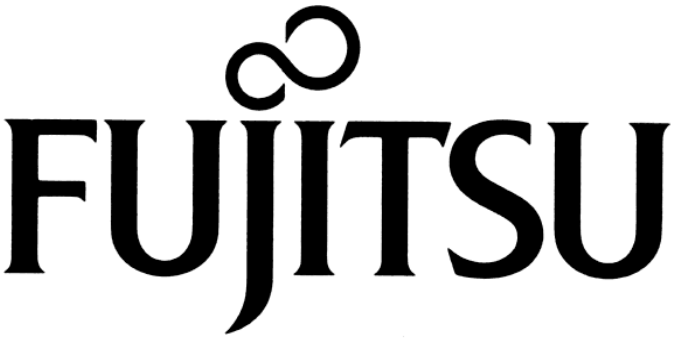


and

TPC Benchmark™C

Full Disclosure Report

Fujitsu
GRANPOWER 7000
Model 200

running

Oracle V7.3

January, 1997

The benchmark results contained in this document were submitted for compliance with version 3.2 of the TPC Benchmark C Standard Specification. The result of that action is to place these benchmark results into the sixty day "under review" status as of January 27, 1997.

Fujitsu and Oracle Corp. 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. Fujitsu and Oracle Corp. 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, Fujitsu provides no warranty of the pricing information in this document.

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

All performance data contained in this report were obtained in a rigorously controlled environment. Results obtained in other operating environments may vary significantly. Fujitsu does not warrant or represent that a user can or will achieve similar performance expressed in transactions per minute (tpmC) or normalized price/performance (¥/tpmC). No warranty of system performance or price/performance is expressed or implied in this report.

Printed in the United States January 27, 1997

UXP/DS V20 is derived from UNIX System V Release 4.2

UXP/DS is a trademark of Fujitsu Limited in Japan.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/OPEN Company Limited.

ORACLE, SQL*DBA, SQL*Loader, SQL*Net, SQL*Plus, Oracle7, Pro*C and PL/SQL are trademarks of Oracle Corporation.

TP-Base V20 is derived from TUXEDO, which is a registered trademark of Novell, Inc.

TP-Base is a trademark of Fujitsu Limited in Japan.

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

Preface

The TPC Benchmark C was developed by the Transaction Processing Performance Council (TPC). The TPC was founded to define transaction processing benchmarks and to disseminate objective, verifiable performance data to the industry. This full disclosure report is based on the TPC Benchmark C Standard Specifications Version 3.2, released August 27, 1996.

TPC Benchmark C Overview

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

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

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

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

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

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

Abstract

Overview

This report documents the methodology and results of the TPC Benchmark C test conducted by Fujitsu Ltd. and Oracle Corp. on the Fujitsu GRANPOWER 7000 Model 200. The operating system used for the benchmark was UXP/DS V20. The DBMS used was Oracle V7.3.

TPC Benchmark C Metrics

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

4,718.73 tpmC
¥101,988 per tpmC
Available as of July, 1997

Standard and Executive Summary Statements

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

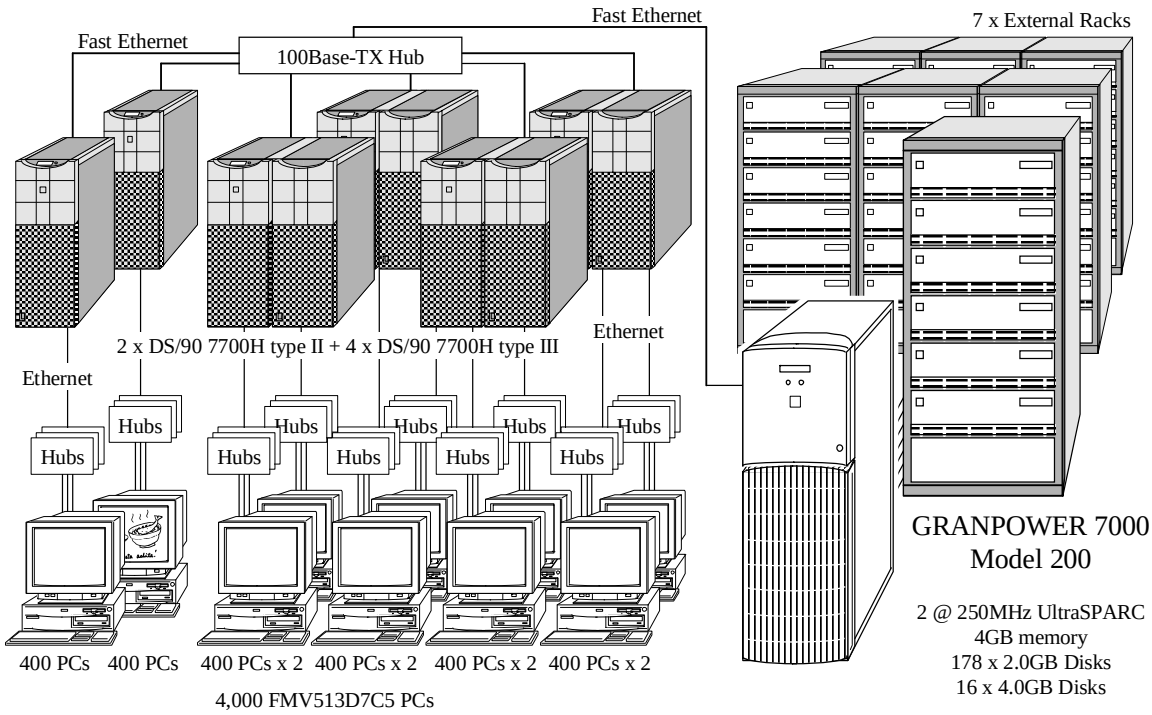
Auditor

The benchmark configuration, environment and methodology, along with the pricing model used to calculate the cost per tpmC, were audited by Lorna Livingtree of Performance Metrics, Inc. to verify compliance with the relevant TPC specifications.

Priced Configuration

FUJITSU / ORACLE®	Fujitsu GRANPOWER 7000 Model 200 C/S with 6 Front-Ends		TPC-C Rev. 3.2
			Report Date: Jan. 1997
Total System Cost	TPC-C Throughput	Price/Performance	Availability Date
481,255,900 Yen	4,718.73 tpmC	101,988 Yen/tpmC	July 1997

Processors	Database Manager	Operating System	Other Software	Number of Users
2 @ 250MHz UltraSPARC	Oracle7 RDBMS Version 7.3 Japanese Version	UXP/DS Basic Software V20	TP-Base V20	4,000



System Components	Qty	Server Description	Qty	Clients Description
Processor	2	UltraSPARC @ 250MHz	4	4 hyperSPARC @ 142MHz
			2	2 hyperSPARC @ 142MHz
Cache Memory		1MB (each processor)		1MB (each processor)
Memory		4,096MB	4	1,024MB
			2	512MB
Disk Controller	1	Wide-SCSI (1 Channel, internal)	6	SCSI-2 (1 Channel)
	17	Wide-SCSI (1 Channel)		
Disks	178	2.0GB Disk	4	2.0GB Disk
	16	4.0GB Disk	2	1GB Disk
Total Disk Storage		420.0GB		10GB
Terminals	1	Console	6	1 Console
Hubs	1	8-port (100Base-TX)	120	36-port (10Base-T)

Detailed Pricing information

FUJITSU / ORACLE®	Fujitsu GRANPOWER 7000 Model 200 C/S with 6 Front-Ends	TPC-C Rev. 3.2
		Report Date: Jan. 1997

1997/1/16

Server Systems Department,Marketing Division,Marketing Group

FUJITSU LIMITED

Order Number	Description	Quantity	Unit Price	Extended Price	Maintenance rate/unit*	5 Years Maintenance
Server Hardware						
GP720A202	GRANPOWER 7000 model 200 (2cpu)	1	6,050,000	6,050,000	25,200	1,360,800
	with 100/10Mbps LAN adapter					
GP762M8202	Additional memory (4GB)	1	50,400,000	50,400,000	0	0
F7978SB2	SBus extention unit	3	900,000	2,700,000	4,500	729,000
F7958HS1	Wide SCSI-2 adapter	17	250,000	4,250,000	0	0
F7945A5E	Addition Wide-SCSI Disk unit (2GB)	177	380,000	67,260,000	1,900	18,160,200
F7973D41A	Addition Wide-SCSI Disk unit (4GB)	16	680,000	10,880,000	3,400	2,937,600
F7949RA3	External Rack	7	630,000	4,410,000	3,200	1,209,600
F7949FU2A	External File unit	34	700,000	23,800,000	3,500	6,426,000
F7960A11	Display unit	1	380,000	380,000	1,500	81,000
DCBL-RCB05	RS-232 cable	1	16,000	16,000	0	0
F7953A4A	8mm Tape device	1	840,000	840,000	4,200	226,800
	Server Hardware Subtotals			170,986,000		31,131,000
Server Software						
B7831DK62	UXP/DS Basic Software V20	1	262,000	262,000	440,000	2,200,000
	ORACLE7 RDBMS & SQL*Net	1	38,471,850	38,471,850	7,694,370	38,471,850
	Server Software Subtotals			38,733,850		40,671,850
Client Hardware						
F7970C5	DS/90 7000 model 7700H type II	2	4,700,000	9,400,000	19,600	2,116,800
	with 10Mbps LAN adapter					
F7970C3	DS/90 7000 model 7700H type III	4	9,050,000	36,200,000	37,700	8,143,200
	with 64MB RAM and 10Mbps LAN adapter					
F7952M31	Additional memory (64MB)	76	864,000	65,664,000	0	0
F7958FE1	100Mbps LAN adapter	6	185,000	1,110,000	0	0
F7930LA2D	10Mbps LAN adapter	4	120,000	480,000	0	0
F7960A11	Display unit	6	380,000	2,280,000	1,500	486,000
DCBL-RCB05	RS-232 cable	6	16,000	96,000	0	0
	Client Hardware Subtotals			115,230,000		10,746,000
Client Software						
B78315K6H	UXP/DS Basic Software V20 (for type III)	1	1,040,000	1,040,000	220,000	1,100,000
B78310K0H	Additional user license	3	880,000	2,640,000	220,000	3,300,000
B78315K6G	UXP/DS Basic Software V20 (for type II)	1	740,000	740,000	220,000	1,100,000
B78310K0G	Additional user license	1	640,000	640,000	220,000	1,100,000
D783HZK60	TP-Base/sdk V20 (1 user)	1	300,000	300,000	270,600	1,353,000
D783HUK62	TP-Base/rt V20 (8 user)	1	500,000	500,000	270,600	1,353,000
S783HUK02	Additional user license	5	400,000	2,000,000	88,000	2,200,000
	Client Software Subtotals			7,860,000		11,506,000
User Connectivity						
SH2500	100BASE-TX Switching Hub units (8ports) *	3	1,280,000	3,840,000	6,000	972,000
LH36XA	Hub units (36ports) *	132	300,000	39,600,000	1,400	9,979,200
	User Connectivity Subtotals			43,440,000		10,951,200
	Totals			376,249,850		105,006,050
	5 Year cost					481,255,900
	tpmC					4718.73
	Yen / tpmC					101,988
	(* 10% or minimum of 2 spares are included.)					

Notes:

- Audited by Performance Metrics Inc.
- Japanese yen prices are not convertible to other currencies at exchange rates.
- GRANPOWER hardware has a 6 month warranty. Thus to cost 5 years of hardware maintenance, a total of 54 months is calculated.

Prices used in TPC benchmarks reflect the actual prices a customer would pay for a one-time purchase of the stated components. Individually negotiated discounts are not permitted. Special prices based on assumptions about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed components. For complete details, see the pricing sections of the TPC benchmark specifications. If you find that the stated prices are not available according to these items, please inform the TPC at pricing@tpc.org. Thank you.

Numerical Quantities Summary

GRANPOWER 7000 Model 200 Oracle V7.3

MQTH, Computed Maximum Qualified Throughput	4718.73 tpmC		
Response Times (in seconds)	**Average**	**90%**	**Max.**
New-Order	**1.54**	**2.99**	**16.11**
Payment	**1.10**	**2.47**	**16.11**
Order-Status	**1.17**	**2.54**	**14.64**
Delivery (interactive portion)	**0.13**	**0.25**	**2.45**
Delivery (deferred portion)	**1.44**	**2.95**	**16.57**
Stock-Level	**1.03**	**2.38**	**15.53**
Menu	**0.17**	**0.27**	**29.52**

Transaction Mix, in percent of total transaction	
New-Order	**44.62**
Payment	**43.29**
Order-Status	**4.05**
Delivery	**4.00**
Stock-Level	**4.04**

Emulation Delay (in seconds)	**Resp. Time**	**Menu**
New-Order	N/A	N/A
Payment	N/A	N/A
Order-Status	N/A	N/A
Delivery (interactive)	N/A	N/A
Stock-Level	N/A	N/A

Keying/Think Times (in seconds)	**Min.**		**Average**		**Max.**	
New-Order	18.02	0.00	18.21	12.17	18.68	120.83
Payment	3.01	0.00	3.07	12.25	3.46	121.33
Order-Status	2.01	0.01	2.06	10.14	2.40	81.29
Delivery (interactive)	2.01	0.00	2.06	5.06	2.54	44.37
Stock-Level	2.01	0.00	2.06	5.04	2.37	48.83

Test Duration	
Ramp-up time (seconds)	**2340**
Measurement interval	**1800**
Transactions during measurement interval	**141562**
Ramp down time	

Checkpointing	
Number of checkpoints	**1**
Checkpoint interval	**1800 sec.**

Reproducibility Run	
Reported measurement	**4718.73**
Reproducibility measurement	**4691.90**
Difference	**26.83**

Table Of Contents

ORACLE

FUJITSU

General Items

Application Code and Definition Statements

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

Appendix A contains all source code implemented in this benchmark.

Test Sponsor

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

Fujitsu and Oracle Corp. were joint sponsors of this TPC Benchmark C.

Parameter Settings

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

- *Database options,*
- *Recover/commit options,*
- *Consistency/locking options*
- *Operating system and application configuration parameter.*

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

Appendix D contains the parameters for the database, the operating system, and the configuration for the transaction monitor.

Configuration Items

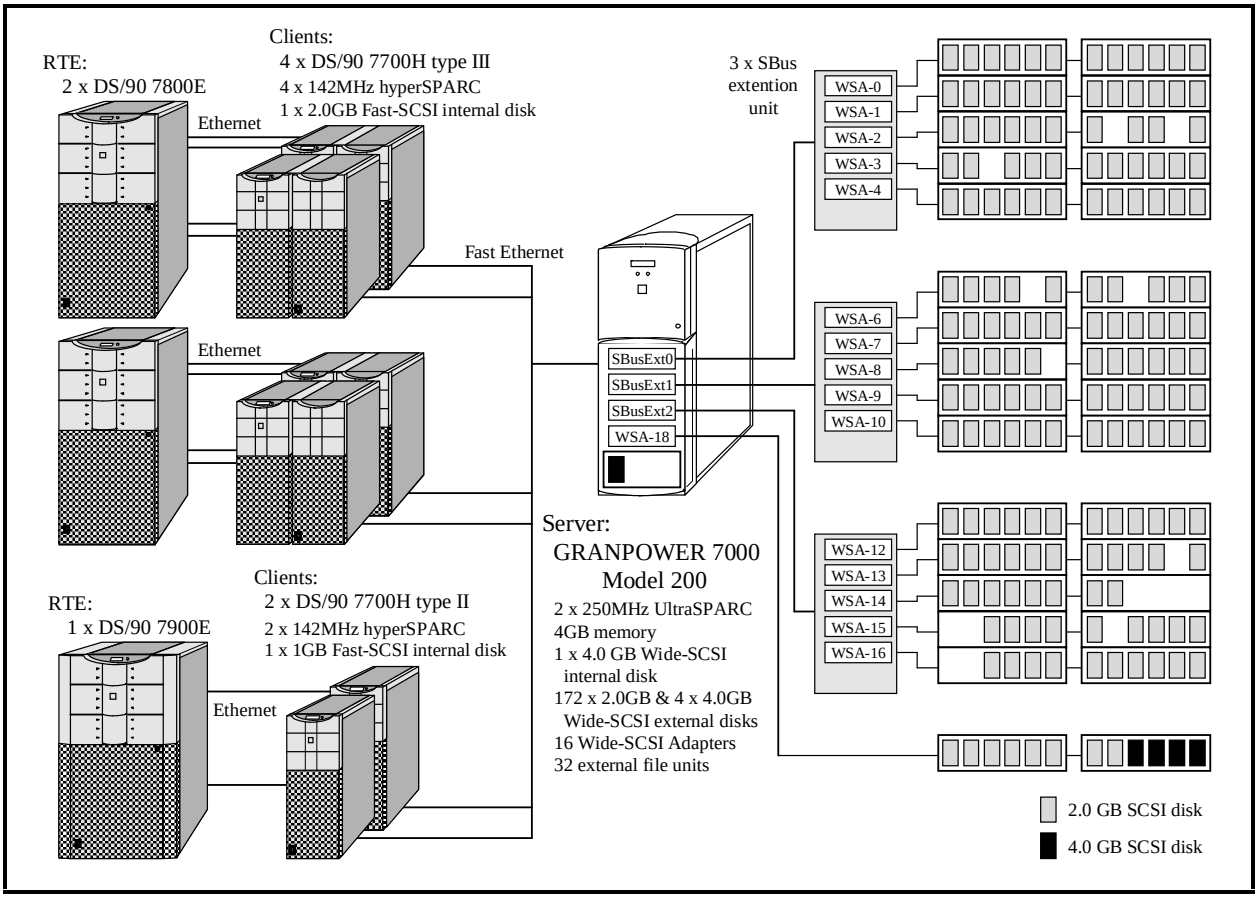
Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences.

The System Under Test (SUT), a GRANPOWER 7000 Model 200, is depicted in the following diagrams.

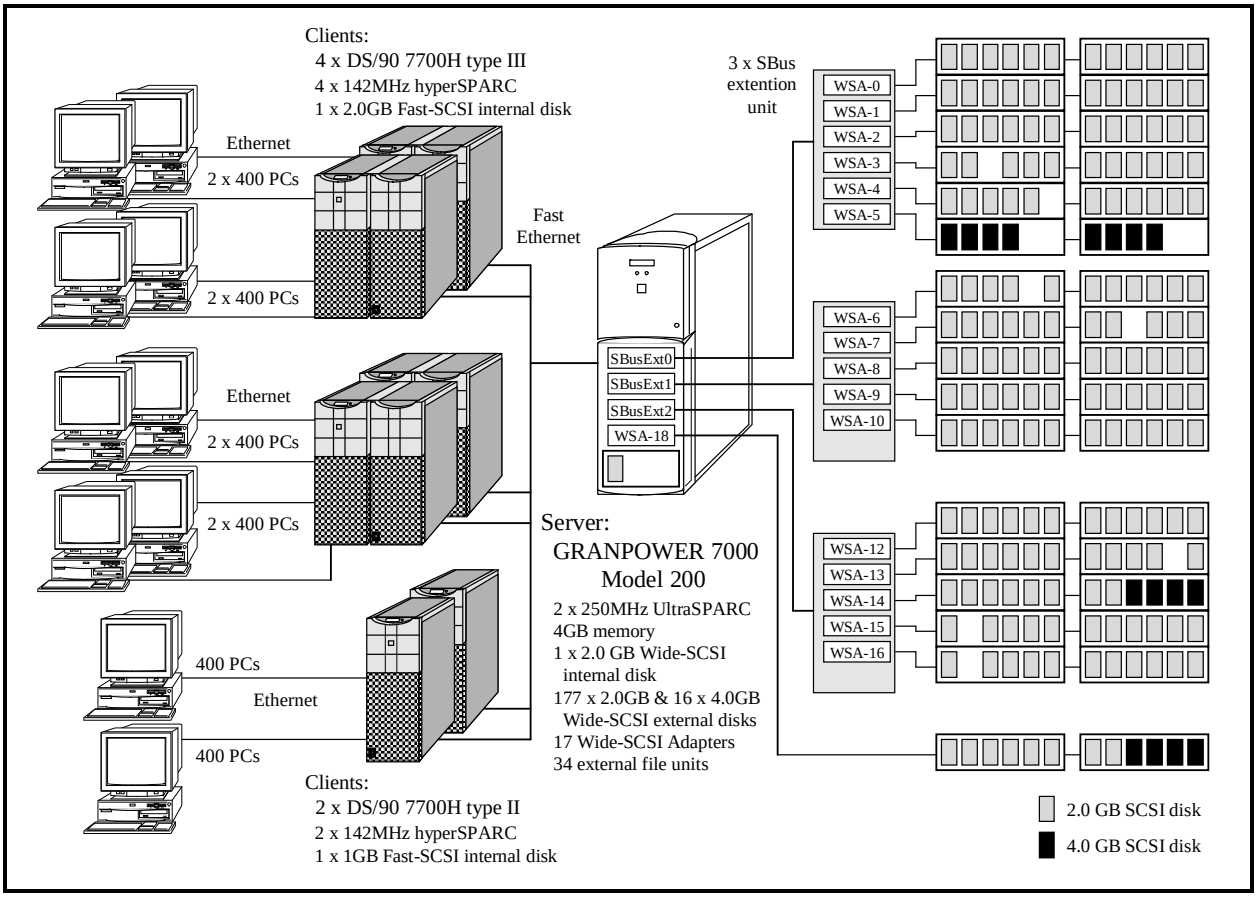
The configuration diagrams for both the tested and priced systems are included on the following pages.

The only difference is the number of disks and the use of the RTE.

GRANPOWER 7000 Tested Configuration



GRANPOWER 7000 Priced Configuration



Clause 1 Related Items

1.1. Table Definitions

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

Appendix E contains the code used to define and load the database tables.

1.2. Physical Organization of Database

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

The following table depicts the organization of tables and indices on the disks.

Distribution of Tables and Logs for GRANPOWER 7000

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
SA	hd00	Operating System		1567	4.0GB
			swap	465	
WSA-0	hda10	HISTORY	/dev/rdsk/hda1001	16	2.0GB
		STOCK	/dev/rdsk/hda1002	159	
		CUSTOMER	/dev/rdsk/hda1003	147	
		ORDERS	/dev/rdsk/hda1004	13	
		NEW ORDER	/dev/rdsk/hda1005	5	
	hda11	HISTORY	/dev/rdsk/hda1101	16	2.0GB
		STOCK	/dev/rdsk/hda1102	159	
		CUSTOMER	/dev/rdsk/hda1103	147	
		ORDERS	/dev/rdsk/hda1104	13	
		NEW ORDER	/dev/rdsk/hda1105	5	
	hda12	HISTORY	/dev/rdsk/hda1201	16	2.0GB
		STOCK	/dev/rdsk/hda1202	159	
		CUSTOMER	/dev/rdsk/hda1203	147	
		ORDERS	/dev/rdsk/hda1204	13	
		NEW ORDER	/dev/rdsk/hda1205	5	
	hda13	HISTORY	/dev/rdsk/hda1301	16	2.0GB
		STOCK	/dev/rdsk/hda1302	159	
		CUSTOMER	/dev/rdsk/hda1303	147	
		ORDERS	/dev/rdsk/hda1304	13	
		NEW ORDER	/dev/rdsk/hda1305	5	
	hda14	HISTORY	/dev/rdsk/hda1401	16	2.0GB
		STOCK	/dev/rdsk/hda1402	159	
		CUSTOMER	/dev/rdsk/hda1403	147	
		ORDERS	/dev/rdsk/hda1404	13	
		NEW ORDER	/dev/rdsk/hda1405	5	
	hda15	HISTORY	/dev/rdsk/hda1501	16	2.0GB
		STOCK	/dev/rdsk/hda1502	159	
		CUSTOMER	/dev/rdsk/hda1503	147	
		ORDERS	/dev/rdsk/hda1504	13	
		NEW ORDER	/dev/rdsk/hda1505	5	
	hda20	HISTORY	/dev/rdsk/hda2001	16	2.0GB
		STOCK	/dev/rdsk/hda2002	159	
		CUSTOMER	/dev/rdsk/hda2003	147	
		ORDERS	/dev/rdsk/hda2004	13	
		NEW ORDER	/dev/rdsk/hda2005	5	
	hda21	HISTORY	/dev/rdsk/hda2101	16	2.0GB
		STOCK	/dev/rdsk/hda2102	159	
		CUSTOMER	/dev/rdsk/hda2103	147	
		ORDERS	/dev/rdsk/hda2104	13	
		NEW ORDER	/dev/rdsk/hda2105	5	
	hda22	HISTORY	/dev/rdsk/hda2201	16	2.0GB
		STOCK	/dev/rdsk/hda2202	159	
		CUSTOMER	/dev/rdsk/hda2203	147	
		ORDERS	/dev/rdsk/hda2204	13	
		NEW ORDER	/dev/rdsk/hda2205	5	
	hda23	HISTORY	/dev/rdsk/hda2301	16	2.0GB
		STOCK	/dev/rdsk/hda2302	159	

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
		CUSTOMER	/dev/rdsk/hda2303	147	
		ORDERS	/dev/rdsk/hda2304	13	
		NEW ORDER	/dev/rdsk/hda2305	5	
	hda24	ORDER LINE INDEX	/dev/rdsk/hda2401	577	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hda2402	27	
		ORDER INDEX 1	/dev/rdsk/hda2403	58	
		CUSTOMER INDEX 2	/dev/rdsk/hda2404	64	
	hda25	ORDER LINE INDEX	/dev/rdsk/hda2501	577	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hda2502	27	
		ORDER INDEX 1	/dev/rdsk/hda2503	58	
		CUSTOMER INDEX 2	/dev/rdsk/hda2504	64	
WSA-1	hda30	HISTORY	/dev/rdsk/hda3001	16	2.0GB
		STOCK	/dev/rdsk/hda3002	159	
		CUSTOMER	/dev/rdsk/hda3003	147	
		ORDERS	/dev/rdsk/hda3004	13	
		NEW ORDER	/dev/rdsk/hda3005	5	
	hda31	HISTORY	/dev/rdsk/hda3101	16	2.0GB
		STOCK	/dev/rdsk/hda3102	159	
		CUSTOMER	/dev/rdsk/hda3103	147	
		ORDERS	/dev/rdsk/hda3104	13	
		NEW ORDER	/dev/rdsk/hda3105	5	
	hda32	HISTORY	/dev/rdsk/hda3201	16	2.0GB
		STOCK	/dev/rdsk/hda3202	159	
		CUSTOMER	/dev/rdsk/hda3203	147	
		ORDERS	/dev/rdsk/hda3204	13	
		NEW ORDER	/dev/rdsk/hda3205	5	
	hda33	HISTORY	/dev/rdsk/hda3301	16	2.0GB
		STOCK	/dev/rdsk/hda3302	159	
		CUSTOMER	/dev/rdsk/hda3303	147	
		ORDERS	/dev/rdsk/hda3304	13	
		NEW ORDER	/dev/rdsk/hda3305	5	
	hda34	HISTORY	/dev/rdsk/hda3401	16	2.0GB
		STOCK	/dev/rdsk/hda3402	159	
		CUSTOMER	/dev/rdsk/hda3403	147	
		ORDERS	/dev/rdsk/hda3404	13	
		NEW ORDER	/dev/rdsk/hda3405	5	
	hda35	HISTORY	/dev/rdsk/hda3501	16	2.0GB
		STOCK	/dev/rdsk/hda3502	159	
		CUSTOMER	/dev/rdsk/hda3503	147	
		ORDERS	/dev/rdsk/hda3504	13	
		NEW ORDER	/dev/rdsk/hda3505	5	
	hda40	HISTORY	/dev/rdsk/hda4001	16	2.0GB
		STOCK	/dev/rdsk/hda4002	159	
		CUSTOMER	/dev/rdsk/hda4003	147	
		ORDERS	/dev/rdsk/hda4004	13	
		NEW ORDER	/dev/rdsk/hda4005	5	
	hda41	HISTORY	/dev/rdsk/hda4101	16	2.0GB
		STOCK	/dev/rdsk/hda4102	159	
		CUSTOMER	/dev/rdsk/hda4103	147	
		ORDERS	/dev/rdsk/hda4104	13	
		NEW ORDER	/dev/rdsk/hda4105	5	

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
	hda42	HISTORY	/dev/rdsk/hda4201	16	2.0GB
		STOCK	/dev/rdsk/hda4202	159	
		CUSTOMER	/dev/rdsk/hda4203	147	
		ORDERS	/dev/rdsk/hda4204	13	
		NEW ORDER	/dev/rdsk/hda4205	5	
	hda43	ROLLBACK	/dev/rdsk/hda4301	50	2.0GB
		TEMP	/dev/rdsk/hda4302	932	
	hda44	ORDER LINE INDEX	/dev/rdsk/hda4401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hda4402	26	
		ORDER INDEX 1	/dev/rdsk/hda4403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hda4404	63	
	hda45	ORDER INDEX 2	/dev/rdsk/hda4501	61	2.0GB
WSA-2	hda50	HISTORY	/dev/rdsk/hda5001	16	2.0GB
		STOCK	/dev/rdsk/hda5002	159	
		CUSTOMER	/dev/rdsk/hda5003	147	
		ORDERS	/dev/rdsk/hda5004	13	
		NEW ORDER	/dev/rdsk/hda5005	5	
	hda51	HISTORY	/dev/rdsk/hda5101	16	2.0GB
		STOCK	/dev/rdsk/hda5102	159	
		CUSTOMER	/dev/rdsk/hda5103	147	
		ORDERS	/dev/rdsk/hda5104	13	
		NEW ORDER	/dev/rdsk/hda5105	5	
	hda52	HISTORY	/dev/rdsk/hda5201	16	2.0GB
		STOCK	/dev/rdsk/hda5202	159	
		CUSTOMER	/dev/rdsk/hda5203	147	
		ORDERS	/dev/rdsk/hda5204	13	
		NEW ORDER	/dev/rdsk/hda5205	5	
	hda53	HISTORY	/dev/rdsk/hda5301	16	2.0GB
		STOCK	/dev/rdsk/hda5302	159	
		CUSTOMER	/dev/rdsk/hda5303	147	
		ORDERS	/dev/rdsk/hda5304	13	
		NEW ORDER	/dev/rdsk/hda5305	5	
	hda54	HISTORY	/dev/rdsk/hda5401	16	2.0GB
		STOCK	/dev/rdsk/hda5402	159	
		CUSTOMER	/dev/rdsk/hda5403	147	
		ORDERS	/dev/rdsk/hda5404	13	
		NEW ORDER	/dev/rdsk/hda5405	5	
	hda55	HISTORY	/dev/rdsk/hda5501	16	2.0GB
		STOCK	/dev/rdsk/hda5502	159	
		CUSTOMER	/dev/rdsk/hda5503	147	
		ORDERS	/dev/rdsk/hda5504	13	
		NEW ORDER	/dev/rdsk/hda5505	5	
	hda60	HISTORY	/dev/rdsk/hda6001	16	2.0GB
		STOCK	/dev/rdsk/hda6002	159	
		CUSTOMER	/dev/rdsk/hda6003	147	
		ORDERS	/dev/rdsk/hda6004	13	
		NEW ORDER	/dev/rdsk/hda6005	5	
	hda62	HISTORY	/dev/rdsk/hda6201	16	2.0GB
		STOCK	/dev/rdsk/hda6202	159	
		CUSTOMER	/dev/rdsk/hda6203	147	
		ORDERS	/dev/rdsk/hda6204	13	

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
		NEW ORDER	/dev/rdsk/hda6205	5	
	hda63	ROLLBACK	/dev/rdsk/hda6301	50	2.0GB
		TEMP	/dev/rdsk/hda6302	932	
	hda65	ORDER INDEX 2	/dev/rdsk/hda6501	61	2.0GB
WSA-3	hdb10	HISTORY	/dev/rdsk/hdb1001	16	2.0GB
		STOCK	/dev/rdsk/hdb1002	159	
		CUSTOMER	/dev/rdsk/hdb1003	147	
		ORDERS	/dev/rdsk/hdb1004	13	
		NEW ORDER	/dev/rdsk/hdb1005	5	
	hdb11	HISTORY	/dev/rdsk/hdb1101	16	2.0GB
		STOCK	/dev/rdsk/hdb1102	159	
		CUSTOMER	/dev/rdsk/hdb1103	147	
		ORDERS	/dev/rdsk/hdb1104	13	
		NEW ORDER	/dev/rdsk/hdb1105	5	
	hdb13	HISTORY	/dev/rdsk/hdb1301	16	2.0GB
		STOCK	/dev/rdsk/hdb1302	159	
		CUSTOMER	/dev/rdsk/hdb1303	147	
		ORDERS	/dev/rdsk/hdb1304	13	
		NEW ORDER	/dev/rdsk/hdb1305	5	
	hdb14	HISTORY	/dev/rdsk/hdb1401	16	2.0GB
		STOCK	/dev/rdsk/hdb1402	159	
		CUSTOMER	/dev/rdsk/hdb1403	147	
		ORDERS	/dev/rdsk/hdb1404	13	
		NEW ORDER	/dev/rdsk/hdb1405	5	
	hdb15	HISTORY	/dev/rdsk/hdb1501	16	2.0GB
		STOCK	/dev/rdsk/hdb1502	159	
		CUSTOMER	/dev/rdsk/hdb1503	147	
		ORDERS	/dev/rdsk/hdb1504	13	
		NEW ORDER	/dev/rdsk/hdb1505	5	
	hdb20	HISTORY	/dev/rdsk/hdb2001	16	2.0GB
		STOCK	/dev/rdsk/hdb2002	159	
		CUSTOMER	/dev/rdsk/hdb2003	147	
		ORDERS	/dev/rdsk/hdb2004	13	
		NEW ORDER	/dev/rdsk/hdb2005	5	
	hdb21	HISTORY	/dev/rdsk/hdb2101	16	2.0GB
		STOCK	/dev/rdsk/hdb2102	159	
		CUSTOMER	/dev/rdsk/hdb2103	147	
		ORDERS	/dev/rdsk/hdb2104	13	
		NEW ORDER	/dev/rdsk/hdb2105	5	
	hdb22	HISTORY	/dev/rdsk/hdb2201	16	2.0GB
		STOCK	/dev/rdsk/hdb2202	159	
		CUSTOMER	/dev/rdsk/hdb2203	147	
		ORDERS	/dev/rdsk/hdb2204	13	
		NEW ORDER	/dev/rdsk/hdb2205	5	
	hdb23	ROLLBACK	/dev/rdsk/hdb2301	50	2.0GB
		TEMP	/dev/rdsk/hdb2302	932	
	hdb24	ORDER LINE INDEX	/dev/rdsk/hdb2401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdb2402	26	
		ORDER INDEX 1	/dev/rdsk/hdb2403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdb2404	63	
	hdb25	ORDER INDEX 2	/dev/rdsk/hdb2501	61	2.0GB

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
WSA-4	hdb30	HISTORY	/dev/rdsk/hdb3001	16	2.0GB
		STOCK	/dev/rdsk/hdb3002	159	
		CUSTOMER	/dev/rdsk/hdb3003	147	
		ORDERS	/dev/rdsk/hdb3004	13	
		NEW ORDER	/dev/rdsk/hdb3005	5	
	hdb31	HISTORY	/dev/rdsk/hdb3101	16	2.0GB
		STOCK	/dev/rdsk/hdb3102	159	
		CUSTOMER	/dev/rdsk/hdb3103	147	
		ORDERS	/dev/rdsk/hdb3104	13	
		NEW ORDER	/dev/rdsk/hdb3105	5	
	hdb32	HISTORY	/dev/rdsk/hdb3201	16	2.0GB
		STOCK	/dev/rdsk/hdb3202	159	
		CUSTOMER	/dev/rdsk/hdb3203	147	
		ORDERS	/dev/rdsk/hdb3204	13	
		NEW ORDER	/dev/rdsk/hdb3205	5	
	hdb33	HISTORY	/dev/rdsk/hdb3301	16	2.0GB
		STOCK	/dev/rdsk/hdb3302	159	
		CUSTOMER	/dev/rdsk/hdb3303	147	
		ORDERS	/dev/rdsk/hdb3304	13	
		NEW ORDER	/dev/rdsk/hdb3305	5	
	hdb34	HISTORY	/dev/rdsk/hdb3401	16	2.0GB
		STOCK	/dev/rdsk/hdb3402	159	
		CUSTOMER	/dev/rdsk/hdb3403	147	
		ORDERS	/dev/rdsk/hdb3404	13	
		NEW ORDER	/dev/rdsk/hdb3405	5	
	hdb35	HISTORY	/dev/rdsk/hdb3501	16	2.0GB
		STOCK	/dev/rdsk/hdb3502	159	
		CUSTOMER	/dev/rdsk/hdb3503	147	
		ORDERS	/dev/rdsk/hdb3504	13	
		NEW ORDER	/dev/rdsk/hdb3505	5	
	hdb40	HISTORY	/dev/rdsk/hdb4001	16	2.0GB
		STOCK	/dev/rdsk/hdb4002	159	
		CUSTOMER	/dev/rdsk/hdb4003	147	
		ORDERS	/dev/rdsk/hdb4004	13	
		NEW ORDER	/dev/rdsk/hdb4005	5	
	hdb41	HISTORY	/dev/rdsk/hdb4101	16	2.0GB
		STOCK	/dev/rdsk/hdb4102	159	
		CUSTOMER	/dev/rdsk/hdb4103	147	
		ORDERS	/dev/rdsk/hdb4104	13	
		NEW ORDER	/dev/rdsk/hdb4105	5	
	hdb42	HISTORY	/dev/rdsk/hdb4201	16	2.0GB
		STOCK	/dev/rdsk/hdb4202	159	
		CUSTOMER	/dev/rdsk/hdb4203	147	
		ORDERS	/dev/rdsk/hdb4204	13	
		NEW ORDER	/dev/rdsk/hdb4205	5	
	hdb43	ROLLBACK	/dev/rdsk/hdb4301	50	2.0GB
		TEMP	/dev/rdsk/hdb4302	932	
	hdb44	ORDER LINE INDEX	/dev/rdsk/hdb4401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdb4402	26	
		ORDER INDEX 1	/dev/rdsk/hdb4403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdb4404	63	

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
	hdb45	ORDER INDEX 2	/dev/rdsk/hdb4501	61	2.0GB
WSA-6	hdc10	HISTORY	/dev/rdsk/hdc1001	16	2.0GB
		STOCK	/dev/rdsk/hdc1002	159	
		CUSTOMER	/dev/rdsk/hdc1003	147	
		ORDERS	/dev/rdsk/hdc1004	13	
		NEW ORDER	/dev/rdsk/hdc1005	5	
	hdc11	HISTORY	/dev/rdsk/hdc1101	16	2.0GB
		STOCK	/dev/rdsk/hdc1102	159	
		CUSTOMER	/dev/rdsk/hdc1103	147	
		ORDERS	/dev/rdsk/hdc1104	13	
		NEW ORDER	/dev/rdsk/hdc1105	5	
	hdc12	HISTORY	/dev/rdsk/hdc1201	16	2.0GB
		STOCK	/dev/rdsk/hdc1202	159	
		CUSTOMER	/dev/rdsk/hdc1203	147	
		ORDERS	/dev/rdsk/hdc1204	13	
		NEW ORDER	/dev/rdsk/hdc1205	5	
	hdc13	HISTORY	/dev/rdsk/hdc1301	16	2.0GB
		STOCK	/dev/rdsk/hdc1302	159	
		CUSTOMER	/dev/rdsk/hdc1303	147	
		ORDERS	/dev/rdsk/hdc1304	13	
		NEW ORDER	/dev/rdsk/hdc1305	5	
	hdc15	HISTORY	/dev/rdsk/hdc1501	16	2.0GB
		STOCK	/dev/rdsk/hdc1502	159	
		CUSTOMER	/dev/rdsk/hdc1503	147	
		ORDERS	/dev/rdsk/hdc1504	13	
		NEW ORDER	/dev/rdsk/hdc1505	5	
	hdc20	HISTORY	/dev/rdsk/hdc2001	16	2.0GB
		STOCK	/dev/rdsk/hdc2002	159	
		CUSTOMER	/dev/rdsk/hdc2003	147	
		ORDERS	/dev/rdsk/hdc2004	13	
		NEW ORDER	/dev/rdsk/hdc2005	5	
	hdc21	HISTORY	/dev/rdsk/hdc2101	16	2.0GB
		STOCK	/dev/rdsk/hdc2102	159	
		CUSTOMER	/dev/rdsk/hdc2103	147	
		ORDERS	/dev/rdsk/hdc2104	13	
		NEW ORDER	/dev/rdsk/hdc2105	5	
	hdc23	ROLLBACK	/dev/rdsk/hdc2301	50	2.0GB
		TEMP	/dev/rdsk/hdc2302	932	
	hdc24	ORDER LINE INDEX	/dev/rdsk/hdc2401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdc2402	26	
		ORDER INDEX 1	/dev/rdsk/hdc2403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdc2404	63	
	hdc25	ORDER INDEX 2	/dev/rdsk/hdc2501	61	2.0GB
WSA-7	hdc30	HISTORY	/dev/rdsk/hdc3001	16	2.0GB
		STOCK	/dev/rdsk/hdc3002	159	
		CUSTOMER	/dev/rdsk/hdc3003	147	
		ORDERS	/dev/rdsk/hdc3004	13	
		NEW ORDER	/dev/rdsk/hdc3005	5	
	hdc31	HISTORY	/dev/rdsk/hdc3101	16	2.0GB
		STOCK	/dev/rdsk/hdc3102	159	
		CUSTOMER	/dev/rdsk/hdc3103	147	

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
		ORDERS	/dev/rdsk/hdc3104	13	
		NEW ORDER	/dev/rdsk/hdc3105	5	
	hdc32	HISTORY	/dev/rdsk/hdc3201	16	2.0GB
		STOCK	/dev/rdsk/hdc3202	159	
		CUSTOMER	/dev/rdsk/hdc3203	147	
		ORDERS	/dev/rdsk/hdc3204	13	
		NEW ORDER	/dev/rdsk/hdc3205	5	
	hdc33	HISTORY	/dev/rdsk/hdc3301	16	2.0GB
		STOCK	/dev/rdsk/hdc3302	159	
		CUSTOMER	/dev/rdsk/hdc3303	147	
		ORDERS	/dev/rdsk/hdc3304	13	
		NEW ORDER	/dev/rdsk/hdc3305	5	
	hdc34	HISTORY	/dev/rdsk/hdc3401	16	2.0GB
		STOCK	/dev/rdsk/hdc3402	159	
		CUSTOMER	/dev/rdsk/hdc3403	147	
		ORDERS	/dev/rdsk/hdc3404	13	
		NEW ORDER	/dev/rdsk/hdc3405	5	
	hdc35	HISTORY	/dev/rdsk/hdc3501	16	2.0GB
		STOCK	/dev/rdsk/hdc3502	159	
		CUSTOMER	/dev/rdsk/hdc3503	147	
		ORDERS	/dev/rdsk/hdc3504	13	
		NEW ORDER	/dev/rdsk/hdc3505	5	
	hdc40	HISTORY	/dev/rdsk/hdc4001	16	2.0GB
		STOCK	/dev/rdsk/hdc4002	159	
		CUSTOMER	/dev/rdsk/hdc4003	147	
		ORDERS	/dev/rdsk/hdc4004	13	
		NEW ORDER	/dev/rdsk/hdc4005	5	
	hdc41	HISTORY	/dev/rdsk/hdc4101	16	2.0GB
		STOCK	/dev/rdsk/hdc4102	159	
		CUSTOMER	/dev/rdsk/hdc4103	147	
		ORDERS	/dev/rdsk/hdc4104	13	
		NEW ORDER	/dev/rdsk/hdc4105	5	
	hdc42	HISTORY	/dev/rdsk/hdc4201	16	2.0GB
		STOCK	/dev/rdsk/hdc4202	159	
		CUSTOMER	/dev/rdsk/hdc4203	147	
		ORDERS	/dev/rdsk/hdc4204	13	
		NEW ORDER	/dev/rdsk/hdc4205	5	
	hdc43	ROLLBACK	/dev/rdsk/hdc4301	50	2.0GB
		TEMP	/dev/rdsk/hdc4302	932	
	hdc44	ORDER LINE INDEX	/dev/rdsk/hdc4401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdc4402	26	
		ORDER INDEX 1	/dev/rdsk/hdc4403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdc4404	63	
	hdc45	ORDER INDEX 2	/dev/rdsk/hdc4501	61	2.0GB
WSA-8	hdc50	HISTORY	/dev/rdsk/hdc5001	16	2.0GB
		STOCK	/dev/rdsk/hdc5002	159	
		CUSTOMER	/dev/rdsk/hdc5003	147	
		ORDERS	/dev/rdsk/hdc5004	13	
		NEW ORDER	/dev/rdsk/hdc5005	5	
	hdc51	HISTORY	/dev/rdsk/hdc5101	16	2.0GB
		STOCK	/dev/rdsk/hdc5102	159	

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
		CUSTOMER	/dev/rdsk/hdc5103	147	
		ORDERS	/dev/rdsk/hdc5104	13	
		NEW ORDER	/dev/rdsk/hdc5105	5	
	hdc52	HISTORY	/dev/rdsk/hdc5201	16	2.0GB
		STOCK	/dev/rdsk/hdc5202	159	
		CUSTOMER	/dev/rdsk/hdc5203	147	
		ORDERS	/dev/rdsk/hdc5204	13	
		NEW ORDER	/dev/rdsk/hdc5205	5	
	hdc53	HISTORY	/dev/rdsk/hdc5301	16	2.0GB
		STOCK	/dev/rdsk/hdc5302	159	
		CUSTOMER	/dev/rdsk/hdc5303	147	
		ORDERS	/dev/rdsk/hdc5304	13	
		NEW ORDER	/dev/rdsk/hdc5305	5	
	hdc54	HISTORY	/dev/rdsk/hdc5401	16	2.0GB
		STOCK	/dev/rdsk/hdc5402	159	
		CUSTOMER	/dev/rdsk/hdc5403	147	
		ORDERS	/dev/rdsk/hdc5404	13	
		NEW ORDER	/dev/rdsk/hdc5405	5	
	hdc60	HISTORY	/dev/rdsk/hdc6001	16	2.0GB
		STOCK	/dev/rdsk/hdc6002	159	
		CUSTOMER	/dev/rdsk/hdc6003	147	
		ORDERS	/dev/rdsk/hdc6004	13	
		NEW ORDER	/dev/rdsk/hdc6005	5	
	hdc61	HISTORY	/dev/rdsk/hdc6101	16	2.0GB
		STOCK	/dev/rdsk/hdc6102	159	
		CUSTOMER	/dev/rdsk/hdc6103	147	
		ORDERS	/dev/rdsk/hdc6104	13	
		NEW ORDER	/dev/rdsk/hdc6105	5	
	hdc62	HISTORY	/dev/rdsk/hdc6201	16	2.0GB
		STOCK	/dev/rdsk/hdc6202	159	
		CUSTOMER	/dev/rdsk/hdc6203	147	
		ORDERS	/dev/rdsk/hdc6204	13	
		NEW ORDER	/dev/rdsk/hdc6205	5	
	hdc63	ROLLBACK	/dev/rdsk/hdc6301	50	2.0GB
		TEMP	/dev/rdsk/hdc6302	932	
	hdc64	ORDER LINE INDEX	/dev/rdsk/hdc6401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdc6402	26	
		ORDER INDEX 1	/dev/rdsk/hdc6403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdc6404	63	
	hdc65	ORDER INDEX 2	/dev/rdsk/hdc6501	61	2.0GB
WSA-9	hdd10	HISTORY	/dev/rdsk/hdd1001	16	2.0GB
		STOCK	/dev/rdsk/hdd1002	159	
		CUSTOMER	/dev/rdsk/hdd1003	147	
		ORDERS	/dev/rdsk/hdd1004	13	
		NEW ORDER	/dev/rdsk/hdd1005	5	
	hdd11	HISTORY	/dev/rdsk/hdd1101	16	2.0GB
		STOCK	/dev/rdsk/hdd1102	159	
		CUSTOMER	/dev/rdsk/hdd1103	147	
		ORDERS	/dev/rdsk/hdd1104	13	
		NEW ORDER	/dev/rdsk/hdd1105	5	
	hdd12	HISTORY	/dev/rdsk/hdd1201	16	2.0GB

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
		STOCK	/dev/rdsk/hdd1202	159	
		CUSTOMER	/dev/rdsk/hdd1203	147	
		ORDERS	/dev/rdsk/hdd1204	13	
		NEW ORDER	/dev/rdsk/hdd1205	5	
	hdd13	HISTORY	/dev/rdsk/hdd1301	16	2.0GB
		STOCK	/dev/rdsk/hdd1302	159	
		CUSTOMER	/dev/rdsk/hdd1303	147	
		ORDERS	/dev/rdsk/hdd1304	13	
		NEW ORDER	/dev/rdsk/hdd1305	5	
	hdd14	HISTORY	/dev/rdsk/hdd1401	16	2.0GB
		STOCK	/dev/rdsk/hdd1402	159	
		CUSTOMER	/dev/rdsk/hdd1403	147	
		ORDERS	/dev/rdsk/hdd1404	13	
		NEW ORDER	/dev/rdsk/hdd1405	5	
	hdd15	HISTORY	/dev/rdsk/hdd1501	16	2.0GB
		STOCK	/dev/rdsk/hdd1502	159	
		CUSTOMER	/dev/rdsk/hdd1503	147	
		ORDERS	/dev/rdsk/hdd1504	13	
		NEW ORDER	/dev/rdsk/hdb1505	5	
	hdd20	HISTORY	/dev/rdsk/hdd2001	16	2.0GB
		STOCK	/dev/rdsk/hdd2002	159	
		CUSTOMER	/dev/rdsk/hdd2003	147	
		ORDERS	/dev/rdsk/hdd2004	13	
		NEW ORDER	/dev/rdsk/hdd2005	5	
	hdd21	HISTORY	/dev/rdsk/hdd2101	16	2.0GB
		STOCK	/dev/rdsk/hdd2102	159	
		CUSTOMER	/dev/rdsk/hdd2103	147	
		ORDERS	/dev/rdsk/hdd2104	13	
		NEW ORDER	/dev/rdsk/hdd2105	5	
	hdd22	HISTORY	/dev/rdsk/hdd2201	16	2.0GB
		STOCK	/dev/rdsk/hdd2202	159	
		CUSTOMER	/dev/rdsk/hdd2203	147	
		ORDERS	/dev/rdsk/hdd2204	13	
		NEW ORDER	/dev/rdsk/hdd2205	5	
	hdd23	ROLLBACK	/dev/rdsk/hdd2301	50	2.0GB
		TEMP	/dev/rdsk/hdd2302	932	
	hdd24	ORDER LINE INDEX	/dev/rdsk/hdd2401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdd2402	26	
		ORDER INDEX 1	/dev/rdsk/hdd2403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdd2404	63	
	hdd25	ORDER INDEX 2	/dev/rdsk/hdd2501	61	2.0GB
WSA-10	hdd30	HISTORY	/dev/rdsk/hdd3001	16	2.0GB
		STOCK	/dev/rdsk/hdd3002	159	
		CUSTOMER	/dev/rdsk/hdd3003	147	
		ORDERS	/dev/rdsk/hdd3004	13	
		NEW ORDER	/dev/rdsk/hdd3005	5	
	hdd31	HISTORY	/dev/rdsk/hdd3101	16	2.0GB
		STOCK	/dev/rdsk/hdd3102	159	
		CUSTOMER	/dev/rdsk/hdd3103	147	
		ORDERS	/dev/rdsk/hdd3104	13	
		NEW ORDER	/dev/rdsk/hdd3105	5	

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
	hdd32	HISTORY	/dev/rdsk/hdd3201	16	2.0GB
		STOCK	/dev/rdsk/hdd3202	159	
		CUSTOMER	/dev/rdsk/hdd3203	147	
		ORDERS	/dev/rdsk/hdd3204	13	
		NEW ORDER	/dev/rdsk/hdd3205	5	
	hdd33	HISTORY	/dev/rdsk/hdd3301	16	2.0GB
		STOCK	/dev/rdsk/hdd3302	159	
		CUSTOMER	/dev/rdsk/hdd3303	147	
		ORDERS	/dev/rdsk/hdd3304	13	
		NEW ORDER	/dev/rdsk/hdd3305	5	
	hdd34	HISTORY	/dev/rdsk/hdd3401	16	2.0GB
		STOCK	/dev/rdsk/hdd3402	159	
		CUSTOMER	/dev/rdsk/hdd3403	147	
		ORDERS	/dev/rdsk/hdd3404	13	
		NEW ORDER	/dev/rdsk/hdd3405	5	
	hdd35	HISTORY	/dev/rdsk/hdd3501	16	2.0GB
		STOCK	/dev/rdsk/hdd3502	159	
		CUSTOMER	/dev/rdsk/hdd3503	147	
		ORDERS	/dev/rdsk/hdd3504	13	
		NEW ORDER	/dev/rdsk/hdd3505	5	
	hdd40	HISTORY	/dev/rdsk/hdd4001	16	2.0GB
		STOCK	/dev/rdsk/hdd4002	159	
		CUSTOMER	/dev/rdsk/hdd4003	147	
		ORDERS	/dev/rdsk/hdd4004	13	
		NEW ORDER	/dev/rdsk/hdd4005	5	
	hdd41	HISTORY	/dev/rdsk/hdd4101	16	2.0GB
		STOCK	/dev/rdsk/hdd4102	159	
		CUSTOMER	/dev/rdsk/hdd4103	147	
		ORDERS	/dev/rdsk/hdd4104	13	
		NEW ORDER	/dev/rdsk/hdd4105	5	
	hdd42	HISTORY	/dev/rdsk/hdd4201	16	2.0GB
		STOCK	/dev/rdsk/hdd4202	159	
		CUSTOMER	/dev/rdsk/hdd4203	147	
		ORDERS	/dev/rdsk/hdd4204	13	
		NEW ORDER	/dev/rdsk/hdd4205	5	
	hdd43	ROLLBACK	/dev/rdsk/hdd4301	50	2.0GB
		TEMP	/dev/rdsk/hdd4302	932	
	hdd44	ORDER LINE INDEX	/dev/rdsk/hdd4401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdd4402	26	
		ORDER INDEX 1	/dev/rdsk/hdd4403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdd4404	63	
	hdd45	ORDER INDEX 2	/dev/rdsk/hdd4501	61	2.0GB
WSA-12	hde10	HISTORY	/dev/rdsk/hde1001	16	2.0GB
		STOCK	/dev/rdsk/hde1002	159	
		CUSTOMER	/dev/rdsk/hde1003	147	
		ORDERS	/dev/rdsk/hde1004	13	
		NEW ORDER	/dev/rdsk/hde1005	5	
	hde11	HISTORY	/dev/rdsk/hde1101	16	2.0GB
		STOCK	/dev/rdsk/hde1102	159	
		CUSTOMER	/dev/rdsk/hde1103	147	
		ORDERS	/dev/rdsk/hde1104	13	

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
		NEW ORDER	/dev/rdsk/hde1105	5	
	hde12	HISTORY	/dev/rdsk/hde1201	16	2.0GB
		STOCK	/dev/rdsk/hde1202	159	
		CUSTOMER	/dev/rdsk/hde1203	147	
		ORDERS	/dev/rdsk/hde1204	13	
		NEW ORDER	/dev/rdsk/hde1205	5	
	hde13	HISTORY	/dev/rdsk/hde1301	16	2.0GB
		STOCK	/dev/rdsk/hde1302	159	
		CUSTOMER	/dev/rdsk/hde1303	147	
		ORDERS	/dev/rdsk/hde1304	13	
		NEW ORDER	/dev/rdsk/hde1305	5	
	hde14	HISTORY	/dev/rdsk/hde1401	16	2.0GB
		STOCK	/dev/rdsk/hde1402	159	
		CUSTOMER	/dev/rdsk/hde1403	147	
		ORDERS	/dev/rdsk/hde1404	13	
		NEW ORDER	/dev/rdsk/hde1405	5	
	hde15	HISTORY	/dev/rdsk/hde1501	16	2.0GB
		STOCK	/dev/rdsk/hde1502	159	
		CUSTOMER	/dev/rdsk/hde1503	147	
		ORDERS	/dev/rdsk/hde1504	13	
		NEW ORDER	/dev/rdsk/hde1505	5	
	hde20	HISTORY	/dev/rdsk/hde2001	16	2.0GB
		STOCK	/dev/rdsk/hde2002	159	
		CUSTOMER	/dev/rdsk/hde2003	147	
		ORDERS	/dev/rdsk/hde2004	13	
		NEW ORDER	/dev/rdsk/hde2005	5	
	hde21	HISTORY	/dev/rdsk/hde2101	16	2.0GB
		STOCK	/dev/rdsk/hde2102	159	
		CUSTOMER	/dev/rdsk/hde2103	147	
		ORDERS	/dev/rdsk/hde2104	13	
		NEW ORDER	/dev/rdsk/hde2105	5	
	hde22	HISTORY	/dev/rdsk/hde2201	16	2.0GB
		STOCK	/dev/rdsk/hde2202	159	
		CUSTOMER	/dev/rdsk/hde2203	147	
		ORDERS	/dev/rdsk/hde2204	13	
		NEW ORDER	/dev/rdsk/hde2205	5	
	hde23	ROLLBACK	/dev/rdsk/hde2301	50	2.0GB
		TEMP	/dev/rdsk/hde2302	932	
	hde24	ORDER LINE INDEX	/dev/rdsk/hde2401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hde2402	26	
		ORDER INDEX 1	/dev/rdsk/hde2403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hde2404	63	
	hde25	ORDER INDEX 2	/dev/rdsk/hde2501	61	2.0GB
WSA-13	hde30	ORDER LINE	/dev/rdsk/hde3001	932	2.0GB
	hde31	ORDER LINE	/dev/rdsk/hde3101	932	2.0GB
	hde32	ORDER LINE	/dev/rdsk/hde3201	932	2.0GB
	hde33	ORDER LINE	/dev/rdsk/hde3301	932	2.0GB
	hde34	ORDER LINE	/dev/rdsk/hde3401	932	2.0GB
	hde35	ORDER LINE	/dev/rdsk/hde3501	932	2.0GB
	hde40	ORDER LINE	/dev/rdsk/hde4001	932	2.0GB
	hde41	ORDER LINE	/dev/rdsk/hde4101	932	2.0GB

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
	hde42	ORDER LINE	/dev/rdsk/hde4201	932	2.0GB
	hde43	ORDER LINE	/dev/rdsk/hde4301	932	2.0GB
	hde45	ORDER LINE	/dev/rdsk/hde4501	932	2.0GB
WSA-14	hde50	ORDER LINE	/dev/rdsk/hde5001	932	2.0GB
	hde51	ORDER LINE	/dev/rdsk/hde5101	932	2.0GB
	hde52	ORDER LINE	/dev/rdsk/hde5201	932	2.0GB
	hde53	ORDER LINE	/dev/rdsk/hde5301	932	2.0GB
	hde54	ORDER LINE	/dev/rdsk/hde5401	932	2.0GB
	hde55	ORDER LINE	/dev/rdsk/hde5501	932	2.0GB
	hde60	ORDER LINE	/dev/rdsk/hde6001	932	2.0GB
	hde61	ORDER LINE	/dev/rdsk/hde6101	932	2.0GB
WSA-15	hdf12	STOCK	/dev/rdsk/hdf1201	159	2.0GB
	hdf13	STOCK	/dev/rdsk/hdf1301	159	2.0GB
	hdf14	STOCK	/dev/rdsk/hdf1401	159	2.0GB
	hdf15	STOCK	/dev/rdsk/hdf1501	159	2.0GB
	hdf20	STOCK	/dev/rdsk/hdf2001	159	2.0GB
	hdf22	ORDER LINE INDEX	/dev/rdsk/hdf2201	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdf2202	26	
		ORDER INDEX 1	/dev/rdsk/hdf2203	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdf2204	63	
	hdf24	STOCK INDEX	/dev/rdsk/hdf2401	680	2.0GB
	hdf25	ORACLE-SYSTEM	/dev/rdsk/hdf2501	600	2.0GB
		WAREHOUSE	/dev/rdsk/hdf2502	24	
		+DISTRICT	/dev/rdsk/hdf2502		
		ITEM	/dev/rdsk/hdf2503	20	
WSA-16	hdf32	STOCK	/dev/rdsk/hdf3201	159	2.0GB
	hdf33	STOCK	/dev/rdsk/hdf3301	159	2.0GB
	hdf34	STOCK	/dev/rdsk/hdf3401	159	2.0GB
	hdf35	STOCK	/dev/rdsk/hdf3501	159	2.0GB
	hdf40	STOCK	/dev/rdsk/hdf4001	159	2.0GB
	hdf41	ORDER LINE INDEX	/dev/rdsk/hdf4101	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdf4102	26	
		ORDER INDEX 1	/dev/rdsk/hdf4103	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdf4104	63	
	hdf42	ORDER LINE INDEX	/dev/rdsk/hdf4201	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdf4202	26	
		ORDER INDEX 1	/dev/rdsk/hdf4203	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdf4204	63	
	hdf43	LTEMP	/dev/rdsk/hdf4301	1000	2.0GB
	hdf44	STOCK INDEX	/dev/rdsk/hdf4401	680	2.0GB
	hdf45	CUSTOMER INDEX 1	/dev/rdsk/hdf4501	795	
WSA-18	hda61	HISTORY	/dev/rdsk/hda6101	16	2.0GB
		STOCK	/dev/rdsk/hda6102	159	
		CUSTOMER	/dev/rdsk/hda6103	147	
		ORDERS	/dev/rdsk/hda6104	13	
		NEW ORDER	/dev/rdsk/hda6105	5	
	hda64	ORDER LINE INDEX	/dev/rdsk/hda6401	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hda6402	26	
		ORDER INDEX 1	/dev/rdsk/hda6403	57	
		CUSTOMER INDEX 2	/dev/rdsk/hda6404	63	
	hdb12	HISTORY	/dev/rdsk/hdb1201	16	2.0GB

SCSI adapter	Device	TABLE NAME	FILE NAME	SIZE (Mbytes)	DISK CAPACITY
		STOCK	/dev/rdsk/hdb1202	159	
		CUSTOMER	/dev/rdsk/hdb1203	147	
		ORDERS	/dev/rdsk/hdb1204	13	
		NEW ORDER	/dev/rdsk/hdb1205	5	
	hdc14	HISTORY	/dev/rdsk/hdc1401	16	2.0GB
		STOCK	/dev/rdsk/hdc1402	159	
		CUSTOMER	/dev/rdsk/hdc1403	147	
		ORDERS	/dev/rdsk/hdc1404	13	
		NEW ORDER	/dev/rdsk/hdc1405	5	
	hdc22	HISTORY	/dev/rdsk/hdc2201	16	2.0GB
		STOCK	/dev/rdsk/hdc2202	159	
		CUSTOMER	/dev/rdsk/hdc2203	147	
		ORDERS	/dev/rdsk/hdc2204	13	
		NEW ORDER	/dev/rdsk/hdc2205	5	
	hdc55	HISTORY	/dev/rdsk/hdc5501	16	2.0GB
		STOCK	/dev/rdsk/hdc5502	159	
		CUSTOMER	/dev/rdsk/hdc5503	147	
		ORDERS	/dev/rdsk/hdc5504	13	
		NEW ORDER	/dev/rdsk/hdc5505	5	
	hde44	ORDER LINE	/dev/rdsk/hde4401	932	2.0GB
	hdf21	ORDER LINE INDEX	/dev/rdsk/hdf2101	576	2.0GB
		NEW ORDER INDEX	/dev/rdsk/hdf2102	26	
		ORDER INDEX 1	/dev/rdsk/hdf2103	57	
		CUSTOMER INDEX 2	/dev/rdsk/hdf2104	63	
	hdf10	LOG/GRP1	/dev/rdsk/hdf1001	2040	4.0GB
	hdf11	LOG MIRROR/GRP2	/dev/rdsk/hdf1101	2040	4.0GB
	hdf30	LOG/GRP2	/dev/rdsk/hdf3001	2040	4.0GB
	hdf31	LOG MIRROR/GRP1	/dev/rdsk/hdf3101	2040	4.0GB

1.3. Insert and Delete Operations

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

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

1.4. Partitioning

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

Partitioning was not used on any table in this benchmark.

1.5. Replication, Duplication or Additions

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

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

Clause 2 Related Items

2.1 Random Number Generation

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

The seeds for each user were generated using the process id. Each RTE machine was given a number incremented by 10,000. The process id was appended to this number to ensure uniqueness across all RTE machines. These seeds were printed to a file and verified by the auditor to be unique.

2.2 Input/Output Screen Layout

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

All screen layouts followed the specifications exactly.

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).

The terminal attributes were verified by the auditor manually exercising each specification during the onsite audit portion of this benchmark.

2.4 Presentation Manager or Intelligent Terminal

Any usage of presentation managers or intelligent terminals must be explained

The PC's in the priced configuration come with Microsoft Windows 95. Presentation is handled by the terminal emulator found in Windows software.

2.5 Transaction Statistics

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

Table 2. 1 Transaction Statistics

Statistic		Value
New Order	Home warehouse order lines	98.99%
	Remote warehouse order lines	1.01%
	Rolled back transactions	0.99%
	Average items per order	10.00
Payment	Home warehouse	84.92%
	Remote warehouse	15.08%
	Accessed by last name	60.25%
Order Status	Accessed by last name	60.96%
Delivery	Skipped transactions	none
Transaction Mix	New Order	44.62%
	Payment	43.29%
	Order status	4.05%
	Delivery	4.00%
	Stock level	4.04%

2.6 Queueing Mechanism

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

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

Clause 3 Related Items

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 SUT must support during the execution of the benchmark. Those properties are Atomicity, Consistency, Isolation and Durability (ACID).

This section defines each of those 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 the database transactions are atomic; the system will either perform all individual operations on the data or will assure that no partially completed operations leave any effects on the data.

3.2.1 Completed Transactions

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

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

3.2.2 Aborted Transactions

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

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

3.3 Consistency

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

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

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

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

A performance run was completed including a full 30 minutes of steady state and checkpoints.

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

3.4 Isolation

Isolation can be defined in terms of phenomena that can occur during the execution of concurrent transactions. These phenomena are P0 ("Dirty Write"), P1 ("Dirty Read"), P2 ("non-repeatable Read"), and P3 ("Phantom"). The table in Clause 3.4.1 of the TPC-C specifications defines the isolation requirements which must be met by the TPC-C transactions. 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 nine required tests to be performed to demonstrate that the required levels of transaction isolation are met. These tests, described in Clauses 3.4.2.1 - 3.4.2.9, were all performed and verified as required.

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

For Isolation test seven, case A was followed.

3.5 Durability

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

3.5.1 Durable Media Failure

3.5.1.1 Loss of Data

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

1. The database was backed up to extra disks.
2. The total number of orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving the beginning count.
3. The RTE was started with 120 users.
4. The test was allowed to run for a minimum of 5 minutes.
5. One of the data disks was powered off by removing it from the cabinet.
6. The RTE was shut down.
7. The data disk was returned to the cabinet, powered back up and reformatted to simulate complete loss of data.
8. Data from the backup disk was copied to it.
9. Oracle was restarted and used the transaction log to roll forward and recover the data from committed transactions.
10. Step 2 was repeated and the difference between the first and second counts was noted.
11. The success file was used to determine the number of NEW-ORDERS successfully returned to the RTE.
12. The counts in step 9 and 10 were compared and the results verified that all committed transactions had been successfully recovered.
13. Data from the success file was used to query the database to demonstrate successful transactions had corresponding rows in the ORDER table, and rolled back transactions did not.

3.5.1.2 Loss of Log

To demonstrate recovery from a permanent failure of durable medium containing Oracle recovery log data the following steps were executed:

1. The total number of orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving the beginning count.
2. The RTE was started with 120 users.
3. The test was allowed to run for a minimum of 6 minutes.
4. One log disk was powered off by removing it from the cabinet.
5. Since the disk was mirrored, processing was not interrupted .
6. The RTE was shut down.
7. The log disk was returned to the cabinet and began normal recovery by synchronizing with its mirror image.
8. Step 2 was repeated and the difference between the first and second counts was noted.
9. The success file was used to determine the number of NEW-ORDERS successfully returned to the RTE.
10. The counts in step 9 and 10 were compared and the results verified that all committed transactions had been successfully recovered.
11. Samples were taken from the RTE files and used to query the database to demonstrate successful transactions had corresponding rows in the ORDER table.

3.5.2 Instantaneous Interruption and Loss of Memory

Because loss of power erases the contents of memory, the instantaneous interruption and the loss of memory tests were combined into a single test. This test was executed on a fully scaled database of 400 warehouses under a full load of 4,000 users. The following steps were executed:

1. The total number of orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving the beginning count.
2. The RTE was started with 4,000 users.
3. The test was allowed to run for a minimum of 5 minutes.
4. The primary power to the processor was shutdown.
5. The RTE was shutdown.
6. Power was restored and the system performed an automatic recovery.
7. Oracle was restarted and performed an automatic recovery .
8. Step 2 was repeated and the difference between the first and second counts was noted.
9. The success file was used to determine the number of NEW-ORDERS successfully returned to the RTE.
10. The counts in step 9 and 10 were compared and the results verified that all committed transactions had been successfully recovered.
11. Data from the success file was used to query the database to demonstrate successful transactions had corresponding rows in the ORDER table, and rolled back transactions did not.

Clause 4 Related Items

4.1 Initial Cardinality of Tables

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

Table 4.1 Number of Rows for Server

Table	Occurrences
Warehouse	400
District	4,000
Customer	12,000,000
History	12,000,000
Order	12,000,000
New Order	3,600,000
Order Line	119,992,912
Stock	40,000,000
Item	100,000

4.2 Database Layout

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

Section 1.2 of this report details the distribution of database tables across all disks. The code that creates the tables is included in Appendix E.

4.3 Type of Database

A statement must be provided that describes:

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

Oracle V7.3 is a relational DBMS.

The interface used was Oracle V7.3 stored procedures accessed using the Oracle Call Interface (OCI) embedded in C code.

4.4 Database Mapping

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

The database was neither partitioned nor replicated.

4.5 180 Day Space

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

To calculate the space required to sustain the database log for 8 hours of growth at steady state the following steps were followed:

- The size of the redo log was queried from the Oracle catalog.
- Transactions were started and checkpoints were initiated automatically every 5 minutes.
- The number of new orders in the period between checkpoints was determined.
- The increase in size to the redo logs was divided by the number of transactions, giving bytes used per new order.
- This amount was multiplied by the reported tpm rate times 480 minutes, giving total space needed for 8 hours..
- This required space was mirrored.

For the dynamic tables the following steps were followed:

1. The database was queried for the size of the dynamic tables.
2. The sum of D-NEXT-O-ID was queried from the DISTRICT table.
3. A full performance run was executed.

4. Steps 1 & 2 were repeated.
5. The change in the size of the dynamic tables was divided by the number of new orders in the run giving growth per new order.
6. The number in the previous step was multiplied by the reported tpm rate times 480 minutes.
7. The numbers in steps 1 & 5 were added giving space needed for 8 hours.
8. The space allocated was verified to be larger than the space needed.

The 180 day space requirement is shown in Appendix F.

Clause 5 Related Items

5.1 Throughput

Measured tpmC must be reported.

Measured tpmC 4,718.73 tpmC
Price per tpmC ¥101,988

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 Times

Type	Average	Maximum	90th %
New-Order	1.54	16.11	2.99
Payment	1.10	16.11	2.47
Order-Status	1.17	14.64	2.54
Interactive Delivery	0.13	2.45	0.25
Deferred Delivery	1.44	16.57	2.95
Stock-Level	1.03	15.53	2.38
Menu	0.17	29.52	0.27

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

Type	Minimum	Average	Maximum
New-Order	18.02	18.21	18.68
Payment	3.01	3.07	3.46
Order-Status	2.01	2.06	2.40
Interactive Delivery	2.01	2.06	2.54
Stock-Level	2.01	2.06	2.37

Table 5.3 Think Times

Type	Minimum	Average	Maximum
New-Order	0.00	12.17	120.83
Payment	0.00	12.25	121.33
Order-Status	0.01	10.14	81.29
Interactive Delivery	0.00	5.06	44.37
Stock-Level	0.00	5.04	48.83

5.4 Response Time Frequency Distribution Curves and Other Graphs

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

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

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

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

Figure 5.1: New Order Response Time Distribution

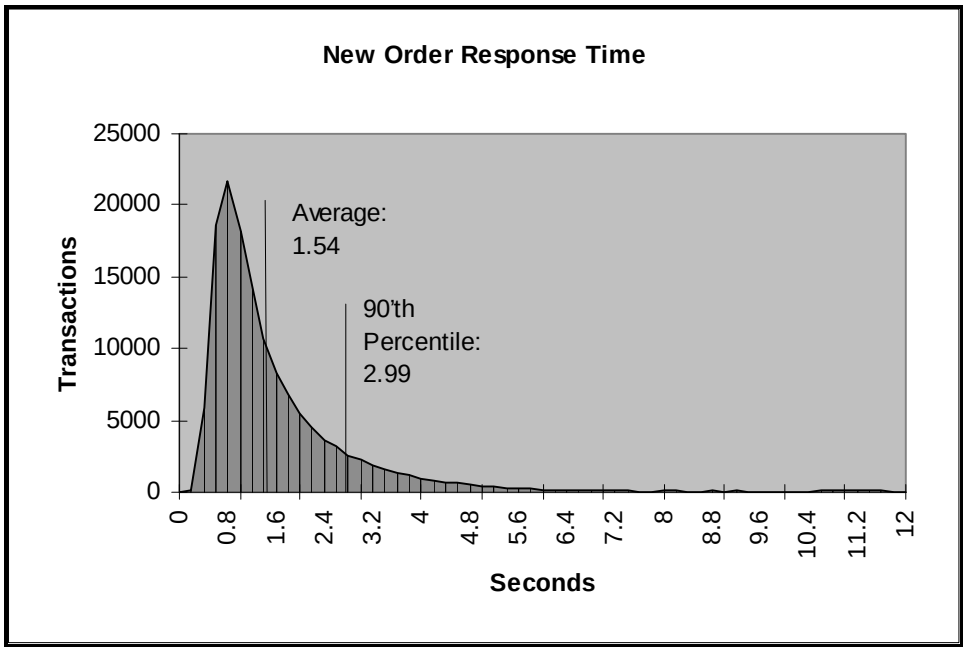


Figure 5.2: Payment Response Time Distribution

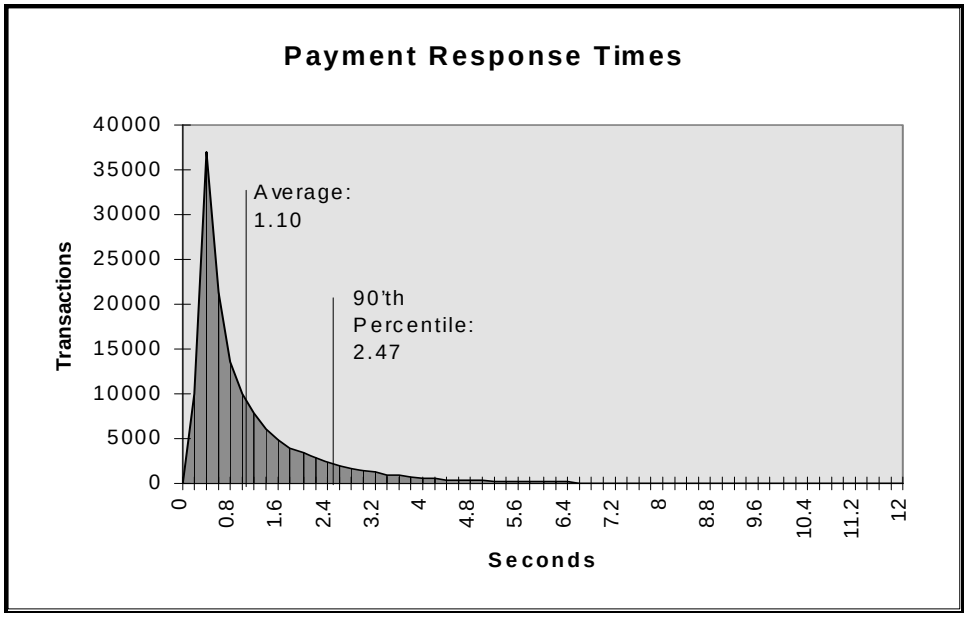


Figure 5.3: Order Status Response Time Distribution

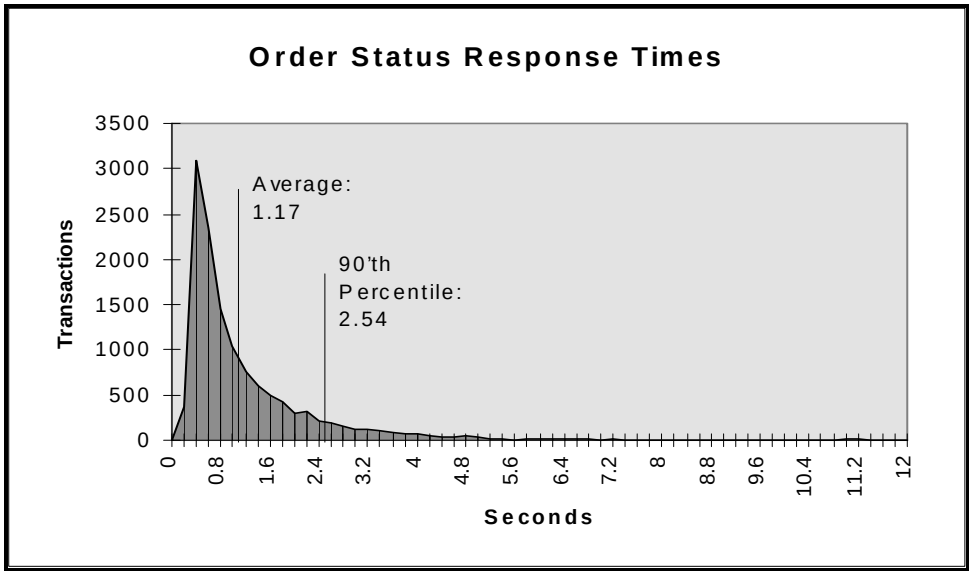


Figure 5.4: Delivery Response Time Distribution

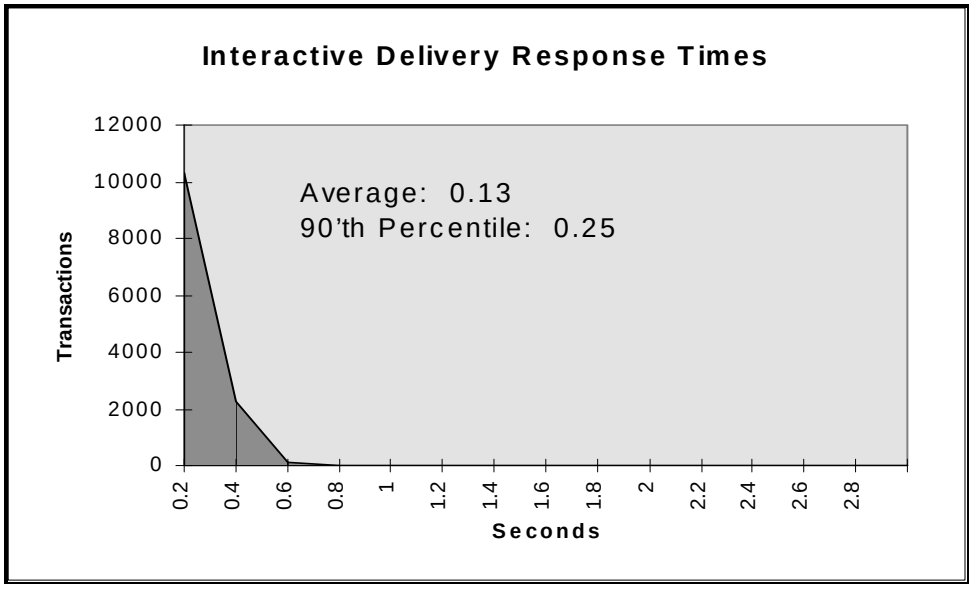


Figure 5.5: Stock Level Response Time Distribution

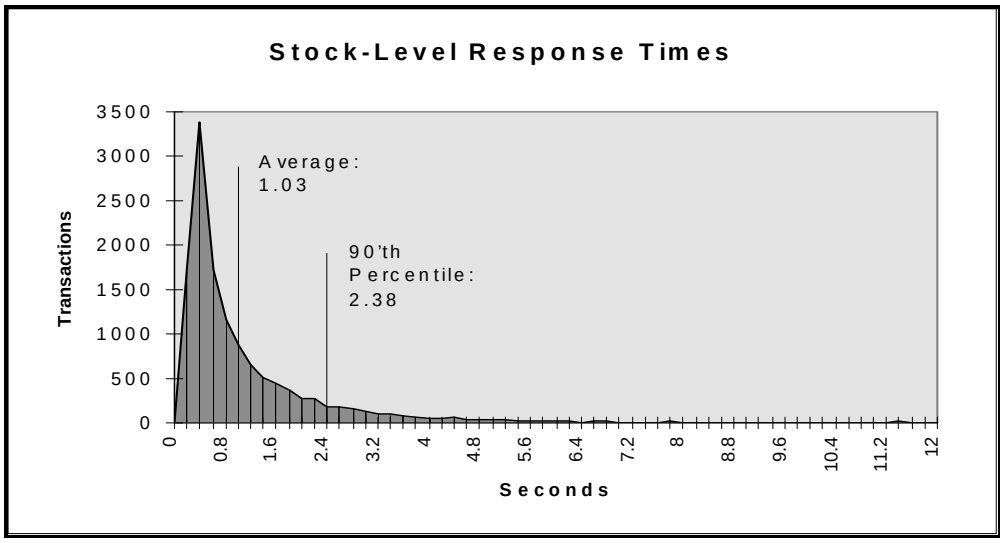


Figure 5.6: New Order Think Time Frequency Distribution

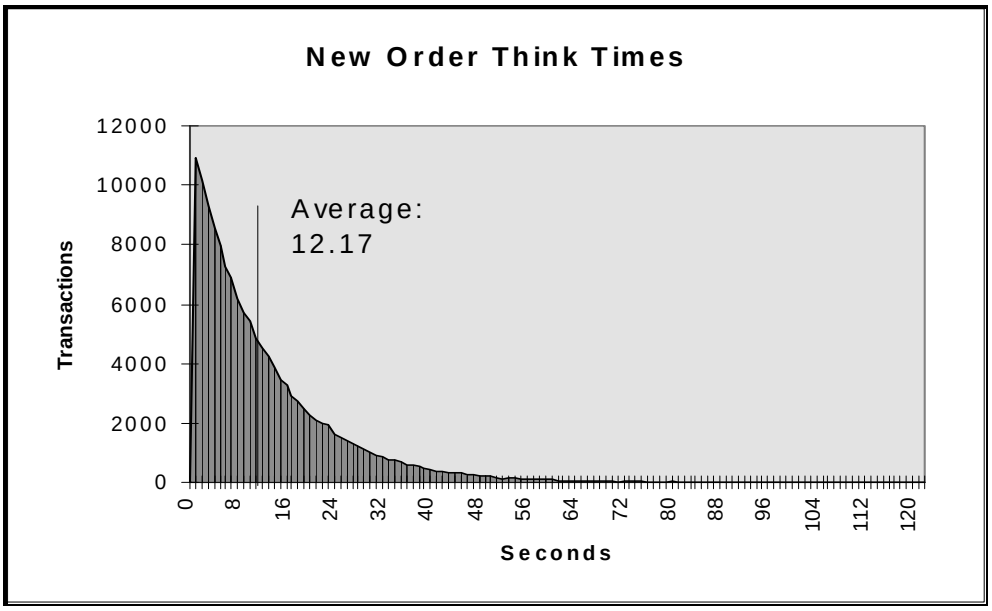


Figure 5.7: Response time versus Throughput

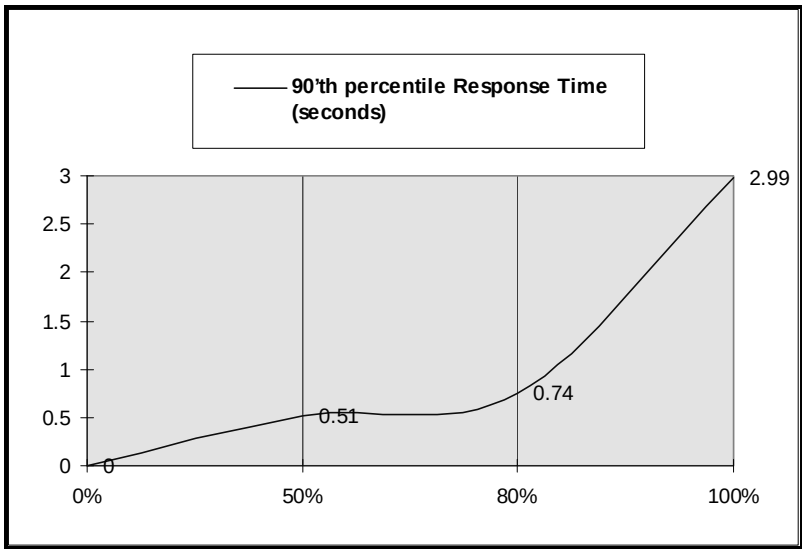
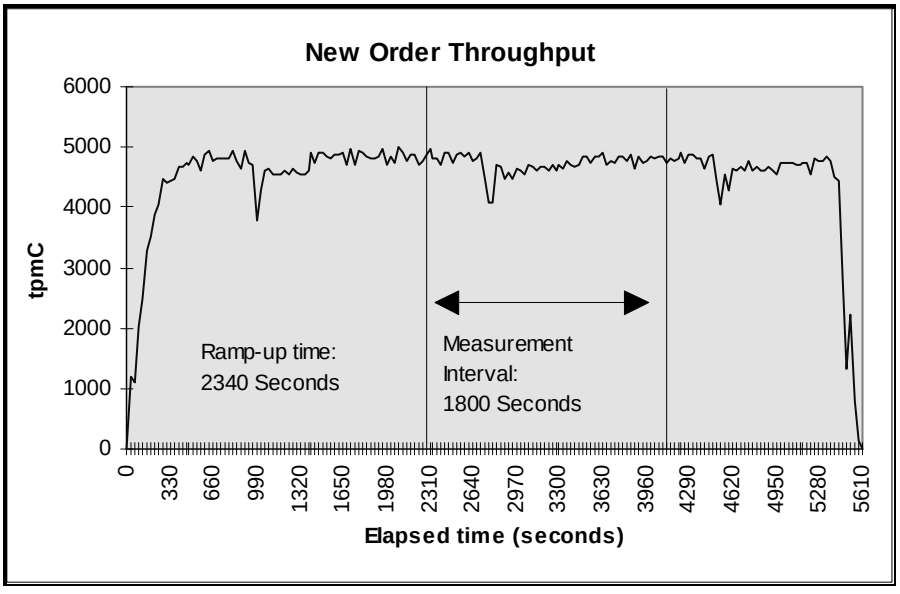


Figure 5.8: New Order Sustained Throughput



5.5 Steady State Determination

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

Steady state was determined by examining data reported for each 30-second interval over the duration of the measured run. Steady state was further confirmed by the throughput data collected during the run and graphed in Figure 5.8.

5.6 Work Performed During Steady State

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

The Oracle logical log is mirrored. 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 checkpoints to an alert file on the server. The scripts included a wait time between each checkpoint equal to the measurement interval, which was 30 minutes. The checkpoint script was started manually after the RTE had all users logged in and sending transactions. At each checkpoint, Oracle7 wrote to disk all buffer pages that had been updated but not yet physically written to disk. The positioning of the checkpoint was verified to be clear of the guard zones and depicted on the graph in Figure 5.8.

For the priced system, the logical log space for an 8-hour period is priced.

Serializable Transactions:

Oracle7 supports serializable transaction isolation in full compliance with the SQL92 and TPC-C requirements. This is implemented by extending multiple concurrency control mechanisms 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, providing a high degree of concurrency and consistency. While this mode does prevent reading dirty data, Oracle's default isolation level also permits a transaction that issues a query twice to see non-repeatable reads and phantoms, as defined in SQL92 and TPC-C.

Beginning with Oracle7 release 7.3, a transaction may request a higher 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 transactions 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 the database, established at the beginning of the transaction.

To ensure proper isolation, a serializable transaction cannot modify the rows that that were changed by other transactions after the beginning of a serializable transaction, or an update (or delete) statement will fail with error ORA_08177: "cannot serialize access" and the statement will rollback.

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

5.7 Reproducibility

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

The measurement procedure was repeated and the throughput verified to be within 2% of the reported measurement.

5.8 Measurement Period Duration

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

The reported measured interval was exactly 30 minutes long.

5.9 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 the a weighted distribution to control the transaction mix, and could not be adjusted during the run.

5.10 Transaction Statistics

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

Table 5.1.: Transaction Statistics

Statistics		Value
Transaction Mix	New Order	44.62%
	Payment	43.29%
	Order status	4.05%
	Delivery	4.00%
	Stock level	4.04%
New Order	Home warehouse order lines	98.99%
	Remote warehouse order lines	1.01%
	Rolled back transactions	0.99%
	Average items per order	10.00
Payment	Home warehouse	84.92%
	Remote warehouse	15.08%
	Accessed by last name	60.25%
Order Status	Accessed by last name	60.96%
Delivery	Skipped transactions	0

5.11 Checkpoint Count and Location

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

One checkpoint was recorded before the measured window opened and another checkpoint was started 483 seconds inside the measured window. Both checkpoints were clear of the guard zone. Checkpoints were started exactly 30 minutes apart.

Clause 6 Related Items

6.1 RTE Descriptions

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

The RTE used was developed at Fujitsu Limited and is proprietary. It consists of an RTE management process as shown in Appendix C, which forks off the individual RTE processes and controls the run. After the run completes, a separate report generator program collects all the log files and generates the final statistics of a run.

Inputs to the RTE include the names of the RTE machine to run, client machines to attach to, the database scale, the ramp-up, measurement and ramp-down times. These come from the configuration script file for the RTE management process.

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 the priced system. The results of the test described in Clause 6.6.3.4 must be disclosed.

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

6.3 Functional Diagrams

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

The driver system performed the data generation and input functions of the display device. It also captured the input and output data and timestamps for post-processing of the reported metrics. No other functionality was included on the driver system

The abstract at the beginning of this report contains detailed diagrams of both the benchmark configuration and the priced configuration, including the driver system.

6.4 Networks

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

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

Ten ethernet 10 MBPS LAN's were used between the emulated users and the client machines. One ethernet 100 MBPS LAN was used between the clients and the server. The abstract at the beginning of this report contains detailed diagrams of the configuration.

6.5 Operator Intervention

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

This configuration does not require any operator intervention to sustain eight hours of the reported throughput, other than beginning the checkpointing process.

Clause 7 Related Items

7.1 System Pricing

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

The total 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 price list is included in the abstract at the beginning of this report .

7.2 Availability, Throughput, and Price Performance

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

All hardware components and software will be available no later than July, 1997.

7.3 Throughput and Price Performance

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.

Maximum Qualified Throughput:	4,718.73 tpmC
Price per tpmC	¥101,988
Available	July, 1997

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

This system is being priced for Japan.

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.*

Oracle support pricing is based on support from Oracle in Japan.

Clause 9 Related Items

9.1 Auditor's Report

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

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

Performance Metrics, Inc.
2229 Benita Dr. Suite 101
Rancho Cordova, CA
(phone) 916/635-2822
(fax) 916/858-0109

9.2 Availability of the Full Disclosure Report

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

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

Transaction Processing Performance Council
c/o Shanley Public Relations
777 North First Street, Suite 6000
San Jose, CA 95112-6311
408/295-8894

Appendix A: Client Source Code

```
****************************************
        Tc.c
****************************************

/*
        Tc.c : Main module for TPC-C client
program

        Version 1.00    1996/12/27
        Version 1.01    1996/12/28
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <time.h>
#include <sys/times.h>
#include <sys/time.h>
#include <sys/param.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <math.h>
#include <stdarg.h>
#include <unistd.h>
#include <signal.h>

#ifdef  __linux__
# include <ncurses/curses.h>
#else
# include <curses.h>
#endif

#include "cwalib.h"
#include "ui.h"
#include "frame.h"
#include "tpcc_info.h"
#include "tpcc.h"

#ifndef SCRTEST
# include "atmi.h"
#endif

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

#define SERVER_ID       1000
#define ECHECK( x, y )  if ( x < 0 ) \
                                { \
                fprintf( stderr, "%s\n", y ); \

                exit( -1 ); \
                                }

#define validdata( fld ) \
        ( ( fld.type & F_PTR ? fld.x.dptr[0] :
fld.x.data[0] ) != 0 )

#define orderstatl( n )         ((n*5) + 10)
#define neworderl( n )  ((n*8) + 11)

#ifdef  roundup                 /* for linux */
#undef  roundup
#endif
#define roundup( a ) ( ((a)>0) ? (int)((a)+0.9) :
((int)(-(a)+0.9))*(-1) )

#define INTNULL         0
#define CHECKOK -1

#define TX_NEWORDER 1
#define TX_PAYMENT 2
#define TX_ORDERSTAT 3
#define TX_DELIVERY 4
#define TX_STOCKLVL 5

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

void Tstatus( char * );
void changestatus1( char * );
void errorstatus( char * );
void convert_datetime( char *, char * );
void convert_date( char *, char * );
int checkfields( DataField *, int, int, int, ... );
int check_neworder_lines( void );
void delivery_screen( void );
void stocklvl_screen( void );
void payment_screen( void );
void orderstat_screen( void );
void neworder_screen( void );

void init_tux();
void clean_tux();
void fatalerror( char * );
void interrupt( int );

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

WINDOW  *win, *statwin;

int             trans_size = 1024;
void            *trans_buf;

int             w_id, d_id;
long            olen;
check( char *CP; )

extern TextField delivery_text[];
extern DataField delivery_data[];
extern TextField stocklvl_text[];
extern DataField stocklvl_data[];
extern TextField payment_text[];
extern DataField payment_data[];
extern TextField orderstat_text[];
extern DataField orderstat_data[];
extern TextField neworder_text[];
extern DataField neworder_data[];

static char     NewOrdername[20];
static char     Paymentname[20];
static char     OrderStatusname[20];
static char     Deliveryname[20];
static char     StockLevelname[20];
static int      svrnum;

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

#define         SETSVRNUM( NTERM )              \
                        if ( num < (NTERM) ) {
                        \
                                num = -1;
                        \
                                /* no need
more check (^^; */          \
                        } else {
                        \
                                num -=
(NTERM);                        \
                                svrnum++;
                        \
                        }

void TPCframe( int tid )
{
        bool            exitflag = FALSE;
        int             c;
        int             num;
                        /* temp variable */

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

        check( CP = "TPCframe" );
        svrnum = 1;
        /* initialize */

        if ( tid <= 800 ) {
        /* ptnds08a/b */
                        num = tid - 1;
                        SETSVRNUM( 50 );
                        /* 1 */
                        SETSVRNUM( 50 );
                        /* 2 */
                        SETSVRNUM( 50 );
                        /* 3 */
                        SETSVRNUM( 50 );
                        /* 4 */
                        SETSVRNUM( 50 );
                        /* 5 */
                        SETSVRNUM( 50 );
                        /* 6 */
                        SETSVRNUM( 50 );
                        /* 7 */
                        SETSVRNUM( 50 );
                        /* 8 */
                        SETSVRNUM( 50 );
                        /* 9 */
                        SETSVRNUM( 50 );
                        /* 10 */
                        SETSVRNUM( 50 );
                        /* 11 */
                        SETSVRNUM( 50 );
                        /* 12 */
                        SETSVRNUM( 50 );
                        /* 13 */
```

```
                SETSVRNUM( 50 );
                /* 14 */
                SETSVRNUM( 50 );
                /* 15 */
                SETSVRNUM( 50 );
                /* 16 */
        } else if ( tid <= 1600 ) {
        /* ptnds07a */
                num = tid - 801;
                SETSVRNUM( 50 );
                /* 1 */
                SETSVRNUM( 50 );
                /* 2 */
                SETSVRNUM( 50 );
                /* 3 */
                SETSVRNUM( 50 );
                /* 4 */
                SETSVRNUM( 50 );
                /* 5 */
                SETSVRNUM( 50 );
                /* 6 */
                SETSVRNUM( 50 );
                /* 7 */
                SETSVRNUM( 50 );
                /* 8 */
                SETSVRNUM( 50 );
                /* 9 */
                SETSVRNUM( 50 );
                /* 10 */
                SETSVRNUM( 50 );
                /* 11 */
                SETSVRNUM( 50 );
                /* 12 */
                SETSVRNUM( 50 );
                /* 13 */
                SETSVRNUM( 50 );
                /* 14 */
                SETSVRNUM( 50 );
                /* 15 */
                SETSVRNUM( 50 );
                /* 16 */
        } else if ( tid <= 2000 ) {
        /* ptnds06a */
                num = tid - 1601;
                SETSVRNUM( 10 );
                /* 1 */
                SETSVRNUM( 10 );
                /* 2 */
                SETSVRNUM( 10 );
                /* 3 */
                SETSVRNUM( 10 );
                /* 4 */
                SETSVRNUM( 10 );
                /* 5 */
                SETSVRNUM( 10 );
                /* 6 */
        } else if ( tid <= 2400 ) {
        /* ptnds05a */
                num = tid - 2001;
                SETSVRNUM( 10 );
                /* 1 */
                SETSVRNUM( 10 );
                /* 2 */
                SETSVRNUM( 10 );
                /* 3 */
                SETSVRNUM( 10 );
                /* 4 */
                SETSVRNUM( 10 );
                /* 5 */
                SETSVRNUM( 10 );
                /* 6 */
        } else if ( tid <= 3200 ) {
        /* ptnds02a */
                num = tid - 2401;
                SETSVRNUM( 50 );
                /* 1 */
                SETSVRNUM( 50 );
                /* 2 */
                SETSVRNUM( 50 );
                /* 3 */
                SETSVRNUM( 50 );
                /* 4 */
                SETSVRNUM( 50 );
                /* 5 */
                SETSVRNUM( 50 );
                /* 6 */
                SETSVRNUM( 50 );
                /* 7 */
                SETSVRNUM( 50 );
                /* 8 */
                SETSVRNUM( 50 );
                /* 9 */
                SETSVRNUM( 50 );
                /* 10 */
                SETSVRNUM( 50 );
                /* 11 */
                SETSVRNUM( 50 );
                /* 12 */
                SETSVRNUM( 50 );
                /* 13 */
                SETSVRNUM( 50 );
                /* 14 */
                SETSVRNUM( 50 );
                /* 15 */
                SETSVRNUM( 50 );
                /* 16 */
        } else if ( tid <= 4000 ) {
        /* ptnds01a/b */
                num = tid - 3201;
                SETSVRNUM( 50 );
                /* 1 */
                SETSVRNUM( 50 );
                /* 2 */
                SETSVRNUM( 50 );
                /* 3 */
                SETSVRNUM( 50 );
                /* 4 */
                SETSVRNUM( 50 );
                /* 5 */
                SETSVRNUM( 50 );
                /* 6 */
                SETSVRNUM( 50 );
                /* 7 */
                SETSVRNUM( 50 );
                /* 8 */
                SETSVRNUM( 50 );
                /* 9 */
                SETSVRNUM( 50 );
                /* 10 */
                SETSVRNUM( 50 );
                /* 11 */
                SETSVRNUM( 50 );
                /* 12 */
                SETSVRNUM( 50 );
                /* 13 */
                SETSVRNUM( 50 );
                /* 14 */
                SETSVRNUM( 50 );
                /* 15 */
                SETSVRNUM( 50 );
                /* 16 */
        } else {
#ifndef SCRTEST
                char            buf[64];

                sprintf( buf, "Illegal T_ID:
%d\n", tid );
                fatalerror( buf );
#endif
        }

#ifdef DEBUG
        fprintf( stderr, "(W:%d D:%d S:%d)\n",
w_id, d_id, svrnum );
#endif

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

#if 0
        sprintf( NewOrdername,
"NEWORDER", svrnum );
        sprintf( Paymentname,
"PAYMENT", svrnum );
        sprintf( OrderStatusname,
"ORDERSTATUS", svrnum );
        sprintf( Deliveryname,    "DELIVERY",
svrnum );
        sprintf( StockLevelname,
"STOCKLEVEL", svrnum );
#endif
        sprintf( NewOrdername,
"TPCC%02d", svrnum );
        sprintf( Paymentname,
"TPCC%02d", svrnum );
        sprintf( OrderStatusname,
"TPCC%02d", svrnum );
        sprintf( Deliveryname,
"TPCC%02d", svrnum );
        sprintf( StockLevelname,
"TPCC%02d", svrnum );

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

        initScreen( "", UI_NOBORDER );
        win = createWindow( 0, 0, 23, 80, "",
WIN_NOBORDER, ATTR_BASE, 0 );
        statwin = createWindow( LINES-1,  0,
1, 80, "", WIN_NOBORDER,
                ATTR_BASE, 0 );

        while ( !exitflag )
        {
                Tstatus( "D:Delivery
S:Stock Level P:Payment O:Order Status"
                                " N:New
Order Q:Quit" );
                wmove( statwin, 0, COLS
);
                touchwin( win );
                nrefresh( 2, win, statwin
);

                check( CP = "Loop"; )
```

```
#if defined(DODELAYEDUPDATE) &&
!defined(USE_OLD)
                c = wgetch_buf( win );
#else
                c = getch();
#endif

                switch( c )
                {
                case 'n' :
                case 'N' :
                        wclear( win );

        changestatus1( "New Order" );

        neworder_screen();
                        break;
                case 'p' :
                case 'P' :
                        wclear( win );

        changestatus1( "Payment" );

        payment_screen();
                        break;
                case 's' :
                case 'S' :
                        wclear( win );

        changestatus1( "Stock Level" );

        stocklvl_screen();
                        break;
                case 'o' :
                case 'O' :
                        wclear( win );

        changestatus1( "Order Status" );

        orderstat_screen();
                        break;
                case 'd' :
                case 'D' :
                        wclear( win );

        changestatus1( "Delivery" );

        delivery_screen();
                        break;
                case 'q' :
                case 'Q' :
                        exitflag =
TRUE;
                        break;
                default :
                        debug( (
stderr, "Illegal Select : '%c'\n", c ) );
                        break;
                }
        }

        closeScreen();
#ifndef SCRTEST
        tpterm();
#endif

}
```

```
void Tstatus( char *status )
{
        int             len = strlen( status );
#if defined( DEBUG ) && DEBUG > 40
        WINDOW    *save_statwin = NULL,
*save_win = NULL;

        debug1( ( stderr, "IN:\t Tstatus()\n" ) );
        debug2( ( stderr, "ARGS:\t status:
'%s'\n", status ) );

        if ( save_statwin == NULL ) {
                save_statwin = statwin;
                save_win = win;
        } else {
                if ( save_statwin !=
statwin || save_win != win ) {
                        printf( "Oops!
Not equal!\n" );
                }
                debug1( ( stderr, "EXIT:\t
Tstatus()\n" ) );
                exit(1);
        }
#else
        debug1( ( stderr, "IN:\t Tstatus()\n" ) );
        debug2( ( stderr, "ARGS:\t status:
'%s'\n", status ) );
#endif

        wmove( statwin, 0, (COLS/2)-(len/2) );
        wdeleteln( statwin );
        waddstr( statwin, status );

        debug1( ( stderr, "OUT:\t Tstatus()\n"
) );
}

void changestatus1( char *s )
{
        char            buf[80];

        debug1( ( stderr, "IN:\t
changestatus1()\n" ) );
        debug2( ( stderr, "ARGS:\t s: '%s'\n",
s ) );

        sprintf( buf, "%s screen...Use  arrow
keys to move "
                        "... Enter
data in fields", s );
        Tstatus( buf );
        wrefresh( statwin );

        debug1( ( stderr, "OUT:\t
changestatus1()\n" ) );
}

void errorstatus( char *s )
{
        char buf[80];

        debug1( ( stderr, "IN:\t errorstatus()\n"
) );
        debug2( ( stderr, "ARGS:\t s: '%s'\n",
s ) );

        sprintf( buf, "%s screen...Insufficient
data ... "
                        "Enter data in
fields", s );
        Tstatus( buf );
        wrefresh( statwin );

        debug( ( stderr, "Insufficient data in:
%s\n", s ) );
        debug1( ( stderr, "OUT:\t
errorstatus()\n" ) );
}

void convert_datetime( char *out, char *in )
{
        int             year, month, day, hour,
min, sec;

        debug1( ( stderr, "IN:\t
convert_datetime()\n" ) );
        debug2( ( stderr, "ARGS:\t in: '%s'\n",
in ) );

        sscanf( in, "%d-%d-%d.%d:%d:%d",
                &day, &month, &year,
&hour, &min, &sec );
        sprintf( out,
"%02d%02d%04d%02d%02d%02d",
                        day, month,
year, hour, min, sec );

        debug2( ( stderr, "RTN:\t out: '%s'\n",
out ) );
        debug1( ( stderr, "OUT:\t
convert_datetime()\n" ) );
}

void convert_date( char *out, char *in )
{
        int             year, month, day;

        debug1( ( stderr, "IN:\t
convert_date()\n" ) );
        debug2( ( stderr, "ARGS:\t in: '%s'\n",
in ) );

        sscanf( in, "%d-%d-%d", &day,
&month, &year );
        sprintf( out, "%02d%02d%04d", day,
month, year );

        debug2( ( stderr, "RTN:\t out: '%s'\n",
out ) );
        debug1( ( stderr, "OUT:\t
convert_date()\n" ) );
}

int checkfields( DataField *fld, int opt1, int opt2, int
n, ... )
{
        va_list         l;
        int             i, f;

        debug1( ( stderr, "IN:\t
checkfields()\n" ) );

        va_start( l, n );
```

```
	if ( ( opt1 >= 0 )  && ( opt2 >= 0 ) )
	{
		if ( !validdata( fld[opt1] )
&& !validdata( fld[opt2] ) )
		{
			if ( !validdata(
fld[opt1] ) )
			{

	debug2( ( stderr, "RTN:\t 1: %d\n",
opt1 ) );

	debug1( ( stderr, "OUT:\t
checkfields()\n" ) );

	return opt1;
			}
			else
			{

	debug2( ( stderr, "RTN:\t 2: %d\n",
opt2 ) );

	debug1( ( stderr, "OUT:\t
checkfields()\n" ) );

	return opt2;
			}
		}
	}

	for ( i = 0; i < n; i++ )
	{
		f = va_arg( l, int );

		if ( !validdata( fld[f] ) )
		{
			debug2( (
stderr, "RTN:\t f: %d\n", f ) );
			debug1( (
stderr, "OUT:\t checkfields()\n" ) );
			return f;
		}
	}

	va_end( l );

	debug2( ( stderr, "RTN:\t 3: %d\n",
CHECKOK ) );
	debug1( ( stderr, "OUT:\t
checkfields()\n" ) );

	return CHECKOK;
}

int check_neworder_lines()
{
	int	i, errf;
	bool	allok = TRUE, allclear =
TRUE;

#define	checkcell( i, j ) \
	if ( validdata(
neworder_data[neworderl( i )+j] ) ) \
	{ \
		allclear = FALSE; \
	} \
	else \
	{ \
		allok = FALSE; \
	}

	debug1( ( stderr, "IN:\t
check_neworder_lines()\n" ) );

	if ( !validdata(
neworder_data[neworderl( 0 )] ) )
	{
		debug1( ( stderr, "OUT:\t
check_neworder_lines()\n" ) );
		return neworderl( 0 );
	}
	else if ( !validdata(
neworder_data[neworderl( 0 )+1] ) )
	{
		debug1( ( stderr, "OUT:\t
check_neworder_lines()\n" ) );
		return neworderl( 0 )+1;
	}
	else if ( !validdata(
neworder_data[neworderl( 0 )+3] ) )
	{
		debug1( ( stderr, "OUT:\t
check_neworder_lines()\n" ) );
		return neworderl( 0 )+3;
	}
	else
	{
		for( i = 1; ( i < 15 ) && (
allok || allclear ); i++ )
		{
			allok =
TRUE;
			allclear =
TRUE;

			checkcell( i,
0 );
			checkcell( i,
1 );
			checkcell( i,
3 );
		}

		if ( i == 15 )
		{
			debug1( (
stderr, "OUT:\t check_neworder_lines()\n" ) );
			return
CHECKOK;
		}
		else
		{
			i--;

			if ( !validdata(
neworder_data[neworderl( i )] ) )
			{

	debug1( ( stderr,

	"OUT:\t check_neworder_lines()\n" ) );

	return neworderl( i );
			}

			if ( !validdata(
neworder_data[neworderl( i )+1] ) )
			{

	debug1( ( stderr,

	"OUT:\t check_neworder_lines( )\n" ) );

	return neworderl( i )+1;
			}

			if ( !validdata(
neworder_data[neworderl( i )+3] ) )
			{

	debug1( ( stderr,

	"OUT:\t check_neworder_lines()\n" ) );

	return neworderl( i )+3;
			}
		}
	}

	debug1( ( stderr, "OUT:\t
check_neworder_lines()\n" ) );
}

/*
	Delivery screen
*/

void delivery_screen()
{
	struct delstruct	*bp;
	int				i;
	int
	rtn;
	struct timeval
	timeque;

	bp = ( struct delstruct * )trans_buf;
	bp->tran_kind = TRANDEL;
	check( CP = "delivery_screen"; )

	/* Preset scre en data */

	for ( i=0; !eos( delivery_data[i] ); i++ )
{
		delivery_data[i].x.data[0]
= '\0';
	}
	sprintf( delivery_data[0].x.data, "%d",
w_id );
	delivery_data[2].x.dptr = "";

	queryframe( win, FR_FULLSCREEN,
delivery_text, delivery_data );

	while ( ( i = checkfields( delivery_data,
-1, -1, 1, 1 ) ) != CHECKOK )
	{
		delivery_data[1].type &=
~F_START;
		delivery_data[i].type |=
F_START;

		errorstatus( "Delivery" );
```

```
                queryframe( win,
FR_RETRY, delivery_text, delivery_data );

                delivery_data[i].type &=
~F_START;
                delivery_data[1].type |=
F_START;
        }


        /* Get screen data and send to
database */

        bp->delin.w_id = w_id;
        bp->delin.o_carrier_id = atoi(
delivery_data[1].x.data );
        bp->delin.in_timing_int = 1;
        /* bp->delin.in_timing_int = (
is_measurement() ) ? 1 : 0 */

        debug2( ( stderr, "Delivery -- w_id :
%d, o_carrier_id : %d\n",
                bp->delin.w_id, bp-
>delin.o_carrier_id ) );

resend_delivery:
        check( CP = "delivery_tpcall"; )
        debug1( ( stderr, "INFO:\t tpacall
delivery!\n" ) );
#ifndef SCRTEST
        gettimeofday(&timeque);
#ifdef  TOOLKIT_ORIGINAL_STRUCTURE
        /* 1996.08.07 */
        bp->delin.qtime = ( double
)timeque.tv_sec

        + ( double )timeque.tv_usec /
1000000.0;
#else   /*
!TOOLKIT_ORIGINAL_STRUCTURE */
        bp->delin.qtime  = timeque.tv_sec;
        bp->delin.uqtime = timeque.tv_usec;
#endif  /*
!TOOLKIT_ORIGINAL_STRUCTURE */

        rtn = tpacall( Deliveryname, ( char *
)trans_buf,
                sizeof( struct delstruct ),
TPSIGRSTRT | TPNOREPLY );
#else
        dummy_delivery( bp );
        rtn = 0;
#endif
        debug2( ( stderr, "RTN:\t
delivery_data %s\n",
                delivery_data[1].x.data )
);

        /* Display messege */

        if ( rtn == -1 ) {
#ifndef SCRTEST
                char    buf[1024];

                switch ( tperrno ) {
                case TPELIMIT:
                case TPETIME:
                case TPGOTSIG:
                        debug( (
stderr,

        "tpacall : Retry in Delivery: tperrno =
%d\n"
                                        "
svc = '%s' carrier = %d\n", tperrno,

        Deliveryname, bp->delin.o_carrier_id )
);
                        goto
resend_delivery;
                        break;
                default:
                        sprintf( buf,

        "tpacall failed in Delivery: tperrno =
%d\n"
                                        "
svc = '%s' carrier = %d\n", tperrno,

        Deliveryname, bp->delin.o_carrier_id
);
                        fatalerror( buf
);
                        break;
                }
#endif
        } else {
                delivery_data[2].x.dptr =
"Delivery has been queued";
                display_fields( win,
FR_FULLSCREEN, delivery_text,
                        delivery_data
);
        }

}

/*
        Stock Level screen
*/

void stocklvl_screen()
{
        struct stostruct        *bp;
        int                     i;

        bp = ( struct stostruct * )trans_buf;
        bp->tran_kind = TRANSTO;
        check( CP = "stocklvl_screen"; )

        /* Preset screen data */

        for ( i = 0; !eos( stocklvl_data[i] ); i++ )
        {
                stocklvl_data[i].x.data[0]
= '\0';
        }
        sprintf( stocklvl_data[0].x.data, "%d",
w_id );
        sprintf( stocklvl_data[1].x.data, "%d",
d_id );

        queryframe( win, FR_FULLSCREEN,
stocklvl_text, stocklvl_data );

        while ( ( i = checkfields( stocklvl_data,
-1, -1, 1, 2 ) ) != CHECKOK )
        {
                stocklvl_data[2].type &=
~F_START;
                stocklvl_data[i].type |=
F_START;

                errorstatus( "Stock Level"
);
                queryframe( win,
FR_RETRY, stocklvl_text, stocklvl_data );

                stocklvl_data[i].type &=
~F_START;
                stocklvl_data[2].type |=
F_START;
        }

        /* Get screen data and send to
database */

        bp->stoin.w_id = atoi(
stocklvl_data[0].x.data );
        bp->stoin.d_id = atoi(
stocklvl_data[1].x.data );
        bp->stoin.threshold = atoi(
stocklvl_data[2].x.data );

        debug2( ( stderr, "Stocklevel -- w_id :
%d, d_id : %d,"
                " threshold : %d\n", bp-
>stoin.w_id, bp->stoin.d_id,
                bp->stoin.threshold ) );

resend_stock:
        check( CP = "stocklevel_tpcall"; )
        debug1( ( stderr, "INFO:\t tpcall
stocklevel!\n" ) );
#ifndef SCRTEST
        if ( tpcall( StockLevelname,
                ( char * )trans_buf,
sizeof( struct stostruct ),
                ( char ** )&trans_buf,
&olen, 0 ) == -1 )
        {
                char    buf[1024];

                sprintf( buf, "tpcall failed
in StockLevel: tperrno = %d\n"
                        " svc = '%s'
threshold = %d\n", tperrno,

        StockLevelname, bp->stoin.threshold
);
                fatalerror( buf );
        }
        bp = ( struct stostruct * )trans_buf;
#else
        dummy_stocklvl( bp );
#endif

        if ( bp->stoout.terror != NOERR )
        {
                if ( bp->stoout.terror ==
IRRECERR ) {
                        fatalerror(
"Irrecoverable error in stocklevel\n" );
```

```
                        }
                        debug( ( stderr, "terror :
%d\n", bp->stoout.terror ) );
                        goto resend_stock;
                }

                debug2( ( stderr, "Stocklevel --
low_stock = %d\n",
                        bp->stoout.low_stock ) );

                sprintf( stocklvl_data[3].x.data, "%d" ,
bp->stoout.low_stock );

                display_fields( win,
FR_FULLSCREEN, stocklvl_text, stocklvl_data );

}

/*
                Payment screen
*/

void payment_screen()
{
                struct paystruct                *bp;
                int                                             i;

                bp = ( struct paystruct * )trans_buf;
                bp->tran_kind = TRANPAY;
                check( CP = "payment_screen"; )

                /* payment screen data */

                for ( i = 0; !eos( paym ent_data[i] ); i++
)
                {
                        if ( payment_data[i].type
& F_PTR )
                        {

                payment_data[i].x.dptr = "";
                        }
                        else
                        {

                payment_data[i].x.data[0] = '\0';
                        }
                }
                sprintf( payment_data[1].x.data, "%d",
w_id );
                payment_data[18].x.dptr = bp-
>payin.c_last;
                payment_data[18].x.dptr[0] = '\0';

                queryframe( win, FR_FULLSCREEN,
payment_text, payment_data );

                while ( ( i = checkfields(
payment_data, 13, 18, 4, 2, 14, 15, 28 ) )
                        != CHECKOK )
                {
                        payment_data[2].type &=
~F_START;
                        payment_data[i].type |=
F_START;

                        errorstatus( "Payment" );
                        queryframe( win,
FR_RETRY, payment_text, payment_data );

                        payment_data[i].type &=
~F_START;
                        payment_data[2].type |=
F_START;
                }


                /* Get screen data and to database*/

                bp->payin.w_id = atoi(
payment_data[1].x.data );
                bp->payin.d_id = atoi(
payment_data[2].x.data );
                bp->payin.c_id = atoi(
payment_data[13].x.data );
                bp->payin.c_w_id = atoi(
payment_data[14].x.data );
                bp->payin.c_d_id = atoi(
payment_data[15].x.data );
                bp->payin.h_amount = (float)( atoi(
payment_data[28].x.data )/100.0 );
                if ( *bp->payin.c_last == '\0' ) {
                        bp->payin.bylastname =
0;
                } else {
                        bp->payin.bylastname =
1;
                }

                debug2( ( stderr, "Payment -- w_id :
%d, d_id : %d, c_id : %d\n"
                        "       -- c_w_id : %d,
c_d_id : %d, h_amount : %d\n",
                        bp->payin.w_id, bp-
>payin.d_id, bp->payin.c_id,
                        bp->payin.c_w_id, bp-
>payin.c_d_id, bp->payin.h_amount ) );

resend_payment:
#if defined( DEBUG ) && ( DEBUG > 40 )
#ifndef         SCRTEST
                fprintf( stderr, "Name : %s\n",
payment_name );
#endif          /* SCRTEST */
                fprintf( stderr, "Pointer : %08X\n", bp
);
                fprintf( stderr, "Length : %08d\n",
sizeof( struct paystruct ) );
                fprintf( stderr, "PointerAddr : %08X\n",
&bp );
#endif


                check( CP = "payment_tpcall"; )
                debug1( ( stderr, "INFO:\t tpcall
payment!\n" ) );
#ifndef         SCRTEST
                if ( tpcall( Paymentname,
                        ( char * )trans_buf,
sizeof( struct paystruct ),
                        ( char ** )&trans_buf,
&olen, 0 ) == -1 )
                {
                        char            buf[1024];
                        sprintf( buf, "tpcall failed
in Payment: tperrno = %d\n"
                        " svc = '%s'
d_id = %d c_id = %d c_last = '%s'\n"
                        " c_w_id =
%d c_d_id = %d h_amount = %d\n",
                        tperrno,
Paymentname, bp->payin.d_id, bp->payin.c_id,
                        bp-
>payin.c_last, bp->payin.c_w_id, bp-
>payin.c_d_id,
                        bp-
>payin.h_amount );
                        fatalerror( buf );
                }
                bp = ( struct paystruct * )trans_buf;
#else
                dummy_payment( bp );
#endif

                debug4( ( stderr ,"RetLength :
%08d\n", olen ) );
                debug4( ( stderr, "RetPointer :
%08X\n", bp ) );

                if ( bp->payout.terror != NOERR )
                {
                        debug2( ( stderr,
                        "Payment --
w_id : %d, d_id : %d, c_id : %d\n"
                        "       --
c_w_id : %d, c_d_id : %d, h_amount : %d\n",
                        bp-
>payin.w_id, bp->payin.d_id, bp->payin.c_id,
                        bp-
>payin.c_w_id, bp->payin.c_d_id,
                        bp-
>payin.h_amount ) );

                        if ( bp->payout.terror ==
IRRECERR ) {
                        fatalerror(
"Irrecoverable error in Payment\n" );
                        }

                        debug( ( stderr, "terror :
%d\n", bp->payout.terror ) );
                        goto resend_payment;
                }

                convert_datetime(
payment_data[0].x.data, bp->payout.h_date );
                payment_data[3].x.dptr = bp-
>payout.w_street_1;
                payment_data[4].x.dptr = bp-
>payout.d_street_1;
                payment_data[5].x.dptr = bp-
>payout.w_street_2;
                payment_data[6].x.dptr = bp-
>payout.d_street_2;
                payment_data[7].x.dptr = bp-
>payout.w_city;
                payment_data[8].x.dptr = bp-
>payout.w_state;
                payment_data[9].x.dptr = bp-
>payout.w_zip;
                payment_data[10].x.dptr = bp-
>payout.d_city;
```

```
		payment_data[11].x.dptr = bp-
>payout.d_state;
		payment_data[12].x.dptr = bp-
>payout.d_zip;
		sprintf( payment_data[13].x.data,
"%d", bp->payout.c_id );
		payment_data[16].x.dptr = bp-
>payout.c_first;
		payment_data[17].x.dptr = bp-
>payout.c_middle;
		payment_data[18].x.dptr = bp-
>payout.c_last;
		convert_date(
payment_data[19].x.data, bp->payout.c_since );
		payment_data[20].x.dptr = bp-
>payout.c_street_1;
		payment_data[21].x.dptr = bp-
>payout.c_credit;
		payment_data[22].x.dptr = bp-
>payout.c_street_2;
		sprintf( payment_data[23].x.data,
"%d",
			roundup( bp-
>payout.c_discount * 10000.0 ) );
		payment_data[24].x.dptr = bp-
>payout.c_city;
		payment_data[25].x.dptr = bp-
>payout.c_state;
		payment_data[26].x.dptr = bp-
>payout.c_zip;
		payment_data[27].x.dptr = bp-
>payout.c_phone;
		/* "%.0f" --> number is rounded to the
appropriate number of digits */
		sprintf( payment_data[29].x.data,
"%.0f", bp->payout.c_balance*100.0 );
		sprintf( payment_data[30].x.data,
"%.0f", bp->payout.c_credit_lim*100.0 );

		i = strlen( bp->payout.c_data );
		strncpy( payment_data[31].x.data, bp-
>payout.c_data, 50 );
		if ( i > 50 )
		{
			strncpy(
payment_data[32].x.data, &bp-
>payout.c_data[50], 50 );
			if ( i > 100 )
			{
				strncpy(
payment_data[33].x.data,

		&bp->payout.c_data[100], 50 );
				if ( i > 150 )
				{

		strncpy( payment_data[34].x.data,

		&bp->payout.c_data[150], 50 );
				}
			}
		}

		display_fields( win,
FR_FULLSCREEN, payment_text, payment_data
);

}
```

```
/*
		Order status screen
*/

void orderstat_screen()
{
		struct ordstruct		*bp;
		int					i;

		bp = ( struct ordstruct * )trans_buf;
		bp->tran_kind = TRANORD;
		check( CP = "orderstat_screen"; )

		/* Preset screen data */

		for( i = 0; !eos( orderstat_data[i] ); i++
)
		{
			if ( orderstat_data[i].type
& F_PTR )
			{

		orderstat_data[i].x.dptr = "";
			}
			else
			{

		orderstat_data[i].x.data[0] = '\0';
			}
		}
		sprintf( orderstat_data[0].x.data, "%d",
w_id );
		orderstat_data[5].x.dptr = bp-
>ordin.c_last;
		orderstat_data[5].x.dptr[0] = '\0';

		queryframe( win, FR_FULLSCREEN,
orderstat_text, orderstat_data );

		while ( ( i = checkfields(
orderstat_data, 2, 5, 1, 1 ) ) != CHECKOK )
		{
			orderstat_data[1].type &=
~F_START;
			orderstat_data[i].type |=
F_START;

			errorstatus( "Order
Status" );
			queryframe( win,
FR_RETRY, orderstat_text, orderstat_data );

			orderstat_data[i].type &=
~F_START;
			orderstat_data[1].type |=
F_START;
		}

		/* Get Screen data and send to
database */

		bp->ordin.w_id = atoi(
orderstat_data[0].x.data );
		bp->ordin.d_id = atoi(
orderstat_data[1].x.data );
		bp->ordin.c_id = atoi(
orderstat_data[2].x.data );
		if ( *bp->ordin.c_last == '\0' ) {
			bp->ordin.bylastname =
0;
		} else {
			bp->ordin.bylastname =
1;
		}

		debug2( ( stderr,
			"Orderstatus -- w_id : %d,
d_id : %d, c_id : %d\n",
			bp->ordin.w_id, bp-
>ordin.d_id, bp->ordin.c_id ) );

resend_orderstatus:
		check( CP = "orderstatus_tpcall"; )
		debug1( ( stderr, "INFO:\t tpcall
orderstatus!\n" ) );
#ifndef SCRTEST
		if ( tpcall( OrderStatusname, ( char *
)trans_buf,
			sizeof( struct ordstruct ), (
char ** )&trans_buf, &olen, 0 )
			== -1 )
		{
			char		buf[1024];

			sprintf( buf, "tpcall failed
in OrderStatus: tperrno = %d\n"
				" svc = '%s'
d_id = %d c_id = %d c_last = '%s'\n",
				tperrno,
OrderStatusname, bp->ordin.d_id,
				bp-
>ordin.c_id, bp->ordin.c_last );
			fatalerror( buf );
		}
		bp = ( struct ordstruct * )trans_buf;

		debug2( ( stderr, "balance : %.0f\n",
bp->ordout.c_balance ) );
		debug2( ( stderr, "first : %s\n", bp-
>ordout.c_first ) );
		debug2( ( stderr, "middle : %s\n", bp-
>ordout.c_middle ) );
		debug2( ( stderr, "entry : %f\n", bp-
>ordout.o_entry_d ) );
		debug2( ( stderr, "o_ol_cnt : %d\n",
bp->ordout.o_ol_cnt ) );
#else
		dummy_orderstat( bp );
#endif

		if ( bp->ordout.terror != NOERR )
		{
			debug2( ( stderr,
"Orderstatus -- w_id : %d, d_id : %d,"
				"c_id : %d\n",
				bp-
>ordin.w_id, bp->ordin.d_id, bp->ordout.c_id ) );

			if ( bp->ordout.terror ==
IRRECERR ) {
				fatalerror(
"Irrecoverable error in orderstatus.\n" );
			}
```

```
                debug( ( stderr, "C_R :
%d\n", bp->ordout.terror ) );
                goto resend_orderstatus;
        }

        sprintf( orderstat_data[2].x.data, "%d",
bp->ordout.c_id );
        orderstat_data[3].x.dptr = bp-
>ordout.c_first;
        orderstat_data[4].x.dptr = bp-
>ordout.c_middle;
        orderstat_data[5].x.dptr = bp-
>ordout.c_last;
        /* "%.0f" --> number is rounded to the
appropriate number of digits */
        sprintf( orderstat_data[6].x.data,
"%.0f", bp->ordout.c_balance*100.0 );
        sprintf( orderstat_data[7].x.data, "%d",
bp->ordout.o_id );
        convert_datetime(
orderstat_data[8].x.data, bp->ordout.o_entry_d );

        if ( bp->ordout.o_carrier_id !=
INTNULL ) {
                sprintf(
orderstat_data[9].x.data, "%d",
                        bp-
>ordout.o_carrier_id );
        }

        for( i = 0; i < bp->ordout.o_ol_cnt; i++
)
        {
                sprintf(
orderstat_data[orderstatl( i )].x.data, "%d",
                        bp-
>ordout.ol_supply_w_id[i] );
                sprintf(
orderstat_data[orderstatl( i )+1].x.data, "%d",
                        bp-
>ordout.ol_i_id[i] );
                sprintf(
orderstat_data[orderstatl( i )+2].x.data, "%d",
                        bp-
>ordout.ol_quantity[i] );
                sprintf(
orderstat_data[orderstatl( i )+3].x.data, "%d",
                        roundup( bp-
>ordout.ol_amount[i]*100.0 ) );
                if( strncmp( bp-
>ordout.ol_delivery_d[i], "NOT DELIVR", 10 )
                        != 0 )
                {

        convert_date(
orderstat_data[orderstatl( i )+4].x.data,

        bp->ordout.ol_delivery_d[i] );

                }
        }

        display_fields( win,
FR_FULLSCREEN, orderstat_text, orderstat_data
);

}

/*
        New Order screen
*/

void neworder_screen( )
{
        struct newstruct        *bp;
        int                             i,
j;

        bp = ( struct newstruct * )trans_buf;
        bp->tran_kind = TRANNEW;
        check( CP = "neworder_screen"; )

        /* Preset screen data */

        for ( i = 0; !eos( neworder_data[i] );
i++ )
        {
                if ( neworder_data[i].type
& F_PTR)
                {

        neworder_data[i].x.dptr = "";
                }
                else
                {

        neworder_data[i].x.data[0] = '\0';
                }
        }
        sprintf( neworder_data[0].x.data,
"%d", w_id );
        neworder_data[4].x.dptr = bp-
>newout.c_last;
        neworder_data[4].x.dptr[0] = '\0';
        neworder_data[131].x.dptr = "";

        queryframe( win, FR_FULLSCREEN,
neworder_text, neworder_data );

        while ( ( ( i = checkfields(
neworder_data, -1, -1, 2, 1, 3 ) )
                                        !=
CHECKOK )
                || ( ( i =
check_neworder_lines() ) != CHECKOK ) )
        {
                if ( i == CHECKOK )
                {
                        i = j;
                }

                neworder_data[1].type &=
~F_START;
                neworder_data[i].type |=
F_START;

                errorstatus( "New Order"
);
                queryframe( win,
FR_RETRY, neworder_text, neworder_data );

                neworder_data[i].type &=
~F_START;
                neworder_data[1].type |=
F_START;
        }

        /* Get Screen data and to database  */

        bp->newin.w_id = atoi(
neworder_data[0].x.data );
        bp->newin.d_id = atoi(
neworder_data[1].x.data );
        bp->newin.c_id = atoi(
neworder_data[3].x.data );

        for ( i = 0; ( neworder_data[neworderl(
i )].x.data[0] != 0 )
                        && ( i < 15 ); i++ )
        {
                bp-
>newin.ol_supply_w_id[i]
                        = atoi(
neworder_data[neworderl( i )].x.data );
                bp->newin.ol_quantity[i]
                        = atoi(
neworder_data[neworderl( i )+3].x.data );
                bp->newin.ol_i_id[i]
                        = atoi(
neworder_data[neworderl( i )+1].x.data );
                if ( bp->newin.ol_i_id[i] ==
0 ) {
                        bp-
>newin.ol_i_id[i] = -1;     /* Invalid Item-ID */

                        /* for Oracle
T.K. */
                }
        }
        if ( i < 15 ) {
                /* terminate */
                bp-
>newin.ol_supply_w_id[i] = 0;
                bp->newin.ol_quantity[i] =
0;
                bp->newin.ol_i_id[i] = 0;
        }

        debug2( ( stderr, "NewOrder -- w_id :
%d, d_id : %d, c_id : %d,"
                " lines : %d\n",
                bp->newin.w_id, bp-
>newin.d_id, bp->newin.c_id, i ) );

resend_neworder:
        check( CP = "neworder_tpcall"; )
        debug1( ( stderr, "INFO:\t tpcall
neworder!\n" ) );
#ifndef     SCRTEST
        if ( tpcall( NewOrdername, ( char *
)trans_buf,
                sizeof( struct newstruct ),
( char ** )&trans_buf, &olen, 0 )
                == -1 )
        {
                char            buf[1024];

                sprintf( buf, "tpcall failed
in NewOrder: tperrno = %d\n"
                        " svc = '%s'
d_id = %d c_id = %d lines = %d\n",
                        tperrno,
NewOrdername, bp->newin.d_id, bp->newin.c_id,
                        i );
```

```
                        fatalerror( buf );
                }
                bp = ( struct newstruct * )trans_buf;
#else
                dummy_neworder( bp );
#endif

                neworder_data[4].x.dptr = bp-
>newout.c_last;
                neworder_data[5].x.dptr = bp-
>newout.c_credit;
                sprintf( neworder_data[7].x.data,
"%d", bp->newout.o_id );

                if ( bp->newout.terror == NOERR )
                {
                        int             cnt = bp-
>newout.o_ol_cnt;

                        convert_datetime(
neworder_data[2].x.data,
                                bp-
>newout.o_entry_d );
                        sprintf(
neworder_data[6].x.data, "%d",
                                roundup( bp-
>newout.c_discount * 10000.0 ) );
                        sprintf(
neworder_data[8].x.data, "%d", cnt );
                        sprintf(
neworder_data[9].x.data, "%d",
                                roundup( bp-
>newout.w_tax * 10000.0 ) );
                        sprintf(
neworder_data[10].x.data, "%d",
                                roundup( bp-
>newout.d_tax * 10000.0 ) );

                        for ( i = 0; i < cnt; i++ )
                        {

                                debug3( (
stderr,

                "neworder(%d) :"
                                                "
i_name = %s, s_quantity = %d,"
                                                "
brand_generic = %c, i_price = %d,"
                                                "
ol_amount = %d\n",
                                                i,

                bp->newout.i_name[i], bp-
>newout.s_quantity[i],

                bp->newout.brand_generic[i],

                roundup( bp->newout.i_price[i] * 100.0
),

                roundup( bp->newout.ol_amount[i] *
100.0 ) ) );


                neworder_data[neworderl( i )+2].x.dptr
=
                bp->newout.i_name[i];
                                sprintf(
neworder_data[neworderl( i )+4].x.data, "%d",

                bp->newout.s_quantity[i] );
                                sprintf(
neworder_data[neworderl( i )+5].x.data, "%c",

                bp->newout.brand_generic[i] );
                                sprintf(
neworder_data[neworderl( i )+6].x.data, "%d",

                roundup( bp->newout.i_price[i] * 100.0
) );
                                sprintf(
neworder_data[neworderl( i )+7].x.data, "%d",

                roundup( bp->newout.ol_amount[i] *
100.0 ) );
                        }
                        sprintf(
neworder_data[132].x.data, "%d",
                                roundup( bp-
>newout.total_amount * 100.0 ) );

                        /* "Item number i s not
valid" or "" ('\0') */

                neworder_data[131].x.dptr = bp-
>newout.status;
                }
                else
                {
                        if ( bp->newout.terror ==
IRRECERR )
                        {
                                fatalerror(
"Irrecoverable error in NewOrder\n" );
                        }
                        else
                        {
                                debug( (
stderr, "terror : %d\n", \

                bp->newout.terror ) );
                                goto
resend_neworder;                /* error */
                        }
                }

                display_fields( win,
FR_FULLSCREEN, neworder_text,
neworder_data );

}

/*
                connect/close to tuxedo server
*/

void init_tux()
{

#ifndef SCRTEST
                if ( tpinit( NULL ) == -1 )
                {
                        debug( ( stderr, "Error :
%d\n", tperrno ) );
                        fprintf( stderr, "Failed to
join the application.\n" );
                        exit( 1 );
                }

                if ( ( trans_buf =
                        (void *)tpalloc(
"CARRAY", NULL, trans_size ) )
                        == NULL )
                {
                        fprintf( stderr, "Tpalloc
failed.\n" );
                        exit( 1 );
                }
#else
                if ( ( trans_buf = (void *)malloc(
trans_size ) ) == NULL )
                {
                        fprintf( stderr, "Malloc
failed.\n" );
                        exit( 1 );
                }
#endif
                memset( trans_buf, 0, ( size_t
)trans_size );

}

void clean_tux()
{
#ifndef SCRTEST
                tpterm();
#endif
}

/* Close screen and print the fatal error message
to stderr */

void fatalerror( char *msg )
{
                FILE            *err;
                char            path[32];
                time_t          t;

                clean_tux();
                closeScreen();

                sprintf( path, "/tmp/tcerror.%05d",
(w_id-1)*10+d_id );
                if ( ( err = fopen( path, "a+" ) ) !=
NULL ) {
                        time( &t );
                        fprintf( err, "%s(W:%d
D:%d SVC:%d PID:%d)\n",
                                        ctime( &t ),
w_id, d_id, svrnum, getpid() );
                        fprintf( err, msg );
                        fclose( err );
                }
                exit( -1 );
}

void interrupt( int sig )
{
                if ( sig == SIGHUP ) {
                /* in.telnetd send SIGHUP */
```

```
                        check(      char
        buf[1024];
                                    sprintf( buf,
"Signal is received : sig = %d at %s\n",

sig, CP );
                                    fatalerror( buf
);
                        )
                        exit( -10 );
        } else {
                        char        buf[1024];

                        sprintf( buf, "Signal is
received : sig = %d\n", sig );
                        check( sprintf( buf,
"Signal is received : sig = %d at %s\n",
                                    sig, CP ); )
                        fatalerror( buf );
        }
}

/*
        main function
*/

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

        check( CP = "main"; )
        if ( argc < 2 )
        {
                        fprintf( stderr, "Argument
error!\n" );
                        exit( 1 );
        }

        clone = atoi( argv[1] );

        w_id = (clone-1)/10 + 1;
        d_id = (clone-1)%10 + 1;

#ifdef DEBUG
        {
                        char        buf[32];

                        sprintf( buf,
"/tmp/tcheck.%05d", (w_id-1)*10+d_id );
                        freopen( buf, "w", stderr );
                        setvbuf( stderr, NULL,
_IOLBF, 0 );
        }
#endif

        srand48( getpid() );

        signal( SIGHUP, interrupt );
        signal( SIGINT, interrupt );
        signal( SIGTERM, interrupt );

        init_tux();
#ifndef     DEBUG
        fclose( stderr );
#endif
        TPCframe(clone);

        clean_tux();

        exit( 0 );
}



****************************************
            cwalib.h
****************************************


/*
        cwalib.h :

        Version     1.00        1996/12/26
*/

#ifndef _CWALIB_H_
#define _CWALIB_H_

#include <stdio.h>
#include <errno.h>

#if !defined( TRUE ) && !defined( FALSE ) &&
!defined( BOOL )
typedef enum
{
        FALSE, TRUE
} bool;
#define BOOL
#endif

#ifdef  DEBUG
# define    debug( s )      fprintf s
# define    check( s )      s
# if ( DEBUG >= 10 )
#  define    debug1( s )     fprintf s
#  if ( DEBUG >= 20 )
#   define    debug2( s )     fprintf s
#   if ( DEBUG >= 30 )
#    define    debug3( s )     fprintf s
#    if ( DEBUG >= 40 )
#     define    debug4( s )     fprintf s
#    endif
#   endif
#  endif
# endif
#endif

#ifndef debug
# define    debug( s )
# define    check( s )
#endif
#ifndef debug1
# define    debug1( s )
#endif
#ifndef debug2
# define    debug2( s )
#endif
#ifndef debug3
# define    debug3( s )
#endif
#ifndef debug4
# define    debug4( s )
#endif

#endif      /* _CWALIB_H_ */



****************************************
            dummy.c
****************************************


/*
        dummy.c : functions for test.

        Version     1.00        1996/12/26


        (C)Fujitsu Limited. 1994, 1995, 1996
*/

#ifdef      SCRTEST

#include                <stdio.h>
#include                <stdlib.h>
#include                <string.h>
#include                <sys/types.h>
#include                <time.h>
#include                "cwalib.h"
#include                "tpcc_info.h"
#include                "tpcc.h"

void        dummy_delivery( struct delstruct * );
void        dummy_stocklvl( struct stostruct * );
void        dummy_orderstat( struct ordstruct * );
void        dummy_payment( struct  paystruct * );
void        dummy_neworder( struct newstruct *
);
char        *get_datetimestr( char * );
char        *get_datestr( char * );

char *get_datetimestr( char *buf )
{
        struct tm       *tm;
        time_t                  tim;

        time( &tim );
        tm  = localtime( &tim );

        sprintf( buf, "%2d-%2d-
%4d.%2d:%2d:%2d", tm->tm_mday, tm-
>tm_mon+1,
                        tm->tm_year+1900, tm-
>tm_hour, tm->tm_min, tm->tm_sec );

        return buf;
}

char *get_datestr( char *buf )
{
        struct tm       *tm;
        time_t                  tim;

        time( &tim );
        tm  = localtime( &tim );

        sprintf( buf, "%2d-%2d-%4d",
```

```
                        tm->tm_mday, tm->tm_mon+1, tm->tm_year+1900 );
        return buf;
}

void dummy_delivery( struct delstruct *bp )
{
        bp->delout.terror = NOERR;

        return;
}

void dummy_stocklvl( struct stostruct *bp )
{
        int             i;

        bp->stoout.terror = NOERR;
        do
        {
                i = rand()%1000;
        } while ( i > bp->stoin.threshold );

        bp->stoout.low_stock = i;

        return;
}

void dummy_payment( struct paystruct *bp )
{
        bp->payout.terror = NOERR;

        get_datetimestr( bp->payout.h_date );
        strcpy( bp->payout.w_street_1, "Baker street" );
        strcpy( bp->payout.w_street_2, "221B" );
        strcpy( bp->payout.w_city, "London" );
        strcpy( bp->payout.w_state, "GB" );
        strcpy( bp->payout.w_zip, "88033000" );

        strcpy( bp->payout.d_street_1, "Minato-ku" );
        strcpy( bp->payout.d_street_2, "Azabu 10" );
        strcpy( bp->payout.d_city, "Tokyo" );
        strcpy( bp->payout.d_state, "JP" );
        strcpy( bp->payout.d_zip, "102" );

        bp->payout.c_id = 777;
        strcpy( bp->payout.c_first, "John" );
        strcpy( bp->payout.c_middle, "H" );
        strcpy( bp->payout.c_last, "Watson" );
        strcpy( bp->payout.c_street_1, "Baker street" );
        strcpy( bp->payout.c_street_2, "221B" );
        strcpy( bp->payout.c_credit, "GC" );
        bp->payout.c_discount = 0.20;
        strcpy( bp->payout.c_city, "London" );
        strcpy( bp->payout.c_state, "GB" );
        strcpy( bp->payout.c_zip, "888" );
        strcpy( bp->payout.c_phone, "" );
        bp->payout.c_balance = 67876;
        bp->payout.c_credit_lim = 77777;
        get_datestr( bp->payout.c_since );
        strcpy( bp->payout.c_data, "Migyamigyamigyamigyamigya"

                "migyamigyamigyamigyamigya" );

        return;
}

void dummy_orderstat( struct ordstruct *bp )
{
        int             i, j;

        bp->ordout.terror = NOERR;

        bp->ordout.c_id = rand()%10000;
        strcpy( bp->ordout.c_first, "Robert" );
        strcpy( bp->ordout.c_middle, "L" );
        strcpy( bp->ordout.c_last, "Fish" );
        bp->ordout.c_balance = ( ( rand()*rand()%19999999 )-9999999 ) / 100.0;
        /*
        fprintf( stderr, "ordout.c_balance = %12.4f\n", bp->ordout.c_balance );
        bp->c_balance = -1;
        */

        bp->ordout.o_id = rand()%10000;
        get_datetimestr( bp->ordout.o_entry_d );
        bp->ordout.o_carrier_id = rand()%100;

        bp->ordout.o_ol_cnt = ( rand()%11 )+5;
        j = bp->ordout.o_ol_cnt;
        for ( i = 0; i < j; i++ )
        {
                bp->ordout.ol_supply_w_id[i] = ( rand()%10 )+1;
                bp->ordout.ol_i_id[i] = ( rand()%100000 )+1;
                bp->ordout.ol_quantity[i] = ( rand()%99 )+1;
                bp->ordout.ol_amount[i] = rand();

                debug2( ( stderr, "rand : %f\n", bp->ordout.ol_amount[i] ) );

                get_datetimestr( bp->ordout.ol_delivery_d[i] );
        }

        return;
}

void dummy_neworder( struct newstruct *bp )
{
        static int      o_id = 3001;
        int i;

        bp->newout.terror = NOERR;
        *( bp->newout.status ) = '\0';

        strcpy( bp->newout.c_last, "Holmes" );
        strcpy( bp->newout.c_credit, "GC" );
        bp->newout.o_id = o_id++;
/*
        bp->newout.o_id = ( rand()%100000 )+1;
*/

        get_datetimestr( bp->newout.o_entry_d );
        bp->newout.c_discount = ( rand()%101 )/10000.0;
        bp->newout.w_tax = ( rand()%2001 )/10000.0;
        bp->newout.d_tax = ( rand()%2001 )/10000.0;

        bp->newout.total_amount = 0;

        for ( i = 0; i < 15; i++ )
        {
                if ( bp->newin.ol_supply_w_id[i] == 0 ) {
                        break;
                }
                if ( bp->newin.ol_i_id[i] == -1 ) {
                        strcpy( bp->newout.status, "Item number is not valid" );
                }

                bp->newout.i_name[i][0] = '\0';
                bp->newout.s_quantity[i] = ( rand()%10 )+1;
                bp->newout.brand_generic[i] = ( rand()%26 )+'A';
                bp->newout.i_price[i] = (( rand()%10000 )+1 )/100.0;
                bp->newout.ol_amount[i] = bp->newout.i_price[i]*bp->newin.ol_quantity[i];
                bp->newout.total_amount += bp->newout.ol_amount[i];
        }
        bp->newout.o_ol_cnt = i;

        return;
}

#endif




****************************************
        frame.c
****************************************


/*
        frame.c : module to define user-interface data structures.

        Version     1.00        1996/12/26
*/

#include "ui.h"
#include "frame.h"

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

```
TextField delivery_text[] =
{
        { 0, 37, "Delivery" },
        { 1, 0, "Warehouse:" },
        { 3, 0, "Carrier Number:" },
        { 5, 0, "Execution Status:" },
        { -1, -1, 0 }
};

DataField delivery_data[] =
{
        { 1, 11, "", F_RO|F_NE|F_RJ, "nnnn",
0, 0 },
        { 3, 16, "", F_START|F_RJ, "nn", 0, 0
},
        { 5, 18, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },
        { -1, -1, 0, 0, 0, 0, 0 }
};

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

TextField stocklvl_text[] =
{
        { 0, 34, "Stock-Level" },
        { 1, 0, "Warehouse:" },
        { 1, 18, "District:" },
        { 3, 0, "Stock Level Threshold:" },
        { 5, 0, "low stock:" },
        { -1, -1, 0 }
};

DataField stocklvl_data[] =
{
        { 1, 11, "", F_RO|F_NE|F_RJ, "nnnn",
0, 0 },
        { 1, 28, "", F_RO|F_NE|F_RJ, "nn", 0,
0 },
        { 3, 23, "", F_START|F_RJ, "n9", 0, 0
},
        { 5, 11, "", F_RO|F_ NE|F_RJ, "nnn",
0, 0 },
        { -1, -1, 0, 0, 0, 0, 0 }
};

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

TextField payment_text[] =
{
        { 0, 37, "Payment" },
        { 1, 0, "Date:" },
        { 3, 0, "Warehouse:" },
        { 3, 41, "District:" },

        { 8, 0, "Customer:" },
        { 8, 16, "Cust-Warehouse:" },
        { 8, 38, "Cust-District:" },
        { 9, 0, "Name:" },
        { 9, 49, "Since:" },
        { 10, 49, "Credit:" },
        { 11, 49, "%Disc:" },
        { 12, 49, "Phone:" },
        { 14, 0, "Amount Paid:" },
        { 14, 36, "New Cust-Balance:" },
        { 15, 0, "Credit Limit:" },

        { 17, 0, "Cust-Data:" },
        { -1, -1, 0 }
};

DataField payment_data[] =
{
        { 1, 6, "", F_RO|F_NE, "X9-X9-9999
n9:99:99", 0, 0 },

                                /* H_DATE */
        { 3, 11, "", F_RO|F_NE|F_RJ, "nnnn",
0, 0 },

                                /* W_ID */
        { 3, 51, "", F_START|F_RJ, "nn", 0, 0
},

                                /* D_ID */
        { 4, 0, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /*
W_STREET_1 */
        { 4, 41, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /*
D_STREET_1 */
        { 5, 0, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /*
W_STREET_2 */
        { 5, 41, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /*
D_STREET_2 */
        { 6, 0, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /* W_CITY */
        { 6, 21, "", F_RO|F_NE|F_PTR, "XX",
0, 0 },

                                /* W_STATE
*/
        { 6, 24, "", F_RO|F_NE|F_PTR,
"XXXXX-XXXX", 0, 0 },

                                /* W_ZIP */
        { 6, 41, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /* D_CITY */
        { 6, 62, "", F_RO|F_NE|F_PTR, "XX",
0, 0 },

                                /* D_STATE
*/
        { 6, 65, "", F_RO|F_NE|F_PTR,
"XXXXX-XXXX", 0, 0 },

                                /* D_ZIP */

        { 8, 10, "", F_RJ, "nnnn", 0, 0 },

                                /* C_ID */
        { 8, 32, "", F_RJ, "nnnn", 0, 0 },

                                /* C_W_ID */
        { 8, 53, "", F_RJ, "nn", 0, 0 },

                                /* C_D_ID */
        { 9, 8, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXX", 0, 0 },

                                /* C_FIRST
*/
        { 9, 25, "", F_RO|F_NE|F_PTR, "XX",
0, 0 },

                                /*
C_MIDDLE */
        { 9, 28, "", F_PTR,
"XXXXXXXXXXXXXXXX", 0, 0 },

                                /* C_LAST */
        { 9, 57, "", F_RO|F_NE, "X9-X9-
9999", 0, 0 },

                                /* C_SINCE
*/
        { 10, 8, "", F_RO|F_NE|F_P TR,
"XXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /*
C_STREET_1 */
        { 10, 57, "",
F_RO|F_NE|F_PTR|F_RJ, "XX", 0, 0 },

                                /* C_CREDIT
*/
        { 11, 8, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /*
C_STREET_2 */
        { 11, 57, "", F_RO|F_NE|F_RJ,
"n9.99", 0, 0 },

                                /*
C_DISCOUNT */
        { 12, 8, "", F_RO|F_NE|F_PTR,
"XXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /* C_CITY */
        { 12, 29, "", F_RO|F_NE|F_PTR,
"XX", 0, 0 },

                                /* C_STATE
*/
        { 12, 32, "", F_RO|F_NE|F_PTR,
"XXXXX-XXXX", 0, 0 },

                                /* C_ZIP */
        { 12, 57, "", F_NE|F_RO|F_PTR,
"XXXXXX-XXX-XXX-XXXX", 0, 0 },

                                /* C_PHONE
*/
        { 14, 22, "", F_RJ, "$nnn9.99", 0, 0 },

                                /*
H_AMOUNT */
        { 14, 54, "", F_RO|F_NE|F_RJ,
"$Mnnnnnnnn9.99", 0, 0 },
```

```
                                        /*
C_BALANCE */
                { 15, 16, "", F_RO|F_NE|F_RJ,
"$nnnnnnnnn9.99", 0, 0 },

                                        /*
C_CREDIT_LIM */

                { 17, 11, "", F_RO|F_NE,

                "XXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },
                { 18, 11, "", F_RO|F_NE,

                "XXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },
                { 19, 11, "", F_RO|F_NE,

                "XXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },
                { 20, 11, "", F_RO|F_NE,

                "XXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },
                { -1, -1, 0, 0, 0, 0, 0 }
};

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

TextField orderstat_text[] =
{
                { 0, 34, "Order-Status" },
                { 1, 0, "Warehouse:" },
                { 1, 18, "District:" },
                { 2, 0, "Customer:" },
                { 2, 17, "Name:" },
                { 3, 0, "Cust-Balance:" },
                { 5, 0, "Order-Number:" },
                { 5, 25, "Entry-Date:" },
                { 5, 59, "Carrier-Number:" },

                { 6, 0, "Supply-W" },
                { 6, 13, "Item-Id" },
                { 6, 24, "Qty" },
                { 6, 32, "Amount" },
                { 6, 44, "Delivery-Date" },
                { -1, -1, 0 }
};

DataField orderstat_data[] =
{
                { 1, 11, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                        /* W_ID */
                { 1, 28, "", F_START|F_RJ, "nn", 0, 0
},

                                        /* D_ID */
                { 2, 10, "", F_RJ, "nnnn", 0, 0 },

                                        /* C_ID */
                { 2, 23, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXX", 0, 0 },

                                        /* C_FIRST
*/
                { 2, 40, "", F_NE|F_RO|F_PTR, "XX",
0, 0 },

                                        /*
C_MIDDLE */
                { 2, 43, "", F_PTR,
"XXXXXXXXXXXXXXX", 0, 0 },

                                        /* C_LAST */
                { 3, 14, "", F_NE|F_RO|F_RJ,
"$Mnnnnnn9.99", 0, 0 },

                                        /*
C_BALANCE */

                { 5, 14, "", F_NE|F_RO|F_RJ,
"nnnnnnnn", 0, 0 },

                                        /* O_ID */
                { 5, 37, "", F_NE|F_RO, "X9-X9-9999
X9:99:99", 0, 0 },

                                        /*
O_ENTRY_D */
                { 5, 75, "", F_NE|F_RO|F_RJ, "nn", 0,
0 },

                                        /*
O_CARRIER_ID */

                { 7, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                        /*
OL_SUPPLY_W_ID_1 */
                { 7, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                        /* OL_I_ID_1
*/
                { 7, 24, "", F_NE|F_RO|F_RJ , "n9", 0,
0 },

                                        /*
OL_QUANTITY */
                { 7, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                        /*
OL_AMOUNT */
                { 7, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                        /*
OL_DELIVERY_D_1 */

                { 8, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                        /*
OL_SUPPLY_W_ID_2 */
                { 8, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                        /*
OL_L_ID_2 */
                { 8, 24, "", F_NE|F_RO|F_RJ, "n9", 0,
0 },

                                        /*
OL_QUANTITY_2 */
                { 8, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_2 */
                { 8, 46, "", F_NE|F_RO, "X 9-X9-
9999", 0, 0 },

                                        /*
OL_DELIVERY_D_2 */

                { 9, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                        /*
OL_SUPPLY_W_ID_3 */
                { 9, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                        /* OL_I_ID_1
*/
                { 9, 24, "", F_NE|F_RO|F_RJ, "n9", 0,
0 },

                                        /*
OL_QUANTITY_3 */
                { 9, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_3 */
                { 9, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                        /*
OL_DELIVERY_D_3 */

                { 10, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                        /*
OL_SUPPLY_W_ID_4 */
                { 10, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                        /*
OL_L_ID_4 */
                { 10, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                        /*
OL_QUANTITY_4 */
                { 10, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_4 */
                { 10, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                        /*
OL_DELIVERY_D_4 */

                { 11, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },
```

```
                                /*
OL_SUPPLY_W_ID_5 */
                { 11, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                /* OL_I_ID_5
*/
                { 11, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                /*
OL_QUANTITY_5 */
                { 11, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                /*
OL_AMOUNT_5 */
                { 11, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                /*
OL_DELIVERY_D_5 */

                { 12, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                /*
OL_SUPPLY_W_ID_6 */
                { 12, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                /*
OL_L_ID_6 */
                { 12, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                /*
OL_QUANTITY_6 */
                { 12, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                /*
OL_AMOUNT_6 */
                { 12, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                /*
OL_DELIVERY_D_6 */

                { 13, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                /*
OL_SUPPLY_W_ID_7 */
                { 13, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                /* OL_I_ID_7
*/
                { 13, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                /*
OL_QUANTITY_7 */
                { 13, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                /*
OL_AMOUNT_7 */
                { 13, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                /*
OL_DELIVERY_D_7 */

                { 14, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                /*
OL_SUPPLY_W_ID_8 */
                { 14, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                /*
OL_L_ID_8 */
                { 14, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                /*
OL_QUANTITY_8 */
                { 14, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                /*
OL_AMOUNT_8 */
                { 14, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                /*
OL_DELIVERY_D_8 */

                { 15, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                /*
OL_SUPPLY_W_ID_9 */
                { 15, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                /* OL_I_ID_9
*/
                { 15, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                /*
OL_QUANTITY_9 */
                { 15, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                /*
OL_AMOUNT_9 */
                { 15, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                /*
OL_DELIVERY_D_9 */

                { 16, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                /*
OL_SUPPLY_W_ID_10 */
                { 16, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                /*
OL_L_ID_10 */
                { 16, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                /*
OL_QUANTITY_10 */
                { 16, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                /*
OL_AMOUNT_10 */
                { 16, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                /*
OL_DELIVERY_D_10 */

                { 17, 2, "", F_NE|F_R O|F_RJ, "nnnn",
0, 0 },

                                /*
OL_SUPPLY_W_ID_11 */
                { 17, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                /*
OL_I_ID_11 */
                { 17, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                /*
OL_QUANTITY_11 */
                { 17, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                /*
OL_AMOUNT_11 */
                { 17, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                /*
OL_DELIVERY_D_11 */

                { 18, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                /*
OL_SUPPLY_W_ID_12 */
                { 18, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                /*
OL_L_ID_12 */
                { 18, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                /*
OL_QUANTITY_12 */
                { 18, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                /*
OL_AMOUNT_12 */
                { 18, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },
```

```
                                        /*
OL_DELIVERY_D_12 */

                { 19, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                        /*
OL_SUPPLY_W_ID_13 */
                { 19, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                        /*
OL_L_ID_13 */
                { 19, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                        /*
OL_QUANTITY_13 */
                { 19, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_13 */
                { 19, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                        /*
OL_DELIVERY_D_13 */

                { 20, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                        /*
OL_SUPPLY_W_ID_14 */
                { 20, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                        /*
OL_I_ID_14 */
                { 20, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                        /*
OL_QUANTITY14 */
                { 20, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_14 */
                { 20, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                        /*
OL_DELIVERY_D_14 */

                { 21, 2, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                        /*
OL_SUPPLY_W_ID_15 */
                { 21, 13, "", F_NE|F_RO|F_RJ,
"nnnnnn", 0, 0 },

                                        /*
OL_L_ID_15 */
                { 21, 24, "", F_NE|F_RO|F_RJ, "n9",
0, 0 },

                                        /*
OL_QUANTITY_15 */
                { 21, 31, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_15 */
                { 21, 46, "", F_NE|F_RO, "X9-X9-
9999", 0, 0 },

                                        /*
OL_DELIVERY_D_15 */
                { -1, -1, 0, 0, 0, 0, 0 }
};

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

TextField neworder_text[]=
{
                { 0, 35, "New Order" },
                { 1, 0, "Warehouse:" },
                { 1, 18, "District: " },
                { 1, 54, "Date:" },
                { 2, 0, "Customer:" },
                { 2, 18, "Name:" },
                { 2, 43, "Credit:" },
                { 2, 56, "%Disc:" },
                { 3, 0, "Order number:" },
                { 3, 24, "Number of Lines:" },
                { 3, 51, "W_tax:" },
                { 3, 66, "D_tax:" },

                { 5, 1, "Supp_W" },
                { 5, 9, "Item_id" },
                { 5, 18, "Item Name" },
                { 5, 44, "Qty" },
                { 5, 49, "Stock" },
                { 5, 56, "B/G" },
                { 5, 61, "Price" },
                { 5, 70, "Amount" },

                { 21, 0, "Execution Status:" },
                { 21, 61, "Total:" },
                { -1, -1, 0 }
};

DataField neworder_data[] =
{
                { 1, 11, "", F_NE|F_RO|F_RJ, "nnnn",
0, 0 },

                                        /* W_ID */
                { 1, 28, "", F_START|F_RJ, "nn", 0, 0
},

                                        /* D_ID */
                { 1, 60, "", F_NE|F_RO, "X9-X9-9999
n9:99:99", 0, 0 },

                                        /*
O_ENTRY_D */
                { 2, 11, "", F_RJ, "nnnn", 0, 0 },

                                        /* C_ID */
                { 2, 24, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXX", 0, 0 },

                                        /* C_LAST */
                { 2, 51, "", F_NE|F_RO|F_PTR|F_RJ,
"XX", 0, 0 },

                                        /* C_CREDIT
*/
                { 2, 63, "", F_NE|F_RO|F_RJ|F_RJ,
"n9.99", 0, 0 },

                                        /*
C_DISCOUNT */
                { 3, 14, "", F_NE|F_RO|F_RJ,
"nnnnnnnn", 0, 0 },

                                        /* O_ID */
                { 3, 41, "", F_NE|F_RO|F_RJ, "n9", 0,
0 },

                                        /*
O_OL_CNT */
                { 3, 58, "", F_NE|F_RO|F_RJ, "n9.99",
0, 0 },

                                        /* W_TAX */
                { 3, 73, "", F_NE|F_RO|F_RJ, "n9.99",
0, 0 },

                                        /* D_TAX */

                { 6, 2, "", F_RJ, "nnnn", 0, 0 },

                                        /*
OL_SUPPLY_W_ID_1 */
                { 6, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                        /* OL_I_ID_1
*/
                { 6, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /* I_NAME_1
*/
                { 6, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_1 */
                { 6, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                        /*
S_QUANTITY_1 */
                { 6, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_1 */
                { 6, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /* I_PRICE_1
*/
                { 6, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_1 */
```

```
        { 7, 2, "", F_RJ, "nnnn", 0, 0 },

                                /*
OL_SUPPLY_W_ID_2 */
        { 7, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                /* OL_I_ID_2
*/
        { 7, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /* I_NAME_2
*/
        { 7, 44, "", F_RJ,  "n9", 0, 0 },

                                /*
OL_QUANTITY_2 */
        { 7, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                /*
S_QUANTITY_2 */
        { 7, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                /*
BRAND_GENERIC_2 */
        { 7, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                /* I_PRICE_2
*/
        { 7, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                /*
OL_AMOUNT_2 */

        { 8, 2, "", F_RJ, "nnnn", 0, 0 },

                                /*
OL_SUPPLY_W_ID_3 */
        { 8, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                /* OL_I_ID_3
*/
        { 8, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /* I_NAME_3
*/
        { 8, 44, "", F_RJ, "n9", 0, 0 },

                                /*
OL_QUANTITY_3 */
        { 8, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                /*
S_QUANTITY_3 */
        { 8, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                /*
BRAND_GENERIC_3 */
        { 8, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                /* I_PRICE_3
*/
        { 8, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                /*
OL_AMOUNT_3 */

        { 9, 2, "", F_RJ, "nnnn", 0, 0 },

                                /*
OL_SUPPLY_W_ID_4 */
        { 9, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                /* OL_I_ID_4
*/
        { 9, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /* I_NAME_4
*/
        { 9, 44, "", F_RJ, "n9", 0, 0 },

                                /*
OL_QUANTITY_4 */
        { 9, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                /*
S_QUANTITY_4 */
        { 9, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                /*
BRAND_GENERIC_4 */
        { 9, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                /* I_PRICE_4
*/
        { 9, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                /*
OL_AMOUNT_4 */

        { 10, 2, "", F_RJ, "nnnn", 0, 0 },

                                /*
OL_SUPPLY_W_ID_5 */
        { 10, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                /* OL_I_ID_5
*/
        { 10, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /* I_NAME_5
*/
        { 10, 44, "", F_RJ, "n9", 0, 0 },

                                /*
OL_QUANTITY_5 */
        { 10, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                /*
S_QUANTITY_5 */
        { 10, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                /*
BRAND_GENERIC_5 */
        { 10, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                /* I_PRICE_5
*/
        { 10, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                /*
OL_AMOUNT_5 */

        { 11, 2, "", F_RJ, "nnnn", 0, 0 },

                                /*
OL_SUPPLY_W_ID_6 */
        { 11, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                /* OL_I_ID_6
*/
        { 11, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                /* I_NAME_6
*/
        { 11, 44, "", F_RJ, "n9", 0, 0 },

                                /*
OL_QUANTITY_6 */
        { 11, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                /*
S_QUANTITY_6 */
        { 11, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                /*
BRAND_GENERIC_6 */
        { 11, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                /* I_PRICE_6
*/
        { 11, 70, "", F_NE|F_RO| F_RJ,
"$nnn9.99", 0, 0 },

                                /*
OL_AMOUNT_6 */

        { 12, 2, "", F_RJ, "nnnn", 0, 0 },

                                /*
OL_SUPPLY_W_ID_7 */
        { 12, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                /* OL_I_ID_7
*/
        { 12, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },
```

```
                                        /* I_NAME_7
*/
                { 12, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_7 */
                { 12, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                        /*
S_QUANTITY_7 */
                { 12, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_7 */
                { 12, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /* I_PRICE_7
*/
                { 12, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_7 */

                { 13, 2, "", F_RJ, "nnnn", 0, 0 },

                                        /*
OL_SUPPLY_W_ID_8 */
                { 13, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                        /* OL_I_ID_8
*/
                { 13, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /* I_NAME_8
*/
                { 13, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_8 */
                { 13, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                        /*
S_QUANTITY_8 */
                { 13, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_8 */
                { 13, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /* I_PRICE_8
*/
                { 13, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_8 */

                { 14, 2, "", F_RJ, "nnnn", 0, 0 },

                                        /*
OL_SUPPLY_W_ID_9 */
                { 14, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                        /* OL_I_ID_9
*/
                { 14, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /* I_NAME_9
*/
                { 14, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_9 */
                { 14, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                        /*
S_QUANTITY_9 */
                { 14, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_9 */
                { 14, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /* I_PRICE_9
*/
                { 14, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_9 */

                { 15, 2, "", F_RJ, "nnnn", 0, 0 },

                                        /*
OL_SUPPLY_W_ID_10 */
                { 15, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                        /*
OL_I_ID_10 */
                { 15, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /*
I_NAME_10 */
                { 15, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_10 */
                { 15, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                        /*
S_QUANTITY_10 */
                { 15, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_10 */
                { 15, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /*
I_PRICE_10 */
                { 15, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_10 */

                { 16, 2, "", F_RJ, "nnnn", 0, 0 },

                                        /*
OL_SUPPLY_W_ID_11 */
                { 16, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                        /*
OL_I_ID_11 */
                { 16, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /*
I_NAME_11 */
                { 16, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_11 */
                { 16, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                        /*
S_QUANTITY_11 */
                { 16, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_11 */
                { 16, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /*
I_PRICE_11 */
                { 16, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_11 */

                { 17, 2, "", F_RJ, "nnnn", 0, 0 },

                                        /*
OL_SUPPLY_W_ID_12 */
                { 17, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                        /*
OL_I_ID_12 */
                { 17, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /*
I_NAME_12 */
                { 17, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_12 */
                { 17, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },
```

```
                                        /*
S_QUANTITY_12 */
                { 17, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_12 */
                { 17, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /*
I_PRICE_12 */
                { 17, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_12 */

                { 18, 2, "", F_RJ, "nnnn", 0, 0 },

                                        /*
OL_SUPPLY_W_ID_13 */
                { 18, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                        /*
OL_I_ID_13 */
                { 18, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /*
I_NAME_13 */
                { 18, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_13 */
                { 18, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                        /*
S_QUANTITY_13 */
                { 18, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_13 */
                { 18, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /*
I_PRICE_13 */
                { 18, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_13 */

                { 19, 2, "", F_RJ, "nnnn", 0, 0 },

                                        /*
OL_SUPPLY_W_ID_14 */
                { 19, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                        /*
OL_I_ID_14 */
                { 19, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /*
I_NAME_14 */
                { 19, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_14 */
                { 19, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                        /*
S_QUANTITY_14 */
                { 19, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_14 */
                { 19, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /*
I_PRICE_14 */
                { 19, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_14 */

                { 20, 2, "", F_RJ, "nnnn", 0, 0 },

                                        /*
OL_SUPPLY_W_ID_15 */
                { 20, 9, "", F_RJ, "nnnnnn", 0, 0 },

                                        /*
OL_I_ID_15 */
                { 20, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /*
I_NAME_15 */
                { 20, 44, "", F_RJ, "n9", 0, 0 },

                                        /*
OL_QUANTITY_15 */
                { 20, 50, "", F_NE|F_RO|F_RJ, "nn9",
0, 0 },

                                        /*
S_QUANTITY_15 */
                { 20, 57, "", F_NE|F_RO|F_RJ, "X", 0,
0 },

                                        /*
BRAND_GENERIC_15 */
                { 20, 61, "", F_NE|F_RO|F_RJ,
"$nn9.99", 0, 0 },

                                        /*
I_PRICE_15 */
                { 20, 70, "", F_NE|F_RO|F_RJ,
"$nnn9.99", 0, 0 },

                                        /*
OL_AMOUNT_15 */

                { 21, 18, "", F_NE|F_RO|F_PTR,
"XXXXXXXXXXXXXXXXXXXXXXXX", 0, 0 },

                                        /*
EXEC_STAT */
                { 21, 69, "", F_NE|F_RO|F_RJ,
"$nnnn9.99", 0, 0 },

                                        /*
TOTAL_AMOUNT */
                { -1, -1, 0, 0, 0, 0, 0 }
};




****************************************
        frame.h
****************************************


/*
        frame.h :

        Version 1.00    1996/12/26
*/

#ifndef _FRAME_H_
#define _FRAME_H_

#define FIELDMAXSIZ             63

typedef enum
{
        Eos, Text, Data
} Ftypes;

typedef struct
{
        int             row;
        int             col;
        char            text[FIELDMAXSIZ+1];
} TextField;

typedef struct
{
        int             row;
        int             col;

        union
        {
                char
        data[FIELDMAXSIZ+1];
                char    *dptr;
        } x;

        int             type;
        char            fmt[FIELDMAXSIZ+1];
        void            (*actionf)(int);
        void            (*valuef)(int);
} DataField;

#define FR_FULLSCREEN           1
#define FR_RETRY        2
#define FR_OPERATION            4

#define eos( x )        ( x.row == -1 )

/*
```

```
#define     clrp( fld )             { \
            fld.type &= ~F_PTR; \
                                    }
#define     setp( fld, str )        { \
            fld.type |= F_PTR; \
            fld.x.dptr = str; \
                                    }
#define     setpnclr( fld, str )    { \
            fld.type |= F_PTR; \
            str[0] = 0; \
            fld.x.dptr = str; \
                                    }
*/

#endif      /* _FRAME_H_ */




*****************************************
        ui.c
*****************************************


/*
            ui.c : Module for low level screen
operation

            Version     1.00        1996/12/26
*/

#include    <stdio.h>
#include    <time.h>
#ifdef      __linux__
# include   <ncurses/curses.h>
#else
# include   <curses.h>
#endif
#include    "ui.h"
#include    "cwalib.h"
#include    "frame.h"

#define                 ISCURSESKEY( k )    (
(k) >= KEY_MIN )

long        ATTR_BASE, ATTR_STATUS,
ATTR_MENU, ATTR_MENUBORDER,
ATTR_LININP,
            ATTR_LININPBORDER,
ATTR_FRAME, ATTR_FRAMEBORDER,
ATTR_RO_FIELD,
            ATTR_NE_FIELD,
ATTR_ACTION_FIELD, ATTR_NORMAL_FIELD,
ATTR_DIBOX,
            ATTR_DIBORDER,
ATTR_SCRLBOX, ATTR_SCRLBORDER;

#ifdef      DUR
extern      DataField   neworder_data;
#endif

void setup_attrs( void );
void ctrlC_handler( int );
void printfield ( WINDOW *, int, int, char *, char *,
int, int );
void display_fields( WINDOW *, int, TextField *,
DataField * );
void queryframe( WINDOW *, int, TextField *,
DataField * );

/* display a set of fields */

void display_fields( WINDOW *win, int mode,
TextField *tf, DataField *df )
{
            int         i;
            char        *data;

            debug1( ( stderr, "IN:\t
display_fields()\n" ) );

#ifdef      DUR
            if ( df == &neworder_data ) {
                        wattrset( win,
A_UNDERLINE );
                        wmove( win, 3, 9 );
                        waddstr( win, "ber:
________" );
                        /* '________' will be
rewritten in next block */
                        /* (send "<underline>ber:
xxxxxxxx</underline>") */
            }
#endif
            for ( i = 0; !eos( df[i] ); i++ )
            {
                        if ( df[i].type & F_PTR )
                        {
                                    data =
df[i].x.dptr;
                        }
                        else
                        {
                                    data =
df[i].x.data;
                        }
                        printfield( win, df[i].row,
df[i].col, df[i].fmt, data, -1,
                                    df[i].type );
            }

            wattrset( win, ATTR_BASE );
            wrefresh( win );

#ifdef      DUR
            if ( df == &neworder_data ) {
                        wmove( win, 3, 9 );
                        waddstr( win, "ber: " );
                        wrefresh( win );
                        /* '________' will be
rewritten in next block */
                        /* (send "<underline>ber:
xxxxxxxx</underline>") */
            }
#endif
            debug1( ( stderr, "OUT:\t
display_fields()\n" ) );
}

/* print a field according to the format */

void printfield( WINDOW *win, int row, int col, char
*fmt, char *data,
                                    int curp, int
flags )
{
            int         p, i, l = strlen( data );
            char        outbuf[80];
            long        attr;

            debug1( ( stderr, "IN:\t printfield()\n" )
);

            wmove( win, row, col );

            if ( flags & F_NE )
            {
                        wattrset( win,
ATTR_NE_FIELD );
            }
            else if ( flags & F_ACTION )
            {
                        wattrset( win,
ATTR_ACTION_FIELD );
            }
            else if ( flags & F_RO )
            {
                        wattrset( win,
ATTR_RO_FIELD );
            }
            else
            {
                        wattrset( win,
ATTR_NORMAL_FIELD );
            }

            if ( ( p = format( flags, fmt, data, curp,
outbuf ) ) == -1 )
            {
                        for( i = 0; i < ( int )strlen(
fmt ); i++ )
                        {
                                    waddch( win,
'*' );
                        }

                        debug( ( stderr,
"printfield(Invalid form) : %s %s -> ***\n",
                                    fmt, data ) );
            }
            else
            {
                        waddstr( win, outbuf );
            }

            if ( curp >= 0 )
            /* curp < 0 : no need for move */
            {
                        wmove( win, row, col+p );
            }

            debug1( ( stderr, "OUT:\t printfield()\n"
) );
```

```
}

#ifdef USE_OLD

/* Get a field according to the format  */

int getfield( WINDOW *win, int row, int col, char
*buf, int flags, char *fmt,
                        void (*actionfunc)( ), int
fldn )
{
        int         i, p, k, blen, dlen;
        bool        exitflag = FALSE;
        bool        valid;
        char        choices[80], buf2[80];

        debug1( ( stderr, "IN:\t getfield()\n" ) );
        debug2( ( stderr, "ARGS:\t row: %d
col: %d buf: '%s' flags: 0x%X"
                " fmt: '%s'\n", row, col,
buf, flags, fmt ) );

        blen = strlen( buf );
        p = dlen = blen;

        wmove( win, row, col );
        printfield( win, row, col, fmt, buf, p,
flags );

        while ( !exitflag )
        {
                wrefresh( win );
                k = wgetch( win );

                debug2( ( stderr,
"ARGS:\t k: %d (0x%X)\n", k, k ) );

                switch( k )
                {
                case KEY_UP :
                case KEY_DOWN :
                case ESC :
                case LF :
                case CR :
                case TAB :
                        exitflag =
TRUE;
                        break;

                case KEY_LEFT :
                        if ( ( p > 0 )
&&

        !( ( flags & F_ACTION ) || ( flags &
F_RO ) ) )
                        {

        p--;
                        }
                        break;

                case KEY_RIGHT :
                        if ( ( p < blen
) &&

        !( ( flags & F_ACTION ) || ( flags &
F_RO ) ) )
                        {

        p++;
                        }
                        break;

                /* case TERMBS : */
                case BS :
                case KEY_BACKSPACE
:
                        if ( ( p > 0 )
&&

        !( ( flags & F_ACTION ) || ( flags &
F_RO ) ) )
                        {

        p--;

        for ( i = p; buf[i] != 0; i++ )
                                {

        buf[i] = buf[i+1];
                                }

        blen--;
                        }
                        break;

                /* case CONSOLEBS : */
                case DEL :
                        if ( ( p >= 0 )
&&

        !( ( flags & F_ACTION ) || ( flags &
F_RO ) ) )
                        {

        for ( i = p; buf[i] != 0; i++ )
                                {

        buf[i] = buf[i+1];
                                }

        blen--;
                        }
                        break;

                default :
                        if ( ( flags &
F_ACTION ) && ( k == ' ' ) )
                        {

        (*actionfunc)( fldn );
                        }
                        else
                        {
                                if
( isprint( k ) && !( flags & F_RO ) )
                                {
                        /* insert a character */

        for( i = blen; i >= p ; i-- )

        {

                buf[i+1] = buf[i];

        }

        buf[p++] = ( char )k;

        blen++;


        if ( format( flags, fmt, buf, p, buf2 )

                == -1 )

        {

                /* delete invalid
characters */

                p--;


                for ( i = p; buf[i] != 0; i++ )

                {

                        buf[i] =
buf[i+1];

                }

                blen--;

        }
                                }
                        }
                        break;
                }

                printfield( win, row, col,
fmt, buf, p, flags );
        }

        k <<= 1;
        if ( p != dlen ) {
                k += 1;
        }

        debug2( ( stderr, "RTN:\t k: %d(0x%X)
flag: %d\n",
                k >> 1, k >> 1, k & 1 ) );
        debug1( ( stderr, "OUT:\t getfield()\n"
) );

        return      k;
}

/* Format data according to fmt, and send back in
out */

int format( int flags, char *fmt, char *data, int
realcurp, char *out )
{
        bool        endofdata = FALSE;
        int         i = 0;
        int         j = 0;
        int         len = strlen( fmt );
        int         curp = len;
        int         minusp = -1;

        debug1( ( stderr, "IN:\t format()\n" ) );
```

```
		debug2( ( stderr, "ARGS:\t flags:
0x%X fmt: '%s' data: '%s'"
			" realcurp: %d out:
'%s'\n",
			flags, fmt, data, realcurp,
out ) );

		if ( flags & F_RJ )
		{
			i = len-1;
			out[len] = 0;
			j = strlen( data )-1;
		}

		for( ; fmt[i] != 0; flags & F_RJ ? i-- :
i++ )
		{
			if ( ( data[j] == 0 ) || ( (
flags & F_RJ ) && j < 0 ) )
			{
				endofdata =
TRUE;
			}

			if ( ( fmt[i] == 'X' ) || (
fmt[i] == '9' )
				|| ( fmt[i] ==
'n' ) )
			{
				if (
!endofdata )
				{
					if
( ( fmt[i] == 'X' )

		|| ( ( fmt[i] == '9' )

			&& isdigit( data[j] ) )

		|| ( ( fmt[i] == 'n' )

			&& isdigit( data[j] ) ) )
						{

		if ( j == realcurp )

		{

			curp = i;

		}


		out[i] = data[flags & F_RJ ? j-- : j++];
					}

		else if ( data[j] == '-' )
					{

		if ( minusp == -1 )

		{

			if ( fmt[i] == '9' )

			{

				out[i] = '0';
			}

			else

			{

				out[i] = ' ';

			}

			minusp = -2;

		}

		else

		{

			out[minusp] = '-';

			minusp = -1;

		}

		( flags & F_RJ ) ? j-- : j++;
				}

		else
				{

		return -1;

		debug1( ( stderr,

			"OUT:\t format()\n" ) );
				}
			}
			else
			{
				if
( ( curp == len ) && !( flags & F_RJ ) )
				{

		curp = i;

				}


		switch( fmt[i] )
				{

		case 'X':

		out[i] = ' ';

		break;

		case '9':

		out[i] = '0';

		break;

		case 'n':

		out[i] = ' ';

		break;
				}
			}
			else if ( fmt[i] == 'M' )
			{
				switch (
minusp )
				{
				case -1 :

		out[i] = ' ';

		minusp = i;

		break;
				case -2 :

		out[i] = '-';

		minusp = -1;

		break;
				default :

		return -1;

		debug1( ( stderr, "OUT:\t format()\n" )
);
				}
			}
			else
			{
				out[i] = fmt[i];
			}
		}

		if ( minusp == -2 )
		{
			return -1;
		}
		if ( !( flags & F_RJ ) && ( data[j] != 0 )
)
		{
			return -1;
		}
		if ( ( flags & F_RJ ) && ( j != -1 ) )
		{
			return -1;
		}
		if ( !( flags & F_RJ ) )
		{
			out[i] = 0;
		}

		debug2( ( stderr, "RTN:\t k: %d out:
'%s'\n", k, out ) );
		debug1( ( stderr, "OUT:\t format()\n" )
);

		return curp;
}

void queryframe( WINDOW *win, int mode,
TextField *tf, DataField *df )
{
		int		i, j;
		int		exitflag = FALSE;
		int		writtenflag = FALSE;
```

```
	int		start;
	char		*data;
	int		key;

	debug1( ( stderr, "IN:\t
queryframe()\n" ) );
	debug2( ( stderr, "ARGS:\t Trow: %d
Tcol: %d Drow: %d Dcol: %d\n",
		(*tf).row, (*tf).col,
(*df).row, (*df).col ) );

	if ( mode & FR_RETRY )
	{
		/* some field is filled with
data */

		writtenflag = TRUE;
	}

	wattrset( win, ATTR_BASE );

	for ( i = 0; !eos( tf[i] ); i++ )
	{
		mvwaddstr( win, tf[i].row,
tf[i].col, tf[i].text );

		debug2( ( stderr,
"ARGV:\t Ttext '%s'\n", tf[i].text ) );
	}

	for ( i = 0; !eos( df[i] ); i++ )
	{
		if ( df[i].type & F_PTR )
		{
			data =
df[i].x.dptr;
		}
		else
		{
			data =
df[i].x.data;
		}

		printfield( win, df[i].row,
df[i].col, df[i].fmt, data, -1,
			df[i].type );

		if ( df[i].type & F_START
)
		{
			start = i;
		}
	}

	wmove( win, df[start].row, df[start].col
);
	wrefresh( win );

	i = start;
	while ( exitflag != TRUE )
	{
		if ( df[i].type & F_PTR )
		{
			data =
df[i].x.dptr;
		}
		else
		{
			data =
df[i].x.data;
		}

		key = getfield( win,
df[i].row, df[i].col, data, df[i].type,
			df[i].fmt,
df[i].actionf, 0 );

		if ( ( writtenflag == FALSE
) && ( key & 1 ) )
		{
			writtenflag =
TRUE;
		}

		switch( key >> 1 ) {
		case KEY_UP:
			j = i-1;
			while ( j >= 0
)
			{
				if
( df[j].type & F_RO )
				{

	j--;

	debug3( ( stderr,

		"ARGS: up pos: %d\n", j )
);
				}
	else
				{

	i = j;

	debug3( ( stderr,

		"ARGS: UP pos: %d\n", i
) );

	break;
				}
			}
			break;

		case TAB:
		case KEY_DOWN:
			j = i+1;
			while ( !eos(
df[j] ) )
			{
				if
( df[j].type & F_RO )
				{

	j++;

	debug3( ( stderr,

		"ARGS: down pos: %d\n",
j ) );
				}

	else
				{

	i = j;

	debug3( ( stderr,

		"ARGS: DOWN pos:
%d\n", i ) );

	break;
				}
			}
			break;

		case ESC:
			break;

		case CR:
		case LF:
		default :
			if ( writtenflag
== TRUE )
			{

	exitflag = TRUE;
			}
			break;
		}
	}

	wrefresh( win );

#if 0
	touchwin( win );
	wnoutrefresh( win );
	doupdate();
#endif

	debug1( ( stderr, "OUT:\t
queryframe()\n" ) );
}

#else	/* !USE_OLD */

/* Format data according to fmt, and send back in
out */

int format( int flags, char *fmt, char *data, int
cursorpos, char *out )
{
	int		fmtpos;
	int		datapos;
	int		fmtend;
	int		dataend;
	int		newcursorpos;
	int		minusvalue = FALSE;
	int		delta;

	debug1( ( stderr, "IN:\t format()\n" ) );
	debug2( ( stderr, "ARGS:\t flag:
0x%X, fmt: '%s', data: '%s',"
			" cursor:
%d\n", flags, fmt, data, cursorpos ) );

	if ( data[0] == '-' )
	{
		minusvalue = TRUE;
	}
```

```
        if ( flags & F_RJ )
        {
                delta = -1;
                datapos = strlen( data )-1;
                dataend = -1;
                fmtpos = strlen( fmt )-1;
                fmtend = -1;

                newcursorpos =
fmtpos+1;       /* == strlen( fmt ) */
                out[fmtpos+1] = '\0';
        /* terminator */
        }
        else
        {
                delta = +1;
                datapos = 0;
                dataend = strlen( data );
                fmtpos = 0;
                fmtend = strlen( fmt );

                newcursorpos = -1;
        }

        for ( ; fmtpos != fmtend; fmtpos +=
delta )
        {
                if ( datapos == dataend )
                {
                        break;
                }

                switch ( fmt[fmtpos] )
                {
                case '9':
                        if ( !isdigit(
data[datapos] ) )
                        {
#if 0

        out[fmtpos] = '0';
                                                if
( datapos == cursorpos )
                                                {

        newcursorpos = fmtpos;
                                                }

        datapos += delta;
#endif

        break;
                        }

                case 'n':
                        if ( !isdigit(
data[datapos] ) )
                        {
#if 0

        out[fmtpos] = ' ';
                                                if
( datapos == cursorpos )
                                                {

        newcursorpos = fmtpos;
                                                }

        datapos += delta;
#endif

        break;
                                }

                        case 'X':
                                out[fmtpos] =
data[datapos];
                                if ( datapos
== cursorpos )
                                {

        newcursorpos = fmtpos;
                                }
                                datapos +=
delta;
                                break;

                        case 'M':
                                if (
minusvalue == TRUE )
                                {

        out[fmtpos] = '-';
                                }
                                else
                                {

        out[fmtpos] = ' ';
                                }
                                break;

                        default:
                                out[fmtpos] =
fmt[fmtpos];
                                break;
                        }
                }

                if ( newcursorpos == -1 )
                {
                        newcursorpos = fmtpos;
                }

                for ( ; fmtpos != fmtend; fmtpos +=
delta )
                {
                        switch( fmt[fmtpos] )
                        {
                        case 'X':
                                out[fmtpos] =
' ';
                                break;

                        case '9':
                                out[fmtpos] =
'0';
                                break;

                        case 'n':
                                out[fmtpos] =
' ';
                                break;

                        case 'M':
                                if (
minusvalue == TRUE )
                                {

        out[fmtpos] = '-';
                                }
                                else
                                {

        out[fmtpos] = ' ';
                                }
                                break;

                        default:
                                out[fmtpos] =
fmt[fmtpos];
                                break;
                        }
                }

                if ( datapos != dataend )
                {
                        debug( ( stderr, "Error:\t
Data is %d-char long. "
                                "(pos:%d,
end:%d)\n",
                                datapos-
dataend, datapos, dataend ) );

                        return -1;
                }

                if ( flags & F_RJ )
                {
                        /* already terminated.
(see beginning of this func.) */
                }
                else
                {
                        out[fmtpos] = '\0';
                /* terminator */
                }

                debug2( ( stderr, "RTN:\t New
position: %d\n", newcursorpos ) );
                debug1( ( stderr, "OUT:\t format()\n" )
);

                return newcursorpos;
}

void queryframe( WINDOW *win, int mode,
TextField *tf, DataField *df )
{
        int     i, j, k;
        int     exitflag = FALSE;
        int     writtenflag = FALSE;
        int     start;
        char    *data;
        int     key;
        int     p;
        char    datalen;
        char    outbuf[80];

        debug1( ( stderr, "IN:\t
queryframe()\n" ) );
        debug2( ( stderr, "ARGS:\t win: %X,
mode: 0x%X, Text: %X, Data: %X\n",
```

```
                win, mode, tf, df ) );
        debug3( ( stderr, "\t tf(%d, %d), df(%d,
%d)\n", tf->row, tf->col,
                df->row, df->col ) );

        if ( mode & FR_RETRY )
        {
                writtenflag = TRUE;
        }

        debug2( ( stderr, "\t Display
TextField\n" ) );

        wattrset( win, ATTR_BASE );
        for ( i = 0; !eos( tf[i] ); i++ )
        {
                wmove( win , tf[i].row,
tf[i].col );
                waddstr( win, tf[i].text );

                debug3( ( stderr, "\t
Text(%d) : %s\n", i, tf[i].text ) );
        }

        debug2( ( stderr, "\t Display
DataField\n" ) );

        for ( i = 0; !eos( df[i] ); i++ )
        {
                if ( df[i].type & F_PTR )
                {
                        data =
df[i].x.dptr;
                }
                else
                {
                        data =
df[i].x.data;
                }

                printfield( win, df[i].row,
df[i].col, df[i].fmt, data, -1,
                        df[i].type );

                if ( df[i].type & F_START
)
                {
                        start = i;
                }
        }

        wmove( win, df[start].row, df[start].col
);
        wrefresh( win );

        i = start;
        if ( df[i].type & F_PTR )
        {
                data = df[i].x.dptr;
        }
        else
        {
                data = df[i].x.data;
        }
        datalen = strlen( data );
        p = datalen;

        while ( exitflag != TRUE )
        {
                printfield( win, df[i].row,
df[i].col, df[i].fmt, data, p,
                        df[i].type );

#ifdef  DODELAYEDUPDATE
                if ( isemptybuf() == 0 )
                {
                        wrefresh( win
);
                }

                key = wgetch_buf( win );
#else
                wrefresh( win );
                key = wgetch( win );
#endif
                debug3( ( stderr, "\t Key:
%d(0x%x)\n", key, key ) );

                if ( ISCURSESKEY( key )
|| iscntrl( key ) )
                {
                        switch( key )
                        {
                        case
KEY_UP:
                                j
= i-1;

        while ( j >= 0 && ( df[j].type & F_RO )
)
                                {

        j--;

        debug4( ( stderr, "up %d\n", j ) );
                                }
                                if
( j >= 0 )
                                {

        i = j;

        debug3( ( stderr, "UP %d\n", i ) );
                                }

                                if
( df[i].type & F_PTR )
                                {

        data = df[i].x.dptr;
                                }

        else
                                {

        data = df[i].x.data;
                                }

        datalen = strlen( data );
                                p
= datalen;

        break;

                        case
KEY_DOWN:
                                j
= i+1;

        while ( !eos( df[j] ) && ( df[j].type &
F_RO ) )
                                {

        j++;

        debug4( ( stderr, "down %d\n",j ) );
                                }
                                if
( !eos( df[j] ) )
                                {

        i = j;

        debug3( ( stderr, "DOWN %d\n",i ) );
                                }

                                if
( df[i].type & F_PTR )
                                {

        data = df[i].x.dptr;
                                }

        else
                                {

        data = df[i].x.data;
                                }

        datalen = strlen( data );
                                p
= datalen;

        break;

                        case ESC:

        break;

                        case CR:
                        case LF:
                                if
( writtenflag == TRUE )
                                {

        exitflag = TRUE;
                                }

        break;

                        case
KEY_LEFT:
                                if
( p > 0 )
                                {

        p--;
                                }

        break;

                        case TAB:
```

```
                        case
KEY_RIGHT:
                                if
( p < datalen )
                                {
        p++;
                                }
        break;

                        case BS:
                        case
KEY_BACKSPACE:
                                if
( p > 0 )
                                {
        p--;
        for ( j = p; data[j] != 0; j++ )
        {
                data[j] = data[j+1];
        }
        datalen--;
                                }
        break;
                        case
KEY_DC:
                        case DEL:
                                if
( p >= 0 )
                                {
        for ( j = p; data[j] != 0; j++ )
        {
                data[j] = data[j+1];
        }
        datalen--;
                                }
        break;

                        default:
        break;
                        }
                }
                else if ( ( key == ' ' ) && (
df[i].type & F_ACTION ) )
                {
                        ( df[i].actionf
)( 0 );         /* do action */
                }
                else if ( ( df[i].type &
F_ACTION ) || ( df[i].type & F_RO ) )
                {
                        /* skip */
                }
                else if ( isprint( key ) )
                {
                        for ( j =
datalen; j >= p; j-- )
                        {
        data[j+1] = data[j];
                        }
                        data[p++] = (
char )key;
                        datalen++;

                        if ( format(
df[i].type, df[i].fmt, data, p, outbuf )
        == -1 )
                        {
        p--;
        for ( j = p; data[j] != 0; j++ )
                                {
        data[j] = data[j+1];
                                }
        datalen--;
                        } else {
        writtenflag = TRUE;
                        }
                }

#if 0
        wrefresh( win );
#endif
        debug1( ( stderr, "OUT:\t
queryframe()\n" ) );

}

#ifdef      DODELAYEDUPDATE

/*** For Delayed update ***/

#define     KEYBUFSIZE          8

static int  keybuf[KEYBUFSIZE];
static int  nkey = 0;
static int  nextkey = 0;

int isemptybuf()
{
        return nkey;
}

int wgetch_buf( WINDOW *win )
{
        int                 ch;

        debug1( ( stderr, "IN:\t
wgetch_buf()\n" ) );

        if ( nkey == 0 )
        {
                while ( ( ch = wgetch( win
) ) == ERR )
                {
                        debug( (
stderr,
        "Error:\t wgetch_buf() get ERR
code\n" ) );
                        /* nodelay(
win, FALSE ); */
                        /* ttimeout(
1000 ); */
                }
                keybuf[nkey++] = ch;

                nodelay( win, TRUE );
                /* ttimeout( 0 ); */
                debug4( ( stderr, "\t
clear:%d, leave:%d, immed:%d, sync:%d, "
                        "scroll:%d,
use_idl:%d, notimeout:%d\n",
                        win->_clear,
win->_leave, win->_immed, win->_sync,
                        win->_scroll,
win->_use_idl, win->_notimeout ) );
                debug4( ( stderr, "\t
use_idc:%d, delay:%d, nbyte:%d, "
                        "index:%d\n",
win->_use_idc, win->_delay, win->_nbyte,
                        win->_index )
);

                while ( ( ch = wgetch( win
) ) != ERR )
                {

        keybuf[nkey++] = ch;
                        if ( nkey ==
KEYBUFSIZE )
                        {

        break;
                        }
                }

                nodelay( win, FALSE );
                /* ttimeout( -1 ); */
                debug4( ( stderr, "\t
clear:%d, leave:%d, immed:%d, sync:%d, "
                        "scroll:%d,
use_idl:%d, notimeout:%d\n",
                        win->_clear,
win->_leave, win->_immed, win->_sync,
                        win->_scroll,
win->_use_idl, win->_notimeout ) );
                debug4( ( stderr, "\t
use_idc:%d, delay:%d, nbyte:%d, "
                        "index:%d\n",
win->_use_idc, win->_delay, win->_nbyte,
                        win->_index )
);

                nextkey = 0;
        }

        debug3( ( stderr, "\t nkey = %d,
nextkey = %d\n", nkey, nextkey ) );
```

```
		debug1( ( stderr, "OUT:\t
wgetch_buf()\n" ) );

		nkey--;
		return keybuf[nextkey++];
}

#endif	/* DODELAYEDUPDATE */

#endif	/* !USE_OLD */

/* Change the status line in the main window */

void changestatus( char *status )
{
		int		len = strlen( status );
		long		attr;

		move( LINES-1, (COLS/2)-(len/2) );
		attr = getattrs( stdscr );

		attrset( ATTR_STATUS );
		deleteln();
		addstr( status );
		attrset( attr );

		refresh();
}

/* Refresh several windows at a time with no
flicker */

void nrefresh( int n, ... )
{
		va_list	l;
		WINDOW	*win;
		int		i;

		va_start( l, n );
		for ( i = 0; i < n; i++ )
		{
			win = va_arg( l,
WINDOW * );
			touchwin( win );
			wnoutrefresh( win );
		}
		va_end( l );

		doupdate();
}

/* Open curses and setup */

WINDOW *initScreen( char *title, int flags )
{
		int		len = ( int )strlen( title );

		initscr();
		savetty();
		setup_attrs();
		cbreak();
		noecho();
		nonl();
#ifndef		__linux__
		intrflush( stdscr, FALSE );
#endif
		keypad( stdscr, TRUE );
		nodelay( stdscr, FALSE );
		leaveok( stdscr, FALSE );

		if ( !( flags & WIN_NOBORDER ) )
		{
			drawbox( stdscr, 0, 0,
LINES-1, COLS );
		}
		else
		{
			len = 0;
		}

		if ( len > 0 )
		{
			move( 0, (COLS/2)-
((len+2)/2) );
			printw( " %s ", title );
		}

		refreshScreen();
}

/* Setup the attributes used for the UI depending if
we're on */

void setup_attrs()
{
#ifdef USECOLOUR
		if ( has_colors() )
		{
			start_color();

			init_pair( COL_KOG,
COLOR_BLACK, COLOR_GREEN );
			init_pair( COL_KOY,
COLOR_BLACK, COLOR_YELLOW );
			init_pair( COL_KOC,
COLOR_BLACK, COLOR_CYAN );
			init_pair( COL_KOW,
COLOR_BLACK, COLOR_WHITE );
			init_pair( COL_ROY,
COLOR_RED, COLOR_YELLOW );
			init_pair( COL_YOK,
COLOR_YELLOW, COLOR_BLACK );
			init_pair( COL_YOR,
COLOR_YELLOW, COLOR_RED );
			init_pair( COL_YOB,
COLOR_YELLOW, COLOR_BLUE );
			init_pair( COL_BOC,
COLOR_BLUE, COLOR_CYAN );
			init_pair( COL_BOW,
COLOR_BLUE, COLOR_WHITE );
			init_pair( COL_COB,
COLOR_CYAN, COLOR_BLUE );
			init_pair( COL_COK,
COLOR_CYAN, COLOR_BLACK );
			init_pair( COL_WOR,
COLOR_WHITE, COLOR_RED );
			init_pair( COL_WOG,
COLOR_WHITE, COLOR_GREEN );
			init_pair( COL_WOB,
COLOR_WHITE, COLOR_BLUE );

			ATTR_BASE
		= COL_BASE;
			ATTR_STATUS
		= COL_STATUS;
			ATTR_MENU
		= COL_MENU;
			ATTR_MENUBORDER
		= COL_MENUBORDER;
			ATTR_LININP
		= COL_LININP;
			ATTR_LININPBORDER
		= COL_LININPBORDER;
			ATTR_FRAME
		= COL_FRAME;
			ATTR_FRAMEBORDER
		= COL_FRAMEBORDER;
			ATTR_RO_FIELD
		= COL_RO_FIELD;
			ATTR_NE_FIELD
		= COL_NE_FIELD;
			ATTR_ACTION_FIELD
		= COL_ACTION_FIELD;
			ATTR_NORMAL_FIELD
		= COL_NORMAL_FIELD;
			ATTR_DIBOX
		= COL_DIBOX;
			ATTR_DIBORDER
		= COL_DIBORDER;
			ATTR_SCRLBOX
		= COL_SCRLBOX;
			ATTR_SCRLBORDER
		= COL_SCRLBORDER;
		}
		else
#endif
		{
			ATTR_BASE
		= BW_BASE;
			ATTR_STATUS
		= BW_STATUS;
			ATTR_MENU
		= BW_MENU;
			ATTR_MENUBORDER
		= BW_MENUBORDER;
			ATTR_LININP
		= BW_LININP;
			ATTR_LININPBORDER
		= BW_LININPBORDER;
			ATTR_FRAME
		= BW_FRAME;
			ATTR_FRAMEBORDER
		= BW_FRAMEBORDER;
			ATTR_RO_FIELD
		= BW_RO_FIELD;
			ATTR_NE_FIELD
		= BW_NE_FIELD;
			ATTR_ACTION_FIELD
		= BW_ACTION_FIELD;
			ATTR_NORMAL_FIELD
		= BW_NORMAL_FIELD;
			ATTR_DIBOX
		= BW_DIBOX;
			ATTR_DIBORDER
		= BW_DIBORDER;
			ATTR_SCRLBOX
		= BW_SCRLBOX;
			ATTR_SCRLBORDER
		= BW_SCRLBORDER;
		}

		attrset( ATTR_BASE );
#ifndef		__linux__
```

```
		bkgd( ATTR_BASE );
#endif
}

/* Create a window with a border and title */

WINDOW *createWindow( int row, int col, int
height, int width, char *title,
                                        int flags, long
wattr, long fattr )
{
		WINDOW	*win;
		char		buf[80];
		int		len;

		win = newwin( height, width, row, col
);
		leaveok( stdscr, TRUE );
#ifndef			__linux__
		wbkgd( win, wattr );
#endif

		if ( !( flags & WIN_NOBORDER ) )
		{
			wattrset( win, fattr );
			box( win, ACS_VLINE,
ACS_HLINE );
			len = ( int )strlen( title );
		}
		else
		{
			len = 0;
		}

		wattrset( win, wattr );

		keypad( win, TRUE );
#ifndef			__linux__
		intrflush( win, FALSE );
#endif
		nodelay( win, FALSE );

		if ( len > 0 )
		{
			sprintf( buf, " %s ", title );
			wmove( win, 0, (width/2)-(
(len+2)/2) );
			waddstr( win, buf );
		}

		return win;
}

/* Close a window */

int closeWindow ( WINDOW *win )
{
		wclear( win );
		delwin( win );
}

/* Clear the screen and close curses */

void closeScreen()
{
		slk_clear();
		clear();
		refresh();
		resetty();
		endwin();
}

/* Draw a box */

void drawbox( WINDOW *win, int row, int col, int
height, int width )
{
		wmove( win, row, col );
		waddch( win, ACS_ULCORNER );
		wmove( win, row, col+1 );
		whline( win, ACS_HLINE, width-2 );
		wmove( win, row, col+width-1 );
		waddch( win, ACS_URCORNER );

		wmove( win, row+height-1, col );
		waddch( win, ACS_LLCORNER );
		wmove( win, row+height-1, col+1 );
		whline( win, ACS_HLINE, width-2 );
		wmove( win, row+height-1, col+width-
1 );
		waddch( win, ACS_LRCOR NER );

		wmove( win, row+1, col );
		wvline( win, ACS_VLINE, height-2 );
		wmove( win, row+1, col+width-1 );
		wvline( win, ACS_VLINE, height-2 );
}




****************************************
		ui.h
****************************************


/*
		ui.h : Header of low level screen
operation library

		Version	1.00		1996/12/26
*/

#ifndef _UI_H_
#define _UI_H_

#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <stdarg.h>

#ifndef __linux__
# include <sys/debug.h>
#endif

#ifdef __linux__
# include <ncurses/curses.h>
#else
# include <curses.h>
#endif

#define	BS			8
#define TAB			9
#define	LF			10
#define CR		13
#define ESC			27
#define DEL			127

#define UI_NOBORDER 1
#define WIN_NOBORDER		1

#define BW_BASE			A_NORMAL
#define BW_STATUS	A_NORMAL
#define BW_MENU			A_NORMAL
#define BW_MENUBORDER		A_NORMAL
#define BW_LININP	A_NORMAL
#define BW_LININPBORDER		A_NORMAL
#define BW_FRAME	A_NORMAL
#define BW_FRAMEBORDER		A_NORMAL
#define BW_RO_FIELD A_UNDERLINE
#define BW_NE_FIELD A_UNDERLINE
#define BW_ACTION_FIELD
		A_UNDERLINE
#define BW_NORMAL_FIELD
		A_STANDOUT
#define BW_DIBOX	A_NORMAL
#define BW_DIBORDER		A_NORMAL
#define BW_SCRLBOX A_NORMAL
#define BW_SCRLBORDER		A_NORMAL

#ifdef USECOLOUR
#define COL_KOG			11
#define COL_KOY			12
#define COL_KOC			13
#define COL_KOW			14
#define COL_ROY			15
#define COL_YOK			16
#define COL_YOR			17
#define COL_YOB			18
#define COL_BOC			19
#define COL_BOW			20
#define COL_COK			21
#define COL_COB			22
#define COL_WOR			23
#define COL_WOG			24
#define COL_WOB			25

#define COL_BASE
		COLOR_PAIR(COL_COK)
#define COL_STATUS
		COLOR_PAIR(COL_YOK)
#define COL_MENU
		COLOR_PAIR(COL_WOB)
#define COL_MENUBORDER
		COLOR_PAIR(COL_COB)
#define COL_LININP
		COLOR_PAIR(COL_WOB)
#define COL_LININPBORDER
		COLOR_PAIR(COL_BOC)
#define COL_FRAME
		COLOR_PAIR(COL_WOB)
#define COL_FRAMEBORDER
		COLOR_PAIR(COL_COB)
#define COL_RO_FIELD
		COLOR_PAIR(COL_WOR)
#define COL_NE_FIELD
		COLOR_PAIR(COL_COB) |
A_UNDERLINE
```

```
#define COL_ACTION_FIELD
        COLOR_PAIR(COL_YOR)
#define COL_NORMAL_FIELD
        COLOR_PAIR(COL_KOW)
#define COL_DIBOX
        COLOR_PAIR(COL_WOR)
#define COL_DIBORDER
        COLOR_PAIR(COL_WOR)
#define COL_SCRLBOX
        COLOR_PAIR(COL_WOB)
#define COL_SCRLBORDER
        COLOR_PAIR(COL_COB)
#endif

#define F_RJ                    1
#define F_RO                    2
#define F_ACTION        4
#define F_VALUE                 8
#define F_START                 16
#define F_NE                    32
#define F_PTR                   64

#define refreshScreen()
        refreshWin(stdscr)
#define refreshWin( win )       { \

        touchwin( win ); \

        wrefresh( win ); \
                                        }

WINDOW *initScreen( char *, int );
void closeScreen( void );
WINDOW *createWindow( int, int, int, int, char *,
int, long, long );
int mapWindow( WINDOW * );
int closeWindow( WINDOW * );
int getString( WINDOW *, char *, size_t );
int getfield( WINDOW *, int, int, char *, int, char *,
void (*)(), int );
void printfield( WINDOW *, int, int, char *, char *,
int, int );
void changestatus( char * );
void drawbox( WINDOW *, int, int, int, int );
void fatalerror( char * );
int format( int, char *, char *, int, char * );
void nrefresh( int, ... );

extern long  ATTR_BASE, ATTR_STATUS,
ATTR_MENU, ATTR_MENUBORDER,
                ATTR_LININP,
ATTR_LININPBORDER, ATTR_FRAME,
ATTR_FRAMEBORDER,
                ATTR_RO_FIELD,
ATTR_NE_FIELD, ATTR_ACTION_FIELD,
                ATTR_NORMAL_FIELD,
ATTR_DIBOX, ATTR_DIBORDER,
ATTR_SCRLBOX,
                ATTR_SCRLBORDER;

#endif
```

```
*****************************************
        makefile
*****************************************


#
#       Makefile
#
#       Version 1.00    1996/12/27
#

ORACLE_HOME     = /oracle
SOURCE_DIR      =
$(ORACLE_HOME)/bench/tpc/tpcc/TUX_source
ORACLE_INC      =
$(ORACLE_HOME)/rdbms/demo
TUXEDO_INC      = $(ROOTDIR)/include

MV      = mv
LN      = ln -s
RM      = rm -f
CC      = /usr/ccs/bin/cc
#CC     = /usr/local/bin/gcc

LIBS    = /usr/ccs/lib/libcurses.a
# LIBS  = -lncurses
# MAPOPTION     = -Wl,-M,mapfile
MAPOPTION       =
INCLUDEDIR      = -I. -I${SOURCE_DIR} -
I${ORACLE_INC} -I${TUXEDO_INC} -
I/usr/include/ncurses
CCFLAGSDEFAULT  = -s $(INCLUDEDIR) -O -
K 3  -K TMS -K INF -DDUR -DUSE_OLD
#CFLAGSDEFAULT = -s $(INCLUDEDIR) -O -K 3
-K TMS -K INF -DDUR
#CCFLAGSDEFAULT  = $(INCLUDEDIR) -s  -O -
K 4 -Kinline2  -K TMS -K INF -Kpic -DDUR
#CCFLAGSDEFAULT  = $(INCLUDEDIR) -s -O -
K 4 -K TMS

all     : normal

scrtest :
        make
        CCFLAGS='$(CCFLAGSDEFAULT)
$(MAPOPTION) -DSCRTEST' \
                BLDCLI=$(CC) \
                BLDFIN= \
                BLDFOUT= \
                BLDLIB='$(LIBS)' \
                Tc

2cpu    :
        make
        CCFLAGS='$(CCFLAGSDEFAULT)
$(MAPOPTION) -DCPU2' \

        BLDCLI='${ROOTDIR}/bin/buildclient
-v' \
                BLDFIN='-f "' \
                BLDFOUT='"' \
                BLDLIB='-l $(LIBS)' \
                Tc

normal  :
        make
        CCFLAGS='$(CCFLAGSDEFAULT)
$(MAPOPTION)' \

        BLDCLI='${ROOTDIR}/bin/buildclient
-v' \
                BLDFIN='-f "' \
                BLDFOUT='"' \
                BLDLIB='-l $(LIBS)' \
                Tc

Tc      : Tc.c frame.o ui.o dummy.o
        $(BLDCLI) -o Tc \
                $(BLDFIN) $(CCFLAGS)
Tc.c frame.o ui.o dummy.o $(BLDFOUT) \
                $(BLDLIB)

frame.o: frame.c
        $(CC) -c $(CCFLAGS) frame.c

ui.o    : ui.c
        $(CC) -c $(CCFLAGS) ui.c

dummy.o : dummy.c
        $(CC) -c $(CCFLAGS) $<

version.o : version.c
        -$(RM) version.c
        echo '#define DATE "\c' > version.c
        date >> version.c
        echo '"' >> version.c

clean   :
        -$(RM) Tc *.o
```

Appendix B: Server Source Code

```
*****************************************
            pldel.c
*****************************************


/*=================================
============================+
 |      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(in
t),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(i
nt),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
*****************************************


/*=================================
============================+
 |      Copyright (c) 1995  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 \
  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,VER
7);

  /* 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_c
nt),SIZ(o_ol_cnt),SQLT_INT);


OBNDRV(&tpclda,&curn1,":w_tax",ADR(w_tax),SI
Z(w_tax),SQLT_FLT);

OBNDRV(&tpclda,&curn1,":d_tax",ADR(d_tax),SI
Z(d_tax),SQLT_FLT);

OBNDRV(&tpclda,&curn1,":o_id",ADR(o_id),SIZ(o
_id),SQLT_INT);

OBNDRV(&tpclda,&curn1,":c_discount",ADR(c_di
scount),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_la
st),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,VER
7);

  /* 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_s
upply_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_quanti
ty,SIZ(int),SQLT_INT,
                nctx->nol_quantity_ind,nctx-
>nol_quantity_len,
        nctx->nol_quantity_rcode);

OBNDRA(&tpclda,&curn2,":s_quantity",s_quantity,
SIZ(int),SQLT_INT,
                nctx->s_quant_ind,nctx-
>s_quant_len, nctx->s_quant_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,VE
R7);
    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),SQL
T_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),S
QLT_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,VER
7);

  /* 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_s
upply_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_quanti
ty,SIZ(int),SQLT_INT,
                nctx->nol_quantity_ind,nctx-
>nol_quantity_len,
        nctx->nol_quantity_rcode);

OBNDRA(&tpclda,&curn4,":ol_amount",nol_amou
nt,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) {
                fprintf(stderr,"curn1.rc=%d\n",curn1.rc
);
        orol (&tpclda);
        retries++;
        goto retry;
      }
      else if (errrpt (&tpclda, &curn1) ==
RECOVERR) {
                fprintf(stderr,"curn2.rc=%d\n",curn1.r c
);
        orol (&tpclda);
        retries++;
        goto retry;
      }
      else {
                fprintf(stderr,"curn3.rc=%d\n",curn1.rc
);
        orol (&tpclda);
        return (-1);
      }
    }

#ifdef ISO7
iso7:
#endif

    /* 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 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) {
                fprintf(stderr,"curn4.rc=%d\n",curn3[d
_id-1].rc);
        orol (&tpclda);
        retries++;
        goto retry;
      }
      else if (curn3[d_id-1].rc !=
NO_DATA_FOUND) {
        if (errrpt (&tpclda, &curn3[d_id-1]) ==
RECOVERR) {
                fprintf(stderr,"curn5.rc=%d\n",curn3[d
_id-1].rc);
          orol (&tpclda);
          retries++;
          goto retry;
        }
        else {
                fprintf(stderr,"curn6.rc=%d\n",curn3[d
_id-1].rc);
          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++)
       { fprintf(stderr,"Mark invalid items\n");
        nctx->i_id_ind[i] = NA;
                }

    /* 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;
      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) {
        s_quantity[i] -= nol_quantity[i];
        if (s_quantity[i] < 10)
          s_quantity[i] += 91;
        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 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) {
                fprintf(stderr,"curn8.rc=%d\n",curn2.rc
);
        orol (&tpclda);
        retries++;
        goto retry;
      }
      else {
                fprintf(stderr,"curn9.rc=%d\n",curn2.rc
);
        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) {
                fprintf(stderr,"curA.rc=%d\n",curn2.rc);
            orol (&tpclda);
            retries++;
            goto retry;
          }
          else if (errrpt (&tpclda, &curn2) ==
RECOVERR) {
                fprintf(stderr,"curB.rc=%d\n",curn2.rc);
            orol (&tpclda);
            retries++;
            goto retry;
          }
          else {
                fprintf(stderr,"curC.rc=%d\n",curn2.rc)
;
            orol (&tpclda);
            return (-1);
          }
        }
        rpc += curn2.rpc;
        rowoff += curn2.rpc + 1;
      }
    }

     /* 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);
    }

    /* 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) {
                fprintf(stderr,"curD.rc=%d\n",curn4.rc)
;
          orol (&tpclda);
          retries++;
          goto retry;
        }
        else if (errrpt (&tpclda, &curn4) ==
RECOVERR) {
                fprintf(stderr,"curE.rc=%d\n",curn4.rc);
          orol (&tpclda);
          retries++;
          goto retry;
        }
        else {
                fprintf(stderr,"curF.rc=%d\n",curn4.rc);
          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) {
                fprintf(stderr,"curG.rc=%d\n",curn4.rc)
;
            orol (&tpclda);
            retries++;
            goto retry;
          }
          else if (errrpt (&tpclda, &curn4) ==
RECOVERR) {
                fprintf(stderr,"curH.rc=%d\n",curn4.rc)
;
            orol (&tpclda);
            retries++;
            goto retry;
          }
          else {
                fprintf(stderr,"curI.rc=%d\n",curn4.rc);
            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
*****************************************


/*=================================
============================+
|      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_i
d),SQLT_INT);

OBNDRV(&tpclda,&curo,":c_id",ADR(c_id),SIZ(c_i
d),SQLT_INT);

OBNDRV(&tpclda,&curo,":byln",ADR(bylastname),
SIZ(bylastname),SQLT_INT);

OBNDRV(&tpclda,&curo,":c_last",c_last,SIZ(c_las
t),SQLT_STR);

OBNDRV(&tpclda,&curo,":c_first",c_first,SIZ(c_firs
t),SQLT_STR);

OBNDRV(&tpclda,&curo,":c_middle",c_middle,SIZ
(c_middle),SQLT_STR);

OBNDRV(&tpclda,&curo,":c_balance",ADR(c_bala
nce),SIZ(c_balance),SQLT_FLT);

OBNDRV(&tpclda,&curo,":o_id",ADR(o_id),SIZ(o_i
d),SQLT_INT);

OBNDRV(&tpclda,&curo,":o_entry_d",o_entry_d,S
IZ(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_quantit
y,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_STR,
                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
*****************************************
```

```
/*=================================
============================+
|       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_i
d),SQLT_INT);

OBNDRV(&tpclda,&curp,":c_w_id",ADR(c_w_id),S
IZ(c_w_id),SQLT_INT);

OBNDRV(&tpclda,&curp,":c_d_id",ADR(c_d_id),SI
Z(c_d_id),SQLT_INT);

OBNDRV(&tpclda,&curp,":c_id",ADR(c_id),SIZ(c_i
d),SQLT_INT);

OBNDRV(&tpclda,&curp,":byln",ADR(bylastname),
SIZ(bylastname),SQLT_INT);

OBNDRV(&tpclda,&curp,":h_amount",ADR(h_amo
unt),SIZ(h_amount),SQLT_FLT);

OBNDRV(&tpclda,&curp,":c_last",c_last,SIZ(c_las
t),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_ci
ty),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_cit
y),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_firs
t),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_cr
edit_lim),SIZ(c_credit_lim),
        SQLT_FLT);

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

OBNDRV(&tpclda,&curp,":c_balance",ADR(c_bala
nce),SIZ(c_balance),SQLT_FLT);

OBNDRV(&tpclda,&curp,":c_data",c_data,SIZ(c_d
ata),SQLT_STR);

OBNDRV(&tpclda,&curp,":h_date",h_date,SIZ(h_d
ate),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
*****************************************
```

```
/*=================================================
============================+
 |      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_i
d),SQLT_INT);

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

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

  return (0);
}

plsto ()
{
  OEXEC(&tpclda,&curs);

  return (0);
}

void plstodone ()
{
  if (oclose (&curs))
    errrpt (&tpclda, &curs);
}
```

```
*****************************************
            tpcc.h
*****************************************


/*=================================================
============================+
 |      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
*****************************************


/*=================================
============================+
|       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

/* New order */

struct newinstruct {
  int w_id;
  int d_id;
  int c_id;
  int ol_i_id[15];
  int ol_supply_w_id[15];
  int ol_quantity[15];
};

struct newoutstruct {
  int terror;
  int o_id;
  int o_ol_cnt;
  char c_last[17];
  char c_credit[3];
  float c_discount;
  float w_tax;
  float d_tax;
  char o_entry_d[20];
  float total_amount;
  char i_name[15][25];
  int s_quantity[15];
  char brand_generic[15];
  float i_price[15];
  float ol_amount[15];
  char status[26];
  int retry;
};

struct newstruct {
  struct newinstruct newin;
  struct newoutstruct newout;
};

/* Payment */

struct payinstruct {
  int w_id;
  int d_id;
  int c_w_id;
  int c_d_id;
  int c_id;
  int bylastname;
  float h_amount;
  char c_last[17];
};

struct payoutstruct {
  int terror;
  char w_street_1[21];
  char w_street_2[21];
  char w_city[21];
  char w_state[3];
  char w_zip[10];
  char d_street_1[21];
  char d_street_2[21];
  char d_city[21];
  char d_state[3];
  char d_zip[10];
  int  c_id;
  char c_first[17];
  char c_middle[3];
  char c_last[17];
  char c_street_1[21];
  char c_street_2[21];
  char c_city[21];
  char c_state[3];
  char c_zip[10];
  char c_phone[17];
  char c_since[11];
  char c_credit[3];
  double c_credit_lim;
  float c_discount;
  double c_balance;
  char c_data[201];
  char h_date[20];
  int retry;
};

struct paystruct {
  struct payinstruct payin;
  struct payoutstruct payout;
};

/* Order status */

struct ordinstruct {
  int w_id;
  int d_id;
  int c_id;
  int bylastname;
  char c_last[17];
};

struct ordoutstruct {
  int terror;
  int c_id;
  char c_last[17];
  char c_first[17];
  char c_middle[3];
  double c_balance;
  int o_id;
  char o_entry_d[20];
  int o_carrier_id;
  int o_ol_cnt;
  int ol_supply_w_id[15];
  int ol_i_id[15];
  int ol_quantity[15];
  float ol_amount[15];
  char ol_delivery_d[15][11];
  int retry;
};

struct ordstruct {
  struct ordinstruct ordin;
  struct ordoutstruct ordout;
};

/* Delivery */

struct delinstruct {
  int w_id;
  int o_carrier_id;
  double qtime;
  int in_timing_int;
};

struct deloutstruct {
  int terror;
  int retry;
};

struct delstruct {
  struct delinstruct delin;
  struct deloutstruct delout;
};

/* Stock level */

struct stoinstruct {
  int w_id;
  int d_id;
  int threshold;
};

struct stooutstruct {
  int terror;
  int low_stock;
  int retry;
};

struct stostruct {
  struct stoinstruct stoin;
  struct stooutstruct stoout;
};

#endif



*****************************************
              tpccpl.c
*****************************************


/*=================================
============================+
```

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

/* #define SQLTXT "alter session set
isolation_level = read committed" */
#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;

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)
    return (RECOVERR);
  else
    return (IRRECERR);
}

TPCexit ()
{
  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;
  }

  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_%d.del", proc_no);
  if ((lfp = fopen (filename, "w")) == NULL) {
#ifdef TUX
```

```
    userlog ("Error in TPC-C server %d: Failed to
open %s\n",
            proc_no, filename);
#else
    fprintf (stderr, "Error in TPC-C server %d:
Failed to open %s\n",
            proc_no, filename);
#endif
    return (-1);
  }

  /* 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 */

  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);

  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;

  return (0);

}

TPCnew (str)
struct newstruct *str;
{
  int i;

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

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

  str->newout.terror = NOERR;
  str->newout.o_id = o_id;
  str->newout.o_ol_cnt = o_ol_cnt;
  strncpy (str->newout.c_last, c_last, 17);
  strncpy (str->newout.c_credit, c_credit, 3);
  str->newout.c_discount = c_discount;
  str->newout.w_tax = w_tax;
  str->newout.d_tax = d_tax;
  strncpy (str->newout.o_entry_d, o_entry_d, 20);
  str->newout.total_amount = total_amount;
  for (i = 0; i < o_ol_cnt; i++) {
    strncpy (str->newout.i_name[i], i_name[i], 25);
    str->newout.s_quantity[i] = s_quantity[i];
    str->newout.brand_generic[i] = brand_gen[i];
    str->newout.i_price[i] = i_price[i];
    str->newout.ol_amount[i] = nol_amount[i];
  }
  if (status)
    strcpy (str->newout.status, "Item number is
not valid");
  else
    str->newout.status[0] = '\0';
  str->newout.retry = retries;
  return (0);
}

TPCpay (str)
struct paystruct *str;
{
  w_id = str->payin.w_id;
  d_id = str->payin.d_id;
  c_w_id = str->payin.c_w_id;
  c_d_id = str->payin.c_d_id;
  h_amount = str->payin.h_amount;
  bylastname = str->payin.bylastname;
  if (bylastname) {
    c_id = 0;
    strncpy (c_last, str->payin.c_last, 17);
  }
  else {
    c_id = str->payin.c_id;
    strcpy (c_last, " ");
  }
  retries = 0;

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

  str->payout.terror = NOERR;
  strncpy (str->payout.w_street_1, w_street_1,
21);
  strncpy (str->payout.w_street_2, w_street_2,
21);
  strncpy (str->payout.w_city, w_city, 21);
  strncpy (str->payout.w_state, w_state, 3);
  strncpy (str->payout.w_zip, w_zip, 10);
  strncpy (str->payout.d_street_1, d_street_1, 21);
  strncpy (str->payout.d_street_2, d_street_2, 21);
  strncpy (str->payout.d_city, d_city, 21);
  strncpy (str->payout.d_state, d_state, 3);
  strncpy (str->payout.d_zip, d_zip, 10);
  str->payout.c_id = c_id;
  strncpy (str->payout.c_first, c_first, 17);
  strncpy (str->payout.c_middle, c_middle, 3);
  strncpy (str->payout.c_last, c_last, 17);
  strncpy (str->payout.c_street_1, c_street_1, 21);
  strncpy (str->payout.c_street_2, c_street_2, 21);
  strncpy (str->payout.c_city, c_city, 21);
  strncpy (str->payout.c_state, c_state, 3);
  strncpy (str->payout.c_zip, c_zip, 10);
  strncpy (str->payout.c_phone, c_phone, 17);
  strncpy (str->payout.c_since, c_since, 11);
  strncpy (str->payout.c_credit, c_credit, 3);
  str->payout.c_credit_lim = c_credit_lim;
  str->payout.c_discount = c_discount;
  str->payout.c_balance = c_balance;
  strncpy (str->payout.c_data, c_data, 201);
  strncpy (str->payout.h_date, h_date, 20);
```

```
  str->payout.retry = retries;
  return (0);
}

TPCord (str)
struct ordstruct *str;
{
  int i;

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

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

  str->ordout.terror = NOERR;
  str->ordout.c_id = c_id;
  strncpy (str->ordout.c_last, c_last, 17);
  strncpy (str->ordout.c_first, c_first, 17);
  strncpy (str->ordout.c_middle, c_middle, 3);
  str->ordout.c_balance = c_balance;
  str->ordout.o_id = o_id;
  strncpy (str->ordout.o_entry_d, o_entry_d, 20);
  str->ordout.o_carrier_id = o_carrier_id;
  str->ordout.o_ol_cnt = o_ol_cnt;
  for (i = 0; i < o_ol_cnt; i++) {
    ol_delivery_d[i][10] = '\0';
    str->ordout.ol_supply_w_id[i] =
ol_supply_w_id[i];
    str->ordout.ol_i_id[i] = ol_i_id[i];
    str->ordout.ol_quantity[i] = ol_quantity[i];
    str->ordout.ol_amount[i] = ol_amount[i];
    strncpy (str->ordout.ol_delivery_d[i],
ol_delivery_d[i], 11);
  }
  str->ordout.retry = retries;
  return (0);
}

TPCdel (str)
struct delstruct *str;
{
  double tr_end;
  int i;

  w_id = str->delin.w_id;
  o_carrier_id = str->delin.o_carrier_id;
  retries = 0;

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

  tr_end = gettime ();
  fprintf (lfp, "%d %d %f %f %d %d", str-
>delin.in_timing_int,
           (tr_end - str->delin.qtime) <= DELRT ? 1 :
0,
           str->delin.qtime, tr_end, w_id,
o_carrier_id);
  for (i = 0; i < 10; i++) {
    fprintf (lfp, " %d %d", i + 1, del_o_id[i]);
    if (del_o_id[i] <= 0) {
#ifdef TUX
      userlog ("DELIVERY: no new order for w_id:
%d, d_id %d\n",
               w_id, i + 1);
#else
      fprintf (stderr, "DELIVERY: no new order for
w_id: %d, d_id %d\n",
               w_id, i + 1);
#endif
    }
  }
  fprintf (lfp, " %d\n", retries);
  str->delout.terror = NOERR;
  str->delout.retry = retries;
  return (0);
}

TPCsto (str)
struct stostruct *str;
{
  w_id = str->stoin.w_id;
  d_id = str->stoin.d_id;
  threshold = str->stoin.threshold;
  retries = 0;

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

  str->stoout.terror = NOERR;
  str->stoout.low_stock = low_stock;
  str->stoout.retry = retries;
  return (0);
}




****************************************
            tpccpl.h
****************************************


/*==================================
============================+
|      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 */

#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),(p
rogvl),(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),(p
rogvl),(ftype),NA,\

(indp),(alen),(arcode),(ub4)0,(ub4*)0,(text*)0,NA,N
A))\
      {errrpt(lda,cursor);return(-1);}\
   else\
      DISCARD 0

#define
OBNDRAA(lda,cursor,sqlvar,progv,progvl,ftype,ind
p,alen,arcode,ms,cs)\
   if
(obndra((cursor),(text*)(sqlvar),NA,(ub1*)(progv),(p
rogvl),(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,f
mt,fmtl,fmtt,rlen,rcode)\
   if
(odefin((cursor),(pos),(ub1*)(buf),(bufl),(ftype),(sca
le),(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)(l
ngflg)))\
      {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




*****************************************
            tpccsvr.c
*****************************************


/*=================================
============================+
```

```
|       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>

#define FJ
#ifdef FJ
union infostruct
{
   int tran_kind;
   struct newstruct newinfo;
   struct paystruct payinfo;
   struct ordstruct ordinfo;
   struct delstruct delinfo;
   struct stostruct stoinfo;
} *info;
#endif

struct newstruct *newinfo;
struct paystruct *payinfo;
struct ordstruct *ordinfo;
struct delstruct *delinfo;
struct stostruct *stoinfo;

tpsvrinit (argc, argv)
int argc;
char *argv[];
{
  int id;
  char *uid;

  if (argc >= 2) {
    id = atoi (argv[argc - 2]);
    uid = argv[argc - 1];
    return (TPCinit (id, uid));
  }
  else {
    userlog ("Error: not enough arguments to
tpsvrinit\n");
    return (-1);
  }
}

void tpsvrdone ()
{
  TPCexit ();

}

#ifdef FJ
TPCC01(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newst ruct *)info;
            if (TPCnew (newinfo))
              tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
              tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
              tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
              tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

   case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
              tpreturn (TPFAI L, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
              tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

   case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
              tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
              tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

   case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
              tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
              tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
   }
}

TPCC02(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
              tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
              tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
              tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
              tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

   case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
              tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
              tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

   case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
              tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
              tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

   case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
              tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
              tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
   }
}

TPCC03(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
              tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
              tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
```

```
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
                tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
    }
}

TPCC04(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
                tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
    }
}

TPCC05(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinf o, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
                tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
    }
}

TPCC06(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
```

```
                tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
    }
}

TPCC07(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
                tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
    }
}

TPCC08(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
                tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
    }
}

TPCC09(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0,  ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
                tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
    }
}

TPCC10(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
```

```
            else
               tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

   case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
               tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

   case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
               tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
               tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

   case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
               tpreturn (TPFAIL,  0, stoinfo, sizeof
(struct stostruct), 0);
            else
               tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
   }
}

TPCC11(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
               tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
               tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
               tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

   case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
               tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

   case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
               tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
               tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

   case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
               tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
               tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
   }
}

TPCC12(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
               tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
               tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
               tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

   case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
               tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

   case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
               tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
               tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

   case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
               tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
               tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
   }
}

TPCC13(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
               tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
               tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
               tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

   case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
               tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

   case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
               tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
               tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

   case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
               tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
```

```
            else
               tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
   }
}

TPCC14(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
               tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
               tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
               tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

   case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
               tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

   case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
               tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
               tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

   case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
               tpreturn (TPFAIL, 0, stoinfo, sizeof
(struct stostruct), 0);
            else
               tpreturn (TPSUCCESS, 0, stoinfo,
sizeof (struct stostruct), 0);
            break;
   }
}

TPCC15(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
               tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
               tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
               tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

   case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
               tpreturn (TPFAIL, 0, ordinf o, sizeof
(struct ordstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

   case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
               tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
               tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

   case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
               tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
            else
               tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
            break;
   }
}

TPCC16(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
               tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
               tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
               tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

   case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
               tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
               tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

   case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
               tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
               tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

   case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
               tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
            else
               tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
            break;
   }
}

TPCC17(msg)
TPSVCINFO *msg;
{
   info = (union infostruct *)msg->data;
   switch(info->tran_kind)
   {
   case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
               tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
               tpreturn (TPSUC CESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

   case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
               tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
```

```
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
                break;

    case TRANORD:
                ordinfo = (struct ordstruct *)info;
                if (TPCord (ordinfo))
                   tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
                break;

    case TRANDEL:
                delinfo = (struct delstruct *)info;
                if (TPCdel (delinfo))
                   tpreturn (TPFAIL, 0, NULL, 0, 0);
                else
                   tpreturn (TPSUCCESS, 0, NULL,
0, 0);
                break;

    case TRANSTO:
                stoinfo = (struct stostruct *) msg-
>data;
                if (TPCsto (stoinfo))
                   tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
                break;
    }
}

TPCC18(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
                newinfo = (struct newstruct *)info;
                if (TPCnew (newinfo))
                   tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
                break;

    case TRANPAY:
                payinfo = (struct paystruct *)info;
                if (TPCpay (payinfo))
                   tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
                break;

    case TRANORD:
                ordinfo = (struct ordstruct *)info;
                if (TPCord (ordinfo))
                   tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
                break;

    case TRANDEL:
                delinfo = (struct delstruct *)info;
                if (TPCdel (delinfo))
                   tpreturn (TPFAIL, 0, NULL, 0, 0);
                else
                   tpreturn (TPSUCCESS, 0, NULL,
0, 0);
                break;

    case TRANSTO:
                stoinfo = (struct stostruct *) msg-
>data;
                if (TPCsto (stoinfo))
                   tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
                break;
    }
}

TPCC19(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
                newinfo = (struct newstruct *)info;
                if (TPCnew (newinfo))
                   tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
                break;

    case TRANPAY:
                payinfo = (struct paystruct *)info;
                if (TPCpay (payinfo))
                   tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
                break;

    case TRANORD:
                ordinfo = (struct ordstruct *)info;
                if (TPCord (ordinfo))
                   tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
                break;

    case TRANDEL:
                delinfo = (struct delstruct *)info;
                if (TPCdel (delinfo))
                   tpreturn (TPFAIL, 0, NULL, 0, 0 );
                else
                   tpreturn (TPSUCCESS, 0, NULL,
0, 0);
                break;

    case TRANSTO:
                stoinfo = (struct stostruct *) msg-
>data;
                if (TPCsto (stoinfo))
                   tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
                break;
    }
}

TPCC20(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
                newinfo = (struct newstruct *)info;
                if (TPCnew (newinfo))
                   tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
                break;

    case TRANPAY:
                payinfo = (struct paystruct *)info;
                if (TPCpay (payinfo))
                   tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
                break;

    case TRANORD:
                ordinfo = (struct ordstruct *)info;
                if (TPCord (ordinfo))
                   tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
                else
                   tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
                break;

    case TRANDEL:
                delinfo = (struct delstruct *)info;
                if (TPCdel (delinfo))
                   tpreturn (TPFAIL, 0, NULL, 0, 0);
                else
                   tpreturn (TPSUCCESS, 0, NULL,
0, 0);
                break;

    case TRANSTO:
                stoinfo = (struct stostruct *) msg-
>data;
                if (TPCsto (stoinfo))
                   tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
```

```
            else
                tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
            break;
    }
}

TPCC21(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS , 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
                tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
            break;
    }
}

TPCC22(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
                tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
            break;
    }
}

TPCC23(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
            break;

    case TRANORD:
            ordinfo = (struct ordstruct *)info;
            if (TPCord (ordinfo))
                tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
            break;

    case TRANDEL:
            delinfo = (struct delstruct *)info;
            if (TPCdel (delinfo))
                tpreturn (TPFAIL, 0, NULL, 0, 0);
            else
                tpreturn (TPSUCCESS, 0, NULL,
0, 0);
            break;

    case TRANSTO:
            stoinfo = (struct stostruct *) msg-
>data;
            if (TPCsto (stoinfo))
                tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
            else
                tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
            break;
    }
}

TPCC24(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
            newinfo = (struct newstruct *)info;
            if (TPCnew (newinfo))
                tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
            else
                tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
            break;

    case TRANPAY:
            payinfo = (struct paystruct *)info;
            if (TPCpay (payinfo))
                tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
            else
```

```
                tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
                break;

    case TRANORD:
                ordinfo = (struct ordstruct *)info;
                if (TPCord (ordinfo))
                  tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
                else
                  tpreturn (TPSUCCESS, 0, ordinfo,
sizeof (struct ordstruct), 0);
                break;

    case TRANDEL:
                delinfo = (struct delstruct *)info;
                if (TPCdel (delinfo))
                  tpreturn (TPFAIL, 0, NULL, 0, 0);
                else
                  tpreturn (TPSUCCESS,  0, NULL,
0, 0);
                break;

    case TRANSTO:
                stoinfo = (struct stostruct *) msg-
>data;
                if (TPCsto (stoinfo))
                  tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
                else
                  tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
                break;
    }
}

TPCC25(msg)
TPSVCINFO *msg;
{
    info = (union infostruct *)msg->data;
    switch(info->tran_kind)
    {
    case TRANNEW:
                newinfo = (struct newstruct *)info;
                if (TPCnew (newinfo))
                  tpreturn (TPFAIL, 0, newinfo, sizeof
(struct newstruct), 0);
                else
                  tpreturn (TPSUCCESS, 0, newinfo,
sizeof (struct newstruct), 0);
                break;

    case TRANPAY:
                payinfo = (struct paystruct *)info;
                if (TPCpay (payinfo))
                  tpreturn (TPFAIL, 0, payinfo, sizeof
(struct paystruct), 0);
                else
                  tpreturn (TPSUCCESS, 0, payinfo,
sizeof (struct paystruct), 0);
                break;

    case TRANORD:
                ordinfo = (struct ordstruct *)info;
                if (TPCord (ordinfo))
                  tpreturn (TPFAIL, 0, ordinfo, sizeof
(struct ordstruct), 0);
                else
                  tpreturn (TPSUCCESS, 0,  ordinfo,
sizeof (struct ordstruct), 0);
                break;

    case TRANDEL:
                delinfo = (struct delstruct *)info;
                if (TPCdel (delinfo))
                  tpreturn (TPFAIL, 0, NULL, 0, 0);
                else
                  tpreturn (TPSUCCESS, 0, NULL,
0, 0);
                break;

    case TRANSTO:
                stoinfo = (struct stostruct *) msg-
>data;
                if (TPCsto (stoinfo))
                  tpreturn (TPFAIL, 0, (char *)stoinfo,
sizeof (struct stostruct), 0);
                else
                  tpreturn (TPSUCCESS, 0, (char
*)stoinfo, sizeof (struct stostruct), 0);
                break;
    }
}
#else

NEWORDER (msg)
TPSVCINFO *msg;
{
  newinfo = (struct newstruct *) msg->data;
  if (TPCnew (newinfo))
    tpreturn (TPFAIL, 0, newinfo, sizeof (struct
newstruct), 0);
  else
    tpreturn (TPSUCCESS, 0, newinfo, sizeof
(struct newstruct), 0);
}

PAYMENT (msg)
TPSVCINFO *msg;
{
  payinfo = (struct paystruct *) msg->data;
  if (TPCpay (payinfo))
    tpreturn (TPFAIL, 0, payinfo, sizeof (struct
paystruct), 0);
  else
    tpreturn (TPSUCCESS, 0, payinfo, sizeof
(struct paystruct), 0);
}

ORDERSTATUS (msg)
TPSVCINFO *msg;
{
  ordinfo = (struct ordstruct *) msg->data;
  if (TPCord (ordinfo))
    tpreturn (TPFAIL, 0, ordinfo, sizeof (struct
ordstruct), 0);
  else
    tpreturn (TPSUCCESS, 0, ordinfo, sizeof
(struct ordstruct), 0);
}

DELIVERY (msg)
TPSVCINFO *msg;
{
  delinfo = (struct delstruct *) msg->data;
  if (TPCdel (delinfo))
    tpreturn (TPFAIL, 0, NULL, 0, 0);
  else
    tpreturn (TPSUCCESS, 0, NULL, 0, 0);
}

STOCKLEVEL (msg)
TPSVCINFO *msg;
{
  stoinfo = (struct stostruct *) msg->data;
  if (TPCsto (stoinfo))
    tpreturn (TPFAIL, 0, stoinfo, sizeof (struct
stostruct), 0);
  else
    tpreturn (TPSUCCESS, 0, stoinfo, sizeof
(struct stostruct), 0);
}
#endif




*****************************************
            Makefile
*****************************************



#
#====================================
============================+
#       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
#

I_SYM=-I
INCLUDE=$(I_SYM).
$(I_SYM)$(ORACLE_HOME)/rdbms/demo
ITUX=$(I_SYM)$(ROOTDIR)/include
ARLOCAL=
AR=ar
ARCREATE=$(AR) cr$(ARLOCAL)
```

```
DOAR=$(ARCREATE) $@ $?
#CC=oraxlc -F$(ORACLE_HOME)/bench/xlc.cfg
#CC=cc -F$(ORACLE_HOME)/bench/xlc.cfg
CC=cc
#CFLAGS=-O
CFLAGS=-O
#LD=ld
LD=cc
#LDFLAGS=-H512 -T512 -bhalt:4
#LDOBJS=/lib/crt0.o -lc
#  $(ORACLE_HOME)/rdbms/lib/epcni.o \
#  $(ORACLE_HOME)/rdbms/lib/epcni.o \

SGL_TASK_FLAGS= -
bI:$(ORACLE_HOME)/lib/mili.exp \
  -bI:$(ORACLE_HOME)/lib/pw-syscall.exp -
bI:${ORACLE_HOME)/lib/ksms.imp
SGL_TASK_LIB=
$(ORACLE_HOME)/lib/osntabst.o
${ORACLE_HOME}/lib/config.o \
  $(ORACLE_HOME)/rdbms/lib/osnsgl.o -lbench
$(ORACLE_HOME)/lib/libepc.a \
  -lclient -lserver -lcommon -lgeneric -lknlopt \
  -lapps -lcog -lcox -lidl -lknlde -lpkg -lpls -lsem -
lsyn \
  -lclient -lserver -lcommon -lgeneric -lknlopt \
  -lapps -lcog -lcox -lidl -lknlde -lpkg -lpls -lsem -
lsyn \
  -lclient -lserver -lcommon -lgeneric \
  -lapps -lcog -lcox -lidl -lknlde -lpkg -lpls -lsem -
lsyn \
  -lsqlnet -lclient -lserver -lcommon -lgeneric \
  -lsqlnet -lclient -lserver -lcommon -lgeneric \
  -lnlsrtl3 -lc3v6 -lcore3 -lnlsrtl3 -lcore3 \
  'cat $(ORACLE_HOME)/rdbms/lib/psoliblist' \
  -lm -lld -lm -lc
#TWO_TASK_FLAGS= -bI
$(ORACLE_HOME)/lib/mili.exp
TWO_TASK_FLAGS=
TWO_TASK_LIB=-lbench \
  -lsqlnet $(ORACLE_HOME)/lib/libncr.a \
  -lclient -lserver -lcommon -lgeneric \
  -lsqlnet $(ORACLE_HOME)/lib/libncr.a \
  -lsqlnet \
  -lclient -lserver -lcommon -lgeneric \
  $(ORACLE_HOME)/lib/libepc.a \
  -lnlsrtl3 -lc3v6 -lcore3 -lnlsrtl3 -lcore3 \
  -lm -lsocket -lm -lc -lnsl

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

TPCBIN=$(ORACLE_HOME)/bench/tpc/bin

all: compile load setup

compile: $(OBJS)

load: tpccload.ott tpcc.ott getrand 90per

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 -f 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 -f 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 -f 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
$(INCLUDE) $(ITUX) -c tpccpl_tux.c
        rm -f 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 -f 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)
$(TWO_TASK_FLAGS) -o $@ getrand.o

90per: 90per.o
        $(CC) $(CFLAGS)
$(TWO_TASK_FLAGS) -o $@ 90per.o


tpccload.ott: tpccload.o
        $(LD) $(LDFLAGS) -o $@ \
        $(TWO_TASK_FLAGS) -
L$(ORACLE_HOME)/lib -
L$(ORACLE_HOME)/rdbms/lib \
        tpccload.o \
        $(TWO_TASK_LIB) $(LDOBJS)


tpcc.ott: c_drv.o c_trans.o tpccpl.o c_dump.o
        $(LD) $(LDFLAGS) -o $@ \
        $(TWO_TASK_FLAGS) -
L$(ORACLE_HOME)/lib -
L$(ORACLE_HOME)/rdbms/lib \
        c_drv.o c_trans.o tpccpl.o c_dump.o \
        $(TWO_TASK_LIB) $(LDOBJS)

test_drv: c_drv_val.o test_drv.o c_dump.o
        $(LD) $(LDFLAGS) -o $@ \
        $(TWO_TASK_FLAGS) \
        -L$(ORACLE_HOME)/lib -
L$(ORACLE_HOME)/rdbms/lib \
        c_drv_val.o test_drv.o c_dump.o \
        -lbench -lm $(LDOBJS)

test_sample: c_drv.o test_sample.o c_dump.o
        $(LD) $(LDFLAGS) -o $@ \
        $(TWO_TASK_FLAGS) \
        -L$(ORACLE_HOME)/lib -
L$(ORACLE_HOME)/rdbms/lib \
```

```
                c_drv.o test_sample.o c_dump.o \
                -lbench -lm $(LDOBJS)

test_tran.ott: test_tran.o c_trans.o tpccpl.o
c_dump.o
                $(LD) $(LDFLAGS) -o $@ \
                $(TWO_TASK_FLAGS) -
L$(ORACLE_HOME)/lib -
L$(ORACLE_HOME)/rdbms/lib \
                test_tran.o c_trans.o tpccpl.o
c_dump.o \
                $(TWO_TASK_LIB) $(LDOBJS)

tpcccli: c_drv_tux.o c_dump.o
                (CFLAGS="$(CFLAGS)"; export
CFLAGS; CC=$(CC); export CC; \
                 buildclient -v -o $@ -f '-
L$(ORACLE_HOME)/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 $@ \
                  -s
NEWORDER,PAYMENT,ORDERSTATUS,DELIV
ERY,STOCKLEVEL \
                  -f '$(TWO_TASK_FLAGS)' \
                  -f '-L$(ORACLE_HOME)/lib -
L$(ORACLE_HOME)/rdbms/lib' \
                  -f 'tpccsvr.o c_trans_tux.o
tpccpl_tux.o' \
                  -l '$(TWO_TASK_LIB)')

setup:  tpcc.ott getrand 90per
                rm -f $(TPCBIN)/tpccload
                (cd $(TPCBIN); ln -s
../tpcc/source/tpccload.ott tpccload)
                rm -f $(TPCBIN)/tpcc.ott
                (cd $(TPCBIN); ln -s
../tpcc/source/tpcc.ott .)
                rm -f $(TPCBIN)/getrand
                (cd $(TPCBIN); ln -s
../tpcc/source/getrand .)
                rm -f $(TPCBIN)/90per
                (cd $(TPCBIN); ln -s
../tpcc/source/90per .)
```

Appendix C: RTE Scripts

```
****************************************
      tpcC.conf (No.1)
****************************************


#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
            STARTRTE
                        RTEHOST = ptnfm02
                        STARTSUT
                                    SUTHOST
= ptnds08a,400
                                    SUTLOGIN
= oracle

            SUTPASSWD = oracle
                                    SUTCMD
= Tc
                        ENDSUT
            ENDRTE
#           STRCMD    = tpcCstartCmdSH
#           TSCOM     = tpcCtscomSH
#           TECOM     = tpcCtecomSH
            LOGOUT    = NONE
            LOGMODE   = ALL
            LOGCOMMENT= COMOFF
            LOGFILE   = tpcC.log
            SIMFILE   = ../data/tpcc.pps
            PROTCOL   = telnet,9237
#WAREHOUSE SCALE
            VAL             = U1I = 400
#RAMP-UP TIME
            VAL             = U2I = 1800
#MEASUERMENT TIME
            VAL             = U3I = 1800
#RAMP-DOWN TIME
            VAL             = U4I = 1800
#NEW THINKTIME (msec)
            VAL             = U5I = 12100
#PAY THINKTIME (msec)
            VAL             = U6I = 12100
#
            VAL             = U7I = 0
            VAL             = U8I = 0
            VAL             = U9I = 0
#
#ORD THINKTIME (msec)
            VAL             = U10I= 10250
#DEL THINKTIME (msec)
            VAL             = U11I= 5100
#STK THINKTIME (msec)
            VAL             = U12I= 5100
#NURAND CONSTANT c_id
            VAL             = U13I= 0
#NURAND CONSTANT c_last
            VAL             = U14I= 0
#NURAND CONSTANT ol_i_id
            VAL             = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#           VAL             = U16I= 2
            VAL             = U16I= 1
#NEW KEYING-TIME (msec)
            VAL             = U17I = 18300
#PAY KEYING-TIME (msec)
            VAL             = U18I = 3030
#ORD KEYING-TIME (msec)
            VAL             = U19I= 2030
#DEL KEYING-TIME (msec)
            VAL             = U20I= 2030
#STK KEYING-TIME (msec)
            VAL             = U21I= 2030
ENDGROUP



****************************************
      tpcC.conf (No.2)
****************************************


#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
            STARTRTE
                        RTEHOST = ptnfm03
                        STARTSUT
                                    SUTHOST
= ptnds01b,400
                                    SUTLOGIN
= oracle

            SUTPASSWD = oracle
                                    SUTCMD
= Tc
                        ENDSUT
            ENDRTE
#           STRCMD    = tpcCstartCmdSH
#           TSCOM     = tpcCtscomSH
#           TECOM     = tpcCtecomSH
            LOGOUT    = NONE
            LOGMODE   = ALL
            LOGCOMMENT= COMOFF
            LOGFILE   = tpcC.log
            SIMFILE   = ../data/tpcc.pps
            PROTCOL   = telnet,9237
#WAREHOUSE SCALE
            VAL             = U1I = 400
#RAMP-UP TIME
            VAL             = U2I = 1800
#MEASUERMENT TIME
            VAL             = U3I = 1800
#RAMP-DOWN TIME
            VAL             = U4I = 1800
#NEW THINKTIME (msec)
            VAL             = U5I = 12100
#PAY THINKTIME (msec)
            VAL             = U6I = 12100
#
            VAL             = U7I = 0
            VAL             = U8I = 0
            VAL             = U9I = 0
#
#ORD THINKTIME (msec)
            VAL             = U10I= 10250
#DEL THINKTIME (msec)
            VAL             = U11I= 5100
#STK THINKTIME (msec)
            VAL             = U12I= 5100
#NURAND CONSTANT c_id
            VAL             = U13I= 0
#NURAND CONSTANT c_last
            VAL             = U14I= 0
#NURAND CONSTANT ol_i_id
            VAL             = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#           VAL             = U16I= 2
            VAL             = U16I= 1
#NEW KEYING-TIME (msec)
            VAL             = U17I = 18300
#PAY KEYING-TIME (msec)
            VAL             = U18I = 3030
#ORD KEYING-TIME (msec)
            VAL             = U19I= 2030
#DEL KEYING-TIME (msec)
            VAL             = U20I= 2030
#STK KEYING-TIME (msec)
            VAL             = U21I= 2010
ENDGROUP



****************************************
      tpcC.conf (No.3)
****************************************


#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
            STARTRTE
                        RTEHOST = ptnfm02
                        STARTSUT
                                    SUTHOST
= ptnds08b,400
                                    SUTLOGIN
= oracle

            SUTPASSWD = oracle
                                    SUTCMD
= Tc
                        ENDSUT
            ENDRTE
#           STRCMD    = tpcCstartCmdSH
#           TSCOM     = tpcCtscomSH
#           TECOM     = tpcCtecomSH
            LOGOUT    = NONE
            LOGMODE   = ALL
            LOGCOMMENT= COMOFF
            LOGFILE   = tpcC.log
```

```
            SIMFILE   = ../data/tpcc.pps
            PROTCOL   = telnet,9237
#WAREHOUSE SCALE
            VAL       = U1I = 400
#RAMP-UP TIME
            VAL       = U2I = 1800
#MEASUERMENT TIME
            VAL       = U3I = 1800
#RAMP-DOWN TIME
            VAL       = U4I = 1800
#NEW THINKTIME (msec)
            VAL       = U5I = 12100
#PAY THINKTIME (msec)
            VAL       = U6I = 12100
#
            VAL       = U7I = 0
            VAL       = U8I = 0
            VAL       = U9I = 0
#
#ORD THINKTIME (msec)
            VAL       = U10I= 10250
#DEL THINKTIME (msec)
            VAL       = U11I= 5100
#STK THINKTIME (msec)
            VAL       = U12I= 5100
#NURAND CONSTANT c_id
            VAL       = U13I= 0
#NURAND CONSTANT c_last
            VAL       = U14I= 0
#NURAND CONSTANT ol_i_id
            VAL       = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#           VAL       = U16I= 2
            VAL       = U16I= 1
#NEW KEYING-TIME (msec)
            VAL       = U17I = 18300
#PAY KEYING-TIME (msec)
            VAL       = U18I = 3030
#ORD KEYING-TIME (msec)
            VAL       = U19I= 2030
#DEL KEYING-TIME (msec)
            VAL       = U20I= 2030
#STK KEYING-TIME (msec)
            VAL       = U21I= 2030
ENDGROUP




****************************************
      tpcC.conf (No.4)
****************************************



#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
            STARTRTE
                        RTEHOST = ptnfm02
                        STARTSUT
                                    SUTHOST
= ptnds07a,400
                                    SUTLOGIN
= oracle

            SUTPASSWD = oracle
                                    SUTCMD
= Tc
                        ENDSUT
            ENDRTE
#           STRCMD    = tpcCstartCmdSH
#           TSCOM     = tpcCtscomSH
#           TECOM     = tp cCtecomSH
            LOGOUT    = NONE
            LOGMODE   = ALL
            LOGCOMMENT= COMOFF
            LOGFILE   = tpcC.log
            SIMFILE   = ../data/tpcc.pps
            PROTCOL   = telnet,9237
#WAREHOUSE SCALE
            VAL       = U1I = 400
#RAMP-UP TIME
            VAL       = U2I = 1800
#MEASUERMENT TIME
            VAL       = U3I = 1800
#RAMP-DOWN TIME
            VAL       = U4I = 1800
#NEW THINKTIME (msec)
            VAL       = U5I = 12100
#PAY THINKTIME (msec)
            VAL       = U6I = 12100
#
            VAL       = U7I = 0
            VAL       = U8I = 0
            VAL       = U9I = 0
#
#ORD THINKTIME (msec)
            VAL       = U10I= 10250
#DEL THINKTIME (msec)
            VAL       = U11I= 5100
#STK THINKTIME (msec)
            VAL       = U12I= 5100
#NURAND CONSTANT c_id
            VAL       = U13I= 0
#NURAND CONSTANT c_last
            VAL       = U14I= 0
#NURAND CONSTANT ol_i_id
            VAL       = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#           VAL       = U16I= 2
            VAL       = U16I= 1
#NEW KEYING-TIME (msec)
            VAL       = U17I = 18300
#PAY KEYING-TIME (msec)
            VAL       = U18I = 3030
#ORD KEYING-TIME (msec)
            VAL       = U19I= 2030
#DEL KEYING-TIME (msec)
            VAL       = U20I= 2030
#STK KEYING-TIME (msec)
            VAL       = U21I= 2030
ENDGROUP




****************************************
      tpcC.conf (No.5)
****************************************



#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
            STARTRTE
                        RTEHOST = ptnfm02
                        STARTSUT
                                    SUTHOST
= ptnds07b,400
                                    SUTLOGIN
= oracle

            SUTPASSWD = oracle
                                    SUTCMD
= Tc
                        ENDSUT
            ENDRTE
#           STRCMD    = tpcCstartCmdSH
#           TSCOM     = tpcCtscomSH
#           TECOM     = tpcCtecomSH
            LOGOUT    = NONE
            LOGMODE   = ALL
            LOGCOMMENT= COMOFF
            LOGFILE   = tpcC.log
            SIMFILE   = ../data/tpcc.pps
            PROTCOL   = telnet,9237
#WAREHOUSE SCALE
            VAL       = U1I = 400
#RAMP-UP TIME
            VAL       = U2I = 1800
#MEASUERMENT TIME
            VAL       = U3I = 1800
#RAMP-DOWN TIME
            VAL       = U4I = 1800
#NEW THINKTIME (msec)
            VAL       = U5I = 12100
#PAY THINKTIME (msec)
            VAL       = U6I = 12100
#
            VAL       = U7I = 0
            VAL       = U8I = 0
            VAL       = U9I = 0
#
#ORD THINKTIME (msec)
            VAL       = U10I= 10250
#DEL THINKTIME (msec)
            VAL       = U11I= 5100
#STK THINKTIME (msec)
            VAL       = U12I= 5100
#NURAND CONSTANT c_id
            VAL       = U13I= 0
#NURAND CONSTANT c_last
            VAL       = U14I= 0
#NURAND CONSTANT ol_i_id
            VAL       = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#           VAL       = U16I= 2
            VAL       = U16I= 1
#NEW KEYING-TIME (msec)
            VAL       = U17I = 18300
#PAY KEYING-TIME (msec)
            VAL       = U18I = 3030
#ORD KEYING-TIME (msec)
            VAL       = U19I= 2030
#DEL KEYING-TIME (msec)
            VAL       = U20I= 2030
#STK KEYING-TIME (msec)
```

```
        VAL             = U21I= 2030
ENDGROUP




****************************************
      tpcC.conf (No.6)
****************************************


#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
        STARTRTE
                RTEHOST = ptnfm01
                STARTSUT
                        SUTHOST
= ptnds06a,400
                        SUTLOGIN
= oracle

        SUTPASSWD = oracle
                        SUTCMD
= Tc
                ENDSUT
        ENDRTE
#       STRCMD    = tpcCstartCmdSH
#       TSCOM     = tpcCtscomSH
#       TECOM     = tpcCtecomSH
        LOGOUT    = NONE
        LOGMODE   = ALL
        LOGCOMMENT= COMOFF
        LOGFILE   = tpcC.log
        SIMFILE   = ../data/tpcc.pps
        PROTCOL   = telnet,9237
#WAREHOUSE SCALE
        VAL             = U1I = 400
#RAMP-UP TIME
        VAL             = U2I = 1800
#MEASUERMENT TIME
        VAL             = U3I = 1800
#RAMP-DOWN TIME
        VAL             = U4I = 1800
#NEW THINKTIME (msec)
        VAL             = U5I = 12100
#PAY THINKTIME (msec)
        VAL             = U6I = 12100
#
        VAL             = U7I = 0
        VAL             = U8I = 0
        VAL             = U9I = 0
#
#ORD THINKTIME (msec)
        VAL             = U10I= 10250
#DEL THINKTIME (msec)
        VAL             = U11I= 5100
#STK THINKTIME (msec)
        VAL             = U12I= 5100
#NURAND CONSTANT c_id
        VAL             = U13I= 0
#NURAND CONSTANT c_last
        VAL             = U14I= 0
#NURAND CONSTANT ol_i_id
        VAL             = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#       VAL             = U16I= 2
        VAL             = U16I= 1
#NEW KEYING-TIME (msec)
        VAL             = U17I = 18300
#PAY KEYING-TIME (msec)
        VAL             = U18I = 3030
#ORD KEYING-TIME (msec)
        VAL             = U19I= 2030
#DEL KEYING-TIME (msec)
        VAL             = U20I= 2030
#STK KEYING-TIME (msec)
        VAL             = U21I= 2030
ENDGROUP




****************************************
      tpcC.conf (No.7)
****************************************


#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
        STARTRTE
                RTEHOST = ptnfm01
                STARTSUT
                        SUTHOST
= ptnds05a,400
                        SUTLOGIN
= oracle

        SUTPASSWD = oracle
                        SUTCMD
= Tc
                ENDSUT
        ENDRTE
#       STRCMD    = tpcCst artCmdSH
#       TSCOM     = tpcCtscomSH
#       TECOM     = tpcCtecomSH
        LOGOUT    = NONE
        LOGMODE   = ALL
        LOGCOMMENT= COMOFF
        LOGFILE   = tpcC.log
        SIMFILE   = ../data/tpcc.pps
        PROTCOL   = telnet,9237
#WAREHOUSE SCALE
        VAL             = U1I = 400
#RAMP-UP TIME
        VAL             = U2I = 1800
#MEASUERMENT TIME
        VAL             = U3I = 1800
#RAMP-DOWN TIME
        VAL             = U4I = 1800
#NEW THINKTIME (msec)
        VAL             = U5I = 12100
#PAY THINKTIME (msec)
        VAL             = U6I = 12100
#
        VAL             = U7I = 0
        VAL             = U8I = 0
        VAL             = U9I = 0
#
#ORD THINKTIME (msec)
        VAL             = U10I= 10250
#DEL THINKTIME (msec)
        VAL             = U11I= 5100
#STK THINKTIME (msec)
        VAL             = U12I= 5100
#NURAND CONSTANT c_id
        VAL             = U13I= 0
#NURAND CONSTANT c_last
        VAL             = U14I= 0
#NURAND CONSTANT ol_i_id
        VAL             = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#       VAL             = U16I= 2
        VAL             = U16I= 1
#NEW KEYING-TIME (msec)
        VAL             = U17I = 18300
#PAY KEYING-TIME (msec)
        VAL             = U18I = 3030
#ORD KEYING-TIME (msec)
        VAL             = U19I= 2030
#DEL KEYING-TIME (msec)
        VAL             = U20I= 2030
#STK KEYING-TIME (msec)
        VAL             = U21I= 2030
ENDGROUP




****************************************
      tpcC.conf (No.8)
****************************************


#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
        STARTRTE
                RTEHOST = ptnfm03
                STARTSUT
                        SUTHOST
= ptnds02a,400
                        SUTLOGIN
= oracle

        SUTPASSWD = oracle
                        SUTCMD
= Tc
                ENDSUT
        ENDRTE
#       STRCMD    = tpcCstartCmdSH
#       TSCOM     = tpcCtscomSH
#       TECOM     = tpcCtecomSH
        LOGOUT    = NONE
        LOGMODE   = ALL
        LOGCOMMENT= COMOFF
        LOGFILE   = tpcC.log
        SIMFILE   = ../ data/tpcc.pps
        PROTCOL   = telnet,9237
#WAREHOUSE SCALE
        VAL             = U1I = 400
#RAMP-UP TIME
        VAL             = U2I = 1800
#MEASUERMENT TIME
```

```
                VAL             = U3I = 1800
#RAMP-DOWN TIME
                VAL             = U4I = 1800
#NEW THINKTIME (msec)
                VAL             = U5I = 12100
#PAY THINKTIME (msec)
                VAL             = U6I = 12100
#
                VAL             = U7I = 0
                VAL             = U8I = 0
                VAL             = U9I = 0
#
#ORD THINKTIME (msec)
                VAL             = U10I= 10250
#DEL THINKTIME (msec)
                VAL             = U11I= 5100
#STK THINKTIME (msec)
                VAL             = U12I= 5100
#NURAND CONSTANT c_id
                VAL             = U13I= 0
#NURAND CONSTANT c_last
                VAL             = U14I= 0
#NURAND CONSTANT ol_i_id
                VAL             = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#               VAL             = U16I= 2
                VAL             = U16I= 1
#NEW KEYING-TIME (msec)
                VAL             = U17I = 18300
#PAY KEYING-TIME (msec)
                VAL             = U18I = 3030
#ORD KEYING-TIME (msec)
                VAL             = U19I= 2030
#DEL KEYING-TIME (msec)
                VAL             = U20I= 2030
#STK KEYING-TIME (msec)
                VAL             = U21I= 2010
ENDGROUP




*****************************************
        tpcC.conf (No.9)
*****************************************


#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
                STARTRTE
                                RTEHOST = ptnfm03
                                STARTSUT
                                                SUTHOST
= ptnds02b,400
                                                SUTLOGIN
= oracle

                SUTPASSWD = oracle
                                                SUTCMD
= Tc
                                ENDSUT
                ENDRTE
#               STRCMD    = tpcCstartCmdSH
#               TSCOM     = tpcCtscomSH
#               TECOM     = tpcCtecomSH
                LOGOUT    = NONE
                LOGMODE   = ALL
                LOGCOMMENT= COMOFF
                LOGFILE   = tpcC.log
                SIMFILE   = ../data/tpcc.pps
                PROTCOL   = telnet,9237
#WAREHOUSE SCALE
                VAL             = U1I = 400
#RAMP-UP TIME
                VAL             = U2I = 1800
#MEASUERMENT TIME
                VAL             = U3I = 1800
#RAMP-DOWN TIME
                VAL             = U4I = 1800
#NEW THINKTIME (msec)
                VAL             = U5I = 12100
#PAY THINKTIME (msec)
                VAL             = U6I = 12100
#
                VAL             = U7I = 0
                VAL             = U8I = 0
                VAL             = U9I = 0
#
#ORD THINKTIME (msec)
                VAL             = U10I= 10250
#DEL THINKTIME (msec)
                VAL             = U11I= 5100
#STK THINKTIME (msec)
                VAL             = U12I= 5100
#NURAND CONSTANT c_id
                VAL             = U13I= 0
#NURAND CONSTANT c_last
                VAL             = U14I= 0
#NURAND CONSTANT ol_i_id
                VAL             = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#               VAL             = U16I= 2
                VAL             = U16I= 1
#NEW KEYING-TIME (msec)
                VAL             = U17I = 18300
#PAY KEYING-TIME (msec)
                VAL             = U18I = 3030
#ORD KEYING-TIME (msec)
                VAL             = U19I= 2030
#DEL KEYING-TIME (msec)
                VAL             = U20I= 2030
#STK KEYING-TIME (msec)
                VAL             = U21I= 2010
ENDGROUP




*****************************************
        tpcC.conf (No.10)
*****************************************


#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
                STARTRTE
                                RTEHOST = ptnfm03
                                STARTSUT
                                                SUTHOST
= ptnds01a,400
                                                SUTLOGIN
= oracle

                SUTPASSWD = oracle
                                                SUTCMD
= Tc
                                ENDSUT
                ENDRTE
#               STRCMD    = tpcCstartCmdSH
#               TSCOM     = tpcCtscomSH
#               TECOM     = tpcCtecomSH
                LOGOUT    = NONE
                LOGMODE   = ALL
                LOGCOMMENT= COMOFF
                LOGFILE   = tpcC.log
                SIMFILE   = ../data/tpcc.pps
                PROTCOL   = telnet,9237
#WAREHOUSE SCALE
                VAL             = U1I = 400
#RAMP-UP TIME
                VAL             = U2I = 1800
#MEASUERMENT TIME
                VAL             = U3I = 1800
#RAMP-DOWN TIME
                VAL             = U4I = 1800
#NEW THINKTIME (msec)
                VAL             = U5I = 12100
#PAY THINKTIME (msec)
                VAL             = U6I = 12100
#
                VAL             = U7I = 0
                VAL             = U8I = 0
                VAL             = U9I = 0
#
#ORD THINKTIME (msec)
                VAL             = U10I= 10250
#DEL THINKTIME (msec)
                VAL             = U11I= 5100
#STK THINKTIME (msec)
                VAL             = U12I= 5100
#NURAND CONSTANT c_id
                VAL             = U13I= 0
#NURAND CONSTANT c_last
                VAL             = U14I= 0
#NURAND CONSTANT ol_i_id
                VAL             = U15I= 0
#MSG  OFF:0, Each Term:1, Field:2
#               VAL             = U16I= 2
                VAL             = U16I= 1
#NEW KEYING-TIME (msec)
                VAL             = U17I = 18300
#PAY KEYING-TIME (msec)
                VAL             = U18I = 3030
#ORD KEYING-TIME (msec)
                VAL             = U19I= 2030
#DEL KEYING-TIME (msec)
                VAL             = U20I= 2030
#STK KEYING-TIME (msec)
                VAL             = U21I= 2010
ENDGROUP
```

Appendix D: System Tunables

```
****************************************
        async
****************************************

*#ident    "@(#)async.cf         6.1.1.1
9/16/96 15:57:48"
*
* ASYNC
*
*FLAG  #VEC   PREFIX  SOFT   #DEV   IPL
DEPENDENCIES/VARIABLES
csn        -        aio_       36        -
        -

        numaio(%i) = {NUMAIO}
$$
* NUMBER OF AIO CONTROL BLOCKS
NUMAIO = 3000

****************************************
        kernel
****************************************

* All Rights Reserved, Copyright (c) PFU &
FUJITSU LIMITED 1996
*#ident    "@(#)kernel.cf         6.1.7.1
96/11/22 DS"
*
* KERNEL
*
*FLAG     #VEC      PREFIX     SOFT
          #DEV      IPL
          DEPENDENCIES/VARIABLES
none       -          -          -
-          -

        v(%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i
%i%i%i%i%i%i%i%i%i) = {

                NBUF,

                NCALL,

                NPROC,

                NLWP,

                0,

                MAXCLSYSPRI,

                0,

                MAXUP,

                NHBUF,

                NHBUF-1,

                NPBUF,

                0,

                MAXPMEM,

                NAUTOUP,

                BUFHWM,
* XENIX Support

                0,

                0,

                0,

                0,

                0,
* End XENIX Support

                MAXULWP,
* XXX - make into real tunables:
* v_nonexclusive

                1,
* v_max_proc_exbind: a guess only

                100,
* v_static_sq

                128

        }

        nullptr_default(%i) = { NULLPTR }

        nullptr_log(%i) = { NULLPTRLOG }


        npgoutbuf (%i) = { NPGOUTBUF }

        rstchown (%i) = { RSTCHOWN }

        rootfstype (%15c) = { ROOTFSTYPE }

        ncsize (%i) = { DNLCSIZE }


        rwood_dumpflag(%i) = { DUMPFLAG
}

        rwood_rebootflag(%i) = {
REBOOTFLAG }

        maxusrlarge (%i) = { MAXUSRLARGE
}

        usrtxtrsvmem (%i) = {
USRTXTRSVMEM }

        usrbssrsvmem (%i) = {
USRBSSRSVMEM }

        brkrsvmem (%i) = { BRKRSVMEM }

        usrtxtrsvmin (%i) = {
USRTXTRSVMIN }

        usrbssrsvmin (%i) = {
USRBSSRSVMIN }

        brkrsvmin (%i) = { BRKRSVMIN }

        segkmem_lock_page(%i) = {
SEGKMEM_LOCK_PAGE }
$$

*******************************************************
********************
* kernel tunable parameters
*
* NCALL   - number of callout (timeout) entries
* NPROC   - max number of processes system
wide
* NLWP    - max number of LWPs system wide
* MAXUP   - max number of processes per user
* ARG_MAX - maximum length of argument
strings for exec
* FLCKREC - max number of active file/record
locks system-wide
*******************************************************
********************
NCALL = 512
NPROC = 2000
NLWP = 3000
MAXUP = 512
ARG_MAX = 1048576
FLCKREC = 300

*******************************************************
********************
* Default per process resource limits (set to
0x7FFFFFFF for infinite limit)
* S prefix is for soft limits, H prefix is for hard
limits
*
* CPULIM - maximum combined user and system
time in seconds
* FSZLIM - maximum file size in bytes
* DATLIM - maximum writeable mapped memory
(swap space) in bytes
* STKLIM - maximum size of current stack in
bytes
* CORLIM - maximum size of core file in bytes
* FNOLIM - maximum number of file descriptors
* VMMLIM - maximum amount of simultaneously
mapped virtual memory in bytes
*******************************************************
********************
SCPULIM = 0x7FFFFFFF
```

```
HCPULIM = 0x7FFFFFFF
SFSZLIM = 0xFFFFFFFFFFFFFFFD
HFSZLIM = 0xFFFFFFFFFFFFFFFD
SDATLIM = 0x7FFFFFFF
HDATLIM = 0x7FFFFFFF
SSTKLIM = 0x1000000
HSTKLIM = 0x1000000
SCORLIM = 0x7FFFFFFF
HCORLIM = 0x7FFFFFFF
SFNOLIM = 0x2000
HFNOLIM = 0x2000
SVMMLIM = 0x7FFFFFFF
HVMMLIM = 0x7FFFFFFF

************************************************************
**********
* buffer cache parameters
*
* NBUF   - number of I/O buffers
* NHBUF  - number of hash buffers to allocate
* NPBUF  - number of physical I/O buffers
* BUFHWM - high-water-mark of buffer cache
memory usage, in units of K Bytes
************************************************************
**********
NBUF = 100
NHBUF = 64
NPBUF = 20
BUFHWM = 0
NPGOUTBUF = 16

************************************************************
**********
* paging parameters
*
* FSFLUSHR     - time interval in seconds at
which fsflush is run
* NAUTOUP      -
* SPTMAP       - ?
* GPGSLO       - if freemem < GPGSLO, start to
steal pages from processes
* MINARMEM     - ?
* MINASMEM     - ?
* PAGES_UNLOCK - not used
************************************************************
**********
FSFLUSHR = 10
NAUTOUP = 60
SPTMAP = 100
GPGSLO = 25
MINARMEM = 20
MINASMEM = 25
PAGES_UNLOCK = 20

************************************************************
**********
* file access feature
*
* RSTCHOWN    - multiple groups and chown(2)
restrictions
* NGROUPS_MAX - maximum number of groups
per process (default, min, max)
************************************************************
**********
RSTCHOWN = 1
NGROUPS_MAX = 16
ROOTFSTYPE = ""
DNLCSIZE = 0x800

************************************************************
**********
* streams parameters
*
* NSTRPUSH - max number of modules that can
be pushed on a stream
* STRTHRESH  - maximum bytes stream to
allocate
* STRMSGSZ - max size of the data portion of a
streams message
* STRCTLSZ - max size of the data portion of a
streams message
* STRNSCHED - Max number of service
procedures to run in any given runqueues
*                    invocation
************************************************************
**********
NSTRPUSH = 9
STRMSGSZ = 0
STRCTLSZ = 1024
STRNSCHED = 16

************************************************************
**********
* UXP/DS family-specific parameters
*
* OFFTIME  -
* SYSSEGSZ -
* FILEMAP  -
************************************************************
**********
OFFTIME = 10
SYSSEGSZ = 0
FILEMAP = 0

************************************************************
**********
* Others parameters
*
* MAXCLSYSPRI - max global priority used by
system class
* MAXPMEM     - maximum physical memory to
use.
* MAXULWP     - per-uid number of lwps limit
* NULLPTR     - Null-pointer workaround default (0
= disable, 1,2 = enable)
* NULLPTRLOG  - Null-pointer workaround default
(0 = disable, 1 = enable)
* INITCLASS   - Scheduling class of init process
* REBOOTFLAG  - Reboot after memory dump (0
= disable, 1 = enable)
* DUMPFLAG    - Memory dump control (0 =
disable, 1 = enable)
* STRCTLSZ    - max size of the data portion of a
streams message
* STRMSGSZ    - max size of the data portion of a
streams message
* PUTBUFSZ    -
************************************************************
**********
MAXCLSYSPRI = 99
MAXPMEM = 0
MAXULWP = 192
NULLPTR = 0
NULLPTRLOG = 0
INITCLASS = "TS"
REBOOTFLAG = 1
DUMPFLAG = 1
CPUTIMEMODE = 0
KDBFLAG = 0
ADJRATE = 5

************************************************************
**********
* High-speed Process memory map facility - Large
Page reserved memory size
*
* MAXUSRLARGE  - maximum number of
applications managed large page
* USRTXTRSVMEM - user text area large page
reserved memory size
* USRBSSRSVMEM - user bss area large page
reserved memory size
* BRKRSVMEM    - break area large page
reserved memory size
* USRTXTRSVMIN - user text area large page
available minmum size
* USRBSSRSVMIN - user bss area large page
available minmum size
* BRKRSVMIN    - break area large page available
minmum size
* SEGKMEM_LOCK_PAGE  - size of kernel
managed large page (mega byte)
************************************************************
**********
MAXUSRLARGE = 64
USRTXTRSVMEM = 16
USRBSSRSVMEM = 0
BRKRSVMEM = 0
USRTXTRSVMIN = 4
USRBSSRSVMIN = 4
BRKRSVMIN = 4
SEGKMEM_LOCK_PAGE = 32




****************************************
            mem
****************************************


*#ident       "@(#)mem.cf                6.1.2.1
11/22/96 19:35:18 - FUJITSU/SCCS"
*
* MEM
*
*FLAG       #VEC       PREFIX     SOFT
           #DEV       IPL
           DEPENDENCIES/VARIABLES
orx        -          kvm_       -          -
           -
*
* Kernel segment driver aging control parameters.
*

           segmap_age_time(%l) =
{SEGMAP_AGE_TIME * HZ}

           segkvn_age_time(%l) =
{SEGKVN_AGE_TIME * HZ}

           segmap_agings(%l) =
{SEGMAP_AGINGS}
```

```
	tune(%i%i%i%i%i%i%i%i) = {

		GPGSLO,

		FSFLUSHR,

		MINAMEM,

		KMEM_RESV,

		FLCKREC,

		MAXDMAPAGE,

		0,

		0 }

	pages_pp_maximum(%i) =
{PAGES_UNLOCK}

	pages_dkma_maximum(%i) =
{PAGES_NODISKMA}

	scale_maxpgio(%i) =
{SCALE_MAXPGIO}

	deficit_age(%i) = {DEFICIT_AGE}

	io_weight(%i) = {IO_WEIGHT}

	cpu_weight(%i) = {CPU_WEIGHT}

	swap_weight(%i) = {SWAP_WEIGHT}

	sleep_weight(%i) =
{SLEEP_WEIGHT}

	maxslp(%i) = {MAXSLP}

	swap_maxdev(%i) =
{SWAP_MAXDEV}
*
* Miscellaneous Aging Parameters
*

* Elapsed time aging: interval under memory
stress
	et_age_interval_fast(%l) =
{ET_AGE_INTERVAL * HZ}

* Maximum permitted value for short term deficit
due to swapins
	max_deficit(%i) = {MAX_DEFICIT}

* Minimum number of nonlocked pages a process
must have, for getting aged
	nonlocked_minpg(%i) =
{NONLOCKED_MINPG}
	maxrss(%i) = {MAXRSS}

* The aging quanta defined below are in units of
clock ticks *
	init_agequantum(%l) =
{INIT_AGEQUANTUM}
	min_agequantum(%l) =
{MIN_AGEQUANTUM}
	max_agequantum(%l) =
{MAX_AGEQUANTUM}

*
* Threshold RSS growth rates (in units of pages
over RSS sampling period)
* for performing growth rate based short term
aging quantum adjustment.
*
	lo_grow_rate(%i) =
{LO_GROW_RATE}
	hi_grow_rate(%i) =
{HI_GROW_RATE}
*
* The following are kernel configuration
parameters to request
* the size of the kernel virtual space managed by
each of the
* kernel segment managers.
*
* See carve_kvspace() for a discussion of how
these are used.
*

	segkmem_bytes(%l) =
{SEGKMEM_BYTES}

	segkmem_percent(%l) =
{SEGKMEM_PERCENT}

	segmap_bytes(%l) =
{SEGMAP_BYTES}

	segmap_percent(%l) =
{SEGMAP_PERCENT}

	segkvn_bytes(%l) =
{SEGKVN_BYTES}

	segkvn_percent(%l) =
{SEGKVN_PERCENT}

	segkmem_max_bytes(%l) =
{SEGKMEM_MAX_BYTES}
*
	syssegsz(%i) = {SYSSEGSZ}
	filemap(%i) = {FILEMAP}
$$
* Kernel Virtual Address Space ----------
SEGKMEM_BYTES = 0x1000000
SEGKMEM_PERCENT = 50
SEGKVN_BYTES = 0x1000000
SEGKVN_PERCENT = 15
SEGMAP_BYTES = 0x1000000
SEGMAP_PERCENT = 20
SEGKMEM_MAX_BYTES = 0x40000000
* Segment Driver Parameters ----------
SEGMAP_AGE_TIME    = 60
SEGMAP_AGINGS =    20
SEGKVN_AGE_TIME    = 60
* Paging Parameters ------------------------
MINAMEM  = 16
KMEM_RESV = 16
PAGES_NODISKMA = 16
* Swapping Parameters ----------------------
SCALE_MAXPGIO = 1
DEFICIT_AGE = 10
IO_WEIGHT = 1
CPU_WEIGHT = 10
SWAP_WEIGHT = 1
SLEEP_WEIGHT = 0
MAXSLP = 600
SWAP_MAXDEV =      32
MAX_DEFICIT =      256
* Aging Parameters -------------------------
ET_AGE_INTERVAL    = 5
NONLOCKED_MINPG  = 0
MAXRSS    =        512
INIT_AGEQUANTUM    = 50
MIN_AGEQUANTUM    = 25
MAX_AGEQUANTUM    = 60
LO_GROW_RATE =     2
HI_GROW_RATE =     8
* Parameters for Restricted-DMA Support -----
MAXDMAPAGE = 16384



****************************************
	msg
****************************************


* All Rights Reserved, Copyright (c) PFU &
FUJITSU LIMITED 1996
*#ident "@(#)msg.cf      6.1.2.1 96/10/22 DS"
*
* MSG
*
*FLAG      #VEC      PREFIX    SOFT
          #DEV      IPL
          DEPENDENCIES/VARIABLES
ox        -         msg       -         -
          -         ipc


	msginfo(%i%i%i%i%i%i%s)

			={MSGMAP,

				MSGMAX,

				MSGMNB,

				MSGMNI,

				MSGSSZ,

				MSGTQL,

				MSGSEG}
			msgalc(%i) = {MSGALC}
			msghd [((MSGTQL *
MSGALC + 1) * (16 + MSGSSZ * 2)) / 4](%i)
			msgtxt[((MSGSEG *
MSGALC + 1) * MSGSSZ) / 4](%i)

$$$

MSGMAP   = 200
MSGMAX   = 16384
MSGMNB   = 32768
MSGMNI   = 512
MSGSSZ   = 8
MSGTQL   = 8192
```

```
MSGSEG    = 32768
MSGALC    = 0




****************************************
            shm
****************************************


* All Rights Reserved, Copyright (c) PFU &
FUJITSU LIMITED 1996
*#ident "@(#)shm.cf       6.1.2.1 96/10/22 DS"
*
* SHM
*
*FLAG       #VEC        PREFIX      SOFT
            #DEV        IPL
            DEPENDENCIES/VARIABLES
ox          -           shm         -           -
            -           ipc

                        shminfo(%i%i%i%i)

                            ={SHMMAX,

                                    SHMMIN,

                                    SHMMNI,

                                    SHMSEG}


            shmrsvmem(%i)={SHMRSVMEM}


            shmrsvmin(%i)={SHMRSVMIN}

                        shmzerothrtim(%i)


={SHMZEROTHRTIM}

$$$

SHMMAX = 0x7ffff000
SHMMIN = 1
SHMMNI = 100
SHMSEG = 100
SHMRSVMEM = 3584
SHMRSVMIN = 64
SHMZEROTHRTIM = 10




****************************************
            p_run.ora
****************************************


#
#====================================
============================+
#       Copyright (c) 1995  Oracle Corp, Redwood
Shores, CA      |
#                OPEN SYSTEMS PERFORMANCE
GROUP               |
#                   All Rights Reserved
|
#====================================
============================+
# FILENAME
#      p_run.ora
# DESCRIPTION
#      Oracle parameter file for running TPC-C.
#====================================
============================
#

serializable                = FALSE
optimizer_mode                          = CHOOSE
db_writers                  = 1
async_read                  = true
async_write                 = true
cpu_count                   = 2
db_block_lru_latches        = 8
spin_count                  = 2000
parallel_max_servers        = 30
checkpoint_process          = TRUE
compatible                  = 7.3.2.2.0
db_name                     = tpcc
db_files                    = 1000
db_file_multiblock_read_count = 32
db_block_buffers            = 1615920
_db_block_write_batch       = 1024
db_block_checkpoint_batch = 1024
dml_locks                   = 0
log_archive_start           = FALSE
log_archive_buffer_size = 32
log_checkpoint_interval = 10000000000
log_checkpoints_to_alert= TRUE
log_buffer                  = 1048576
log_simultaneous_copies = 8
log_small_entry_max_size= 800
gc_rollback_segments        = 220
gc_db_locks                             = 100
gc_releasable_locks         = 100
max_rollback_segments = 220
open_cursors                = 200
processes                   = 200
sessions                    = 400
transactions                = 400
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,t21,t22,t23,t24,t25,t26,t27,t28,t29,t
30,t31,t32,t33,t34,t35,t36,t37,t38,t39,t40,t41,t41,t4
3,t44,t45,t46,t48,t49,t50,t51,t52,t53,t54,t55,t56,t57
,t58,t59,t60,t61,t62,t63,t64,t65,t66,t67,t68,t69,t70,t
71,t72,t73,t74,t75)
shared_pool_size            = 7000000
discrete_transactions_enabled = FALSE
cursor_space_for_time       = TRUE




****************************************
TUXDO Tunable parameters (No.1)
****************************************


*RESOURCES
IPCKEY              80952
MASTER              SITE1
UID                 101
GID                 102
PERM                0660
MAXACCESSERS        1100
MAXSERVERS          30
MAXSERVICES         200
MODEL               SHM
LDBAL               N

*MACHINES
ptnds01     LMID=SITE1
            TUXCONFIG="/home/oracle/client/tux
config"
            ROOTDIR="/opt/uxptuxt"
            APPDIR="/oracle/bench/tpc/tpcc/TUX
_source"
            ULOGPFX="/home/oracle/client/ulog"

*GROUPS
GROUP1                  LMID=SITE1
            GRPNO=1

*SERVERS
DEFAULT:  RESTART=Y MAXGEN=5
REPLYQ=N RQPERM=0660
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=1
CLOPT ="-s TPCC01 1 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=2
CLOPT ="-s TPCC02 2 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=3
CLOPT ="-s TPCC03 3 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=4
CLOPT ="-s TPCC04 4 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=5
CLOPT ="-s TPCC05 5 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=6
CLOPT ="-s TPCC06 6 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=7
CLOPT ="-s TPCC07 7 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=8
CLOPT ="-s TPCC08 8 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=9
CLOPT ="-s TPCC09 9 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=10
CLOPT ="-s TPCC10 10 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=11
CLOPT ="-s TPCC11 11 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=12
CLOPT ="-s TPCC12 12 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=13
CLOPT ="-s TPCC13 13 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=14
CLOPT ="-s TPCC14 14 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=15
CLOPT ="-s TPCC15 15 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=16
CLOPT ="-s TPCC16 16 tpcc/tpcc"

*SERVICES
TPCC01
TPCC02
TPCC03
TPCC04
TPCC05
TPCC06
```

```
TPCC07
TPCC08
TPCC09
TPCC10
TPCC11
TPCC12
TPCC13
TPCC14
TPCC15
TPCC16

*ROUTING

# *NETWORK




****************************************
TUXDO Tunable Parameters (No.2)
****************************************


*RESOURCES
IPCKEY                  80952
MASTER                  SITE1
UID                     101
GID                     102
PERM                    0660
MAXACCESSERS            550
MAXSERVERS              15
MAXSERVICES             100
MODEL                   SHM
LDBAL                   N

*MACHINES
ptnds05     LMID=SITE1
            TUXCONFIG="/home/oracle/client/tux
config"
            ROOTDIR="/opt/uxptuxt"
            APPDIR="/oracle/bench/tpc/tpcc/TUX
_source"
            ULOGPFX="/home/oracle/client/ulog"

*GROUPS
GROUP1                  LMID=SITE1
            GRPNO=1

*SERVERS
DEFAULT:  RESTART=Y MAXGEN=5
REPLYQ=N RQPERM=0660
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=1
CLOPT ="-s TPCC01 1 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=2
CLOPT ="-s TPCC02 2 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=3
CLOPT ="-s TPCC03 3 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=4
CLOPT ="-s TPCC04 4 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=5
CLOPT ="-s TPCC05 5 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=6
CLOPT ="-s TPCC06 6 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=7
CLOPT ="-s TPCC07 7 tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=8
CLOPT ="-s TPCC08 8 tpcc/tpcc"

*SERVICES
TPCC01
TPCC02
TPCC03
TPCC04
TPCC05
TPCC06
TPCC07
TPCC08

*ROUTING

# *NETWORK
```

Appendix E: Database Creation Code

```
****************************************
            addfile.sh
****************************************


#
#====================================
============================+
#       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;
!



****************************************
            alter.sh
****************************************


#
#====================================
============================+
#       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    7M);
  alter cluster ccluster  storage (next   29M);
  alter cluster scluster  storage (next   31M);
  alter table orders      storage (next   11M);
  alter table order_line  storage (next  155M);
  alter table new_order   storage (next 3584K);
  alter index iorders     storage (next   14M);
  alter index iorders2    storage (next   14M);
  alter index inew_order  storage (next    5M);
  alter index iorder_line storage (next  115M);
  alter index istock      storage (next   45M);
  alter index icustomer   storage (next   36M);
  alter index icustomer2  storage (next   31M);
  quit;
!



****************************************
            benchdb.sh
****************************************


#
#====================================
============================+
#       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

while [ "$#" != "0" ]
do
  case $1 in
    -n) shift
      NO_CREATE="y"
      ;;
    -c) shift
      NO_CAT="y"
      ;;
     *) echo "Bag arg: $1"
      exit 1;
      ;;
  esac
done

#
# Create database if NO_CREATE unset
#

if [ "$NO_CREATE" = "" ]
then
sqldba <<!
            set echo on
            connect internal
            startup
pfile=$TPCC_ADMIN/p_create.ora  nomount
            create database tpcc controlfile reuse
maxdatafiles 1000
          datafile '/dev/rdsk/hdf2501' size  599M
reuse
              logfile  '/dev/rdsk/hdf1001' size
2039M reuse,
                                '/dev/rdsk/hdf300 1'
size 2039M reuse;
            exit
!

#
# FOR OPS:#   connect system/manager
# Need to add MAXINSTANCES and adjust
MAXLOGFILES for the create database
# statement.
#

#
# Create more rollback segments
#

sqldba <<!
  connect internal
  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);
```

```
  create rollback segment s11 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s12 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s13 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s14 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s15 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s16 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s17 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s18 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s19 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s20 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s21 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s22 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s23 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s24 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s25 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s26 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s27 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s28 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s29 storage (initial 600k
minextents 2 next 600k);
  create rollback segment s30 storage (initial 600k
minextents 2 next 600k);
  disconnect;
  connect internal;
  shutdown;
  exit;
!
fi

#
#  Startup database with params file that includes
new rollback segments
#

sqldba <<!
        set echo on
    connect internal
    startup pfile=$TPCC_ADMIN/p_build.ora;
    exit;
!
create.sh hist   /dev/rdsk/hda1001   15M
create.sh stocks /dev/rdsk/hda1002  158M
create.sh cust   /dev/rdsk/hda1003  146M
create.sh ord    /dev/rdsk/hda1004   12M
create.sh nord   /dev/rdsk/hda1005    4M

create.sh iordl  /dev/rdsk/hda2401  576M
create.sh inord  /dev/rdsk/hda2402   26M
create.sh iord1  /dev/rdsk/hda2403   57M
create.sh icust2 /dev/rdsk/hda2404   63M

create.sh ordl   /dev/rdsk/hde3001  931M

create.sh ware   /dev/rdsk/hdf2502   24M
create.sh items  /dev/rdsk/hdf2503   19M
create.sh icust1 /dev/rdsk/hdf4501  794M

create.sh roll1  /dev/rdsk/hda4301   49M
create.sh roll2  /dev/rdsk/hda6301   49M
create.sh roll3  /dev/rdsk/hdb2301   49M
create.sh roll4  /dev/rdsk/hdb4301   49M
create.sh roll5  /dev/rdsk/hdc2301   49M
create.sh roll6  /dev/rdsk/hdc4301   49M
create.sh roll7  /dev/rdsk/hdc6301   49M
create.sh roll8  /dev/rdsk/hdd2301   49M
create.sh roll9  /dev/rdsk/hdd4301   49M
create.sh roll10 /dev/rdsk/hde2301   49M

create.sh temp   /dev/rdsk/hda4302  931M

create.sh istk   /dev/rdsk/hdf2401  679M
create.sh iord2  /dev/rdsk/hda4501   60M

create.sh ltemp  /dev/rdsk/hdf4301  999M

#
# FOR OPS:
# Needs to create more rollback segment
tablespace to house extra
# rollback segments for the additional instances.
#

#
#  Add datafiles to tablespaces in parallel
#

#   [TEMP]
addfile.sh temp   /dev/rdsk/hda6302  931M &
addfile.sh temp   /dev/rdsk/hdb2302  931M &
addfile.sh temp   /dev/rdsk/hdb4302  931M &
addfile.sh temp   /dev/rdsk/hdc2302  931M &
addfile.sh temp   /dev/rdsk/hdc4302  931M &
addfile.sh temp   /dev/rdsk/hdc6302  931M &
addfile.sh temp   /dev/rdsk/hdd2302  931M &
addfile.sh temp   /dev/rdsk/hdd4302  931M &
addfile.sh temp   /dev/rdsk/hde2302  931M &
wait

#   [HISTORY]
addfile.sh hist   /dev/rdsk/hda1101   15M &
addfile.sh hist   /dev/rdsk/hda1201   15M &
addfile.sh hist   /dev/rdsk/hda1301   15M &
addfile.sh hist   /dev/rdsk/hda1401   15M &
addfile.sh hist   /dev/rdsk/hda1501   15M &
addfile.sh hist   /dev/rdsk/hda2001   15M &
addfile.sh hist   /dev/rdsk/hda2101   15M &
addfile.sh hist   /dev/rdsk/hda2201   15M &
addfile.sh hist   /dev/rdsk/hda2301   15M &
addfile.sh hist   /dev/rdsk/hda3001   15M &
wait
addfile.sh hist   /dev/rdsk/hda3101   15M &
addfile.sh hist   /dev/rdsk/hda3201   15M &
addfile.sh hist   /dev/rdsk/hda3301   15M &
addfile.sh hist   /dev/rdsk/hda3401   15M &
addfile.sh hist   /dev/rdsk/hda3501   15M &
addfile.sh hist   /dev/rdsk/hda4001   15M &
addfile.sh hist   /dev/rdsk/hda4101   15M &
addfile.sh hist   /dev/rdsk/hda4201   15M &
addfile.sh hist   /dev/rdsk/hda5001   15M &
addfile.sh hist   /dev/rdsk/hda5101   15M &
wait
addfile.sh hist   /dev/rdsk/hda5201   15M &
addfile.sh hist   /dev/rdsk/hda5301   15M &
addfile.sh hist   /dev/rdsk/hda5401   15M &
addfile.sh hist   /dev/rdsk/hda5501   15M &
addfile.sh hist   /dev/rdsk/hda6001   15M &
addfile.sh hist   /dev/rdsk/hda6101   15M &
addfile.sh hist   /dev/rdsk/hda6201   15M &
addfile.sh hist   /dev/rdsk/hdb1001   15M &
addfile.sh hist   /dev/rdsk/hdb1101   15M &
addfile.sh hist   /dev/rdsk/hdb1201   15M &
wait
addfile.sh hist   /dev/rdsk/hdb1301   15M &
addfile.sh hist   /dev/rdsk/hdb1401   15M &
addfile.sh hist   /dev/rdsk/hdb1501   15M &
addfile.sh hist   /dev/rdsk/hdb2001   15M &
addfile.sh hist   /dev/rdsk/hdb2101   15M &
addfile.sh hist   /dev/rdsk/hdb2201   15M &
addfile.sh hist   /dev/rdsk/hdb3001   15M &
addfile.sh hist   /dev/rdsk/hdb3101   15M &
addfile.sh hist   /dev/rdsk/hdb3201   15M &
addfile.sh hist   /dev/rdsk/hdb3301   15M &
wait
addfile.sh hist   /dev/rdsk/hdb3401   15M &
addfile.sh hist   /dev/rdsk/hdb3501   15M &
addfile.sh hist   /dev/rdsk/hdb4001   15M &
addfile.sh hist   /dev/rdsk/hdb4101   15M &
addfile.sh hist   /dev/rdsk/hdb4201   15M &
addfile.sh hist   /dev/rdsk/hdc1001   15M &
addfile.sh hist   /dev/rdsk/hdc1101   15M &
addfile.sh hist   /dev/rdsk/hdc1201   15M &
addfile.sh hist   /dev/rdsk/hdc1301   15M &
addfile.sh hist   /dev/rdsk/hdc1401   15M &
wait
addfile.sh hist   /dev/rdsk/hdc1501   15M &
addfile.sh hist   /dev/rdsk/hdc2001   15M &
addfile.sh hist   /dev/rdsk/hdc2101   15M &
addfile.sh hist   /dev/rdsk/hdc2201   15M &
addfile.sh hist   /dev/rdsk/hdc3001   15M &
addfile.sh hist   /dev/rdsk/hdc3101   15M &
addfile.sh hist   /dev/rdsk/hdc3201   15M &
addfile.sh hist   /dev/rdsk/hdc3301   15M &
addfile.sh hist   /dev/rdsk/hdc3401   15M &
addfile.sh hist   /dev/rdsk/hdc3501   15M &
wait
addfile.sh hist   /dev/rdsk/hdc4001   15M &
addfile.sh hist   /dev/rdsk/hdc4101   15M &
addfile.sh hist   /dev/rdsk/hdc4201   15M &
addfile.sh hist   /dev/rdsk/hdc5001   15M &
addfile.sh hist   /dev/rdsk/hdc5101   15M &
addfile.sh hist   /dev/rdsk/hdc5201   15M &
addfile.sh hist   /dev/rdsk/hdc5301   15M &
addfile.sh hist   /dev/rdsk/hdc5401   15M &
addfile.sh hist   /dev/rdsk/hdc5501   15M &
addfile.sh hist   /dev/rdsk/hdc6001   15M &
wait
addfile.sh hist   /dev/rdsk/hdc6101   15M &
addfile.sh hist   /dev/rdsk/hdc6201   15M &
addfile.sh hist   /dev/rdsk/hdd1001   15M &
addfile.sh hist   /dev/rdsk/hdd1101   15M &
addfile.sh hist   /dev/rdsk/hdd1201   15M &
addfile.sh hist   /dev/rdsk/hdd1301   15M &
addfile.sh hist   /dev/rdsk/hdd1401   15M &
addfile.sh hist   /dev/rdsk/hdd1501   15M &
addfile.sh hist   /dev/rdsk/hdd2001   15M &
addfile.sh hist   /dev/rdsk/hdd2101   15M &
```

```
wait
addfile.sh hist  /dev/rdsk/hdd2201  15M &
addfile.sh hist  /dev/rdsk/hdd3001  15M &
addfile.sh hist  /dev/rdsk/hdd3101  15M &
addfile.sh hist  /dev/rdsk/hdd3201  15M &
addfile.sh hist  /dev/rdsk/hdd3301  15M &
addfile.sh hist  /dev/rdsk/hdd3401  15M &
addfile.sh hist  /dev/rdsk/hdd3501  15M &
addfile.sh hist  /dev/rdsk/hdd4001  15M &
addfile.sh hist  /dev/rdsk/hdd4101  15M &
addfile.sh hist  /dev/rdsk/hdd4201  15M &
wait
addfile.sh hist  /dev/rdsk/hde1001  15M &
addfile.sh hist  /dev/rdsk/hde1101  15M &
addfile.sh hist  /dev/rdsk/hde1201  15M &
addfile.sh hist  /dev/rdsk/hde1301  15M &
addfile.sh hist  /dev/rdsk/hde1401  15M &
addfile.sh hist  /dev/rdsk/hde1501  15M &
addfile.sh hist  /dev/rdsk/hde2001  15M &
addfile.sh hist  /dev/rdsk/hde2101  15M &
addfile.sh hist  /dev/rdsk/hde2201  15M &
wait

#  [STOCKS]
addfile.sh stocks /dev/rdsk/hda1102  158M &
addfile.sh stocks /dev/rdsk/hda1202  158M &
addfile.sh stocks /dev/rdsk/hda1302  158M &
addfile.sh stocks /dev/rdsk/hda1402  158M &
addfile.sh stocks /dev/rdsk/hda1502  158M &
addfile.sh stocks /dev/rdsk/hda2002  158M &
addfile.sh stocks /dev/rdsk/hda2102  158M &
addfile.sh stocks /dev/rdsk/hda2202  158M &
addfile.sh stocks /dev/rdsk/hda2302  158M &
addfile.sh stocks /dev/rdsk/hda3002  158M &
wait
addfile.sh stocks /dev/rdsk/hda3102  158M &
addfile.sh stocks /dev/rdsk/hda3202  158M &
addfile.sh stocks /dev/rdsk/hda3302  158M &
addfile.sh stocks /dev/rdsk/hda3402  158M &
addfile.sh stocks /dev/rdsk/hda3502  158M &
addfile.sh stocks /dev/rdsk/hda4002  158M &
addfile.sh stocks /dev/rdsk/hda4102  158M &
addfile.sh stocks /dev/rdsk/hda4202  158M &
addfile.sh stocks /dev/rdsk/hda5002  158M &
addfile.sh stocks /dev/rdsk/hda5102  158M &
wait
addfile.sh stocks /dev/rdsk/hda5202  158M &
addfile.sh stocks /dev/rdsk/hda5302  158M &
addfile.sh stocks /dev/rdsk/hda5402  158M &
addfile.sh stocks /dev/rdsk/hda5502  158M &
addfile.sh stocks /dev/rdsk/hda6002  158M &
addfile.sh stocks /dev/rdsk/hda6102  158M &
addfile.sh stocks /dev/rdsk/hda6202  158M &
addfile.sh stocks /dev/rdsk/hdb1002  158M &
addfile.sh stocks /dev/rdsk/hdb1102  158M &
addfile.sh stocks /dev/rdsk/hdb1202  158M &
wait
addfile.sh stocks /dev/rdsk/hdb1302  158M &
addfile.sh stocks /dev/rdsk/hdb1402  158M &
addfile.sh stocks /dev/rdsk/hdb1502  158M &
addfile.sh stocks /dev/rdsk/hdb2002  158M &
addfile.sh stocks /dev/rdsk/hdb2102  158M &
addfile.sh stocks /dev/rdsk/hdb2202  158M &
addfile.sh stocks /dev/rdsk/hdb3002  158M &
addfile.sh stocks /dev/rdsk/hdb3102  158M &
addfile.sh stocks /dev/rdsk/hdb3202  158M &
addfile.sh stocks /dev/rdsk/hdb3302  158M &
wait
addfile.sh stocks /dev/rdsk/hdb3402  158M &
addfile.sh stocks /dev/rdsk/hdb3502  158M &
addfile.sh stocks /dev/rdsk/hdb4002  158M &
addfile.sh stocks /dev/rdsk/hdb4102  158M &
addfile.sh stocks /dev/rdsk/hdb4202  158M &
addfile.sh stocks /dev/rdsk/hdc1002  158M &
addfile.sh stocks /dev/rdsk/hdc1102  158M &
addfile.sh stocks /dev/rdsk/hdc1202  158M &
addfile.sh stocks /dev/rdsk/hdc1302  158M &
addfile.sh stocks /dev/rdsk/hdc1402  158M &
wait
addfile.sh stocks /dev/rdsk/hdc1502  158M &
addfile.sh stocks /dev/rdsk/hdc2002  158M &
addfile.sh stocks /dev/rdsk/hdc2102  158M &
addfile.sh stocks /dev/rdsk/hdc2202  158M &
addfile.sh stocks /dev/rdsk/hdc3002  158M &
addfile.sh stocks /dev/rdsk/hdc3102  158M &
addfile.sh stocks /dev/rdsk/hdc3202  158M &
addfile.sh stocks /dev/rdsk/hdc3302  158M &
addfile.sh stocks /dev/rdsk/hdc3402  158M &
addfile.sh stocks /dev/rdsk/hdc3502  158M &
wait
addfile.sh stocks /dev/rdsk/hdc4002  158M &
addfile.sh stocks /dev/rdsk/hdc4102  158M &
addfile.sh stocks /dev/rdsk/hdc4202  158M &
addfile.sh stocks /dev/rdsk/hdc5002  158M &
addfile.sh stocks /dev/rdsk/hdc5102  158M &
addfile.sh stocks /dev/rdsk/hdc5202  158M &
addfile.sh stocks /dev/rdsk/hdc5302  158M &
addfile.sh stocks /dev/rdsk/hdc5402  158M &
addfile.sh stocks /dev/rdsk/hdc5502  158M &
addfile.sh stocks /dev/rdsk/hdc6002  158M &
wait
addfile.sh stocks /dev/rdsk/hdc6102  158M &
addfile.sh stocks /dev/rdsk/hdc6202  158M &
addfile.sh stocks /dev/rdsk/hdd1002  158M &
addfile.sh stocks /dev/rdsk/hdd1102  158M &
addfile.sh stocks /dev/rdsk/hdd1202  158M &
addfile.sh stocks /dev/rdsk/hdd1302  158M &
addfile.sh stocks /dev/rdsk/hdd1402  158M &
addfile.sh stocks /dev/rdsk/hdd1502  158M &
addfile.sh stocks /dev/rdsk/hdd2002  158M &
addfile.sh stocks /dev/rdsk/hdd2102  158M &
wait
addfile.sh stocks /dev/rdsk/hdd2202  158M &
addfile.sh stocks /dev/rdsk/hdd3002  158M &
addfile.sh stocks /dev/rdsk/hdd3102  158M &
addfile.sh stocks /dev/rdsk/hdd3202  158M &
addfile.sh stocks /dev/rdsk/hdd3302  158M &
addfile.sh stocks /dev/rdsk/hdd3402  158M &
addfile.sh stocks /dev/rdsk/hdd3502  158M &
addfile.sh stocks /dev/rdsk/hdd4002  158M &
addfile.sh stocks /dev/rdsk/hdd4102  158M &
addfile.sh stocks /dev/rdsk/hdd4202  158M &
wait
addfile.sh stocks /dev/rdsk/hde1002  158M &
addfile.sh stocks /dev/rdsk/hde1102  158M &
addfile.sh stocks /dev/rdsk/hde1202  158M &
addfile.sh stocks /dev/rdsk/hde1302  158M &
addfile.sh stocks /dev/rdsk/hde1402  158M &
addfile.sh stocks /dev/rdsk/hde1502  158M &
addfile.sh stocks /dev/rdsk/hde2002  158M &
addfile.sh stocks /dev/rdsk/hde2102  158M &
addfile.sh stocks /dev/rdsk/hde2202  158M &
wait
addfile.sh stocks /dev/rdsk/hdf1201  158M &
addfile.sh stocks /dev/rdsk/hdf1301  158M &
addfile.sh stocks /dev/rdsk/hdf1401  158M &
addfile.sh stocks /dev/rdsk/hdf1501  158M &
addfile.sh stocks /dev/rdsk/hdf2001  158M &
addfile.sh stocks /dev/rdsk/hdf3201  158M &
addfile.sh stocks /dev/rdsk/hdf3301  158M &
addfile.sh stocks /dev/rdsk/hdf3401  158M &
addfile.sh stocks /dev/rdsk/hdf3501  158M &
addfile.sh stocks /dev/rdsk/hdf4001  158M &
wait

#  [CUSTOMER]
addfile.sh cust  /dev/rdsk/hda1103  146M &
addfile.sh cust  /dev/rdsk/hda1203  146M &
addfile.sh cust  /dev/rdsk/hda1303  146M &
addfile.sh cust  /dev/rdsk/hda1403  146M &
addfile.sh cust  /dev/rdsk/hda1503  146M &
addfile.sh cust  /dev/rdsk/hda2003  146M &
addfile.sh cust  /dev/rdsk/hda2103  146M &
addfile.sh cust  /dev/rdsk/hda2203  146M &
addfile.sh cust  /dev/rdsk/hda2303  146M &
addfile.sh cust  /dev/rdsk/hda3003  146M &
wait
addfile.sh cust  /dev/rdsk/hda3103  146M &
addfile.sh cust  /dev/rdsk/hda3203  146M &
addfile.sh cust  /dev/rdsk/hda3303  146M &
addfile.sh cust  /dev/rdsk/hda3403  146M &
addfile.sh cust  /dev/rdsk/hda3503  146M &
addfile.sh cust  /dev/rdsk/hda4003  146M &
addfile.sh cust  /dev/rdsk/hda4103  146M &
addfile.sh cust  /dev/rdsk/hda4203  146M &
addfile.sh cust  /dev/rdsk/hda5003  146M &
addfile.sh cust  /dev/rdsk/hda5103  146M &
wait
addfile.sh cust  /dev/rdsk/hda5203  146M &
addfile.sh cust  /dev/rdsk/hda5303  146M &
addfile.sh cust  /dev/rdsk/hda5403  146M &
addfile.sh cust  /dev/rdsk/hda5503  146M &
addfile.sh cust  /dev/rdsk/hda6003  146M &
addfile.sh cust  /dev/rdsk/hda6103  146M &
addfile.sh cust  /dev/rdsk/hda6203  146M &
addfile.sh cust  /dev/rdsk/hdb1003  146M &
addfile.sh cust  /dev/rdsk/hdb1103  146M &
addfile.sh cust  /dev/rdsk/hdb1203  146M &
wait
addfile.sh cust  /dev/rdsk/hdb1303  146M &
addfile.sh cust  /dev/rdsk/hdb1403  146M &
addfile.sh cust  /dev/rdsk/hdb1503  146M &
addfile.sh cust  /dev/rdsk/hdb2003  146M &
addfile.sh cust  /dev/rdsk/hdb2103  146M &
addfile.sh cust  /dev/rdsk/hdb2203  146M &
addfile.sh cust  /dev/rdsk/hdb3003  146M &
addfile.sh cust  /dev/rdsk/hdb3103  146M &
addfile.sh cust  /dev/rdsk/hdb3203  146M &
addfile.sh cust  /dev/rdsk/hdb3303  146M &
wait
addfile.sh cust  /dev/rdsk/hdb3403  146M &
addfile.sh cust  /dev/rdsk/hdb3503  146M &
addfile.sh cust  /dev/rdsk/hdb4003  146M &
addfile.sh cust  /dev/rdsk/hdb4103  146M &
addfile.sh cust  /dev/rdsk/hdb4203  146M &
addfile.sh cust  /dev/rdsk/hdc1003  146M &
addfile.sh cust  /dev/rdsk/hdc1103  146M &
addfile.sh cust  /dev/rdsk/hdc1203  146M &
addfile.sh cust  /dev/rdsk/hdc1303  146M &
addfile.sh cust  /dev/rdsk/hdc1403  146M &
wait
addfile.sh cust  /dev/rdsk/hdc1503  146M &
addfile.sh cust  /dev/rdsk/hdc2003  146M &
addfile.sh cust  /dev/rdsk/hdc2103  146M &
```

```
addfile.sh cust  /dev/rdsk/hdc2203  146M &
addfile.sh cust  /dev/rdsk/hdc3003  146M &
addfile.sh cust  /dev/rdsk/hdc3103  146M &
addfile.sh cust  /dev/rdsk/hdc3203  146M &
addfile.sh cust  /dev/rdsk/hdc3303  146M &
addfile.sh cust  /dev/rdsk/hdc3403  146M &
addfile.sh cust  /dev/rdsk/hdc3503  146M &
wait
addfile.sh cust  /dev/rdsk/hdc4003  146M &
addfile.sh cust  /dev/rdsk/hdc4103  146M &
addfile.sh cust  /dev/rdsk/hdc4203  146M &
addfile.sh cust  /dev/rdsk/hdc5003  146M &
addfile.sh cust  /dev/rdsk/hdc5103  146M &
addfile.sh cust  /dev/rdsk/hdc5203  146M &
addfile.sh cust  /dev/rdsk/hdc5303  146M &
addfile.sh cust  /dev/rdsk/hdc5403  146M &
addfile.sh cust  /dev/rdsk/hdc5503  146M &
addfile.sh cust  /dev/rdsk/hdc6003  146M &
wait
addfile.sh cust  /dev/rdsk/hdc6103  146M &
addfile.sh cust  /dev/rdsk/hdc6203  146M &
addfile.sh cust  /dev/rdsk/hdd1003  146M &
addfile.sh cust  /dev/rdsk/hdd1103  146M &
addfile.sh cust  /dev/rdsk/hdd1203  146M &
addfile.sh cust  /dev/rdsk/hdd1303  146M &
addfile.sh cust  /dev/rdsk/hdd1403  146M &
addfile.sh cust  /dev/rdsk/hdd1503  146M &
addfile.sh cust  /dev/rdsk/hdd2003  146M &
addfile.sh cust  /dev/rdsk/hdd2103  146M &
wait
addfile.sh cust  /dev/rdsk/hdd2203  146M &
addfile.sh cust  /dev/rdsk/hdd3003  146M &
addfile.sh cust  /dev/rdsk/hdd3103  146M &
addfile.sh cust  /dev/rdsk/hdd3203  146M &
addfile.sh cust  /dev/rdsk/hdd3303  146M &
addfile.sh cust  /dev/rdsk/hdd3403  146M &
addfile.sh cust  /dev/rdsk/hdd3503  146M &
addfile.sh cust  /dev/rdsk/hdd4003  146M &
addfile.sh cust  /dev/rdsk/hdd4103  146M &
addfile.sh cust  /dev/rdsk/hdd4203  146M &
wait
addfile.sh cust  /dev/rdsk/hde1003  146M &
addfile.sh cust  /dev/rdsk/hde1103  146M &
addfile.sh cust  /dev/rdsk/hde1203  146M &
addfile.sh cust  /dev/rdsk/hde1303  146M &
addfile.sh cust  /dev/rdsk/hde1403  146M &
addfile.sh cust  /dev/rdsk/hde1503  146M &
addfile.sh cust  /dev/rdsk/hde2003  146M &
addfile.sh cust  /dev/rdsk/hde2103  146M &
addfile.sh cust  /dev/rdsk/hde2203  146M &
wait

#  [ORDERS]
addfile.sh ord  /dev/rdsk/hda1104  12M &
addfile.sh ord  /dev/rdsk/hda1204  12M &
addfile.sh ord  /dev/rdsk/hda1304  12M &
addfile.sh ord  /dev/rdsk/hda1404  12M &
addfile.sh ord  /dev/rdsk/hda1504  12M &
addfile.sh ord  /dev/rdsk/hda2004  12M &
addfile.sh ord  /dev/rdsk/hda2104  12M &
addfile.sh ord  /dev/rdsk/hda2204  12M &
addfile.sh ord  /dev/rdsk/hda2304  12M &
addfile.sh ord  /dev/rdsk/hda3004  12M &
wait
addfile.sh ord  /dev/rdsk/hda3104  12M &
addfile.sh ord  /dev/rdsk/hda3204  12M &
addfile.sh ord  /dev/rdsk/hda3304  12M &
addfile.sh ord  /dev/rdsk/hda3404  12M &
addfile.sh ord  /dev/rdsk/hda3504  12M &
addfile.sh ord  /dev/rdsk/hda4004  12M &
addfile.sh ord  /dev/rdsk/hda4104  12M &
addfile.sh ord  /dev/rdsk/hda4204  12M &
addfile.sh ord  /dev/rdsk/hda5004  12M &
addfile.sh ord  /dev/rdsk/hda5104  12M &
wait
addfile.sh ord  /dev/rdsk/hda5204  12M &
addfile.sh ord  /dev/rdsk/hda5304  12M &
addfile.sh ord  /dev/rdsk/hda5404  12M &
addfile.sh ord  /dev/rdsk/hda5504  12M &
addfile.sh ord  /dev/rdsk/hda6004  12M &
addfile.sh ord  /dev/rdsk/hda6104  12M &
addfile.sh ord  /dev/rdsk/hda6204  12M &
addfile.sh ord  /dev/rdsk/hdb1004  12M &
addfile.sh ord  /dev/rdsk/hdb1104  12M &
addfile.sh ord  /dev/rdsk/hdb1204  12M &
wait
addfile.sh ord  /dev/rdsk/hdb1304  12M &
addfile.sh ord  /dev/rdsk/hdb1404  12M &
addfile.sh ord  /dev/rdsk/hdb1504  12M &
addfile.sh ord  /dev/rdsk/hdb2004  12M &
addfile.sh ord  /dev/rdsk/hdb2104  12M &
addfile.sh ord  /dev/rdsk/hdb2204  12M &
addfile.sh ord  /dev/rdsk/hdb3004  12M &
addfile.sh ord  /dev/rdsk/hdb3104  12M &
addfile.sh ord  /dev/rdsk/hdb3204  12M &
addfile.sh ord  /dev/rdsk/hdb3304  12M &
wait
addfile.sh ord  /dev/rdsk/hdb3404  12M &
addfile.sh ord  /dev/rdsk/hdb3504  12M &
addfile.sh ord  /dev/rdsk/hdb4004  12M &
addfile.sh ord  /dev/rdsk/hdb4104  12M &
addfile.sh ord  /dev/rdsk/hdb4204  12M &
addfile.sh ord  /dev/rdsk/hdc1004  12M &
addfile.sh ord  /dev/rdsk/hdc1104  12M &
addfile.sh ord  /dev/rdsk/hdc1204  12M &
addfile.sh ord  /dev/rdsk/hdc1304  12M &
addfile.sh ord  /dev/rdsk/hdc1404  12M &
wait
addfile.sh ord  /dev/rdsk/hdc1504  12M &
addfile.sh ord  /dev/rdsk/hdc2004  12M &
addfile.sh ord  /dev/rdsk/hdc2104  12M &
addfile.sh ord  /dev/rdsk/hdc2204  12M &
addfile.sh ord  /dev/rdsk/hdc3004  12M &
addfile.sh ord  /dev/rdsk/hdc3104  12M &
addfile.sh ord  /dev/rdsk/hdc3204  12M &
addfile.sh ord  /dev/rdsk/hdc3304  12M &
addfile.sh ord  /dev/rdsk/hdc3404  12M &
addfile.sh ord  /dev/rdsk/hdc3504  12M &
wait
addfile.sh ord  /dev/rdsk/hdc4004  12M &
addfile.sh ord  /dev/rdsk/hdc4104  12M &
addfile.sh ord  /dev/rdsk/hdc4204  12M &
addfile.sh ord  /dev/rdsk/hdc5004  12M &
addfile.sh ord  /dev/rdsk/hdc5104  12M &
addfile.sh ord  /dev/rdsk/hdc5204  12M &
addfile.sh ord  /dev/rdsk/hdc5304  12M &
addfile.sh ord  /dev/rdsk/hdc5404  12M &
addfile.sh ord  /dev/rdsk/hdc5504  12M &
addfile.sh ord  /dev/rdsk/hdc6004  12M &
wait
addfile.sh ord  /dev/rdsk/hdc6104  12M &
addfile.sh ord  /dev/rdsk/hdc6204  12M &
addfile.sh ord  /dev/rdsk/hdd1004  12M &
addfile.sh ord  /dev/rdsk/hdd1104  12M &
addfile.sh ord  /dev/rdsk/hdd1204  12M &
addfile.sh ord  /dev/rdsk/hdd1304  12M &
addfile.sh ord  /dev/rdsk/hdd1404  12M &
addfile.sh ord  /dev/rdsk/hdd1504  12M &
addfile.sh ord  /dev/rdsk/hdd2004  12M &
addfile.sh ord  /dev/rdsk/hdd2104  12M &
wait
addfile.sh ord  /dev/rdsk/hdd2204  12M &
addfile.sh ord  /dev/rdsk/hdd3004  12M &
addfile.sh ord  /dev/rdsk/hdd3104  12M &
addfile.sh ord  /dev/rdsk/hdd3204  12M &
addfile.sh ord  /dev/rdsk/hdd3304  12M &
addfile.sh ord  /dev/rdsk/hdd3404  12M &
addfile.sh ord  /dev/rdsk/hdd3504  12M &
addfile.sh ord  /dev/rdsk/hdd4004  12M &
addfile.sh ord  /dev/rdsk/hdd4104  12M &
addfile.sh ord  /dev/rdsk/hdd4204  12M &
wait
addfile.sh ord  /dev/rdsk/hde1004  12M &
addfile.sh ord  /dev/rdsk/hde1104  12M &
addfile.sh ord  /dev/rdsk/hde1204  12M &
addfile.sh ord  /dev/rdsk/hde1304  12M &
addfile.sh ord  /dev/rdsk/hde1404  12M &
addfile.sh ord  /dev/rdsk/hde1504  12M &
addfile.sh ord  /dev/rdsk/hde2004  12M &
addfile.sh ord  /dev/rdsk/hde2104  12M &
addfile.sh ord  /dev/rdsk/hde2204  12M &
wait

#  [NEW ORDER]
addfile.sh nord  /dev/rdsk/hda1105  4M &
addfile.sh nord  /dev/rdsk/hda1205  4M &
addfile.sh nord  /dev/rdsk/hda1305  4M &
addfile.sh nord  /dev/rdsk/hda1405  4M &
addfile.sh nord  /dev/rdsk/hda1505  4M &
addfile.sh nord  /dev/rdsk/hda2005  4M &
addfile.sh nord  /dev/rdsk/hda2105  4M &
addfile.sh nord  /dev/rdsk/hda2205  4M &
addfile.sh nord  /dev/rdsk/hda2305  4M &
addfile.sh nord  /dev/rdsk/hda3005  4M &
wait
addfile.sh nord  /dev/rdsk/hda3105  4M &
addfile.sh nord  /dev/rdsk/hda3205  4M &
addfile.sh nord  /dev/rdsk/hda3305  4M &
addfile.sh nord  /dev/rdsk/hda3405  4M &
addfile.sh nord  /dev/rdsk/hda3505  4M &
addfile.sh nord  /dev/rdsk/hda4005  4M &
addfile.sh nord  /dev/rdsk/hda4105  4M &
addfile.sh nord  /dev/rdsk/hda4205  4M &
addfile.sh nord  /dev/rdsk/hda5005  4M &
addfile.sh nord  /dev/rdsk/hda5105  4M &
wait
addfile.sh nord  /dev/rdsk/hda5205  4M &
addfile.sh nord  /dev/rdsk/hda5305  4M &
addfile.sh nord  /dev/rdsk/hda5405  4M &
addfile.sh nord  /dev/rdsk/hda5505  4M &
addfile.sh nord  /dev/rdsk/hda6005  4M &
addfile.sh nord  /dev/rdsk/hda6105  4M &
addfile.sh nord  /dev/rdsk/hda6205  4M &
addfile.sh nord  /dev/rdsk/hdb1005  4M &
addfile.sh nord  /dev/rdsk/hdb1105  4M &
addfile.sh nord  /dev/rdsk/hdb1205  4M &
wait
addfile.sh nord  /dev/rdsk/hdb1305  4M &
addfile.sh nord  /dev/rdsk/hdb1405  4M &
addfile.sh nord  /dev/rdsk/hdb1505  4M &
addfile.sh nord  /dev/rdsk/hdb2005  4M &
addfile.sh nord  /dev/rdsk/hdb2105  4M &
addfile.sh nord  /dev/rdsk/hdb2205  4M &
addfile.sh nord  /dev/rdsk/hdb3005  4M &
```

```
addfile.sh nord  /dev/rdsk/hdb3105   4M &
addfile.sh nord  /dev/rdsk/hdb3205   4M &
addfile.sh nord  /dev/rdsk/hdb3305   4M &
wait
addfile.sh nord  /dev/rdsk/hdb3405   4M &
addfile.sh nord  /dev/rdsk/hdb3505   4M &
addfile.sh nord  /dev/rdsk/hdb4005   4M &
addfile.sh nord  /dev/rdsk/hdb4105   4M &
addfile.sh nord  /dev/rdsk/hdb4205   4M &
addfile.sh nord  /dev/rdsk/hdc1005   4M &
addfile.sh nord  /dev/rdsk/hdc1105   4M &
addfile.sh nord  /dev/rdsk/hdc1205   4M &
addfile.sh nord  /dev/rdsk/hdc1305   4M &
addfile.sh nord  /dev/rdsk/hdc1405   4M &
wait
addfile.sh nord  /dev/rdsk/hdc1505   4M &
addfile.sh nord  /dev/rdsk/hdc2005   4M &
addfile.sh nord  /dev/rdsk/hdc2105   4M &
addfile.sh nord  /dev/rdsk/hdc2205   4M &
addfile.sh nord  /dev/rdsk/hdc3005   4M &
addfile.sh nord  /dev/rdsk/hdc3105   4M &
addfile.sh nord  /dev/rdsk/hdc3205   4M &
addfile.sh nord  /dev/rdsk/hdc3305   4M &
addfile.sh nord  /dev/rdsk/hdc3405   4M &
addfile.sh nord  /dev/rdsk/hdc3505   4M &
wait
addfile.sh nord  /dev/rdsk/hdc4005   4M &
addfile.sh nord  /dev/rdsk/hdc4105   4M &
addfile.sh nord  /dev/rdsk/hdc4205   4M &
addfile.sh nord  /dev/rdsk/hdc5005   4M &
addfile.sh nord  /dev/rdsk/hdc5105   4M &
addfile.sh nord  /dev/rdsk/hdc5205   4M &
addfile.sh nord  /dev/rdsk/hdc5305   4M &
addfile.sh nord  /dev/rdsk/hdc5405   4M &
addfile.sh nord  /dev/rdsk/hdc5505   4M &
addfile.sh nord  /dev/rdsk/hdc6005   4M &
wait
addfile.sh nord  /dev/rdsk/hdc6105   4M &
addfile.sh nord  /dev/rdsk/hdc6205   4M &
addfile.sh nord  /dev/rdsk/hdd1005   4M &
addfile.sh nord  /dev/rdsk/hdd1105   4M &
addfile.sh nord  /dev/rdsk/hdd1205   4M &
addfile.sh nord  /dev/rdsk/hdd1305   4M &
addfile.sh nord  /dev/rdsk/hdd1405   4M &
addfile.sh nord  /dev/rdsk/hdd1505   4M &
addfile.sh nord  /dev/rdsk/hdd2005   4M &
addfile.sh nord  /dev/rdsk/hdd2105   4M &
wait
addfile.sh nord  /dev/rdsk/hdd2205   4M &
addfile.sh nord  /dev/rdsk/hdd3005   4M &
addfile.sh nord  /dev/rdsk/hdd3105   4M &
addfile.sh nord  /dev/rdsk/hdd3205   4M &
addfile.sh nord  /dev/rdsk/hdd3305   4M &
addfile.sh nord  /dev/rdsk/hdd3405   4M &
addfile.sh nord  /dev/rdsk/hdd3505   4M &
addfile.sh nord  /dev/rdsk/hdd4005   4M &
addfile.sh nord  /dev/rdsk/hdd4105   4M &
addfile.sh nord  /dev/rdsk/hdd4205   4M &
wait
addfile.sh nord  /dev/rdsk/hde1005   4M &
addfile.sh nord  /dev/rdsk/hde1105   4M &
addfile.sh nord  /dev/rdsk/hde1205   4M &
addfile.sh nord  /dev/rdsk/hde1305   4M &
addfile.sh nord  /dev/rdsk/hde1405   4M &
addfile.sh nord  /dev/rdsk/hde1505   4M &
addfile.sh nord  /dev/rdsk/hde2005   4M &
addfile.sh nord  /dev/rdsk/hde2105   4M &
addfile.sh nord  /dev/rdsk/hde2205   4M &
wait

#  [ORDER LINE]
addfile.sh ordl  /dev/rdsk/hde3101  931M &
addfile.sh ordl  /dev/rdsk/hde3201  931M &
addfile.sh ordl  /dev/rdsk/hde3301  931M &
addfile.sh ordl  /dev/rdsk/hde3401  931M &
addfile.sh ordl  /dev/rdsk/hde3501  931M &
addfile.sh ordl  /dev/rdsk/hde4001  931M &
addfile.sh ordl  /dev/rdsk/hde4101  931M &
addfile.sh ordl  /dev/rdsk/hde4201  931M &
addfile.sh ordl  /dev/rdsk/hde4301  931M &
addfile.sh ordl  /dev/rdsk/hde4401  931M &
wait
addfile.sh ordl  /dev/rdsk/hde4501  931M &
addfile.sh ordl  /dev/rdsk/hde5001  931M &
addfile.sh ordl  /dev/rdsk/hde5101  931M &
addfile.sh ordl  /dev/rdsk/hde5201  931M &
addfile.sh ordl  /dev/rdsk/hde5301  931M &
addfile.sh ordl  /dev/rdsk/hde5401  931M &
addfile.sh ordl  /dev/rdsk/hde5501  931M &
addfile.sh ordl  /dev/rdsk/hde6001  931M &
addfile.sh ordl  /dev/rdsk/hde6101  931M &
wait

#  [ORDER LINE INDEX]
addfile.sh iordl  /dev/rdsk/hda2501  576M &
addfile.sh iordl  /dev/rdsk/hda4401  576M &
addfile.sh iordl  /dev/rdsk/hda6401  576M &
addfile.sh iordl  /dev/rdsk/hdb2401  576M &
addfile.sh iordl  /dev/rdsk/hdb4401  576M &
addfile.sh iordl  /dev/rdsk/hdc2401  576M &
addfile.sh iordl  /dev/rdsk/hdc4401  576M &
addfile.sh iordl  /dev/rdsk/hdc6401  576M &
wait
addfile.sh iordl  /dev/rdsk/hdd2401  576M &
addfile.sh iordl  /dev/rdsk/hdd4401  576M &
addfile.sh iordl  /dev/rdsk/hde2401  576M &
addfile.sh iordl  /dev/rdsk/hdf2101  576M &
addfile.sh iordl  /dev/rdsk/hdf2201  576M &
addfile.sh iordl  /dev/rdsk/hdf4101  576M &
addfile.sh iordl  /dev/rdsk/hdf4201  576M &
wait

#  [NEW ORDER INDEX]
addfile.sh inord  /dev/rdsk/hda2502   26M &
addfile.sh inord  /dev/rdsk/hda4402   26M &
addfile.sh inord  /dev/rdsk/hda6402   26M &
addfile.sh inord  /dev/rdsk/hdb2402   26M &
addfile.sh inord  /dev/rdsk/hdb4402   26M &
addfile.sh inord  /dev/rdsk/hdc2402   26M &
addfile.sh inord  /dev/rdsk/hdc4402   26M &
addfile.sh inord  /dev/rdsk/hdc6402   26M &
wait
addfile.sh inord  /dev/rdsk/hdd2402   26M &
addfile.sh inord  /dev/rdsk/hdd4402   26M &
addfile.sh inord  /dev/rdsk/hde2402   26M &
addfile.sh inord  /dev/rdsk/hdf2102   26M &
addfile.sh inord  /dev/rdsk/hdf2202   26M &
addfile.sh inord  /dev/rdsk/hdf4102   26M &
addfile.sh inord  /dev/rdsk/hdf4202   26M &
wait

#  [ORDER INDEX 1]
addfile.sh iord1  /dev/rdsk/hda2503   57M &
addfile.sh iord1  /dev/rdsk/hda4403   57M &
addfile.sh iord1  /dev/rdsk/hda6403   57M &
addfile.sh iord1  /dev/rdsk/hdb2403   57M &
addfile.sh iord1  /dev/rdsk/hdb4403   57M &
addfile.sh iord1  /dev/rdsk/hdc2403   57M &
addfile.sh iord1  /dev/rdsk/hdc4403   57M &
addfile.sh iord1  /dev/rdsk/hdc6403   57M &
wait
addfile.sh iord1  /dev/rdsk/hdd2403   57M &
addfile.sh iord1  /dev/rdsk/hdd4403   57M &
addfile.sh iord1  /dev/rdsk/hde2403   57M &
addfile.sh iord1  /dev/rdsk/hdf2103   57M &
addfile.sh iord1  /dev/rdsk/hdf2203   57M &
addfile.sh iord1  /dev/rdsk/hdf4103   57M &
addfile.sh iord1  /dev/rdsk/hdf4203   57M &
wait

#  [CUSTOMER INDEX 2]
addfile.sh icust2 /dev/rdsk/hda2504   63M &
addfile.sh icust2 /dev/rdsk/hda4404   63M &
addfile.sh icust2 /dev/rdsk/hda6404   63M &
addfile.sh icust2 /dev/rdsk/hdb2404   63M &
addfile.sh icust2 /dev/rdsk/hdb4404   63M &
addfile.sh icust2 /dev/rdsk/hdc2404   63M &
addfile.sh icust2 /dev/rdsk/hdc4404   63M &
addfile.sh icust2 /dev/rdsk/hdc6404   63M &
wait
addfile.sh icust2 /dev/rdsk/hdd2404   63M &
addfile.sh icust2 /dev/rdsk/hdd4404   63M &
addfile.sh icust2 /dev/rdsk/hde2404   63M &
addfile.sh icust2 /dev/rdsk/hdf2104   63M &
addfile.sh icust2 /dev/rdsk/hdf2204   63M &
addfile.sh icust2 /dev/rdsk/hdf4104   63M &
addfile.sh icust2 /dev/rdsk/hdf4204   63M &
wait

#  [ORDER INDEX 2]
addfile.sh iord2  /dev/rdsk/hda6501   60M &
addfile.sh iord2  /dev/rdsk/hdb2501   60M &
addfile.sh iord2  /dev/rdsk/hdb4501   60M &
addfile.sh iord2  /dev/rdsk/hdc2501   60M &
addfile.sh iord2  /dev/rdsk/hdc4501   60M &
wait
addfile.sh iord2  /dev/rdsk/hdc6501   60M &
addfile.sh iord2  /dev/rdsk/hdd2501   60M &
addfile.sh iord2  /dev/rdsk/hdd4501   60M &
addfile.sh iord2  /dev/rdsk/hde2501   60M &
wait

#  [STOCK INDEX]
addfile.sh istk   /dev/rdsk/hdf4401  679M

#
# run catalog if NO_CAT unset
#

if [ "$NO_CAT" = "" ]
then
sqldba <<!
        set echo off;
#       connect sys/change_on_install;
        connect internal
        @?/rdbms/admin/catalog;
        @?/rdbms/admin/catproc;
        @?/rdbms/admin/catparr;
        exit;
!
fi
```

```
*****************************************
              benchsetup.sh
*****************************************



#
#====================================
============================+
#       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_pro
c
TPCC_LOADER=$BENCH_HOME/tpcc/loader
TPCC_SCRIPTS=$BENCH_HOME/tpcc/scripts
TPCC_UTILS=$TPCC_SCRIPTS/utils

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

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

#
# Create database.
#

if [ "$NO_DB" = "" ]
then
  benchdb.sh
fi

switchlog.sh

#
#  Create tables.
#

if [ "$NO_TAB" = "" ]
then
  sqlplus system/manager
@$TPCC_SQL/tpcc_tab
  sqlplus system/manager
@$TPCC_SQL/tpcc_rol
fi

#
#  Load history, new-order, order, order-line tables
#

pload.sh -mu $MULT

switchlog.sh

#
#  Create customer and stock tables while loading
other tables
#

if [ "$NO_TAB" = "" ]
then
  sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_tab2 &
  sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_tab3 &
fi

wait

switchlog.sh

#
#  Load warehouse, district, item tables
#

tpccload -M $MULT -w
tpccload -M $MULT -d
tpccload -M $MULT -i

#
#  Load customer table
#

I=1
SW=1
EW=40
INC=40
while [ $I -le 10 ]
do
  tpccload -M $MULT -c -b $SW -e $EW &
  I=`expr $I + 1`
  SW=`expr $SW + $INC`
  EW=`expr $EW + $INC`
done

wait

#
#  Load stock table
#

tpccload -M $MULT -S -j      1 -k   2500 &
tpccload -M $MULT -S -j   2501 -k   5000 &
tpccload -M $MULT -S -j   5001 -k   7500 &
tpccload -M $MULT -S -j   7501 -k  10000 &
tpccload -M $MULT -S -j  10001 -k  12500 &
tpccload -M $MULT -S -j  12501 -k  15000 &
tpccload -M $MULT -S -j  15001 -k  17500 &
tpccload -M $MULT -S -j  17501 -k  20000 &
tpccload -M $MULT -S -j  20001 -k  22500 &
tpccload -M $MULT -S -j  22501 -k  25000 &
tpccload -M $MULT -S -j  25001 -k  27500 &
tpccload -M $MULT -S -j  27501 -k  30000 &
tpccload -M $MULT -S -j  30001 -k  32500 &
tpccload -M $MULT -S -j  32501 -k  35000 &
tpccload -M $MULT -S -j  35001 -k  37500 &
tpccload -M $MULT -S -j  37501 -k  40000 &
tpccload -M $MULT -S -j  40001 -k  42500 &
tpccload -M $MULT -S -j  42501 -k  45000 &
tpccload -M $MULT -S -j  45001 -k  47500 &
tpccload -M $MULT -S -j  47501 -k  50000 &
wait
tpccload -M $MULT -S -j  50001 -k  52500 &
tpccload -M $MULT -S -j  52501 -k  55000 &
tpccload -M $MULT -S -j  55001 -k  57500 &
tpccload -M $MULT -S -j  57501 -k  60000 &
tpccload -M $MULT -S -j  60001 -k  62500 &
tpccload -M $MULT -S -j  62501 -k  65000 &
tpccload -M $MULT -S -j  65001 -k  67500 &
tpccload -M $MULT -S -j  67501 -k  70000 &
tpccload -M $MULT -S -j  70001 -k  72500 &
tpccload -M $MULT -S -j  72501 -k  75000 &
tpccload -M $MULT -S -j  75001 -k  77500 &
tpccload -M $MULT -S -j  77501 -k  80000 &
tpccload -M $MULT -S -j  80001 -k  82500 &
tpccload -M $MULT -S -j  82501 -k  85000 &
tpccload -M $MULT -S -j  85001 -k  87500 &
tpccload -M $MULT -S -j  87501 -k  90000 &
tpccload -M $MULT -S -j  90001 -k  92500 &
tpccload -M $MULT -S -j  92501 -k  95000 &
```

```
tpccload -M $MULT -S -j  95001 -k  97500 &
tpccload -M $MULT -S -j  97501 -k 100000 &
wait

switchlog.sh

#
#  Create indexes
#

if [ "$NO_IND" = "" ]
then
  sqldba <<!
    connect internal
    alter tablespace temp
      default storage (initial 100M next 100M
pctincrease 0);
    exit;
!
  sqlplus tpcc/tpcc @$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;
!
fi

alter.sh

#
#  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
#

$TPCC_UTILS/ext_all.sh >
$TPCC_OUTPUT/ext_all.out 2>&1

$TPCC_UTILS/space_init.sh
$TPCC_UTILS/space_get.sh 4600 400
$TPCC_UTILS/space_rpt.sh
$TPCC_OUTPUT/space.rpt


sqlplus system/manager <<!
  alter user tpcc temporary tablespace ltemp;
  quit;
!

sqlplus sys/change_on_install <<!
  grant execute on dbms_lock to public;
  grant execute on dbms_pipe to public;
  quit;
!

alter.sh

mirror.sh 1 /dev/rdsk/hdf3101 &
mirror.sh 2 /dev/rdsk/hdf1101 &
wait

#
#  Shutdown database
#

sqldba <<!
  connect internal;
  shutdown;
  exit;
!




****************************************
            create.sh
****************************************


#
#=====================================
============================+
# FILENAME
#     create.sh
# DESCRIPTION
#     create  a  new  tablespace
# USAGE
#     create.sh <tablespace> <data file> <size>
#=====================================
============================*/

sqldba <<!
  connect internal
  create tablespace $1 datafile '$2' size $3 reuse;
  exit;
!




****************************************
            mirror.sh
****************************************


#
#=====================================
============================+
# FILENAME
#     mirror.sh
# DESCRIPTION
#     Mirroring Redo-Log
# USAGE
#     mirror.sh <log group #> <data file>
#=====================================
============================*/

sqldba <<!
  connect internal
  alter database add logfile member '$2' reuse to
group $1;
  exit;
!




****************************************
            pload.sh
****************************************


#
#=====================================
============================+
#       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_pro
c
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

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

#
#  Load history table
#

I=1
while [ $I -le 10 ]
do
  J1=`expr \( $I - 1 \) \* 10 + 1`
  J2=`expr $J1 + 1`
  J3=`expr $J2 + 1`
  J4=`expr $J3 + 1`
  J5=`expr $J4 + 1`
  J6=`expr $J5 + 1`
  J7=`expr $J6 + 1`
  J8=`expr $J7 + 1`
  J9=`expr $J8 + 1`
  J10=`expr $J9 + 1`
  I1=`expr \( $I - 1 \) \* 40 + 1`
  I2=`expr $I1 + 4`
  I3=`expr $I2 + 4`
  I4=`expr $I3 + 4`
  I5=`expr $I4 + 4`
  I6=`expr $I5 + 4`
  I7=`expr $I6 + 4`
  I8=`expr $I7 + 4`
  I9=`expr $I8 + 4`
  I10=`expr $I9 + 4`
  I1E=`expr $I1 + 3`
  I2E=`expr $I2 + 3`
  I3E=`expr $I3 + 3`
  I4E=`expr $I4 + 3`
  I5E=`expr $I5 + 3`
  I6E=`expr $I6 + 3`
  I7E=`expr $I7 + 3`
  I8E=`expr $I8 + 3`
  I9E=`expr $I9 + 3`
  I10E=`expr $I10 + 3`
  tpccload -M $MULT -h -g -b  $I1 -e  $I1E >
/ora7322/data${J1}/hist${J1}.dat &
  tpccload -M $MULT -h -g -b  $I2 -e  $I2E >
/ora7322/data${J2}/hist${J2}.dat &
  tpccload -M $MULT -h -g -b  $I3 -e  $I3E >
/ora7322/data${J3}/hist${J3}.dat &
  tpccload -M $MULT -h -g -b  $I4 -e  $I4E >
/ora7322/data${J4}/hist${J4}.dat &
  tpccload -M $MULT -h -g -b  $I5 -e  $I5E >
/ora7322/data${J5}/hist${J5}.dat &
  tpccload -M $MULT -h -g -b  $I6 -e  $I6E >
/ora7322/data${J6}/hist${J6}.dat &
  tpccload -M $MULT -h -g -b  $I7 -e  $I7E >
/ora7322/data${J7}/hist${J7}.dat &
  tpccload -M $MULT -h -g -b  $I8 -e  $I8E >
/ora7322/data${J8}/hist${J8}.dat &
  tpccload -M $MULT -h -g -b  $I9 -e  $I9E >
/ora7322/data${J9}/hist${J9}.dat &
  tpccload -M $MULT -h -g -b  $I10 -e  $I10E >
/ora7322/data${J10}/hist${J10}.dat &
  wait
  I=`expr $I + 1`
done

I=1
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hda1001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hda3001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hda5001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hdb1001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hdb3001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hdc1001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hdc3001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hdc5001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hdd1001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hdd3001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hde1001 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hda1101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hda3101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hda5101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
      bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
      file=/dev/rdsk/hdb1101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
```

```
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdb3101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc1101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc3101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc5101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd1101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd3101 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hde1101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda1201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda3201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda5201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdb1201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdb3201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc1201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc3201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc5201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd1201 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd3201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hde1201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda1301 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda3301 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda5301 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdb1301 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdb3301 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc1301 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc3301 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc5301 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd1301 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd3301 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hde1301 &
```

```
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hda1401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hda3401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hda5401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdb1401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdb3401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdc1401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdc3401 &
l=`expr $l + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdc5401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdd1401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdd3401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hde1401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hda1501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hda3501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hda5501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdb1501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdb3501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdc1501 &
l=`expr $l + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdc3501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdc5501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdd1501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdd3501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hde1501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hda2001 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hda4001 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hda6001 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdb2001 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdb4001 &
l=`expr $l + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
    bad=hist${l}.bad
data=/ora7322/data${l}/hist${l}.dat
discard=hist${l}.dsc \
    file=/dev/rdsk/hdc2001 &
l=`expr $l + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${l}.log \
```

```
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc4001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc6001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd2001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd4001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hde2001 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda2101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda4101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda6101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdb2101 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdb4101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc2101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc4101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc6101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd2101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd4101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hde2101 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda2201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda4201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda6201 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdb2201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdb4201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc2201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc4201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdc6201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd2201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hdd4201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hde2201 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist${I}.log \
    bad=hist${I}.bad
data=/ora7322/data${I}/hist${I}.dat
discard=hist${I}.dsc \
    file=/dev/rdsk/hda2301 &
wait

I=1
while [ $I -le 100 ]
do
  rm -f /ora7322/data${I}/hist${I}.dat &
  I=`expr $I + 1`
done
```

```
wait

#
#  Load new-order table
#

tpccload -M $MULT -n -g >
/ora7322/data1/neword1.dat
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword1.log \
      bad=neword1.bad
data=/ora7322/data1/neword1.dat
discard=neword1.dsc
rm -f /ora7322/data1/neword1.dat

#
#  Load order and order-line table
#

I=1
while [ $I -le 10 ]
do
  J1=`expr \( $I - 1 \) \* 10 + 1`
  J2=`expr $J1 + 1`
  J3=`expr $J2 + 1`
  J4=`expr $J3 + 1`
  J5=`expr $J4 + 1`
  J6=`expr $J5 + 1`
  J7=`expr $J6 + 1`
  J8=`expr $J7 + 1`
  J9=`expr $J8 + 1`
  J10=`expr $J9 + 1`
  I1=`expr \( $I - 1 \) \* 40 + 1`
  I2=`expr $I1 + 4`
  I3=`expr $I2 + 4`
  I4=`expr $I3 + 4`
  I5=`expr $I4 + 4`
  I6=`expr $I5 + 4`
  I7=`expr $I6 + 4`
  I8=`expr $I7 + 4`
  I9=`expr $I8 + 4`
  I10=`expr $I9 + 4`
  I1E=`expr $I1 + 3`
  I2E=`expr $I2 + 3`
  I3E=`expr $I3 + 3`
  I4E=`expr $I4 + 3`
  I5E=`expr $I5 + 3`
  I6E=`expr $I6 + 3`
  I7E=`expr $I7 + 3`
  I8E=`expr $I8 + 3`
  I9E=`expr $I9 + 3`
  I10E=`expr $I10 + 3`
  tpccload -M $MULT -o
/ora7322/data${J1}/ordline${J1}.dat  -g -b $I1  -e
$I1E  > /ora7322/data${J1}/order${J1}.dat &
  tpccload -M $MULT -o
/ora7322/data${J2}/ordline${J2}.dat  -g -b $I2  -e
$I2E  > /ora7322/data${J2}/order${J2}.dat &
  tpccload -M $MULT -o
/ora7322/data${J3}/ordline${J3}.dat  -g -b $I3  -e
$I3E  > /ora7322/data${J3}/order${J3}.dat &
  tpccload -M $MULT -o
/ora7322/data${J4}/ordline${J4}.dat  -g -b $I4  -e
$I4E  > /ora7322/data${J4}/order${J4}.dat &
  tpccload -M $MULT -o
/ora7322/data${J5}/ordline${J5}.dat  -g -b $I5  -e
$I5E  > /ora7322/data${J5}/order${J5}.dat &
  tpccload -M $MULT -o
/ora7322/data${J6}/ordline${J6}.dat  -g -b $I6  -e
$I6E  > /ora7322/data${J6}/order${J6}.dat &
  tpccload -M $MULT -o
/ora7322/data${J7}/ordline${J7}.dat  -g -b $I7  -e
$I7E  > /ora7322/data${J7}/order${J7}.dat &
  tpccload -M $MULT -o
/ora7322/data${J8}/ordline${J8}.dat  -g -b $I8  -e
$I8E  > /ora7322/data${J8}/order${J8}.dat &
  tpccload -M $MULT -o
/ora7322/data${J9}/ordline${J9}.dat  -g -b $I9  -e
$I9E  > /ora7322/data${J9}/order${J9}.dat &
  tpccload -M $MULT -o
/ora7322/data${J10}/ordline${J10}.dat -g -b $I10 -e
$I10E > /ora7322/data${J10}/order${J10}.dat &
  wait
  I=`expr $I + 1`
done

I=1
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hda1004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hda3004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hda5004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hdb1004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hdb3004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hdc1004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hdc3004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hdc5004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hdd1004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hdd3004 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hde1004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hda1104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hda3104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hda5104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hdb1104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
      bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
      file=/dev/rdsk/hdb3104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
```

```
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc1104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc3104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc5104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd1104 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd3104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hde1104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda1204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda3204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda5204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb1204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb3204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc1204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc3204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc5204 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd1204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd3204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hde1204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda1304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda3304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda5304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb1304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb3304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc1304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc3304 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc5304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd1304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd3304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hde1304 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda1404 &
```

```
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda3404 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda5404 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb1404 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb3404 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc1404 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc3404 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc5404 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd1404 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd3404 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hde1404 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda1504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda3504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda5504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb1504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb3504 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc1504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc3504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc5504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd1504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd3504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hde1504 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda2004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda4004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda6004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb2004 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb4004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc2004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc4004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
```

```
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc6004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd2004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd4004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hde2004 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda2104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda4104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda6104 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb2104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb4104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc2104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc4104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc6104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd2104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd4104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hde2104 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda2204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda4204 &
I=`expr $I + 1`
wait
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda6204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb2204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdb4204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc2204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc4204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdc6204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd2204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hdd4204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hde2204 &
I=`expr $I + 1`
sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order${I}.log \
    bad=order${I}.bad
data=/ora7322/data${I}/order${I}.dat
discard=order${I}.dsc \
    file=/dev/rdsk/hda2304 &
wait

I=1
while [ $I -le 100 ]
do
  rm -f /ora7322/data${I}/order${I}.dat &
  I=`expr $I + 1`
done
wait

I=1
while [ $I -le 20 ]
do
  I1=`expr \( $I - 1 \) \* 5 + 1`
  I2=`expr $I1 + 1`
```

```
  l3=`expr $l2 + 1`
  l4=`expr $l3 + 1`
  l5=`expr $l4 + 1`

  cat /ora7322/data${l1}/ordline${l1}.dat \
      /ora7322/data${l2}/ordline${l2}.dat \
      /ora7322/data${l3}/ordline${l3}.dat \
      /ora7322/data${l4}/ordline${l4}.dat \
      /ora7322/data${l5}/ordline${l5}.dat >
/ora7322/data${l}/ordline_mul${l}.dat

  l=`expr $l + 1`
done

l=1
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde3001 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde3101 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde3201 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde3301 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde3401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde3501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde4001 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde4101 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde4201 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde4301 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde4401 &
l=`expr $l + 1`
wait
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde4501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde5001 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde5101 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde5201 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde5301 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde5401 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde5501 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde6001 &
l=`expr $l + 1`
sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline${l}.log \
      bad=ordline${l}.bad
data=/ora7322/data${l}/ordline_mul${l}.dat
discard=ordline${l}.dsc \
      file=/dev/rdsk/hde6101 &
wait

l=1
while [ $l -le 100 ]
do
  rm -f /ora7322/data${l}/ordline${l}.dat &
  l=`expr $l + 1`
done
l=1
while [ $l -le 20 ]
do
  rm -f /ora7322/data${l}/ordline_mul${l}.dat &
  l=`expr $l + 1`
done
wait




****************************************
            tpccload.c
****************************************


/*=================================
============================+
```

```
|       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 () % 1000) <= 110) {
      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);
}
```

```
****************************************
            p_build.ora
****************************************


#
#===================================
============================+
#       Copyright (c) 1995  Oracle Corp, Redwood
Shores, CA       |
#                OPEN SYSTEMS PERFORMANCE
GROUP                 |
#                    All Rights Reserved
|
#===================================
============================+
# FILENAME
#      p_build.ora
# DESCRIPTION
#      Oracle parameter file for building TPC-C
database.
#===================================
============================
#

db_writers                      = 25
sort_area_size                                  = 2097152
parallel_server_idle_time = 30
parallel_max_servers = 60
checkpoint_process      = TRUE
compatible              = 7.3.2.0.0
db_name                 = tpcc
db_files                        = 1022
db_file_multiblock_read_count = 32
db_block_buffers        = 30000
_db_block_write_batch   = 128
db_block_checkpoint_batch = 512
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
max_rollback_segments = 220
processes               = 200
sessions                = 400
transactions            = 400
distributed_transactions = 0
transactions_per_rollback_segment = 1
```

```
rollback_segments       = (s1, s2, s3, s4, s5, s6,
s7, s8, s9, s10, s11, s12, s13, s14, s15, s16, s17,
s18, s19, s20, s21, s22, s23, s24, s25, s26, s27,
s28, s29, s30)
shared_pool_size          = 7000000
discrete_transactions_enabled = FALSE
cursor_space_for_time   = TRUE




****************************************
            p_create.ora
****************************************



#
#===================================
============================+
#       Copyright (c) 1995  Oracle Corp, Redwood
Shores, CA      |
#                OPEN SYSTEMS PERFORMANCE
GROUP                |
#                    All Rights Reserved
|
#===================================
============================+
# FILENAME
#     p_create.ora
# DESCRIPTION
#     Oracle parameter file for creating TPC-C
database.
#===================================
============================
#
compatible              = 7.3.2.0.0
db_name                 = tpcc
db_files                        = 1000
db_block_buffers        = 2000
dml_locks               = 500
log_checkpoint_interval = 999999999
log_buffer              = 32768
sessions                = 70
processes               = 50
transactions            = 50




****************************************
            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

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;
quit;




****************************************
            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

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 9M next 9M 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 31M next 31M pctincrease 0)
pctfree 1;

create unique index istock on stock(s_i_id,
s_w_id)
      tablespace istk
      initrans 3
      parallel 10
      storage (initial 15M next 15M pctincrease 0)
pctfree 1;

exit;




****************************************
            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

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 28M next 28M 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 29M next 29M 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 25600K next 25600K
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 115M next 115M pctincrease 0
            freelist groups 13 freelists 24);

exit;




****************************************
            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


  CREATE ROLLBACK SEGMENT t1
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t2
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t3
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t4
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t5
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t6
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t7
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t8
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t9
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t10
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t11
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t12
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t13
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t14
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t15
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t16
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t17
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t18
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t19
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t20
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t21
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t22
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t23
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t24
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t25
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t26
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t27
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t28
    TABLESPACE roll8
```

```
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t29
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t30
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t31
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t32
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t33
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t34
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t35
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t36
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t37
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t38
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t39
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t40
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t41
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t42
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t43
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t44
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t45
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t46
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t47
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t48
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t49
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t50
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t51
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t52
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t53
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t54
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t55
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t56
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t57
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t58
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t59
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t60
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t61
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t62
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t63
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t64
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t65
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t66
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t67
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t68
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t69
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t70
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t71
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t72
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t73
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t74
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t75
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t76
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t77
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t78
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t79
    TABLESPACE roll9
```

```
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t80
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t81
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t82
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t83
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t84
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t85
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t86
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t87
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t88
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t89
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t90
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t91
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t92
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t93
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t94
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t95
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t96
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t97
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t98
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t99
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t100
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t101
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t102
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t103
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t104
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t105
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t106
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t107
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t108
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t109
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t110
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t111
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t112
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t113
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t114
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t115
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t116
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t117
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t118
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t119
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t120
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t121
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t122
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t123
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t124
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t125
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t126
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t127
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t128
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t129
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t130
    TABLESPACE roll10
```

```
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t131
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t132
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t133
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t134
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t135
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t136
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t137
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t138
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t139
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t140
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t141
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t142
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t143
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t144
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t145
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t146
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t147
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t148
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t149
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t150
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t151
    TABLESPACE roll1
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t152
    TABLESPACE roll2
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t153
    TABLESPACE roll3
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t154
    TABLESPACE roll4
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t155
    TABLESPACE roll5
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t156
    TABLESPACE roll6
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t157
    TABLESPACE roll7
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t158
    TABLESPACE roll8
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t159
    TABLESPACE roll9
    STORAGE (initial 600K next 600K minextents
2);
  CREATE ROLLBACK SEGMENT t160
    TABLESPACE roll10
    STORAGE (initial 600K next 600K minextents
2);

  exit;



****************************************
            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 FIRST, create TPCC userid and connect to it.
rem
                grant connect,resource,unlimited
tablespace to tpcc identified by tpcc;
        alter user tpcc temporary tablespace temp;
                connect tpcc/tpcc

rem
rem NEXT, DROP all first
rem
        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 1000K 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 14M 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 11M 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  3584K 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 310M 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;

**************************************
            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
        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        12000000
        hash is         (c_w_id * 30000 + c_d_id *
3000 + c_id)
        size            850
        initrans        3
        pctfree         0
        tablespace      cust
        storage (initial 145M next 145M 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;



*****************************************
          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
    drop cluster scluster including tables;
            drop table stock;

set timing on

rem
rem STOCK table
rem

    create cluster scluster (
      s_i_id      number(6,0),
      s_w_id      number(4,0)
    )
    hashkeys      40000000
    hash is       (s_i_id * 400 + s_w_id)
    size          350
    initrans      3
    pctfree       0
    tablespace    stocks
    storage (initial 157M next 157M pctincrease
0 minextents 16);

            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;



*****************************************
          del.sql
*****************************************


rem
rem
================================================
============================+
rem         Copyright (c) 1993  Oracle Corp,
Belmont, 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);
    CURSOR n_cur IS
      SELECT no_o_id, rowid
      FROM new_order
      WHERE no_w_id = ware_id AND no_d_id =
dist_id AND no_o_id =
        (SELECT min(no_o_id)
         FROM new_order
         WHERE no_w_id = ware_id AND no_d_id
= dist_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
THEN
            ROLLBACK;
            retry := retry + 1;
        END;

      END LOOP;
    END LOOP;
  END;
END;
/

quit;




****************************************
            new.sql
****************************************


rem
rem
================================================
============================+
rem          Copyright (c) 1993  Oracle Corp,
Belmont, 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);
  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
THEN
          ROLLBACK;
          retry := retry + 1;
      END;
    END LOOP;
  END;
END;
/

quit;




****************************************
            ord.sql
****************************************


rem
rem
================================================
============================+
rem          Copyright (c) 1993  Oracle Corp,
Belmont, 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;
    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

    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;

  END;
END;
/

quit;
```

```
****************************************
            pay.sql
****************************************


rem
rem
=====================================
==========================+
rem          Copyright (c) 1993  Oracle Corp,
Belmont, 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);
    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_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

        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;

      cust_balance := cust_balance -
hist_amount;

      IF cust_credit = 'BC' THEN

        UPDATE customer
          SET c_balance = c_balance -
hist_amount,
              c_ytd_payment = c_ytd_payment +
hist_amount,
              c_payment_cnt = c_payment_cnt + 1,
              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

        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;

        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
THEN
          ROLLBACK;
          retry := retry + 1;
      END;

    END LOOP;
  END;
```

```
END;
/

quit;




****************************************
            sto.sql
****************************************


rem
rem
=====================================
============================+
rem           Copyright (c) 1993  Oracle Corp,
Belmont, 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
  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;
  END;
END;
/

quit;
```

Appendix F: 180 Day Space Calculations

TPM	4,718.73				Warehouses	400
SEGMENT	TYPE	TSPACE	BLOCKS	FIVE %	DAILY GROWTH	TOTAL
CUSTOMER	TABLE	CUST	6,000,009	300,000	0	6,300,009
DISTRICT	TABLE	WARE	4,004	200	0	4,204
HISTORY	TABLE	HIST	337,614	0	63,471	401,085
ICUSTOMER	INDEX	ICUST1	138,676	6,934	0	145,610
ICUSTOMER2	INDEX	ICUST2	316,823	15,841	0	332,664
IDISTRICT	INDEX	WARE	1,000	50	0	1,050
IITEM	INDEX	ITEMS	1,000	50	0	1,050
INEW_ORDER	INDEX	INORD	53,159	2,658	0	55,817
IORDERS	INDEX	IORD1	150,295	7,515	0	157,810
IORDERS2	INDEX	IORD2	233,620	11,681	0	245,301
IORDER_LINE	INDEX	IORDL	1,627,965	81,398	0	1,709,363
ISTOCK	INDEX	ISTK	418,008	20,900	0	438,908
ITEM	TABLE	ITEMS	6,667	333	0	7,000
IWAREHOUSE	INDEX	WARE	100	5	0	105
NEW_ORDER	TABLE	NORD	31,658	1,583	0	33,241
ORDERS	TABLE	ORD	240,305	0	45,177	285,482
ORDER_LINE	TABLE	ORDL	4,349,118	0	817,634	5,166,752
ROLL_SEG	SYS	ROLL	250,880	0	0	250,880
STOCK	TABLE	STOCKS	8,000,001	400,000	0	8,400,001
SYSTEM	SYS	SYSTEM	306,688	0	0	306,688
WAREHOUSE	TABLE	WARE	400	20	0	420
Total			**22,467,990**	**849,168**	**926,282**	**24,243,440**

Dynamic space	**4,927,037**	
Static space	**18,390,121**	
Free space	**926,282**	
Daily growth	**926,282**	
Daily spread	**0**	(Oracle may be configured such that daily spread is 0)
180-day space (blk.)	**185,120,881**	
Block size (bytes)	**2,048**	
180-day (GB)	**353.09**	
Log block size	**1,024**	
Log blocks/tpmC	**13.89**	(Number of log blocks used in one minute)
8-hour log (GB)	**30.00**	

DISKS PRICED		
SIZE	Count	Capacity (GB)
2.0 GB DISK	**178**	**356.0**
4.0 GB DISK	**16**	**64.0**
Total	**194**	**420.0**

SPACE USAGE (GB)	
180-day	**353.09**
Log(mirror)	**60.01**
OS,swap,etc	**1.31**
Total	**414.41**

Appendix G: Auditor's attestation letter