOME= $(SRCHOME)/orakit/mot
TK2HOME= $(SRCHOME)/tk2
TK2CHOME= $(SRCHOME)/tk2/uicm
TK2XHOME= $(SRCHOME)/tk2/uix
TK2MHOME= $(SRCHOME)/tk2/uimotif
OCOREHOME= $(SRCHOME)/ocore
BINHOME= $(SRCHOME)/bin
UTILHOME= $(SRCHOME)/utl
DBINHOME= $(SRCHOME)/dbin
DBSHOME= $(SRCHOME)/dbs
SQLPLUSHOME= $(SRCHOME)/sqlplus
SRWHOME= $(SRCHOME)/sqlreport
ALLY1HOME= $(SRCHOME)/mail/ally
ALLY2HOME= $(SRCHOME)/sqlreport/ally
MAILHOME= $(SRCHOME)/mail
CASEDICTHOME= $(SRCHOME)/dict50
CASEGENHOME= $(SRCHOME)/cgen20
CASEDESIGNHOME= $(SRCHOME)/cdes
QMXHOME= $(SRCHOME)/qmx
EASYSQLHOME= $(SRCHOME)/easysql
PROGRAPHHOME= $(SRCHOME)/prograph
OGRAPHHOME= $(SRCHOME)/ograph
SQLTRHOME= $(SRCHOME)/sqltr
DATALENSHOME= $(SRCHOME)/datalens
NLSWBHOME= $(SRCHOME)/nlswb
XAHOME= $(SRCHOME)/xa
MAKESHIPHOME=$(SRCHOME)/port/unix/makeship

#
```

```
# The following list should contain an entry for any library or object
module
# which is refered to in load lists outside the Makefile in its own
directory.
#

OSNTAB= $(LIBHOME)/osntab.o
OSNTABST= $(LIBHOME)/osntabst.o

LIBXA= $(LIBHOME)/libxa.a
LIBSQL= $(LIBHOME)/libsql.a

LIBSQLNET=$(LIBHOME)/libsqlnet.a
LLIBSQLNET=-lsqlnet

#
#
#

LIBFORMS= $(LIBHOME)/libforms.a
LIBMENU= $(LIBHOME)/libmenu.a
LIBFORMS30= $(LIBHOME)/libforms30.a
LIBFORMS30P= $(LIBHOME)/libforms30p.a
LIBFORMS30C= $(LIBHOME)/libforms30c.a
LIBFORMS30X= $(LIBHOME)/libforms30x.a
LIBFORMS30M= $(LIBHOME)/libforms30m.a
LIBMENU5= $(LIBHOME)/libmenu5.a
LIBMENU5P= $(LIBHOME)/libmenu5p.a
LIBMENU5C= $(LIBHOME)/libmenu5c.a
LIBMENU5X= $(LIBHOME)/libmenu5x.a
LIBMENU5M= $(LIBHOME)/libmenu5m.a
LIBORATERM= $(LIBHOME)/liboraterm.a
LIBOKT= $(LIBHOME)/libokt.a
LIBOKTC= $(LIBHOME)/liboktc.a
LIBOKTX= $(LIBHOME)/liboktx.a
LIBOKTM= $(LIBHOME)/liboktm.a
LIBTK2C= $(LIBHOME)/libtk2c.a
LIBTK2X= (Tk2 X toolkit is replaced by Motif toolkit in 2.0.7)
LIBTK2M= $(LIBTK2ML) $(LIBTK2P) $(LIBTK2ML)
LIBTK2ML= $(LIBHOME)/libtk2m.a
LIBTK2P= $(LIBHOME)/libtk2p.a
LIBTK2UC= $(LIBHOME)/libuc.a
LIBTK2REM= $(LIBHOME)/librem.a
LIBTK2ROS= $(LIBHOME)/libros.a
LIBTK2OT= $(LIBHOME)/libot.a
LIBTK2UT= $(LIBHOME)/libut.a
LIBTK2SL= $(LIBHOME)/libsl.a
LIBTK2RE= $(LIBHOME)/libre.a

# Here Define CASE*Dictionary libraries
LIBARCHV=$(CASEDICTHOME)/archv/libarchv.a
LIBCDL=$(CASEDICTHOME)/cdl/libcdl.a
LIBCDX=$(CASEDICTHOME)/cdx/libcdx.a
LIBCREAT=$(CASEDICTHOME)/creat/libcreat.a
LIBFEX=$(CASEDICTHOME)/fex/libfex.a
LIBFREZE=$(CASEDICTHOME)/freze/libfreze.a
LIBINDCD=$(CASEDICTHOME)/indcd/libindcd.a
LIBMSG=$(CASEDICTHOME)/msg/libmsg.a
LIBNVERS=$(CASEDICTHOME)/nvers/libnvers.a
LIBCDI=$(CASEDICTHOME)/cdi/libcdi.a
LIBCDCDIN=$(CASEDICTHOME)/cdcdin/libcdcdin.a
LIBSDDIAP=$(CASEDICTHOME)/sddiap/libsddiap.a
LIBTRETR=$(CASEDICTHOME)/tretr/libtretr.a

#
# Define CASE*Generator libraries
#
LIBCFBLD=$(CASEGENHOME)/cfbld/libcfbld.a
LIBCFDCT=$(CASEGENHOME)/cfdct/libcfdct.a
LIBCFFIF=$(CASEGENHOME)/cffif/libcffif.a
LIBCFIAS=$(CASEGENHOME)/cfias/libcfias.a
LIBCFLAY=$(CASEGENHOME)/cflay/libcflay.a
LIBCFOBJ=$(CASEGENHOME)/cfobj/libcfobj.a
LIBCFRET=$(CASEGENHOME)/cfret/libcfret.a
LIBCFSTR=$(CASEGENHOME)/cfstr/libcfstr.a
LIBCFTRG=$(CASEGENHOME)/cftrg/libcftrg.a
LIBCFUTL=$(CASEGENHOME)/cfutl/libcfutl.a
LIBCGDLL=$(CASEGENHOME)/cgdll/libcgdll.a
LIBCGOSD=$(CASEGENHOME)/cgosd/libcgosd.a
LIBCGSF=$(CASEGENHOME)/cgsf/libcgsf.a
LIBCGSM=$(CASEGENHOME)/cgsm/libcgsm.a
LIBCGSPR=$(CASEGENHOME)/cgspr/libcgspr.a
LIBCGSRW=$(CASEGENHOME)/cgsrw/libcgsrw.a
LIBCGUTL=$(CASEGENHOME)/cgutl/libcgutl.a
LIBCMLIN=$(CASEGENHOME)/cmlin/libcmlin.a
LIBCMOUT=$(CASEGENHOME)/cmout/libcmout.a
LIBCMPRO=$(CASEGENHOME)/cmpro/libcmpro.a
LIBCRARP=$(CASEGENHOME)/crarp/libcrarp.a
LIBCRDB=$(CASEGENHOME)/crdb/libcrdb.a
LIBCREQG=$(CASEGENHOME)/creqg/libcreqg.a
LIBCRFMT=$(CASEGENHOME)/crfmt/libcrfmt.a
LIBCRUTL=$(CASEGENHOME)/crutl/libcrutl.a
LIBGIAC=$(CASEGENHOME)/giac/libgiac.a
LIBGDLL=$(CASEGENHOME)/gdll/libgdll.a
LIBCFG=$(CASEGENHOME)/cfg/libcfg.a
LIBLAPP=$(CASEGENHOME)/lapp/liblapp.a
LIBPMOD=$(CASEGENHOME)/pmod/libpmod.a
LIBLMOD=$(CASEGENHOME)/lmod/liblmod.a
LIBGUTIL=$(CASEGENHOME)/gutil/libgutil.a

#
# Define CASE*Designer libraries
#

LIBDM=$(CASEDESIGNHOME)/dm/libdm.a
LIBCOMM=$(CASEDESIGNHOME)/comm/libcomm.a
LIBDFD=$(CASEDESIGNHOME)/dfd/libdfd.a
LIBERD=$(CASEDESIGNHOME)/erd/liberd.a
LIBFHD=$(CASEDESIGNHOME)/fhd/libfhd.a
LIBFRONT=$(CASEDESIGNHOME)/front/libfront.a
LIBMD=$(CASEDESIGNHOME)/md/libmd.a
LIBMDIAP=$(CASEDESIGNHOME)/mdiap/libmdiap.a
LIBOWM=$(CASEDESIGNHOME)/owm/libowm.a

LIBCDTOOL=$(CASEDESIGNHOME)/Xwin/libXwin.a

LIBDRAW=$(CASEDESIGNHOME)/draw/libdraw.a
LIBGRA=$(CASEDESIGNHOME)/utl/gra/libgra.a
LIBCDMSG=$(CASEDESIGNHOME)/utl/cdmsg/libcdmsg.a
LIBCXTERM=$(CASEDESIGNHOME)/utl/cxterm/libcxterm.a
LIBPOPUP=$(CASEDESIGNHOME)/utl/popup/libpopup.a
LIBCDUTIL=$(CASEDESIGNHOME)/utl/cdutil/libcdutil.a
```

```
LIBSDDI=$(CASEDESIGNHOME)/sddi/libsddi.a

# Define ally library so Oracle*Mail can find it.
LIBALLY1=$(ALLY1HOME)/libally.a

# Define ally library so SQL*ReportWriter can find it.
LIBALLY2=$(ALLY2HOME)/libally.a

# Oracle*Graphics Libraries
LIBOGCHART= $(LIBHOME)/libchart.a
LIBOGGC= $(LIBHOME)/libgc.a

LKLIBS=-lstublm

LIBSTUBLKMGR= $(LIBHOME)/libstublm.a

#
# In Unix, ORACORE is currently configured to use the native math library
# (within ORACORE), so the math library needs to be added to LDLIBS.
# We will look into this issue.
#
MATHLIB=-lm

# Define special C libraries to be linked with.
# LDLIBS= $(SPLIBS) $(CLIBS)
LDLIBS= $(SPLIBS) $(CLIBS) $(MATHLIB)
LDPLIBS= $(LDLIBS)

# Define orakit system link libraries.
OKTCLIBS= $(CLIBS)
OKTXLIBS= $(XLIBS) $(CLIBS)
OKTMLIBS= $(MOTIFLIBS) $(CLIBS)

# Define ToolKit-2 system link libraries.
TK2CLIBS= $(CLIBS)
TK2XLIBS= $(XLIBS) $(CLIBS)
TK2MLIBS= $(MOTIFLIBS) $(CLIBS)

TK2LIBS= $(LIBTK2UT) $(LIBTK2UTIL) $(LIBTK2SL)
MTK2LIBS= $(LIBTK2RE) $(LIBTK2M) $(TK2LIBS) $(LIBTK2ROS) $(MOTIFLIBS)
CTK2LIBS= $(LIBTK2RE) $(TK2LIBS) $(LIBTK2C) $(LIBTK2ROS) $(LIBTK2SL)

# Oracle Financials Macros
ALR_TOP= $(SRCHOME)/alr
AP_TOP= $(SRCHOME)/ap
AR_TOP= $(SRCHOME)/ar
FA_TOP= $(SRCHOME)/fa
FND_TOP= $(SRCHOME)/fnd
GL_TOP= $(SRCHOME)/gl
INST_TOP= $(SRCHOME)/inst
INV_TOP= $(SRCHOME)/inv
PER_TOP= $(SRCHOME)/per
PO_TOP= $(SRCHOME)/po
RA_TOP= $(SRCHOME)/ra
SA_TOP= $(SRCHOME)/sa
SO_TOP= $(SRCHOME)/so

FND_INCLUDE= $(I_SYM)$(FND_TOP)/include
APPL_PCCINCLUDE= $(I_SYM)$(ORACLE_HOME)/proc/lib
ALR_INCLUDE= $(I_SYM)$(ALR_TOP)/include
AP_INCLUDE= $(I_SYM)$(AP_TOP)/include
AR_INCLUDE= $(I_SYM)$(AR_TOP)/include
FA_INCLUDE= $(I_SYM)$(FA_TOP)/include
GL_INCLUDE= $(I_SYM)$(GL_TOP)/include
INV_INCLUDE= $(I_SYM)$(INV_TOP)/include
INST_INCLUDE= $(I_SYM)$(INST_TOP)/include
PER_INCLUDE= $(I_SYM)$(PER_TOP)/include
PO_INCLUDE= $(I_SYM)$(PO_TOP)/include
RA_INCLUDE= $(I_SYM)$(RA_TOP)/include
SA_INCLUDE= $(I_SYM)$(SA_TOP)/include
SO_INCLUDE= $(I_SYM)$(SO_TOP)/include

# Product library macros (conform to appl division naming convension)
# fnd
LFNDLIB= $(FND_TOP)/lib/libfnd.a
LUSRLIB= $(FND_TOP)/usrxit/libusr.a
#alr
#MAILLIB = /src/appl/fin7/alr/2.0.11/lib/libmh.a
#LALRLIB = $(ALR_TOP)/lib/libbalr.a $(MAILLIB)
LALRLIB = $(ALR_TOP)/lib/libalr.a
# gl
LGLLIB = $(GL_TOP)/lib/libgl.a
# ap
LAPLIB = $(AP_TOP)/lib/libap.a
# po
LPOLIB = $(PO_TOP)/lib/libpo.a
# fa
LFALIB = $(FA_TOP)/lib/libfa.a
# ar
LARLIB = $(AR_TOP)/lib/libar.a
# ra
LRALIB = $(RA_TOP)/lib/libra.a
# so
LSOLIB = $(SO_TOP)/lib/libso.a
# sa
LSALIB = $(SA_TOP)/lib/libsa.a
# inv
LINVLIB = $(INV_TOP)/lib/libinv.a
# per
LPERLIB = $(PER_TOP)/lib/libper.a
# inst
LINSTLIB = $(INST_TOP)/lib/libinst.a

# Objet macros (other product may refer)
XITOBJ= $(FND_TOP)/lib/xitiap.o $(FND_TOP)/lib/fdaiap.o

# Link libraries
FORMS_LINK = $(LOCAL_OBJS) $(SCP) $(FRMLIBS) $(PROLIBS) $(NETLIBS) \
       $(ORALIBS) $(CLIBS)
AIAP_LINK = $(LOCAL_OBJS) $(FRMLIBS) $(PROLIBS) $(NETLIBS) $(ORALIBS)
$(CLIBS)

#
# Define the above again for inter-module linting.  We define more
libraries
# than exist above because we won't allow two different makefiles to
# contribute to the same lint library.
#
LINTSTANDARD=$(LINTDIR)/llib-lstd_lib.ln
LINTLIB=$(LINTDIR)/llib-llib.ln
LINTSOSD=$(LINTDIR)/llib-lsosd.ln
LINTOSN=$(LINTDIR)/llib-losn.ln
```

```
LINTOSNTAB=$(LINTDIR)/llib-losntab.ln
LINTOSNTABST=$(LINTDIR)/llib-losntabst.ln

LINTTCPTLI=$(LINTDIR)/llib-ltcptli.ln

LINTSTAR=$(LINTDIR)/llib-lstar.ln

LINTOSNTLPTB=$(LINTDIR)/llib-losntlptb.ln
LINTTLI=$(LINTDIR)/llib-ltli.ln

LINTASYNC=$(LINTDIR)/llib-lasync.ln

LINTUTT=$(LINTDIR)/llib-lutt.ln
LINTUPI=$(LINTDIR)/llib-lupi.ln
LINTIAC=$(LINTDIR)/llib-liac.ln
LINTIAD=$(LINTDIR)/llib-liad.ln
LINTIAG=$(LINTDIR)/llib-liag.ln
LINTIAP=$(LINTDIR)/llib-liap.ln
LINTIAC30=$(LINTDIR)/llib-liac30.ln
LINTIAD30=$(LINTDIR)/llib-liad30.ln
LINTIAG30=$(LINTDIR)/llib-liag30.ln
LINTIAP30=$(LINTDIR)/llib-liap30.ln
LINTOKT=$(LINTDIR)/llib-lokt.ln
LINTOKTC=$(LINTDIR)/llib-loktc.ln
LINTOKTX=$(LINTDIR)/llib-loktx.ln
LINTOKTM=$(LINTDIR)/llib-loktm.ln
LINTTK2=$(LINTDIR)/llib-ltk2.ln
LINTTK2C=$(LINTDIR)/llib-ltk2c.ln
LINTTK2X=$(LINTDIR)/llib-ltk2x.ln
LINTTK2M=$(LINTDIR)/llib-ltk2m.ln

LINTSQL=$(LINTDIR)/llib-lsql.ln
LINTPCC=$(LINTDIR)/llib-lpcc.ln
LINTPSD=$(LINTDIR)/llib-lpsd.ln
LINTPROC=$(LINTDIR)/llib-lproc.ln
LINTPROFORTRAN=$(LINTDIR)/llib-lfor.ln
LINTPROCOBOL=$(LINTDIR)/llib-lcob.ln
LINTOCIC=$(LINTDIR)/llib-locic.ln

LINTPLS=$(LINTDIR)/llib-lpls.ln

LINTKNL=$(LINTDIR)/llib-lknl.ln
LINTRDBMSA=$(LINTDIR)/llib-lrdbmsa.ln
#LINTRDBMSB=$(LINTDIR)/llib-lrdbmsb.ln
LINTRDBMSC=$(LINTDIR)/llib-lrdbmsc.ln
LINTRDBMSD=$(LINTDIR)/llib-lrdbmsd.ln
LINTRDBMSE=$(LINTDIR)/llib-lrdbmse.ln
LINTRDBMSF=$(LINTDIR)/llib-lrdbmsf.ln
LINTRDBMSG=$(LINTDIR)/llib-lrdbmsg.ln
#LINTRDBMSH=$(LINTDIR)/llib-lrdbmsh.ln
LINTRDBMSI=$(LINTDIR)/llib-lrdbmsi.ln
#LINTRDBMSJ=$(LINTDIR)/llib-lrdbmsj.ln
#LINTRDBMSK=$(LINTDIR)/llib-lrdbmsk.ln
LINTRDBMSL=$(LINTDIR)/llib-lrdbmsl.ln
LINTRDBMSM=$(LINTDIR)/llib-lrdbmsm.ln
LINTRDBMSN=$(LINTDIR)/llib-lrdbmsn.ln
LINTRDBMSO=$(LINTDIR)/llib-lrdbmso.ln
LINTRDBMSP=$(LINTDIR)/llib-lrdbmsp.ln
LINTRDBMSQ=$(LINTDIR)/llib-lrdbmsq.ln
LINTRDBMSR=$(LINTDIR)/llib-lrdbmsr.ln
LINTRDBMSS=$(LINTDIR)/llib-lrdbmss.ln
LINTRDBMST=$(LINTDIR)/llib-lrdbmst.ln
LINTRDBMSU=$(LINTDIR)/llib-lrdbmsu.ln
LINTRDBMSV=$(LINTDIR)/llib-lrdbmsv.ln
#LINTRDBMSW=$(LINTDIR)/llib-lrdbmsw.ln
LINTRDBMSX=$(LINTDIR)/llib-lrdbmsx.ln
#LINTRDBMSY=$(LINTDIR)/llib-lrdbmsy.ln
#LINTRDBMSZ=$(LINTDIR)/llib-lrdbmsz.ln

LINTORA=$(LINTKNL) $(LINTRDBMSA) $(LINTRDBMSB) $(LINTRDBMSC) $(LINTRDBMSD)
\
        $(LINTRDBMSE) $(LINTRDBMSF) $(LINTRDBMSG) $(LINTRDBMSH)
$(LINTRDBMSI) \
        $(LINTRDBMSJ) $(LINTRDBMSK) $(LINTRDBMSL) $(LINTRDBMSM)
$(LINTRDBMSN) \
        $(LINTRDBMSO) $(LINTRDBMSP) $(LINTRDBMSQ) $(LINTRDBMSR)
$(LINTRDBMSS) \
        $(LINTRDBMST) $(LINTRDBMSU) $(LINTRDBMSV) $(LINTRDBMSW)
$(LINTRDBMSX) \
        $(LINTRDBMSY) $(LINTRDBMSZ)

TTLINT= $(LINTUPI) $(LINTNET) $(LINTOSN)
STLINT= $(LINTUPI) $(LINTOSNST) $(LINTSOSD) $(LINTCONFIG)
NETLINT= $(LINTUTT) $(LINTTCP) $(LINTDNT) $(LINTASYNC) $(LINTTLI)
$(LIBX25)
OLINT= $(LINTLIB) $(LINTSOSD) $(LINTSTANDARD)
LDLINT= $(OLINT) $(CLINT)

#
#  Include file directory definitions
#
NETINCLUDE=$(I_SYM)$(NETHOME)/public
FORMSINCLUDE=$(I_SYM)$(FORMSHOME)/include
MENUINCLUDE=$(I_SYM)$(MENUHOME)/include
FORMS30CINCLUDE=$(I_SYM)$(FORMS30HOME)/headers
$(I_SYM)$(FORMS30HOME)/include
FORMS30XINCLUDE=$(I_SYM)$(FORMS30HOME)/x/headers
$(I_SYM)$(FORMS30HOME)/include
FORMS30MINCLUDE=$(I_SYM)$(FORMS30HOME)/mot/headers
$(I_SYM)$(FORMS30HOME)/include
MENU5INCLUDE=$(I_SYM)$(MENU5HOME)/headers $(I_SYM)$(MENU5HOME)/include
MENU5CINCLUDE=$(I_SYM)$(MENU5HOME)/headers $(I_SYM)$(MENU5HOME)/include
MENU5XINCLUDE=$(I_SYM)$(MENU5HOME)/x/headers $(I_SYM)$(MENU5HOME)/include
MENU5MINCLUDE=$(I_SYM)$(MENU5HOME)/mot/headers
$(I_SYM)$(MENU5HOME)/include
ORATERMINCLUDE=$(I_SYM)$(ORATERMHOME)/headers
$(I_SYM)$(ORATERMHOME)/include

OKTCINCLUDE= \
 $(I_SYM)$(OKTCHOME)/include \
 $(I_SYM)$(OKTCHOME)/headers \
 $(I_SYM)$(OKTCHOME)/private \
 $(I_SYM)$(OKTCHOME)/termio/include \
 $(I_SYM)$(OKTHOME)/sl \
 $(I_SYM)$(OKTHOME)/resource/include

OKTXINCLUDE= \
 $(I_SYM)$(OKTXHOME)/include \
 $(I_SYM)$(OKTXHOME)/headers \
 $(I_SYM)$(OKTXHOME)/private \
 $(I_SYM)$(OKTHOME)/sl \
 $(I_SYM)$(OKTHOME)/resource/include
```

```
OKTMINCLUDE= \
 $(I_SYM)$(OKTMHOME)/include \
 $(I_SYM)$(OKTMHOME)/headers \
 $(I_SYM)$(OKTMHOME)/private \
 $(I_SYM)$(OKTHOME)/sl \
 $(I_SYM)$(OKTHOME)/resource/include

TK2PUBLIC= \
 $(COREPUBLIC) \
 $(RDBMSPUBLIC) \
 $(I_SYM)$(TK2HOME)/include \
 $(I_SYM)$(TK2HOME)/defs

TK2INCLUDE=$(TK2PUBLIC)

TK2PRIVATE= \
 $(COREPUBLIC) \
 $(RDBMSPUBLIC) \
 $(I_SYM)$(TK2HOME)/include \
 $(I_SYM)$(TK2HOME)/defs \
 $(I_SYM)$(TK2HOME)/re/include \
 $(I_SYM)$(TK2HOME)/rem/include \
 $(I_SYM)$(TK2HOME)/uc/include \
 $(I_SYM)$(TK2HOME)/ut/include \
 $(I_SYM)$(TK2HOME)/ot/include \
 $(I_SYM)$(TK2HOME)/ros/include \
 $(I_SYM)$(TK2HOME)/uipr/include \
 $(I_SYM)$(TK2HOME)/sl

TK2PRIVATEA= \
 $(COREPUBLIC) \
 $(RDBMSPUBLIC) \
 -DUI_PROTO \
 $(I_SYM)$(TK2HOME)/include.ansi \
 $(I_SYM)$(TK2HOME)/defs \
 $(I_SYM)$(TK2HOME)/re/include \
 $(I_SYM)$(TK2HOME)/rem/include \
 $(I_SYM)$(TK2HOME)/uc/include \
 $(I_SYM)$(TK2HOME)/ut/include \
 $(I_SYM)$(TK2HOME)/ot/include \
 $(I_SYM)$(TK2HOME)/ros/include \
 $(I_SYM)$(TK2HOME)/uipr/include \
 $(I_SYM)$(TK2HOME)/sl

SQLPLUSINCLUDE=$(I_SYM)$(SQLPLUSHOME)/include
MAILINCLUDE= $(I_SYM)$(MAILHOME)/include
CASEDICTINCLUDE= $(I_SYM)$(CASEDICTHOME)/include
CASEGENINCLUDE= $(I_SYM)$(CASEGENHOME)/include
TYPESINCLUDE= $(I_SYM)$(CASEDESIGNHOME)/include
BITMAPSINCLUDE= $(I_SYM)$(CASEDESIGNHOME)/admin/bitmaps
QMXINCLUDE= $(I_SYM)$(QMXHOME)/include
DECWININCLUDE=$(I_SYM)$(CASEDESIGNHOME)/include/dwin

CASEDESIGNINCLUDE= $(I_SYM)$(CASEDESIGNHOME)/include/Xwin
CASEDESIGNPCCINCLUDE= $(PCCINCLUDE) include=$(CASEDESIGNHOME)/include/Xwin

SQLTRINCLUDE= $(I_SYM)$(SQLTRHOME)/include
NLSWBINCLUDE= $(I_SYM)$(NLSWBHOME)/include
MAKESHIPINCLUDE= $(I_SYM)$(MAKESHIPHOME)/include
```

```
# Default library name
L=lib$(LIB).a
#
# Definitions of makefile capabilities
#

#
# Recursive command macros #

DOTARGS=\
       TARG=`echo $@ | sed 's/\..*//'`; export TARG; \
       if [ "$(DIRS)" ]; then \
       for dir in $(DIRS); do \
              echo; echo " cd $$dir; $(MAKE) $$TARG"; \
              BASE=`basename $$dir`;\
              DATE=`date`;echo " --------- Start $$BASE: $$DATE ---------
"; \
              (cd $$dir; $(MAKE) $$TARG); \
              DATE=`date`;echo " ---------   End $$BASE: $$DATE ---------
"; \
       done \
       else echo "WARNING: macro DIRS is NOT set!  Check dir: `pwd`"; \
       fi

DOFILES=if [ "$(DIRS)" ]; then \
       for dir in $(DIRS); do \
              echo; echo " cd $$dir; $(MAKE.FILE)"; \
              (cd $$dir; $(MAKE.FILE)); \
       done \
       else echo "WARNING: macro DIRS is NOT set!  Check dir: `pwd`"; \
       fi

CLEANFILES=if [ "$(DIRS)" ]; then \
       for dir in $(DIRS); do \
              echo; echo " cd $$dir; rm -f Makefile"; \
              (cd $$dir; rm -f Makefile); \
       done \
       else echo "WARNING: macro DIRS is NOT set!  Check dir: `pwd`"; \
       fi

BUILDLIB=for dir in $(LIBDIRS); do \
              echo; echo "cd $$dir; $(MAKE) archive"; \
              cd $$dir; $(MAKE) archive; \
       done

DEHEAD=if [ "$(DATA_DIRS)" ]; then \
       for dir in $(DATA_DIRS); do \
              echo; echo " cd $$dir; dehead -q *.d"; \
              (cd $$dir; dehead -q *.d); \
       done \
       else echo "WARNING: macro DATA_DIRS is NOT set!  Check dir:
`pwd`"; \
       fi

CLEANOKTC=files=""; \
       okfiles=`echo *.ok 2>/dev/null`; \
       if [ "$$okfiles" = "*.ok" -o -z "$$okfiles" ] ; then :; \
              else for file in $$okfiles; do \
                     r=`echo $$file | sed s/[.]ok$$//`; \
                     files=$$files" $$r.c"; \
                     done  ; \
```

```
                echo "        rm -f $$files"; rm -f $$files ; \
        fi

CLEANOKTH=files=""; \
        okfiles=`echo *.hok 2>/dev/null`; \
        if [ "$$okfiles" = "*.hok" -o -z "$$okfiles" ] ; then :; \
                else for file in $$okfiles; do \
                        r=`echo $$file | sed s/[.]hok$$//`; \
                        files=$$files" $$r.h"; \
                        done  ; \
                echo "        rm -f $$files"; rm -f $$files ; \
        fi

#
# Compound commands for make machine
#
DOMACH= if test -r $@ ; then \
                if cmp $@ $@$(MACHINE) > /dev/null 2>&1 ; then rm -f $@ ; \
                else if test -w $@ ; then mv $@ $@.OLD ; \
                        else rm -f $@ ; \
                        fi ; \
                fi ; \
        fi ; \
        echodo cp $@$(MACHINE) $@ ; \
        chmod a-w $@
#
# Suffix rules
#

.c:
        $(CC) $(CFLAGS) $(LDFLAGS) $<

.c.o:
        $(COMPO)

.c.s:
        $(COMPS)

.c.i:
        $(CC) -Xc -P $(PFLAGS) $*.c > $*.i
        crunch $*.i > $*.ii

.c.d:
        $(COMPD)
.ok.c:
        @rm -f $*.c
        $(OKPPC)
.ok.o:
        $(OKPPC); $(COMPO)

.ok.a:
        $(OKPPC)
        $(COMPO)
.hok.h:
        @rm -f $*.h
        $(OKPPH)
.scr.r:
        @rm -f $*.r
        $(OKSCR)
.c.a:
        $(COMPO)

.s.a:
        $(AS) $(ASFLAGS) -o $% $<

.pc.c:
        @rm -f $*.c
        proc $(PCCFLAGS) $(TEST_ARGS) iname=$*.pc
.pc.o:
        proc $(PCCFLAGS) $(TEST_ARGS) iname=$*.pc
        $(COMPO)

CCFLAGS= -Wl,-z,norearrange -Xa -Hon=Read_only_strings -Hnocopyr -Hih -
Hon=use_frame_pointer -Hoff=optimize_fp
OTHERLIBS=-lnet -lsocket -lnsl -lbt -lelf -lm -lstublm -lasyncio -lx -lgen
LDFLAGS= -Wl,-z,norearrange -o $@ -L$(LIBHOME)

.NOEXPORT:

F77=lpifortran
F77LD=lpild
F77LIBS= $(ORACLE_HOME)/profor/lib/lpisqlif.a
COBOL=lpicobol
COBEXT=cob
COBDIR=/usr/lib/cobol
COBFLAGS=-si
COBOLLD= ldcobol
COBLIBS=$(ORACLE_HOME)/procob/lib/lpisqlif.a
RTSORALIB=-lnetv2 -lnttcp -lora -ltcp -lutt -lnetwork -lora

XLIBS=-lXt -lXmu -lX11 -lXaw -lXext
MOTIFLIBS=-lXm -lXwchar -lXt -lX11 -lgen -lm
MOTIFINCLUDE=$(I_SYM)/usr/include/Xm $(I_SYM)/usr/include/X11

SO=a

NLSRTLHOME= $(SRCHOME)/nlsrtl3

NLSRTLLIB= -lnlsrtl3.a

LIBNLSRTL= $(LIBHOME)/libnlsrtl3.$(SO)
LLIBNLSRTL=-lnlsrtl3

NLSRTLINC = $(I_SYM)$(NLSRTLHOME)/public $(I_SYM)$(NLSRTLHOME)/sosd/public
\
$(I_SYM)$(NLSRTLHOME)/include $(I_SYM)$(NLSRTLHOME)/src/nlsdata

NLSRTLPUBLIC = $(I_SYM)$(NLSRTLHOME)/public \
$(I_SYM)$(NLSRTLHOME)/sosd/public $(STDINCLUDE)

COREHOME= $(SRCHOME)/oracore3

COREPUBLIC= $(I_SYM)$(COREHOME)/public
COREINCLUDES= $(NLSRTLPUBLIC) $(COREPUBLIC)

LIBCORE= $(LIBHOME)/libcore3.$(SO)
LIBCV6= $(LIBHOME)/libc3v6.$(SO)
LLIBCORE= -lcore3
LLIBCV6= -lc3v6

CORELIBD= $(LIBNLSRTL) $(LIBCV6) $(LIBCORE)
CORELIBS= $(LLIBNLSRTL) $(LLIBCV6) $(LLIBCORE) $(LLIBNLSRTL) \
```

```
       $(LLIBCCOREFUND) $(LLIBCORE)

CORESRCINCLUDE=$(I_SYM)$(COREHOME)/include $(I_SYM)$(COREHOME)/lnx \
  $(COREINCLUDES) $(I_SYM).

SRCINCLUDE=$(CORESRCINCLUDE)

CORELIB= $(LIBCORE)

NET2HOME= $(SRCHOME)/network
LIBHOME= $(SRCHOME)/lib

LIBNETWORK= $(LIBHOME)/libsqlnet.a
NTCONTAB= $(LIBHOME)/ntcontab.o

LIBNTTCP= $(LIBHOME)/libnttcp.a
NTTCPLIBS=-lnttcp

LIBNTO= $(LIBHOME)/libnto.a
NTOLIBS=-lnto

LIBNTTLI = $(LIBHOME)/libnttli.a
NTILIBS=-lnttli

LIBNTYS= $(LIBHOME)/libntys.a
NTYSLIBS=-lntys

TNSLIBS= $(NTCONTAB) $(NTTCPLIBS) $(NTITLIBS) $(NTDLIBS) $(NTIDLIBS)
$(NTXLIBS)\
$(NTSLIBS) $(NTRLIBS) $(NMPLIBS) $(NTBLIBS) $(NTOLIBS) $(NTILIBS)
$(NTLU62LIBS)\
$(NTYSLIBS)
TNSLIBD= $(NTCONTAB) $(LIBNTTCP) $(LIBNTIT) $(LIBNTD) $(LIBNTID) $(LIBNTX)
\
$(LIBNTS) $(LIBNTR) $(LIBNMP) $(LIBNTB) $(LIBNTO) $(LIBNTTLI) $(LIBNTLU62)
$(LIBNTYS)

NTNSLIBS= $(NTTCPLIBS) $(NTITLIBS) $(NTDLIBS) $(NTIDLIBS) $(NTXLIBS)\
$(NTSLIBS) $(NTRLIBS) $(NMPLIBS) $(NTBLIBS) $(NTOLIBS) $(NTILIBS)
$(NTLU62LIBS)\
$(NTYSLIBS)
NTNSLIBD= $(LIBNTTCP) $(LIBNTIT) $(LIBNTD) $(LIBNTID) $(LIBNTX) \
$(LIBNTS) $(LIBNTR) $(LIBNMP) $(LIBNTB) $(LIBNTO) $(LIBNTTLI) $(LIBNTLU62)
\
$(LIBNTYS)

LIBSQLNET=$(LIBHOME)/libsqlnet.a
LLIBSQLNET=-lsqlnet
NETLIBD= $(OSNTAB) $(LIBSQLNET)
NETLIBS=  $(OSNTAB) $(LLIBSQLNET)

TNSINCLUDE= $(I_SYM)$(NET2HOME)/public
TCPINCLUDE= $(I_SYM)$(NET2HOME)/nt/tcp/src
DNTINCLUDE= $(I_SYM)$(NET2HOME)/nt/decnet/src
IDNTINCLUDE= $(I_SYM)$(NET2HOME)/nt/dnttli/src
IPCINCLUDE= $(I_SYM)$(NET2HOME)/nt/ipc
BEQINCLUDE= $(I_SYM)$(NET2HOME)/nt/beq
NPLINCLUDE= $(I_SYM)$(NET2HOME)/npl/src
NSINCLUDE= $(I_SYM)$(NET2HOME)/ns/src
NMPINCLUDE= $(I_SYM)$(NET2HOME)/nm/nmp/src
NNGINCLUDE= $(I_SYM)$(NET2HOME)/nng/src
NNIINCLUDE= $(I_SYM)$(NET2HOME)/nn/nni/src
NLINCLUDE= $(I_SYM)$(NET2HOME)/nl/src

NETHOME= $(SRCHOME)/sqlnet

OSNTAB= $(LIBHOME)/osntab.o
OSNTABST= $(LIBHOME)/osntabst.o
LIBSQLNET= $(LIBHOME)/libsqlnet.a

SQLNETLIBS= -lsqlnet

LIBTCPTLI= $(LIBHOME)/libtcptli.a

TCPTLILIBS=-ltcptli

LIBSTAR= $(LIBHOME)/libstar.a

LIBSPX= $(LIBHOME)/libspx.a

LIBOSI= $(LIBHOME)/libosi.a

OSNTLPTB= $(LIBHOME)/osntlptb.o

LIBTLI= $(LIBHOME)/libtli.a $(OSNTLPTB) $(LIBSPX) $(LIBTCPTLI)
$(LIBX25TLI) \
       $(LIBSTAR) $(LIBOSI) $(LIBATK)

LIBASYNC= $(LIBHOME)/libasync.a

ASYNCLIBS=-lasync

NETV1LIBD= $(LIBTCP) $(LIBDNT) $(LIBASYNC) $(LIBTLI) $(LIBLU62) $(LIBX25)
$(LIBCMX) $(LIBNP)

NETINCLUDE= $(I_SYM)$(NETHOME)/public

LINTOSN= $(LINTDIR)/llib-losn.ln
LINTOSNTAB= $(LINTDIR)/llib-losntab.ln
LINTOSNTABST= $(LINTDIR)/llib-losntabst.ln

LINTTCPTLI= $(LINTDIR)/llib-ltcptli.ln

LINTSTAR= $(LINTDIR)/llib-lstar.ln

LINTOSNTLPTB= $(LINTDIR)/llib-losntlptb.ln
LINTTLI= $(LINTDIR)/llib-ltli.ln

LINTASYNC= $(LINTDIR)/llib-lasync.ln

PLS=pls

P2C=p2c

PLSHOME= $(SRCHOME)/plsql
PLSPUBLIC=$(I_SYM)$(PLSHOME)/public

PLSSRCINCLUDE= $(SRCINCLUDE) $(PLSPUBLIC) $(I_SYM)$(PLSHOME)/include
PLSLIBPSD=$(PLSHOME)/lib/libpsd.a

PLSINCLUDE=$(PLSPUBLIC)
```

```
LIBPLS= $(LIBHOME)/libpls.a
LLIBPLS= -lpls

LIBNLSRTL= $(LIBHOME)/libnlsrtl3.a $(LIBHOME)/libnlsrtl.a
LLIBNLSRTL=-lnlsrtl3 -lnlsrtl
LIBCORE= $(LIBHOME)/libcore3.$(SO) $(LIBHOME)/libcore.$(SO)
LLIBCORE= -lcore3 -lcore

ROSHOME= $(TK2HOME)/ros

ROSPUBLIC= $(I_SYM)$(ROSHOME)/include
ROSDEFS= $(I_SYM)$(ROSHOME)/defs

LIBROS= $(LIBHOME)/libros.a
LLIBROS= -lros

TK2HOME= $(SRCHOME)/tk2
TK2CHOME= $(SRCHOME)/tk2/uicm
TK2MHOME= $(SRCHOME)/tk2/uimotif
TK2UIXIL=uixil

OTHERLIBS_TK2C= $(LLIBORA) $(CORELIBS)

OTHERLIBS_TK2M= $(CORELIBS) $(MOTIFLIBS)

COREINC_TK2= $(COREINCLUDES)

TK2PUBLIC= \
 $(COREINC_TK2) \
 $(RDBMSPUBLIC) \
 $(ROSPUBLIC) \
 $(I_SYM)$(TK2HOME)/include \
 $(I_SYM)$(TK2HOME)/defs

TK2INCLUDE=$(TK2PUBLIC)

TK2PUBLICA= \
 $(COREINC_TK2) \
 $(RDBMSPUBLIC) \
 -DUI_PROTO \
 $(ROSPUBLIC) \
 $(I_SYM)$(TK2HOME)/include.ansi \
 $(I_SYM)$(TK2HOME)/defs

TK2PRIVATE= \
 $(CFLAGS_TK2) \
 $(COREINC_TK2) \
 $(RDBMSPUBLIC) \
 $(ROSPUBLIC) \
 $(I_SYM)$(TK2HOME)/include \
 $(I_SYM)$(TK2HOME)/defs \
 $(I_SYM)$(TK2HOME)/re/include \
 $(I_SYM)$(TK2HOME)/rem/include \
 $(I_SYM)$(TK2HOME)/uc/include \
 $(I_SYM)$(TK2HOME)/ut/include \
 $(I_SYM)$(TK2HOME)/ot/include \
 $(I_SYM)$(TK2HOME)/ros/include \
 $(I_SYM)$(TK2HOME)/uipr/include \
 $(I_SYM)$(TK2HOME)/sl

TK2PRIVATEA= \
 $(CFLAGS_TK2) \
 $(COREINC_TK2) \
 $(RDBMSPUBLIC) \
 -DUI_PROTO \
 $(ROSPUBLIC) \
 $(I_SYM)$(TK2HOME)/include.ansi \
 $(I_SYM)$(TK2HOME)/defs \
 $(I_SYM)$(TK2HOME)/re/include \
 $(I_SYM)$(TK2HOME)/rem/include \
 $(I_SYM)$(TK2HOME)/uc/include \
 $(I_SYM)$(TK2HOME)/ut/include \
 $(I_SYM)$(TK2HOME)/ot/include \
 $(I_SYM)$(TK2HOME)/ros/include \
 $(I_SYM)$(TK2HOME)/uipr/include \
 $(I_SYM)$(TK2HOME)/sl

LINTTK2=$(LINTDIR)/llib-ltk2.ln
LINTTK2C=$(LINTDIR)/llib-ltk2c.ln
LINTTK2X=$(LINTDIR)/llib-ltk2x.ln
LINTTK2M=$(LINTDIR)/llib-ltk2m.ln

TK2COUNTRIES=D DK E HU JA KO NL PT  SF US

LIBTK2C= $(LIBHOME)/libtk2c.a
LLIBTK2C= -ltk2c
LIBTK2M= $(LIBHOME)/libtk2m.a
LLIBTK2M= -ltk2m

LIBQAP= $(LIBHOME)/libqapstub.a
LLIBQAP= -lqapstub

LIBTK2P= $(LIBHOME)/libtk2p.a
LLIBTK2P= -ltk2p
LIBTK2UC= $(LIBHOME)/libuc.a
LLIBTK2UC= -luc
LIBTK2REM= $(LIBHOME)/librem.a
LLIBTK2REM= -lrem
LIBTK2OT= $(LIBHOME)/libot.a
LLIBTK2OT= -lot
LIBTK2OTX= $(LIBHOME)/libotx.a

LLIBTK2OTX= -lotx

LIBTK2UT= $(LIBHOME)/libut.a

LLIBTK2UT= -lut

LIBTK2SL= $(LIBHOME)/libsl.a
LLIBTK2SL= -lsl
LIBTK2RE= $(LIBHOME)/libre.a

LLIBTK2RE= -lre

TK2UICLIBD= $(LIBTK2RE) $(LIBTK2UC) $(LIBTK2C) $(LIBTK2OT) $(LIBTK2REM) \
        $(LIBROS) $(LIBTK2P) $(LIBTK2UT) $(LIBTK2SL)

TK2UICLIBS= $(DYNAMIC_ON_TK2) \
        $(LLIBTK2SL) $(LLIBTK2RE) $(LLIBTK2UC) $(LLIBTK2C) $(LLIBTK2OT) \
        $(LLIBTK2RE) $(LLIBTK2REM) $(LLIBROS) $(LLIBTK2C) $(LLIBTK2P) \
        $(LLIBTK2C) $(LLIBTK2P) -lm $(LLIBTK2REM) $(LLIBTK2C) $(LLIBTK2OT)
\
```

```
        $(LLIBTK2UT) $(LLIBTK2UC) $(LLIBTK2SL) $(LLIBTK2C) \
        $(DYNAMIC_OFF_TK2)

TK2UIMLIBD= $(LIBTK2OT) $(LIBTK2RE) $(LIBTK2UC) $(LIBTK2REM) $(LIBROS) \
        $(LIBTK2M) $(LIBQAP) $(LIBTK2P) $(LIBTK2UT) $(LIBTK2SL)

TK2UIMLIBS= $(DYNAMIC_ON_TK2) \
        $(LLIBTK2OT) $(LLIBTK2RE) $(LLIBTK2UC) $(LLIBTK2REM) $(LLIBROS) \
        $(LLIBTK2M) $(LLIBQAP) $(LLIBTK2P) $(LLIBTK2M) $(LLIBTK2P) \
         $(LLIBTK2M) $(LLIBTK2OT) $(LLIBTK2UT) $(LLIBTK2SL)
$(DYNAMIC_OFF_TK2)

PLSPECFILES=

SRCINCLUDE= $(COREINCLUDES) $(RDBMSPUBLIC) $(MLSINCLUDE) $(I_SYM).

LIBOCIC= $(LIBHOME)/libocic.a
LLIBOCIC= -locic

TTLIBD= $(NETLIBD) $(LIBORA) $(CORELIBD)

TTLIBS= $(NETLIBS) $(LLIBORA) $(LLIBSQLNET) $(LLIBORA) $(LIBPLSHACK)
$(CORELIBS) $(LDLIBS)

STLIBDNOPLS= $(OSNTABST) $(CONFIG) $(CORELIBD) $(NETV2LIBD)
$(RDBMSLIBDNOPLS)
STLIBD= $(OSNTABST) $(CONFIG) $(CORELIBD) $(NETV2LIBD) $(RDBMSLIBD)
STLIBSNOPLS= $(OSNTABST) $(CONFIG) $(RDBMSLIBSNOPLS) $(RDBMSLIBSNOPLS) \
 $(PLSPECFILES) $(LLIBSQLNET) $(LLIBORA) $(LLIBSQLNET) $(CORELIBS)
$(LKLIBS) \
 $(LDLIBS)
STLIBS= $(OSNTABST) $(CONFIG) $(RDBMSLIBS) $(RDBMSLIBS) \
 $(PLSPECFILES) $(LLIBSQLNET) $(LLIBORA) $(LLIBSQLNET) $(CORELIBS)
$(LKLIBS) \
 $(LDLIBS)

RDBMSHOME= $(SRCHOME)/rdbms
RDBMSPUBLIC= $(I_SYM)$(RDBMSHOME)/public
RDBMSINCLUDE= $(I_SYM)$(RDBMSHOME)/include
RDBMSSOSD= $(I_SYM)$(RDBMSHOME)/sosd
RDBMSSRCINCLUDE= $(COREINCLUDES) $(RDBMSPUBLIC) $(RDBMSINCLUDE)
$(RDBMSSOSD) \
 $(PLSINCLUDE) $(TNSINCLUDE) $(NETINCLUDE) $(I_SYM).
MLSINCLUDE=

LLIBNETV2= $(LIBNETV2)
LLIBNETWORK= $(LIBNETWORK)

CONFIG= $(LIBHOME)/config.o
OPIMAI= $(LIBHOME)/opimai.o
LIBORA= $(LIBHOME)/libora.a
LIBKNL= $(LIBHOME)/libknl.a
LIBKNLOPT= $(LIBHOME)/libknlopt.a

LLIBORA= -lora

LLIBKNL= -lknl
LLIBKNLOPT= -lknlopt

RDBMSLIBDNOPLS= $(LIBORA) $(LIBKNLOPT) $(LIBKNL)
RDBMSLIBD= $(LIBORA) $(LIBKNLOPT) $(LIBPLS) $(LIBKNL)
RDBMSLIBSNOPLS= $(LLIBORA) $(LLIBKNLOPT) $(LLIBKNL)
RDBMSLIBS= $(LLIBORA) $(LLIBKNLOPT) $(LLIBPLS) $(LLIBKNL)

LIBSOSD=
LLIBSOSD=

SOSDINCLUDE= $(RDBMSSOSD)
BIGORALIB= $(LIBORA)

DIR=/net/home/tdevita/tpcc_722/tpc/tpcc/source
LIB=source
SRCS=90per.c tpccsvr.c getrand.c plsto.c pldel.c plord.c plpay.c
test_tran.c\
test_drv.c plnew.c tpccload.c tpccpl.c c_dump.c c_drv.c
OBJS=90per.o tpccsvr.o getrand.o plsto.o pldel.o plord.o plpay.o
test_tran.o\
test_drv.o plnew.o tpccload.o tpccpl.o c_dump.o c_drv.o
MEMBS=$L(90per.o) $L(tpccsvr.o) $L(getrand.o) $L(plsto.o) $L(pldel.o)\
$L(plord.o) $L(plpay.o) $L(test_tran.o) $L(test_drv.o) $L(plnew.o)\
$L(tpccload.o) $L(tpccpl.o) $L(c_dump.o) $L(c_drv.o)

#
# $Header: tpcc_src.mk 7030100.2 95/08/01 15:30:39 plai Generic<base> $
Copyr (c) 1994 Oracle
#
#
#
#=================================================================+
#        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA      |
#                   OPEN SYSTEMS PERFORMANCE GROUP                |
#                        All Rights Reserved                      |
#=================================================================+
# FILENAME
#      tpcc_src.mk
# DESCRIPTION
#      Makefile suffix for bench/tpc/tpcc/source directory
#=================================================================
#
# Suffixes:
#   .ott : two-task program
#   .ost : single-task program
.SUFFIXES: .ott .ost
#
#  Programs:
#
#      tpcc.ott, tpcc.ost:         OCI TPC-C generator
#      tpccload.ott, tpccload.ost: Database loader for TPC-C
#      getrand:                    Program to generate random number
#      90per:                      Program to find 90th percentile
#

DIRS=

INCLUDE=$(I_SYM). $(I_SYM)$(SRCHOME)/rdbms/demo
ITUX=$(I_SYM)$(ROOTDIR)/include

L=
LIBRARY=$(L)

MEMBS=
OBJS=tpccload.o c_trans.o c_drv.o c_dump.o tpccpl.o getrand.o 90per.o
```

```
CTRAN_OBJS=plnew.o plpay.o plord.o pldel.o plsto.o
CTRANTUX_OBJS=plnew_tux.o plpay.o plord.o pldel.o plsto.o
OTHER_OBJS=c_drv_val.o test_drv.o test_sample.o test_tran.o
TUX_OBJS=c_drv_tux.o tpccpl_tux.o tpccsvr.o

TARGS=files load compile setup

all: files compile load setup

$(TARGS):

files:
	@-$(DOFILES)
	@-$(DOTARGS)

compile: $(OBJS)
	@-$(DOTARGS)

load: tpcc.ost tpcc.ott tpccload.ost tpccload.ott getrand 90per
	@-$(DOTARGS)

cleanup:
	rm -f $(OBJS) $(CTRAN_OBJS) $(CTRANTUX_OBJS) $(OTHER_OBJS) \
	$(TUX_OBJS)

tpccload.o: tpccload.c tpcc.h
	$(CC) $(CFLAGS) $(INCLUDE) -c tpccload.c

c_drv.o: c_drv.c tpcc.h tpcc_info.h
	$(CC) $(CFLAGS) $(INCLUDE) -c c_drv.c

c_drv_val.o: c_drv.c tpcc.h tpcc_info.h
	cp c_drv.c c_drv_val.c
	$(CC) $(CFLAGS) -DVALIDATE $(INCLUDE) -c c_drv_val.c
	rm c_drv_val.c

c_drv_tux.o: c_drv.c tpcc.h tpcc_info.h
	cp c_drv.c c_drv_tux.c
	$(CC) $(CFLAGS) -DTUX $(INCLUDE) $(ITUX) -c c_drv_tux.c
	rm c_drv_tux.c

c_dump.o: c_dump.c tpcc.h tpcc_info.h
	$(CC) $(CFLAGS) $(INCLUDE) -c c_dump.c

test_drv.o: test_drv.c tpcc.h tpcc_info.h
	$(CC) $(CFLAGS) $(INCLUDE) -c test_drv.c

test_sample.o: test_drv.c tpcc.h tpcc_info.h
	cp test_drv.c test_sample.c
	$(CC) $(CFLAGS) -DSAMPLE $(INCLUDE) -c test_sample.c
	rm test_sample.c

test_tran.o: test_tran.c tpcc.h tpcc_info.h
	$(CC) $(CFLAGS) $(INCLUDE) -c test_tran.c

c_trans.o: $(CTRAN_OBJS)
	ld -r -o c_trans.o $(CTRAN_OBJS)

c_trans_tux.o: $(CTRANTUX_OBJS)
	ld -r -o c_trans_tux.o $(CTRANTUX_OBJS)

tpccpl.o: tpccpl.c tpcc.h tpcc_info.h tpccpl.h
	$(CC) $(CFLAGS) $(INCLUDE) -c tpccpl.c

tpccpl_tux.o: tpccpl.c tpcc.h tpcc_info.h tpccpl.h
	cp tpccpl.c tpccpl_tux.c
	$(CC) $(CFLAGS) -DTUX -DNULL_TRANS $(INCLUDE) $(ITUX) -c
tpccpl_tux.c
	rm tpccpl_tux.c

plnew_tux.o: plnew.c tpcc.h tpccpl.h
	cp plnew.c plnew_tux.c
	$(CC) $(CFLAGS) -DTUX $(INCLUDE) $(ITUX) -c plnew_tux.c
	rm plnew_tux.c

plnew.o: plnew.c tpcc.h tpccpl.h
	$(CC) $(CFLAGS) $(INCLUDE) -c plnew.c

plpay.o: plpay.c tpcc.h tpccpl.h
	$(CC) $(CFLAGS) $(INCLUDE) -c plpay.c

plord.o: plord.c tpcc.h tpccpl.h
	$(CC) $(CFLAGS) $(INCLUDE) -c plord.c

pldel.o: pldel.c tpcc.h tpccpl.h
	$(CC) $(CFLAGS) $(INCLUDE) -c pldel.c

plsto.o: plsto.c tpcc.h tpccpl.h
	$(CC) $(CFLAGS) $(INCLUDE) -c plsto.c

tpccsvr.o: tpccsvr.c tpcc.h tpcc_info.h
	$(CC) $(CFLAGS) $(INCLUDE) $(ITUX) -c tpccsvr.c

getrand.o: getrand.c
	$(CC) $(CFLAGS) $(INCLUDE) -c getrand.c

90per.o: 90per.c
	$(CC) $(CFLAGS) $(INCLUDE) -c 90per.c

getrand: getrand.o
	$(CC) $(CFLAGS) -o $@ getrand.o

90per: 90per.o
	$(CC) $(CFLAGS) -o $@ 90per.o

tpccload.ost: tpccload.o
	$(CC) $(CFLAGS) -o $@ $(INCLUDE) \
	-L$(SRCHOME)/lib tpccload.o \
	-lbench $(LIBOCIC) \
	$(STLIBS) $(SPLIBS) $(BIGORALIB) $(LDCOM) $(LIBCORE) \
	$(CLIBS) -lm

tpccload.ott: tpccload.o
	$(CC) $(CFLAGS) -o $@ $(INCLUDE) \
	-L$(SRCHOME)/lib tpccload.o \
	-lbench $(LIBOCIC) $(BIGORALIB) $(TTLIBS) $(CLIBS) -lm

tpcc.ost: c_drv.o c_trans.o tpccpl.o c_dump.o
	$(CC) $(CFLAGS) -o $@ $(INCLUDE) \
	-L$(SRCHOME)/lib c_drv.o c_trans.o tpccpl.o c_dump.o \
	 -lbench $(LIBOCIC) $(STLIBS) $(BIGORALIB) $(LDCOM) \
	$(LIBCORE) $(CLIBS) -lm
```

```
tpcc.ott: c_drv.o c_trans.o tpccpl.o c_dump.o
        $(CC) $(CFLAGS) -o $@ $(INCLUDE) \
        -L$(SRCHOME)/lib c_drv.o c_trans.o tpccpl.o c_dump.o \
        -lbench $(LIBOCIC) $(BIGORALIB) $(TTLIBS) $(CLIBS) -lm

test_drv: c_drv_val.o test_drv.o c_dump.o
        $(CC) $(CFLAGS) -o $@ $(INCLUDE) \
        -L$(SRCHOME)/lib c_drv_val.o test_drv.o c_dump.o \
         -lbench $(CLIBS) -lm

test_sample: c_drv.o test_sample.o c_dump.o
        $(CC) $(CFLAGS) -o $@ $(INCLUDE) \
        -L$(SRCHOME)/lib c_drv.o test_sample.o c_dump.o \
        -lbench $(CLIBS) -lm

test_tran.ost: test_tran.o c_trans.o tpccpl.o c_dump.o
        $(CC) $(CFLAGS) -o $@ $(INCLUDE) \
        -L$(SRCHOME)/lib test_tran.o c_trans.o tpccpl.o c_dump.o \
         -lbench $(LIBOCIC) $(STLIBS) $(BIGORALIB) $(LDCOM) \
        $(LIBCORE) $(CLIBS) -lm

test_tran.ott: test_tran.o c_trans.o tpccpl.o c_dump.o
        $(CC) $(CFLAGS) -o $@ $(INCLUDE) \
        -L$(SRCHOME)/lib test_tran.o c_trans.o tpccpl.o c_dump.o \
        -lbench $(LIBOCIC) $(BIGORALIB) $(TTLIBS) $(CLIBS) -lm

tpcccli: c_drv_tux.o c_dump.o
        (CFLAGS="$(CFLAGS)"; export CFLAGS; CC=$(CC); export CC; \
         buildclient -v -o $@ -f '-L$(SRCHOME)/lib c_drv_tux.o c_dump.o' \
           -f '-lbench' -l '-lm')

tpccsvr.ott: tpccsvr.o c_trans_tux.o tpccpl_tux.o
        (CFLAGS="$(CFLAGS)"; export CFLAGS; CC=$(CC); export CC; \
         buildserver -v -o $@ \
           -s NEWORDER,PAYMENT,ORDERSTATUS,DELIVERY,STOCKLEVEL \
           -f '-L$(SRCHOME)/lib tpccsvr.o c_trans_tux.o tpccpl_tux.o' \
           -l '-lbench $(LIBOCIC) $(BIGORALIB) $(TTLIBS) $(CLIBS) -lm')

tpccsvr.ost: tpccsvr.o c_trans_tux.o tpccpl_tux.o
        (CFLAGS="$(CFLAGS)"; export CFLAGS; CC=$(CC); export CC; \
         buildserver -v -o $@ \
           -s NEWORDER,PAYMENT,ORDERSTATUS,DELIVERY,STOCKLEVEL \
           -f '-L$(SRCHOME)/lib tpccsvr.o c_trans_tux.o tpccpl_tux.o' \
           -l '-lbench $(LIBOCIC) $(STLIBS) $(BIGORALIB) $(LDCOM)
$(LIBCORE)' \
            -l '$(CLIBS) -lm')

setup: tpccload.ost tpcc.ost tpcc.ott getrand 90per
        rm -f $(ORACLE_HOME)/bench/tpc/bin/tpccload
        ln -s $(ORACLE_HOME)/bench/tpc/tpcc/source/tpccload.ost \
              $(ORACLE_HOME)/bench/tpc/bin/tpccload
        rm -f $(ORACLE_HOME)/bench/tpc/bin/tpcc.ost
        ln -s $(ORACLE_HOME)/bench/tpc/tpcc/source/tpcc.ost \
              $(ORACLE_HOME)/bench/tpc/bin/tpcc.ost
        rm -f $(ORACLE_HOME)/bench/tpc/bin/tpcc.ott
        ln -s $(ORACLE_HOME)/bench/tpc/tpcc/source/tpcc.ott \
              $(ORACLE_HOME)/bench/tpc/bin/tpcc.ott
        rm -f $(ORACLE_HOME)/bench/tpc/bin/getrand
        ln -s $(ORACLE_HOME)/bench/tpc/tpcc/source/getrand \
              $(ORACLE_HOME)/bench/tpc/bin/getrand
        rm -f $(ORACLE_HOME)/bench/tpc/bin/90per
        ln -s $(ORACLE_HOME)/bench/tpc/tpcc/source/90per \
              $(ORACLE_HOME)/bench/tpc/bin/90per
        @-$(DOTARGS)
```

Appendix B - Database Design

loaddb/addfile.sh

```
#
# $Header: addfile.sh 7010000.2 94/10/12 13:34:31 plai Generic<base> $
Copyr (c) 1994 Oracle
#

#=====================================================================+
#        Copyright (c) 1994  Oracle Corp, Redwood Shores, CA          |
#                    OPEN SYSTEMS PERFORMANCE GROUP                   |
#                         All Rights Reserved                         |
#=====================================================================+
# FILENAME
#     addfile.sh
# DESCRIPTION
#     Add datafile to a tablespace.
# USAGE
#     addfile.sh <tablespace> <data file> <size>
#=====================================================================*/

sqldba <<!
   connect internal
   alter tablespace $1 add datafile '$2' size $3 reuse;
   exit;
!
```

loaddb/alter.sh

```
#
# $Header: alter.sh 7030100.1 95/07/17 14:36:20 plai Generic<base> $ Copyr
(c) 1995 Oracle
#

#=====================================================================+
#        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA          |
#                    OPEN SYSTEMS PERFORMANCE GROUP                   |
#                         All Rights Reserved                         |
#=====================================================================+
# FILENAME
#     alter.sh
# DESCRIPTION
#     Change next extent size for TPC-C tables and indexes.
# USAGE
#     alter.sh
#=====================================================================*/

sqlplus tpcc/tpcc <<!
   alter table history     storage (next 740M);
   alter cluster ccluster  storage (next  50M);
   alter cluster scluster  storage (next  50M);
   alter table orders      storage (next 620M);
   alter table order_line  storage (next 398M);
   alter table new_order   storage (next 310M);
   alter index iorders     storage (next 590M);
   alter index iorders2    storage (next 750M);
   alter index inew_order  storage (next 250M);
   alter index iorder_line storage (next 480M);
   alter index istock      storage (next  50M);
   alter index icustomer   storage (next  50M);
   alter index icustomer2  storage (next  50M);
   quit;
!
```

loaddb/cr_db.sh

```
# Modified for 600 warehouses
#
# Create database
#

sqldba lmode=y<<!
       set echo on
       connect internal
       startup  pfile=/home/oracle/dbs/inittpcc_0.ora nomount
       create database tpcc maxdatafiles 220
            maxlogfiles 32
            character set "US7ASCII"
           datafile '/dev/vx/rdsk/rootdg/system' size  608M reuse
           logfile  '/dev/vx/rdsk/rootdg/log1' size 2020M reuse,
                    '/dev/vx/rdsk/rootdg/log2' size 2020M reuse;

       connect system/manager;
   create rollback segment s1  storage (initial 600k minextents 2 next
600k);
   create rollback segment s2  storage (initial 600k minextents 2 next
600k);
   create rollback segment s3  storage (initial 600k minextents 2 next
600k);
```

```
   create rollback segment s4  storage (initial 600k minextents 2 next
600k);
   create rollback segment s5  storage (initial 600k minextents 2 next
600k);
   create rollback segment s6  storage (initial 600k minextents 2 next
600k);
   create rollback segment s7  storage (initial 600k minextents 2 next
600k);
   create rollback segment s8  storage (initial 600k minextents 2 next
600k);
   create rollback segment s9  storage (initial 600k minextents 2 next
600k);
   create rollback segment s10 storage (initial 600k minextents 2 next
600k);
disconnect;
connect internal;
shutdown;
exit;
!
```

loaddb/cr_ts.sh

```
# for 600 warehouses
#
# $Header: benchdb.sh 7030100.1 95/07/17 14:38:05 plai Generic<base> $
Copyr (c) 1993 Oracle
#
#
#=================================================================+
#        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA      |
#                    OPEN SYSTEMS PERFORMANCE GROUP               |
#                         All Rights Reserved                     |
#=================================================================+
# FILENAME
#      benchdb.sh
# DESCRIPTION
#      Usage: benchdb.sh [options]
#             -n           do not create new tpcc database
#             -c           do not run catalog scripts
#=================================================================
#


BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_DBA=$BENCH_HOME/tpcc/dba
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin

#
#  Startup database with params file that includes new rollback segments
#

sqldba <<!
        connect internal
        set termout on
        spool tablespaces
        startup pfile=/home/oracle/dbs/inittpcc.ora
        connect system/manager
        create tablespace roll datafile
           '/dev/vx/rdsk/rootdg/roll1'   size 619M reuse;
        create tablespace hist datafile
           '/dev/vx/rdsk/rootdg/hist1'   size 2011M reuse;
        create tablespace ware datafile
           '/dev/vx/rdsk/rootdg/ware1'   size   27M reuse;
        create tablespace cust datafile
           '/dev/vx/rdsk/rootdg/cust0'   size  100K reuse;
        create tablespace items datafile
           '/dev/vx/rdsk/rootdg/item1'   size   27M reuse;
        create tablespace ord datafile
           '/dev/vx/rdsk/rootdg/ord1'    size 1547M reuse;
        create tablespace nord datafile
           '/dev/vx/rdsk/rootdg/nord'   size  608M reuse;
        create tablespace ordl datafile
           '/dev/vx/rdsk/rootdg/ordl0'   size  100K reuse;
        create tablespace stocks datafile
           '/dev/vx/rdsk/rootdg/stk0'    size  100K reuse;
        create tablespace icust1 datafile
           '/dev/vx/rdsk/rootdg/icust1'  size  1216M reuse;
        create tablespace icust2 datafile
           '/dev/vx/rdsk/rootdg/icust2'  size 1584M reuse;
        create tablespace istk datafile
           '/dev/vx/rdsk/rootdg/istk1'   size 2045M reuse;
        create tablespace iord1 datafile
           '/dev/vx/rdsk/rootdg/iord1'   size  1376M reuse;
        create tablespace iord2 datafile
           '/dev/vx/rdsk/rootdg/iord2'   size 2027M reuse;
        create tablespace inord datafile
           '/dev/vx/rdsk/rootdg/inord1'  size  619M reuse;
        create tablespace iordl datafile
           '/dev/vx/rdsk/rootdg/iordl0'  size  100K reuse;
        create tablespace temp datafile
           '/dev/vx/rdsk/rootdg/temp0'   size  100K reuse;
       spool off;
       exit;
!
```

loaddb/add_datafiles_to_ts.sh

```
# for 600 warehouses
#
# $Header: benchdb.sh 7030100.1 95/07/17 14:38:05 plai Generic<base> $
Copyr (c) 1993 Oracle
#
#
#=================================================================+
#        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA      |
#                    OPEN SYSTEMS PERFORMANCE GROUP               |
#                         All Rights Reserved                     |
#=================================================================+
# FILENAME
#      benchdb.sh
# DESCRIPTION
#      Usage: benchdb.sh [options]
#             -n           do not create new tpcc database
#             -c           do not run catalog scripts
#=================================================================
#
```

```
BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_DBA=$BENCH_HOME/tpcc/dba
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin

#
#  Startup database with params file that includes new rollback segments
#

addfile.sh cust /dev/vx/rdsk/rootdg/cust1 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust2 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust3 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust4 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust5 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust6 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust7 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust8 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust9 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust10 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust11 1832M &
addfile.sh cust /dev/vx/rdsk/rootdg/cust12 1832M &
wait
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl1 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl2 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl3 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl4 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl5 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl6 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl7 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl8 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl9 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl10 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl11 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl12 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl13 2011M &
addfile.sh ordl /dev/vx/rdsk/rootdg/ordl14 2011M &
wait
addfile.sh stocks /dev/vx/rdsk/rootdg/stk1 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk2 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk3 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk4 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk5 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk6 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk7 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk8 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk9 1687M &
wait
addfile.sh stocks /dev/vx/rdsk/rootdg/stk10 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk11 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk12 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk13 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk14 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk15 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk16 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk17 1687M &
addfile.sh stocks /dev/vx/rdsk/rootdg/stk18 1687M &
wait
addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl1 1993M &
addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl2 1993M &
addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl3 1993M &
addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl4 1993M &
addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl5 1993M &
addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl6 1993M &
addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl7 1993M &
wait
addfile.sh temp   /dev/vx/rdsk/rootdg/temp1 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp2 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp3 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp4 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp5 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp6 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp7 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp8 1773M &

wait
```

loaddb/pload.sh

```
#
# $Header: pload.sh 7030100.1 95/07/17 14:37:24 plai Generic<base> $ Copyr
(c) 1995 Oracle
#

#
#+=================================================================+
#        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA       |
#                    OPEN SYSTEMS PERFORMANCE GROUP                |
#                         All Rights Reserved                      |
#+=================================================================+
# FILENAME
#      pload.sh
# DESCRIPTION
#      Usage: pload.sh [options]
#             -mu <multiplier>     (# of warehouses)
#=================================================================
#
BENCH_HOME=$ORACLE_HOME/bench/tpc
BENCH_GEN=$ORACLE_HOME/bench/gen
GEN_SQL=$BENCH_GEN/sql
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin
TPCC_STORE=$BENCH_HOME/tpcc/stored_proc
TPCC_LOADER=$BENCH_HOME/tpcc/loader
TPCC_SCRIPTS=$BENCH_HOME/tpcc/scripts

PATH=${PATH}:$TPCC_SOURCE
export PATH

if echo "\c" | grep c >/dev/null 2>&1; then
    N='-n'
else
    C='\c'
fi
export N C

MULT=600
```

```
while [ "$#" != "0" ]
do
   case $1 in
       -mu) shift
            if [ "$1" != "" ]
            then
               MULT=$1
               shift
            fi
            ;;
       -nd) shift
            NO_DB="y"
            ;;
       -nt) shift
            NO_TAB="y"
            ;;
       -nx) shift
            NO_IND="y"
             ;;
         *) echo "Bag arg: $1"
            exit 1;
            ;;
   esac
done

if [ "$MULT" = "" ]
then
   echo $N "Database multiplier (# of warehouses)? [1]" $C
   read MULT
   if [ "$MULT" = "" ]
   then
       MULT=1
   fi
fi

LDIR1=/load1/data1
LDIR2=/load1/data2
LDIR3=/load2/data3
LDIR4=/load2/data4
LDIR5=/load3/data5
LDIR6=/load3/data6
LDIR7=/load4/data7
LDIR8=/load4/data8
LDIR9=/load5/data9
LDIR10=/load5/data10
LDIR11=/load6/data11
LDIR12=/load6/data12
LDIR13=/load7/data13
LDIR14=/load7/data14

#
#  Load history table
#

tpccload -M $MULT -h -g -b   1 -e  40 > ${LDIR1}/hist1.dat &
tpccload -M $MULT -h -g -b  41 -e  80 > ${LDIR2}/hist2.dat &
tpccload -M $MULT -h -g -b  81 -e 120 > ${LDIR3}/hist3.dat &
tpccload -M $MULT -h -g -b 121 -e 160 > ${LDIR4}/hist4.dat &
tpccload -M $MULT -h -g -b 161 -e 200 > ${LDIR5}/hist5.dat &
tpccload -M $MULT -h -g -b 201 -e 240 > ${LDIR6}/hist6.dat &
tpccload -M $MULT -h -g -b 241 -e 280 > ${LDIR7}/hist7.dat &
tpccload -M $MULT -h -g -b 281 -e 320 > ${LDIR8}/hist8.dat &
tpccload -M $MULT -h -g -b 321 -e 360 > ${LDIR9}/hist9.dat &
tpccload -M $MULT -h -g -b 361 -e 400 > ${LDIR10}/hist10.dat &
tpccload -M $MULT -h -g -b 401 -e 440 > ${LDIR11}/hist11.dat &
tpccload -M $MULT -h -g -b 441 -e 480 > ${LDIR12}/hist12.dat &
tpccload -M $MULT -h -g -b 481 -e 520 > ${LDIR13}/hist13.dat &
tpccload -M $MULT -h -g -b 521 -e 560 > ${LDIR14}/hist14.dat &
tpccload -M $MULT -h -g -b 561 -e 600 > ${LDIR14}/hist15.dat &
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist1.log \
       bad=hist1.bad data=${LDIR1}/hist1.dat discard=hist1.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist2.log \
       bad=hist2.bad data=${LDIR2}/hist2.dat discard=hist2.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist3.log \
       bad=hist3.bad data=${LDIR3}/hist3.dat discard=hist3.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist4.log \
       bad=hist4.bad data=${LDIR4}/hist4.dat discard=hist4.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist5.log \
       bad=hist5.bad data=${LDIR5}/hist5.dat discard=hist5.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist6.log \
       bad=hist6.bad data=${LDIR6}/hist6.dat discard=hist6.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist7.log \
       bad=hist7.bad data=${LDIR7}/hist7.dat discard=hist7.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist8.log \
       bad=hist8.bad data=${LDIR8}/hist8.dat discard=hist8.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist9.log \
       bad=hist9.bad data=${LDIR9}/hist9.dat discard=hist9.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist10.log \
       bad=hist10.bad data=${LDIR10}/hist10.dat discard=hist10.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist11.log \
       bad=hist11.bad data=${LDIR11}/hist11.dat discard=hist11.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist12.log \
       bad=hist12.bad data=${LDIR12}/hist12.dat discard=hist12.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist13.log \
       bad=hist13.bad data=${LDIR13}/hist13.dat discard=hist13.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist14.log \
       bad=hist14.bad data=${LDIR14}/hist14.dat discard=hist14.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl log=hist15.log \
       bad=hist15.bad data=${LDIR14}/hist15.dat discard=hist15.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/hist1 &
wait
rm -f ${LDIR1}/hist1.dat \
      ${LDIR2}/hist2.dat \
      ${LDIR3}/hist3.dat \
      ${LDIR4}/hist4.dat \
      ${LDIR5}/hist5.dat \
```

```
      ${LDIR6}/hist6.dat \
      ${LDIR7}/hist7.dat \
      ${LDIR8}/hist8.dat \
      ${LDIR9}/hist9.dat \
      ${LDIR10}/hist10.dat \
      ${LDIR11}/hist11.dat \
      ${LDIR12}/hist12.dat \
      ${LDIR13}/hist13.dat \
      ${LDIR14}/hist14.dat \
      ${LDIR14}/hist15.dat

#
#  Load new-order table
#

tpccload -M $MULT -n -g > ${LDIR1}/neword1.dat
sqlldr tpcc/tpcc control=$TPCC_LOADER/neword.ctl log=neword1.log \
       bad=neword1.bad data=${LDIR1}/neword1.dat discard=neword1.dsc
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/nord1
rm -f ${LDIR1}/neword1.dat

#
#  Load order and order-line table
#

tpccload -M $MULT -o ${LDIR1}/ordline1.dat -g -b   1 -e  43 >
${LDIR1}/order1.dat &
tpccload -M $MULT -o ${LDIR2}/ordline2.dat -g -b  44 -e  86 >
${LDIR2}/order2.dat &
tpccload -M $MULT -o ${LDIR3}/ordline3.dat -g -b  87 -e  129 >
${LDIR3}/order3.dat &
tpccload -M $MULT -o ${LDIR4}/ordline4.dat -g -b  130 -e 172 >
${LDIR4}/order4.dat &
tpccload -M $MULT -o ${LDIR5}/ordline5.dat -g -b  173 -e 215 >
${LDIR5}/order5.dat &
tpccload -M $MULT -o ${LDIR6}/ordline6.dat -g -b 216 -e 258 >
${LDIR6}/order6.dat &
tpccload -M $MULT -o ${LDIR7}/ordline7.dat -g -b 259 -e 301 >
${LDIR7}/order7.dat &
tpccload -M $MULT -o ${LDIR8}/ordline8.dat -g -b 302 -e 344 >
${LDIR8}/order8.dat &
tpccload -M $MULT -o ${LDIR9}/ordline9.dat -g -b 345 -e 387 >
${LDIR9}/order9.dat &
tpccload -M $MULT -o ${LDIR10}/ordline10.dat -g -b 388 -e 430 >
${LDIR10}/order10.dat &
tpccload -M $MULT -o ${LDIR11}/ordline11.dat -g -b 431 -e 473 >
${LDIR11}/order11.dat &
tpccload -M $MULT -o ${LDIR12}/ordline12.dat -g -b 474 -e 516 >
${LDIR12}/order12.dat &
tpccload -M $MULT -o ${LDIR13}/ordline13.dat -g -b 517 -e 558 >
${LDIR13}/order13.dat &
tpccload -M $MULT -o ${LDIR14}/ordline14.dat -g -b 559 -e 600 >
${LDIR14}/order14.dat &
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order1.log \
       bad=order1.bad data=${LDIR1}/order1.dat discard=order1.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order2.log \
       bad=order2.bad data=${LDIR2}/order2.dat discard=order2.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order3.log \
       bad=order3.bad data=${LDIR3}/order3.dat discard=order3.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order4.log \
       bad=order4.bad data=${LDIR4}/order4.dat discard=order4.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order5.log \
       bad=order5.bad data=${LDIR5}/order5.dat discard=order5.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order6.log \
       bad=order6.bad data=${LDIR6}/order6.dat discard=order6.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order7.log \
       bad=order7.bad data=${LDIR7}/order7.dat discard=order7.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order8.log \
       bad=order8.bad data=${LDIR8}/order8.dat discard=order8.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order9.log \
       bad=order9.bad data=${LDIR9}/order9.dat discard=order9.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order10.log \
       bad=order10.bad data=${LDIR10}/order10.dat discard=order10.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order11.log \
       bad=order11.bad data=${LDIR11}/order11.dat discard=order11.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order12.log \
       bad=order12.bad data=${LDIR12}/order12.dat discard=order12.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order13.log \
       bad=order13.bad data=${LDIR13}/order13.dat discard=order13.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl log=order14.log \
       bad=order14.bad data=${LDIR14}/order14.dat discard=order14.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ord1 &
wait
rm -f ${LDIR1}/order1.dat \
      ${LDIR2}/order2.dat \
      ${LDIR3}/order3.dat \
      ${LDIR4}/order4.dat \
      ${LDIR5}/order5.dat \
      ${LDIR6}/order6.dat \
      ${LDIR7}/order7.dat \
      ${LDIR8}/order8.dat \
      ${LDIR9}/order9.dat \
      ${LDIR10}/order10.dat \
      ${LDIR11}/order11.dat \
      ${LDIR12}/order12.dat \
      ${LDIR13}/order13.dat \
      ${LDIR14}/order14.dat
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline1.log \
       bad=ordline1.bad data=${LDIR1}/ordline1.dat discard=ordline1.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline2.log \
       bad=ordline2.bad data=${LDIR2}/ordline2.dat discard=ordline2.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl1 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline3.log \
       bad=ordline3.bad data=${LDIR3}/ordline3.dat discard=ordline3.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl2 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline4.log \
       bad=ordline4.bad data=${LDIR4}/ordline4.dat discard=ordline4.dsc &
```

```
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl2 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline5.log \
       bad=ordline5.bad data=${LDIR5}/ordline5.dat discard=ordline5.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl3 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline6.log \
       bad=ordline6.bad data=${LDIR6}/ordline6.dat discard=ordline6.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl3 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline7.log \
       bad=ordline7.bad data=${LDIR7}/ordline7.dat discard=ordline7.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl4 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline8.log \
       bad=ordline8.bad data=${LDIR8}/ordline8.dat discard=ordline8.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl4 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline9.log \
       bad=ordline9.bad data=${LDIR9}/ordline9.dat discard=ordline9.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl5 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline10.log \
       bad=ordline10.bad data=${LDIR10}/ordline10.dat
discard=ordline10.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl5 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline11.log \
       bad=ordline11.bad data=${LDIR11}/ordline11.dat
discard=ordline11.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl6 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline12.log \
       bad=ordline12.bad data=${LDIR12}/ordline12.dat
discard=ordline12.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl6 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline13.log \
       bad=ordline13.bad data=${LDIR13}/ordline13.dat
discard=ordline13.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl7 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/ordline.ctl log=ordline14.log \
       bad=ordline14.bad data=${LDIR14}/ordline14.dat
discard=ordline14.dsc &
     #  file=${ORACLE_HOME}/dbs/tpcc_disks/ordl7 &
wait
rm -f ${LDIR1}/ordline1.dat \
      ${LDIR2}/ordline2.dat \
      ${LDIR3}/ordline3.dat \
      ${LDIR4}/ordline4.dat \
      ${LDIR5}/ordline5.dat \
      ${LDIR6}/ordline6.dat \
      ${LDIR7}/ordline7.dat \
      ${LDIR8}/ordline8.dat \
      ${LDIR9}/ordline9.dat \
      ${LDIR10}/ordline10.dat \
      ${LDIR11}/ordline11.dat \
      ${LDIR12}/ordline12.dat \
      ${LDIR13}/ordline13.dat \
      ${LDIR14}/ordline14.dat
```

loaddb/cr_clust_stk_cust.sh

```
BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_DBA=$BENCH_HOME/tpcc/dba
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin


sqldba <<!
        set echo off;
        connect sys/change_on_install;
        @?/rdbms/admin/catalog;
        @?/rdbms/admin/catproc;
        @?/rdbms/admin/catparr;
        exit;
!
   sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_tab2 > tab2.out 2>&1 &
   sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_tab3 > tab3.out 2>&1 &
wait
```

loaddb/cr_index.sh

```
#
# $Header: benchsetup.sh 7030100.1 95/07/17 14:37:46 plai Generic<base> $
Copyr (c) 1993 Oracle
#
#
#=================================================================+
#        Copyright (c) 1995  Oracle Corp, Redwood Shores, CA      |
#                    OPEN SYSTEMS PERFORMANCE GROUP               |
#                         All Rights Reserved                     |
#=================================================================+
# FILENAME
#      benchsetup.sh
# DESCRIPTION
#      Usage: benchsetup.sh [options]
#             -mu <multiplier>     (# of warehouses)
#             -nd                  do not run benchdb.sh
#             -nt                  do not create tpcc tables
#             -nx                  do not create index for tpcc tables
#=================================================================
#

BENCH_HOME=$ORACLE_HOME/bench/tpc
BENCH_GEN=$ORACLE_HOME/bench/gen
GEN_SQL=$BENCH_GEN/sql
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin
TPCC_STORE=$BENCH_HOME/tpcc/stored_proc
TPCC_LOADER=$BENCH_HOME/tpcc/loader
TPCC_SCRIPTS=$BENCH_HOME/tpcc/scripts

PATH=${PATH}:$TPCC_SOURCE
export PATH

MULT=600

#
#  Load warehouse, district, item tables
#

tpccload -M $MULT -w
tpccload -M $MULT -d
tpccload -M $MULT -i
```

```
#
#  Create indexes
#

if [ "$NO_IND" = "" ]
then
   sqldba <<!
      connect internal
      alter tablespace temp
         default storage (initial 50M next 50M pctincrease 0);
      exit;
!

sqlplus system/manager <<!
   alter user tpcc temporary tablespace temp;
   quit;
!

   sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_ix1
   sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_ix2
   sqldba <<!
      connect internal
      alter tablespace temp
         default storage (initial 20K next 20K pctincrease 50);
      exit;
!

alter.sh
# Need to lock/unlock table index(s)

#
#  Analyze tables and indexes
#

sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_ana

#
#  Create table for processing benchmark results
#

sqlplus sys/change_on_install @$GEN_SQL/orst_cre
sqlplus sys/change_on_install @$TPCC_SQL/c_stat
sqlplus sys/change_on_install @$GEN_SQL/pst_c

#
#  Create stored procedures
#

sqlplus tpcc/tpcc @$TPCC_STORE/new
sqlplus tpcc/tpcc @$TPCC_STORE/pay
sqlplus tpcc/tpcc @$TPCC_STORE/ord
sqlplus tpcc/tpcc @$TPCC_STORE/del
sqlplus tpcc/tpcc @$TPCC_STORE/sto

#
#  Get some statistics
#

cd $TPCC_SCRIPTS/utils

ext_all.sh > ext_all.out 2>&1

space_init.sh
space_get.sh 1440 120
space_rpt.sh

cd $TPCC_SCRIPTS/samples

sqlplus system/manager <<!
   alter user tpcc temporary tablespace system;
   quit;
!

sqlplus sys/change_on_install <<!
   grant execute on dbms_lock to public;
   grant execute on dbms_pipe to public;
   quit;
!

sqlplus tpcc/tpcc @$BENCH_HOME/tpcc/audit/sql/plsql_mon
sqlplus tpcc/tpcc @$BENCH_HOME/tpcc/audit/sql/cre_tab

#
#  Shutdown database
#

sqldba <<!
   connect internal;
   alter system switch logfile;
   alter system switch logfile;
   shutdown;
   exit;
!
```

loaddb/cr_iordl.sh

```
BENCH_HOME=$ORACLE_HOME/bench/tpc
BENCH_GEN=$ORACLE_HOME/bench/gen
GEN_SQL=$BENCH_GEN/sql
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin
TPCC_STORE=$BENCH_HOME/tpcc/stored_proc
TPCC_LOADER=$BENCH_HOME/tpcc/loader
TPCC_SCRIPTS=$BENCH_HOME/tpcc/scripts

PATH=${PATH}:$TPCC_SOURCE
export PATH

MULT=600


sqlplus system/manager <<!
   alter user tpcc temporary tablespace temp;
   quit;
!
   sqldba <<!
      connect internal
```

```
      alter tablespace temp
         default storage (initial 50M next 50M pctincrease 0);
      exit;
!

sqlplus tpcc/tpcc @$TPCC_SQL/cr_iordl

   sqldba <<!
      connect internal
      alter tablespace temp
         default storage (initial 20K next 20K pctincrease 50);
      exit;
!

sqlplus system/manager <<!
   alter user tpcc temporary tablespace system;
   quit;
!

sqlplus tpcc/tpcc @$TPCC_SQL/unlock
sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_ana
sqlplus tpcc/tpcc @$TPCC_SQL/lock
#  Shutdown database
#

sqldba <<!
   connect internal;
   alter system switch logfile;
   alter system switch logfile;
   shutdown;
   exit;
!
```

loaddb/cr_stk.sh

```
BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_DBA=$BENCH_HOME/tpcc/dba
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin

MULTI=600
sqldba <<!
        connect internal
       set termout on
       set timing on
        startup pfile=/home/oracle/dbs/init_load.ora
        connect system/manager
        create tablespace stocks datafile
           '/dev/vx/rdsk/rootdg/stk0'    size  100K reuse;
      exit;
!

 addfile.sh stocks /dev/vx/rdsk/rootdg/stk1 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk2 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk3 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk4 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk5 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk6 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk7 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk8 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk9 1687M &
wait
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk10 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk11 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk12 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk13 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk14 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk15 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk16 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk17 1687M &
 addfile.sh stocks /dev/vx/rdsk/rootdg/stk18 1687M &
 wait
```

loaddb/cr_stk_clust.sh

```
BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_DBA=$BENCH_HOME/tpcc/dba
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin

sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_tab3 > tab3.out 2>&1 &
wait
```

loaddb/cr_temp_ts.sh

```
BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_DBA=$BENCH_HOME/tpcc/dba
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin

sqldba <<!
        connect internal
       set termout on
        create tablespace temp datafile
           '/dev/vx/rdsk/rootdg/temp0'   size  100K reuse;
      exit;
!
addfile.sh temp   /dev/vx/rdsk/rootdg/temp1 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp2 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp3 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp4 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp5 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp6 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp7 1773M &
addfile.sh temp   /dev/vx/rdsk/rootdg/temp8 1773M &

wait
```

loaddb/crdb2_ware.sql

```
# Modified for 600 warehouses
```

```
sqldba lmode=y <<!
connect internal;
set termout on
set echo on
spool crdb2_ware
startup pfile=/home/oracle/dbs/inittpcc_0.ora
connect sys/change_on_install
@/home/oracle/rdbms/admin/catalog.sql

connect system/manager
@$ORACLE_HOME/rdbms/admin/catdbsyn.sql

spool off
```

loaddb/iordl.sh

```
sqldba lmode=y <<!
connect internal;
set spool on;
set termout on;
set timing on;
drop tablespace iordl including contents;
create tablespace iordl datafile
   '/dev/vx/rdsk/rootdg/iordl0'  size  100K reuse;
spool off;
exit;
!
 addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl1 1993M &
 addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl2 1993M &
 addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl3 1993M &
 addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl4 1993M &
 addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl5 1993M &
 addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl6 1993M &
 addfile.sh iordl  /dev/vx/rdsk/rootdg/iordl7 1993M &
 wait
```

loaddb/load_cust.sh

```
BENCH_HOME=$ORACLE_HOME/bench/tpc
BENCH_GEN=$ORACLE_HOME/bench/gen
GEN_SQL=$BENCH_GEN/sql
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin
TPCC_STORE=$BENCH_HOME/tpcc/stored_proc
TPCC_LOADER=$BENCH_HOME/tpcc/loader
TPCC_SCRIPTS=$BENCH_HOME/tpcc/scripts
cout=$TPCC_SCRIPTS/samples/cout
export cout

PATH=${PATH}:$TPCC_SOURCE
export PATH

MULT=600

#
#  Load customer table
#

tpccload -M $MULT -c -b     1 -e  10 > out/c1.out 2>&1 &
tpccload -M $MULT -c -b    51 -e  60  > out/c51.out 2>&1 &
tpccload -M $MULT -c -b   101 -e  110  > out/c101.out 2>&1 &
tpccload -M $MULT -c -b   151 -e  160  > out/c151.out 2>&1 &
tpccload -M $MULT -c -b   201 -e  210  > out/c201.out 2>&1 &
tpccload -M $MULT -c -b   251 -e  260  > out/c251.out 2>&1 &
tpccload -M $MULT -c -b   301 -e  310  > out/c301.out 2>&1 &
tpccload -M $MULT -c -b   351 -e  360  > out/c351.out 2>&1 &
tpccload -M $MULT -c -b   401 -e  410  > out/c401.out 2>&1 &
tpccload -M $MULT -c -b   451 -e  460  > out/c451.out 2>&1 &
tpccload -M $MULT -c -b   501 -e  510  > out/c501.out 2>&1 &
tpccload -M $MULT -c -b   551 -e  560  > out/c551.out 2>&1 &
wait
sqldba lmode=y <<!
connect internal;
alter system checkpoint local;
exit;
!
sleep 200
tpccload -M $MULT -c -b    11 -e  20 > out/c11.out 2>&1 &
tpccload -M $MULT -c -b    61 -e  70 > out/c61.out 2>&1 &
tpccload -M $MULT -c -b   111 -e 120 > out/c111.out 2>&1 &
tpccload -M $MULT -c -b   161 -e  170  > out/c161.out 2>&1 &
tpccload -M $MULT -c -b   211 -e  220  > out/c211.out 2>&1 &
tpccload -M $MULT -c -b   261 -e  270  > out/c261.out 2>&1 &
tpccload -M $MULT -c -b   311 -e  320  > out/c311.out 2>&1 &
tpccload -M $MULT -c -b   361 -e  370  > out/c361.out 2>&1 &
tpccload -M $MULT -c -b   411 -e  420  > out/c411.out 2>&1 &
tpccload -M $MULT -c -b   461 -e  470  > out/c461.out 2>&1 &
tpccload -M $MULT -c -b   511 -e  520  > out/c511.out 2>&1 &
tpccload -M $MULT -c -b   561 -e  570  > out/c561.out 2>&1 &
wait
sqldba lmode=y <<!
connect internal;
alter system checkpoint local;
exit;
!
sleep 200
tpccload -M $MULT -c -b  21 -e  30 > out/c21.out 2>&1 &
tpccload -M $MULT -c -b  71 -e  80 > out/c71.out 2>&1 &
tpccload -M $MULT -c -b   121 -e  130  > out/c121.out 2>&1 &
tpccload -M $MULT -c -b   171 -e  180  > out/c171.out 2>&1 &
tpccload -M $MULT -c -b   221 -e  230  > out/c221.out 2>&1 &
tpccload -M $MULT -c -b   271 -e  280  > out/c271.out 2>&1 &
tpccload -M $MULT -c -b   321 -e  330  > out/c321.out 2>&1 &
tpccload -M $MULT -c -b   371 -e  380  > out/c371.out 2>&1 &
tpccload -M $MULT -c -b   421 -e  430  > out/c421.out 2>&1 &
tpccload -M $MULT -c -b   471 -e  480  > out/c471.out 2>&1 &
tpccload -M $MULT -c -b   521 -e  530  > out/c521.out 2>&1 &
tpccload -M $MULT -c -b   571 -e  580  > out/c571.out 2>&1 &
wait
sqldba lmode=y <<!
connect internal;
alter system checkpoint local;
exit;
!
sleep 200
tpccload -M $MULT -c -b  31 -e  40 > out/c31.out 2>&1 &
tpccload -M $MULT -c -b  81 -e  90 > out/c81.out 2>&1 &
```

```
tpccload -M $MULT -c -b   131 -e  140  > out/c131.out 2>&1 &
tpccload -M $MULT -c -b   181 -e  190  > out/c181.out 2>&1 &
tpccload -M $MULT -c -b   231 -e  240  > out/c231.out 2>&1 &
tpccload -M $MULT -c -b   281 -e  290  > out/c281.out 2>&1 &
tpccload -M $MULT -c -b   331 -e  340  > out/c331.out 2>&1 &
tpccload -M $MULT -c -b   381 -e  390  > out/c381.out 2>&1 &
tpccload -M $MULT -c -b   431 -e  440  > out/c431.out 2>&1 &
tpccload -M $MULT -c -b   481 -e  490  > out/c481.out 2>&1 &
tpccload -M $MULT -c -b   531 -e  540  > out/c531.out 2>&1 &
tpccload -M $MULT -c -b   581 -e  590  > out/c581.out 2>&1 &
wait
sqldba lmode=y <<!
connect internal;
alter system checkpoint local;
exit;
!
sleep 200
tpccload -M $MULT -c -b  41 -e  50 > out/c41.out 2>&1 &
tpccload -M $MULT -c -b  91 -e 100 > out/c91.out 2>&1 &
tpccload -M $MULT -c -b   141 -e  150  > out/c141.out 2>&1 &
tpccload -M $MULT -c -b   191 -e  200  > out/c191.out 2>&1 &
tpccload -M $MULT -c -b   241 -e  250  > out/c241.out 2>&1 &
tpccload -M $MULT -c -b   291 -e  300  > out/c291.out 2>&1 &
tpccload -M $MULT -c -b   341 -e  350  > out/c341.out 2>&1 &
tpccload -M $MULT -c -b   391 -e  400  > out/c391.out 2>&1 &
tpccload -M $MULT -c -b   441 -e  450  > out/c441.out 2>&1 &
tpccload -M $MULT -c -b   491 -e  500  > out/c491.out 2>&1 &
tpccload -M $MULT -c -b   541 -e  550  > out/c541.out 2>&1 &
tpccload -M $MULT -c -b   591 -e  600  > out/c591.out 2>&1 &
wait
sqldba lmode=y <<!
connect internal;
alter system checkpoint local;
exit;
!
```

loaddb/load_stk_cr_indx.sh

```
BENCH_HOME=$ORACLE_HOME/bench/tpc
BENCH_GEN=$ORACLE_HOME/bench/gen
GEN_SQL=$BENCH_GEN/sql
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin
TPCC_STORE=$BENCH_HOME/tpcc/stored_proc
TPCC_LOADER=$BENCH_HOME/tpcc/loader
TPCC_SCRIPTS=$BENCH_HOME/tpcc/scripts

PATH=${PATH}:$TPCC_SOURCE
export PATH

MULT=600


sqlplus system/manager <<!
   alter user tpcc temporary tablespace temp;
   quit;
!
#load_stock.sh


   sqldba <<!
      connect internal
      alter tablespace temp
         default storage (initial 50M next 50M pctincrease 0);
      exit;
!

sqldba <<!
   connect internal;
   alter system switch logfile;
   shutdown;
   startup pfile=/home/oracle/dbs/init_load.ora;
   exit;
!
sqlplus tpcc/tpcc @$TPCC_SQL/stkidx

   sqldba <<!
      connect internal
      alter tablespace temp
         default storage (initial 20K next 20K pctincrease 50);
      exit;
!

sqlplus system/manager <<!
   alter user tpcc temporary tablespace system;
   quit;
!

#  Shutdown database
#

sqldba <<!
   connect internal;
   alter system switch logfile;
   alter system switch logfile;
   shutdown;
   exit;
!
```

loaddb/load_stock.sh

```
BENCH_HOME=$ORACLE_HOME/bench/tpc
BENCH_GEN=$ORACLE_HOME/bench/gen
GEN_SQL=$BENCH_GEN/sql
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_SQL=$BENCH_HOME/tpcc/sql
TPCC_OUTPUT=$BENCH_HOME/tpcc/output
TPCC_ADMIN=$BENCH_HOME/tpcc/admin
TPCC_STORE=$BENCH_HOME/tpcc/stored_proc
TPCC_LOADER=$BENCH_HOME/tpcc/loader
TPCC_SCRIPTS=$BENCH_HOME/tpcc/scripts

PATH=${PATH}:$TPCC_SOURCE
export PATH

MULT=600


tpccload -M $MULT -S -j      1 -k   2500 >out/stk1.out 2>&1 &
```

```
tpccload -M $MULT -S -j   2501 -k   5000 >out/stk2.out 2>&1 &
tpccload -M $MULT -S -j   5001 -k   7500 >out/stk3.out 2>&1 &
tpccload -M $MULT -S -j   7501 -k  10000 >out/stk4.out 2>&1 &
tpccload -M $MULT -S -j  10001 -k  12500 >out/stk5.out 2>&1 &
tpccload -M $MULT -S -j  12501 -k  15000 >out/stk6.out 2>&1 &
tpccload -M $MULT -S -j  15001 -k  17500 >out/stk7.out 2>&1 &
tpccload -M $MULT -S -j  17501 -k  20000 >out/stk8.out 2>&1 &
tpccload -M $MULT -S -j  20001 -k  22500 >out/stk9.out 2>&1 &
tpccload -M $MULT -S -j  22501 -k  25000 >out/stk10.out 2>&1 &
wait
tpccload -M $MULT -S -j  25001 -k  27500 >out/stk11.out 2>&1 &
tpccload -M $MULT -S -j  27501 -k  30000 >out/stk12.out 2>&1 &
tpccload -M $MULT -S -j  30001 -k  32500 >out/stk13.out 2>&1 &
tpccload -M $MULT -S -j  32501 -k  35000 >out/stk14.out 2>&1 &
tpccload -M $MULT -S -j  35001 -k  37500 >out/stk15.out 2>&1 &
tpccload -M $MULT -S -j  37501 -k  40000 >out/stk16.out 2>&1 &
tpccload -M $MULT -S -j  40001 -k  42500 >out/stk17.out 2>&1 &
tpccload -M $MULT -S -j  42501 -k  45000 >out/stk18.out 2>&1 &
tpccload -M $MULT -S -j  45001 -k  47500 >out/stk19.out 2>&1 &
tpccload -M $MULT -S -j  47501 -k  50000 >out/stk20.out 2>&1 &
wait
tpccload -M $MULT -S -j  50001 -k  52500 >out/stk21.out 2>&1 &
tpccload -M $MULT -S -j  52501 -k  55000 >out/stk22.out 2>&1 &
tpccload -M $MULT -S -j  55001 -k  57500 >out/stk23.out 2>&1 &
tpccload -M $MULT -S -j  57501 -k  60000 >out/stk24.out 2>&1 &
tpccload -M $MULT -S -j  60001 -k  62500 >out/stk25.out 2>&1 &
tpccload -M $MULT -S -j  62501 -k  65000 >out/stk26.out 2>&1 &
tpccload -M $MULT -S -j  65001 -k  67500 >out/stk27.out 2>&1 &
tpccload -M $MULT -S -j  67501 -k  70000 >out/stk28.out 2>&1 &
tpccload -M $MULT -S -j  70001 -k  72500 >out/stk29.out 2>&1 &
tpccload -M $MULT -S -j  72501 -k  75000 >out/stk30.out 2>&1 &
wait
tpccload -M $MULT -S -j  75001 -k  77500 >out/stk31.out 2>&1 &
tpccload -M $MULT -S -j  77501 -k  80000 >out/stk32.out 2>&1 &
tpccload -M $MULT -S -j  80001 -k  82500 >out/stk33.out 2>&1 &
tpccload -M $MULT -S -j  82501 -k  85000 >out/stk34.out 2>&1 &
tpccload -M $MULT -S -j  85001 -k  87500 >out/stk35.out 2>&1 &
tpccload -M $MULT -S -j  87501 -k  90000 >out/stk36.out 2>&1 &
tpccload -M $MULT -S -j  90001 -k  92500 >out/stk37.out 2>&1 &
tpccload -M $MULT -S -j  92501 -k  95000 >out/stk38.out 2>&1 &
tpccload -M $MULT -S -j  95001 -k  97500 >out/stk39.out 2>&1 &
tpccload -M $MULT -S -j  97501 -k 100000 >out/stk40.out 2>&1 &

wait
```

bench/tpc/tpcc/sql/lock.sql

```
spool lock;
alter table warehouse enable table lock;
alter table district enable table lock;
alter table history enable table lock;
alter table stock enable table lock;
alter table customer enable table lock;
alter table orders enable table lock;
alter table new_order enable table lock;
alter table order_line enable table lock;
alter table item enable table lock;
commit work;
spool off;
exit;
```

bench/tpc/tpcc/sql/tpcc_ana.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       tpcc_ana.sql
rem  DESCRIPTION
rem       Analyze all tables and indexes of TPC-C database.
rem ====================================================================
rem

spool tpcc_ana;
set timing on;
analyze table warehouse compute statistics;
analyze table district compute statistics;
analyze table item estimate statistics;
analyze table history estimate statistics;
analyze table customer estimate statistics;
analyze table stock estimate statistics;
analyze table orders estimate statistics;
analyze table new_order estimate statistics;
analyze table order_line estimate statistics;
analyze cluster icluster estimate statistics;
analyze cluster scluster estimate statistics;
analyze cluster ccluster estimate statistics;
analyze index iwarehouse compute statistics;
analyze index idistrict compute statistics;
analyze index icustomer estimate statistics;
analyze index icustomer2 estimate statistics;
analyze index istock estimate statistics;
analyze index iitem estimate statistics;
analyze index iorders estimate statistics;
analyze index iorders2 estimate statistics;
analyze index inew_order estimate statistics;
analyze index iorder_line estimate statistics;
spool off;
quit;
```

bench/tpc/tpcc/sql/tpcc_iordl.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       tpcc_ix2.sql
rem  DESCRIPTION
rem       Create indexes for TPC-C database.
rem ====================================================================
rem
```

```
spool iordl;
set timing on

drop index iorder_line;

create unique index iorder_line on order_line(ol_w_id, ol_d_id, ol_o_id,
ol_number)
tablespace iordl
initrans 4
parallel 10
pctfree 1
storage (initial 50M next 50M pctincrease 0
freelist groups 13 freelists 24);
exit;
```

bench/tpc/tpcc/sql/tpcc_ix1.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       tpcc_ix1.sql
rem  DESCRIPTION
rem       Create indexes for TPC-C database.
rem ================================================================
rem

spool tpcc_ix1;
set timing on;
drop index iwarehouse;
drop index idistrict;
drop index icustomer;
drop index icustomer2;
drop index istock;
drop index iitem;

create unique index iwarehouse on warehouse(w_id)
       tablespace ware
       initrans 3
       storage (initial 200K next 20K pctincrease 0) pctfree 1;

create unique index idistrict on district(d_w_id, d_id)
       tablespace ware
       initrans 3
       storage (initial 2000K next 60K pctincrease 0) pctfree 1;

create unique index iitem on item(i_id)
       tablespace items
       storage (initial 2000K next 100K pctincrease 0) pctfree 1;

create unique index icustomer on customer(c_w_id, c_d_id, c_id)
       tablespace icust1
       initrans 3
       parallel 10
       storage (initial 8M next 8M pctincrease 0) pctfree 1;

create unique index icustomer2 on customer(c_w_id, c_d_id, c_last,
c_first, c_id)
       tablespace icust2
       initrans 3
       parallel 10
       storage (initial 16M next 16M pctincrease 0) pctfree 1;

create unique index istock on stock(s_i_id, s_w_id)
       tablespace istk
       initrans 3
       parallel 10
       storage (initial 20M next 20M pctincrease 0) pctfree 1;

spool off;
exit;
```

bench/tpc/tpcc/sql/tpcc_ix2.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       tpcc_ix2.sql
rem  DESCRIPTION
rem       Create indexes for TPC-C database.
rem ================================================================
rem

spool tpcc_ix2;
set timing on

drop index iorders;
drop index iorders2;
drop index inew_order;
drop index iorder_line;

create unique index iorders on orders(o_w_id, o_d_id, o_id)
tablespace iord1
initrans 3
parallel 10
pctfree 1
storage (initial 20M next 20M pctincrease 0
freelist groups 13 freelists 24);

create unique index iorders2 on orders(o_w_id, o_d_id, o_c_id, o_id)
tablespace iord2
initrans 3
parallel 10
pctfree 25
storage (initial 30M next 30M pctincrease 0
freelist groups 13 freelists 24);

create unique index inew_order on new_order(no_w_id, no_d_id, no_o_id)
tablespace inord
initrans 4
parallel 10
pctfree 5
```

```
storage (initial 7M next 7M pctincrease 0
freelist groups 13 freelists 24);

create unique index iorder_line on order_line(ol_w_id, ol_d_id, ol_o_id,
ol_number)
        tablespace iordl
        initrans 4
        parallel 10
        pctfree 1
        storage (initial 50M next 50M pctincrease 0
                 freelist groups 13 freelists 24);

exit;
```

bench/tpc/tpcc/sql/tpcc_rol.sql

```
rem
rem ================================================================+
rem         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA      |
rem                    OPEN SYSTEMS PERFORMANCE GROUP                |
rem                         All Rights Reserved                      |
rem ================================================================+
rem  FILENAME
rem       tpcc_rol.sql
rem  DESCRIPTION
rem       Create rollback segments for TPCC database.
rem ================================================================
rem

spool tpcc_rol;
set timing on;
   CREATE ROLLBACK SEGMENT t1
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t2
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t3
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t4
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t5
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t6
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t7
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t8
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t9
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t10
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t11
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t12
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t13
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t14
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t15
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t16
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t17
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t18
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t19
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t20
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t21
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t22
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t23
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t24
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t25
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t26
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t27
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t28
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t29
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t30
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
```

```
CREATE ROLLBACK SEGMENT t31
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t32
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t33
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t34
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t35
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t36
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t37
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t38
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t39
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t40
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t41
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t42
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t43
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t44
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t45
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t46
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t47
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t48
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t49
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t50
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t51
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t52
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t53
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t54
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t55
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t56
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t57
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t58
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t59
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t60
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t61
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t62
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t63
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t64
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t65
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t66
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t67
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t68
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t69
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t70
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t71
   TABLESPACE roll
```

```
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t72
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t73
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t74
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t75
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t76
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t77
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t78
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t79
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t80
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t81
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t82
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t83
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t84
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t85
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t86
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t87
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t88
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t89
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t90
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t91
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t92
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t93
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t94
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t95
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t96
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t97
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t98
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t99
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t100
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t101
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t102
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t103
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t104
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t105
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t106
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t107
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t108
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t109
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t110
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t111
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t112
```

```
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t113
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t114
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t115
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t116
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t117
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t118
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t119
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t120
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t121
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t122
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t123
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t124
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t125
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t126
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t127
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t128
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t129
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t130
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t131
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t132
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t133
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t134
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t135
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t136
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t137
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t138
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t139
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t140
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t141
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t142
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t143
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t144
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t145
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t146
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t147
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t148
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t149
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t150
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t151
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t152
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
```

```
CREATE ROLLBACK SEGMENT t153
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t154
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t155
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t156
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t157
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t158
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t159
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t160
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t161
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t162
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t163
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t164
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t165
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t166
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t167
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t168
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t169
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t170
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t171
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t172
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t173
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t174
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t175
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t176
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t177
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t178
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t179
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t180
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t181
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t182
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t183
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t184
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t185
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t186
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t187
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t188
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t189
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t190
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t191
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t192
   TABLESPACE roll
   STORAGE (initial 400K next 400K minextents 2);
CREATE ROLLBACK SEGMENT t193
   TABLESPACE roll
```

```
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t194
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t195
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t196
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t197
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t198
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t199
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);
   CREATE ROLLBACK SEGMENT t200
      TABLESPACE roll
      STORAGE (initial 400K next 400K minextents 2);

   exit;
```

bench/tpc/tpcc/sql/tpcc_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       tpcc_tab.sql
rem  DESCRIPTION
rem       Create tables for TPC-C database.
rem ====================================================================
rem


rem

rem
rem NEXT, DROP all first
rem
        spool tpcc_tab;
        drop cluster icluster including tables;
       drop table warehouse;
       drop table district;
       drop table history;
       drop table orders;
       drop table new_order;
       drop table order_line;
       drop table item;

set timing on

rem
rem LAST, CREATE all tables
rem


rem
rem WAREHOUSE table
rem

       create table warehouse (
              w_id            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),
              w_tax           number,
              w_ytd           number
       )
       tablespace ware
       initrans 4
       pctfree 95 pctused 4
       storage (initial 1000K next 40K pctincrease 0);


rem
rem DISTRICT table
rem

       create table district (
       d_id            number,
       d_w_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),
       d_tax           number,
       d_ytd           number,
       d_next_o_id     number
       )
       tablespace ware
       initrans 4
       pctfree 95 pctused 4
       storage (initial 10000K next 100K pctincrease 0);


rem
rem HISTORY table
rem

       create table history (
       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)
       )
       tablespace hist
       initrans 3
```

```
        pctfree 1
        storage (initial 20K next 85M pctincrease 0
                 freelist groups 13 freelists 24);


rem
rem ORDER table
rem

        create table orders (
        o_id            number,
        o_d_id          number,
        o_w_id          number,
        o_c_id          number,
        o_entry_d       date,
        o_carrier_id    number,
        o_ol_cnt        number,
        o_all_local     number
        )
        tablespace ord
        initrans 3
        pctfree 5
        storage (initial 20K next 30M pctincrease 0
                 freelist groups 13 freelists 24);


rem
rem NEW_ORDER table
rem

        create table new_order (
        no_o_id         number,
        no_d_id         number,
        no_w_id         number
        )
        tablespace nord
        initrans 4
        pctfree 5
        storage (initial 20K next 80M pctincrease 0
                 freelist groups 13 freelists 24);


rem
rem ORDER_LINE table
rem

        create table order_line (
        ol_o_id         number,
        ol_d_id         number,
        ol_w_id         number,
        ol_number       number,
        ol_i_id         number,
        ol_supply_w_id  number,
        ol_delivery_d   date,
        ol_quantity     number,
        ol_amount       number,
        ol_dist_info    char(24)
        )
        tablespace ordl
        initrans 4
        pctfree 5
        storage (initial 20K next 530M pctincrease 0
                 freelist groups 13 freelists 24);


rem
rem ITEM table
rem length = 4 + 24 + 5 + 50 = 83
rem

        create cluster icluster (
        i_id            number(6,0)
        )
        hashkeys        100000
        hash is         i_id
        size            120
        initrans        3
        pctfree         0
        tablespace      items
        storage (initial 14M next 720K pctincrease 0);

        create table item (
        i_id            number(6,0),
        i_im_id         number,
        i_name          varchar2(24),
        i_price         number,
        i_data          varchar2(50)
        )
        cluster icluster(i_id);


rem
rem done
rem

        exit;
```

bench/tpc/tpcc/sql/tpcc_tab2.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       tpcc_tab2.sql
rem  DESCRIPTION
rem       Create customer table for TPC-C database.
rem ====================================================================
rem


rem
rem DROP all first
rem
       spool tpcc_tab2;
        drop cluster ccluster including tables;
       drop table customer;

set timing on
```

```
rem
rem CUSTOMER table
rem

        create cluster ccluster (
           c_id         number(5,0),
           c_d_id       number(2,0),
           c_w_id       number(4,0)
        )
        hashkeys        18000000
        hash is         (c_w_id * 30000 + c_d_id * 3000 + c_id)
        size            850
        initrans        3
        pctfree         0
        tablespace      cust
        storage (initial 1475M next 1475M pctincrease 0 minextents 12);

       create table customer (
        c_id            number(5,0),
        c_d_id          number(2,0),
        c_w_id          number(4,0),
        c_first         varchar2(16),
        c_middle        char(2),
        c_last          varchar2(16),
        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_credit        char(2),
        c_credit_lim    number,
        c_discount      number,
        c_balance       number,
        c_ytd_payment   number,
        c_payment_cnt   number,
        c_delivery_cnt  number,
        c_data          varchar2(500)
        )
        cluster ccluster (c_id, c_d_id, c_w_id);

rem
rem done
rem

        exit;
```

bench/tpc/tpcc/sql/tpcc_tab3.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       tpcc_tab3.sql
rem  DESCRIPTION
rem       Create stock table for TPC-C database.
rem ====================================================================
rem

rem
rem DROP all first
rem
spool tpcc_tab3;
set timing on;

        drop cluster scluster including tables;
       drop table stock;


rem
rem STOCK table
rem

        create cluster scluster (
           s_i_id       number(6,0),
           s_w_id       number(4,0)
        )
        hashkeys        60000000
        hash is         (s_i_id * 600 + s_w_id)
        size            350
        initrans        3
        pctfree         0
        tablespace      stocks
        storage (initial 1356M next 1356M pctincrease 0 minextents 18);

       create table stock (
        s_i_id          number(6,0),
        s_w_id          number(4,0),
        s_quantity      number,
        s_dist_01       char(24),
        s_dist_02       char(24),
        s_dist_03       char(24),
        s_dist_04       char(24),
        s_dist_05       char(24),
        s_dist_06       char(24),
        s_dist_07       char(24),
        s_dist_08       char(24),
        s_dist_09       char(24),
        s_dist_10       char(24),
        s_ytd           number,
        s_order_cnt     number,
        s_remote_cnt    number,
        s_data          varchar2(50)
        )
        cluster scluster (s_i_id, s_w_id);

rem
rem done
rem

        exit;
```

bench/tpc/tpcc/sql/unlock.sql

```
spool unlock;
alter table warehouse disable table lock;
```

```
alter table district disable table lock;
alter table history disable table lock;
alter table stock disable table lock;
alter table customer disable table lock;
alter table orders disable table lock;
alter table new_order disable table lock;
alter table order_line disable table lock;
alter table item disable table lock;
commit work;
spool off;
exit;
```

bench/tpc/tpcc/source/tpccload.c

```
#ifdef RCSID
static char *RCSid =
   "$Header: tpccload.c 7010000.7 95/05/05 16:55:22 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 warehouses> [options]
  |               options: -A  load all tables
  |                        -w  load warehouse table
  |                        -d  load district table
  |                        -c  load customer table
  |                        -i  load item table
  |                        -s  load stock table (cluster around s_w_id)
  |                        -S  load stock table (cluster around s_i_id)
  |                        -h  load history table
  |                        -n  load new-order table
  |                        -o <oline file> load order and order-line table
  |                        -b <ware#>  beginning warehouse number
  |                        -e <ware#>  ending warehouse 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"

#define DISTARR 10            /* district insert array size */
#define CUSTARR 100           /* customer insert array size */
#define STOCARR 100           /* stock insert array size    */
#define ITEMARR 100           /* item insert array size     */
#define HISTARR 100             /* history 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. customer id    */
#define STOCFAC 100000       /* max. stock id       */
#define ITEMFAC 100000       /* max. item id                */
#define HISTFAC 30000          /* history / warehouse  */
#define ORDEFAC 3000           /* order / district     */
#define NEWOFAC 900            /* new order / district */

#define C       0              /* constant in non-uniform dist. eqt. */
#define CNUM1   1              /* first constant in non-uniform dist.
eqt. */
#define CNUM2   2              /* second constant in non-uniform dist.
eqt. */
#define CNUM3   3              /* third constant in non-uniform dist.
eqt. */

#define SEED    2              /* seed for random functions */

#define SQLTXTW "INSERT INTO warehouse VALUES (:w_id, :w_name,
:w_street_1, \
   :w_street_2, :w_city, :w_state, :w_zip, :w_tax, 300000.0)"

#define SQLTXTD "INSERT INTO district VALUES (:d_id, :d_w_id, :d_name, \
   :d_street_1, :d_street_2, :d_city, :d_state, :d_zip, :d_tax, 30000.0,
3001)"

#define SQLTXTC "INSERT INTO customer 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, 50000.0, :c_discount, -10.0,
10.0, 1, \
   0, :c_data)"

#define SQLTXTH "INSERT INTO history VALUES (:h_c_id, :h_c_d_id,
:h_c_w_id, \
   :h_d_id, :h_w_id, SYSDATE, 10.0, :h_data)"

#define SQLTXTS "INSERT INTO stock 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 VALUES (:i_id, :i_im_id, :i_name,
:i_price, \
   :i_data)"

#define SQLTXTO1 "INSERT INTO orders VALUES (:o_id, :o_d_id, :o_w_id,
:o_c_id, \
   SYSDATE, :o_carrier_id, :o_ol_cnt, 1)"

#define SQLTXTO2 "INSERT INTO orders VALUES (:o_id, :o_d_id, :o_w_id,
:o_c_id, \
   SYSDATE, NULL, :o_ol_cnt, 1)"

#define SQLTXTOL1 "INSERT INTO order_line VALUES (:ol_o_id, :ol_d_id, \
   :ol_w_id, :ol_number, :ol_i_id, :ol_supply_w_id, SYSDATE, 5, 0.0, \
   :ol_dist_info)"
```

```
#define SQLTXTOL2 "INSERT INTO order_line VALUES (:ol_o_id, :ol_d_id, \
   :ol_w_id, :ol_number, :ol_i_id, :ol_supply_w_id, NULL, 5, :ol_amount, \
   :ol_dist_info)"

#define SQLTXTNO "INSERT INTO new_order 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];



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 warehouse table\n");
   fprintf (stderr, "\t-d :\tload district table\n");
   fprintf (stderr, "\t-c :\tload customer table\n");
   fprintf (stderr, "\t-i :\tload item table\n");
   fprintf (stderr, "\t-s :\tload stock table (cluster around s_w_id)\n");
   fprintf (stderr, "\t-S :\tload stock table (cluster around s_i_id)\n");
   fprintf (stderr, "\t-h :\tload history 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 warehouse number\n");
   fprintf (stderr, "\t-e <ware#> :\tending warehouse 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);

}



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);
   }

}



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");

}
```

```
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];
   float 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];
   float 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];

   double begin_time, end_time;
   double begin_cpu, end_cpu;
   double gettime(), getcpu();

   extern int getopt();
   extern char *optarg;
   extern int optind, opterr;

   char *argstr="M:AwdcisShno:b:e:j:k:g";
   int opt;
   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];

/*---------------------------------------------------------------+
 | Parse command line -- look for scale factor.                  |
 +---------------------------------------------------------------*/

   if (argc == 1) {
      myusage ();
   }

   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 ();
      }
   }

/*---------------------------------------------------------------*|
|      Rudimentary error checking                                 |
|*---------------------------------------------------------------*/

   if (scale  < 1) {
      fprintf (stderr, "Invalid scale factor: '%d'\n", scale);
      myusage ();
   }

   if (!(do_A || do_w || do_d || do_c || do_i || do_s || do_S || do_h ||
do_o ||
         do_n)) {
      fprintf (stderr, "What should I load???\n");
      myusage ();
   }

   if (gen && (do_A || (do_w + do_d + do_c + do_i + do_s + do_S + do_h +
do_o +
                        do_n > 1))) {
      fprintf (stderr, "Can only generate table one at a time\n");
      myusage ();
   }

   if (do_S && (do_A || do_s)) {
      fprintf (stderr, "Cluster stock table around s_w_id or s_i_id?\n");
      myusage ();
   }

   if (eware <= 0)
      eware = scale;
   if (eitem <= 0)
      eitem = STOCFAC;

   if (do_S) {
      if ((bitem  < 1) || (bitem > STOCFAC)) {
         fprintf (stderr, "Invalid beginning item number: '%d'\n", bitem);
         myusage ();
      }

      if ((eitem  < bitem) || (eitem > STOCFAC)) {
         fprintf (stderr, "Invalid ending item number: '%d'\n", eitem);
         myusage ();
      }
   }

   if ((bware  < 1) || (bware > scale)) {
      fprintf (stderr, "Invalid beginning warehouse number: '%d'\n",
bware);
      myusage ();
   }

   if ((eware  < bware) || (eware > scale)) {
      fprintf (stderr, "Invalid ending warehouse number: '%d'\n", eware);
      myusage ();
   }
```

```
   if (gen && do_o) {
      if ((olfp = fopen (olfname, "w")) == NULL) {
         fprintf (stderr, "Can't open '%s' for writing order lines\n",
olfname);
         myusage ();
      }
   }

/*---------------------------------------------------------------+
 | Prepare to insert into database.                              |
 +---------------------------------------------------------------*/

   sysdate (sdate);
   if (!gen) {

      /* log on to Oracle */

      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 */

   /* warehouse */
```

```
      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);
      }

      /* district */

      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);
      }

      /* customer */

      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 (float), SQLT_FLT, -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);
      }

      /* stock */

      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);
      }

      /* history */

      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);
      }

      /* order_line (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);
      }

      /* order_line (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 (float), SQLT_FLT, -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);
      }

      /* orders (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);
      }

      /* orders (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 (getpid ());
   srand48 (getpid ());
   initperm ();
```

```
/*---------------------------------------------------------------+
 | Load the WAREHOUSE table.                                     |
 +---------------------------------------------------------------*/

   if (do_A || do_w) {
      nrows = eware - bware + 1;

      fprintf (stderr, "Loading/generating warehouse: w%d - w%d (%d
rows)\n",
               bware, eware, nrows);

      begin_time = gettime ();
      begin_cpu = getcpu ();

      for (loop = bware; loop <= eware; loop++) {

         w_tax = (rand () % 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 %s %s %s %s %s %s %6.4f 300000.0\n", loop,
                    w_name, w_street_1, w_street_2, w_city, str2, num9,
w_tax);
            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 warehouse %d\n", loop);
               quit ();
               exit (1);
            }
            else if (ocom (&tpclda)) {
               errrpt (&tpclda, &tpclda);
               orol (&tpclda);
               fprintf (stderr, "Aborted at warehouse %d\n", loop);
               quit ();
               exit (1);
            }
         }
      }

      end_time = gettime ();
      end_cpu = getcpu ();
      fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec.
(%10.2f cpu)\n\n",
               nrows, end_time - begin_time, end_cpu - begin_cpu);
   }
```

```
/*-------------------------------------------------------------+
 | Load the DISTRICT table.                                    |
 +-------------------------------------------------------------*/

   if (do_A || do_d) {
      nrows = (eware - bware + 1) * DISTFAC;

      fprintf (stderr, "Loading/generating district: w%d - w%d (%d
rows)\n",
               bware, eware, nrows);

      begin_time = gettime ();
      begin_cpu = getcpu ();

      dwid = bware - 1;

      for (row = 0; row < nrows; ) {
         dwid++;

         for (i = 0; i < DISTARR; i++, row++) {
            d_tax[i] = (rand () % 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 %6.4f 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]);
            }
            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 warehouse %d, district 1\n",
dwid);
               quit ();
               exit (1);
            }
            else if (ocom (&tpclda)) {
               errrpt (&tpclda, &tpclda);
               orol (&tpclda);
               fprintf (stderr, "Aborted at warehouse %d, district 1\n",
dwid);
               quit ();
               exit (1);
            }
         }
      }

      end_time = gettime ();
      end_cpu = getcpu ();
      fprintf (stderr, "Done.  %d rows loaded/generated in %10.2f sec.
(%10.2f cpu)\n\n",
               nrows, end_time - begin_time, end_cpu - begin_cpu);
   }

/*-------------------------------------------------------------+
 | Load the CUSTOMER table.                                    |
 +-------------------------------------------------------------*/

   if (do_A || do_c) {
      nrows = (eware - bware + 1) * CUSTFAC * DISTFAC;

      fprintf (stderr, "Loading/generating customer: w%d - w%d (%d rows)\n
",
               bware, eware, nrows);

      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 district cycle */
               if (cdid > DISTFAC) {
                  cdid = 1;
                  cwid++;               /* shift warehouse 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 (rand () % 10)
               c_credit[i][0] = 'G';
            else
               c_credit[i][0] = 'B';
            c_discount[i] = (rand () % 5001) * 0.0001;
            randstr (c_first[i], 8, 16);
            randstr (c_street_1[i], 10, 20);
            randstr (c_street_2[i], 10, 20);
            randstr (c_city[i], 10, 20);
            randstr (str2, 2, 2);
            randnum (num9, 9);
```

```
            num9[4] = num9[5] = num9[6] = num9[7] = num9[8] = '1';
            randnum (num16, 16);
            randstr (c_data[i], 300, 500);

            if (gen) {
               printf ("%d %d %d %s OE %s %s %s %s %s %s %s %s %cC 50000.0
%6.4f -10.0 10.0 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 %10.2f sec.
(%10.2f 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] = (rand () % 10000) + 1;
            i_price[i] = ((rand () % 9901) + 100) * 0.01;
            randstr (i_name[i], 14, 24);
            randdatastr (i_data[i], 26, 50);

            if (gen) {
               printf ("%d %d %s %6.2f %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 %10.2f sec.
(%10.2f cpu)\n\n",
               nrows, end_time - begin_time, end_cpu - begin_cpu);
   }


/*-------------------------------------------------------------+
```

```
 | Load the STOCK table.                                          |
 +---------------------------------------------------------------*/

   if (do_A || do_s) {

      nrows = (eware - bware + 1) * STOCFAC;

      fprintf (stderr, "Loading/generating stock: w%d - w%d (%d rows)\n
",
               bware, eware, nrows);

      begin_time = gettime ();
      begin_cpu = getcpu ();

      sid = 0;
      swid = bware;
      loopcount = 0;

      for (row = 0; row < nrows; ) {
         for (i = 0; i < STOCARR; i++, row++) {
            if (++sid > STOCFAC) {        /* cheap mod */
               sid = 1;
               swid++;
            }
            s_quantity[i] = (rand () % 91) + 10;
            randstr (str24[0], 24, 24);
            randstr (str24[1], 24, 24);
            randstr (str24[2], 24, 24);
            randstr (str24[3], 24, 24);
            randstr (str24[4], 24, 24);
            randstr (str24[5], 24, 24);
            randstr (str24[6], 24, 24);
            randstr (str24[7], 24, 24);
            randstr (str24[8], 24, 24);
            randstr (str24[9], 24, 24);
            randdatastr (s_data[i], 26, 50);

            if (gen) {
               printf ("%d %d %d %s %s %s %s %s %s %s %s %s %s 0 0 0
%s\n",
                       sid, swid, s_quantity[i], str24[0], str24[1],
str24[2],
                       str24[3], str24[4], str24[5], str24[6], str24[7],
                       str24[8], str24[9], s_data[i]);
            }
            else {
               s_i_id[i] = sid;
               s_w_id[i] = swid;
               strncpy (s_dist_01[i], str24[0], 24);
               strncpy (s_dist_02[i], str24[1], 24);
               strncpy (s_dist_03[i], str24[2], 24);
               strncpy (s_dist_04[i], str24[3], 24);
               strncpy (s_dist_05[i], str24[4], 24);
               strncpy (s_dist_06[i], str24[5], 24);
               strncpy (s_dist_07[i], str24[6], 24);
               strncpy (s_dist_08[i], str24[7], 24);
               strncpy (s_dist_09[i], str24[8], 24);
               strncpy (s_dist_10[i], str24[9], 24);
            }
         }
         if (gen) {
            fflush (stdout);
         }
         else {
            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 %10.2f sec.
(%10.2f cpu)\n\n",
               nrows, end_time - begin_time, end_cpu - begin_cpu);
   }

/*---------------------------------------------------------------+
 | Load the STOCK table (cluster around s_i_id).                 |
 +---------------------------------------------------------------*/

   if (do_S) {

      nrows = (eitem - bitem + 1) * (eware - bware + 1);

      fprintf (stderr, "Loading/generating stock: i%d - i%d, w%d - w%d (%d
rows)\n   ",
               bitem, eitem, bware, eware, nrows);

      begin_time = gettime ();
      begin_cpu = getcpu ();

      sid = bitem;
      swid = bware - 1;
      loopcount = 0;

      for (row = 0; row < nrows; ) {
         for (i = 0; i < STOCARR; i++, row++) {
            if (++swid > eware) {          /* cheap mod */
               swid = bware;
               sid++;
```

```
            }
            s_quantity[i] = (rand () % 91) + 10;
            randstr (str24[0], 24, 24);
            randstr (str24[1], 24, 24);
            randstr (str24[2], 24, 24);
            randstr (str24[3], 24, 24);
            randstr (str24[4], 24, 24);
            randstr (str24[5], 24, 24);
            randstr (str24[6], 24, 24);
            randstr (str24[7], 24, 24);
            randstr (str24[8], 24, 24);
            randstr (str24[9], 24, 24);
            randdatastr (s_data[i], 26, 50);

            if (gen) {
               printf ("%d %d %d %s %s %s %s %s %s %s %s %s %s 0 0 0
%s\n",
                       sid, swid, s_quantity[i], str24[0], str24[1],
str24[2],
                       str24[3], str24[4], str24[5], str24[6], str24[7],
                       str24[8], str24[9], s_data[i]);
            }
            else {
               s_i_id[i] = sid;
               s_w_id[i] = swid;
               strncpy (s_dist_01[i], str24[0], 24);
               strncpy (s_dist_02[i], str24[1], 24);
               strncpy (s_dist_03[i], str24[2], 24);
               strncpy (s_dist_04[i], str24[3], 24);
               strncpy (s_dist_05[i], str24[4], 24);
               strncpy (s_dist_06[i], str24[5], 24);
               strncpy (s_dist_07[i], str24[6], 24);
               strncpy (s_dist_08[i], str24[7], 24);
               strncpy (s_dist_09[i], str24[8], 24);
               strncpy (s_dist_10[i], str24[9], 24);
            }
         }

         if (gen) {
            fflush (stdout);
         }
         else {
            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 %10.2f sec.
(%10.2f cpu)\n\n",
               nrows, end_time - begin_time, end_cpu - begin_cpu);
   }

/*---------------------------------------------------------------+
 | Load the HISTORY table.                                       |
 +---------------------------------------------------------------*/

   if (do_A || do_h) {
      nrows = (eware - bware + 1) * HISTFAC;

      fprintf (stderr, "Loading/generating history: w%d - w%d (%d rows)\n
",
               bware, eware, nrows);

      begin_time = gettime ();
      begin_cpu = getcpu ();

      cid = 0;
      cdid = 1;
      cwid = bware;
      loopcount = 0;

      for (row = 0; row < nrows; ) {
         for (i = 0; i < HISTARR; i++, row++) {
            cid++;
            if (cid > CUSTFAC) {        /* cycle cust id */
               cid = 1;                 /* cheap mod */
               cdid++;                  /* shift district cycle */
               if (cdid > DISTFAC) {
                  cdid = 1;
                  cwid++;               /* shift warehouse cycle */
               }
            }
            h_c_id[i] = cid;
            h_d_id[i] = cdid;
            h_w_id[i] = cwid;
            randstr (h_data[i], 12, 24);
            if (gen) {
               printf ("%d %d %d %d %d %s 10.0 %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 %10.2f sec.
(%10.2f cpu)\n\n",
               nrows, end_time - begin_time, end_cpu - begin_cpu);
   }

/*---------------------------------------------------------------+
 | Load the ORDERS and ORDER-LINE table.                         |
 +---------------------------------------------------------------*/

   if (do_A || do_o) {
      nrows = (eware - bware + 1) * ORDEFAC * DISTFAC;

      fprintf (stderr, "Loading/generating orders and order-line: w%d -
w%d (%d ord, ~%d ordl)\n   ",
               bware, eware, nrows, nrows * 10);

      begin_time = gettime ();
      begin_cpu = getcpu ();

      cid = 0;
      cdid = 1;
      cwid = bware;
      loopcount = 0;

      for (row = 0; row < nrows; ) {
         for (i = 0; i < ORDEARR; i++, row++) {
            cid++;
            if (cid > ORDEFAC) {        /* cycle cust id */
               cid = 1;                 /* cheap mod */
               cdid++;                  /* shift district cycle */
               if (cdid > DISTFAC) {
                  cdid = 1;
                  cwid++;               /* shift warehouse cycle */
               }
            }
            o_carrier_id[i] = rand () % 10 + 1;
            o_ol_cnt[i] = olcnt = rand () % 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 {
                  printf ("%d %d %d %d %s \"\" %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] = 0.0;
               else
                  ol_amount[j] = (lrand48 () % 999999 + 1) * 0.01;
               randstr (str24[j], 24, 24);

               if (gen) {
                  if (cid < 2101) {
                     fprintf (olfp, "%d %d %d %d %d %d %s 5 %7.2f %s\n",
cid,
                              cdid, cwid, j + 1, ol_i_id[j], cwid, sdate,
                              ol_amount[j], str24[j]);
                  }
                  else {
                     fprintf (olfp, "%d %d %d %d %d %d \"\" 5 %7.2f %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 orders committed\n   ", row);
   }

   end_time = gettime ();
   end_cpu = getcpu ();
   fprintf (stderr, "Done.  %d orders loaded/generated in %10.2f sec.
(%10.2f cpu)\n\n",
            nrows, end_time - begin_time, end_cpu - begin_cpu);
}

/*---------------------------------------------------------------+
 | Load the NEW-ORDER table.                                     |
 +---------------------------------------------------------------*/

if (do_A || do_n) {
   nrows = (eware - bware + 1) * NEWOFAC * DISTFAC;

   fprintf (stderr, "Loading/generating new-order: w%d - w%d (%d
rows)\n   ",
            bware, eware, nrows);

   begin_time = gettime ();
   begin_cpu = getcpu ();

   cid = 0;
   cdid = 1;
   cwid = bware;
   loopcount = 0;

   for (row = 0; row < nrows; ) {
      for (i = 0; i < NEWOARR; i++, row++) {
         cid++;
         if (cid > NEWOFAC) {
            cid = 1;
            cdid++;
            if (cdid > DISTFAC) {
```

```
                cdid = 1;
                cwid++;
             }
          }

          if (gen) {
             printf ("%d %d %d\n", cid + 2100, cdid, cwid);
          }
          else {
             no_o_id[i] = cid + 2100;
             no_d_id[i] = cdid;
             no_w_id[i] = cwid;
          }
       }

       if (gen) {
          fflush (stdout);
       }
       else {
          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 %10.2f sec.
(%10.2f cpu)\n\n",
             nrows, end_time - begin_time, end_cpu - begin_cpu);
 }

/*-------------------------------------------------------------+
 | clean up and exit.                                          |
 +-------------------------------------------------------------*/

 if (olfp)
    fclose (olfp);
 if (!gen)
    quit ();
 exit (0);
}


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 = rand () % i;
    temp = randperm3000[i - 1];
    randperm3000[i - 1] = randperm3000[pos];
    randperm3000[pos] = temp;
 }

}



randstr (str, x, y)

char *str;
int x;
int y;

{

 int i;
 int len;

 len = (rand () % (y - x + 1)) + x;
 for (i = 0; i < len; i++)
    str[i] = (char) (rand () % 26 + 'a');
 str[len] = '\0';

}



randdatastr (str, x, y)

char *str;
int x;
int y;

{

 int i;
 int len;
 int pos;

 len = (rand () % (y - x + 1)) + x;
```

```
   for (i = 0; i < len; i++)
      str[i] = (char) (rand () % 26 + 'a');
   str[len] = '\0';
   if ((rand () % 10) == 0) {
      pos = (rand () % (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';
   }

}



randnum (str, len)

char *str;
int len;

{

   int i;

   for (i = 0; i < len; i++)
      str[i] = (char) (rand () % 10 + '0');
   str[len] = '\0';

}



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]);

}



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);

}



sysdate (sdate)

char *sdate;

{

   time_t tp;
   struct tm *tmptr;

   time (&tp);
   tmptr = localtime (&tp);
   strftime (sdate, 29, "%d-%b-%y", tmptr);

}
```

bench/tpc/tpcc/source/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

#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

#include <oratypes.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 int TPCexit ();
extern int TPCdumpinit ();
extern int TPCdumpnew ();
extern int TPCdumppay ();
extern int TPCdumpord ();
extern int TPCdumpdel ();
extern int TPCdumpsto ();
extern int TPCdumpexit ();


/* Error codes */

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

#endif
```

bench/tpc/tpcc/loader/cust.ctl

```
--
-- $Header: cust.ctl 7030100.1 95/08/07 15:46:36 plai Generic<base> $
Copyr (c) 1994 Oracle
--
-- +---------------------------------------------------------------+
--          Copyright (c) 1994  Oracle Corp, Redwood Shores, CA      |
--                       OPEN SYSTEMS PERFORMANCE GROUP              |
--                            All Rights Reserved                    |
-- +---------------------------------------------------------------+
--  FILENAME
--       cust.ctl
--  DESCRIPTION
--       This is a SQL*Loader control file.  It is used for
--       loading customers to the tpcc database.
--  USAGE
--       sqlldr[st] <user_name>/<password> <SQL*Loader control file>
-- +---------------------------------------------------------------+*/

OPTIONS (DIRECT = TRUE, PARALLEL = TRUE)

UNRECOVERABLE

LOAD DATA
APPEND

INTO TABLE customer
APPEND
FIELDS TERMINATED BY WHITESPACE OPTIONALLY ENCLOSED BY '"'
(
   c_id                integer external,
   c_d_id              integer external,
   c_w_id              integer external,
   c_first             char(16),
   c_middle            char(2),
   c_last              char(16),
   c_street_1          char(20),
   c_street_2          char(20),
   c_city              char(20),
   c_state             char(2),
   c_zip               char(9),
   c_phone             char(16),
   c_since             date,
   c_credit            char(2),
   c_credit_lim        float external,
   c_discount          float external,
   c_balance           float external,
   c_ytd_payment       float external,
   c_payment_cnt       integer external,
   c_delivery_cnt      integer external,
   c_data              char(500)
)
```

bench/tpc/tpcc/loader/del.ctl

```
--
-- $Header: del.ctl 7010000.1 94/10/03 10:18:07 pswong Generic<base> $
Copyr (c) 1993 Oracle
--
-- +---------------------------------------------------------------+
--          Copyright (c) 1994  Oracle Corp, Redwood Shores, CA      |
--                       OPEN SYSTEMS PERFORMANCE GROUP              |
--                            All Rights Reserved                    |
-- +---------------------------------------------------------------+
--  FILENAME
--       del.ctl
--  DESCRIPTION
--       This file is meant to be prepended to the tpcc_*.del files.
-- ================================================================


LOAD DATA

INFILE *                            -- use inline data

INTO TABLE bench_del_res            -- load the BENCH_DEL_RES table
APPEND                              -- OK if there are data already
FIELDS TERMINATED BY WHITESPACE     -- whitespace is the field separator
    (
    config              char(10),
    run                 integer external,
    audit_str           char(10),
    proc_no             integer external,
    rep1                integer external,
    rep2                integer external,
    rep3                integer external,
    rep4                integer external,
    rep5                integer external,
    rep6                integer external,
    rep7                integer external,
    rep8                integer external,
    rep9                integer external,
    rep10               integer external,
    rep11               integer external,
```

```
rep12           integer external,
rep13           integer external,
rep14           integer external,
rep15           integer external,
rep16           integer external,
rep17           integer external,
rep18           integer external,
rep19           integer external,
rep20           integer external,
rep21           integer external,
rep22           integer external,
rep23           integer external,
rep24           integer external,
rep25           integer external,
rep26           integer external,
rep27           integer external,
rep28           integer external,
rep29           integer external,
rep30           integer external,
rep31           integer external,
rep32           integer external,
rep33           integer external,
rep34           integer external,
rep35           integer external,
rep36           integer external,
rep37           integer external,
rep38           integer external,
rep39           integer external,
rep40           integer external,
rep41           integer external,
rep42           integer external,
rep43           integer external,
rep44           integer external,
rep45           integer external,
rep46           integer external,
rep47           integer external,
rep48           integer external,
rep49           integer external,
rep50           integer external,
rep51           integer external,
rep52           integer external,
rep53           integer external,
rep54           integer external,
rep55           integer external,
rep56           integer external,
rep57           integer external,
rep58           integer external,
rep59           integer external,
rep60           integer external,
rep61           integer external,
rep62           integer external,
rep63           integer external,
rep64           integer external,
rep65           integer external,
rep66           integer external,
rep67           integer external,
rep68           integer external,
rep69           integer external,
rep70           integer external,
rep71           integer external,
rep72           integer external,
rep73           integer external,
rep74           integer external,
rep75           integer external,
rep76           integer external,
rep77           integer external,
rep78           integer external,
rep79           integer external,
rep80           integer external,
rep81           integer external,
rep82           integer external,
rep83           integer external,
rep84           integer external,
rep85           integer external,
rep86           integer external,
rep87           integer external,
rep88           integer external,
rep89           integer external,
rep90           integer external,
rep91           integer external,
rep92           integer external,
rep93           integer external,
rep94           integer external,
rep95           integer external,
rep96           integer external,
rep97           integer external,
rep98           integer external,
rep99           integer external,
rep100          integer external,
thk1            integer external,
thk2            integer external,
thk3            integer external,
thk4            integer external,
thk5            integer external,
thk6            integer external,
thk7            integer external,
thk8            integer external,
thk9            integer external,
thk10           integer external,
thk11           integer external,
thk12           integer external,
thk13           integer external,
thk14           integer external,
thk15           integer external,
thk16           integer external,
thk17           integer external,
thk18           integer external,
thk19           integer external,
thk20           integer external,
thk21           integer external,
thk22           integer external,
thk23           integer external,
thk24           integer external,
thk25           integer external,
key1            integer external,
key2            integer external,
key3            integer external,
key4            integer external,
key5            integer external,
key6            integer external,
key7            integer external,
key8            integer external,
```

```
    key9                integer external,
    key10               integer external
    )
BEGINDATA                 -- start inline data
```

bench/tpc/tpcc/loader/hist.ctl

```
--
-- $Header: hist.ctl 7030100.1 95/08/07 15:47:23 plai Generic<base> $
Copyr (c) 1994 Oracle
--
-- +---------------------------------------------------------------+
--         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA       |
--                      OPEN SYSTEMS PERFORMANCE GROUP               |
--                           All Rights Reserved                     |
-- +---------------------------------------------------------------+
--  FILENAME
--      hist.ctl
--  DESCRIPTION
--      This is a SQL*Loader control file.  It is used for
--      loading history rows to the tpcc database.
--  USAGE
--      sqlldr[st] <user_name>/<password> <SQL*Loader control file>
-- +---------------------------------------------------------------+*/

OPTIONS (DIRECT = TRUE, PARALLEL = TRUE)

UNRECOVERABLE

LOAD DATA
APPEND

INTO TABLE history
APPEND
FIELDS TERMINATED BY WHITESPACE OPTIONALLY ENCLOSED BY '"'
(
   h_c_id              integer external,
   h_c_d_id            integer external,
   h_c_w_id            integer external,
   h_d_id              integer external,
   h_w_id              integer external,
   h_date              date,
   h_amount            float external,
   h_data              char(24)
)
```

bench/tpc/tpcc/loader/new.ctl

```
--
-- $Header: new.ctl 7010000.1 94/10/03 10:18:09 pswong Generic<base> $
Copyr (c) 1993 Oracle
--
-- +---------------------------------------------------------------+
--         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA       |
--                      OPEN SYSTEMS PERFORMANCE GROUP               |
--                           All Rights Reserved                     |
-- +---------------------------------------------------------------+
--  FILENAME
--       new.ctl
--  DESCRIPTION
--       This file is meant to be prepended to the tpcc_*.new files.
-- ===================================================================


LOAD DATA

INFILE *                            -- use inline data

INTO TABLE bench_new_res            -- load the BENCH_NEW_RES table
APPEND                              -- OK if there are data already
FIELDS TERMINATED BY WHITESPACE     -- whitespace is the field separator
    (
    config              char(10),
    run                 integer external,
    audit_str           char(10),
    proc_no             integer external,
    rep1                integer external,
    rep2                integer external,
    rep3                integer external,
    rep4                integer external,
    rep5                integer external,
    rep6                integer external,
    rep7                integer external,
    rep8                integer external,
    rep9                integer external,
    rep10               integer external,
    rep11               integer external,
    rep12               integer external,
    rep13               integer external,
    rep14               integer external,
    rep15               integer external,
    rep16               integer external,
    rep17               integer external,
    rep18               integer external,
    rep19               integer external,
    rep20               integer external,
    rep21               integer external,
    rep22               integer external,
    rep23               integer external,
    rep24               integer external,
    rep25               integer external,
    rep26               integer external,
    rep27               integer external,
    rep28               integer external,
    rep29               integer external,
    rep30               integer external,
    rep31               integer external,
    rep32               integer external,
    rep33               integer external,
    rep34               integer external,
    rep35               integer external,
    rep36               integer external,
    rep37               integer external,
    rep38               integer external,
    rep39               integer external,
    rep40               integer external,
    rep41               integer external,
    rep42               integer external,
    rep43               integer external,
    rep44               integer external,
```

```
rep45                integer external,
rep46                integer external,
rep47                integer external,
rep48                integer external,
rep49                integer external,
rep50                integer external,
rep51                integer external,
rep52                integer external,
rep53                integer external,
rep54                integer external,
rep55                integer external,
rep56                integer external,
rep57                integer external,
rep58                integer external,
rep59                integer external,
rep60                integer external,
rep61                integer external,
rep62                integer external,
rep63                integer external,
rep64                integer external,
rep65                integer external,
rep66                integer external,
rep67                integer external,
rep68                integer external,
rep69                integer external,
rep70                integer external,
rep71                integer external,
rep72                integer external,
rep73                integer external,
rep74                integer external,
rep75                integer external,
rep76                integer external,
rep77                integer external,
rep78                integer external,
rep79                integer external,
rep80                integer external,
rep81                integer external,
rep82                integer external,
rep83                integer external,
rep84                integer external,
rep85                integer external,
rep86                integer external,
rep87                integer external,
rep88                integer external,
rep89                integer external,
rep90                integer external,
rep91                integer external,
rep92                integer external,
rep93                integer external,
rep94                integer external,
rep95                integer external,
rep96                integer external,
rep97                integer external,
rep98                integer external,
rep99                integer external,
rep100               integer external,
thk1                 integer external,
thk2                 integer external,
thk3                 integer external,
thk4                 integer external,
thk5                 integer external,
thk6                 integer external,
thk7                 integer external,
thk8                 integer external,
thk9                 integer external,
thk10                integer external,
thk11                integer external,
thk12                integer external,
thk13                integer external,
thk14                integer external,
thk15                integer external,
thk16                integer external,
thk17                integer external,
thk18                integer external,
thk19                integer external,
thk20                integer external,
thk21                integer external,
thk22                integer external,
thk23                integer external,
thk24                integer external,
thk25                integer external,
key1                 integer external,
key2                 integer external,
key3                 integer external,
key4                 integer external,
key5                 integer external,
key6                 integer external,
key7                 integer external,
key8                 integer external,
key9                 integer external,
key10                integer external
)
BEGINDATA                -- start inline data
```

bench/tpc/tpcc/loader/neword.ctl

```
--
-- $Header: neword.ctl 7030100.1 95/08/07 15:47:44 plai Generic<base> $
Copyr (c) 1994 Oracle
--
-- ======================================================================+
--         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA           |
--                      OPEN SYSTEMS PERFORMANCE GROUP                   |
--                           All Rights Reserved                         |
-- ======================================================================+
--  FILENAME
--      neword.ctl
--  DESCRIPTION
--      This is a SQL*Loader control file.  It is used for
--      loading new orders to the tpcc database.
--  USAGE
--      sqlldr[st] <user_name>/<password> <SQL*Loader control file>
-- ======================================================================*/

OPTIONS (DIRECT = TRUE, PARALLEL = TRUE)

UNRECOVERABLE

LOAD DATA
APPEND
```

```
INTO TABLE new_order
APPEND
FIELDS TERMINATED BY WHITESPACE OPTIONALLY ENCLOSED BY '"'
(
   no_o_id             integer external,
   no_d_id             integer external,
   no_w_id             integer external
)
```

bench/tpc/tpcc/loader/ord.ctl

```
--
-- $Header: ord.ctl 7010000.1 94/10/03 10:18:11 pswong Generic<base> $
Copyr (c) 1993 Oracle
--
-- ================================================================+
--         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA      |
--                      OPEN SYSTEMS PERFORMANCE GROUP              |
--                           All Rights Reserved                    |
-- ================================================================+
--  FILENAME
--       ord.ctl
--  DESCRIPTION
--       This file is meant to be prepended to the tpcc_*.ord files.
-- ================================================================


LOAD DATA

INFILE *                          -- use inline data

INTO TABLE bench_ord_res          -- load the BENCH_ORD_RES table
APPEND                            -- OK if there are data already
FIELDS TERMINATED BY WHITESPACE   -- whitespace is the field separator
    (
    config             char(10),
    run                integer external,
    audit_str          char(10),
    proc_no            integer external,
    rep1               integer external,
    rep2               integer external,
    rep3               integer external,
    rep4               integer external,
    rep5               integer external,
    rep6               integer external,
    rep7               integer external,
    rep8               integer external,
    rep9               integer external,
    rep10              integer external,
    rep11              integer external,
    rep12              integer external,
    rep13              integer external,
    rep14              integer external,
    rep15              integer external,
    rep16              integer external,
    rep17              integer external,
    rep18              integer external,
    rep19              integer external,
    rep20              integer external,
    rep21              integer external,
    rep22              integer external,
    rep23              integer external,
    rep24              integer external,
    rep25              integer external,
    rep26              integer external,
    rep27              integer external,
    rep28              integer external,
    rep29              integer external,
    rep30              integer external,
    rep31              integer external,
    rep32              integer external,
    rep33              integer external,
    rep34              integer external,
    rep35              integer external,
    rep36              integer external,
    rep37              integer external,
    rep38              integer external,
    rep39              integer external,
    rep40              integer external,
    rep41              integer external,
    rep42              integer external,
    rep43              integer external,
    rep44              integer external,
    rep45              integer external,
    rep46              integer external,
    rep47              integer external,
    rep48              integer external,
    rep49              integer external,
    rep50              integer external,
    rep51              integer external,
    rep52              integer external,
    rep53              integer external,
    rep54              integer external,
    rep55              integer external,
    rep56              integer external,
    rep57              integer external,
    rep58              integer external,
    rep59              integer external,
    rep60              integer external,
    rep61              integer external,
    rep62              integer external,
    rep63              integer external,
    rep64              integer external,
    rep65              integer external,
    rep66              integer external,
    rep67              integer external,
    rep68              integer external,
    rep69              integer external,
    rep70              integer external,
    rep71              integer external,
    rep72              integer external,
    rep73              integer external,
    rep74              integer external,
    rep75              integer external,
    rep76              integer external,
    rep77              integer external,
    rep78              integer external,
    rep79              integer external,
    rep80              integer external,
    rep81              integer external,
    rep82              integer external,
```

```
    rep83               integer external,
    rep84               integer external,
    rep85               integer external,
    rep86               integer external,
    rep87               integer external,
    rep88               integer external,
    rep89               integer external,
    rep90               integer external,
    rep91               integer external,
    rep92               integer external,
    rep93               integer external,
    rep94               integer external,
    rep95               integer external,
    rep96               integer external,
    rep97               integer external,
    rep98               integer external,
    rep99               integer external,
    rep100              integer external,
    thk1                integer external,
    thk2                integer external,
    thk3                integer external,
    thk4                integer external,
    thk5                integer external,
    thk6                integer external,
    thk7                integer external,
    thk8                integer external,
    thk9                integer external,
    thk10               integer external,
    thk11               integer external,
    thk12               integer external,
    thk13               integer external,
    thk14               integer external,
    thk15               integer external,
    thk16               integer external,
    thk17               integer external,
    thk18               integer external,
    thk19               integer external,
    thk20               integer external,
    thk21               integer external,
    thk22               integer external,
    thk23               integer external,
    thk24               integer external,
    thk25               integer external,
    key1                integer external,
    key2                integer external,
    key3                integer external,
    key4                integer external,
    key5                integer external,
    key6                integer external,
    key7                integer external,
    key8                integer external,
    key9                integer external,
    key10               integer external
    )
BEGINDATA                 -- start inline data
```

bench/tpc/tpcc/loader/order.ctl

```
--
-- $Header: order.ctl 7030100.1 95/08/07 15:54:01 plai Generic<base> $
Copyr (c) 1994 Oracle
--
-- ======================================================================+
--         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA           |
--                      OPEN SYSTEMS PERFORMANCE GROUP                   |
--                           All Rights Reserved                         |
-- ======================================================================+
--  FILENAME
--      order.ctl
--  DESCRIPTION
--      This is a SQL*Loader control file.  It is used for
--      loading orders to the tpcc database.
--  USAGE
--      sqlldr[st] <user_name>/<password> <SQL*Loader control file>
-- ======================================================================*/

OPTIONS (DIRECT = TRUE, PARALLEL = TRUE)

UNRECOVERABLE

LOAD DATA
APPEND

INTO TABLE orders
APPEND
FIELDS TERMINATED BY WHITESPACE OPTIONALLY ENCLOSED BY '"'
(
   o_id                integer external,
   o_d_id              integer external,
   o_w_id              integer external,
   o_c_id              integer external,
   o_entry_d           date,
   o_carrier_id        integer external,
   o_ol_cnt            integer external,
   o_all_local         integer external
)
```

bench/tpc/tpcc/loader/ordline.ctl

```
--
-- $Header: ordline.ctl 7030100.1 95/08/07 15:54:11 plai Generic<base> $
Copyr (c) 1994 Oracle
--
-- ======================================================================+
--         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA           |
--                      OPEN SYSTEMS PERFORMANCE GROUP                   |
--                           All Rights Reserved                         |
-- ======================================================================+
--  FILENAME
--      ordline.ctl
--  DESCRIPTION
--      This is a SQL*Loader control file.  It is used for
--      loading order lines to the tpcc database.
--  USAGE
--      sqlldr[st] <user_name>/<password> <SQL*Loader control file>
-- ======================================================================*/

OPTIONS (DIRECT = TRUE, PARALLEL = TRUE)
```

```
UNRECOVERABLE

LOAD DATA
APPEND

INTO TABLE order_line
APPEND
FIELDS TERMINATED BY WHITESPACE OPTIONALLY ENCLOSED BY '"'
(
   ol_o_id             integer external,
   ol_d_id             integer external,
   ol_w_id             integer external,
   ol_number           integer external,
   ol_i_id             integer external,
   ol_supply_w_id      integer external,
   ol_delivery_d       date,
   ol_quantity         integer external,
   ol_amount           float external,
   ol_dist_info        char(24)
)
```

bench/tpc/tpcc/loader/pay.ctl

```
--
-- $Header: pay.ctl 7010000.1 94/10/03 10:18:15 pswong Generic<base> $
Copyr (c) 1993 Oracle
--
-- ================================================================+
--         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA      |
--                      OPEN SYSTEMS PERFORMANCE GROUP              |
--                           All Rights Reserved                    |
-- ================================================================+
--  FILENAME
--       pay.ctl
--  DESCRIPTION
--       This file is meant to be prepended to the tpcc_*.pay files.
-- ================================================================


LOAD DATA

INFILE *                          -- use inline data

INTO TABLE bench_pay_res          -- load the BENCH_PAY_RES table
APPEND                            -- OK if there are data already
FIELDS TERMINATED BY WHITESPACE   -- whitespace is the field separator
    (
    config             char(10),
    run                integer external,
    audit_str          char(10),
    proc_no            integer external,
    rep1               integer external,
    rep2               integer external,
    rep3               integer external,
    rep4               integer external,
    rep5               integer external,
    rep6               integer external,
    rep7               integer external,
    rep8               integer external,
    rep9               integer external,
    rep10              integer external,
    rep11              integer external,
    rep12              integer external,
    rep13              integer external,
    rep14              integer external,
    rep15              integer external,
    rep16              integer external,
    rep17              integer external,
    rep18              integer external,
    rep19              integer external,
    rep20              integer external,
    rep21              integer external,
    rep22              integer external,
    rep23              integer external,
    rep24              integer external,
    rep25              integer external,
    rep26              integer external,
    rep27              integer external,
    rep28              integer external,
    rep29              integer external,
    rep30              integer external,
    rep31              integer external,
    rep32              integer external,
    rep33              integer external,
    rep34              integer external,
    rep35              integer external,
    rep36              integer external,
    rep37              integer external,
    rep38              integer external,
    rep39              integer external,
    rep40              integer external,
    rep41              integer external,
    rep42              integer external,
    rep43              integer external,
    rep44              integer external,
    rep45              integer external,
    rep46              integer external,
    rep47              integer external,
    rep48              integer external,
    rep49              integer external,
    rep50              integer external,
    rep51              integer external,
    rep52              integer external,
    rep53              integer external,
    rep54              integer external,
    rep55              integer external,
    rep56              integer external,
    rep57              integer external,
    rep58              integer external,
    rep59              integer external,
    rep60              integer external,
    rep61              integer external,
    rep62              integer external,
    rep63              integer external,
    rep64              integer external,
    rep65              integer external,
    rep66              integer external,
    rep67              integer external,
    rep68              integer external,
    rep69              integer external,
    rep70              integer external,
```

```
    rep71               integer external,
    rep72               integer external,
    rep73               integer external,
    rep74               integer external,
    rep75               integer external,
    rep76               integer external,
    rep77               integer external,
    rep78               integer external,
    rep79               integer external,
    rep80               integer external,
    rep81               integer external,
    rep82               integer external,
    rep83               integer external,
    rep84               integer external,
    rep85               integer external,
    rep86               integer external,
    rep87               integer external,
    rep88               integer external,
    rep89               integer external,
    rep90               integer external,
    rep91               integer external,
    rep92               integer external,
    rep93               integer external,
    rep94               integer external,
    rep95               integer external,
    rep96               integer external,
    rep97               integer external,
    rep98               integer external,
    rep99               integer external,
    rep100              integer external,
    thk1                integer external,
    thk2                integer external,
    thk3                integer external,
    thk4                integer external,
    thk5                integer external,
    thk6                integer external,
    thk7                integer external,
    thk8                integer external,
    thk9                integer external,
    thk10               integer external,
    thk11               integer external,
    thk12               integer external,
    thk13               integer external,
    thk14               integer external,
    thk15               integer external,
    thk16               integer external,
    thk17               integer external,
    thk18               integer external,
    thk19               integer external,
    thk20               integer external,
    thk21               integer external,
    thk22               integer external,
    thk23               integer external,
    thk24               integer external,
    thk25               integer external,
    key1                integer external,
    key2                integer external,
    key3                integer external,
    key4                integer external,
    key5                integer external,
    key6                integer external,
    key7                integer external,
    key8                integer external,
    key9                integer external,
    key10               integer external
    )
BEGINDATA                 -- start inline data
```

bench/tpc/tpcc/loader/sto.ctl

```
--
-- $Header: sto.ctl 7010000.1 94/10/03 10:18:16 pswong Osd<base> $ Copyr
(c) 1993 Oracle
--
-- ================================================================+
--         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA      |
--                      OPEN SYSTEMS PERFORMANCE GROUP              |
--                           All Rights Reserved                    |
-- ================================================================+
--  FILENAME
--       sto.ctl
--  DESCRIPTION
--       This file is meant to be prepended to the tpcc_*.sto files.
-- ================================================================


LOAD DATA

INFILE *                            -- use inline data

INTO TABLE bench_sto_res            -- load the BENCH_STO_RES table
APPEND                              -- OK if there are data already
FIELDS TERMINATED BY WHITESPACE     -- whitespace is the field separator
    (
    config              char(10),
    run                 integer external,
    audit_str           char(10),
    proc_no             integer external,
    rep1                integer external,
    rep2                integer external,
    rep3                integer external,
    rep4                integer external,
    rep5                integer external,
    rep6                integer external,
    rep7                integer external,
    rep8                integer external,
    rep9                integer external,
    rep10               integer external,
    rep11               integer external,
    rep12               integer external,
    rep13               integer external,
    rep14               integer external,
    rep15               integer external,
    rep16               integer external,
    rep17               integer external,
    rep18               integer external,
    rep19               integer external,
    rep20               integer external,
    rep21               integer external,
    rep22               integer external,
    rep23               integer external,
```

```
    rep24               integer external,
    rep25               integer external,
    rep26               integer external,
    rep27               integer external,
    rep28               integer external,
    rep29               integer external,
    rep30               integer external,
    rep31               integer external,
    rep32               integer external,
    rep33               integer external,
    rep34               integer external,
    rep35               integer external,
    rep36               integer external,
    rep37               integer external,
    rep38               integer external,
    rep39               integer external,
    rep40               integer external,
    rep41               integer external,
    rep42               integer external,
    rep43               integer external,
    rep44               integer external,
    rep45               integer external,
    rep46               integer external,
    rep47               integer external,
    rep48               integer external,
    rep49               integer external,
    rep50               integer external,
    rep51               integer external,
    rep52               integer external,
    rep53               integer external,
    rep54               integer external,
    rep55               integer external,
    rep56               integer external,
    rep57               integer external,
    rep58               integer external,
    rep59               integer external,
    rep60               integer external,
    rep61               integer external,
    rep62               integer external,
    rep63               integer external,
    rep64               integer external,
    rep65               integer external,
    rep66               integer external,
    rep67               integer external,
    rep68               integer external,
    rep69               integer external,
    rep70               integer external,
    rep71               integer external,
    rep72               integer external,
    rep73               integer external,
    rep74               integer external,
    rep75               integer external,
    rep76               integer external,
    rep77               integer external,
    rep78               integer external,
    rep79               integer external,
    rep80               integer external,
    rep81               integer external,
    rep82               integer external,
    rep83               integer external,
    rep84               integer external,
    rep85               integer external,
    rep86               integer external,
    rep87               integer external,
    rep88               integer external,
    rep89               integer external,
    rep90               integer external,
    rep91               integer external,
    rep92               integer external,
    rep93               integer external,
    rep94               integer external,
    rep95               integer external,
    rep96               integer external,
    rep97               integer external,
    rep98               integer external,
    rep99               integer external,
    rep100              integer external,
    thk1                integer external,
    thk2                integer external,
    thk3                integer external,
    thk4                integer external,
    thk5                integer external,
    thk6                integer external,
    thk7                integer external,
    thk8                integer external,
    thk9                integer external,
    thk10               integer external,
    thk11               integer external,
    thk12               integer external,
    thk13               integer external,
    thk14               integer external,
    thk15               integer external,
    thk16               integer external,
    thk17               integer external,
    thk18               integer external,
    thk19               integer external,
    thk20               integer external,
    thk21               integer external,
    thk22               integer external,
    thk23               integer external,
    thk24               integer external,
    thk25               integer external,
    key1                integer external,
    key2                integer external,
    key3                integer external,
    key4                integer external,
    key5                integer external,
    key6                integer external,
    key7                integer external,
    key8                integer external,
    key9                integer external,
    key10               integer external
    )
BEGINDATA                 -- start inline data
```

bench/tpc/tpcc/loader/stock.ctl

```
--
-- $Header: stock.ctl 7030100.1 95/08/07 15:54:18 plai Osd<base> $ Copyr
(c) 1994 Oracle
--
```

```
-- ================================================================+
--         Copyright (c) 1994  Oracle Corp, Redwood Shores, CA     |
--                     OPEN SYSTEMS PERFORMANCE GROUP              |
--                          All Rights Reserved                    |
-- ================================================================+
--  FILENAME
--      stock.ctl
--  DESCRIPTION
--      This is a SQL*Loader control file.  It is used for
--      loading stocks to the tpcc database.
--  USAGE
--      sqlldr[st] <user_name>/<password> <SQL*Loader control file>
-- ================================================================*/

OPTIONS (DIRECT = TRUE, PARALLEL = TRUE)

UNRECOVERABLE

LOAD DATA
APPEND

INTO TABLE stock
APPEND
FIELDS TERMINATED BY WHITESPACE OPTIONALLY ENCLOSED BY '"'
(
   s_i_id              integer external,
   s_w_id              integer external,
   s_quantity          integer external,
   s_dist_01           char(24),
   s_dist_02           char(24),
   s_dist_03           char(24),
   s_dist_04           char(24),
   s_dist_05           char(24),
   s_dist_06           char(24),
   s_dist_07           char(24),
   s_dist_08           char(24),
   s_dist_09           char(24),
   s_dist_10           char(24),
   s_ytd               integer external,
   s_order_cnt         integer external,
   s_remote_cnt        integer external,
   s_data              char(50)
)
```

bench/tpc/tpcc/stored_proc/del.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       del.sql
rem  DESCRIPTION
rem       SQL script to create a stored procedure for delivery
rem       transactions.
rem ================================================================
rem

CREATE OR REPLACE PACKAGE delivery
IS
   TYPE intarray IS TABLE OF INTEGER INDEX BY BINARY_INTEGER;
   PROCEDURE deliver
   (
      ware_id           INTEGER,
      carrier_id        INTEGER,
      order_id   IN OUT intarray,
      retry      IN OUT INTEGER
   );
END;
/


CREATE OR REPLACE PACKAGE BODY delivery
IS
   PROCEDURE deliver
   (
      ware_id           INTEGER,
      carrier_id        INTEGER,
      order_id   IN OUT intarray,
      retry      IN OUT INTEGER
   )
   IS
      dist_id       INTEGER;
      cust_id       INTEGER;
      amount_sum    NUMBER;
      no_rowid      ROWID;
      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 n_cur IS
         SELECT no_o_id, rowid
         FROM new_order
         WHERE no_w_id = ware_id AND no_d_id = dist_id
         ORDER BY no_w_id, no_d_id, no_o_id;
   BEGIN

      FOR i IN 1 .. 10 LOOP
         dist_id := i;

         LOOP BEGIN
            OPEN n_cur;
            FETCH n_cur INTO order_id(i), no_rowid;

            IF (n_cur%NOTFOUND) THEN             -- no new order
               CLOSE n_cur;
               COMMIT;
               order_id(i) := 0;
               EXIT;
            END IF;

            CLOSE n_cur;

            DELETE FROM new_order
               WHERE rowid = no_rowid;

            UPDATE orders
```

```
            SET o_carrier_id = carrier_id
            WHERE o_d_id = dist_id AND o_w_id = ware_id AND
                  o_id = order_id(i);

         SELECT o_c_id
            INTO cust_id
            FROM orders
            WHERE o_d_id = dist_id AND o_w_id = ware_id AND
                  o_id = order_id(i);

         UPDATE order_line
            SET ol_delivery_d = SYSDATE
            WHERE ol_d_id = dist_id AND ol_w_id = ware_id AND
                  ol_o_id = order_id(i);

         SELECT sum(ol_amount)
            INTO amount_sum
            FROM order_line
            WHERE ol_d_id = dist_id AND ol_w_id = ware_id AND
                  ol_o_id = order_id(i);

         UPDATE customer
            SET c_balance = c_balance + amount_sum,
                c_delivery_cnt = c_delivery_cnt + 1
            WHERE c_id = cust_id AND c_d_id = dist_id AND c_w_id =
ware_id;

         COMMIT;
         EXIT;

         EXCEPTION
            WHEN not_serializable OR deadlock OR snapshot_too_old THEN
               ROLLBACK;
               retry := retry + 1;
         END;

      END LOOP;
   END LOOP;
END;
END;
/

quit;
```

bench/tpc/tpcc/stored_proc/new.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       new.sql
rem  DESCRIPTION
rem       SQL script to create a stored package for new order
rem       transactions.
rem ============================================================
rem
```

```
CREATE OR REPLACE PACKAGE neworder
IS
   PROCEDURE enterorder
   (
      ware_id                  INTEGER,
      dist_id                  INTEGER,
      cust_id                  INTEGER,
      ord_ol_cnt               INTEGER,
      ord_all_local            INTEGER,
      cust_discount        OUT NUMBER,
      cust_last            OUT VARCHAR2,
      cust_credit          OUT VARCHAR2,
      dist_tax             OUT NUMBER,
      ware_tax             OUT NUMBER,
      ord_id            IN OUT INTEGER,
      ord_entry_d       IN OUT VARCHAR2,
      retry             IN OUT INTEGER
   );
END;
/

CREATE OR REPLACE PACKAGE BODY neworder
IS
   PROCEDURE enterorder
   (
      ware_id                  INTEGER,
      dist_id                  INTEGER,
      cust_id                  INTEGER,
      ord_ol_cnt               INTEGER,
      ord_all_local            INTEGER,
      cust_discount        OUT NUMBER,
      cust_last            OUT VARCHAR2,
      cust_credit          OUT VARCHAR2,
      dist_tax             OUT NUMBER,
      ware_tax             OUT NUMBER,
      ord_id            IN OUT INTEGER,
      ord_entry_d       IN OUT VARCHAR2,
      retry             IN OUT INTEGER
   )
   IS
      timestamp                DATE;
      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 district SET d_next_o_id = d_next_o_id + 1
            WHERE d_id = dist_id AND d_w_id = ware_id;
         SELECT d_tax, d_next_o_id - 1
            INTO dist_tax, ord_id
            FROM district
            WHERE d_id = dist_id AND d_w_id = ware_id;
         SELECT c_discount, c_last, c_credit
            INTO cust_discount, cust_last, cust_credit
            FROM customer
            WHERE c_id = cust_id AND c_d_id = dist_id AND c_w_id =
ware_id;
         timestamp := SYSDATE;
```

```
         ord_entry_d := TO_CHAR(timestamp,'DD-MM-YYYY.HH24:MI:SS');
         INSERT INTO new_order VALUES (ord_id, dist_id, ware_id);
         INSERT INTO orders VALUES (ord_id, dist_id, ware_id, cust_id,
                                    timestamp, NULL, ord_ol_cnt,
ord_all_local);
         SELECT w_tax INTO ware_tax FROM warehouse
            WHERE w_id = ware_id;
         EXIT;

         EXCEPTION
            WHEN not_serializable OR deadlock OR snapshot_too_old THEN
               ROLLBACK;
               retry := retry + 1;
         END;
      END LOOP;
   END;
END;
/

quit;
```

bench/tpc/tpcc/stored_proc/ord.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       ord.sql
rem  DESCRIPTION
rem       SQL script to create a stored package for order status
rem       transactions.
rem ====================================================================
rem

CREATE OR REPLACE PACKAGE orderstatus
IS
   TYPE intarray IS TABLE OF INTEGER INDEX BY BINARY_INTEGER;
   TYPE numarray IS TABLE OF NUMBER INDEX BY BINARY_INTEGER;
   TYPE strarray IS TABLE OF VARCHAR2(11) INDEX BY BINARY_INTEGER;
   TYPE rowidarray IS TABLE OF ROWID INDEX BY BINARY_INTEGER;
   PROCEDURE getstatus
   (
      ware_id                  INTEGER,
      dist_id                  INTEGER,
      cust_id           IN OUT INTEGER,
      bylastname               INTEGER,
      cust_last         IN OUT VARCHAR2,
      cust_first           OUT VARCHAR2,
      cust_middle          OUT VARCHAR2,
      cust_balance         OUT NUMBER,
      ord_id            IN OUT INTEGER,
      ord_entry_d          OUT VARCHAR2,
      ord_carrier_id       OUT INTEGER,
      ord_ol_cnt           OUT INTEGER,
      oline_supply_w_id IN OUT intarray,
      oline_i_id        IN OUT intarray,
      oline_quantity    IN OUT intarray,
      oline_amount      IN OUT numarray,
      oline_delivery_d  IN OUT strarray
   );
END;
/

CREATE OR REPLACE PACKAGE BODY orderstatus
IS
   PROCEDURE getstatus
   (
      ware_id                  INTEGER,
      dist_id                  INTEGER,
      cust_id           IN OUT INTEGER,
      bylastname               INTEGER,
      cust_last         IN OUT VARCHAR2,
      cust_first           OUT VARCHAR2,
      cust_middle          OUT VARCHAR2,
      cust_balance         OUT NUMBER,
      ord_id            IN OUT INTEGER,
      ord_entry_d          OUT VARCHAR2,
      ord_carrier_id       OUT INTEGER,
      ord_ol_cnt           OUT INTEGER,
      oline_supply_w_id IN OUT intarray,
      oline_i_id        IN OUT intarray,
      oline_quantity    IN OUT intarray,
      oline_amount      IN OUT numarray,
      oline_delivery_d  IN OUT strarray
   )
   IS
      cust_rowid               ROWID;
      ol                       BINARY_INTEGER;
      c_num                    BINARY_INTEGER;
      row_id                   rowidarray;
      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 o_cur IS
         SELECT ol_i_id, ol_supply_w_id, ol_quantity, ol_amount,
                nvl(to_char(ol_delivery_d,'DD-MM-YYYY'), 'NOT DELIVR')
del_date
         FROM order_line
         WHERE ol_d_id = dist_id AND ol_w_id = ware_id AND ol_o_id =
ord_id;
      CURSOR c_cur IS
         SELECT rowid
         FROM customer
         WHERE c_d_id = dist_id AND c_w_id = ware_id AND c_last =
cust_last
         ORDER BY c_w_id, c_d_id, c_last, c_first;
   BEGIN

      LOOP BEGIN

         IF bylastname != 0 THEN

            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);

         SELECT c_id, c_balance, c_first, c_middle, c_last
            INTO cust_id, cust_balance, cust_first, cust_middle,
cust_last
            FROM customer
            WHERE rowid = cust_rowid;

      ELSE

         SELECT c_balance, c_first, c_middle, c_last
            INTO cust_balance, cust_first, cust_middle, cust_last
            FROM customer
            WHERE c_id = cust_id AND c_d_id = dist_id AND c_w_id =
ware_id;

      END IF;

      SELECT o_id, to_char(o_entry_d, 'DD-MM-YYYY.HH24:MI:SS'),
             nvl(o_carrier_id,0), o_ol_cnt
         INTO ord_id, ord_entry_d, ord_carrier_id, ord_ol_cnt
         FROM orders
         WHERE o_d_id = dist_id AND o_w_id = ware_id AND o_id =
            (SELECT max(o_id)
             FROM orders
             WHERE o_d_id = dist_id AND o_w_id = ware_id AND
                   o_c_id = cust_id);

      ol := 0;
      FOR o_cur_rec IN o_cur LOOP
         ol := ol + 1;
         oline_i_id(ol) := o_cur_rec.ol_i_id;
         oline_supply_w_id(ol) := o_cur_rec.ol_supply_w_id;
         oline_quantity(ol) := o_cur_rec.ol_quantity;
         oline_amount(ol) := o_cur_rec.ol_amount;
         oline_delivery_d(ol) := o_cur_rec.del_date;
      END LOOP;

      COMMIT;
      EXIT;

      EXCEPTION
         WHEN not_serializable OR deadlock OR snapshot_too_old THEN
            ROLLBACK;
      END;
   END LOOP;

END;
END;
/

quit;
```

bench/tpc/tpcc/stored_proc/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
   (
      ware_id                  INTEGER,
      dist_id                  INTEGER,
      cust_w_id                INTEGER,
      cust_d_id                INTEGER,
      cust_id           IN OUT INTEGER,
      bylastname               INTEGER,
      hist_amount              NUMBER,
      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
   );
END;
/

CREATE OR REPLACE PACKAGE BODY payment
IS
   PROCEDURE dopayment
   (
      ware_id                  INTEGER,
      dist_id                  INTEGER,
      cust_w_id                INTEGER,
      cust_d_id                INTEGER,
```

```
      cust_id           IN OUT INTEGER,
      bylastname              INTEGER,
      hist_amount             NUMBER,
      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
   )
   IS
      TYPE rowidarray IS TABLE OF ROWID INDEX BY BINARY_INTEGER;
      cust_rowid               ROWID;
      dist_name                VARCHAR2(11);
      ware_name                VARCHAR2(11);
      history_date             DATE;
      c_num                    BINARY_INTEGER;
      row_id                   rowidarray;
      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 customer
         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
      LOOP BEGIN

         IF bylastname != 0 THEN

            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);

            UPDATE customer
               SET c_balance = c_balance - hist_amount,
                   c_ytd_payment = c_ytd_payment + hist_amount,
                   c_payment_cnt = c_payment_cnt + 1
               WHERE rowid = cust_rowid;

            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
               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
               FROM customer
               WHERE rowid = cust_rowid;

         ELSE

            UPDATE customer
               SET c_balance = c_balance - hist_amount,
                   c_ytd_payment = c_ytd_payment + hist_amount,
                   c_payment_cnt = c_payment_cnt + 1
               WHERE c_id = cust_id AND c_d_id = cust_d_id AND
                     c_w_id = cust_w_id;

            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
               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
               FROM customer
               WHERE c_id = cust_id AND c_d_id = cust_d_id AND
                     c_w_id = cust_w_id;

         END IF;

         IF cust_credit = 'BC' THEN

            UPDATE customer
               SET c_data = 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') || '
| ')
                                    || c_data, 1, 500)
               WHERE rowid = cust_rowid;
```

```
            SELECT substr (c_data, 1, 200)
               INTO cust_data
               FROM customer
               WHERE rowid = cust_rowid;

         ELSE

            cust_data := ' ';

         END IF;

         UPDATE district
            SET d_ytd = d_ytd + hist_amount
            WHERE d_id = dist_id AND d_w_id = ware_id;

         SELECT d_name, d_street_1, d_street_2, d_city, d_state, d_zip
            INTO dist_name, dist_street_1, dist_street_2, dist_city,
                 dist_state, dist_zip
            FROM district
            WHERE d_id = dist_id AND d_w_id = ware_id;

         UPDATE warehouse
            SET w_ytd = w_ytd + hist_amount
            WHERE w_id = ware_id;

         SELECT w_name, w_street_1, w_street_2, w_city, w_state, w_zip
            INTO ware_name, ware_street_1, ware_street_2, ware_city,
                 ware_state, ware_zip
            FROM warehouse
            WHERE w_id = ware_id;

         history_date := sysdate;
         hist_date := to_char (history_date, 'DD-MM-YYYY.HH24:MI:SS');

         INSERT INTO history VALUES
            (cust_id, cust_d_id, cust_w_id, dist_id, ware_id,
history_date,
             hist_amount, ware_name || '    ' || dist_name);

         COMMIT;
         EXIT;

         EXCEPTION
            WHEN not_serializable OR deadlock OR snapshot_too_old THEN
               ROLLBACK;
               retry := retry + 1;
         END;

      END LOOP;
   END;
END;
/

quit;
```

bench/tpc/tpcc/stored_proc/sto.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       sto.sql
rem  DESCRIPTION
rem       SQL script to create a stored procedure for stock level
rem       transactions.
rem ======================================================================
rem

CREATE OR REPLACE PACKAGE stocklevel
IS
   PROCEDURE getstocklevel
   (
      ware_id       INTEGER,
      dist_id       INTEGER,
      threshold     INTEGER,
      low_stock OUT INTEGER
   );
END;
/

CREATE OR REPLACE PACKAGE BODY stocklevel
IS
   PROCEDURE getstocklevel
   (
      ware_id       INTEGER,
      dist_id       INTEGER,
      threshold     INTEGER,
      low_stock OUT INTEGER
   )
   IS
      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 count (DISTINCT s_i_id)
            INTO low_stock
            FROM order_line, stock, district
            WHERE d_id = dist_id AND d_w_id = ware_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);
         COMMIT;
         EXIT;

         EXCEPTION
            WHEN not_serializable OR deadlock OR snapshot_too_old THEN
               ROLLBACK;
         END;
      END LOOP;
```

```
   END;
END;
/

quit;
```

Appendix C - Tunable Parameters

stune.sut

```
NUMREGPT        256
NUMSCOPT        33
BUFHWM          8192
SFSNINODE       200
VXFSNINODE      4096
DNLCSIZE        4096
**
NPROC           2000
**
HDATLIM         0x7FFFFFFF
SDATLIM         0x7FFFFFFF
**
MAXRSS          250000
PSE_PHYSMEM     1887436800
SHMMAX          1887436800
* PSE_PHYSMEM   1929379840
* SHMMAX               1929379840
* PSE_PHYSMEM   1887436800
* SHMMAX               1887436800
*SHMMAX         1782579200
*PSE_PHYSMEM    1782579200
*MAXRSS         125000
*SHMMAX         998244352
*PSE_PHYSMEM    998244352
ET_AGE_INTERVAL         10
INIT_AGEQUANTUM         100
MIN_AGEQUANTUM 50
MAX_AGEQUANTUM 120
AIO_LISTIO_MAX 512
NUMAIO  2048
USER_RDTSC      1
SFSZLIM         0x7FFFFFFF
HFSZLIM         0x7FFFFFFF
SFNOLIM         128
SVMMLIM         0x7FFFFFFF
HVMMLIM         0x7FFFFFFF
NHBUF   256
MAXUP           2000
MAXULWP         2000
*TCP_FOREIGN_HASHBKTS  50
ODIMEM_NUMBUF  0
ODIMEM_MBLK_NUMBUF     0
PUTBUFSZ        8192
PDI_TIMEOUT     0
PRIV_L1_SIZE    1
*SEGKMEM_PERCENT       45
*SEGMAP_PERCENT        10
*SEGKVN_PERCENT        10
*PAGES_UNLOCK  10000
*MINAMEM                       2000
*KMEM_RESV             160
*PAGES_NODISKMA        1000
NPBUF                  2048
*FDFLUSHR       60
*NAUTOUP        600
*SEGKMEM_PSE_BYTES     20971520
SCORLIM 0x500000
HCORLIM 0x500000
IPFORWARDING    1
```

inittpcc.ora

```
#
ifile=/home/oracle/dbs/configtpcc.ora
#
#       Oracle parameter file for running TPC-C.
#

post_wait_device = /dev/pw
use_async_io = TRUE

spin_count = 2000
use_post_wait_driver    = TRUE


checkpoint_process      = TRUE
compatible              = 7.3.1
db_name                 = tpcc
db_files                = 100
db_file_multiblock_read_count = 32
db_block_buffers        = 834500
```

```
_db_block_hash_buckets = 100000

_db_block_write_batch   =  512

db_block_lru_latches      = 10

db_block_checkpoint_batch = 384
dml_locks                 = 500
log_archive_start         = FALSE
log_archive_buffer_size = 32
log_checkpoint_interval = 1000000000
log_checkpoints_to_alert = TRUE
log_buffer                = 1048576
gc_rollback_segments   = 220
gc_db_locks            = 100
gc_releasable_locks    = 100
max_rollback_segments  = 220
open_cursors             = 250
processes                = 250
sessions                 = 500
transactions             = 500
distributed_transactions = 0
transactions_per_rollback_segment = 1
rollback_segments        =
(t1,t2,t3,t4,t5,t6,t7,t8,t9,t10,t11,t12,t13,t14,t15,t16,t17,t18,t19,t20,t2
1,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,t5
8,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,t9
5,t96,t97,t98,t99,t100,t101,t102,t103,t104,t105,t106,t107,t108,t109,t110,t
111,t112,t113,t114,t115,t116,t117,t118,t119,t120)
shared_pool_size       = 7000000
discrete_transactions_enabled = FALSE
cursor_space_for_time  = TRUE
sql_trace=FALSE
timed_statistics=FALSE
_db_handles_cached = 5
```

stune.client

```
NUMREGPT       256
NUMSCOPT       32
BUFHWM  4096
SFSNINODE      200
VXFSNINODE     4096
DNLCSIZE       4096
SFNOLIM 128
MAXRSS  100000
ET_AGE_INTERVAL        10
INIT_AGEQUANTUM        100
MAX_AGEQUANTUM 120
MIN_AGEQUANTUM 50
AIO_LISTIO_MAX 256
NUMAIO  256
USER_RDTSC     1
SDATLIM 0X7FFFFFFF
HDATLIM 0X7FFFFFFF
SVMMLIM 0X7FFFFFFF
HVMMLIM 0X7FFFFFFF
SFSZLIM 0X7FFFFFFF
HFSZLIM 0X7FFFFFFF
NHBUF   256
NPROC   2000
MAXUP   2000
MAXULWP 2000
MSGMAX  8192
MSGMNB  131072
MSGMNI  1650
MSGSSZ  1048576
MSGTQL  1650
SHMSEG  15
SEMMNI  1450
SEMMSL  300
MAXLINK 4096
IPFORWARDING   0
ARG_MAX 65000
SHMMAX  20971520
MAXPMEM 0xD000000
SCORLIM 0x500000
HCORLIM 0x500000
```

ubbmp

```
#--------------------------------------------#
*RESOURCES                                   #
#--------------------------------------------#
#
IPCKEY          70952           # IPC KEY from 32,768 to 16,777,215
UID             105             # user id as displayed by command "id"
GID             102             # group id as displayed by command "id"
PERM            0666            # UNIX permission from 0001 to 0777 in octal
MAXACCESSERS    1000    # was 1800     # max no of processes accesing
bulletin board
MAXSERVERS      42              # maximum number of servers
MAXSERVICES     42              # maximum number of services
MASTER          SITE1           # machine on which master copy is found
MODEL           SHM             # SHM=single processor, MP=multi processor
LDBAL           Y               # load balancing, Y=yes, N=no
MAXGTT          500             # maximum simultaneous global transactions
#MAXBUFTYPE     8               # maximum buffer types
#MAXBUFSTYPE    16              # maximum buffer subtypes
SCANUNIT        30              # scan program wake-up time in secs.
SANITYSCAN      5               # sanity scan wake-up
BBLQUERY        60              # check out wake-up time
BLOCKTIME       10              # blocking call time-out
TAGENT          "TAGENT"        # alphanumeric service code

#
#--------------------------------------------#
*MACHINES                                    #
#--------------------------------------------#
#
reno_d      LMID=SITE1
          ROOTDIR="/home/oltp-t"
          APPDIR="/home/audit/server"
          TUXCONFIG="/home/audit/tux/tuxconfig"
```

```
          ULOGPFX="/tmp/ULOG"

#
#--------------------------------------------#
*GROUPS                                      #
#--------------------------------------------#
#
group1          GRPNO=1         # server group number
                LMID=SITE1      # logical machine identifier
group2          GRPNO=2         # server group number
                LMID=SITE1      # logical machine identifier
group3          GRPNO=3         # server group number
                LMID=SITE1      # logical machine identifier
group4          GRPNO=4         # server group number
                LMID=SITE1      # logical machine identifier
group5          GRPNO=5         # server group number
                LMID=SITE1      # logical machine identifier

#               OPENINFO=NONE  # resource mgr. info
#               CLOSEINFO=""   # resource mgr. info
#
#--------------------------------------------#
#*NETWORK                                    #
#--------------------------------------------#
#
#SITE1  NADDR="0x0000138881e233a5"
#       NLSADDR="0x0000138b81e233a5"
#       BRIDGE="/dev/tcp"
#--------------------------------------------#
*SERVERS                                     #
#--------------------------------------------#
#
#DEFAULT:       SRVGRP=GROUP1           # server group
#DEFAULT:       REPLYQ=N                # reply queue, Y=yes, N=no
#DEFAULT:       MAXGEN=5                # re-startable times
#DEFAULT:       RESTART=Y               # re-start status, Y=yes, N=no
#DEFAULT:       CLOPT="-A"              # booted options
#

# new_order servers
tpccsvr  SRVGRP=group1
        CLOPT="-s NEWO_SVC -- -n101 -- -Stpcc/tpcc"
        RQADDR=neworder REPLYQ=Y SRVID=101 MIN=19 MAX=20

# payment servers
tpccsvr  SRVGRP=group2
        CLOPT="-s PMT_SVC -- -n201 -- -Stpcc/tpcc "
        RQADDR=payment REPLYQ=Y SRVID=201 MIN=7 MAX=20

# order_status servers
tpccsvr  SRVGRP=group3
        CLOPT="-s ORDS_SVC -- -n301 -- -Stpcc/tpcc "
        RQADDR=orderstat REPLYQ=Y SRVID=301 MIN=1 MAX=20

# delivery servers
tpccsvr  SRVGRP=group4
        CLOPT="-s DVRY_SVC -- -n401 -- -Stpcc/tpcc "
        RQADDR=delivery REPLYQ=Y SRVID=401 MIN=3 MAX=20

# stock_level servers
tpccsvr  SRVGRP=group5
        CLOPT="-s STKL_SVC -- -n501 -- -Stpcc/tpcc "
        RQADDR=stocklevel REPLYQ=Y SRVID=501 MIN=7 MAX=20

#
#
#--------------------------------------------#
*SERVICES                                    #
#--------------------------------------------#
#
```

Appendix D - RTE Code

```
/*
% Sets up global variables, set up probability tables etc.          %
*/
void
INITIALIZE()
{
   PROB_ARRAY[NEW_ORDERV]     = NEW_ORDER_PROB;
   PROB_ARRAY[PAYMENTV]       = PAYMENT_PROB;
   PROB_ARRAY[ORDER_STATUSV] = ORDER_STATUS_PROB;
   PROB_ARRAY[DELIVERV]       = DELIVER_PROB;
   PROB_ARRAY[STOCK_LEVELV]   = STOCK_LEVEL_PROB;

}
/*
   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   % This function returns a value that represents the next transaction%
   % type to be executed. The value is picked with a weighting that    %
   % results in the transaction mix described in clause 5.2.5.7 of the %
   % TPCC spec.                                                        %
   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
*/
PICK()
{
        int HOLD,P;
        double T;
        double R;

        R = RND();
        if (R < (T = PROB_ARRAY[NEW_ORDERV]))
                P = NEW_ORDERV;
        else
                if (R < (T = T+PROB_ARRAY[PAYMENTV]))
                                P = PAYMENTV;
                        else
                                if (R < (T = T+PROB_ARRAY[ORDER_STATUSV]))
                                                P = ORDER_STATUSV;
                                        else
                                         if (R < (T = T+PROB_ARRAY[DELIVERV]))
                                                P = DELIVERV;
                                        else
                                                P = STOCK_LEVELV;
      return(P);

   }
```

Appendix E - Disk Storage

TPM	6253.32					
WAREHOUSES	600					
SEGMENT	**TYPE**	**TSPACE**	**BLOCKS**	**FIVE_PCT**	**DAILY_GROW**	**TOTAL**
CUSTOMER	TABLE	CUST	9000021	450001	0	9450022
DISTRICT	TABLE	WARE	6004	300	0	6304
HISTORY	TABLE	HIST	507820	0	84682	605321
ICUSTOMER	INDEX	ICUST1	206916	10346	0	217262
ICUSTOMER2	INDEX	ICUST2	473815	23691	0	497506
IDISTRICT	INDEX	WARE	1000	50	0	1050
IITEM	INDEX	ITEMS	1000	50	0	1050
INEW_ORDER	INDEX	INORD	69479	3474	0	72953
IORDERS	INDEX	IORD1	216452	10823	0	227275
IORDERS2	INDEX	IORD2	349113	17456	0	366569
IORDER_LINE	INDEX	IORDL	2415904	120795	0	2536699
ISTOCK	INDEX	ISTK	627994	31400	0	659394
ITEM	TABLE	ITEMS	6667	333	0	7000
IWAREHOUSE	INDEX	WARE	100	5	0	105
NEW_ORDER	TABLE	NORD	47721	2386	0	50107
ORDERS	TABLE	ORD	361011	0	60201	430325
ORDER_LINE	TABLE	ORDL	6542092	0	1090928	7798174
ROLL_SEG	SYS	ROLL	316928	0	0	316928
STOCK	TABLE	STOCKS	12000003	600000	0	12600003
SYSTEM	SYS	SYSTEM	311296	0	0	311296
WAREHOUSE	TABLE	WARE	604	30	0	634

TSPACE	**BLOCKS**	**REQUIRED**	**STATIC**	**DYNAMIC**	**OVERSIZE**
CUST	11255858	9450022	9450022	0	1805836
HIST	1029632	605321	0	507820	424311
ICUST1	622592	217262	217262	0	405330
ICUST2	811008	497506	497506	0	313502
INORD	316928	72953	72953	0	243975
IORD1	704512	227275	227275	0	477237
IORD2	1037824	366569	366569	0	671255
IORDL	7142962	2536699	2536699	0	4606263
ISTK	1047040	659394	659394	0	387646
ITEMS	13824	8050	8050	0	5774
NORD	311296	50107	50107	0	261189
ORD	792064	430325	0	361011	361739
ORDL	14414898	7798174	0	6542092	6616724
ROLL	316928	316928	316928	0	0
STOCKS	15547442	12600003	12600003	0	2947439
SYSTEM	311296	311296	311296	0	0
TEMP	7262258	0	0	0	7262258
WARE	13824	8093	8093	0	5731

TOTAL SPACE:			
STATIC	27322157		
DYNAMIC	7410923		
OVERSIZE	26796209		
DAILY_GROW	1235811		
DAILY_SPREAD	0		
180-DAY SPACE	249768137	= 476.39 GB	
USABLE DISK CAPACITY:		3.99 GB	
TOTAL DISKS:			
DATABASE	120	disks	
AUDIT LOG (8-hour)	20	disks	10 mirrored pairs
OS + ORACLE	1	disk	
TOTAL	**141**	**disks**	

Appendix F - Third-Party Price Quotations

ID:12030551386 JUN 12'96 6:09 No.023 P.02

Data Comm\ Warehouse

47 Water Street, South Norwalk, CT 06854

Phone 1-800-328-2261
Fax 203-866-4207

June 12 1996
Richard Garcia

Thank you for your interest in Data Comm Warehouse
Please accept this quotation for the following items:

ITEM #		DESCRIPTION	UNIT	TOTAL
DEH1477	230	Ultra Slimline 24 port hub	$275.00	$63,250.00
			TOTAL	$63,250.00
		Prices are subject to change		

Your Account Manager. Carol Gaffney 1-800-328-2261 ext 7232

*** CONFIDENTIAL ***

SYSTECH CORPORATION
386 Main Street
Redwood City, California 94063

TEL 415-365-5340
FAX 415-365-5448
tom_keating@systech.com at unixlink

DATE: June 24, 1996

TO: Unisys Corp.

FAX:408-954-8009

Pages including this cover sheet: ___1__

From: Tom Keating

Re: Etherplex Prices

Model	Quantity	Unit Price
UCC-2048	12	$3496

The above price is for 12 25MHz RISC-based EISA Etherplex boards with SCO and UnixWare device drivers, each board supports 2048 TCP/IP sessions and includes a 1 year warranty.

For a five [5] year warranty please add $1290 per unit.

Prices are FOB San Diego, CA.

For purchase or additional information please contact Systech at 1-800-205-0610.

Tom Keating

BEA Systems Inc
130 Campus Drive
Edison, New Jersey 08818-6646

Telephone (908) 346-3100
Fax (908) 417 9775

BEA

Enterprise Middleware Solutions

July 12, 1996

To. Unisys Corp.

Subj: TUXEDO 6.1 Pricing

List price for TUXEDO version 6.1 on the U6000/230 is $20,000

Maintenance for 5 x 8 support is $3000 per year.

Sincerely,

E. D. Ott

E. D. Ott
Director - Indirect Channels