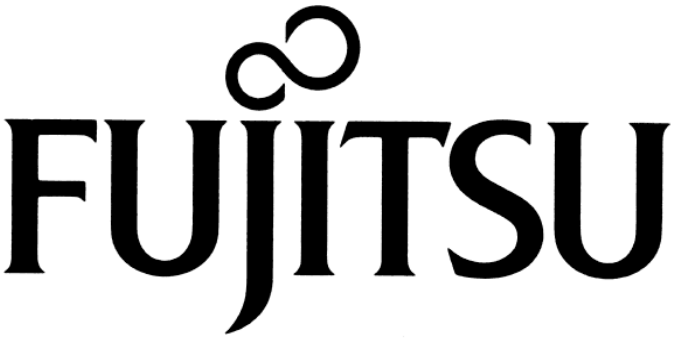


and

TPC Benchmark™C

Full Disclosure Report

Fujitsu
GRANPOWER 7000
Model 800 c/s w/ 29 Front-Ends

running

Oracle8 Enterprise Edition for VLM V80L10

January, 1998

The benchmark results contained in this document were submitted for compliance with version 3.3.2 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 February 6th, 1998.

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 February 6th, 1998

UXP/DS Basic Software V20L30 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, Oracle8, Pro*C and PL/SQL are trademarks of Oracle Corporation.

SPARC is a registered trademark of SPARC International.

Ultra is a trademark of Sun Microsystems, Inc.

Microsoft, Windows, MS-DOS and the Microsoft logo are registered trademarks of Microsoft 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 3.3.2, released June 15, 1997.

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.

ORACLE

FUJITSU

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 800 c/s w/ 29 Front-Ends. The operating system used for the benchmark was UXP/DS Basic Software V20L30. The DBMS used was Oracle8 Enterprise Edition for VLM V80L10.

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:

34,116.93 tpmC
¥57,883 per tpmC
July 31st, 1998

Standard and Executive Summary Statements

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

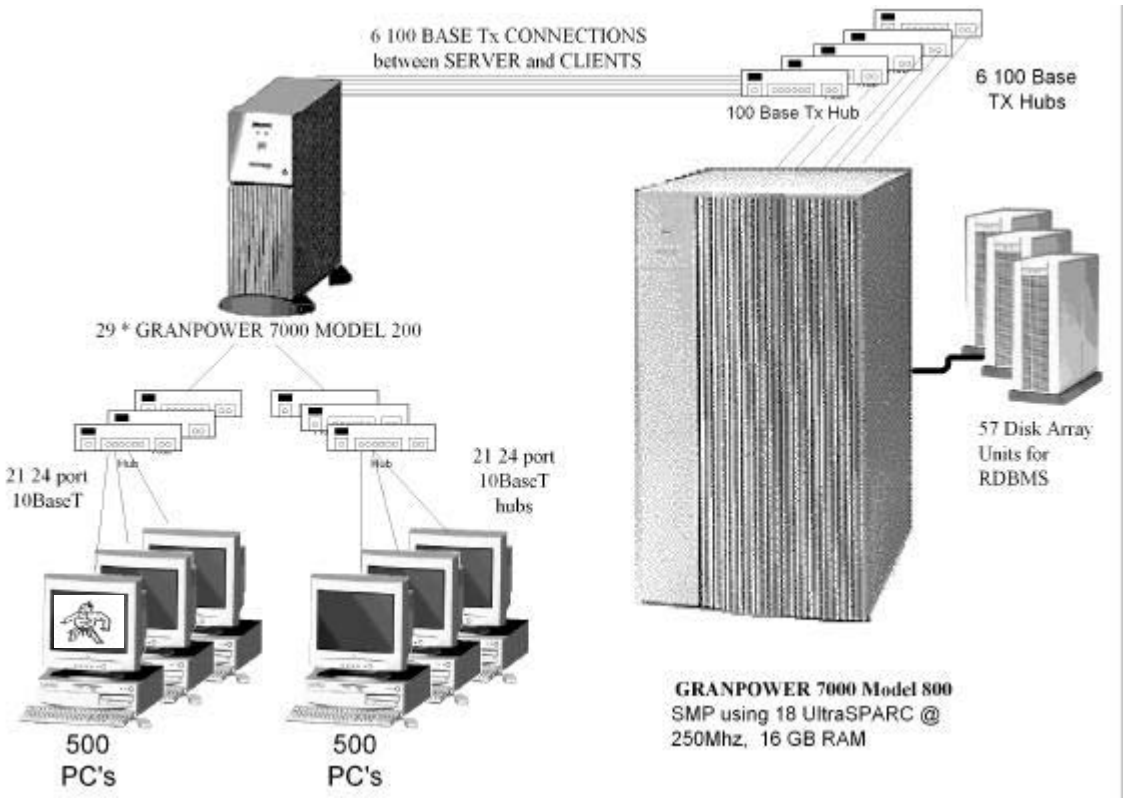
Auditor

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

Priced Configuration

FUJITSU ORACLE	GRANPOWER 7000 Model 800 c/s w/ 29 Front-Ends		TPC-C Rev 3.3.2 Report Date: February 6th, 1998
Total System Cost	**TPC-C Throughput**	**Price/Performance**	**Availability Date**
¥1,974,793,750	34,116.93 tpmC	¥57,883/tpmC	July 31st, 1998

Processors	Database Manager	Operating system	Other Software	Number of users
18 UltraSPARC @250Mhz	Oracle8 Enterprise Edition for VLM V80L10	UXP/DS Basic Software V20L30	TP-Base V20 LVCF V22 Oracle WebServer V02	29,000



RDBMS SERVER			CLIENTS	
SYSTEM COMPONENTS	QTY	DESCRIPTION	QTY	DESCRIPTION
PROCESSOR	18	ULTRASPARC @ 250MHZ	29	28 WITH 2 ULTRASPARC @ 250MHZ, 1 WITH 2 ULTRASPARC @ 300MHZ
CACHE MEMORY		4MB (EACH PROCESSOR)		2 MB (EACH) FOR 300MHZ, 1MB (EACH) FOR 250MHZ
MEMORY		16GB	29	256 MB
DISK CONTROLLER	56	WIDE SCSI-2	29	WIDE SCSI 2(1 CHANNEL)
DISKS	760	2.0GB DISK	29	2.0GB DISK
	380	4.0GB DISK		
TERMINAL	1	CONSOLE	58	CONSOLE
NETWORK INTERFACE	6	100 BASE-TX	58	10BASET
HUBS	6	8-PORT (100 BASE-TX)	1,218	CENTRECOM 3024TR HUB UNITS (24 PORTS)
PC'S			29,000	PC'S RUNNING WEB BROWSERS

Numerical Quantities Summary

GRANPOWER 7000 Model 800 c/s w/ 29 Front-Ends

Oracle8 Enterprise Edition for VLM V80L10

MQTH, Computed Maximum Qualified Throughput	34,116.93 tpmC		
Response Times (in seconds)	**Average**	**90%**	**Max.**
New-Order	**1.53**	**2.82**	**206.69**
Payment	**1.19**	**2.46**	**208.68**
Order-Status	**1.27**	**2.58**	**112.82**
Delivery (interactive portion)	**0.24**	**0.32**	**95.22**
Delivery (deferred portion)	**1.60**	**2.54**	**7.37**
Stock-Level	**1.69**	**3.46**	**148.51**
Menu	**0.30**	**0.32**	**227.87**

Transaction Mix, in percent of total transaction	
New-Order	**44.68**
Payment	**43.16**
Order-Status	**4.07**
Delivery	**4.05**
Stock-Level	**4.04**

Emulation Delay (in seconds)	**Resp. Time**	**Menu**
New-Order	0.1	0.1
Payment	0.1	0.1
Order-Status	0.1	0.1
Delivery (interactive)	0.1	0.1
Stock-Level	0.1	0.1

Keying/Think Times (in seconds)	**Min.**		**Average**		**Max.**	
New-Order	18.09	0.01	18.11	12.10	18.38	121.18
Payment	3.04	0.01	3.06	12.12	3.32	121.19
Order-Status	2.04	0.01	2.06	10.31	2.32	100.75
Delivery (interactive)	2.04	0.01	2.06	5.10	2.32	49.88
Stock-Level	2.04	0.01	2.06	5.14	2.32	49.73

Test Duration	
Ramp-up time (seconds)	**1630**
Measurement interval	**1800**
Transactions during measurement interval	**1,023,508**
Ramp down time	

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

Reproducibility Run	
Reported measurement	**34,116.93**
Reproducibility measurement	**33,776.27**
Difference	**340.66**

Detailed Pricing Information

FUJITSU
ORACLE

Detailed Pricing Information

GRANPOWER 7000

Model 800 c/s w/ 29 Front-Ends

TPC-C Rev 3.3.2

Report Date:

February 6th, 1998

Order Number	Description	Quantity	Third Party	Unit Price	Extended Price	Maintenance rate/unit*		5 Years Maintenance
Server Hardware								
GP780A1	GRANPOWER 7000 model 800	1		60,000,000	60,000,000	250,000	/mo	13,500,000
	4 UltraSPARC 250MHz/4MB cache + 2GB memory + 5 4GB/DE							
GP781A11	CPU boards w/ 4 UltraSPARC 250MHz/4MB cache	3		16,000,000	48,000,000	66,700	/mo	10,805,400
GP788SB1	SBus I/O boards	2		2,140,000	4,280,000	10,700	/mo	1,155,600
GP781A12	CPU board w/ 2 UltraSPARC 250MHz/4MB cache + Sbus I/O Board	1		9,070,000	9,070,000	38,700	/mo	2,089,800
GP782M61	Additional memory (1GB)	14		9,560,000	133,840,000	0	/mo	0
F7949RA31	External Cabinet	2		630,000	1,260,000	3,200	/mo	345,600
F7978SB2	SBus extention unit	4		900,000	3,600,000	4,500	/mo	972,000
F7978SB3	Additional SBus extention unit	3		600,000	1,800,000	3,000	/mo	486,000
F7958HS1	Wide SCSI-2 adapter	56		250,000	14,000,000	0	/mo	0
GP-D8S1	Disk Array unit w/ 5 2GB/DE	30		4,850,000	145,500,000	24,300	/mo	39,366,000
GP-D8HD1	Additional Disk Array drives (5 2GB/DE)	90		1,080,000	97,200,000	5,400	/mo	26,244,000
GP-D16S1	Disk Array unit w/ 5 4GB/DE	19		5,700,000	108,300,000	28,500	/mo	29,241,000
GP-D16HD1	Additional Disk Array drives (5 4GB/DE)	57		1,930,000	110,010,000	9,700	/mo	29,856,600
GP-DSC1	Additional Disk Array controller	40		870,000	34,800,000	4,400	/mo	9,504,000
F7915AR11	Disk Array unit w/ 5 2GB/DE	8		5,100,000	40,800,000	25,500	/mo	11,016,000
F7915AR31A	Additional Disk Array drives (5 2GB/DE)	24		1,200,000	28,800,000	6,000	/mo	7,776,000
F7915AR51	Additional Disk Array controller	8		950,000	7,600,000	4,800	/mo	2,073,600
F987AA11	Uninterrupted Power Supply	48		186,000	8,928,000	1,400	/mo	3,628,800
F7960A11	Display unit	1		380,000	380,000	1,500	/mo	81,000
SCBL-RSA05	RS-232 cable	1		25,000	25,000	0	/mo	0
GP723ET2	8mm Tape device	1		600,000	600,000	3,000	/mo	162,000
Server Software	**Server Hardware Subtotals**				**858,793,000**			**188,303,400**
D7831DK95	UXP/DS Basic Software V20	1		962,000	962,000	990,000	/yr	4,950,000
	Oracle8 Enterprise Edition for VLM V80L10 & SQL*Net	1	1	193,228,425	193,228,425	193,228,425	/5yr	193,228,425
B7836KM2N	LVCF V22	1		1,600,000	1,600,000	0	/yr	0
Client Hardware	**Server Software Subtotals**				**195,790,425**			**198,178,425**
GP720B2	GRANPOWER 7000 model 200 (250MHz, 2GB disk)	28		2,620,000	73,360,000	10,900	/yr	16,480,800
GP721B11	Additional CPU module (250MHz)	28		2,300,000	64,400,000	9,600	/yr	14,515,200
GP720B1	GRANPOWER 7000 model 200 (300MHz, 2GB disk)	1		3,320,000	3,320,000	13,800	/yr	745,200
GP721B11	Additional CPU module (300MHz)	1		3,000,000	3,000,000	12,500	/yr	675,000
GP722M31A	Additional memory (128MB)	58		880,000	51,040,000	0	/yr	0
F7958LA1	10Mbps LAN adapter	58		120,000	6,960,000	0	/yr	0
F7960A11	Display unit	29		380,000	11,020,000	1,500	/yr	2,349,000
DCBL-RCB05	RS-232 cable	29		16,000	464,000	0	/yr	0
Client Software	**Client Hardware Subtotals**				**213,564,000**			**34,765,200**
D7831DK6Z	UXP/DS Basic Software V20	1		2,062,000	2,062,000	440,000	/yr	2,200,000
S7831DK0Z	Additional user license	2		1,698,000	3,396,000	352,000	/yr	3,520,000
S7831DK0Z	Additional user license	3		1,698,000	5,094,000	286,000	/yr	4,290,000
S7831DK0Z	Additional user license	23		1,698,000	39,054,000	242,000	/yr	27,830,000
B783KE23A	Oracle7 WebServer V02	29		380,000	11,020,000	100,000	/yr	14,500,000
D783HZK60	TP-Base/sdk V20 (1 user)	1		300,000	300,000	33,000	/yr	165,000
D783HUK62	TP-Base/rt V20 (8 user)	1		500,000	500,000	27,500	/yr	137,500
S783HUK02	Additional user license	28		400,000	11,200,000	27,500	/yr	3,850,000
User Connectivity	**Client Software Subtotals**				**72,626,000**			**56,492,500**
SH2510	100BASE-TX Switching Hub units (8ports) *	8		700,000	5,600,000	3,500	/mo	1,512,000
3024TR	CentreCOM 3024TR Hub units (24ports) *	1340	2	89,800	120,332,000	5,380	/yr	28,836,800
	User Connectivity Subtotals				**125,932,000**			**30,348,800**
		Totals			**1,466,705,425**			**508,088,325**
	5 Year cost							**1,974,793,750**
	tpmC							34116.93
	Yen / tpmC							**57,883**

Third Party Pricing:
1 = Oracle Corporation Japan
2 = Allied Telesis, K.K.

Notes:

- Allied Telesys hubs have a 1 year warranty. An additional 48 months of maintenance has been included in the above pricing.
- 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.

Table Of Contents

PREFACE I

TPC BENCHMARK C OVERVIEW I

ABSTRACT III

OVERVIEW III
TPC BENCHMARK C METRICS III
STANDARD AND EXECUTIVE SUMMARY STATEMENTS III
AUDITOR III
PRICED CONFIGURATION IV
NUMERICAL QUANTITIES SUMMARY V
DETAILED PRICING INFORMATION VI

TABLE OF CONTENTS VII

GENERAL ITEMS 11

APPLICATION CODE AND DEFINITION STATEMENTS 11
TEST SPONSOR 11
PARAMETER SETTINGS 11
CONFIGURATION ITEMS 12

CLAUSE 1 RELATED ITEMS 15

1.1. TABLE DEFINITIONS 15
1.2. PHYSICAL ORGANIZATION OF DATABASE 15
1.3. INSERT AND DELETE OPERATIONS 15
1.4. PARTITIONING 16
1.5. REPLICATION, DUPLICATION OR ADDITIONS 16

CLAUSE 2 RELATED ITEMS 17

2.1 RANDOM NUMBER GENERATION 17
2.2 INPUT/OUTPUT SCREEN LAYOUT 17
2.3 PRICED TERMINAL FEATURE VERIFICATION 17
2.4 PRESENTATION MANAGER OR INTELLIGENT TERMINAL 18
2.5 TRANSACTION STATISTICS 18
2.6 QUEUEING MECHANISM 18

CLAUSE 3 RELATED ITEMS 19

3.1 TRANSACTION SYSTEM PROPERTIES (ACID) 19
3.2 ATOMICITY 19
3.3 CONSISTENCY 20
3.4 ISOLATION 20
3.5 DURABILITY 21

CLAUSE 4 RELATED ITEMS 23

4.1 INITIAL CARDINALITY OF TABLES 23
4.2 DATABASE LAYOUT 24
4.3 TYPE OF DATABASE 24
4.4 DATABASE MAPPING 24
4.5 180 DAY SPACE 24

CLAUSE 5 RELATED ITEMS 27

5.1 THROUGHPUT 27
5.2 RESPONSE TIMES 27
5.3 KEYING AND THINK TIMES 28
5.4 RESPONSE TIME FREQUENCY DISTRIBUTION CURVES AND OTHER GRAPHS 28
5.5 STEADY STATE DETERMINATION 33
5.6 WORK PERFORMED DURING STEADY STATE 33
5.7 REPRODUCIBILITY 34
5.8 MEASUREMENT PERIOD DURATION 34
5.9 REGULATION OF TRANSACTION MIX 34
5.10 TRANSACTION STATISTICS 34
5.11 CHECKPOINT COUNT AND LOCATION 35

CLAUSE 6 RELATED ITEMS 37

6.1 RTE DESCRIPTIONS 37
6.2 EMULATED COMPONENTS 37
6.3 FUNCTIONAL DIAGRAMS 37
6.4 NETWORKS 38
6.5 OPERATOR INTERVENTION 38

CLAUSE 7 RELATED ITEMS 39

7.1 SYSTEM PRICING 39
7.2 AVAILABILITY 39
7.3 THROUGHPUT AND PRICE PERFORMANCE 40
7.4 COUNTRY SPECIFIC PRICING 40
7.5 USAGE PRICING 40

CLAUSE 9 RELATED ITEMS 41

9.1 AUDITOR'S REPORT 41
9.2 AVAILABILITY OF THE FULL DISCLOSURE REPORT 41

APPENDIX A: CLIENT SOURCE CODE 43

APPENDIX B: SERVER SOURCE CODE 65

APPENDIX C: RTE SCRIPTS 107

APPENDIX D: SYSTEM TUNABLES 109

APPENDIX E: DATABASE CREATION CODE 121

APPENDIX F: DISTRIBUTION OF TABLES AND LOGS 155

APPENDIX G: 180 DAY SPACE CALCULATIONS 185

APPENDIX H: SCREEN LAYOUTS 187

APPENDIX I: PRICE QUOTES 198

APPENDIX J: AUDITORS ATTESTATION LETTER 199

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 and B contain 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

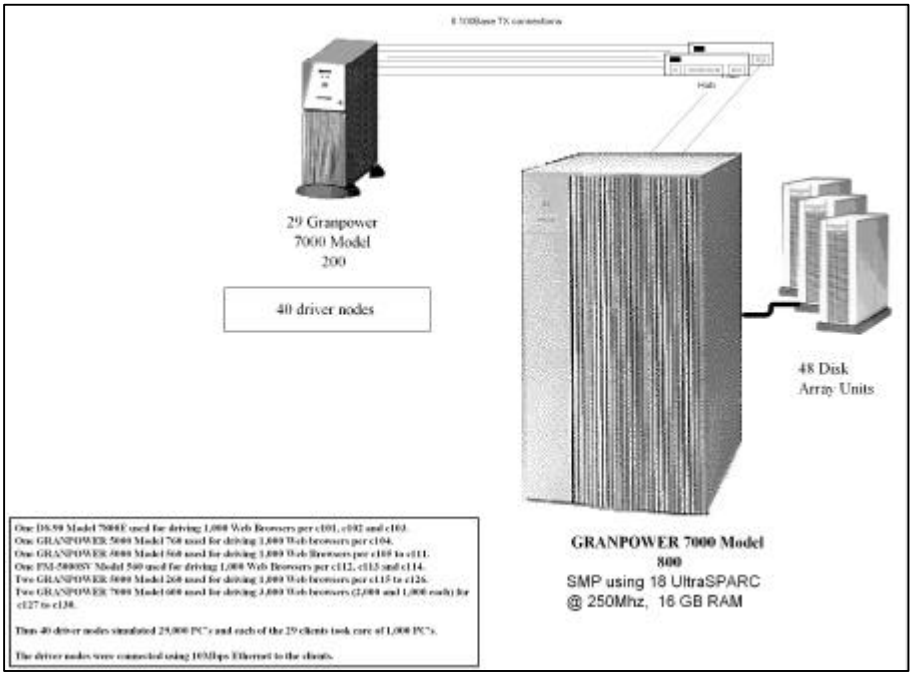
Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences.

The System Under Test (SUT), aGRANPOWER 7000 Model 800 c/s w/ 29 Front-Ends, is depicted in the following diagrams.

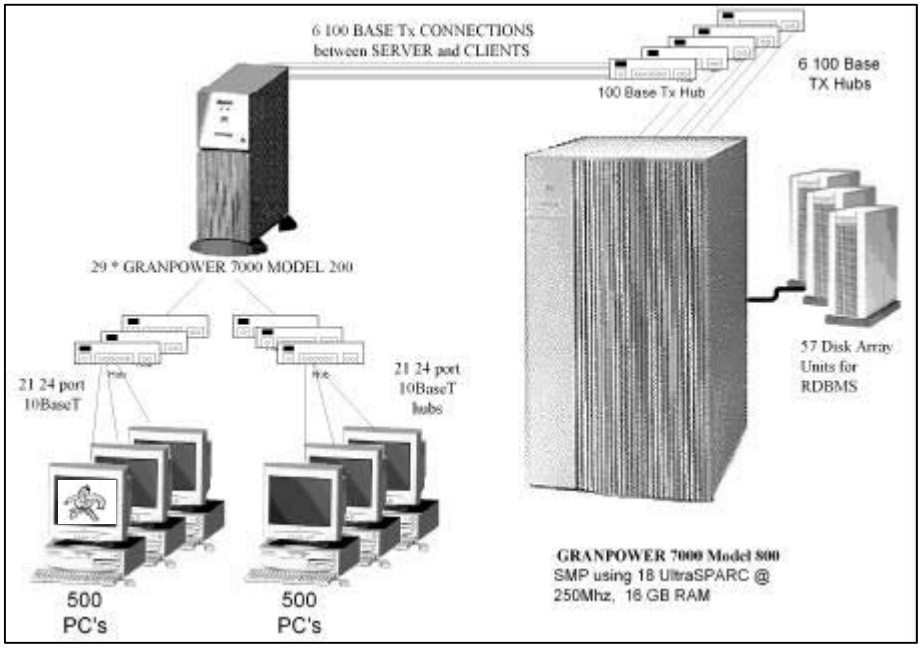
The configuration diagrams for both the tested and priced systems are included on the following pages.

The only difference is the 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.

All tables and logs, excluding STOCK, CUSTOMER, ITEM and HISTORY are striped with LVCF. All database tables are organized on RAID0 and the redo log files on RAID5. Appendix F discloses the organization of tables and indices on the disks.

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 30,000. 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 are described in Appendix H.

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 Internet Explorer 3.0.

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	99.00%
	Remote warehouse order lines	1.00%
	Rolled back transactions	0.10%
	Average items per order	10.00
Payment	Home warehouse	85.06%
	Remote warehouse	14.94%
	Accessed by last name	59.95%
Order Status	Accessed by last name	59.92%
Delivery	Skipped transactions	None
Transaction Mix	New Order	44.68%
	Payment	43.16%
	Order status	4.07%
	Delivery	4.05%
	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 D 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 Log And Data

Write caching is used for both log and data disks. The disk controller cache is mirrored with the dual controller. Each cache is provided with a 72-hour battery back-up. When external power is lost, the UPS will supply power to the disk array and the array will detect the loss of power and force the cache to be flushed to disk.

To demonstrate recovery from a permanent failure of durable medial containing the Oracle recovery log data and TPC-C tables, the following steps were executed on a database scaled to 10 WAREHOUSES including the minimum of two components that were used for the measurements:

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's were started with 100 users.
4. The test was allowed to run for a minimum of five minutes.
5. One of the log disks was powered off by removing it from the cabinet. Since the log was configured on a RAID5 array, the transactions continued to run without interruption.
6. The test was allowed to run for another 5 minutes and the data disks were powered off by stopping the power supply to the diskarray unit.
7. The RTE's were shut down
8. The power supply to the failed diskarray unit was returned and the data disks were reformatted to simulate a complete loss of data.
9. The Oracle were shutdown abort.
10. Data from the backup disks was restored.
11. Oracle was restarted and the media recovery utility started.
12. Step two was repeated and the difference between the first and second counts noted.

13. The success file was used to determine the number of NEW_ORDERS successfully returned to the RTE's.
14. The counts in step twelve and thirteen were compared, verifying that all committed transactions were successfully recovered.
15. Data from the success file was used to query the database to demonstrate that successful transactions had corresponding rows in the ORDER table and that rolled back transactions did not.

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 2,900 warehouses under a full load of 29,000 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 29,000 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.

The TPC-C database was configured with 3,000 warehouses. Before the measurement, warehouses 2,901 to 3,000 were deleted.

Table 4.1 Number of Rows for Server

Table	Occurrences
Warehouse	2,900
District	30,000
Customer	90,000,000
History	90,000,000
Order	90,000,000
New Order	27,000,000
Order Line	899,894,288
Stock	300,000,000
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.*

Oracle8 Enterprise Edition for VLM V80L10is a relational DBMS.

The interface used was Oracle8 Enterprise Edition for VLM V80L10stored 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.
- A full performance run was executed.
- 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..

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 G.

Clause 5 Related Items

5.1 Throughput

Measured tpmC must be reported.

Measured tpmC	34,116.93
Price per tpmC	¥57,883

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	1.53	206.69	2.82
Payment	1.19	208.68	2.46
Order-Status	1.27	112.82	2.58
Interactive Delivery	0.24	95.22	0.32
Deferred Delivery	1.60	7.37	2.54
Stock-Level	1.69	148.51	3.46
Menu	0.30	227.87	0.32

5.3 Keying and Think Times

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

An emulation delay of .1 second is added to response time and menu to compensate for the browser delay.

Table 5.2 Keying Times

Type	Minimum	Average	Maximum
New-Order	18.09	18.11	18.38
Payment	3.04	3.06	3.32
Order-Status	2.04	2.06	2.32
Interactive Delivery	2.04	2.06	2.32
Stock-Level	2.04	2.06	2.32

Table 5.3 Think Times

Type	Minimum	Average	Maximum
New-Order	0.01	12.10	121.18
Payment	0.01	12.12	121.19
Order-Status	0.01	10.31	100.75
Interactive Delivery	0.01	5.10	49.88
Stock-Level	0.01	5.14	49.73

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 for the New-Order transaction.

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

Figure 5.1: New Order Response Time Distribution

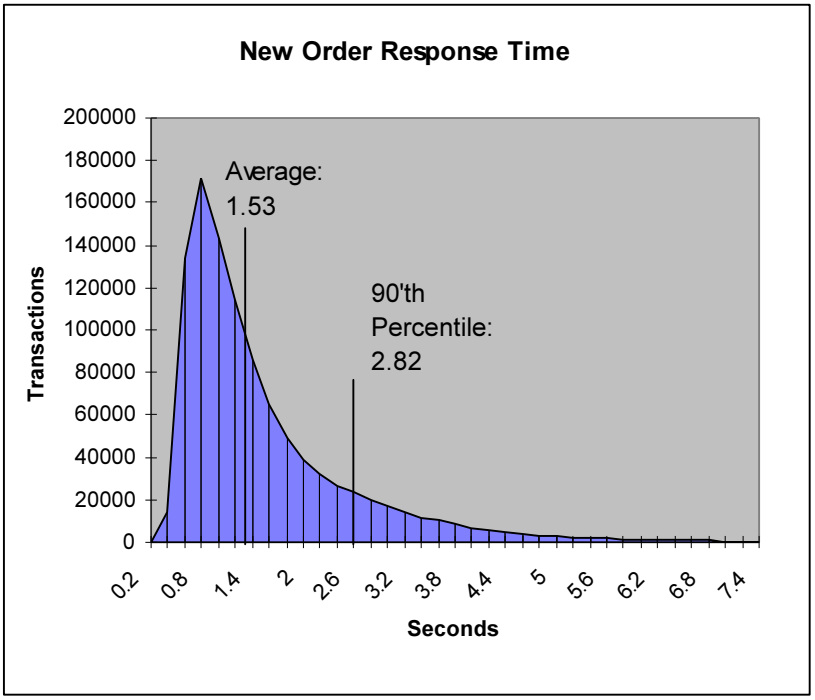


Figure 5.2: Payment Response Time Distribution

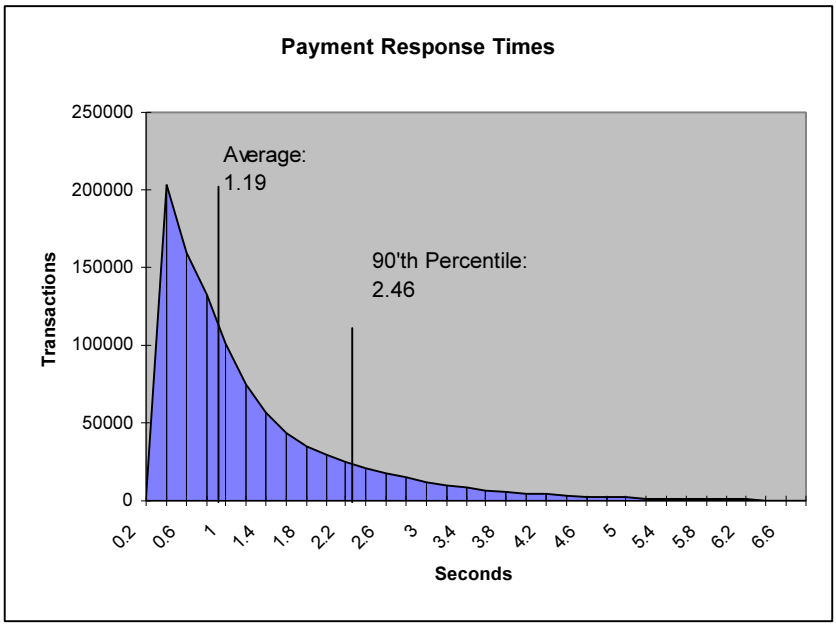


Figure 5.3: Order Status Response Time Distribution

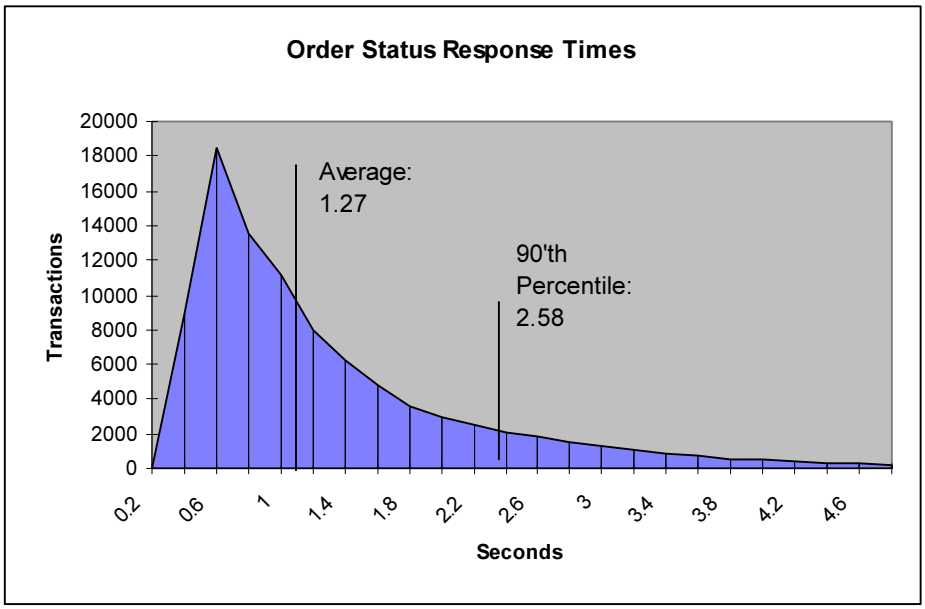


Figure 5.4: Delivery Response Time Distribution

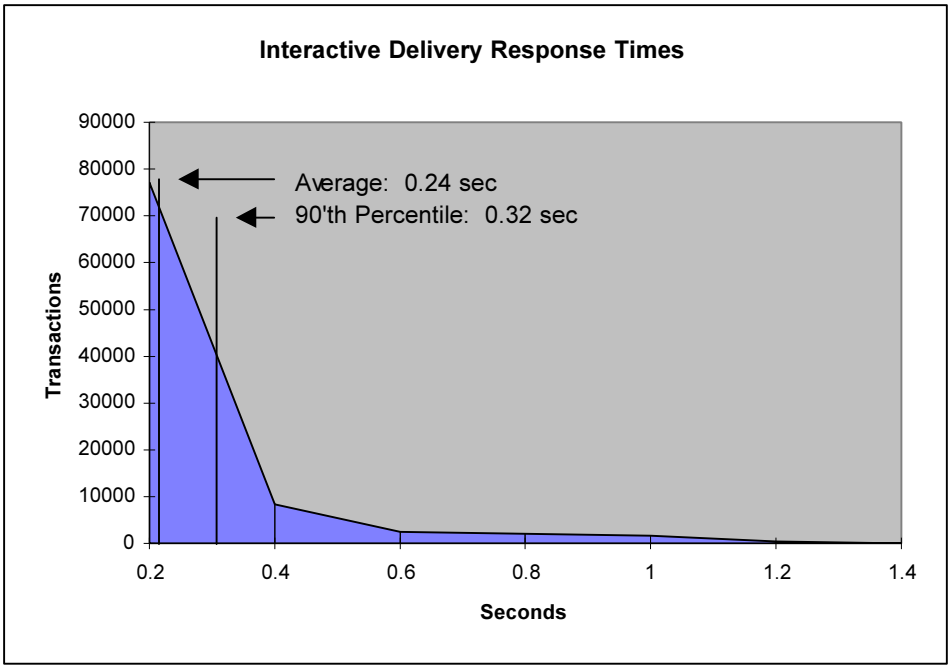


Figure 5.5: Stock Level Response Time Distribution

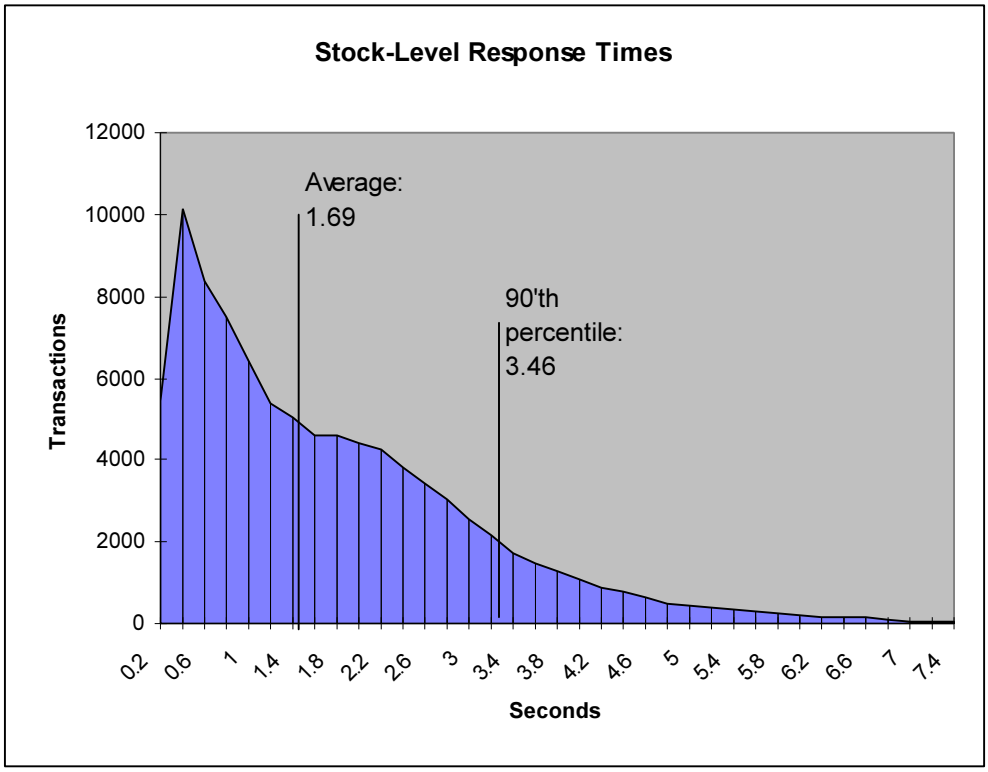


Figure 5.6: New Order Think Time Frequency Distribution

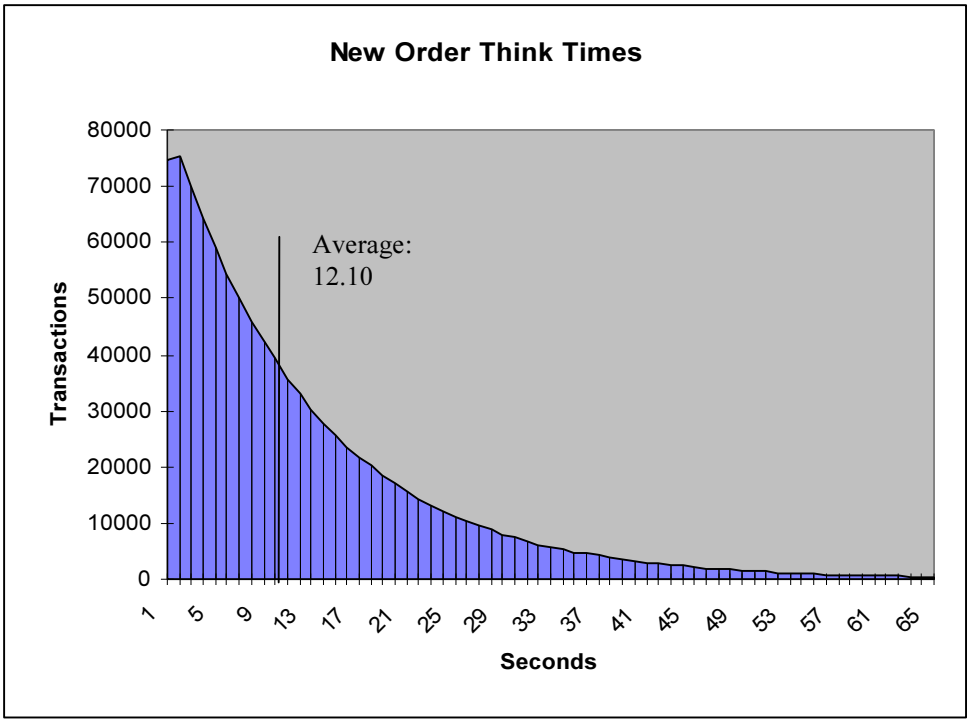


Figure 5.7: Response time versus Throughput

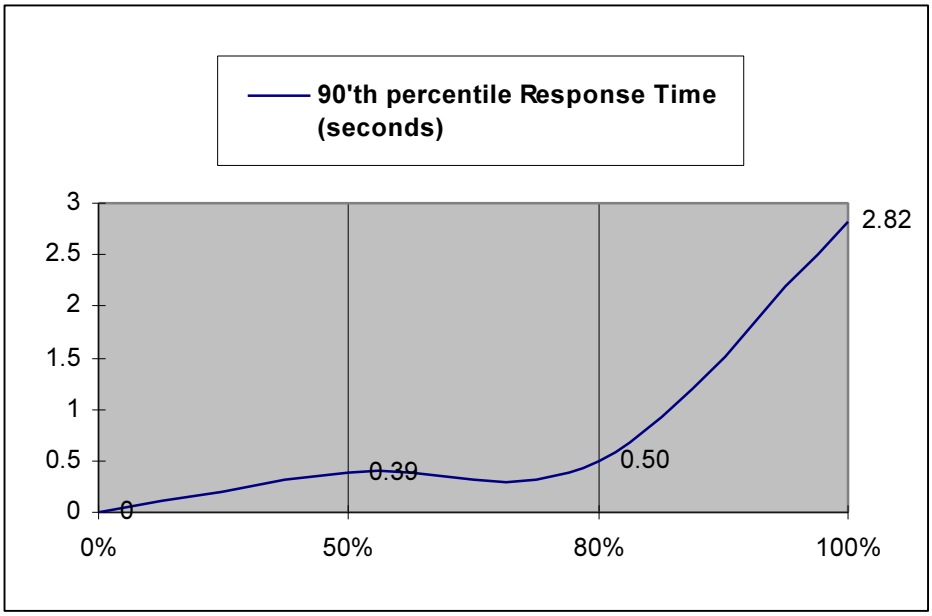
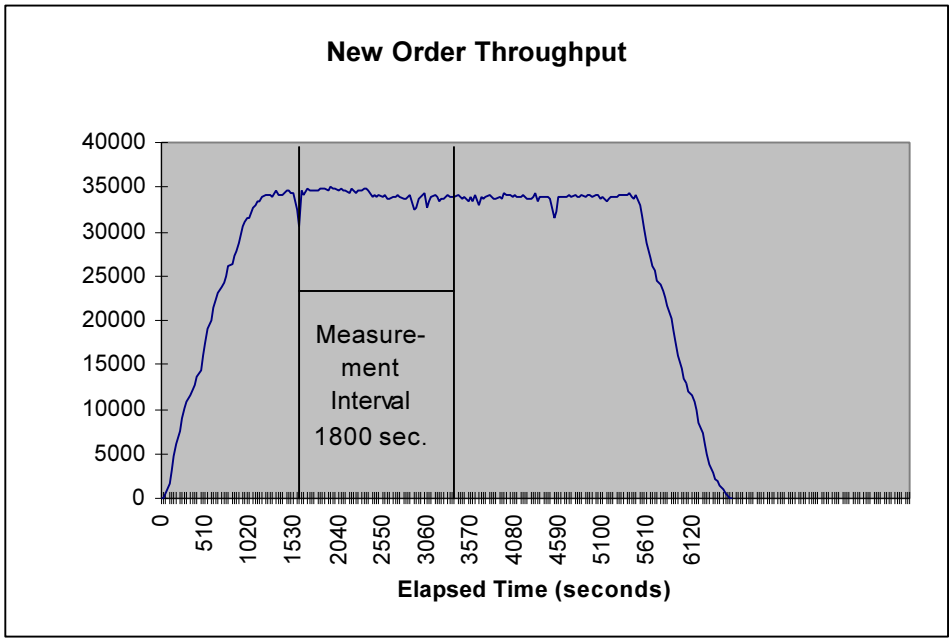


Figure 5.8: New Order Sustained Throughput



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 on a RAID5 array. To perform checkpoints at specific intervals, we set Oracle'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, Oracle 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:

Oracle 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 UNIX function lrand48() 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	44.68%
	Payment	43.16%
	Order status	4.07%
	Delivery	4.05%
	Stock level	4.04%
New Order	Home warehouse order lines	99.00%
	Remote warehouse order lines	1.00%
	Rolled back transactions	0.1%
	Average items per order	10.00
Payment	Home warehouse	85.06%
	Remote warehouse	14.94%
	Accessed by last name	59.95%
Order Status	Accessed by last name	59.92%
Delivery	Skipped transactions	None

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 600 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.

Six 100Mbps ethernet LAN connections were used between each client and the server. Fifty-five 10Mbps ethernet LAN connections were used between the emulated users and the client machines.

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

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 and software componenets will be available no later thanJuly 31st, 1998.

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:	34,116.93
Price per tpmC	¥57,883
Available	July 31st, 1998

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 Richard Gimarc 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

WEBENV

```
# The number of Client machine which executes test
clent_num=30
#
# Number of terminals which executes test
max_user=30000
#
# The maximum number of application programs which
operate with Client machine
ott_num=20
#
# The maximum number of application programs which
operate with one Client machine,
# Number of application programs used with each Client
machine
clent={1,1}
    {1001,1}
    {2001,1}
    {3001,1}
    {4001,1}
    {5001,1}
    {6001,1}
    {7001,1}
    {8001,1}
    {9001,1}
    {10001,1}
    {11001,1}
    {12001,1}
    {13001,1}
    {14001,1}
    {15001,1}
    {16001,1}
    {17001,1}
    {18001,1}
    {19001,1}
    {20001,1}
    {21001,1}
    {22001,1}
    {23001,1}
    {24001,1}
    {25001,1}
    {26001,1}
    {27001,1}
    {28001,1}
    {29001,1}
#
# Number of terminals of which each application
program takes charge
ott={1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
   {1000}
```

MAKEFILE

```
#===================================
#Makefile for building WRB Cartridges
#===================================


OWEB     =    $(ORACLE_HOME)/ows21
INCHOME  =    $(OWEB)/wrbsdk/inc
TOP      =    $(HOME)/orweb
LIBHOME  =    $(TOP)

SRCDIR   =    /oracle/webtc/slim-type3
ORAINC   =    $(ORACLE_HOME)/rdbms/demo
TPCINC   =    $(SRCDIR)/inc
WEBINC   =    $(OWEB)/wrbsdk/inc
TUXINC   =    /opt/uxptuxt/include
SVINC    =    /oracle/server
CLINC    =    /oracle/client
ALLINC   =    -I$(TPCINC) -I${WEBINC} -I${SVINC} -
I${ORAINC} -I${TUXINC}

TUXLIB   =    /opt/uxptuxt/lib
WEBLIB   =    $(OWEB)/wrbsdk/lib
LIBDIR   =    -L$(TUXLIB) -L$(WEBLIB)

LDCOM    =    -xs -L$(LIBHOME)
OKLDCOM  =    -dy $(LIBDIR)

OLIBS    =    -ltux -lbuft -ltux2 -lfml -lfml32 -lgp
SLIBS    =    -lnsl -lm -lsocket -ldl
#SLIBS   =    -lnsl -lm -lsocket -ldl -laio

#DBGF     = -DDBPRT
BESTF     = -O -K 3 -K TMS -K INF
#RFLAG    = -DALLOC_IN_INIT -DNULLCLR
#CFLAG    = -DCONTEX -DALLOC_IN_INIT ${DBGF}
-DCHKCKI -DMAKEID -DABORT -DNOAREA -
DCONTEX
#SFLAG    = -DSCRTEST
CFLAG     = ${BESTF} ${RFLAG} ${DBGF} ${SFLAG}
-DGETENV

RM            = rm

OBJ = tpcc.o anlypara.o str2num.o strct_init.o stocklv.o
p_err.o\
            orderst.o 2str.o date2str.o dec2str.o
payment.o zip2str.o\
            neworder.o deliver.o select_trn.o fastlogin.o
dsppara.o tsp.o\
            gettpccenv.o

#SOBJ = tpcc.o anlypara.o str2num.o strct_init.o
stocklv.o p_err.o dummy.o\
#           orderst.o 2str.o date2str.o dec2str.o
payment.o zip2str.o\
#           neworder.o deliver.o select_trn.o fastlogin.o
dsppara.o tsp.o\
#           gettpccenv.o

all: tpcc3s.so
            $(RM) -f $(OWEB)/tpcc/tpcc3s.so
            cp tpcc3s.so tpcc.so
            cp tpcc3s.so $(OWEB)/tpcc/tpcc3s.so

fast: tpcc3.so
            cp tpcc3s.so tpcc.so
            cp tpcc3s.so  $(OWEB)/tpcc/tpcc3s.so

# Cartridge call back
#back: tpcc3bk.so
#           $(RM) -f $(OWEB)/tpcc/tpcc3bk.so
#           cp tpcc3bk.so $(OWEB)/tpcc/tpcc3bk.so

#fastlogin.o: $(TPCINC)/menupage.h fastlogin1.c
fastlogin2.c
fastlogin.o: $(TPCINC)/menupage.h
            $(CC) -c $<  $(CFLA G) $(ALLINC)

#anlypara.o: anlypara1.c anlypara2.c
#           $(CC) -c $<  $(CFLAG) $(ALLINC)

select_trn.o: $(TPCINC)/tpcinweb.h
            $(CC) -c $<  $(CFLAG) $(ALLINC)

deliver.o: $(TPCINC)/delpage.h
            $(CC) -c $<  $(CFLAG) $(ALLINC)

stocklv.o: $(TPCINC)/stpage.h
            $(CC) -c $<  $(CFLAG) $(ALLINC)

orderst.o: $(TPCINC)/odrpage.h
            $(CC) -c $<  $(CFLAG) $(ALLINC)

payment.o: $(TPCINC)/paypage.h
            $(CC) -c $<  $(CFLAG) $(ALLINC)

neworder.o: $(TPCINC)/newpage.h
            $(CC) -c $<  $(CFLAG) $(ALLINC)

p_err.o: $(TPCINC)/tpcweb.h
            $(CC) -c $<  $(CFLAG) $(ALLINC)

.c.o :
            $(CC) -c $<  $(CFLAG) $(ALLINC)
```

```
tpcc3s.so: $(OBJ)
        $(CC) -G -o $@ $(OBJ) $(OKLDCOM)
$(OLIBS) $(SLIBS)

#tpcc3bk.so: $(SOBJ)
#       $(CC) -G -o $@ $(SOBJ) $(OKLDCOM)
$(OLIBS) $(SLIBS)

clean:
        $(RM) -f tpcc3.so tpcc3bk.so t pcc.so
        $(RM) -f $(SOBJ)
```

Header Files

```
/*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_ */


/* ----------------------------------------------------------------
-
  delpage.h
  data of delivery transaction result screen (HTML
format)
  ----------------------------------------------------------------
- */

/* Header data */
#define h_del1 "\
Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-WINDOW
</TITLE></HEAD><BODY>\
<CENTER><H3>Delivery<br></H3></CENTER>\
<PRE>"

/* Screen data*/
#define h_del2 "\
Warehouse:    \n\
\n\
Carrier Number:   \n\
\n\
Execution Status:                    \n\
\n\n\n\n\n\n\n\n\n\n\n\n\
"
/* Tailer data */
#define h_del3 "\
</PRE><FORM ACTION=\"%s\" METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"New
order\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Payment\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Delivery\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Order
Status\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Stock
Level\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Quit\">\
</FORM></BODY></HTML>\n"

/* Offset to field which should set data */
int  delp[] = { 0xb, 0x21, 0x37 }; /* w_id, carrier number,
status */

/*========================================
================================+
  FILENAME: ex_tpcc_info.h
  This file is used for the function which refers to
the external variable

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

extern  long    olen;
extern  void    *trans_buf;
extern  int     trans_size;
extern  int     svrnum;

/*
extern  int     logincnt;
extern  int     base_cok;
extern  char    NewOrdername[];
extern  char    Paymentname[];
extern  char    OrderStatusname[];
extern  char    Deliveryname[];
extern  char    StockLevelname[];
*/

extern  int     now_cookie;
extern  int     now_w_id;
extern  int     now_d_id;
extern  char    now_ottname[];
extern  char    SOPATH[];

extern  char    s_work[];

#ifdef DBPRT
extern  FILE    *test_fp;
#endif

#include "trans.h"

/* ----------------------------------------------------------------
-
  menupage.h
  ----------------------------------------------------------------
- */

/* Transaction select screen data */
#define h_menu "\
Content-type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-
WINDOW</TITLE></HEAD>\n\
<BODY><CENTER><H3>Transaction
Menu<BR></H3></CENTER>\n\
<PRE>\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n</PRE>\
<FORM ACTION=\"%s\" METHOD=\"GET\">\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"New
order\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Payment\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Delivery\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Order
Status\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Stock
Level\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Quit\">\
</FORM></BODY></HTML>\n"

/* If client can not connect server then use this format */
#define noconnt "\
Content-type: text/html\n\n\
<HTML><HEAD><TITLE>ERROR: Can't
Connect</TITLE></HEAD><BODY>\n\
<P>Sorry. all %d database connections are currently in
use. \
Please try again later. terminal number is %d.
</P></BODY></HTML>\n"

/* If connect client by WWW browser then use this
format */
/* Terminal verification use olny */
#define loginpage "\
Content-type: text/html\n\n\
<HTML><HEAD><TITLE>Welcome to TPC-
C</TITLE></HEAD><BODY>\n\
```

```
<P>Please identify your terminal number for this
session.</P>\n\
<FORM ACTION=\"%s\" METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"f\" VALUE=\"1\">\n\
Your Terminal Number:<INPUT NAME=\"c\"
SIZE=5><BR><HR>\n\
<INPUT TYPE=\"submit\">\n\
</FORM></BODY></HTML>\n"

/* LOGIN ERROR PAGE */
/* If terminal number not find then use this format */
#define loginerr2 "\
Content-type: text/html\n\n\
<HTML><HEAD><TITLE>ERROR :Welcome to TPC-
C</TITLE></HEAD><BODY>\n\
<P>Please identify your terminal number for this
session.</P>\n\
<FORM ACTION=\"%s\" METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"f\" VALUE=\"1\">\n\
Your Terminal Number:<INPUT NAME=\"c\"
SIZE=5><BR><HR>\n\
<INPUT TYPE=\"submit\">\n\
</FORM></BODY></HTML>\n"

/*
#define endpage "\
Content-type: text/html\n\n\
<HTML><HEAD><TITLE>Welcome to TPC-
C</TITLE></HEAD><BODY>\n\
<P>Please identify your Warehouse and District for this
session.</P>\n\
<FORM ACTION=\"%s\" METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<INPUT TYPE=\"hidden\" NAME=\"f\" VALUE=\"1\">\n\
Your Warehouse ID: <INPUT NAME=\"W\"
SIZE=4><BR>\n\
Your District ID:  <INPUT NAME=\"d\"
SIZE=2><BR><HR>\n\
<INPUT TYPE=\"submit\">\n\
</FORM></BODY></HTML>\n"
*/

/* ----------------------------------------------------------------
-
  newpage.h
  data of neworder transaction result screen (HTML
form)
  ----------------------------------------------------------------
- */

/* Header data */
#define h_new1 "\
Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-
WINDOW</TITLE></HEAD><BODY>\
<CENTER><H3>New-Order<BR></H3></CENTER>\
<font size=3>\n<PRE>"

/* Screen data*/
#define h_new2 "\
Warehouse:       District:                Date:   -  -
:  :  \n\
Customer:       Name:                Credit:     %%Disc:
.  \n\
Order Number:          Number of Lines:
W_tax:   .     D-tax:   .  \n\n\
Supp-W   Item-Id  Item Name                          Qty
Stock  B/G  Price    Amount\n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
                                                $   .   $    . \n\
Execution Status:                       Total:  $
.  \n"

/* Tailer data */
#define h_new3 "\
</PRE>\
<FORM ACTION=\"%s\" METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"New
order\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Payment\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Delivery\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Order
Status\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Stock
Level\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Quit\">\
</FORM></BODY></HTML>"

/* Offset to field which should set data */
int newp[] = {
0xb, 0x1c, 0x3c,
0x5b, 0x68, 0x83, 0x8f,
0xa3, 0xbe, 0xcb, 0xda,
0x127, 0x12f, 0x138, 0x152, 0x158, 0x15f, 0x164,
0x16d,
0x5d9, 0x60d
};


/* ----------------------------------------------------------------
-
  odrpage.h
  data of OrderStatus transaction result screen
(HTML form)
  ----------------------------------------------------------------
- */

/* Header data */
#define h_order1 "\
Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-
WINDOW</TITLE></HEAD>\
<BODY><CENTER><H3>Order-
Status<BR></H3></CENTER>\n\
<font size=3>\n<PRE>"

/* Screen data*/
#define h_order2 "\
Warehouse:       District:   \n\
Customer:       Name:                          \n\
Cust-Balance: $      .  \n\n\
Order-Number:          Entry-Date:  -  -      :  :
Carrier-Number:  \n\
Supply-W    Item-Id    Qty     Amount      Delivery-
Date\n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
                                $    .        -  -   \n\
"

/* Tailer data */
#define h_order3 \
"</PRE>\
<FORM ACTION=\"%s\" METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"New
order\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Payment\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Delivery\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Order
Status\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Stock
Level\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Quit\">\
</FORM></BODY></HTML>"

/* Offset to field which should set data */
int orderp[] = {
0x0B, 0x1C,
0x29, 0x36, 0x47, 0x4a,
0x6a,
0x83,  0x9a,  0xc0,
0xFF, 0x10a,  0x115, 0x11d, 0x12b,   /* offset 0x39 */
0x145, 0x150, 0x15b, 0x163, 0x171};

/* ----------------------------------------------------------------
-
  paypage.h
  data of Payment transaction result screen (HTML
form)
  ----------------------------------------------------------------
- */

/* Header data */
#define h_pay1 "\
Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-
WINDOW</TITLE></HEAD><BODY>\
<CENTER><H3>Payment<BR></H3></CENTER><font
size=3>\n<PRE>"

/* Screen data*/
```

```
#define h_pay2 "\
Date:  - -     : :  \n\
\n\
Warehouse:                    District:  \n\
                                  \n\
                                  \n\
                -                          -  \n\
\n\
Customer:     Cust-Warehouse:     Cust-District:  \n\
Name:                          Since:  - -  \n\
                           Credit:  \n\
                           %%Disc:   .  \n\
                -          Phone:      - - -
\n\
\n\
Amount Paid:       $   .      New Cust-Balance: $
.  \n\
Credit Linit:  $       .  \n\
\n\
Cust-Data:                          \n\
                                  \n\
                                  \n\
                                  \n"

/* Tailer data */
#define h_pay3 "\
</PRE>\
<FORM ACTION=\"%s\" METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"New
order\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Payment\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Delivery\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Order
Status\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Stock
Level\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Quit\">\
</fORM></BODY></HTML>"

/* Offset to field which should set data */
int payp[] = {
0x06,
0x27, 0x4f,
0x52, 0x7b,
0x90, 0xb9,
0xce, 0xe3, 0xe6, 0xec, 0xf7, 0x10c, 0x10f, 0x115,
0x125, 0x13b, 0x150,
0x15b, 0x16c, 0x16f, 0x18c,          /* 18 - 21 */
0x19f, 0x1d0,                                /* 22,
23 */
0x1db, 0x20c,                                /* 24,
25 */
0x21a, 0x22f, 0x232, 0x24b,
0x277, 0x297,
0x2b7,

0x2d1,                                       /*
offset 0x3e */
0x30f,
0x34d,
0x38b};

/* ----------------------------------------------------------------
-
```

```
  stopage.h
  data of Stock Level transaction result screen
(HTML form)
  ----------------------------------------------------------------
- */

/* Header data */
#define h_stock1 "\
Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-
WINDOW</TITLE></HEAD><BODY>\
<CENTER><H3>Stock-Level<BR></H3></CENTER>\
<font size=3>\n<PRE>"

/* Screen data*/
#define h_stock2 "\
Warehouse:       District:  \n\
\n\
Stock Level Threshold:  \n\
\n\
low stock:   \n\
\n\n\n\n\n\n\n\n\n\n\n\n"

/* Tailer data */
#define h_stock3 "\
</PRE>\
<FORM ACTION=\"%s\" METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"New
order\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Payment\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Delivery\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Order
Status\">\
<INPUT TYPE=\"submit\" NAME=\"b\" VALUE=\"Stock
Level\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Quit\">\
</FORM></BODY></HTML>\n"

/* Offset to field which should set data */
int stockp[] = {
0x0B, 0x1C,
0x37,
0x46};


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

  tpcc.h

  this file use all function of the TPC-C client
application
  ----------------------------------------------------------------
*/
#ifndef ORATYPES_ORACLE
# include <oratypes.h>
#endif

#ifndef WRB_ORACLE
# include <wrb.h>
#endif

#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>
#include <sys/stat.h>
#include <fcntl.h>
#include <malloc.h>

#include <ctype.h>
/*
#include <ocidfn.h>

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

/* TP-Base(Tuxedo) include */
#include "atmi.h"

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 ();

#define  MAXWH     3000                      /* Max
WareHouse scale */

/* The maximum value of client matchine which can be
processed */
#define MAXCLIENT  32

/* The maximum value of TP application program of 1
client matchine
  which can be processed */
#define MAXOTT     80

/* number of Transaction */
#define TRANNEW 1
#define TRANPAY 2
#define TRANORD 3
#define TRANDEL 4
#define TRANSTO 5

/* Error codes : send from TP application program */
```

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

#define NG  0
#define OK  1

#define BUF_S           4096        /* size of the send
baffer area */
#define WORK_S          2400        /* size of the work
buffer area */

/* Debug Print proc define : debug use only */
#ifdef DBPRT

 #if ( DBPRT > 5 )
#define DBGP(proc)      proc
#define DBGR(proc)      proc

 #else
#define DBGP(proc)
#define DBGR(proc)      proc
 #endif

#else
#define DBGP(proc)
#define DBGR(proc)
#endif


/*=========================================
========================+
  FILENAME : tpcc_info.h
  DESCRIPTION
  The external variable is declared. (this file use
tpcc.c only)

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

#ifndef TPCC_INFO_H
#define TPCC_INFO_H

long            olen;                           /*
buffer size : TP application use */
void    *trans_buf;             /* pointer of
Interface area */
int             trans_size = 1024;      /* interfase area
size */
int     svrnum;                         /* TP
application program number */

/*
int     logincnt = 0;
int     base_cok = 0;
char    NewOrdername[20];
char    Paymentname[20];
char    OrderStatusname[20];
char    Deliveryname[20];
char    StockLevelname[20];
*/

char    s_buf[BUF_S];           /* send buffer area
*/
char    s_work[WORK_S];                 /*
working buffer area */
int             now_cookie = 0;                 /*
terminal number */
int             now_w_id = 0;                   /*
w_id number */
int             now_d_id = 0;                   /* d_id
number */
char            now_ottname[8] =
{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}; /* save
TP applicaton program name */
char            SOPATH[64];                     /*
save URL name */

#ifdef DBPRT                                    /* for
debug */
FILE   *test_fp;
#endif

#include "trans.h"

#endif


/* ---------------------------------------------------------------------
  tpcinweb.h
  Transaction input data screen data
  ---------------------------------------------------------------------
*/

/* -------------------------
   delivery page
 * -------------------------*/
char in_delpage[] =
"Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-C:
Delivery</TITLE></HEAD>\n\
<BODY><FORM ACTION=\"%s\"
METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"f\" VALUE=\"D\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<center><H3>Delivery<br></H3></center>\n\
<font size=3><PRE>\
Warehouse:%4d\n\
\n\
Carrier Number:<INPUT NAME=\"OC\" SIZE=2
maxlength=2>\n\
\n\
Execution Status:\n\
</PRE><INPUT
TYPE=\"submit\"></FORM></BODY></HTML>\n";

/* -------------------------
   neworder page
 * -------------------------*/
char  in_newpage[] =
"Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-C: New
Order</TITLE></HEAD>\n\
<BODY><FORM ACTION=\"%s\"
METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"f\" VALUE=\"N\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<center><H3>New Order<br></H3></center>\n\
<PRE><font size=3>\
Warehouse:%4d       District: <INPUT NAME=\"D\"
SIZE=2 maxlength=2>                 Date:\n\
Customer:  <INPUT NAME=\"CI\" SIZE=4
maxlength=4>  Name:                 Credit:
%%Disc:\n\
Order Number:       Number of Lines:        W_tax:
D_tax:\n\
\n\
 Supp_W  Item_Id  Item Name             Qty  Stock
B/G  Price    Amount\n\
 <INPUT NAME=\"OS01\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI01\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ01\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS02\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI02\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ02\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS03\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI03\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ03\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS04\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI04\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ04\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS05\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI05\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ05\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS06\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI06\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ06\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS07\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI07\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ07\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS08\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI08\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ08\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS09\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI09\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ09\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS10\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI10\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ10\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS11\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI11\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ11\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS12\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI12\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ12\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS13\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI13\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ13\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS14\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI14\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ14\" SIZE=2 maxlength=2>\n\
 <INPUT NAME=\"OS15\" SIZE=4 maxlength=4>
<INPUT NAME=\"OI15\" SIZE=6 maxlength=6>
<INPUT NAME=\"OQ15\" SIZE=2 maxlength=2>\n\
Execution Status:                       Total:
$\n\
</PRE><INPUT
TYPE=\"submit\"></FORM></BODY></HTML>\n";

/* -------------------------
   orderstatus page
 * -------------------------*/
char in_odrpage[] =
"Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-C: Order-
Status</TITLE></HEAD>\n\
<BODY><FORM ACTION=\"%s\"
METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"f\" VALUE=\"O\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<center><H3>Order Status<br></H3></center>\n\
<font size=3><PRE>\
```

```
Warehouse:%4d       District:<INPUT NAME=\"D\"
SIZE=2 maxlength=2>\n\
Customer:<INPUT NAME=\"CI\" SIZE=4 maxlength=4>
Name:                <INPUT NAME=\"CL\" SIZE=16
maxlength=16>\n\
Cust-Balance: \n\
\n\
Order-Number:          Entry-Date:
Carrier-Number:\n\
Supply-W    Item-Id   Qty    Amount     Delivery-
Date\n\
\n\
\n\
\n\
\n\
\n\
\n\
\n\
\n\
\n\
\n\
\n\
\n\
\n\
\n\
</PRE>\n\
<INPUT
TYPE=\"submit\"></FORM></BODY></HTML>\n";

/* ---------------------------
   payment page
 * ---------------------------*/
char in_paypage[] =
"Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-C:
Payment</TITLE></HEAD>\n\
<BODY><FORM ACTION=\"%s\"
METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"f\" VALUE=\"P\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<center><H3>Payment<br></H3></center>\n\
<font size=3><PRE>\
Date:\n\
\n\
Warehouse:%4d                    District:<INPUT
NAME=\"D\" SIZE=2 maxlength=2>\n\
\n\
\n\
\n\
\n\
Customer:<INPUT NAME=\"CI\" SIZE=4 maxlength=4>
Cust-Warehouse:<INPUT NAME=\"CW\" SIZE=4
maxlength=4>  Cust-District:<INPUT NAME=\"CD\"
SIZE=2 maxlength=2>\n\
Name:                <INPUT NAME=\"CL\" SIZE=16
maxlength=16>     Since:\n\
                                 Credit:\n\
                                 %Disc:\n\
                                 Phone:\n\
\n\
Amount Paid:       $<INPUT NAME=\"H\" SIZE=7
maxlength=7>     New Cust-Balance:\n\
Credit Limit:\n\
\n\
Cust-Data:\n\
\n\
\n\
</PRE>\n\
<INPUT
TYPE=\"submit\"></FORM></BODY></HTML>\n";

/* ---------------------------
   stock level page
 * ---------------------------*/
char in_stkpage[] =
"Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>TPC-C: Stock-
Level</TITLE></HEAD>\n\
<BODY><FORM ACTION=\"%s\"
METHOD=\"GET\">\n\
<INPUT TYPE=\"hidden\" NAME=\"f\" VALUE=\"S\">\n\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\n\
<center><H3>Stock Level<br></H3></center>\n\
<font size=3><PRE>\
Warehouse:%4d       District:%2d\n\
\n\
Stock Level Threshold:<INPUT NAME=\"t\" SIZE=2
maxlength=2>\n\
\n\
low stock:\n\
</PRE>\n\
<INPUT
TYPE=\"submit\"></FORM></BODY></HTML>\n";

/* ---------------------------------------------------------------------
   tpcweb.h
   ---------------------------------------------------------------------
--*/

/* If transaction input data is abnormal then use this
format. */
#define errhtml  "\
Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>ERROR: TPC-
C</TITLE></HEAD><BODY>\
<p>You did something bad.  The error message
was:</p>\
<PRE>%s</PRE>\
<p>Either hit the \"back\" button on your browser and fix
the problem, \
or hit the \"Quit\" button below to terminate this session.
</P><HR>\
<P><FORM ACTION=\"%s\" METHOD=\"GET\">\
<INPUT TYPE=\"hidden\" NAME=\"c\" VALUE=%d>\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Quit\">\
</FORM></P></BODY></HTML>\r\n"

/* If TP application terminated abnormally then use this
format. */
#define tuxerr "\
Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>ERROR: Tuxedo
</TITLE></HEAD><BODY>\
<P>The database could not process your request.  \
tpcall terminated abnormally.</P>\
<HR><PRE>%s</PRE><HR>\
<FORM ACTION=\"%s\" METHOD=\"GET\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Quit\">\
</BODY></HTML>"

/* If Oracle application terminated abnormally then use
this format. */
#define oraerr "\
Content-Type: text/html\n\n\
<HTML><HEAD><TITLE>ERROR: ORACLE
</TITLE></HEAD><BODY>\
<P>The database could not process your request.  \
Transaction terminated abnormally.</P>\
<HR><PRE>%s</PRE><HR>\
<FORM ACTION=\"%s\" METHOD=\"GET\">\
<INPUT TYPE=\"submit\" NAME=\"b\"
VALUE=\"Quit\">\
</BODY></HTML>"


/*=========================================
=========================+
  FILENAME : trans.h
  the work struct acoording to transaction is
declared.


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

/* New order struct */
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 {
   int tran_kind;
   struct newinstruct newin;
   struct newoutstruct newout;
};


/* Payment struct */
struct payinstruct {
   int w_id;
   int d_id;
   int c_w_id;
   int c_d_id;
   int c_id;
   int bylastname;
/* float h_amount; */ /* old-tool.kit */
```

```
   int h_amount;
   char c_last[17];
};

struct payoutstruct {
   int terror;
   char w_street_1[21];
   char w_street_2[21];
   char w_city[21];
   char w_state[3];
   char w_zip[10];
   char d_street_1[21];
   char d_street_2[21];
   char d_city[21];
   char d_state[3];
   char d_zip[10];
   int  c_id;
   char c_first[17];
   char c_middle[3];
   char c_last[17];
   char c_street_1[21];
   char c_street_2[21];
   char c_city[21];
   char c_state[3];
   char c_zip[10];
   char c_phone[17];
   char c_since[11];
   char c_credit[3];
   double c_credit_lim;
   float c_discount;
   double c_balance;
   char c_data[201];
   char h_date[20];
   int retry;
};

struct paystruct {
   int tran_kind;
   struct payinstruct payin;
   struct payoutstruct payout;
};

/* Order status struct */
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 {
   int tran_kind;
   struct ordinstruct ordin;
   struct ordoutstruct ordout;
};


/* Delivery struct */
struct delinstruct {
   int w_id;
   int o_carrier_id;
   long qtime;
   long uqtime;
   int in_timing_int;
};

struct deloutstruct {
   int terror;
   int retry;
};

struct delstruct {
   int tran_kind;
   struct delinstruct delin;
   struct deloutstruct delout;
};


/* Stock level struct */
struct stoinstruct {
   int w_id;
   int d_id;
   int threshold;
};

struct stooutstruct {
   int terror;
   int low_stock;
   int retry;
};

struct stostruct {
   int tran_kind;
   struct stoinstruct stoin;
   struct stooutstruct stoout;
};

/* Client context area (oracle web server) */

typedef struct {
   void *trans_b;            /* pointer of interface area
with TP application */
   int  prt_cnt;                        /* print counter :
for debug */
   int  clent_num;                      /* maximam client
matchine number */
   int  max_user;                       /* maximam user
number */
   int  ott_num;                        /* maximam TP
applicaton program of 1 client */
   int  clent[MAXCLIENT][2];            /* client matchine
infomation */
   int  ott  [MAXCLIENT][MAXOTT];       /* TP application
program information*/
}tpc_struct;

/* RTE - Client interface struct */
typedef struct {
   char *button,
   *cookie,
   *form,
   *O_CARRIER_ID,
   *threshold,
   *D_ID,
   *C_ID,
   *C_W_ID,
   *C_D_ID,
   *C_LAST,
   *H_AMOUNT,
   *OL_SUPPLY_W_ID[15],
   *OL_I_ID[15],
   *OL_QUANTITY[15];
} rte_input_data;
```

C Source

alp2str.c

```
#include "tpcc.h"

#include "ex_tpcc_info.h"



/*

  alp2str : Outputs a string into the memory space
supplied.



  field = the destination field

  field_size = number of characters to output
  string = alpha string to be displayed
*/

void alp2str(char *str, int len, char *alp)
{
   int cnt;

   cnt = strlen (alp);
   strncpy (str, alp, len);    /* copy to destination area */

   /* lenÊ¬¤ËËþ¤¿¤Ê¤¤¾¡¹çÏ¶õÇò¤òµÍ¤á¤ë*/
   /* If not coming up to the specified length then set
the blank. */
   if ( len - cnt > 0 )
                memset ( &str[cnt], ' ', len - cnt);
}

/*
  int2str : Converts an integer value to a string of a
specified length and
  outputs the string to the memory buffer supplied.

  field = the destination field
  field_size = number of characters to output
  value = integer to be displayed
*/

void int2str(char *str, int len, int num)
{
   int cnt;
```

```
    for (cnt = len - 1; cnt >= 0 ; cnt--){

                str[cnt] = (num % 10) + '0';
                num /= 10;
    }

    for (cnt = 0; cnt < len-1; cnt++){

                if (str[cnt] == '0')
                   str[cnt] = ' ';
                else
                  return;
    }
}

/*
  int3str : Converts an integer value to a string of a
specified length and
  outputs the string to the memory buffer supplied.

  field = the destination field
  field_size = number of characters to output
  value = integer to be displayed
*/

void int3str(char *str, int len, int num)
{
    int cnt;

    for (cnt = len - 1; cnt >= 0 ; cnt--){

                str[cnt] = (num % 10) + '0';
                num /= 10;
    }
}
```

anlypara.c

```
#include "tpcc.h"
#include "ex_tpcc_info.h"
#include <stdlib.h>

/*
  The input data received from RTE is analyzed

  WRBGetURL ->
    Returns the URL received from the HTTP Server.
The cartridge developers
    may extract additional information from the URL if
required.

  WRB Get Content ->
    Returns either the Query String or the POST data
content depending on the
    type of the request method. Returns NULL in case
on an error.

*/

int  dataparse ( void *WRBCtx, rte_input_data *in_data
)
{
    char  *quer;

    quer = WRBGetURL(WRBCtx);
    sprintf (SOPATH, "%s", quer);
    DBGR(fprintf (test_fp, "ptn1: SOPATH=%s\n",quer));

    quer = WRBGetContent(WRBCtx);

    DBGR(fprintf (test_fp, "ptn1: quer=%s\n",quer));

    return (anly_para (quer, in_data ));

}

/*
  para_split :
  Split divides up a string based on the first instance of
a specified
  delimiter('&' or '='). The first instance of '&'or '=' is
converted to
  a NULL and the address of the first character of the
second half is
  returned.
*/

char *para_split(char *para, char delimita) {
    int i;
    char *point;

    /* The address of the delimitation character is
calculated */
    /* ¶èÀÚ¤êÊ¸»ú¤ÎÎ¥¢¥É¥¡¥¹¤òµá¤á¤ë */
    if ((point = strchr (para, delimita)) == NULL)
        return (char *)0;

    /* The delimitation character is replaced with NULL*/
    *point = '\0';                              /*
¶èÀÚ¤êÊ¸»ú¤òNULL¤ËÊÑ´¹¤¹¤ë */

    /* The first position of the analyzed variable is
returned.*/
    return (point + 1);         /*
¥Ç¡¼¥¿¤ÎÀèÆ¬¥¢¥É¥¡¥¹¤òÊÖ¤¹ */
}

/*
  anly_para :
  Gets the query string and finds every variable=value
pair contained
  within it.  For every pair, it runs the variable name
through a really
  big compound switch statement that matches for
specific variables we
  want to catch.  When we find a known variable name,
we stick a pointer
  to its corresponding value into the appropriate
member of 'in_data.'

  para    - the query string.
  in_data - a raw_form_data structure to hold pointers.
*/

int anly_para (char *para, rte_input_data *in_data) {
    char *val, *rest;
    int  num = 0;

    if(!para) return 0;

    while(para) {
                rest = para_split(para, '&'); /* next parameta
point */
                val  = para_split(para, '='); /* now value
point */

                switch(para[0]) {
                case 'c':
                   in_data->cookie = val;
break;

                case 'b':
                   in_data->button = val;     break;

                case 'f':
                   in_data->form = val;       break;

                case 't':
                   in_data->threshold = val;
break;

                case 'D':
                   in_data->D_ID = val;       break;

                case 'H':
                   in_data->H_AMOUNT = val;
break;

                case 'C':
                   switch(para[1]) {
                   case 'I':
                                in_data->C_ID = val;      break;

                   case 'W':
                                in_data->C_W_ID = val;  break;

                   case 'L':
                                in_data->C_LAST = val;  break;

                   case 'D':
                                in_data->C_D_ID = val;  break;
                   }
                   break;

                case 'O':
                   switch(para[1]) {
                   case 'C':
                                in_data->O_CARRIER_ID =
val;                            break;

                   case 'S':
                                switch(para[2]) {
                                case '0':
                                   if (para[3] >= 0x31 &&
para[3] <= 0x39 ){
                                                num = (int)(para[3]
- 0x30);
                                                if (strlen(val) != 0 )
                                                   in_data-
>OL_SUPPLY_W_ID[num - 1] = val;
                                   }
                                   break;

                                case '1':
                                   if (para[3] >= 0x30 &&
para[3] <= 0x35 ){
                                                num = (int)(para[3]
- 0x30) + 10;
                                                if (strlen(val) != 0 )
                                                   in_data-
>OL_SUPPLY_W_ID[num - 1] = val;
                                   }
                                   break;
                                }
                   break;
```

```
                case 'l':
                        switch(para[2]) {
                        case '0':
                          if (para[3] >= 0x 31 &&
para[3] <= 0x39 ){
                                num = (int)(para[3]
- 0x30);
                                if (strlen(val) != 0 )
                                  in_data-
>OL_I_ID[num - 1] = val;
                          }
                          break;

                        case '1':
                          if (para[3] >= 0x30 &&
para[3] <= 0x35 ){
                                num = (int)(para[3]
- 0x30) + 10;
                                if (strlen(val) != 0 )
                                  in_data-
>OL_I_ID[num - 1] = val;
                          }
                          break;
                        }
                        break;

                case 'Q':
                        switch(para[2]) {
                        case '0':
                          if (para[3] >= 0x31 &&
para[3] <= 0x39 ){
                                num = (int)(para[3]
- 0x30);
                                if (strlen(val) != 0 )
                                  in_data-
>OL_QUANTITY[num - 1] = val;
                          }
                          break;

                        case '1':
                          if (para[3] >= 0x30 &&
para[3] <= 0x35 ){
                                num = (int)(para[3]
- 0x30) + 10;
                                if (strlen(val) != 0 )
                                  in_data-
>OL_QUANTITY[num - 1] = val;
                          }
                          break;
                        }
                        break;
                }
                break;
              }

              para = rest;
    }

    if (in_data->cookie != 0)
        return(atoi (in_data->cookie));
    else
        return(0);
}
```

date2str.c

```
#include "tpcc.h"
#include "ex_tpcc_info.h"
#include <sys/time.h>
/*
   date2str : Outputs a date in the supplied buffer in the
following format:
   DD-MM-YYYY

   field = the destination field
   date = date to be converted and displayed
*/
void date2str(char *str, char *time)
{
    int  year, month, day, hour, min, sec;

#ifdef DBPRT
    fprintf (test_fp, "date2: %s\n", time);
#endif
    sscanf( time, "%d-%d-%d", &day, &month, &year );

    int3str (str, 2, day);
    str[2] = '-';
    int3str (&str[3], 2, month);
    str[5] = '-';
    int3str (&str[6], 4, year);
}


/*
   time2str:
   Outputs a date and time in the supplied buffer in the
following format:
   DD-MM-YYYY hh:mm:ss

   field = the destination field
   date = date and time to be converted and displayed
*/
void time2str (char *str, char *time)
{
    int  year, month, day, hour, min, sec;

#ifdef DBPRT
    fprintf (test_fp, "time2: %s\n", time);
#endif
    sscanf( time, "%d-%d-%d %d:%d:%d",
          &day, &month, &year, &hour, &min, &sec );

    int3str (str, 2, day);
    str[2] = '-';

    int3str (&str[3], 2, month);
    str[5] = '-';

    int3str (&str[6], 4, year);
    str[10] = ' ';

    int3str (&str[11], 2, hour);
    str[13] = ':';

    int3str (&str[14], 2, min);
    str[16] = ':';

    int3str (&str[17], 2, sec);
}
```

dec2str .c

```
#include "tpcc.h"
#include "ex_tpcc_info.h"
#include <floatingpoint.h>

/*
   dec2str:
   Converts a double precision floating point value to a
string of
   a specified length and outputs the string to the
memory buffer supplied.
   This routine assumes the following restrictions apply:
   Precision is fixed at 2 places to the right of the
decimal point.
   No string length will be less than 4.

   field = the destination field
   field_size = number of characters to output
   value = floating point number to be displayed
*/

void dec2str(char *str, int len, double num)
{
    int dec, sign, i, cnt;
    char *string;

    string = ecvt(num, len-1, &dec, &sign);

    /* dec = À°¿ôÉô¤Î·å¿ô,sign =
À¸¤Ï0,Éé¤Ï1,string=¾®¿ôÀÀ¤ò½ü¤¤¤¿¿ô»ú¤¬ÀßÄê¤
¸¡¤¤ë */

    if ( dec > 0 ) {
                /* if the integer part is not zero .. Exsample
:mum data is 1234.56 */
                cnt = (len - 3) - dec;

        /* À°¿ôÉô¤Î¿ô»ú¤Ç¥¼¥í¤ò¶õÇò¤ËÀßÄê¤¹¤ë:
"0012" -> "  12" */
        /* If the high-order digit is zero , zero is changed at
the blank */
                for (i = 0; i < cnt; i++){
                    /* pad with blank in the high p art of the
number */
                    str[i] = ' ';
                }

                /* The high-order digit set to the output
area: À°¿ôÉô¤Î¿ô»ú¤òÀßÄê¤¹¤ë*/
                for (; i < (len - 3); i++){
                    str[i] = *(string++);
                }
    }
    else {
                /* If the integer part is zero ... Exsample:
num data is 0.12 */
                cnt = len - 4;

                for (i = 0; i < cnt; i++){
                    /* pad with blank in the high part of the
number */
                    str[i] = ' ';
                }
                str[i++] = '0';
    }

    str[i] = '.';
    i++;

    for (; dec < 0 && i < len; dec++,i++){
                /* pad with 0's in  the high part of the
fraction */
```

```
                str[i] = '0';
    }

    for (; i < len; i++){
                /* copy the decimal portion (2 places) */
                str[i] = *(string++);
    }
}

/*
  sigdec2str:
  Converts a double precision floating point value to a
string of
  a specified length and outputs the string to the
supplied buffer.
  If the value is negative, the first character will be a
minus sign (-).

  field = the destination field
  field_size = number of characters to output
  value = floating point number to be displayed
*/

void sigdec2str(char *str, int len, double num)
{
    if (num >= 0.0) {
                str[0] = ' ';
                dec2str (&str[1], len - 1, num);
    } else {
                str[0] = '-';
                dec2str (&str[1], len - 1, -num);
    }
}
```

delivery.c

```
#include "tpcc.h"
#include "tpcweb.h"
#include "delpage.h"
#include "ex_tpcc_info.h"

#include <sys/time.h>

/*
    delivery : this function proccesses the delivery
transaction.
*/
int delivery (char *s_buf, rte_input_data *in_data,
tpc_struct *tpc_area)
{
    struct delstruct *bp;
    int  i;
    int  ol_cnt, cnt, rtn;

    struct timeval timeque;

    bp = ( struct delstruct * )tpc_area->trans_b;
    bp->tran_kind = TRANDEL;

 /* ------------------------------------------------- Check the Input
data */
    bp->delin.w_id = now_w_id;

    if ((bp->delin.o_carrier_id = str2int(in_data-
>O_CARRIER_ID, 2)) < 0 )
        return set_errpage(s_buf, now_cookie, 5, bp-
>delin.o_carrier_id);

    bp->delin.in_timing_int = 1;

 /* ---------------------------------------- Execute Delivery
transaction */

 resend_delivery:

#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 */

    DBGR(fprintf ( test_fp, "tpacall delivery!\n" ));

    DBGR(tsp(0));
    rtn = tpacall(now_ottname, ( char * )tpc_area-
>trans_b,
                                sizeof( struct delstruct ),
TPSIGRSTRT | TPNOREPLY );
    DBGR(tsp(1));

#else
    dummy_delivery( bp );
    rtn = 0;
#endif

    sprintf (s_work, h_del2);

 /* ---------------------------------------- The execution result is
checke. */

    if ( rtn == -1 ) {

    /* Display messege */

#ifndef SCRTEST
                char    buf[1024];

                switch ( tperrno ) {
                case TPELIMIT:  /*
¥·¥â¡¼¤¬°ìÇÕ¤Ë¤Ê¤Ã¤¿¤¿¤á¼è¦ý¤Ç¤-¤Ê¤¤ */
                case TPETIME:   /*
¥È¥é¥ó¥¶¥¯¨¥·¥ç¥ó¥¿¥¤¥à¥¢¥¦¥ÈÈ¯À¸ */
                case TPGOTSIG:  /* ¥·¥° ¥Ê¥ë¤¬È¯À¸¤·¤¿
*/

/*                  sprintf ( s_work, "tpacall : Retry in
Delivery: tperrno = %d\n"
                                " svc = '%s' carrier = %d\n",
tperrno,
                                now_ottname, bp-
>delin.o_carrier_id );
*/
                /* Because it is an executable again error,
                   processing is executed again. */
                    goto resend_delivery;
                    break;

                case TPESVCFAIL:
        sprintf( s_work, "Oracle failed to process
Delivery Transaction.\n"
                "tperrno = %d svc = '%s' carrier = %d\n",
tperrno,
                                now_ottname, bp-
>delin.o_carrier_id );

        set_oraerr( s_buf, s_work );
        return (-1);

                default:
                    /* The error which was not able to be
executed again occurred */
                    sprintf( s_work, "tpacall failed in
Delivery: tperrno = %d\n"
                                " svc = '%s' car rier = %d\n",
tperrno,
                                now_ottname, bp-
>delin.o_carrier_id );

                    set_tuxerr (s_buf, s_work);
                    return (-1);
                }
#endif

    } else {
                int2str ((s_work + delp[0]), 4, bp-
>delin.w_id);
                int2str ((s_work + delp[1]), 2, bp-
>delin.o_carrier_id);
                alp2str ((s_work + delp[2]), 25, "Delivery
has been queued");
    }

/* -------- The execution result data notified RTE is made
by the HTML form */

    sprintf(s_buf, h_del1);
    strcat (s_buf, s_work);

    sprintf(s_work, h_del3, SOPATH, now_cookie);
    strcat (s_buf, s_work);

    return 0;
}
```

dsppara.c

```
#include "tpcc.h"
#include "tpcweb.h"
#include "ex_tpcc_info.h"

/*
   !! Debug use only.
   Print ransaction input and output data
*/

#ifdef DBPRT

int  oder_dsp(rte_input_data *in_data, tpc_struct
*tpc_area, int d_flag)
{
    int i;
    struct ordstruct *bp;

    bp = ( struct ordstruct * )tpc_area->trans_b;

    if (d_flag == 0){
                fprintf (test_fp, "----- in data area -----\n\n");
```

```
                fprintf(test_fp,"w_id = %d ",now_w_id);

                fprintf (test_fp,  "d_id = %s ",in_data->D_ID);
                if (in_data->C_ID != 0)
                   fprintf (test_fp, "c_id = %s \n",in_data->C_ID);
                if (in_data->C_LAST != 0)
                   fprintf (test_fp, "c_last = %s \n",in_data->C_LAST);

                fprintf (test_fp, "----- trans buf area -----\n\n");
                fprintf (test_fp, "w_id = %d ", bp->ordin.w_id);
                fprintf (test_fp, "d_id = %d ", bp->ordin.d_id);
                fprintf (test_fp, "c_id = %d\n", bp->ordin.c_id);
                if ( bp->ordin.c_last[0] == '\0') {
                   fprintf (test_fp, "byname = %d \n", bp->ordin.bylastname);
                } else {
                   fprintf (test_fp, "c_last = %s :byname = %d\n",bp->ordin.c_last,
                                bp->ordin.bylastname);
                }
   }
   else {
                fprintf (test_fp, "----- trans buf area (after) -----\n\n");
      fprintf(test_fp,"w_id = %d  ", bp->ordin.w_id);
      fprintf(test_fp,"d_id = %d  ", bp->ordin.d_id);
      fprintf(test_fp,"c_id = %d\n", bp->ordout.c_id);
      fprintf(test_fp,"c_first=%s ", bp->ordout.c_first);
      fprintf(test_fp,"c_middl=%s ", bp->ordout.c_middle);
      fprintf(test_fp,"c_last =%s\n", bp->ordout.c_last);

      fprintf(test_fp,"c_balan=%f ", bp->ordout.c_balance);
      fprintf(test_fp,"o_id =%d ", bp->ordout.o_id);
      fprintf(test_fp,"o_entry_d=%s\n", bp->ordout.o_entry_d );

      if ( bp->ordout.o_carrier_id != 0 ) {
         fprintf(test_fp,"o_carrier_id=%d\n", bp->ordout.o_carrier_id);
      }

      for( i = 0; i < bp->ordout.o_ol_cnt; i++ ){

         fprintf(test_fp,"ol_supp=%d ", bp->ordout.ol_supply_w_id[i]);
         fprintf(test_fp,"ol_i_id=%d ", bp->ordout.ol_i_id[i]);
         fprintf(test_fp,"ol_quan=%d ", bp->ordout.ol_quantity[i]);
         fprintf(test_fp,"ol_amou=%f\n", bp->ordout.ol_amount[i]);
      }
   }
}

int  pay_dsp(rte_input_data *in_data, tpc_struct *tpc_area, int d_flag)
{
   struct paystruct *bp;
   int    i, rtn;
   bp = ( struct paystruct * )tpc_area->trans_b;

   if (d_flag == 0){
                fprintf (test_fp, "----- in data area -----\n\n");

                fprintf(test_fp,"w_id = %d ",now_w_id);
                fprintf (test_fp, "d_id = %s ",in_data->D_ID);
                fprintf (test_fp, "c_w_id=%s ",in_data->C_W_ID);
                fprintf (test_fp, "c_d_id=%s ",in_data->C_D_ID);
                fprintf (test_fp, "h_amount=%s \n",in_data->H_AMOUNT);

                if (in_data->C_ID != 0)
                   fprintf (test_fp, "c_id = %s \n",in_data->C_ID);
                if (in_data->C_LAST != 0)
                   fprintf (test_fp, "c_last = %s \n",in_data->C_LAST);

                fprintf (test_fp, "----- trans buf area -----\n\n");
                fprintf (test_fp, "w_id = %d ", bp->payin.w_id);
                fprintf (test_fp, "d_id = %d ", bp->payin.d_id);
                fprintf (test_fp, "c_id = %d ",  bp->payin.c_id);
                if ( bp->payin.c_last[0] == '\0') {
                   fprintf (test_fp, "byname = %d \n", bp->payin.bylastname);
                } else {
                   fprintf (test_fp, "c_last = %s :byname = %d\n",bp->payin.c_last,
                                bp->payin.bylastname);
                }
                fprintf (test_fp, "c_w_id=% d ",bp->payin.c_w_id);
                fprintf (test_fp, "c_d_id=%d ",bp->payin.c_d_id);
                fprintf (test_fp, "h_amount=%f \n",bp->payin.h_amount);
   }
   else {
                fprintf (test_fp, "----- trans buf area (after) -----\n\n");
      fprintf(test_fp,"w_id = %d  ", bp->payin.w_id);
      fprintf(test_fp,"d_id = %d  ", bp->payin.d_id);
                fprintf(test_fp,"c_id = %d\n", bp->payout.c_id);

                fprintf(test_fp,"w_str_1=%s  ",bp->payout.w_street_1);
                fprintf(test_fp,"w_str_2=%s\n",bp->payout.w_street_2);
                fprintf(test_fp,"d_str_ 1=%s  ",bp->payout.d_street_1);
                fprintf(test_fp,"d_str_2=%s\n",bp->payout.d_street_2);
                fprintf(test_fp,"w_city=%s ", bp->payout.w_city);
                fprintf(test_fp,"w_state=%s\n", bp->payout.w_state);
                fprintf(test_fp,"d_city=%s ", bp->payout.d_city);
                fprintf(test_fp,"d_state=%s\n", bp->payout.d_state);

                fprintf(test_fp,"c_w_id=%d  ", bp->payin.c_w_id);
                fprintf(test_fp,"d_w_id=%d\n", bp->payin.c_d_id);

      fprintf(test_fp,"c_first=%s ", bp->payout.c_first);
      fprintf(test_fp,"c_middl=%s ", bp->payout.c_middle);
      fprintf(test_fp,"c_last =%s\n", bp->payout.c_last);

                fprintf(test_fp,"c_str_1=%s  ",bp->payout.c_street_1);
                fprintf(test_fp,"c_str_2=%s\n",bp->payout.c_street_2);
                fprintf(test_fp,"c_city=%s\n", bp->payout.c_city);
                fprintf(test_fp,"c_credi=%s  ",bp->payout.c_credit);
                fprintf(test_fp,"c_state=%s\n", bp->payout.c_state);

      fprintf(test_fp,"c_balan=%f\n", bp->payout.c_balance);

                i = strlen( bp->payout.c_data );
      fprintf(test_fp,"c_date=%s\n", bp->payout.c_data);
   }
}

int  sto_dsp(rte_input_data *in_data, tpc_struct *tpc_area, int d_flag)
{

   struct stostruct *bp;
   int i;

   bp = ( struct stostruct * )tpc_area->trans_b;

   if (d_flag == 0){
      fprintf (test_fp, "----- in data area -----\n\n");

      fprintf(test_fp,"w_id = %d ",now_w_id);
      fprintf(test_fp,"d_id = %d  ",now_d_id);

      fprintf (test_fp, "threshold= %s \n",in_data->threshold);

      fprintf (test_fp, "----- trans buf area -----\n\n");
      fprintf (test_fp, "w_id = %d   ", bp->stoin.w_id);
      fprintf (test_fp, "d_id = %d   ", bp->stoin.d_id);
      fprintf (test_fp, "threshold= %d \n",bp->stoin.threshold);
   }
   else{
      fprintf (test_fp, "----- trans buf area (after) -----\n\n");

      fprintf (test_fp, "w_id = %d   ", bp->stoin.w_id);
      fprintf (test_fp, "d_id = %d   ", bp->stoin.d_id);
      fprintf (test_fp, "threshold= %d   ",bp->stoin.threshold);
      fprintf (test_fp, "low_stock= %d \n",bp->stoout.low_stock);
```

```
    }
}

int
new_dsp(rte_input_data *in_data, tpc_struct *tpc_area,
int d_flag, int cnt)
{

    struct newstruct *bp;
    int i, loop;

    bp = ( struct newstruct * )tpc_area->trans_b;

    if (d_flag == 0){
        fprintf (test_fp, "----- in data area -----\n\n");

        fprintf(test_fp,"w_id = %d ",now_w_id);

        fprintf (test_fp, "d_id = %s ",in_data->D_ID);
        fprintf (test_fp, "c_id = %s \n", in_data->C_ID);

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

            if( in_data->OL_SUPPLY_W_ID[i] != 0 ){
                fprintf(test_fp,"ol_sup_w_id=%s ",in_data-
>OL_SUPPLY_W_ID[i]);
            }

            if( in_data->OL_I_ID[i] != 0 ){
                fprintf (test_fp, "ol_i_id=%s ", in_data-
>OL_I_ID[i]);
            }

            if( in_data->OL_QUANTITY[i] != 0 ){
                fprintf (test_fp, "ol_quan=%s\n", in_data-
>OL_QUANTITY[i]);
            }
        }

        fprintf (test_fp, "----- trans buf area -----\n\n");
        fprintf (test_fp, "w_id = %d ", bp->newin.w_id);
        fprintf (test_fp, "d_id = %d ", bp->newin.d_id);
        fprintf (test_fp, "c_id = %d\n", bp->newin.c_id);

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

            fprintf (test_fp, "ol_sup_w_id=%d ",  bp-
>newin.ol_supply_w_id[i]);
            fprintf (test_fp, "ol_i_id=%d ",  bp-
>newin.ol_i_id[i]);
            fprintf (test_fp, "ol_quan=%d\n", bp-
>newin.ol_quantity[i]);
        }
    }
    else{
        fprintf (test_fp, "----- trans buf area (after) -----
\n\n");

        fprintf (test_fp, "c_last=%s  ", bp->newout.c_last);
        fprintf (test_fp, "c_credit=%s\n", bp-
>newout.c_credit);
        fprintf (test_fp, "o_id=%d  ", bp->newout.o_id);

        fprintf (test_fp, "o_entry_d=%s\n", bp-
>newout.o_entry_d);
        fprintf (test_fp, "c_discnt=%f\n", bp-
>newout.c_discount * 100.0);
        fprintf (test_fp, "o_ol_cnt=%d  ", bp-
>newout.o_ol_cnt);

        fprintf (test_fp, "w_tax=%f  ", bp->newout.w_tax *
100.0);
        fprintf (test_fp, "d_tax=%f\n", bp->newout.d_tax *
100.0);

        loop = bp->newout.o_ol_cnt;
        for ( i = 0; i < loop; i++ ) {

            fprintf(test_fp," - - - - - - \no_sup_w_id=%d ",
                                                bp-
>newin.ol_supply_w_id[i]);
            fprintf(test_fp,"o_i_id=%d  ", bp-
>newin.ol_i_id[i]);
            fprintf(test_fp,"i_name=%s\n", &bp-
>newout.i_name[i][0]);
            fprintf(test_fp,"o_quant=%d ",bp-
>newin.ol_quantity[i]);
            fprintf(test_fp,"s_quant=%d ",bp-
>newout.s_quantity[i]);
            fprintf(test_fp,"brand=%c ",bp-
>newout.brand_generic[i]);
            fprintf(test_fp,"i_price=%f ",bp-
>newout.i_price[i]);
            fprintf(test_fp,"ol_amnt=%f\n",bp-
>newout.ol_amount[i]);
        }
        fprintf (test_fp, "status=%s\n", bp->newout.status);
        fprintf (test_fp, "total_a=%f\n", bp-
>newout.total_amount);
    }
}
int  env_dsp(tpc_struct *tpc_area)
{
    int cnt, cnt2;

    fprintf (test_fp, "cl_num = %d\n", tpc_area-
>clent_num );
    fprintf (test_fp, "max_user = %d\n", tpc_area-
>max_user );
    fprintf (test_fp, "ott_num = %d\n", tpc_area->ott_num
);

    for (cnt = 0; cnt < tpc_area->clent_num; cnt++){

        fprintf (test_fp, "cl-%d,0 = %d  ", cnt, tpc_area-
>clent[cnt][0]);
        fprintf (test_fp, "cl-%d,1 = %d\n", cnt, tpc_area-
>clent[cnt][1]);
        }

   for (cnt = 0; cnt < tpc_area->clent_num; cnt++){

        fprintf (test_fp, "ott = %d",tpc_area->ott[cnt][0]);

        for (cnt2 = 1; cnt2 < tpc_area->ott_num; cnt2++){
            fprintf (test_fp, ",%d", tpc_area->ott[cnt][cnt2]);
        }
        fprintf (test_fp, "\n");
    }
}
#endif
```

fastlogin.c

```
#include "tpcc.h"
#include "tpcweb.h"
#include "menupage.h"
#include "ex_tpcc_info.h"

/*
   fastlogin:
   This function reads a user's responses to the login
form, sets
   up the user context, and returns the menu page.
*/

int fastlogin ( char *s_buf, rte_input_data *in_data,
tpc_struct *tpc_area ){

    int  cnt, diff, old, loop_cnt;
    char flag=0;

    if(in_data->cookie ==  0 || !strlen(in_data->cookie)){
                sprintf (s_buf, loginerr2, SOPATH );
                return 0;
    }

    now_w_id = (now_cookie - 1)/10 + 1;
    now_d_id = (now_cookie - 1)%10 + 1;

    if ((svrnum = getsvrnam ( now_cookie , tpc_area )) <
0 ){
                /*The terminal number exceeded the
maximum value */
                sprintf(s_buf, noconnt, tpc_area-
>max_user, now_cookie);
                return 0;
    }

    sprintf( now_ottname, "TPCC%02d", svrnum );

    DBGR(fprintf( test_fp, "ottname=%s : term NO = %d
(w_id = %d)\n",\
                    now_ottname, now_cookie, now_w_id));

    sprintf(s_buf, h_menu, SOPATH, now_cookie);
    return 0;
}

/* ----------------------------------------------------------------------
---
   The function number of the TP application program
which requests processing
   is acquired.  ( Get TPCCxx name in tpccsvr.ott )
   ----------------------------------------------------------------------
-- */
int getsvrnam ( int cookie, tpc_struct *tpc_area ){

    int svnum = 1;
    int num, cnt, loop;

    num = 0;

    if (tpc_area->max_user < cookie || cookie == 0){
        DBGP(fprintf (test_fp, "Term NO(%d) is not
support!\n", cookie));
                return(-1);
    }
/*  COMMENT OUT : 98.01.13
========================================
=====
    for (cnt = 1; cnt < tpc_area->clent_num; cnt++){
        if ( cookie < tpc_area->clent[cnt][0]){
```

```
            break;
        }
    }

    cnt--;
    num = tpc_area->clent[cnt][0];
    DBGP(fprintf (test_fp,"cookie=%d (client=%d,
max=%d, num=%d ->",\
                cookie, cnt, tpc_area->clent[cnt][0], num));

    for (loop = 0; loop < tpc_area->clent[cnt][1]; loop++){

        num += tpc_area->ott[cnt][loop];

        if (num > cookie){
            break;
        }
        else{
            svnum++;
        }
    }
================================================
================================== */
    DBGP(fprintf(test_fp," svn=%d ott-cnt=%d,
num=%d)\n", svnum, loop, num));

    return (svnum);
}

/* ---------------------------------------------------------------------
    The w_id, d_id are acquired. and call the function
getsvnam
   ---------------------------------------------------------------------
*/
int idget ( char *s_buf, tpc_struct *tpc_area ){

    /* make w_id, d_id */
    now_w_id = (now_cookie - 1)/10 + 1;
    now_d_id = (now_cookie - 1)%10 + 1;

    if ((svrnum = getsvrnam ( now_cookie , tpc_area)) <
0 ){

                /*The terminal number exceeded the
maximum value */
                sprintf(s_buf, noconnt, tpc_area-
>max_user, now_cookie);
                return -1;
    }

    /* make the TP Application program name */
    sprintf( now_ottname, "TPCC%02d", svrnum );

    DBGR(fprintf( test_fp, "ottname=%s : term NO = %d
(w_id = %d)\n",\
                    now_ottname, now_c ookie, now_w_id));

    sprintf(s_buf, h_menu, SOPATH, now_cookie);
    return 0;
}
```

gettpccenv.c

```
#include "tpcc.h"
#include "tpcweb.h"
#include "menupage.h"
#include "ex_tpcc_info.h"

char            *point;
FILE            *envget;

/* ---------------------------------------------------------------------
  toval
  Convert the value after the delimitation character into
numeric data.
  this function read infomation for the cartridge program
to operate.
 ---------------------------------------------------------------------  */
int toval (char *line, char deli){

    int   val;

REGET:
    if (fgets (line, 256, envget) != NULL){

/* The line which starts by # is a comment line.*/
                if (strchr (line, '#') != NULL) goto REGET;

/* Convert the value after the delimitation character into
numeric data.*/

                if ((point = strchr (line, deli)) != NULL){
                    point++;
                    val = atoi (point);
                    return ( val );
                }
                else
                    return 0;
    }
    else
                return 0;
}

/* ---------------------------------------------------------------------
  nextval
  Convert the value after the delimitation character into
numeric data.

 ---------------------------------------------------------------------  */
int nextval (char deli){

    int    val;
    char  *v_pt;

/* Convert the value after the delimitation character into
numeric data.*/
    if ((v_pt = strchr (point, deli)) != NULL){
        v_pt++;
        val = atoi (v_pt);
                point = v_pt;
        return ( val );
    }
    else
        return 0;
}

/* ---------------------------------------------------------------------
  gettpccenv
  Information for the cartridge program to operate is
read from the file,
  and information is set in an external variable.
 ---------------------------------------------------------------------  */

int gettpccenv ( tpc_struct *tpc_area )
{
    int             cnt, cnt2;
    char  line[256];

    envget = fopen( "/weblog/env/webenv", "r");

    /* The number of client machine which executes test
*/
    tpc_area->clent_num = toval(&line[0], '=');

    /* Number of terminals(user) which executes test */
    tpc_area->max_user = toval(&line[0], '=');

    /* The maximam number of application programs
which operate with client
       machine */
    tpc_area->ott_num  = toval(&line[0], '=');

    /* The first number of terminal(user) which individual
client simulates,
       and Number of application programs which operate
by each client */
    for (cnt = 0; cnt < tpc_area->clent_num; cnt++){

        tpc_area->clent[cnt][0] = toval(&line[0],'{' );
        tpc_area->clent[cnt][1] = nextval(',');
    }

    /* Number of terminals(users) which individual TP
application program
       simulates */
    for (cnt = 0; cnt < tpc_area->clent_num; cnt++){

        tpc_area->ott[cnt][0] = toval(&line[0],'{');

        for (cnt2 = 1; cnt2 < tpc_area->ott_num; cnt2++){
            tpc_area->ott[cnt][cnt2] = nextval(',');
                }
    }

    fclose (envget);
}
```

neworder.c

```
#include "tpcc.h"
#include "tpcweb.h"
#include "newpage.h"
#include "ex_tpcc_info.h"

#define  SUPPLY_NG     0x01
#define  I_ID_NG       0x02
#define  QUANTITY_NG   0x04

/* ---------------------------------------------------------------------
  chk_NOdata :
  VerifyNewOrderLine verifies that a user's inputs for a
line in
  the New Order form are okay.
      return -5  : w_id abnormal value : Not Number or
more than 3000WH
      return -6  : i_id abnormal value : Not Number
      return -7  : ol_quantity abnormal value : Not
Number
 ---------------------------------------------------------------------
- */
int chk_NOdata (struct newstruct *bp, int cnt,
rte_input_data *in_data)
{

    char flag = 0;
```

```
    if( in_data->OL_SUPPLY_W_ID[cnt] == 0 &&
in_data->OL_I_ID[cnt] == 0 &&
        in_data->OL_QUANTITY[cnt] == 0 ){

                /* Find last order line :
1'ÔÊ¬¤ÎÆþÏ¥Ç¡¼¿¥¿¡Ì¡¤ */
        bp->newin.ol_i_id[cnt] = 0;
        bp->newin.ol_quantity[cnt] = 0;
        bp->newin.ol_supply_w_id[cnt] = 0;
                return 0;
    }

    if( in_data->OL_SUPPLY_W_ID[cnt] != 0 ){

        if((bp->newin.ol_supply_w_id[cnt] =
            str2int (in_data->OL_SUPPLY_W_ID[cnt], 4)) <
1 ||
           bp->newin.ol_supply_w_id[cnt] > MAXWH )
            return -5;                                  /*
w_id abnormal */

    }
    else
                flag |= SUPPLY_NG;

    if( in_data->OL_I_ID[cnt] != 0 ){

        if((bp->newin.ol_i_id[cnt] =
                    str2int (in_data->OL_I_ID[cnt], 6)) < 0)
            return -6;                                  /* i_id
abnormal value */

                /* Convert 0 into -1. if this function set 0
then the TP application
                   send the return code of abnormal end */
                else if (bp->newin.ol_i_id[cnt] == 0)
                    bp->newin.ol_i_id[cnt] = -1;
    }
    else
                flag |= I_ID_NG;

    if( in_data->OL_QUANTITY[cnt] != 0 ){

        if((bp->newin.ol_quantity[cnt] =
            str2int (in_data->OL_QUANTITY[cnt], 2)) < 0)
                    return -7;                          /*
ol_quantity abnormal value */
    }
    else
                flag |= QUANTITY_NG;

    if (flag != 0){

                /* the order lien data is abnormal : there is
a uninput item */
        DBGR(fprintf(test_fp, "neworder ol data check
flag=%d\n",flag));

        if((flag & SUPPLY_NG) != 0 )   return -8;
        if((flag & I_ID_NG) != 0)      return -1;
        if((flag & QUANTITY_NG) != 0)  return -2;
    }
    else{
                /* the order lien data is normal */
                return 1;
    }
}
```

```
/* ----------------------------------------------------------------
   setNOdata : This function set the execution result
data of the TP
                        applicatin program.

   OF is an offset value to the next line data.
   cnt is line number
 ----------------------------------------------------------------
- */

int setNOdata (char *s_work,int OF,int cnt,struct
newstruct *bp,rte_input_data *in_data)
{

    if(!bp->newin.ol_i_id[cnt] ) {
                alp2str ((s_work + OF + newp[11]), 78, " ");
                return -1;
    } else {

                int2str((s_work + OF + newp[11]), 4, bp-
>newin.ol_supply_w_id[cnt]);

                alp2str((s_work + OF + newp[12]), 6,
in_data->OL_I_ID[cnt]);

                alp2str((s_work + OF + newp[13]), 24,bp-
>newout.i_name[cnt]);

                int2str((s_work + OF + newp[14]), 2, bp-
>newin.ol_quantity[cnt]);
                int2str((s_work + OF + newp[15]), 3, bp-
>newout.s_quantity[cnt]);
                alp2str((s_work + OF + newp[16]), 1, &bp-
>newout.brand_generic[cnt]);
                dec2str((s_work + OF + newp[17]), 6,
(double)bp->newout.i_price[cnt]);
                dec2str((s_work + OF + newp[18]),
7,(double)bp->newout.ol_amount[cnt]);
                return 0;
    }
}

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

  neworder : this function proccesses the NewOrder
transaction.

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

int neworder (char *s_buf, rte_input_data *in_data,
tpc_struct *tpc_area)
{
    struct newstruct *bp;
    int  i;
    int  ol_cnt, cnt, rtn;
    int  work_i;

    bp = ( struct newstruct * )tpc_area->trans_b;
    bp->tran_kind = TRANNEW;

/* ------------------------------------------------- check the
Input data */
    bp->newin.w_id = now_w_id;

    if((bp->newin.d_id = str2int (in_data->D_ID, 2)) < 1 ||
       bp->newin.d_id > 10 )
        return set_errpage(s_buf, now_cookie, 2, bp-
>newin.d_id);

    if((bp->newin.c_id = str2int (in_data->C_ID, 4)) < 0)
        return set_errpage(s_buf, now_cookie, 6, bp-
>newin.c_id);

    ol_cnt = 0;
    for (cnt = 0; cnt < 15; cnt++){

        if ((rtn = chk_NOdata( bp, cnt, in_data)) < 0 ){
            return set_errpage(s_buf, now_cookie, 13 +
ol_cnt, rtn);
                }
                else if (rtn == 0 && ol_cnt == 0){
            return set_errpage(s_buf, now_cookie, 13 +
ol_cnt, -5);
                }
        else if (rtn == 0)
                    break;
                else /* Order Line data is normal: rtn == 1
*/
                    ol_cnt++;
    }


    DBGR(new_dsp(in_data, tpc_area, 0, ol_cnt));

/* ------------------------------------------- Execute NewOrder
transaction */
resend_neworder:

#ifndef SCRTEST

    DBGR(tsp(0));
    if ( tpcall( now_ottname, ( char * )tpc_area->trans_b,
                                sizeof( struct newstruct ),
            ( char ** )&tpc_area->trans_b, &olen, 0 ) == -
1 ) {

        if ( tperrno == TPESVCFAIL ) {
            sprintf( s_work, "Oracle failed to process
NewOrder Transaction.\n"
                "tperrno = %d svc = '%s' d_id = %d c_id = %d
lines = %d\n",
                tperrno, now_ottname, bp->newin.d_id, bp-
>newin.c_id, ol_cnt );

            set_oraerr( s_buf, s_work );
            return (-1);
        }

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

        set_tuxerr(s_buf, s_work);
        return (-1);
    }
    DBGR(tsp(1));

    bp = ( struct newstruct * )tpc_area->trans_b;
#else
    dummy_neworder( bp );
#endif
```

```
    DBGR(new_dsp(in_data, tpc_area, 1, 1));

    sprintf (s_work, h_new2);

    int2str ((s_work + newp[0]), 4, bp->newin.w_id);
    int2str ((s_work + newp[1]), 2, bp->newin.d_id);
    int2str ((s_work + newp[3]), 4, bp->newin.c_id);

    alp2str ((s_work + newp[4]), 16, bp->newout.c_last);
    alp2str ((s_work + newp[5]), 2, bp->newout.c_credit);
    int2str ((s_work + newp[7]), 8, bp->newout.o_id);

    if ( bp->newout.terror == NOERR ){

        cnt = bp->newout.o_ol_cnt;

        time2str((s_work + newp[2]), bp-
>newout.o_entry_d);
        dec2str ((s_work + newp[6]), 5, (double)(bp-
>newout.c_discount*100.0));

        int2str ((s_work + newp[8]), 2,  bp-
>newout.o_ol_cnt);

        dec2str ((s_work + newp[9]),  5, (double)(bp-
>newout.w_tax * 100.0));
        dec2str ((s_work + newp[10]), 5, (double)(bp-
>newout.d_tax * 100.0));

        for ( i = 0; i < cnt; i++ ) {
            setNOdata (s_work, 0x4f*i, i, bp, in_data);
        }

        /* "Item number is not valid" or "" ('\0') */
        alp2str ((s_work + newp[19]), 24, bp-
>newout.status);
        dec2str ((s_work + newp[20]), 8,
                    (double)(bp->newout.total_amount));
    }

#ifndef SCRTEST
    else{
        if ( bp->newout.terror == IRRECERR ){

            sprintf (s_work, "Irrecoverable error in
NewOrder\n" );
                    set_tuxerr (s_buf, s_work);
                    return (-1);
        }
        else{
            goto resend_neworder;          /* Retry
NewOrder transaction */
        }
    }
#endif

/* ------- The execution result data notified RTE is make
by the HTML form */

    sprintf(s_buf, h_new1);
    strcat (s_buf, s_work);

    sprintf(s_work, h_new3, SOPATH, now_cookie);
    strcat (s_buf, s_work);

    return (0);
}
```

orderst.c

```
#include "tpcc.h"
#include "tpcweb.h"
#include "odrpage.h"
#include "ex_tpcc_info.h"

/*------------------------------------------------------------------------
  orderstatus : this function proccesses the Orderstatus
transaction
 ------------------------------------------------------------------------
*/
int orderstatus (char *s_buf, rte_input_data *in_data,
tpc_struct *tpc_area)
{
    struct ordstruct *bp;
    int  i, rtn;

    char c_id_flag = NG;

    bp = ( struct ordstruct * )tpc_area->trans_b;
    bp->tran_kind = TRANORD;

/* ------------------------------------------------------ check the
Input data */
    bp->ordin.w_id = now_w_id;

    /* check d_id data */
    if ((bp->ordin.d_id = str2int(in_data->D_ID, 2)) < 1 ||
                bp->ordin.d_id > 10)
        return set_errpage(s_buf, now_cookie, 2, bp-
>ordin.d_id);

    /* check c_id data */
    if ((bp->ordin.c_id = str2int(in_data->C_ID, 4)) != -3 ){

                DBGP(fprintf (test_fp,     "check
c_id=%x\n", bp->ordin.c_id ));

                if (bp->ordin.c_id < 0 ){
            return set_errpage(s_buf, now_cookie, 6, bp-
>ordin.c_id);
        }
        else{
                    c_id_flag = OK;
                }
    }
    else{
                bp->ordin.c_id = 0;
    }

    /* check c_last data */
    if((rtn = str2str(in_data->C_LAST, 16)) < 0){
        return set_errpage(s_buf, now_cookie, 7, rtn);
    }
    else{
        if ( rtn == 0 || *(in_data->C_LAST) == '\0') {
                    bp->ordin.bylastname = 0;
                } else {
                    strcpy (bp->ordin.c_last, in_data-
>C_LAST);
                    bp->ordin.bylastname = 1;
                    c_id_flag = OK;
        }
    }

    /* c_id and c_last is nothing */
    if (c_id_flag == NG)
        return set_errpage(s_buf, now_cookie, 11, 0);

 DBGP(oder_dsp ( in_data, tpc_area, 0));

/* ---------------------------------------- Execute Orderstatus
transaction */
resend_orderstatus:

#ifndef SCRTEST

    DBGR(tsp(0));
    if ( tpcall( now_ottname, ( char * )tpc_area->trans_b,
                            sizeof(struct ordstruct),
                            ( char ** )&tpc_area->trans_b,
&olen, 0 ) == -1 ){

        if ( tperrno == TPESVCFAIL ) {
            sprintf( s_work, "Oracle failed to process
OrderStatus Transaction.\n"
                "tperrno = %d svc = '%s' d_id = %d c_id = %d
c_last = '%s'\n",
                tperrno, now_ottname, bp->ordin.d_id, bp-
>ordin.c_id,
                bp->ordin.c_last );

            set_oraerr( s_buf, s_work );
            return (-1);
        }

                sprintf( s_work, "tpcall failed in
OrderStatus: tperrno = %d\n"
                            " svc = '%s' d_id = %d c_id =
%d c_last = '%s'\n",
                            tperrno, now_ottname, bp-
>ordin.d_id, bp->ordin.c_id,
                            bp->ordin.c_last );

                set_tuxerr (s_buf, s_work);
                return (-1);
    }
    DBGR(tsp(1));

    bp = ( struct ordstruct * )tpc_area->trans_b;

/* ---------------------------------------- Check the execution
result */
    if ( bp->ordout.terror != NOERR ){

                if ( bp->ordout.terror == IRRECERR ) {

                    sprintf (s_work, " Irrecoverable error in
orderstatus.\n" );
                    set_tuxerr (s_buf, s_work);
                    return (-1);
                }
                goto resend_orderstatus;  /* tuxedo busy ->
retry!! */
    }

#else
    dummy_orderstat( bp );
#endif

    DBGP(oder_dsp ( in_data, tpc_area, 1));

    sprintf(s_work, h_order2);
    int2str ((s_work + orderp[0]), 4, bp->ordin.w_id);
```

```
    int2str ((s_work + orderp[1]), 2, bp->ordin.d_id);
    int2str ((s_work + orderp[2]), 4, bp->ordout.c_id);
    alp2str ((s_work + orderp[3]), 16, bp->ordout.c_first);
    alp2str ((s_work + orderp[4]), 2,  bp-
>ordout.c_middle);
    alp2str ((s_work + orderp[5]), 16, bp->ordout.c_last);

    sigdec2str ((s_work + orderp[6]), 9, (double)bp-
>ordout.c_balance);

    int2str ((s_work + orderp[7]), 8, bp->ordout.o_id );
    time2str ((s_work + orderp[8]), bp->ordout.o_entry_d
);

    if ( bp->ordout.o_carrier_id != 0 ) {
                int2str ((s_work + orderp[9]), 2, bp-
>ordout.o_carrier_id);
    }

    /* 0x39 is an offset value to the same filed of the next
line */
    for( i = 0; i < bp->ordout.o_ol_cnt; i++ ){

        int2str ((s_work+i*0x39+orderp[10]), 4, bp-
>ordout.ol_supply_w_id[i]);
        int2str ((s_work+i*0x39+orderp[11]), 6, bp-
>ordout.ol_i_id[i]);
        int2str ((s_work+i*0x39+orderp[12]), 2, bp-
>ordout.ol_quantity[i]);
        sigdec2str ((s_work+i*0x39+orderp[13]), 8,
                    (double)bp->ordout.ol_amount[i]);

                if( strncmp( bp->ordout.ol_delivery_d[i],
"NOT DELIVR", 10 ) != 0 ){
          date2str ((s_work+i*0x39+orderp[14]), bp-
>ordout.ol_delivery_d[i]);
                }
    }

/* ------- The execuction result data notified RTE is make
by the HTML form */

    sprintf(s_buf, h_order1);               /* set Header Data
*/
    strcat (s_buf, s_work);                 /* set Result Data
*/

    sprintf (s_work, h_order3, SOPATH, now_cookie); /*
set Tailer Data */
    strcat (s_buf, s_work);

    return 0;
}
```

p_err.c

```
#include <stdio.h>
#include "tpcc.h"
#include "tpcweb.h"
#include "ex_tpcc_info.h"

/* Error message list : these are notified from CLINET
to RTE */
char errstrings[17][166] = {
"The function you selected doesn't exist.\r\n"
"Don't enter URLs manually!\r\n%s",                 /* 0 */

"You seem to have responded to a form that doesn't
exist.\r\n"
"Don't enter URLs manually!\r\n%s",                 /* 1 */

"The District ID you entered isn't valid.\r\n"
"It must be an integer in the range 1 to 10.\r\n%s", /* 2
*/

"The threshold value you entered isn't valid.\r\n"
"It must be an integer of 2 or fewer digits.\r\n%s", /* 3 */

"The terminal number you entered isn't valid.\r\n"
"It must be an integer in the range 1 to
max_user.\r\n%s", /* 4 */

"The Carrier ID you entered isn't valid.\r\n"
"It must be an integer of 4 or fewer digits.\r\n%s", /* 5 */

"The Customer ID you entered isn't valid.\r\n"
"It must be an integer of 4 or fewer digits.\r\n%s", /* 6 */

"The Customer Last Name you entered isn't valid.\r\n"
"It must be a string shorter than 16 characters.\r\n%s",
/* 7 */

"The Payment Amount you entered isn't valid.\r\n"
"It must be a dollar amount, without the dollar sign,
between $1.00 and $9999.99.\r\n%s", /* 8 */

"The Customer Warehouse ID you entered isn't
valid.\r\n"
"It must be an integer in the range 1 to
MAXWH.\r\n%s", /* 9 */

"The Customer District ID you entered isn't valid.\r\n"
"It must be an integer of 2 or fewer digits.\r\n%s", /* 10
*/

"You must enter either a Customer ID or a Customer
Last Name.\r\n"
"You left both fields blank.\r\n%s", /* 11 */

"The Warehouse ID you entered isn't valid.\r\n"
"It must be an integer in the range 1 to
MAXWH.\r\n%s", /* 12 */

"On entry line %d, the data you entered for the %s field
isn't valid.\r\n"
"It must be an integer of %d or fewer digits.", /* 13 */

"Supply Warehouse ID", /* 14 */

"Item ID", /* 15 */

"Quantity", /* 16 */
};

/*
  set_errpage:
  a generic error page generator. If the user does
anything screwy,
  s/he gets here.  The function generates an error page
based on the
  two errlvl arguments and returns it for the user..

  When err_no is 13 or more, Order Line Data is
Abnormal.
  ( err_no is the error data line number )
 */
int set_errpage (char *buf, int user, int err_no, int
err_inf) {
    char errmsg[1024];
    int nchar;

    if(err_no >= 13) {                  /* OrderLIne
Data(Neworder) is Abnormal */
            switch(err_inf) {
            case -5:                    /* S_W_ID data is
abnormal */
            case -8:                    /* S_W_ID data is
uninput */
                nchar = 4;                          /*
S_W_ID data length */
                err_inf = 14;           /* set sub error
messeage number (S_W_ID) */
                break;

            case -1:                    /* I_ID data is
uninput */
            case -6:                    /* I_ID data is
abnormal */
                nchar = 6;                          /* I_ID
data length */
                err_inf = 15;           /* set sub error
messeage number (item id) */
                break;

            case -2:                    /* Quantity data is
abnormal */
            case -7:                    /* Quantity data is
uninput */
                nchar = 2;                          /*
Quantity data length */
                err_inf = 16;           /* set sub error
messeage number (Quantity) */
                break;

            default:
                break;
            }

            sprintf(errmsg, errstrings[13],err_no-
12,errstrings[err_inf],nchar);
            sprintf(buf, errhtml, errmsg, SOPATH,
user);

    } else {
            switch(err_inf) {
            case -3: /* There is not Input data */
                sprintf(errmsg, errstrings[err_no],
                        "You didn't enter anything
for the field.");
                break;

            case -1: /* too many characters */
                sprintf(errmsg, errstrings[err_no],
                        "Your entry contained too
many characters.");
                break;

            case -2: /* Not all digits */
                sprintf(errmsg,  errstrings[err_no],
                        "The characters you typed
were not all digits.");
                break;

            case 0:
```

```
                sprintf(errmsg, errstrings[err_no], " ");
                break;

            default: /* Other error */
                sprintf(errmsg, errstrings[err_no],
                        "Your entry was too l arge
numerically.");
                break;
            }

            sprintf(buf, errhtml, errmsg, SOPATH,
user);
            printf("%s", buf);
    }

    DBGR(fprintf (test_fp, "This Transaction is parameter
ERROR\n"));
    return 0;
}

/*
  set_tuxerr :
  this function make error message of the TP-
application program.
*/
int set_tuxerr (char *page, char *err_inf) {

    tpterm ( );
    sprintf(page, tuxerr, err_inf, SOPATH);

    return 0;
}

/*
  set_oraerr :
  this function make error message of the Oracle
application program.
*/
int set_oraerr (char *page, char *err_inf) {

    sprintf(page, oraerr, err_inf, SOPATH);

    return 0;
}
```

payment.c

```
#include "tpcc.h"
#include "tpcweb.h"
#include "paypage.h"
#include "ex_tpcc_info.h"

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

  payment : this function proccesses the Payment
transaction.

  ----------------------------------------------------------------------
-*/
int payment (char *s_buf, rte_input_data *in_data,
tpc_struct *tpc_area)
{
    struct paystruct *bp;
    int    i, rtn;
    float  h_amount; /* tool kit ?? */
    char   c_id_flag = NG;

    bp = ( struct paystruct * )tpc_area->trans_b;
    bp->tran_kind = TRANPAY;

/* ------------------------------------------------- check the
Input data */
    bp->payin.w_id = now_w_id;

    /* check d_id data */
    if((bp->payin.d_id = str2int (in_data->D_ID, 2)) < 1 ||
                bp->payin.d_id > 10)
        return set_errpage(s_buf, now_cookie, 2, bp-
>payin.d_id);

    /* check c_id data */
    if((bp->payin.c_id = str2int (in_data->C_ID, 4)) != -3
){
        if (bp->payin.c_id < 0 ){
            return set_errpage(s_buf, now_cookie, 6, bp-
>payin.c_id);
        }
        else{
            c_id_flag = OK;
        }
    }
    else{
        bp->payin.c_id = 0;
    }

    /* check c_last data */
    if((rtn = str2str(in_data->C_LAST, 16)) < 0){
        return set_errpage(s_buf, now_cookie, 7, rtn);
    }
    else{
        if ( rtn == 0 || *(in_data->C_LAST) == '\0') {
                    bp->payin.bylastname = 0;
                } else {
                    strcpy (bp->payin.c_last, in_data-
>C_LAST);
                    bp->payin.bylastname = 1;
                    c_id_flag = OK;
        }
    }

    /* c_id and c_last data is nothing */
    if (c_id_flag == NG)
                return set_errpage(s_buf, now_cookie, 11,
0);

    /* check c_w_id data */
    if((bp->payin.c_w_id = str2int (in_data->C_W_ID, 4))
< 1 ||
        bp->payin.c_w_id > MAXWH )
        return set_errpage(s_buf, now_cookie, 9, bp-
>payin.c_w_id);

    /* check c_d_id data */
    if((bp->payin.c_d_id = str2int (in_data->C_D_ID, 2))
< 1 ||
                bp->payin.c_d_id > 10 )
        return set_errpage(s_buf, now_cookie, 10, bp-
>payin.c_d_id);

    /* check h_amount data :
       str2dbl do hundredfold of the H_ANOUNT. The
pupose of if is to prosess
       H_AMOUNT by the integer */
    if((bp->payin.h_amount = str2dbl (in_data-
>H_AMOUNT, 7)) < 100 ||
        bp->payin.h_amount > 999999)
        return set_errpage(s_buf, now_cookie, 8, bp-
>payin.h_amount);

    DBGP(pay_dsp(in_data, tpc_area, 0));


/* ------------------------------------------- Execute Payment
transaction */
resend_payment:

#ifndef SCRTEST

    DBGR(tsp(0));

    if ( tpcall( now_ottname,
                                (char *)tpc_area->trans_b,
sizeof(struct paystruct),
                ( char ** )&tpc_area->trans_b, &olen, 0 ) == -
1 ) {

        if ( tperrno == TPESVCFAIL ) {
            sprintf( s_work, "Oracle failed to process
Payment Transaction.\n"
                "tperrno = %d svc = '%s' d_id = %d c_id = %d
c_last = '%s'\n",
                                tperrno, now_ottname, 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 );

            set_oraerr( s_buf, s_work );
            return (-1);
        }

                sprintf( s_work, "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 = %f\n",
                                tperrno, now_ottname, 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 );

                set_tuxerr( s_buf, s_work);
                return (-1);
    }
    DBGR(tsp(1));

    bp = ( struct paystruct * )tpc_area->trans_b;

/* ------------------------------------------- Check the execution
result */
    if ( bp->payout.terror != NOERR ){

                if ( bp->payout.terror == IRRECERR ) {

            sprintf (s_work, "Irrecoverable error in
Payment\n" );
                    set_tuxerr( s_buf, s_work);
                    return (-1);
                }
```

```
                goto resend_payment;    /* TP_base busy.
Try again */
    }
#else
    dummy_payment( bp );
#endif

    DBGR(pay_dsp(in_data, tpc_area, 1));

    sprintf (s_work, h_pay2);
    time2str ((s_work + payp[0]), bp->payout.h_date );

    int2str ((s_work + payp[1]), 4, bp->payin.w_id);
    int2str ((s_work + payp[2]), 2, bp->payin.d_id);

    alp2str ((s_work + payp[3]), 20, bp-
>payout.w_street_1);
    alp2str ((s_work + payp[4]), 20, bp-
>payout.d_street_1);
    alp2str ((s_work + payp[5]), 20, bp-
>payout.w_street_2);
    alp2str ((s_work + payp[6]), 20, bp-
>payout.d_street_2);
    alp2str ((s_work + payp[7]), 20, bp->payout.w_city);
    alp2str ((s_work + payp[8]), 2, bp->payout.w_state);
    zip2str ((s_work + payp[9]), bp->payout.w_zip);
    alp2str ((s_work + payp[11]), 20, bp->payout.d_city);
    alp2str ((s_work + payp[12]), 2, bp->payout.d_state);
    zip2str ((s_work + payp[13]), bp->payout.d_zip);

    int2str ((s_work + payp[15]), 4, bp->payout.c_id);
    int2str ((s_work + payp[16]), 4, bp->payin.c_w_id);
    int2str ((s_work + payp[17]), 2, bp->payin.c_d_id);

    alp2str ((s_work + payp[18]), 16, bp->payout.c_first);
    alp2str ((s_work + payp[19]), 2,  bp-
>payout.c_middle);
    alp2str ((s_work + payp[20]), 16, bp->payout.c_last);

    date2str ((s_work + payp[21]), bp->payout.c_since);

    alp2str ((s_work + payp[22]), 20, bp-
>payout.c_street_1);
    alp2str ((s_work + payp[23]), 2, bp->payout.c_credit);

    alp2str ((s_work + payp[24]), 20, bp-
>payout.c_street_2);
    dec2str ((s_work + payp[25]), 5, (double)(bp-
>payout.c_discount * 100.0));

    alp2str ((s_work + payp[26]), 20, bp->payout.c_city);
    alp2str ((s_work + payp[27]), 20, bp-
>payout.c_state);
    zip2str ((s_work + payp[28]), bp->payout.c_zip);
    phone2str ((s_work + payp[29]), bp-
>payout.c_phone);

    h_amount = (float)bp->payin.h_amount / (float)100;
    dec2str ((s_work + payp[30]), 7, (double)h_amount);

    sigdec2str ((s_work + payp[31]), 14, (double)bp-
>payout.c_balance );
    dec2str ((s_work + payp[32]), 13, (double)bp-
>payout.c_credit_lim );

    i = strlen( bp->payout.c_data );
    alp2str ((s_work + payp[33]), 50, bp->payout.c_data);
    if ( i > 50 ){
                alp2str ((s_work + payp[34]), 50, &bp-
>payout.c_data[50] );
                if ( i > 100 ){
                    alp2str ((s_work + payp[35]), 50, &bp-
>payout.c_data[100]);
                    if ( i > 150 ){
                        alp2str ((s_work + payp[36]), 50, &bp-
>payout.c_data[150]);
                    }
                }
    }

/* ------- The execuction result data notified RTE is make
by the HTML form */

    sprintf(s_buf, h_pay1);                 /* set Header Data
*/
    strcat (s_buf, s_work);                 /* set Result Data
*/

    sprintf(s_work, h_pay3, SOPATH, now_cookie); /*
set Tailer Data */
    strcat (s_buf, s_work);

    return (0);
}
```

select_tran.c

```
#include "tpcc.h"
#include "tpcweb.h"
#include "tpcinweb.h"
#include "menupage.h"
#include "ex_tpcc_info.h"


/* ---------------------------------------------------------------
   select_trn:
   interprets information from the user's input data to
determine which
   page should be displayed back to the user.

   in_data - a pointer to a raw_form_data structure with
pointers
            to values in 'query'.
   ---------------------------------------------------------------- */

int select_trn ( rte_input_data *in_data, char *s_buf,
tpc_struct *tpc_area) {

/*
    if(in_data->form) {
*/
    if (in_data->form && (in_data->form[0] != 'M') ) {

                if (in_data->form[0] == 'I' ){
                    /* send the transaction select screen
page */
                    return fastlogin (s_buf, in_dat a,
tpc_area);
                }
                else{

                    /* Get w_id, d_id, TP applicatin function
name */
                    if (idget (s_buf, tpc_area) < 0)
                                return -1;
                    /* check transaction type */
                    switch(in_data->form[0]) {
                    case 'N':
                        return neworder (s_buf, in_ data,
tpc_area);

                    case 'D':
                        return delivery(s_buf, in_data,
tpc_area);

                    case 'P':
                        return payment (s_buf, in_data,
tpc_area);

                    case 'S':
                        return stocklevel(s_buf, in_data,
tpc_area);

                    case 'O':
                        ret urn orderstatus (s_buf, in_data,
tpc_area);

                    default:
                                /* uninput transaction type */
                        set_errpage(s_buf, now_cookie, 1, 0);
                        return 1;
                    }
                }
    }
    else if(in_data->button) {

                /* Get w_id, d_id, TP application function
name */
                if (idget (s_buf, tpc_area ) < 0)
return -1;

                /* send the data input screen page */
        switch(in_data->button[0]) {
        case 'N':
            sprintf(s_buf, in_newpage, SOPATH,
now_cookie, now_w_id);
            return 0;

        case 'D':
            sprintf(s_buf, in_delpage, SOPATH,
now_cookie, now_w_id);
            return 0;

        case 'P':
            sprintf(s_buf, in_paypage, SOPATH,
now_cookie, now_w_id);
            return 0;

        case 'S':
            sprintf(s_buf, in_stkpage, SOPATH,
now_cookie, now_w_id, now_d_id);
            return 0;

        case 'O':
            sprintf(s_buf, in_odrpage, SOPATH,
now_cookie, now_w_id);
            return 0;

        case 'Q':
                    /* This value use WWW brouser only. */
            now_w_id = 0; now_d_id = 0;
```

```
        sprintf (s_buf, loginpage , SOPATH);
        return 0;

      default:
                /* uninput transaction type */
        set_errpage(s_buf, now_cookie, 0, 0);
        return 0;
      }
   }
   else {

            /* if there is not parameta then send login
page data.
              this part use WWW brouser only */
            sprintf (s_buf, loginpage, &SOPATH);
            return 0;
   }
}
```

stocklv.c

```
#include "tpcc.h"
#include "stpage.h"
#include "ex_tpcc_info.h"

#include <stdio.h>

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

  stocklevel : this function proccesses the StockLevel
transaction.

  -------------------------------------------------------------------------
-*/
int stocklevel (char *s_buf, rte_input_data *in_data,
tpc_struct *tpc_area)
{
   struct stostruct  *bp;
   int loopc = 0;
   int rtn = 0;

   bp = ( struct stostruct *)tpc_area->trans_b;
   bp->tran_kind = TRANSTO;

/* ------------------------------------------------- check the
Input data */
   bp->stoin.w_id = now_w_id;
   bp->stoin.d_id = now_d_id;

   /* check threshold data */
   if((bp->stoin.threshold = str2short(in_data->threshold,
2)) < 0)
            return set_errpage(s_buf, now_cookie, 3,
bp->stoin.threshold);

   DBGP(sto_dsp(in_data, tpc_area, 0));


/* ---------------------------------------- Execute Stock Level
transaction */
resend_stock:

#ifndef SCRTEST

   DBGR(tsp(0));
   if ( tpcall( now_ottname, ( char * )tpc_area->trans_b,
                        sizeof( struct stostruct ),
            ( char ** )&tpc_area->trans_b, &olen, 0 ) == -
1 ){

      if ( tperrno == TPESVCFAIL ) {
         sprintf( s_work, "Oracle failed to process
StockLevel Transaction.\n"
            "tperrno = %d svc = '%s' threshold = %d\n",
tperrno,
                        now_ottname, bp-
>stoin.threshold );

         set_oraerr( s_buf, s_work );
         return (-1);
      }

            sprintf( s_work, "stockLevel: tperrno =
%d\n"
                        " svc = '%s' threshold = %d\n",
tperrno,
                        now_ottname, bp-
>stoin.threshold );

            set_tuxerr (s_buf, s_wo rk);
            return (-1);
   }
   DBGR(tsp(1));

   bp = ( struct stostruct * )tpc_area->trans_b;

/* ------------------------------------------ Check the execution
result */
   if ( bp->stoout.terror != NOERR ){
            if ( bp->stoout.terror == IRRECERR ) {

               sprintf( s_work, "Irrecoverable error in
stocklevel \n");
               set_tuxerr (s_buf, s_work);
               return (-1);
            }
            goto resend_stock;      /* TP application
busy. Try again */
   }
#else
   dummy_stocklvl ( bp );
#endif

   DBGP(sto_dsp(in_data, tpc_area, 1));

   sprintf (s_work, h_stock2);
   int2str ((s_work + stockp[0]), 4, bp->stoin.w_id);
   int2str ((s_work + stockp[1]), 2, bp->stoin.d_id);
   int2str ((s_work + stockp[2]), 2, bp->stoin.threshold);
   int2str ((s_work + stockp[3]), 3, bp-
>stoout.low_stock);

/* ------- The execuction result data notified RTE is make
by the HTML form */

   sprintf(s_buf, h_stock1);            /* Set Header data
*/
   strcat (s_buf, s_work);              /* Set Result data
*/

   sprintf(s_work, h_stock3, SOPATH, now_cookie); /*
Set Tailer data */
   strcat (s_buf, s_work);

   return (0);
}
```

str2num.c

```
#include "tpcc.h"
#include "ex_tpcc_info.h"
#include <stdlib.h>

#define  numcheck(num) ( 0x30 <= num && num <=
0x39 ) /* 0 - 9 */
#define  alpcheck(num) ( 0x41 <= num && num <= 0x5a
) /* A - Z */

/*
  str2int :
  takes a string, makes sure it's not too long, and
ensures that it
  represents an integer.
  If it does, the corresponding int value is returned.

  -3: there is not string data.
  -2: find not character data.
  -1: string data is too many long
*/

int str2int(char *str, int field_len) {
   int x;

   if(str == 0 || !(x = strlen(str))) return -3;
   if(x > field_len){
            return -1;
   }
   else{
            for( ; x ; x--){
               if (!numcheck(str[x-1])) {
                        return -2;
               }
      }
   }
   return atoi(str);
}

/*
  str2short :
  takes a string, makes sure it's not too long, and
ensures that it
  represents an integer.
  If it does, the corresponding short value is returned.

  -3: there is not string data.
  -2: find not character data.
  -1: string data is too many long
*/

short str2short(char *str, int field_len) {
   int x;

   if(str == 0 || !(x = strlen(str))) return -3;
   if(x > field_len) return -1;
   else {
            for( ; x; x--){
               if (!numcheck(str[x-1]))
                        return -2;
      }
   }
   x = atoi(str);
   return (short)x;
}
```

```
/*
   str2str :
   makes sure the string exists and isn't too long.

   -3: string data is too many long
   -2: find not figure data.
    0: there is not string data.
    1: normal end
*/
int str2str(char *str, int field_len) {
    int x;

    if(str == 0) return 0;
    x = strlen(str);
    if(x == 0) return 0;

    if(x > field_len ) return -3;
    else {
                for( ; x ; x--){
                    if (!alpcheck(str[x-1]))
                                return -2;
        }
    }
    return 1;
}


/*
   str2dbl :
   takes a string, makes sure it's not too long, and
makes sure that it
   represents a floating point number.
   If so, delete the decimal point.
   As a result, the value is increased hundredfold.
   this function is returned integer value.

   !! This function use Payment transaction only.

   -3: there is not string data.
   -2: find not character data.
   -1: string data is too many long

 */
int str2dbl(char *str, int field_len) {
    int x, len, cnt;
    double num;
    char   NUM[7];

    char   pointf = 0;
    int    fcnt = 2;              /* */


    if(str == 0 || !(x = strlen(str))) return -3;
    len = x;
    if(x > field_len)
                return -1;
    else{
                /* check string data */
                for(;x;x--) {
                    if(numcheck(str[x-1]));
                    else if((str[x-1] == '.') && ((len - x) < 3));
                    else if((str[x-1] == '-') && (x == 1));
                    else if((str[x-1] == '+') && (x == 1));
                    else return -2;
                }
    }
    /* delete the decimal point. As a result,do
hundredfold the value.*/
    for (cnt = 0, x = 0; x < len; x++){

                if ( str[x] == '.') {
                    /* find the decimal point. set point flag.*/
                    pointf = 1;
                } else {
                    /* set character to work buffer.*/
                    NUM[cnt] = str[x]; cnt++;

                    /* The figure below the deci mal point
was detected */
                    if ( pointf == 1 ) {fcnt--;}
        }
    }

    if ( pointf == 1 && fcnt > 0 ){

                /*There was no figure below the decimal
point or only one digit was
                  found.:
¾®¿ôÅÀ°Ê²¼¤Î¿ô»ú¤¬Ì¤¤¤¤Þ¤¿¤ï£±·å¤Î¾¡ç¤Î½èÍý */
                for ( ; fcnt > 0 ; fcnt--) {
                    NUM[cnt++] = '0';
                }
    }
    else if ( pointf == 0 ) {
                /* There is no decimal point.:
¾®¿ôÅÀ¤¬¤Ê¤¤¾¡ç¤Î½èÍý */
                NUM[cnt++] = '0'; NUM[cnt++] = '0';
    }

    NUM[cnt] = 0;

    return (atoi(NUM));
}
```

strct_init.c

```
#include "tpcc.h"
#include "ex_tpcc_info.h"
/*
   struct_init :
   init_ptrs sets everything in the annoyingly long
raw_form_data structure
   to zero.
*/

void strct_init (rte_input_data *in_data) {

    int  cnt = 0;

    in_data->button = 0;
    in_data->cookie = 0;
    in_data->form = 0;
    in_data->O_CARRIER_ID = 0;
    in_data->threshold = 0;
    in_data->D_ID = 0;
    in_data->C_ID = 0;
    in_data->C_W_ID = 0;
    in_data->C_D_ID = 0;
    in_data->C_LAST = 0;
    in_data->H_AMOUNT = 0;

    for (cnt = 0; cnt < 15; cnt++)
                in_data->OL_SUPPLY_W_ID[cnt] = 0;

    for (cnt = 0; cnt < 15; cnt++)
        in_data->OL_I_ID[cnt] = 0;

    for (cnt = 0; cnt < 15; cnt++)
        in_data->OL_QUANTITY[cnt] = 0;
}
```

tpcc.c/* ----------

```
------
   Client application program of TPC-C
   (this program is dynamic link library of Oracle
WebServer)
   ---------- */

#include "tpcc.h"
#include "tpcweb.h"
#include "tpcc_info.h"

WRBReturnCode tpcc_init( );
WRBReturnCode tpcc_exec( );
WRBReturnCode tpcc_shut( );

int   nument;
FILE  *errfile;

/* ----------
  tpccentry

  Function: This function registers the callback function.
  Note        : This function is to register three functions
as callback function
                with the WRB Execution Engine (WRBX),
which is the program that
                actually dynamically loads cartridge at
runtime.

                Three function is tpcc_init, tpcc_exec a nd
tpcc_shut.

   ----------
*/
WRBReturnCode tpccentry (WRBCalls)
WRBCallbacks *WRBCalls;
{
  WRBCalls->init_WRBCallback     = tpcc_init;
  WRBCalls->exec_WRBCallback     = tpcc_exec;
  WRBCalls->shut_WRBCallback     = tpcc_shut;

  return (WRB_DONE);
}

/* ----------
  tpcc_init

  Function: Initialization function of TPCC cartridge
  Note        : When the program starts, this function  is
executed only once.
                This function executes the following
processing.

                1) The interface area with WRB program
is acquired.
                2) The interface area with TP_base
program is acquired.
                3) Information is read from the operation
definition file,and stores
                  in an internal area.
   ----------
*/
```

```
WRBReturnCode tpcc_init( WRBCtx, tpcc_ctxp )
void   *WRBCtx;
void  **tpc_ctxp;
{
   tpc_struct *tpc_area;

   /* The interface area with WRB program is acquired.
: 1) */
   tpc_area = (tpc_struct *)malloc( sizeof(tpc_struct) );
   memset(tpc_area, 0x00, sizeof (tpc_struct));

   *tpc_ctxp= tpc_area;

 #ifndef SCRTEST
   if ( tpinit( NULL ) == -1 ){
               fprintf( stderr, "Failed to join the
application.\n" );
               exit( 1 );
   }
   /* The interface area with TP_base program is
acquired. : 2) */
   if ((tpc_area->trans_b = (void *)tpalloc(
"CARRAY",NULL, trans_size ))
               == NULL ){
               fprintf( stderr, "Tpalloc failed.\n" );
               exit( 1 );
   }
 #else
   if ((tpc_area->trans_b = (void *)malloc( trans_size ))
== NULL ) {
               /* fprintf( stderr, "Malloc failed.\n" ); */
               exit( 1 );
   }
 #endif

   memset( tpc_area->trans_b, 0, (size_t)trans_size );

   DBGR(tpc_area->prt_cnt = 1);

#ifdef GETENV
   /* Information is read from the operation definition
file,and stores
      in an internal area. :3) */
   gettpccenv(tpc_area);
#endif

   return (WRB_DONE);
}

/* ----------------------------------------------------------------
tpcc_exec

Function: Transaction execution function
Note          : When the transaction processing demand
by RTE is received, this
                function is executed.
                This function executes the following
processing.

                1) The input data storage region is
initialized.
                        (call function strct_init)

                2) The transaction data is analyzed.
                        (call function dataparse)

                3) Initialization of work area.

                4) The processing demanded by RTE is
analyzed,
                   and the demanded processing is
executed.
                        (call function select_trn)

                5) The result of processing is notified to
RTE by the HTML data form.
                6) When abnormal processing is
detected,information is stored in the
                   log file.
  ----------------------------------------------------------------
*/
WRBReturnCode tpcc_exec( WRBCtx, tpcc_ctxp )
void   *WRBCtx;
void   *tpc_ctxp;
{
   tpc_struct *tpc_area = (tpc_struct *)tpc_ctxp;

   rte_input_data in_data_area;

   int          rtn, len, cnt;
   char path_w[96];

#ifdef DBPRT
   long s_pid;

   s_pid = getpid( );
   sprintf (path_w, "/weblog/web%d", s_pid);
   test_fp = fopen (path_w, "aw+");
#endif

  trans_buf = tpc_area->trans_b;

#ifdef DBPRT
   fprintf (test_fp, "-------------------------\n");
   fprintf (test_fp, "tpc_areap = %x(trans_buf= %x)\n",
                tpc_area, tpc_area->trans_b);
#endif

   DBGP(env_dsp(tpc_area));

   /* Initialize of input data area : 1) */
   strct_init( &in_data_area );

   /* Parse input data : 2) */
   now_cookie = dataparse( WRBCtx ,&in_data_area );

   DBGP(fprintf (test_fp, "cookie=%d ,w_id=%d[%x]
,d_id=%d[%x]\n",\
              now_cookie, now_w_id, &now_w_id,
now_d_id, &now_d_id));

#if defined(DBPRT)
   fflush (test_fp);
#endif

   memset(s_buf,  0x00, BUF_S);
   memset(s_work, 0x00, WORK_S);

   /* Check the transactin type.and execute transaction
: 4) */
   rtn = select_trn ( &in_data_area, s_buf , tpc_area);

   len = strlen(s_buf);

   /* send the HTML Form DATA : 5) */
   WRBClientWrite (WRBCtx, s_buf, len);

   DBGR(fprintf (test_fp, "len=%d\n",len));
   DBGR(fprintf (test_fp, "%s",s_buf));

   DBGR(fprintf (test_fp, "\n output = %d\n", tpc_area-
>prt_cnt));

#ifdef DBPRT
   if (tpc_area->prt_cnt > 0x7FFFFFFF) tpc_area-
>prt_cnt = 0;
   fflush(test_fp);
   fclose(test_fp);
#endif

   /* If TP application returned abnormal end : make
error log file : 5) */
   if (rtn == -1){
      sprintf (path_w, "/weblog/tcerr%d", now_cookie);
      errfile = fopen (path_w, "w+");
      fprintf (errfile, "%s\n output = %d\n", s_buf,
tpc_area->prt_cnt);
               fflush  (errfile);
               fclose (errfile);
   }

#ifdef DBPRT
   tpc_area->prt_cnt++;
#endif

   return (WRB_DONE);
}

/* ----------------------------------------------------------------
  tpcc_shut

  Function: Postprocessed function of TPCC cartridge
  Note          : When the Oracle Webserver Cartridge
program ends, this function is
                  executed. This function executes the
following processing.

                  1) The interface area with the TP_base
program is released.
                  2) The interface area with the WRB
program is released.

  ----------------------------------------------------------------
*/
WRBReturnCode tpcc_shut( void *WRBCtx, void
*tpc_ctxp )
{
   tpc_struct *tpc_area = (tpc_struct *)tpc_ctxp;

   /* The interface area with the TP_base program is
released : 1)*/
   tpfree(tpc_area->trans_b);

   /* The interface area with the WRB program is
released. : 2) */
   free(tpc_ctxp);

  return (WRB_DONE);
}
```

tsp.c

```
#include "tpcc.h"
#include "ex_tpcc_info.h"
```

```
#ifdef DBPRT
/*
  Print Time stamp :  !! Debug use only
*/

int tsp( char flag )
{
    struct timeval tp;
    gettimeofday ( &tp );

    if ( flag == 0 )
        fprintf (test_fp, "start = %d.%d\n", tp.tv_sec,
tp.tv_usec);
    else
        fprintf (test_fp, "end   = %d.%d\n", tp.tv_sec,
tp.tv_usec);

    return 0;
}
#endif
```

zip2str.c

```
#include "tpcc.h"
#include "ex_tpcc_info.h"

/*
   zip2str:
   Outputs a zipcode in the supplied buffer in the
following format:
   XXXXX-XXXX

   str = the destination field
   zip = the zipcode to be output
 */

void zip2str (char *str, char *zip)
{
    alp2str (str, 5, zip);
    str[5] = '-';
    alp2str (&str[6], 4, &zip[5]);
}


/*
   phone2str:
   Outputs a phone number in the supplied buffer in the
following format:
   XXXXXX-XXX-XXX-XXXX

   str   = the destination field
   phone = the phone number to be output
*/
void phone2str(char *str, char *phone)
{
    alp2str (str, 6, phone);
    str[6] = '-';

    alp2str (&str[7], 3, &phone[6]);
    str[10] = '-';

    alp2str (&str[11], 3, &phone[9]);
    str[14] = '-';

    alp2str (&str[15], 4, &phone[12]);
}
```

Appendix B: Server Source Code

```
#

# $Header: Makefile 7030100.3 96/06/21 16:39:16 plai
Generic<base> $ Copyr (c) 1995 Oracle

#

#

#=========================================
========================+
#       Copyright (c) 1996  Oracle Corp, Redwood
Shores, CA       |
#                OPEN SYSTEMS PERFORMANCE
GROUP                |
#                    All Rights Reserved                    |
#=========================================
========================+
# FILENAME
#     Makefile
# DESCRIPTION
#     Makefile for bench/tpc/tpcc/source directory
#=========================================
========================
#
#  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
#

all: compile load setup

include $(ORACLE_HOME)/bench/buildtools/prefix.mk

DIRS=vendors
TARGS=compile load setup cleanup

# Le Kha added the line network/public

TPCBIN=$(ORACLE_HOME)/bench/tpc/bin
INCLUDE=$(I_SYM).
$(I_SYM)$(ORACLE_HOME)/rdbms/demo \
      $(I_SYM)$(ORACLE_HOME)/rdbms/public \
      $(I_SYM)$(ORACLE_HOME)/network/public \
      $(I_SYM)$(ORACLE_HOME)/rdbms/include \
      $(I_SYM)$(ORACLE_HOME)/plsql/public
ITUX=$(I_SYM)$(ROOTDIR)/include

MEMBS=
OBJS=tpccload.o c_trans.o c_drv.o c_dump.o tpccpl.o
getrand.o 90per.o
CTRAN_OBJS=plnew.o plpay.o plord.o pldel.o plsto.o
CTRANTUX_OBJS=plnew_tux.o plpay.o plord.o
pldel_tux.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
GETTIMEOBJ=${ORACLE_HOME}/bench/gen/source/
gettime.o

files:

compile: $(OBJS)
        @-$(DOTARGS)

load: tpcc.ott tpccload.ott getrand 90per
        @-$(DOTARGS)

cleanup:
        $(REMOVE) $(OBJS) $(CTRAN_OBJS)
$(CTRANTUX_OBJS) $(OTHER_OBJS) \
        $(TUX_OBJS)
        @-$(DOTARGS)

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
        $(REMOVE) 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
        $(REMOVE) 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
        $(REMOVE) test_sample.c

test_tran.o: test_tran.c tpcc.h tpcc_info.h
        $(CC) $(CFLAGS) $(INCLUDE) -c
test_tran.c

test_tran_new.o: test_tran_new.c tpcc.h tpcc_info.h
        $(CC) $(CFLAGS) $(INCLUDE) -c
test_tran_new.c

test8.o: test8.c
        $(CC) $(CFLAGS) $(INCLUDE) -c test8.c

c_trans.o: $(CTRAN_OBJS)
        $(LD) -r -o $@ $(CTRAN_OBJS)

c_trans_tux.o: $(CTRANTUX_OBJS)
        $(LD) -r -o $@ $(CTRANTUX_OBJS)

tpccpl.o: tpccpl.c tpcc.h tpcc_info.h tpccpl.h
        $(CC) $(CFLAGS) $(INCLUDE) -c tpccpl.c

tpccpl_trc.o: tpccpl.c tpcc.h tpcc_info.h tpccpl.h
        $(CC) $(CFLAGS) -DSQL_TRACE
$(INCLUDE) -c tpccpl.c
        mv tpccpl.o tpccpl_trc.o

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
        $(REMOVE) tpccpl_tux.c

plnew_tux.o: plnew.c tpcc.h tpccpl.h
        $(CP) plnew.c plnew_tux.c
        -$(CC) $(CFLAGS) -DTUX $(INCLU DE)
$(ITUX) -c plnew_tux.c
        $(REMOVE) plnew_tux.c

plnew.o: plnew.c tpcc.h tpccpl.h
        $(CC) $(CFLAGS) $(INCLUDE) -c plnew.c

plpay.o: plpay.c plpay.h 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_tux.o: pldel.c tpcc.h tpccpl.h
        $(CP) pldel.c pldel_tux.c
        -$(CC) $(CFLAGS) -DTUX $(INCLUDE)
$(ITUX) -c pldel_tux.c
        $(REMOVE) pldel_tux.c

pldel.o: pldel.c tpcc.h tpccpl.h
        $(CC) $(CFLAGS) $(INCLUDE) -c pldel.c

plsto.o: plsto.c tpcc.h tpccpl.h
        $(CC) $(CFLAGS) $(INCLUDE) -c plsto.c

tpccsvr.o: tpccsvr.c tpcc.h tpcc_info.h
        $(CC) $(CFLAGS) $(INCLUDE) $(ITUX) -c
tpccsvr.c

getrand.o: getrand.c
        $(CC) $(CFLAGS) $(INCLUDE) -c
getrand.c

90per.o: 90per.c
        $(CC) $(CFLAGS) $(INCLUDE)  -c 90per.c

getrand: getrand.o
        $(CC) $(CFLAGS) -o $@ getrand.o

90per: 90per.o
        $(CC) $(CFLAGS) -o $@ 90per.o

tpccload.ost: tpccload.o
        $(LINK) -o $@ $(LDFLAGS)
$(LDFLAGS_ORA) \
        tpccload.o $(GETTIMEOBJ) \
        $(STLIBS)
```

```
tpccload.ott: tpccload.o
		$(LINK) -o $@ $(LDFLAGS) \
		tpccload.o $(GETTIMEOBJ) \
		$(TTLIBS)

tpcc.ost: c_drv.o c_trans.o tpccpl.o c_dump.o
		$(LINK) -o $@ $(LDFLAGS)
$(LDFLAGS_ORA) \
		c_drv.o c_trans.o tpccpl.o c_dump.o
$(GETTIMEOBJ) \
		$(STLIBS)

tpcc.ott: c_drv.o c_trans.o tpccpl.o c_dump.o
		$(LINK) -o $@ $(LDFLAGS) \
		c_drv.o c_trans.o tpccpl.o c_dump.o
$(GETTIMEOBJ) \
		$(TTLIBS)

tpcc_trc.ott: c_drv.o c_trans.o tpccpl_trc.o c_dump.o
		$(LINK) -o $@ $(LDFLAGS) \
		c_drv.o c_trans.o tpccpl_trc.o c_dump.o
$(GETTIMEOBJ) \
		$(TTLIBS)

test_drv: c_drv_val.o test_drv.o c_dump.o
		$(CC) -o $@ $(LDFLAGS) \
		c_drv_val.o test_drv.o c_dump.o
$(GETTIMEOBJ) \
	-lm

test_sample: c_drv.o test_sample.o c_dump.o
		$(CC) -o $@ $(LDFLAGS) \
		c_drv.o test_sample.o c_dump.o
$(GETTIMEOBJ) \
		-lm

test_tran.ost: test_tran.o c_trans.o tpccpl.o c_dump.o
		$(LINK) -o $@ $(LDFLAGS)
$(LDFLAGS_ORA) \
		test_tran.o c_trans.o tpccpl.o c_dump.o
$(GETTIMEOBJ) \
		$(STLIBS)

test_tran.ott: test_tran.o c_trans.o tpccpl.o c_dump.o
		$(LINK) -o $@ $(LDFLAGS) \
		test_tran.o c_tran s.o tpccpl.o c_dump.o
$(GETTIMEOBJ) \
		$(TTLIBS)

ttn.ott: test_tran_new.o c_trans.o tpccpl.o c_dump.o
		$(LINK) -o $@ $(LDFLAGS) \
		test_tran_new.o c_trans.o tpccpl.o
c_dump.o $(GETTIMEOBJ) \
		$(TTLIBS)

test8: test8.o
		$(LINK) -o $@ $(LDFLAGS) \
		test8.o \
		$(TTLIBS)

tpcccli: c_drv_tux.o c_dump.o
		(CFLAGS="$(CFLAGS)"; export CFLAGS;
CC=$(CC); export CC; \
		 buildclient -v -o $@ \
		  -f '$(LDFLAGS) c_drv_tux.o c_dump.o
$(GETTIMEOBJ)' \
		  -l '-lm')

#tpccsvr.ott: tpccsvr.o c_trans_tux.o tpccpl_tux.o
#		(CFLAGS="$(CFLAGS)"; export CFLAGS;
CC=$(CC); export CC; \
#		 buildserver -v -o $@ \
#		  -s
NEWORDER,PAYMENT,ORDERSTATUS,DELIVERY,
STOCKLEVEL \
#		  -f '$(LDFLAGS) tpccsvr.o c_trans_tux.o
tpccpl_tux.o $(GETTIMEOBJ)' \
#		  -l '$(TTLIBS)')

tpccsvr.ott: tpccsvr.o c_trans_tux.o tpccpl_tux.o
		(CFLAGS="$(CFLAGS)"; export CFLAGS;
CC=$(CC); export CC; \
		 buildserver -v -o $@ \
		  -s
TPCC01,TPCC02,TPCC03,TPCC04,TPCC05,TPCC06,
TPCC07,TPCC08,TPCC09,TPCC10,TPCC11,TPCC12,
TPCC13,TPCC14,TPCC15,TPCC16,TPCC17,TPCC18,
TPCC19,TPCC20,TPCC21,TPCC22,TPCC23,TPCC24,
TPCC25,TPCC26,TPCC27,TPCC28,TPCC29,TPCC30,
TPCC31,TPCC32,TPCC33,TPCC34,TPCC35,TPCC36,
TPCC37,TPCC38,TPCC39,TPCC40 \
		  -f '$(LDFLAGS) tpccsvr.o c_trans_tux.o
tpccpl_tux.o $(GETTIMEOBJ)' \
		  -l '$(TTLIBS)')

tpccsvr.ost: tpccsvr.o c_trans_tux.o tpccpl_tux.o
		(CFLAGS="$(CFLAGS)"; export CFLAGS;
CC=$(CC); export CC; \
		 buildserver -v -o $@ \
		  -s
NEWORDER,PAYMENT,ORDERSTATUS,DELIVERY,
STOCKLEVEL \
		  -f '$(LDFLAGS) $(LDFLAGS_ORA)' \
		  -f 'tpccsvr.o c_trans_tux.o tpccpl_t ux.o
$(GETTIMEOBJ)' \
		  -l '$(STLIBS)')

setup: tpcc.ott tpccload.ott getrand 90per
		if [ ! -d $(TPCBIN) ]; then \
		  $(MKDIR) $(TPCBIN); \
		fi;
		$(REMOVE) $(TPCBIN)/tpccload
		($(CD) $(TPCBIN); $(SYM_LINK)
../tpcc/source/tpccload.ott tpccload)
		$(REMOVE) $(TPCBIN)/tpcc.ott
		($(CD) $(TPCBIN); $(SYM_LINK)
../tpcc/source/tpcc.ott tpcc.ott)
		$(REMOVE) $(TPCBIN)/getrand
		($(CD) $(TPCBIN); $(SYM_LINK)
../tpcc/source/getrand getrand)
		$(REMOVE) $(TPCBIN)/90per
		($(CD) $(TPCBIN); $(SYM_LINK)
../tpcc/source/90per 90per)
		@-$(DOTARGS)
```

plpay.h

```
#include <oci.h>



struct payctx {

  OCIStmt *curpi;

  OCIStmt *curp0;

  OCIStmt *curp1;

  OCIBind *w_id_bp;

  OCIBind *w_id_bp1;

  sb2 w_id_ind;

  ub2 w_id_len;
  ub2 w_id_rc;

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

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

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

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

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

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

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

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

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

  OCIBind *w_state_bp;
  OCIBind *w_state_bp1;
  sb2 w_state_ind;
```

```
ub2 w_state_len;
ub2 w_state_rc;

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

OCIBind *c_credit_lim_bp;
OCIBind *c_credit_lim_bp1;
sb2 c_credit_lim_ind;
ub2 c_credit_lim_len;
ub2 c_credit_lim_rc;

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

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

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

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

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

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

OCIBind *byln_bp;
sb2 byln_ind;
ub2 byln_len;
ub2 byln_rc;
};

typedef struct payctx payctx;
```

tpcc.h

```
/*

 * $Header: tpcc.h 7030100.1 95/07/19 15:10:55 plai
Generic<base> $ Copyr (c) 1993 Oracle

 */

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

|       Copyright (c) 1995  Oracle Corp, Redwood
Shores, CA     |

|                OPEN SYSTEMS PERFORMANCE
GROUP              |
|                   All Rights Reserved                    |

+=========================================
========================+
| FILENAME
|    tpcc.h
| DESCRIPTION
|    Include file for TPC-C benchmark programs.

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

#ifndef TPCC_H
#define TPCC_H


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

#include "oratypes.h"
#include "oci.h"
#include "ocidfn.h"
/*
#ifdef __STDC__
#include "ociapr.h"
#else
#include "ocikpr.h"
#endif
*/

typedef struct cda_def csrdef;
typedef struct cda_def ldadef;
```

```
/* TPC-C transaction functions */

extern int TPCinit ();
extern int TPCnew ();
extern int TPCpay ();
extern int TPCord ();
extern int TPCdel ();
extern int TPCsto ();
extern 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
#define DEL_ERROR -666
#define DEL_DATE_LEN 7
#define NDISTS 10
#define NITEMS 15
#define SQL_BUF_SIZE 8192
#endif
```

tpcc_info.h

```
/*

 * $Header: tpcc_info.h 7030100.1 95/07/19 15:11:37
plai Generic<base> $ Copyr (c) 1995 Oracle

 */

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

|       Copyright (c) 1995  Oracle Corp, Redwood
Shores, CA     |
|                OPEN SYSTEMS PERFORMANCE
GROUP               |
|                    All Rights Reserved                    |

+=========================================
========================+
| FILENAME
|    tpcc_info.h
| DESCRIPTION
|    Include file for TPC-C benchmark programs.

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

#ifndef TPCC_INFO_H
#define TPCC_INFO_H

/* [ADD] 1997.09.03 CREO)yamamoto */
/*   Kind of transactions */
#define TRANNEW 1
#define TRANPAY 2
#define TRANORD 3
#define TRANDEL 4
#define TRANSTO 5
/* [END] */

/* 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 {
/* [ADD] 1997.10.06 CREO)yamamoto */
/*   Kind of transactions */
   int tran_kind;
/* [END] */
   struct newinstruct newin;
   struct newoutstruct newout;
};


/* Payment */

struct payinstruct {
   int w_id;
   int d_id;
   int c_w_id;
   int c_d_id;
   int c_id;
   int bylastname;
   int h_amount;
   char c_last[17];
};

struct payoutstruct {
   int terror;
   char w_street_1[21];
   char w_street_2[21];
   char w_city[21];
   char w_state[3];
   char w_zip[10];
   char d_street_1[21];
   char d_street_2[21];
   char d_city[21];
   char d_state[3];
   char d_zip[10];
   int  c_id;
   char c_first[17];
   char c_middle[3];
   char c_last[17];
   char c_street_1[21];
   char c_street_2[21];
   char c_city[21];
   char c_state[3];
   char c_zip[10];
   char c_phone[17];
   char c_since[11];
   char c_credit[3];
   double c_credit_lim;
   float c_discount;
   double c_balance;
   char c_data[201];
   char h_date[20];
   int retry;
};

struct paystruct {
/* [ADD] 1997.09.03 CREO)yamamoto */
/*   Kind of transactions */
   int tran_kind;
/* [END] */

   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 {
/* [ADD] 1997.09.03 CREO)yamamoto */
/*   Kind of transactions */
   int tran_kind;
/* [END] */

   struct ordinstruct ordin;
   struct ordoutstruct ordout;
```

```
};


/* Delivery */

struct delinstruct {
  int w_id;
  int o_carrier_id;
/* update 1997.09.03 CREO)yamamoto
  double qtime;
*/
  long qtime;
  long uqtime;

  int in_timing_int;
};

struct deloutstruct {
  int terror;
  int retry;
};

struct delstruct {
/* [ADD] 1997.09.03 CREO)yamamoto */
/*   Kind of transactions */
  int tran_kind;
/* [END] */

  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 {
/* [ADD] 1997.09.03 CREO)yamamoto */
/*   Kind of transactions */
  int tran_kind;
/* [END] */

  struct stoinstruct stoin;
  struct stooutstruct stoout;
};

#endif
```

tpccpl.h

```
/*

 * $Header: tpccpl.h 7030100.1 96/04/02 18:03:35 plai
Generic<base> $ Copyr (c) 1994 Oracle

 */

/*=========================================
========================+
|       Copyright (c) 1994  Oracle Corp, Redwood
Shores, CA       |
|                 OPEN SYSTEMS PERFORMANCE
GROUP                |
|                   All Rights Reserved                    |

+=========================================
========================+
| FILENAME
|    tpccpl.h
| DESCRIPTION
|    Header file for TPC-C transactions in PL/SQL.

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

#ifndef TPCCPL_H
#define TPCCPL_H

#include <stdio.h>
#include "tpcc.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 int ocierror(char *fname, int lineno,OCIError
*errhp, sword status);
extern int sqlfile(char *fname, text *linebuf);

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

extern int execstatus;
extern int errcode;

extern OCIEnv *tpcenv;
extern OCIServer *tpcsrv;
extern OCIError *errhp;
extern OCISvcCtx *tpcsvc;
extern OCISession *tpcusr;
extern OCIStmt *curn, *curn1, *curn2, *curn3[10],
*curn4;
extern OCIStmt *curntest;
/* The bind and define handles for each transaction are
   included in their respective header files. */
extern ldadef tpclda;
extern csrdef curs;
extern csrdef curd;
extern csrdef curo2;
extern csrdef curo0;
extern csrdef curo1;
/*extern csrdef curp0;
extern csrdef curp1;
*/
/* 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 int ol_amount[15];
extern char ol_delivery_d[15][11];

/* for payment transaction */

extern int c_w_id;
extern int c_d_id;
extern int h_amount;
extern char w_street_1[21];
extern char w_street_2[21];
extern char w_city[21];
extern char w_state[3];
extern char w_zip[10];
extern char d_street_1[21];
extern char d_street_2[21];
extern char d_city[21];
extern char d_state[3];
extern char d_zip[10];
extern char c_street_1[21];
extern char c_street_2[21];
extern char c_city[21];
extern char c_state[3];
extern char c_zip[10];
extern char c_phone[17];
extern char c_since_d[11];
extern char c_credit[3];
extern int c_credit_lim;
```

```
extern int 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 int nol_quanti10[15];
extern int nol_quanti91[15];
extern int nol_ytdqty[15];
extern int nol_amount[15];
extern int o_all_local;
extern int w_tax;
extern int d_tax;
extern float total_amount;
extern char i_name[15][25];
extern int i_name_strlen[15];
extern ub2 i_name_strlen_len[15];
extern ub2 i_name_strlen_rcode[15];
extern ub4 i_name_strlen_csize;
extern int s_quantity[15];
extern char brand_gen[15];
extern ub2 brand_gen_len[15];
extern ub2 brand_gen_rcode[15];
extern ub4 brand_gen_csize;
extern int i_price[15];
extern int status;

/* Miscellaneous */
extern unsigned char cr_date[7];
extern unsigned char c_since[7];
extern unsigned char o_entry_d_base[7];
extern unsigned char ol_d_base[15][7];

#ifndef DISCARD
# define DISCARD (void)
#endif

#ifndef sword
# define sword int
#endif

#define VER7          2

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

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

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

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

#define OCIERROR(errp,function)\
      ocierror(__FILE__,__LINE__,(errp),(function));

#define OCIBND(stmp, bndp, errp, sqlvar, progv,
progvl, ftype)\
     ocierror(__FILE__,__LINE__,(errp), \

OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_
BIND,0,(dvoid**)0)); \
     ocierror(__FILE__,__LINE__, (errp), \
        OCIBindByName((stmp), &(bndp), (errp), \
                      (text *)(sqlvar),
strlen((sqlvar)),\
                      (progv), (progvl),
(ftype),0,0,0,0,0,OCI_DEFAULT));

#define
OCIBNDRA(stmp,bndp,errp,sqlvar,progv,progvl,ftype,in
dp,alen,arcode) \
     ocierror(__FILE__,__LINE__,(errp), \

OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_
BIND,0,(dvoid**)0)); \
     ocierror(__FILE__,__LINE__,(errp), \
        OCIBindByName((stmp),&(bndp),(errp),(text
*)(sqlvar),strlen((sqlvar)),\

(progv),(progvl),(ftype),(indp),(alen),(arcode),0,0,OCI_D
EFAULT));

#define
OCIBNDRAD(stmp,bndp,errp,sqlvar,progvl,ftype,indp,ct
xp,cbf_nodata,cbf_data) \
     ocierror(__FILE__,__LINE__,(errp), \

OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_
BIND,0,(dvoid**)0)); \
     ocierror(__FILE__,__LINE__,(errp), \
        OCIBindByName((stmp),&(bndp),(errp),(text
*)(sqlvar), \
                  strlen((sqlvar)),0,(progvl),(ftype), \
                  indp,0,0,0,0,OCI_DATA_AT_EXEC)); \
     ocierror(__FILE__,__LINE__,(errp), \

OCIBindDynamic((bndp),(errp),(ctxp),(cbf_nodata),(ctxp
),(cbf_data)));



#define
OCIBNDR(stmp,bndp,errp,sqlvar,progv,progvl,ftype,ind
p,alen,arcode) \
     ocierror(__FILE__,__LINE__,(errp), \

OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_
BIND,0,(dvoid**)0)); \
     ocierror(__FILE__,__LINE__,(errp), \
       OCIBindByName((stmp),&(bndp),(errp),(text
*)(sqlvar),strlen((sqlvar)),\

(progv),(progvl),(ftype),(indp),(alen),(arcode),0,0,OCI_D
EFAULT));

/* Fix c.13 */
/* Fix c.14 */
#define
OCIBNDRAA(stmp,bndp,errp,sqlvar,progv,progvl,ftype,i
ndp,alen,arcode,ms,cu) \
     ocierror(__FILE__,__LINE__, (errp), \

OCIHandleAlloc((stmp),&(bndp),OCI_HTYPE_BIND,0,(
dvoid**)0)); \
     ocierror(__FILE__,__LINE__,(errp),\
       OCIBindByName((stmp),&(bndp),(errp),(text
*)(sqlvar),strlen((sqlvar)),\

(progv),(progvl),(ftype),(indp),(alen),(arcode),(ms),(cu),
OCI_DEFAULT));

#define
OCIDEFINE(stmp,dfnp,errp,pos,progv,progvl,ftype)\

OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progv),(pr
ogvl),(ftype),\
                              0,0,0,OCI_DEFAULT);


#define OCIDEF(stmp,dfnp,errp,pos,progv,progvl,ftype)
\

OCIHandleAlloc((stmp),(dvoid**)&(dfnp),OCI_HTYPE_
DEFINE,0,\

(dvoid**)0);\

OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progv),(pr
ogvl),\

(ftype),NULL,NULL,NULL,OCI_DEFAULT); \


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

OCIHandleAlloc((stmp),(dvoid**)&(dfnp),OCI_HTYPE_
DEFINE,0,\

(dvoid**)0);\

OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progv),\

(progvl),(ftype),(indp),(alen),\

(arcode),OCI_DEFAULT);\
/*
     OCIDefineArrayOfStruct((dfnp),(errp),(progvl),\

                  sizeof((indp)[0]),\

                  sizeof((alen)[0]),\

                  sizeof((arcode)[0]));
*/
/*
#define
OCIDFNRA(stmp,dfnp,errp,pos,progv,progvl,ftype,indp,
alen,arcode) \
     ocierror(__FILE__,__LINE__,(errp), \

OCIHandleAlloc(tpcenv,&(dfnp),OCI_HTYPE_DEFINE,
0,\

(dvoid**)0));\
     ocierror(__FILE__,__LINE__,(errp), \
```

```
OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progv),\

(progvl),(ftype),(indp),(alen),\

(arcode),OCI_DEFAULT));\
      ocierror(__FILE__,__LINE__,(errp),\

OCIDefineArrayOfStruct((dfnp),(errp),(progvl),\

                  sizeof((indp)[0]),\

                  sizeof((alen)[0]),\

                  sizeof((arcode)[0])));
*/

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

#define
OBNDRA(lda,cursor,sqlvar,progv,progvl,ftype,indp,alen,
arcode)\
   if
(obndra((cursor),(text*)(sqlvar),NA,(ub1*)(progv),(progvl
),(ftype),NA,\

(indp),(alen),(arcode),(ub4)0,(ub4*)0,(text*)0,NA,NA))\
      {errrpt(lda,cursor);return(-1);}\
   else\
      DISCARD 0

#define
OBNDRAA(lda,cursor,sqlvar,progv,progvl,ftype,indp,ale
n,arcode,ms,cs)\
   if
(obndra((cursor),(text*)(sqlvar),NA,(ub1*)(progv),(progvl
),(ftype),NA,\

(indp),(alen),(arcode),(ub4)(ms),(ub4*)(cs),(text*)0,NA,N
A))\
      {errrpt(lda,cursor);return(-1);}\
   else\
      DISCARD 0

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

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

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

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

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

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


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


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

#endif
```

C Source Files
pldel.c

```
#ifdef RCSID

static char *RCSid =

  "$Header: pldel.c 7030100.5 96/06/24 16:26:06 plai
Generic<base> $ Copyr (c) 1994 Oracle";

#endif /* RCSID */



/*=========================================
========================+
|       Copyright (c) 1996  Oracle Corp, Redwood
Shores, CA       |
|                OPEN SYSTEMS PERFORMANCE
GROUP              |
|                    All Rights Reserved                    |

+=========================================
========================+
| FILENAME
|    pldel.c
| DESCRIPTION
|    OCI version of DELIVERY transaction in TPC-C
benchmark.

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

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


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

#define SQLTXT1 "\
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 1, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 1 AND
o_w_id = :w_id AND o_d_id = 1 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 2, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 2 AND
o_w_id = :w_id AND o_d_id = 2 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 3, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 3 AND
o_w_id = :w_id AND o_d_id = 3 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 4, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 4 AND
o_w_id = :w_id AND o_d_id = 4 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 5, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 5 AND
o_w_id = :w_id AND o_d_id = 5 AND \
   o_id = no_o_id AND rownum <= 1 UNION ALL \
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 6, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
```

```
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 6 AND
o_w_id = :w_id AND o_d_id = 6 AND \
  o_id = no_o_id AND rownum <= 1 UNION ALL \
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 7, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 7 AND
o_w_id = :w_id AND o_d_id = 7 AND \
  o_id = no_o_id AND rownum <= 1 UNION ALL \
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 8, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 8 AND
o_w_id = :w_id AND o_d_id = 8 AND \
  o_id = no_o_id AND rownum <= 1 UNION ALL \
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 9, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 9 AND
o_w_id = :w_id AND o_d_id = 9 AND \
  o_id = no_o_id AND rownum <= 1 UNION ALL \
SELECT /*+ USE_NL(NEW_ORDER ORDERS)
ORDERED */ 10, no_o_id, new_order.rowid, o_c_id,
orders.rowid \
FROM new_order, orders \
WHERE no_w_id = :w_id AND no_d_id = 10 AND
o_w_id = :w_id AND o_d_id = 10 AND \
  o_id = no_o_id AND rownum <= 1"

#define SQLTXTTEST "INSERT INTO vv VALUES
(:no_rowid, :o_rowid)"
#define SQLTXT2 "DELETE FROM new_order WHERE
rowid = :no_rowid"
/*
#define SQLTXT2 "DELETE FROM new_order WHERE
no_o_id = :o_id and no_d_id = :d_id and \
            no_w_id = :w_id"
*/
#define SQLTXT3 "UPDATE orders SET o_carrier_id =
:carrier_id \
  WHERE rowid = :o_rowid"

/*
#define SQLTXT3 "UPDATE orders SET o_carrier_id =
:carrier_id \
            WHERE o_id = :o_id and o_d_id = :d_id and
o_w_id = :w_id \
            and o_c_id = :c_id"

*/
#define SQLTXT4 "UPDATE order_line SET
ol_delivery_d = :cr_date \
  WHERE ol_w_id = :w_id AND ol_d_id = :d_id AND
ol_o_id = :o_id"

#define SQLTXT5 "\
SELECT :d_id1, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id1 AND ol_o_id = :o_id1 UNION ALL \
SELECT :d_id2, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id2 AND ol_o_id = :o_id2 UNION ALL \
SELECT :d_id3, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id3 AND ol_o_id = :o_id3 UNION ALL \
SELECT :d_id4, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id4 AND ol_o_id = :o_id4 UNION ALL \
SELECT :d_id5, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id5 AND ol_o_id = :o_id5 UNION ALL \
SELECT :d_id6, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id6 AND ol_o_id = :o_id6 UNION ALL \
SELECT :d_id7, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id7 AND ol_o_id = :o_id7 UNION ALL \
SELECT :d_id8, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id8 AND ol_o_id = :o_id8 UNION ALL \
SELECT :d_id9, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id9 AND ol_o_id = :o_id9 UNION ALL \
SELECT :d_id10, SUM(ol_amount) FROM order_line
WHERE ol_w_id = :w_id AND \
  ol_d_id = :d_id10 AND ol_o_id = :o_id10"

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

#define NDISTS 10
#define ROWIDLEN 20


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

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

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

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

  OCIBind *w_id_bp;
  OCIBind *d_id_bp;
  OCIBind *o_id_bp;
  OCIBind *cr_date_bp;
  OCIBind *c_id_bp;
  OCIBind *no_rowid_bp;
  OCIBind *carrier_id_bp;
  OCIBind *o_rowid_bp;
  OCIBind *del_o_id_bp;
  OCIBind *amt_bp;
  OCIBind *bstr1_bp[10];
  OCIBind *bstr2_bp[10];
  OCIDefine *inum_dp;
  OCIDefine *d_id_dp;
  OCIDefine *del_o_id_dp;
  OCIDefine *no_rowid_dp;
  OCIDefine *c_id_dp;
  OCIDefine *o_rowid_dp;
  OCIDefine *cons_dp;
  OCIDefine *amt_dp;

  int norow;
};

typedef struct delctx delctx;

delctx *dctx;
```

```
pldelinit ()

{

  int i,j;
  char bstr1[10];
  char bstr2[10];
  text stmbuf[4096];

  dctx = (delctx *) malloc (sizeof(delctx));
  memset(dctx,(char)0,sizeof(delctx));
  dctx->norow = 0;
  for(i=0;i<NDISTS;i++) {
/*
    dctx->o_rowid_ptr[i] = &(dctx->o_rowid[i][0]);
    dctx->no_rowid_ptr[i] = &(dctx->no_rowid[i][0]);
*/
   OCIERROR(errhp,
OCIDescriptorAlloc(tpcenv,(dvoid**)&dctx->o_rowid_ptr[i],
          OCI_DTYPE_ROWID,0,(dvoid**)0));
   OCIERROR(errhp,
OCIDescriptorAlloc(tpcenv,(dvoid**)&dctx->no_rowid_ptr[i],
          OCI_DTYPE_ROWID,0,(dvoid**)0));
  }

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

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

  /* open first cursor */

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

  OCIERROR(errhp,
    OCIAttrSet(dctx->curd1,OCI_HTYPE_STM,(dvoid*)&dctx->norow,0,
           OCI_ATTR_PREFETCH_ROWS,errhp));



  /* bind variables */

  OCIBND(dctx->curd1, dctx->w_id_bp,errhp,":w_id",ADR(w_id),SIZ(w_id),SQLT_INT);

  OCIDFNRA(dctx->curd1, dctx->d_id_dp,errhp,1,dctx->d_id,SIZ(dctx->d_id[0]),SQLT_INT,
                dctx->d_id_ind,dctx->d_id_len,dctx->d_id_rcode);
  OCIDFNRA(dctx->curd1, dctx->del_o_id_dp,errhp,2,dctx->del_o_id,SIZ(dctx->del_o_id[0]),
                SQLT_INT,dctx->del_o_id_ind,dctx->del_o_id_len,
                dctx->del_o_id_rcode);
  OCIDFNRA(dctx->curd1, dctx->no_rowid_dp,errhp,3,dctx->no_rowid_ptr,
                sizeof(dctx->no_rowid_ptr[0]),
SQLT_RDD,dctx->no_rowid_ind,
         dctx->no_rowid_len, dctx->no_rowid_rcode);
  OCIDFNRA(dctx->curd1, dctx->c_id_dp,errhp,4,dctx->c_id,SIZ(dctx->c_id[0]),SQLT_INT,
                dctx->c_id_ind,dctx->c_id_len,dctx->c_id_rcode);
  OCIDFNRA(dctx->curd1, dctx->o_rowid_dp,errhp,5,dctx->o_rowid_ptr,
         sizeof(dctx->o_rowid_ptr[0]), SQLT_RDD,dctx->o_rowid_ind,
         dctx->o_rowid_len, dctx->o_rowid_rcode);

  /* open test cursor */
/*  OCIHandleAlloc(tpcenv, (dvoid **)(&dctx->curdtest),
OCI_HTYPE_STMT, 0, (dvoid**)0);
  sprintf ((char *) stmbuf, SQLTXTTEST);
  OCIStmtPrepare(dctx->curdtest, errhp, stmbuf,
strlen((char *)stmbuf),OCI_NTV_SYNTAX,
OCI_DEFAULT);

  OCIBNDRA(dctx->curdtest,dctx->no_rowid_bp,errhp,":no_rowid",&(dctx->no_rowid_ptr[0]),
                SIZ(dctx->no_rowid_ptr[0]),SQLT_RDD,dctx->no_rowid_ind,
                dctx->no_rowid_len,dctx->no_rowid_rcode);
  OCIBNDRA(dctx->curdtest,dctx->o_rowid_bp,errhp,":o_rowid",&(dctx->o_rowid_ptr[0]),
                SIZ(dctx->o_rowid_ptr[0]),SQLT_RDD,dctx->o_rowid_ind,
                dctx->o_rowid_len,dctx->o_rowid_rcode);
*/
  /* open second cursor */

  OCIHandleAlloc(tpcenv, (dvoid **)(&dctx->curd2),
OCI_HTYPE_STMT, 0, (dvoid**)0);
  sprintf ((char *) stmbuf, SQLTXT2);
  OCIStmtPrepare(dctx->curd2, errhp, stmbuf,
strlen((char *)stmbuf),
                          OCI_NTV_SYNTAX,
OCI_DEFAULT);

  /* bind variables */
  OCIBNDRA(dctx->curd2, dctx->no_rowid_bp,errhp,":no_rowid",&(dctx->no_rowid_ptr[0]),
                SIZ(dctx->no_rowid_ptr[0]),SQLT_RDD,dctx->no_rowid_ind,
                dctx->no_rowid_len,dctx->no_rowid_rcode);
/*
  OCIBNDRA(dctx->curd2, dctx->w_id_bp, errhp,
":w_id", dctx->w_id,SIZ(dctx->w_id[0]),
                SQLT_INT,dctx->w_id_ind,dctx->w_id_len,dctx->w_id_rcode);
  OCIBNDRA(dctx->curd2, dctx->d_id_bp, errhp,
":d_id", dctx->d_id,SIZ(dctx->d_id[0]),
                SQLT_INT,dctx->d_id_ind,dctx->d_id_len,dctx->d_id_rcode);
  OCIBNDRA(dctx->curd2, dctx->del_o_id_bp, errhp,
":o_id", dctx->del_o_id,
                SIZ(dctx->del_o_id[0] ),
                SQLT_INT,dctx->del_o_id_ind,dctx->del_o_id_len,dctx->del_o_id_rcode);
*/
  /* open third cursor */

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


  /* bind variables */

  OCIBNDRA(dctx->curd3, dctx->carrier_id_bp,errhp,":carrier_id",dctx->carrier_id,
       SIZ(dctx->carrier_id[0]),SQLT_INT,dctx->carrier_id_ind,
       dctx->carrier_id_len,dctx->carrier_id_rcode);
/*
  OCIBNDRA(dctx->curd3, dctx->w_id_bp, errhp,
":w_id", dctx->w_id,SIZ(dctx->w_id[0]),
                SQLT_INT,dctx->w_id_ind,dctx->w_id_len,dctx->w_id_rcode);
  OCIBNDRA(dctx->curd3, dctx->d_id_bp, errhp,
":d_id", dctx->d_id,SIZ(dctx->d_id[0]),
                SQLT_INT,dctx->d_id_ind,dctx->d_id_len,dctx->d_id_rcode);
  OCIBNDRA(dctx->curd3, dctx->del_o_id_bp, errhp,
":o_id", dctx->del_o_id,
                SIZ(dctx->del_o_id[0]),
                SQLT_INT,dctx->del_o_id_ind,dctx->del_o_id_len,dctx->del_o_id_rcode);
  OCIBNDRA(dctx->curd3, dctx->c_id_bp, errhp,
":c_id", dctx->c_id,SIZ(dctx->c_id[0]),
                SQLT_INT,dctx->c_id_ind,dctx->c_id_len,dctx->c_id_rcode);
*/

  OCIBNDRA(dctx->curd3, dctx->o_rowid_bp,errhp,":o_rowid",&(dctx->o_rowid_ptr[0]),
                SIZ(dctx->o_rowid_ptr[0]),SQLT_RDD,dctx->o_rowid_ind,
                dctx->o_rowid_ptr_len,dctx->o_rowid_rcode);
  /* open fourth cursor */

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

  /* bind variables */

  OCIBNDRA(dctx->curd4, dctx->w_id_bp,errhp,":w_id",dctx->w_id,SIZ(dctx->w_id[0]),
```

```
                SQLT_INT,dctx->w_id_i nd,dctx-
>w_id_len,dctx->w_id_rcode);
  OCIBNDRA(dctx->curd4, dctx-
>d_id_bp,errhp,":d_id",dctx->d_id,SIZ(dctx->d_id[0]),
                SQLT_INT,dctx->d_id_ind,dctx-
>d_id_len,dctx->d_id_rcode);
  OCIBNDRA(dctx->curd4, dctx-
>o_id_bp,errhp,":o_id",dctx->del_o_id,
                SIZ(dctx->del_o_id[0]),SQLT_INT,dctx-
>del_o_id_ind,
                dctx->del_o_id_len,dctx-
>del_o_id_rcode);
  OCIBNDRA(dctx->curd4, dctx-
>cr_date_bp,errhp,":cr_date", dctx-
>del_date,DEL_DATE_LEN,
                SQLT_DAT,dctx->del_date_ind, dctx-
>del_date_len,
                dctx->d el_date_rcode);


  /* open fifth cursor */

  OCIHandleAlloc(tpcenv, (dvoid **)(&dctx->curd5),
OCI_HTYPE_STMT, 0, (dvoid**)0);
  sprintf ((char *) stmbuf, SQLTXT5);
  OCIStmtPrepare(dctx->curd5, errhp, stmbuf,
strlen((char *)stmbuf),
                        OCI_NTV_SYNTA X,
OCI_DEFAULT);

  OCIERROR(errhp,
    OCIAttrSet(dctx-
>curd5,OCI_HTYPE_STM,(dvoid*)&dctx->norow,0,
          OCI_ATTR_PREFETCH_ROWS,errhp));

  /* bind variables */

  OCIBND(dctx->curd5,dctx-
>w_id_bp,errhp,":w_id",ADR(w_id),SIZ(w_id),SQLT_IN
T);
  for (i = 0; i < NDISTS; i++) {
    sprintf (bstr1, ":d_id%d", i + 1);
    sprintf (bstr2, ":o_id%d", i + 1);
    OCIBNDRA(dctx->curd5,dctx-
>bstr1_bp[i],errhp,bstr1,ADR(dctx->d_id[i]),
                SIZ(dctx->d_id[0]),SQLT_INT, &(dctx-
>d_id_ind[i]),
                &(dctx->d_id_len[i]),&(dctx-
>d_id_rcode[i]));
    OCIBNDRA(dctx->curd5,dctx-
>bstr2_bp[i],errhp,bstr2,ADR(dctx->del_o_id[i]),
                SIZ(dctx->del_o_id[0]),SQLT_INT,
&(dctx->del_o_id_ind[i]),
                &(dctx->del_o_id_len[i]),&(dctx-
>del_o_id_rcode[i]));
  }

  OCIDFNRA(dctx->curd5,dctx->cons_dp,errhp,1,dctx-
>cons,SIZ(dctx->cons[0]),SQLT_INT,
      dctx->cons_ind,dctx->cons_len,dctx-
>cons_rcode);
  OCIDFNRA(dctx->curd5,dctx->amt_dp,errhp,2,dctx-
>amt,SIZ(dctx->amt[0]),SQLT_INT,
      dctx->amt_ind,dctx->amt_len,dctx->amt_rcode);

  /* open sixth cursor */

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

  /* bind variables */

  OCIBNDRA(dctx->curd6,dctx-
>amt_bp,errhp,":amt",dctx->amt,SIZ(dctx-
>amt[0]),SQLT_INT,
      dctx->amt_ind,dctx->amt_len,dctx->amt_rcode);
  OCIBNDRA(dctx->curd6,dctx-
>w_id_bp,errhp,":w_id",dctx->w_id,SIZ(dctx->w_id[0]),
                SQLT_INT,dctx->w_id_ind,dctx-
>w_id_len,dctx->w_id_rcode);
  OCIBNDRA(dctx->curd6,dctx-
>d_id_bp,errhp,":d_id",dctx->d_id,SIZ(dctx->d_id[0]),
                SQLT_INT,dctx->d_id_ind,dctx-
>d_id_len,dctx->d_id_rcode);
  OCIBNDRA(dctx->curd6,dctx-
>c_id_bp,errhp,":c_id",dctx->c_id,SIZ(dctx->c_id[0]),
                SQLT_INT, dctx->c_id_ind,dctx-
>c_id_len,dctx->c_id_rcode);

  return (0);

}



pldel ()

{

  int i, j,v;
  int rpc,rcount;
  int invalid;
  int tmp_id;
  /* float tmp_amt; changed form float to int */
  int tmp_amt;

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

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

retry:

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

iso:

  invalid = 0;

  /* initialization for array operations */

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

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

    dctx->w_id[i] = w_id;
    dctx->carrier_id[i] = o_carrier_id;
    memcpy(dctx-
>del_date[i],cr_date,DEL_DATE_LEN);
  }

  /* array select from new_order and orders tables */

  execstatus=OCIStmtExecute(tpcsvc,dctx-
>curd1,errhp,NDISTS,0,0,0,OCI_DEFAULT);
  if((execstatus != OCI_SUCCESS) && (execstatus !=
OCI_NO_DATA)) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      retries++;
      goto retry;
    } else if (errcode == RECOVERR) {
      retries++;
      goto retry;
    } else {
      return -1;
    }
  }
  /* mark districts with no new order */
  OCIAttrGet(dctx-
>curd1,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_RO
WCNT,errhp);
  rpc = rcount;
  invalid = NDISTS - rcount;
  for (i = rpc; i < NDISTS; i++) {
    dctx->del_o_id_ind[i] = NA;
    dctx->w_id_ind[i] = NA;
    dctx->d_id_ind[i] = NA;
    dctx->c_id_ind[i] = NA;
    dctx->carrier_id_ind[i] = NA;
    dctx->no_rowid_ind[i] = NA;
    dctx->o_rowid_ind[i] = NA;
```

```
  }

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

  /* array delete of new_order table */
  execstatus=OCIStmtExecute(tpcsvc,dctx-
>curd2,errhp,rpc,0,0,0,OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      retries++;
      goto retry;
    } else if (errcode == RECOVERR) {
      retries++;
      goto retry;
    } else {
      return -1;
    }
  }

  /* mark districts with no new order */
  OCIAttrGet(dctx-
>curd2,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_RO
WCNT,errhp);

  if (rcount != rpc) {
#ifdef TUX
    userlog ("Error in TPC-C server %d: %d rows
selected, %d rows deleted\n",
        proc_no, rpc, dctx->curd2.rpc);
#else
    /* fprintf (stderr,
        "Error in TPC-C server %d: %d rows selected,
%d rows deleted\n",
        proc_no, rpc, dctx->curd2.rpc); */
#endif
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    return (DEL_ERROR);
  }

  execstatus=OCIStmtExecute(tpcsvc,dctx-
>curd3,errhp,rpc,0,0,0,OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      retries++;
      goto retry;
    } else if (errcode == RECOVERR) {
      retries++;
      goto retry;
    } else {
      return -1;
    }
  }

  OCIAttrGet(dctx-
>curd3,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_RO
WCNT,errhp);

  if (rcount != rpc) {
#ifdef TUX
    userlog ("Error in TPC-C server %d: %d rows
selected, %d ords updated\n",
        proc_no, rpc, rcount);
#else
    fprintf (stderr,
        "Error in TPC-C server %d: %d rows selected,
%d ords updated\n",
        proc_no, rpc, rcount);
#endif
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    return (-1);
  }

  /* array update of order_line table */
  execstatus=OCIStmtExecute(tpcsvc,dctx-
>curd4,errhp,rpc,0,0,0,OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      retries++;
      goto retry;
    } else if (errcode == RECOVERR) {
      retries++;
      goto retry;
    } else {
      return -1;
    }
  }

  /* array select from order_line table */
  execstatus=OCIStmtExecute(tpcsvc,dctx-
>curd5,errhp,rpc,0,0,0,OCI_DEFAULT);
  if((execstatus != OCI_SUCCESS) && (execstatus !=
OCI_NO_DATA)) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      retries++;
      goto retry;
    } else if (errcode == RECOVERR) {
      retries++;
      goto retry;
    } else {
      return -1;
    }
  }

  OCIAttrGet(dctx-
>curd5,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_RO
WCNT,errhp);
  if (rcount != rpc) {
#ifdef TUX
    userlog ("Error in TPC-C server %d: %d rows
selected, %d ordl selected\n",
        proc_no, rpc, rcount);
#else
    fprintf (stderr,
        "Error in TPC-C server %d: %d rows selected,
%d ordl selected\n",
        proc_no, rpc, rcount);
#endif
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    return (-1);
  }

  /* reorder amount selected if necessary */

  for (i = 0; i < rpc; i++) {
    if (dctx->cons[i] != dctx->d_id[i]) {
#ifdef TUX
      userlog ("TPC-C server %d: reordering
amount\n", proc_no);
#else
      fprintf (stderr, "TPC-C server %d: reordering
amount\n", proc_no);
#endif
      for (j = i + 1; j < rpc; j++) {
        if (dctx->cons[j] == dctx->d_id[i]) {
          tmp_id = dctx->cons[i];
          dctx->cons[i] = dctx->cons[j];
          dctx->cons[j] = tmp_id;
          tmp_amt = dctx->amt[i];
          dctx->amt[i] = dctx->amt[j];
          dctx->amt[j] = tmp_amt;
          break;
        }
      }
      if (j >= rpc) {
#ifdef TUX
        userlog ("Error in TPC-C server %d: missing
ordl?\n", proc_no);
#else
        fprintf (stderr,
            "Error in TPC-C server %d: missing
ordl?\n", proc_no);
#endif

OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
        return (-1);
      }
    }
  }

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

  /* array update of customer table */
#if defined(ISO5) || defined (ISO6)
  execstatus=OCIStmtExecute(tpcsvc,dctx-
>curd6,errhp,rpc,0,0,0,
```

```
                OCI_DEFAULT);
#else
  execstatus=OCIStmtExecute(tpcsvc,dctx-
>curd6,errhp,rpc,0,0,0,
                    OCI_COMMIT_ON_SUCCESS |
OCI_DEFAULT);
#endif

  if(execstatus != OCI_SUCCESS) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      retries++;
      goto retry;
    } else if (errcode == RECOVERR) {
      retries++;
      goto retry;
    } else {
      return -1;
    }
  }

  OCIAttrGet(dctx-
>curd6,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_RO
WCNT,errhp);

  if (rcount != rpc) {
#ifdef TUX
     userlog ("Error in TPC-C server %d: %d rows
selected, %d cust updated\n",
           proc_no, rpc, rcount);
#else
     fprintf (stderr,
           "Error in TPC-C server %d: %d rows selected,
%d cust updated\n",
           proc_no, rpc, rcount);
#endif
     OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
     return (-1);
  }

#if defined(ISO5) || defined(ISO6)
  sysdate (sdate);
#ifdef ISO5
  printf ("Delivery sleep before commit at %s\n", sdate);
#else
  printf ("Delivery sleep before abort at %s\n", sdate);
#endif
  sleep (60);
  sysdate (sdate);
  printf ("Delivery wake up at %s\n", sdate);
#endif

#ifdef ISO6
  printf("Delivery ISO6 Rolling back.\n");
  OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
#endif

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

#if defined(ISO5) || defined(ISO6)
  sysdate (sdate);
  printf ("Delivery completed at: %s\n", sdate);
#endif
  /* return o_id's in district id order */

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

  return (0);

}



void pldeldone ()

{

  if (dctx)
  {
#if defined(ISO) || defined(ISO5) || defined(ISO6) ||
defined(ISO8)
    OCIHandleFree((dvoid *)dctx-
>curd0,OCI_HTYPE_STMT);
#endif
    OCIHandleFree((dvoid *)dctx-
>curd1,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx-
>curd2,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx-
>curd3,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx-
>curd4,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx-
>curd5,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)dctx-
>curd6,OCI_HTYPE_STMT);
    free (dctx);
  }

}
```

plnew.c

```
#ifdef RCSID

static char *RCSid =

  "$Header: plnew.c 7030100.4 96/07/01 17:33:40 plai
Generic<base> $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=========================================
========================+
|      Copyright (c) 1996  Oracle Corp, Redwood
Shores, CA      |
|                OPEN SYSTEMS PERFORMANCE
GROUP               |
|                   All Rights Reserved                   |

+=========================================
========================+
| FILENAME
|   plnew.c
| DESCRIPTION
|   OCI version (using PL/SQL stored procedure) of
|   NEW ORDER transaction in TPC-C benchmark.

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

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


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

#define SQLTXT3 "\
SELECT
0,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :10 AND s_w_id = :30
AND s_i_id = i_id UNION ALL \
SELECT
1,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :11 AND s_w_id = :31
AND s_i_id = i_id UNION ALL \
SELECT
2,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :12 AND s_w_id = :32
AND s_i_id = i_id UNION ALL \
SELECT
3,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :13 AND s_w_id = :33
AND s_i_id = i_id UNION ALL \
SELECT
4,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :14 AND s_w_id = :34
AND s_i_id = i_id UNION ALL \
SELECT
5,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :15 AND s_w_id = :35
AND s_i_id = i_id UNION ALL \
SELECT
6,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :16 AND s_w_id = :36
AND s_i_id = i_id UNION ALL \
SELECT
7,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :17 AND s_w_id = :37
AND s_i_id = i_id UNION ALL \
```

```
SELECT
8,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :18 AND s_w_id = :38
AND s_i_id = i_id UNION ALL \
SELECT
9,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_dat
a,s_quantity \
FROM item,stock WHERE i_id = :19 AND s_w_id = :39
AND s_i_id = i_id UNION ALL \
SELECT
10,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_da
ta,s_quantity \
FROM item,stock WHERE i_id = :20 AND s_w_id = :40
AND s_i_id = i_id UNION ALL \
SELECT
11,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_da
ta,s_quantity \
FROM item,stock WHERE i_id = :21 AND s_w_id = :41
AND s_i_id = i_id UNION ALL \
SELECT
12,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_da
ta,s_quantity \
FROM item,stock WHERE i_id = :22 AND s_w_id = :42
AND s_i_id = i_id UNION ALL \
SELECT
13,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_da
ta,s_quantity \
FROM item,stock WHERE i_id = :23 AND s_w_id = :43
AND s_i_id = i_id UNION ALL \
SELECT
14,stock.rowid,i_price,i_name,i_data,s_dist_%02d,s_da
ta,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 \
  (ol_o_id, ol_d_id, ol_w_id, oL_number, ol_delivery_d,
ol_i_id, \
   ol_supply_w_id, ol_quantity,ol_amount,ol_dist_info)\
  VALUES (:ol_o_id, :ol_d_id, \
  :ol_w_id, :ol_number, :null_date, :ol_i_id,
:ol_supply_w_id, :ol_quantity, \
  :ol_amount, :ol_dist_info)"

#define NITEMS 15
#define ROWIDLEN 20
#define OCIROWLEN 20



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

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

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



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

  OCIRowid  *s_rowid_ptr[NITEMS];

  int s_remote[NITEMS];
  char i_data[NITEMS][51];
  char s_data[NITEMS][51];
  char s_dist_info[NITEMS][25];
  unsigned char null_date[NITEMS][7]; /* base date for
null date entry */
  OCIStmt *curn;
  OCIStmt *curn1;
  OCIStmt *curn2;
  OCIStmt *curn3[10];
  OCIStmt *curn4;
  OCIBind *w_id_bp;
  OCIBind *d_id_bp;
  OCIBind *c_id_bp;
  OCIBind *o_all_local_bp;
  OCIBind *o_all_cnt_bp;
  OCIBind *w_tax_bp;
  OCIBind *d_tax_bp;
  OCIBind *o_id_bp;
  OCIBind *c_discount_bp;
  OCIBind *c_credit_bp;
  OCIBind *c_last_bp;
  OCIBind *retries_bp;
  OCIBind *cr_date_bp;
  OCIBind *ol_i_id_bp;
  OCIBind *ol_supply_w_id_bp;
  OCIBind *s_quantity_bp;
  OCIBind *s_rowid_bp;
  OCIBind *ol_quantity_bp;
  OCIBind *s_remote_bp;
  OCIBind *id_bp[10][15];
  OCIBind *sd_bp[10][15];
  OCIDefine *Dcons[10];
  OCIDefine *Ds_rowid[10];
  OCIDefine *Di_price[10];
  OCIDefine *Di_data[10];
  OCIDefine *Ds_dist_info[10];
  OCIDefine *Ds_data[10];
  OCIDefine *Ds_quantity[10];
  OCIDefine *Di_name[10];
  OCIBind *ol_o_id_bp;
  OCIBind *ol_d_id_bp;
  OCIBind *ol_w_id_bp;
  OCIBind *ol_number_bp;
  OCIBind *ol_amount_bp;
  OCIBind *ol_dist_info_bp;
  OCIBind *null_date_bp;
  sb2 w_id_ind;
  ub2 w_id_len;
  ub2 w_id_rc;

  sb2 d_id_ind;
  ub2 d_id_len;
  ub2 d_id_rc;

  sb2 c_id_ind;
  ub2 c_id_len;
  ub2 c_id_rc;

  sb2 o_all_local_ind;
  ub2 o_all_local_len;
  ub2 o_all_local_rc;

  sb2 o_ol_cnt_ind;
  ub2 o_ol_cnt_len;
  ub2 o_ol_cnt_rc;

  sb2 w_tax_ind;
  ub2 w_tax_len;
  ub2 w_tax_rc;

  sb2 d_tax_ind;
  ub2 d_tax_len;
  ub2 d_tax_rc;

  sb2 o_id_ind;
  ub2 o_id_len;
  ub2 o_id_rc;
```

```
  sb2 c_discount_ind;
  ub2 c_discount_len;
  ub2 c_discount_rc;

  sb2 c_credit_ind;
  ub2 c_credit_len;
  ub2 c_credit_rc;

  sb2 c_last_ind;
  ub2 c_last_len;
  ub2 c_last_rc;

  sb2 retries_ind;
  ub2 retries_len;
  ub2 retries_rc;

  sb2 cr_date_ind;
  ub2 cr_date_len;
  ub2 cr_date_rc;


  int cs;
  int norow;
};

typedef struct newctx newctx;

newctx *nctx;


plnewinit ()
{

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

  nctx = (newctx *) malloc (sizeof(newctx));
  memset(nctx,(char)0,sizeof(newctx));
  nctx->cs = 1;
  nctx->norow=0;
  for(i=0;i<NITEMS;i++) {
    OCIERROR(errhp,
OCIDescriptorAlloc(tpcenv,(dvoid**)&nctx-
>s_rowid_ptr[i],
      OCI_DTYPE_ROWID,0,(dvoid**)0));
  }
  nctx->w_id_ind = TRUE;
  nctx->w_id_len = sizeof(w_id);
  nctx->d_id_ind = TRUE;
  nctx->d_id_len = sizeof(d_id);
  nctx->c_id_ind = TRUE;
  nctx->c_id_len = sizeof(c_id);
  nctx->o_all_local_ind = TRUE;
  nctx->o_all_local_len = sizeof(o_all_local);
  nctx->o_ol_cnt_ind = TRUE;
  nctx->o_ol_cnt_len = sizeof(o_ol_cnt);
  nctx->w_tax_ind = TRUE;
  nctx->w_tax_len = 0;
  nctx->d_tax_ind = TRUE;
  nctx->d_tax_len = 0;
  nctx->o_id_ind = TRUE;
  nctx->o_id_len = sizeof(o_id);
  nctx->c_discount_ind = TRUE;
  nctx->c_discount_len = 0;
  nctx->c_credit_ind = TRUE;
  nctx->c_credit_len = 0;
/*  nctx->c_credit_len = 0; */
  nctx->c_last_ind = TRUE;
  nctx->c_last_len = 0;
  nctx->retries_ind = TRUE;
  nctx->retries_len = sizeof(retries);
  nctx->cr_date_ind = TRUE;
  nctx->cr_date_len = sizeof(cr_date);

  /* open first cursor */
  OCIERROR(errhp,OCIHandleAlloc(tpcenv,(dvoid
**)(&nctx->curn1),
        OCI_HTYPE_STMT, 0, (dvoid**)0));
  sqlfile("../blocks/new.sql",stmbuf);
  OCIERROR(errhp,OCIStmtPrepare(nctx->curn1,
errhp, stmbuf, strlen((char *)stmbuf),

                OCI_NTV_SYNTAX, OCI_DEFAULT));

  /* bind variables */

  OCIBNDR(nctx->curn1, nctx->w_id_bp, errhp,
":w_id",ADR(w_id),SIZ(w_id),
        SQLT_INT, &nctx->w_id_ind, &nctx->w_id_len,
&nctx->w_id_rc);
  OCIBNDR(nctx->curn1, nctx->d_id_bp, errhp,
":d_id",ADR(d_id),SIZ(d_id),
        SQLT_INT, &nctx->d_id_ind, &nctx->d_id_len,
&nctx->d_id_rc);
  OCIBNDR(nctx->curn1, nctx->c_id_bp, errhp,
":c_id",ADR(c_id),SIZ(c_id),
        SQLT_INT, &nctx->c_id_ind, &nctx->c_id_len,
&nctx->c_id_rc);
  OCIBNDR(nctx->curn1, nctx->o_all_local_bp, errhp,
":o_all_local",
        ADR(o_all_local), SIZ(o_all_local),SQLT_INT,
&nctx->o_all_local_ind,
        &nctx->o_all_local_len, &nctx->o_all_local_rc);
  OCIBNDR(nctx->curn1, nctx->o_all_cnt_bp, errhp,
":o_ol_cnt",ADR(o_ol_cnt),
                SIZ(o_ol_cnt),SQLT_INT,
                &nctx->o_ol_cnt_ind, &nctx-
>o_ol_cnt_len, &nctx->o_ol_cnt_rc);
  OCIBNDR(nctx->curn1, nctx->w_tax_bp, errhp,
":w_tax",ADR(w_tax),SIZ(w_tax),
        SQLT_INT, &nctx->w_tax_ind, &nctx-
>w_tax_len, &nctx->w_tax_rc);
  OCIBNDR(nctx->curn1, nctx->d_tax_bp, errhp,
":d_tax",ADR(d_tax),SIZ(d_tax),
        SQLT_INT, &nctx->d_tax_ind, &nctx->d_tax_len,
&nctx->d_tax_rc);
  OCIBNDR(nctx->curn1, nctx->o_id_bp, errhp,
":o_id",ADR(o_id),SIZ(o_id),
        SQLT_INT, &nctx->o_id_ind, &nctx->o_id_len,
&nctx->o_id_rc);
  OCIBNDR(nctx->curn1, nctx->c_discount_bp, errhp,
":c_discount",
        ADR(c_discount), SIZ(c_discount),SQLT_INT,
                &nctx->c_discount_ind, &nctx-
>c_discount_len, &nctx->c_discount_rc);
  OCIBNDR(nctx->curn1, nctx->c_credit_bp, errhp,
":c_credit",c_credit,
        SIZ(c_credit),SQLT_CHR,
                &nctx->c_credit_ind, &n ctx->c_credit_len,
&nctx->c_credit_rc);
  OCIBNDR(nctx->curn1, nctx->c_last_bp, errhp,
":c_last",c_last,SIZ(c_last),
        SQLT_STR, &nctx->c_last_ind, &nctx-
>c_last_len, &nctx->c_last_rc);
  OCIBNDR(nctx->curn1, nctx->retries_bp, errhp,
":retry",ADR(retries),
        SIZ(retries),SQLT_INT,
                &nctx->retries_ind, &nctx->retries_len,
&nctx->retries_rc);
  OCIBNDR(nctx->curn1, nctx->cr_date_bp, errhp,
":cr_date",cr_date,SIZ(cr_date),        SQLT_DAT,
&nctx->cr_date_ind, &nctx->cr_date_len, &nctx-
>cr_date_rc);

  /* open second cursor */
  OCIERROR(errhp,OCIHandleAlloc(tpcenv, (dvoid
**)(&nctx->curn2), OCI_HTYPE_STMT,
                                                0,
(dvoid**)0));
  sprintf ((char *) stmbuf, SQLTXT2);
  OCIERROR(errhp,OCIStmtPrepare(nctx->curn2,
errhp, stmbuf,
        strlen((char *)stmbuf), OCI_NTV_SYNTAX,
OCI_DEFAULT));



  /* bind variables */

#ifdef OPS
  OCIBNDRA(nctx->curn2, nctx-
>ol_i_id_bp,errhp,":ol_i_id",nol_i_id,
        SIZ(int), SQLT_INT, nctx->nol_i_id_ind,nctx-
>nol_i_id_len,
        nctx->nol_i_id_rcode);
  OCIBNDRA(nctx->curn2, nctx->ol_supply_w_id_bp,
errhp, ":ol_supply_w_id",
                nol_supply_w_id,SIZ(int),SQLT_INT,
nctx->nol_supply_w_id_ind,
        nctx->nol_supply_w_id_len, nctx-
>nol_supply_w_id_rcode);
#else
  OCIBNDRA(nctx->curn2, nctx-
>s_quantity_bp,errhp,":s_quantity",s_quantity,
        SIZ(int), SQLT_INT,nctx->s_quant_ind,nctx-
>s_quant_len,
        nctx->s_quant_rcode);
  OCIBNDRA(nctx->curn2, nctx->s_rowid_bp, errhp,
":s_rowid",nctx->s_rowid_ptr,
        sizeof(nctx->s_rowid_ptr[0]),SQLT_RDD,nctx-
>s_rowid_ind,
                nctx->s_rowid_len,nctx->s_rowid_rcode);
#endif
  OCIBNDRA(nctx->curn2, nctx-
>ol_quantity_bp,errhp,":ol_quantity",nol_quantity,
                SIZ(int),SQLT_INT,nctx-
>nol_quantity_ind,nctx->nol_quantity_len,
                nctx->nol_quantity_r code);
  OCIBNDRA(nctx->curn2, nctx->s_remote_bp, errhp,
":s_remote",nctx->s_remote,
        SIZ(int), SQLT_INT,nctx->s_remote_ind,nctx-
>s_remote_len,
                nctx->s_remote_rcode);

  /* open third cursor and bind variables */

  for (i = 0; i < 10; i++)
  {
    j = i + 1;
    OCIERROR(errhp,OCIHandleAlloc(tpcenv, (dvoid
**)&((nctx->curn3)[i]),
```

```
OCI_HTYPE_STMT, 0, (dvoid**)0));
   sprintf ((char *) stmbuf, SQLTXT3, j, j, j, j, j, j, j, j, j, j,
j, j,
                    j, j, j);
   OCIERROR(errhp,OCIStmtPrepare((nctx->curn3)[i],
errhp, stmbuf,
                          strlen((char
*)stmbuf),OCI_NTV_SYNTAX,

OCI_DEFAULT));
   OCIERROR(errhp,
         OCIAttrSet(nctx-
>curn3[i],OCI_HTYPE_STM,(dvoid*)&nctx->norow,0,
              OCI_ATTR_PREFETCH_ROWS,errhp));
   for (j = 0; j < NITEMS; j++)
   {
     sprintf (id, ":%d", j + 10);
     sprintf (sd, ":%d", j + 30);
     OCIBNDRA((nctx->curn3)[i],(nctx-
>id_bp)[i][j],errhp,id,ADR(nol_i_id[j]),
                      SIZ(int),SQLT_INT,
                      &nctx->nol_i_i d_ind[j],&nctx-
>nol_i_id_len[j],
                      &nctx->nol_i_id_rcode[j]);
     OCIBNDRA((nctx->curn3)[i],(nctx-
>sd_bp)[i][j],errhp,sd,

ADR(nol_supply_w_id[j]),SIZ(int),SQLT_INT,
                      &nctx->nol_supply_w_id_ind[j],&nctx-
>nol_supply_w_id_len[j],
                      &nctx->nol_supply_w_id_rcode[j]);
     nctx->nol_i_id_ind[j] = NA;
     nctx->nol_supply_w_id_ind[j] = NA;
     nctx->nol_i_id_len[j] = sizeof(int);
     nctx->nol_supply_w_id_len[j] = sizeof(int);
   }

   OCIDFNRA((nctx->curn3)[i],(nctx-
>Dcons)[i],errhp,1,&(nctx->cons[0]),
                    SIZ(nctx->cons[0]),SQLT_INT,
                    nctx->cons_ind,nctx->cons_len, nctx-
>cons_rcode);
   OCIDFNRA((nctx->curn3)[i], (nctx-
>Ds_rowid)[i],errhp,2,
           nctx->s_rowid_ptr,
           sizeof(nctx->s_rowid_ptr[0]),
                    SQLT_RDD,nctx->s_rowid_ind,nctx-
>s_rowid_len,
                    nctx->s_rowid_rcode);
   OCIDFNRA((nctx->curn3)[i], (nctx-
>Di_price)[i],errhp,3,i_price,SIZ(int),
           SQLT_INT, nctx->i_price_ind,nctx-
>i_price_len,nctx->i_price_rcode);

   OCIDFNRA((nctx->curn3)[i], (nctx-
>Di_name)[i],errhp,4,i_name,
           SIZ(i_name[0]),SQLT_STR, nctx-
>i_name_ind,nctx->i_name_len,
           nctx->i_name_rcode);
   OCIDFNRA((nctx->curn3)[i], (nctx-
>Di_data)[i],errhp,5,nctx->i_data,
                    SIZ(nctx->i_data[0]),
                    SQLT_STR,nctx->i_data_ind,nctx-
>i_data_len,nctx->i_data_rcode);
   OCIDFNRA((nctx->curn3)[i], (nctx-
>Ds_dist_info)[i],errhp,6,
           nctx->s_dist_info, SIZ(nctx-
>s_dist_info[0]),SQLT_STR,
           nctx->s_dist_info_ind, nctx->s_dist_info_len,
           nctx->s_dist_info_rcode);
   OCIDFNRA((nctx->curn3)[i],(nctx-
>Ds_data)[i],errhp,7,nctx->s_data,
                    SIZ(nctx->s_data[0]),SQLT_STR,nctx-
>s_data_ind,
                    nctx->s_data_len,nctx->s_data_rcode);
   OCIDFNRA((nctx->curn3)[i],(nctx-
>Ds_quantity)[i],errhp,8,s_quantity,
                    SIZ(int),SQLT_INT, nctx-
>s_quantity_ind,nctx->s_quantity_len,
                    nctx->s_quantity_rcode);


  }

  /* open fourth cursor */
  OCIHandleAlloc(tpcenv, (dvoid **)(&nctx->curn4),
OCI_HTYPE_STMT, 0,
              (dvoid**)0);
  sprintf ((char *) stmbuf, SQLTXT4);
  OCIStmtPrepare(nctx->curn4, errhp, stmbuf,
strlen((char *)stmbuf),
                                OCI_NTV_SYNTAX,
OCI_DEFAULT);
  /* bind variables */

  OCIBNDRA(nctx->curn4, nctx-
>ol_o_id_bp,errhp,":ol_o_id",nctx->ol_o_id,
         SIZ(int),SQLT_INT, nctx->ol_o_id_ind,nctx-
>ol_o_id_len,
         nctx->ol_o_id_rcode);
  OCIBNDRA(nctx->curn4, nctx-
>ol_d_id_bp,errhp,":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);
  OCIBNDRA(nctx->curn4, nctx-
>ol_w_id_bp,errhp,":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);
  OCIBNDRA(nctx->curn4, nctx-
>ol_number_bp,errhp,":ol_number",nctx->ol_number,
         SIZ(int),SQLT_INT, nctx->ol_number_ind,nctx-
>ol_number_len,
         nctx->ol_number_rcode);
  OCIBNDRA(nctx->curn4, nctx-
>ol_i_id_bp,errhp,":ol_i_id",nol_i_id,SIZ(int),
         SQLT_INT, nctx->nol_i_id_ind,nctx-
>nol_i_id_len,
         nctx->nol_i_id_rcode);
  OCIBNDRA(nctx->curn4, nctx-
>ol_supply_w_id_bp,errhp,":ol_supply_w_id",
         nol_supply_w_id,SIZ(int),SQLT_INT, nctx-
>nol_supply_w_id_ind,
         nctx->nol_supply_w_id_len, nctx-
>nol_supply_w_id_rcode);
  OCIBNDRA(nctx->curn4, nctx-
>ol_quantity_bp,errhp,":ol_quantity",nol_quantity,
         SIZ(int),SQLT_INT, nctx->nol_quantity_ind,nctx-
>nol_quantity_len,
                  nctx->nol_quantity_rcode);
  OCIBNDRA(nctx->curn4, nctx-
>ol_amount_bp,errhp,":ol_amount",nol_amount,
         SIZ(int),SQLT_INT, nctx->nol_amount_ind,nctx-
>nol_amount_len,
         nctx->nol_amount_rcode);
  OCIBNDRA(nctx->curn4, nctx-
>ol_dist_info_bp,errhp,":ol_dist_info",
         nctx->s_dist_info, SIZ(nctx-
>s_dist_info[0]),SQLT_AFC,
         nctx->ol_dist_info_ind, nctx->ol_dist_info_len,
         nctx->ol_dist_info_rcode);
  OCIBNDRA(nctx->curn4, nctx-
>null_date_bp,errhp,":null_date",nctx->null_date,
         SIZ(nctx->null_date[0]), SQLT_DAT,nctx-
>null_date_ind,
         nctx->null_date_len, nctx->null_date_rc);

  /* set up the null date Null date is 1-jan-1811 */
  for (i=0;i<NITEMS;i++)
  {
    nctx->null_date[i][0] = 118;
    nctx->null_date[i][1] = 111;
    nctx->null_date[i][2] = 1;
    nctx->null_date[i][3] = 1;
    nctx->null_date[i][4] = 1;
    nctx->null_date[i][5] = 1;
    nctx->null_date[i][6] = 1;
  }

  return (0);

}



plnew ()
{

   int i, j, k;
   int rpc, rpc3, rowoff, iters,rcount;
   ub4 flags;

#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 */

   /* get number of order lines, and check if all are local
*/

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

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

    execstatus = OCIStmtExecute(tpcsvc,nctx-
>curn1,errhp,1,0,0,0,OCI_DEFAULT);
    if(execstatus != OCI_SUCCESS) {
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      errcode = OCIERROR(errhp,execstatus);
      if(errcode == NOT_SERIALIZABLE) {
                retries++;
                goto retry;
      } else if (errcode == RECOVERR) {
                retries++;
                goto retry;
      } else {
                return -1;
      }
    }
    /* initialization for array operations */

    for (i = 0; i < o_ol_cnt; i++) {
      nctx->ol_w_id[i] = w_id;
      nctx->ol_d_id[i] = d_id;
      nctx->ol_number[i] = i + 1;
      nctx->null_date_ind[i]= TRUE;
      nctx->nol_i_id_ind[i] = TRUE;
      nctx->nol_supply_w_id_ind[i] = TRUE;
      nctx->nol_quantity_ind[i] = TRUE;
      nctx->nol_amount_ind[i] = TRUE;
      nctx->ol_w_id_ind[i] = TRUE;
      nctx->ol_d_id_ind[i] = TRUE;
      nctx->ol_o_id_ind[i] = TRUE;
      nctx->ol_number_ind[i] = TRUE;
      nctx->ol_dist_info_ind[i] = TRUE;
      nctx->s_remote_ind[i] = TRUE;
      nctx->s_quant_ind[i] = TRUE;
      nctx->cons_ind[i] = TRUE;
      nctx->s_rowid_ind[i] = TRUE;

      nctx->nol_i_id_len[i] = sizeof(int);
      nctx->nol_supply_w_id_len[i] = sizeof(int);
      nctx->nol_quantity_len[i] = sizeof(int);
      nctx->nol_amount_len[i] = sizeof(int);
      nctx->ol_w_id_len[i] = sizeof(int);
      nctx->ol_d_id_len[i] = sizeof(int);
      nctx->ol_o_id_len[i] = sizeof(int);
      nctx->ol_number_len[i] = sizeof(int);
      /* nctx->ol_dist_info_len[i] = sizeof(nctx-
>s_dist_info[0]);*/
      nctx->ol_dist_info_len[i] = nctx->s_dist_info_len[i];
      nctx->null_date_len[i]=sizeof(nctx->null_date[0]);
      nctx->s_remote_len[i] = sizeof(int);
      nctx->s_quant_len[i] = sizeof(int);
      nctx->s_rowid_len[i] = sizeof(nctx->s_rowid_ptr[0]);
      nctx->cons_len[i] = sizeof(int);
    }
    for (i = o_ol_cnt; i < NITEMS; i++) {
      nctx->nol_i_id_ind[i] = NA;
      nctx->nol_supply_w_id_ind[i] = NA;
      nctx->nol_quantity_ind[i] = NA;
      nctx->nol_amount_ind[i] = NA;
      nctx->ol_w_id_ind[i] = NA;
      nctx->ol_d_id_ind[i] = NA;
      nctx->ol_o_id_ind[i] = NA;
      nctx->ol_number_ind[i] = NA;
      nctx->ol_dist_info_ind[i] = NA;
      nctx->null_date_ind[i]= NA;
      nctx->s_remote_ind[i] = NA;
      nctx->s_quant_ind[i] = NA;
      nctx->cons_ind[i] = NA;
      nctx->s_rowid_ind[i] = NA;

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

#ifdef OPS
    rpc = UpdStk ();
    if (rpc == -2)
      goto retry;
    else if (rpc == -1)
      return (-1);
#endif

#ifdef ISO7
iso7:
#endif

    rpc3 = SelItemStk ();
    if (rpc3 == -2)
      goto retry;
    else if (rpc3 == -1)
      return (-1);

#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 = %d\n", nol_i_id[i],
i_price[i]);
    }
    if (reread) {
      sleep (30);
      reread = 0;
      status = 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) {
#ifndef OPS
        s_quantity[i] -= nol_quantity[i];
        if (s_quantity[i] < 10)
          s_quantity[i] += 91;
#endif
        nol_amount[i] = (nol_quantity[i] * i_price[i]);
        total_amount += nol_amount[i];
        if (strstr (nctx->i_data[i], "ORIGINAL") &&
          strstr (nctx->s_data[i], "ORIGINAL"))
          brand_gen[i] = 'B';
        else
          brand_gen[i] = 'G';
      }

    }
    total_amount *= ((float)(10000 - c_discount)/10000) *
(1.0 + ((float)(d_tax)/10000) + ((float)(w_tax)/10000));
    total_amount = total_amount/100;

#ifndef OPS
    rpc = UpdStk2 ();
    if (rpc == -2)
      goto retry;
    else if (rpc == -1)
      return (-1);
#endif

    /* number of items selected != number of stock
updated */

    if (rpc3 != rpc) {
#ifdef TUX
      userlog ("Error in TPC-C server %d: %d rows of
item read, ",
               proc_no, rpc3);
      userlog ("                    but %d rows of stock
updated\n", rpc);
#else
      fprintf (stderr, "Error in TPC-C server %d: %d rows
of item read, ",
               proc_no, rpc3);
      fprintf (stderr, "                but %d rows of stock
update\n", rpc);
#endif
      /* rollback */
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      return (-1);
    }
```

```
  /* array insert into order line table */
#ifdef ISO1
  flags= OCI_DEFAULT;
#else
  flags= (status ? OCI_DEFAULT :
(OCI_DEFAULT|OCI_COMMIT_ON_SUCCESS));
#endif
  if ((o_ol_cnt - status) > 0)
  {
    execstatus = OCIStmtExecute(tpcsvc,nctx-
>curn4,errhp,o_ol_cnt - status,
                        0,0,0,flags);
    if(execstatus != OCI_SUCCESS) {
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      errcode = OCIERROR(errhp,execstatus);
      if(errcode == NOT_SERIALIZABLE) {
        retries++;
        goto retry;
      } else if (errcode == RECOVERR) {
        retries++;
        goto retry;
      } else {
        return -1;
      }
    }
    OCIAttrGet(nctx-
>curn4,OCI_HTYPE_STMT,&rcount,NULL,
                          OCI_ATTR_ROWCNT, errhp);
    if (rcount != (o_ol_cnt - status))
    {
#ifdef TUX
      userlog ("Error in TPC-C server %d: array insert
failed\n",
                          proc_no);
#else
      fprintf (stderr, "Error in TPC-C server %d: array
insert failed\n",
                          proc_no);
#endif
      /* rollback */
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      return (-1);
    }
  }

#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) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
  }
#ifdef ISO1
  else {
    OCITransCommit(tpcsvc, errhp, OCI_DEFAULT);
  }
#endif

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

  return (0);

}



void plnewdone ()

{

  int i;

  if (nctx)
  {
    OCIHandleFree((dvoid *)nctx-
>curn1,OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)nctx-
>curn2,OCI_HTYPE_STMT);
    for (i = 0; i < 10; i++)
      OCIHandleFree((dvoid *)(nctx-
>curn3)[i],OCI_HTYPE_STMT);
    OCIHandleFree((dvoid *)nctx-
>curn4,OCI_HTYPE_STMT);
    free (nctx);
  }
}

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

shiftitemstock (i, j)

int i, j;

{

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

  /* shift up the order_line values */

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

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

  nctx->nol_supply_w_id_ind [i] =  nctx-
>nol_supply_w_id_ind[j];
  nol_supply_w_id[i] = nol_supply_w_id[j];

}
swapitemstock (i, j)

int i, j;

{

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

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

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

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

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

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

```
   strncpy (nctx->i_data[j], tempstr, 51);

   tempsb2 = nctx->s_quantity_ind[i];
   nctx->s_quantity_ind[i] = nctx->s_quantity_ind[j];
   nctx->s_quantity_ind[j] = tempsb2;
   tempub2 = nctx->s_quantity_len[i];
   nctx->s_quantity_len[i] = nctx->s_quantity_len[j];
   nctx->s_quantity_len[j] = tempub2;
   tempub2 = nctx->s_quantity_rcode[i];
   nctx->s_quantity_rcode[i] = nctx->s_quantity_rcode[j];
   nctx->s_quantity_rcode[j] = tempub2;
   tempi = s_quantity[i];
   s_quantity[i] = s_quantity[j];
   s_quantity[j] = tempi;

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

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

}



SelItemStk ()


{

   int i, j, rpc3,rcount;

   /* array select from item and stock tables */
   execstatus=OCIStmtExecute(tpcsvc,(nctx-
>curn3)[d_id-1],errhp,o_ol_cnt,
                              0,0,0,OCI_DEFAULT);
   if((execstatus != OCI_SUCCESS) && (execstatus !=
OCI_NO_DATA)) {
     errcode = OCIERROR(errhp,execstatus);
     if(errcode == NOT_SERIALIZABLE) {
       retries++;
       OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
       return (-2);
     } else if (errcode == RECOVERR) {
     /* In case of NO_DATA this should NOT return, but
simply fall through */
       OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
       retries++;
       return (-2);
     } else {
       OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
       return (-1);
     }
   }
   /* mark invalid items */
   OCIAttrGet((nctx->curn3)[d_id-1],
OCI_HTYPE_STMT,&rcount,NULL,
                        OCI_ATTR_ROWCNT, errhp);
   rpc3 = rcount;

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

   if ((status = o_ol_cnt - rcount) >1)
   {
#ifdef TUX
      userlog ("TPC-C server %d: more than 1 invalid
item?\n", proc_no);
#else
      fprintf (stderr, "TPC-C server %d: more than 1
invalid item?\n", proc_no);
#endif
     return (rpc3);
   }
   if (status == 0) return (rpc3);

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

   for (i = 0; i < o_ol_cnt; i++) {
/* TSL)HAGINO 970826 Mark Invalid Output */
   fprintf(stdout,".");

     if (nctx->cons[i] != i) break;   /* this item is invalid */
   }


#ifdef TUX
          userlog ("TPC-C server %d: reordering items
and stocks\n",
                   proc_no);
#else
/*
          fprintf (stderr, "TPC-C server %d: reordering
items and stocks\n",
                   proc_no);   */
#endif

   /* not the last item - shift up */

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

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

   return (rpc3);

}



#ifdef OPS
UpdStk ()

{
  int rcount;
   /* array update of stock table */

  execstatus = OCIStmtExecute(tpcsvc,nctx-
>curn2,errhp,o_ol_cnt,
                        0,0,0,OCI_DEFAULT);
   if(execstatus != OCI_SUCCESS) {
     OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
     errcode = OCIERROR(errhp,execstatus);
     if(errcode == NOT_SERIALIZABLE) {
       retries++;
                return (-2);
     } else if (errcode == RECOVERR) {
       retries++;
                return (-2);
     } else {
       return -1;
     }
   }
   OCIAttrGet(nctx-
>curn2,OCI_HTYPE_STMT,&rcount,NULL,
           OCI_ATTR_ROWCNT, errhp);
   if (rcount != (o_ol_cnt) ) {
#ifdef TUX
      userlog ("Error in TPC-C server %d: array update
failed in UpdStk()\n",
             proc_no);
#else
      fprintf (stderr, "Error in TPC-C server %d: array
update failed in UpdStk()\n",
             proc_no);
#endif
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      return (-1);
   }
   return (rcount);
}
```

```
#endif



#ifndef OPS
UpdStk2 ()


{

  int rpc, rowoff, iters,rcount;

  /* array update of stock table */

  execstatus = OCIStmtExecute(tpcsvc,nctx-
>curn2,errhp,o_ol_cnt-status,0,0,0,

OCI_DEFAULT);
  if(execstatus != OCI_SUCCESS) {
    OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
    errcode = OCIERROR(errhp,execstatus);
    if(errcode == NOT_SERIALIZABLE) {
      retries++;
              return (-2);
    } else if (errcode == RECOVERR) {
      retries++;
              return (-2);
    } else {
      return -1;
    }
  }
  OCIAttrGet(nctx-
>curn2,OCI_HTYPE_STMT,&rcount,NULL,
OCI_ATTR_ROWCNT, errhp);
  rpc = rcount;

  if (rpc != (o_ol_cnt - status)) {
#ifdef TUX
     userlog ("Error in TPC-C server %d: array update
failed\n",
            proc_no);
#else
     fprintf (stderr, "Error in TPC-C server %d: array
update failed\n",
            proc_no);
#endif
     OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
     return (-1);
  }

  return (rpc);

}
#endif
```

plord.c

```
#ifdef RCSID

static char *RCSid =

  "$Header: plord.c 7030100.1 95/07/19 14:46:13 plai
Generic<base> $ Copyr (c) 1994 Oracle";

#endif /* RCSID */
```

```
/*=========================================
========================+
|       Copyright (c) 1995  Oracle Corp, Redwood
Shores, CA      |
|                OPEN SYSTEMS PERFORMANCE
GROUP               |
|                   All Rights Reserved                   |

+=========================================
========================+
| FILENAME
|   plord.c
| DESCRIPTION
|   OCI version (using PL/SQL anynomous block) of
|   ORDER STATUS transaction in TPC-C benchmark.

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

#include "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;"
#endif

#define SQLCUR0 "SELECT rowid FROM customer \
              WHERE c_d_id = :d_id AND c_w_id = :w_id
AND c_last = :c_last \
              ORDER BY c_w_id, c_d_id, c_last, c_first"

#define SQLCUR1 "SELECT c_id, c_balance, c_first,
c_middle, \
              o_id, o_entry_d, o_carrier_id, o_ol_cnt,
c_last \
              FROM customer, orders \
              WHERE customer.rowid = :cust_rowid \
              AND   o_d_id = c_d_id AND o_w_id =
c_w_id AND o_c_id = c_id \
              ORDER BY o_w_id, o_d_id, o_c_id, o_id
DESC"


#define SQLCUR2 "SELECT c_balance, c_first,
c_middle, c_last, \
              o_id, o_entry_d, o_carrier_id, o_ol_cnt \
              FROM customer, orders \
              WHERE c_id = :c_id AND c_d_id = :d_id
AND c_w_id = :w_id \
              AND o_d_id = c_d_id AND o_w_id = c_w_id
AND o_c_id = c_id \
              ORDER BY o_w_id, o_d_id, o_c_id, o_id
DESC"


#define SQLCUR3 "SELECT
ol_i_id,ol_supply_w_id,ol_quantity,ol_amount, \
             ol_delivery_d\
              FROM order_line \
              WHERE ol_d_id = :d_id AND ol_w_id = :w_id
AND ol_o_id = :o_id"
```

```
struct ordctx {
  sb2 c_rowid_ind[3000];
  sb2 ol_supply_w_id_ind[NITEMS];
  sb2 ol_i_id_ind[NITEMS];
  sb2 ol_quantity_ind[NITEMS];
  sb2 ol_amount_ind[NITEMS];
  sb2 ol_delivery_d_ind[NITEMS];
  sb2 ol_w_id_ind;
  sb2 ol_d_id_ind;
  sb2 ol_o_id_ind;
  sb2 c_id_ind;
  sb2 c_first_ind;
  sb2 c_middle_ind;
  sb2 c_balance_ind;
  sb2 c_last_ind;
  sb2 o_id_ind;
  sb2 o_entry_d_ind;
  sb2 o_carrier_id_ind;
  sb2 o_ol_cnt_ind;

  ub2 c_rowid_len[3000];
  ub2 ol_supply_w_id_len[NITEMS];
  ub2 ol_i_id_len[NITEMS];
  ub2 ol_quantity_len[NITEMS];
  ub2 ol_amount_len[NITEMS];
  ub2 ol_delivery_d_len[NITEMS];
  ub2 ol_w_id_len;
  ub2 ol_d_id_len;
  ub2 ol_o_id_len;

  ub2 c_rowid_rcode[3000];
  ub2 ol_supply_w_id_rcode[NITEMS];
  ub2 ol_i_id_rcode[NITEMS];
  ub2 ol_quantity_rcode[NITEMS];
  ub2 ol_amount_rcode[NITEMS];
  ub2 ol_delivery_d_rcode[NITEMS];
  ub2 ol_w_id_rcode;
  ub2 ol_d_id_rcode;
  ub2 ol_o_id_rcode;

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

  OCIStmt *curo0;
  OCIStmt *curo1;
  OCIStmt *curo2;
  OCIStmt *curo3;
  OCIBind *w_id_bp0;
  OCIBind *w_id_bp2;
  OCIBind *w_id_bp3;
  OCIBind *d_id_bp0;
  OCIBind *d_id_bp2;
  OCIBind *d_id_bp3;
  OCIBind *c_id_bp;
  OCIBind *byln_bp;
  OCIBind *c_last_bp;
  OCIBind *c_first_bp;
  OCIBind *c_middle_bp;
  OCIBind *c_balance_bp;
  OCIBind *o_id_bp;
```

```
  OCIBind *o_entry_d_bp;
  OCIBind *o_cr_id_bp;
  OCIBind *o_ol_cnt_bp;
  OCIBind *ol_s_w_id_bp;
  OCIBind *ol_i_id_bp;
  OCIBind *ol_quantity_bp;
  OCIBind *ol_amount_bp;
  OCIBind *ol_d_d_bp;
  OCIBind *c_rowid_bp;
  OCIDefine *c_rowid_dp;
  OCIDefine *c_last_dp;
  OCIDefine *c_last_dp1;
  OCIDefine *c_id_dp;
  OCIDefine *c_first_dp1;
  OCIDefine *c_first_dp2;
  OCIDefine *c_middle_dp1;
  OCIDefine *c_middle_dp2;
  OCIDefine *c_balance_dp1;
  OCIDefine *c_balance_dp2;
  OCIDefine *o_id_dp1;
  OCIDefine *o_id_dp2;
  OCIDefine *o_entry_d_dp1;
  OCIDefine *o_entry_d_dp2;
  OCIDefine *o_cr_id_dp1;
  OCIDefine *o_cr_id_dp2;
  OCIDefine *o_ol_cnt_dp1;
  OCIDefine *o_ol_cnt_dp2;
  OCIDefine *ol_d_d_dp;
  OCIDefine *ol_i_id_dp;
  OCIDefine *ol_supply_w_id_dp;
  OCIDefine *ol_quantity_dp;
  OCIDefine *ol_amount_dp;
  OCIDefine *ol_d_base_dp;

  OCIRowid *c_rowid_ptr[3000];
  int cs;
  int cust_idx;
  int norow;
};

typedef struct ordctx ordctx;

ordctx *octx;



plordinit ()

{

  int i;
  text stmbuf[SQL_BUF_SIZE];

  octx = (ordctx *) malloc (sizeof(ordctx));
  memset(octx,(char)0,sizeof(ordctx));
  octx->cs = 1;
  octx->norow = 0;
/* get the rowid handles */
  for(i=0;i<3000;i++) {
   OCIERROR(errhp,
OCIDescriptorAlloc(tpcenv,(dvoid**)&octx-
>c_rowid_ptr[i],
      OCI_DTYPE_ROWID,0,(dvoid**)0));
 }


#ifdef ISO8
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid **)&octx-
>curo0,OCI_HTYPE_STMT,0,(dvoid**)0));
#else
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx-
>curo0,OCI_HTYPE_STMT,0,(dvoid**)0));
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx-
>curo1,OCI_HTYPE_STMT,0,(dvoid**)0));
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx-
>curo2,OCI_HTYPE_STMT,0,(dvoid**)0));
  OCIERROR(errhp,
    OCIHandleAlloc(tpcenv,(dvoid**)&octx-
>curo3,OCI_HTYPE_STMT,0,(dvoid**)0));
#endif

#ifdef ISO8
  sprintf((char *) stmbuf, SQLTXT);
  OCIERROR(errhp,
    OCIStmtPrepare(octx-
>curo0,errhp,stmbuf,strlen((char *)stmbuf),

OCI_NTV_SYNTAX,OCI_DEFAULT));
#else
/*  c_id = 0, use find customer by lastname. Get an
array or rowid's back*/
  sprintf((char *) stmbuf, SQLCUR0);
  OCIERROR(errhp,
    OCIStmtPrepare(octx-
>curo0,errhp,stmbuf,strlen((char *)stmbuf),

OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(errhp,
    OCIAttrSet(octx-
>curo0,OCI_HTYPE_STM,(dvoid*)&octx->norow,0,
            OCI_ATTR_PREFETCH_ROWS,errhp));

/* get order/customer info back based on rowid */
  sprintf((char *) stmbuf, SQLCUR1);
  OCIERROR(errhp,
    OCIStmtPrepare(octx-
>curo1,errhp,stmbuf,strlen((char *)stmbuf),

OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(errhp,
    OCIAttrSet(octx-
>curo1,OCI_HTYPE_STM,(dvoid*)&octx->norow,0,
            OCI_ATTR_PREFETCH_ROWS,errhp));

/*  c_id == 0, use lastname to find customer */
  sprintf((char *) stmbuf, SQLCUR2);
  OCIERROR(errhp,
    OCIStmtPrepare(octx-
>curo2,errhp,stmbuf,strlen((char *)stmbuf),

OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(errhp,
    OCIAttrSet(octx-
>curo2,OCI_HTYPE_STM,(dvoid*)&octx->norow,0,
            OCI_ATTR_PREFETCH_ROWS,errhp));

  sprintf((char *) stmbuf, SQLCUR3);
  OCIERROR(errhp,
    OCIStmtPrepare(octx-
>curo3,errhp,stmbuf,strlen((char *)stmbuf),

OCI_NTV_SYNTAX,OCI_DEFAULT));
  OCIERROR(errhp,
    OCIAttrSet(octx-
>curo3,OCI_HTYPE_STM,(dvoid*)&octx->norow,0,
            OCI_ATTR_PREFETCH_ROWS,errhp));
#endif

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

    octx->ol_supply_w_id_len[i] = sizeof(int);
    octx->ol_i_id_len[i] = sizeof(int);
    octx->ol_quantity_len[i] = sizeof(int);
    octx->ol_amount_len[i] = sizeof(int);
    octx->ol_delivery_d_len[i] = sizeof(ol_d_base[0]);
  }
  octx->ol_supply_w_id_csize = NITEMS;
  octx->ol_i_id_csize = NITEMS;
  octx->ol_quantity_csize = NITEMS;
  octx->ol_amount_csize = NITEMS;
  octx->ol_delivery_d_csize = NITEMS;
  octx->ol_w_id_csize = NITEMS;
  octx->ol_o_id_csize = NITEMS;
  octx->ol_d_id_csize = NITEMS;
  octx->ol_w_id_ind = TRUE;
  octx->ol_d_id_ind = TRUE;
  octx->ol_o_id_ind = TRUE;
  octx->ol_w_id_len = sizeof(int);
  octx->ol_d_id_len = sizeof(int);
  octx->ol_o_id_len = sizeof(int);

  /* bind variables */
#ifdef ISO8

  OCIBND(octx->curo0,octx-
>w_id_bp0,errhp,":w_id",ADR(w_id),SIZ(w_id),
                SQLT_INT);
  OCIBND(octx->curo0, octx->d_id_bp0,
errhp,":d_id",ADR(d_id),SIZ(d_id),
                SQLT_INT);
  OCIBND(octx->curo0, octx->c_id_bp,
errhp,":c_id",ADR(c_id),SIZ(c_id),
                SQLT_INT);
  OCIBND(octx->curo0, octx->byln_bp,
errhp,":byln",ADR(bylastname),
                SIZ(bylastname),SQLT_INT);
  OCIBND(octx->curo0, octx->c_last_bp,
errhp,":c_last",c_last,
                SIZ(c_last),SQLT_STR);
  OCIBND(octx->curo0, octx->c_first_bp,
errhp,":c_first",c_first,
                SIZ(c_first),SQLT_STR);
  OCIBND(octx->curo0, octx->c_middle_bp,
errhp,":c_middle",c_middle,
                SIZ(c_middle),SQLT_STR);
  OCIBND(octx->curo0, octx->c_balance_bp,
errhp,":c_balance",ADR(c_balance),
                SIZ(c_balance),SQLT_FLT);
  OCIBND(octx->curo0, octx->o_id_bp,
errhp,":o_id",ADR(o_id),
                SIZ(o_id),SQLT_INT);
  OCIBND(octx->curo0, octx->o_entry_d_bp,
errhp,":o_entry_d",
```

```
                o_entry_d_base,SIZ(o_entry_d_base),SQL
T_DAT);
  OCIBND(octx->curo0, octx->o_cr_id_bp,
errhp,":o_cr_id",ADR(o_carrier_id),
                        SIZ(o_carrier_id), SQLT_INT);
  OCIBND(octx->curo0, octx->o_ol_cnt_bp,
errhp,":o_ol_cnt",ADR(o_ol_cnt),
                        SIZ(o_ol_cnt),SQLT_INT);
  OCIBNDRAA(octx->curo0, octx->ol_s_w_id_bp,
errhp,":ol_s_w_id",

                ol_supply_w_id,SIZ(int),SQLT_INT,
                octx->ol_supply_w_id_ind,octx-
>ol_supply_w_id_len,
                octx->ol_supply_w_id_rcode,NITEMS,
                        ADR(octx-
>ol_supply_w_id_csize));
  OCIBNDRAA(octx->curo0, octx->ol_i_id_bp,
errhp,":ol_i_id",ol_i_id,
                        SIZ(int),SQLT_INT,
                octx->ol_i_id_ind,octx->ol_i_id_len,octx-
>ol_i_id_rcode,NITEMS,
                ADR(octx->ol_i_id_csize));
  OCIBNDRAA(octx->curo0, octx->ol_quantity_bp,
errhp,":ol_quantity",
                        ol_quantity,SIZ(int),SQLT_INT,
                octx->ol_quantity_ind,octx-
>ol_quantity_len,
                        octx->ol_quantity_rcode,
NITEMS,ADR(octx->ol_quantity_csize));
  OCIBNDRAA(octx->curo0, octx->ol_amount_bp,
errhp,":ol_amount",ol_amount,
                        SIZ(float),SQLT_FLT,
                octx->ol_amount_ind,octx-
>ol_amount_len,octx->ol_amount_rcode,
                NITEMS,ADR(octx->ol_amount_csize));
  OCIBNDRAA(octx->curo0, octx->ol_d_d_bp,
errhp,":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));
#else

  /* c_id (customer id) is not known */
  OCIBND(octx->curo0,octx-
>w_id_bp0,errhp,":w_id",ADR(w_id),SIZ(int),SQLT_INT
);
  OCIBND(octx->curo0,octx-
>d_id_bp0,errhp,":d_id",ADR(d_id),SIZ(int),SQLT_INT);
  OCIBND(octx->curo0,octx-
>c_last_bp,errhp,":c_last",c_last,SIZ(c_last),
        SQLT_STR);
  OCIDFNRA(octx->curo0,octx-
>c_rowid_dp,errhp,1,octx->c_rowid_ptr,
         sizeof(octx->c_rowid_ptr[0]),SQLT_RDD,octx-
>c_rowid_ind,
         octx->c_rowid_len,octx->c_rowid_rcode);

  OCIBND(octx->curo1,octx-
>c_rowid_bp,errhp,":cust_rowid",
        &octx->c_rowid_ptr[octx->cust_idx],
        sizeof( octx->c_rowid_ptr[0]),SQLT_RDD);
  OCIDEF(octx->curo1,octx-
>c_id_dp,errhp,1,ADR(c_id),SIZ(int),SQLT_INT);
  OCIDEF(octx->curo1,octx-
>c_balance_dp1,errhp,2,ADR(c_balance),
        SIZ(double),SQLT_FLT);
  OCIDEF(octx->curo1,octx-
>c_first_dp1,errhp,3,c_first,SIZ(c_first),
        SQLT_STR);
  OCIDEF(octx->curo1,octx-
>c_middle_dp1,errhp,4,c_middle,
        SIZ(c_middle),SQLT_STR);
  OCIDEF(octx->curo1,octx-
>o_id_dp1,errhp,5,ADR(o_id),SIZ(int),SQLT_INT);
  OCIDEF(octx->curo1,octx->o_entry_d_dp1,errhp,6,

o_entry_d_base,SIZ(o_entry_d_base),SQLT_DAT);
  OCIDEF(octx->curo1,octx-
>o_cr_id_dp1,errhp,7,ADR(o_carrier_id),
        SIZ(int),SQLT_INT);
  OCIDEF(octx->curo1,octx-
>o_ol_cnt_dp1,errhp,8,ADR(o_ol_cnt),
        SIZ(int),SQLT_INT);
  OCIDEF(octx->curo1,octx-
>c_last_dp1,errhp,9,c_last,SIZ(c_last), SQLT_STR);

/* Bind for third cursor , no-zero customer id */
  OCIBND(octx->curo2,octx-
>w_id_bp2,errhp,":w_id",ADR(w_id),SIZ(int),SQLT_INT
);
  OCIBND(octx->curo2,octx-
>d_id_bp2,errhp,":d_id",ADR(d_id),SIZ(int),SQLT_INT);
  OCIBND(octx->curo2,octx-
>c_id_bp,errhp,":c_id",ADR(c_id),SIZ(int),SQLT_INT);
  OCIDEF(octx->curo2,octx-
>c_balance_dp2,errhp,1,ADR(c_balance),
        SIZ(double),SQLT_FLT);
  OCIDEF(octx->curo2,octx-
>c_first_dp2,errhp,2,c_first,SIZ(c_first),
        SQLT_STR);
  OCIDEF(octx->curo2,octx-
>c_middle_dp2,errhp,3,c_middle,
        SIZ(c_middle),SQLT_STR);
  OCIDEF(octx->curo2,octx-
>c_last_dp,errhp,4,c_last,SIZ(c_last), SQLT_STR);
  OCIDEF(octx->curo2,octx-
>o_id_dp2,errhp,5,ADR(o_id),SIZ(int),SQLT_INT);
  OCIDEF(octx->curo2,octx->o_entry_d_dp2,errhp,6,

o_entry_d_base,SIZ(o_entry_d_base),SQLT_DAT);
  OCIDEF(octx->curo2, octx-
>o_cr_id_dp2,errhp,7,ADR(o_carrier_id),
        SIZ(int), SQLT_INT);
  OCIDEF(octx->curo2,octx-
>o_ol_cnt_dp2,errhp,8,ADR(o_ol_cnt),
        SIZ(int),SQLT_INT);

/* Bind for last cursor */

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

  OCIDFNRA(octx->curo3, octx->ol_i_id_dp, errhp, 1,
ol_i_id,SIZ(int),SQLT_INT,
                octx->ol_i_id_ind,octx->ol_i_id_len, octx-
>ol_i_id_rcode);
  OCIDFNRA(octx->curo3,octx-
>ol_supply_w_id_dp,errhp,2, ol_supply_w_id,
        SIZ(int),SQLT_INT, octx->ol_supply_w_id_ind,
        octx->ol_supply_w_id_len, octx-
>ol_supply_w_id_rcode);
  OCIDFNRA(octx->curo3, octx-
>ol_quantity_dp,errhp,3, ol_quantity,SIZ(int),
        SQLT_INT, octx->ol_quantity_ind,octx-
>ol_quantity_len,
                octx->ol_quantity_rcode);
  OCIDFNRA(octx->curo3,octx-
>ol_amount_dp,errhp,4,ol_amount, SIZ(int),
        SQLT_INT,octx->ol_amount_ind, octx-
>ol_amount_len,
        octx->ol_amount_rcode);
  OCIDFNRA(octx->curo3,octx-
>ol_d_base_dp,errhp,5,ol_d_base,7, SQLT_DAT,
        octx->ol_delivery_d_ind,octx-
>ol_delivery_d_len,
                octx->ol_delivery _d_rcode);
#endif /* ISO8 */
  return (0);

}



plord ()

{

  int i;
  int rcount;

retry:

  for (i = 0; i < NITEMS; i++) {
    octx->ol_supply_w_id_ind[i] = TRUE;
    octx->ol_i_id_ind[i] = TRUE;
    octx->ol_quantity_ind[i] = TRUE;
    octx->ol_amount_ind[i] = TRUE;
    octx->ol_delivery_d_ind[i] = TRUE;
    octx->ol_supply_w_id_len[i] = sizeof(int);
    octx->ol_i_id_len[i] = sizeof(int);
    octx->ol_quantity_len[i] = sizeof(int);
    octx->ol_amount_len[i] = sizeof(int);
    octx->ol_delivery_d_len[i] = sizeof(ol_d_base[0]);
  }
  octx->ol_supply_w_id_csize = NITEMS;
  octx->ol_i_id_csize = NITEMS;
  octx->ol_quantity_csize = NITEMS;
  octx->ol_amount_csize = NITEMS;
  octx->ol_delivery_d_csize = NITEMS;
#ifdef ISO8
  OCIERROR(errhp,
    OCIStmtExecute(tpcsvc,octx-
>curo0,errhp,1,0,0,0,OCI_DEFAULT));
#else
  if(bylastname) {
   execstatus=OCIStmtExecute(tpcsvc,octx-
>curo0,errhp,3000,0,0,0,OCI_DEFAULT);
    if (execstatus != OCI_NO_DATA)    /* will get
OCI_NO_DATA if <3000 found */
    {
      OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
      errcode = OCIERROR(errhp,execstatus);
```

```
        if(errcode == NOT_SERIALIZABLE) {
            retries++;
            goto retry;
        } else if (errcode == RECOVERR) {
            retries++;
            goto retry;
        } else {
          return -1;
        }
    }
     /* get rowcount, find middle one */
    OCIAttrGet(octx-
>curo0,OCI_HTYPE_STM,&rcount,NULL,OCI_ATTR_R
OWCNT,errhp);
    octx->cust_idx=(rcount+1)/2 ;
    execstatus = OCIStmtExecute(tpcsvc,octx-
>curo1,errhp,1,0,0,0,OCI_DEFAULT);
    if (execstatus != OCI_SUCCESS)
    {
        OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
        errcode = OCIERROR(errhp,execstatus);
        if(errcode == NOT_SERIALIZABLE) {
            retries++;
            goto retry;
        } else if (errcode == RECOVERR) {
            retries++;
            goto retry;
        } else {
          return -1;
        }
    }

  } else {
    execstatus = OCIStmtExecute(tpcsvc,octx-
>curo2,errhp,1,0,0,0,OCI_DEFAULT);
    if (execstatus != OCI_SUCCESS)
    {
        OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
        errcode = OCIERROR(errhp,execstatus);
        if(errcode == NOT_SERIALIZABLE) {
            retries++;
            goto retry;
        } else if (errcode == RECOVERR) {
            retries++;
            goto retry;
        } else {
          return -1;
        }
    }
  }
  octx->ol_w_id_ind = TRUE;
  octx->ol_d_id_ind = TRUE;
  octx->ol_o_id_ind = TRUE;
  octx->ol_w_id_len = sizeof(int);
  octx->ol_d_id_len = sizeof(int);
  octx->ol_o_id_len = sizeof(int);

  execstatus =   OCIStmtExecute(tpcsvc,octx-
>curo3,errhp,o_ol_cnt,0,0,0,
                OCI_DEFAULT |
OCI_COMMIT_ON_SUCCESS);
  if (execstatus != OCI_SUCCESS)
    {
        OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
        errcode = OCIERROR(errhp,execstatus);
        if(errcode == NOT_SERIALIZABLE) {
            retries++;
            goto retry;
        } else if (errcode == RECOVERR) {
            retries++;
            goto retry;
        } else {
          return -1;
        }
    }
#ifdef NOTMORE
  OCIERROR(errhp,
    OCITransCommit(tpcsvc,errhp,OCI_DEFAULT));
#endif

  /* clean up and convert the delivery dates */
  for (i = 0; i < o_ol_cnt; i++) {
    if (octx->ol_delivery_d_ind[i] == -1) /* null date in
field */
                strncpy(ol_delivery_d[i],"1-1-1811",10);
    else
                cvtdmy(ol_d_base[i],ol_delivery_d[i]);
  }

#endif
  return (0);

}



void plorddone ()

{

  if (octx)
    free (octx);

}
```

plpay.c

```
#ifdef RCSID

static char *RCSid =

  "$Header: plpay.c 7030100.1 95/07/19 14:44:59 plai
Generic<base> $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=========================================
========================+
|       Copyright (c) 1995  Oracle Corp, Redwood
Shores, CA       |
|                OPEN SYSTEMS PERFORMANCE
GROUP               |
|                   All Rights Reserved                   |

+=========================================
========================+
| FILENAME
|   plpay.c
| DESCRIPTION
|   OCI version (using PL/SQL stored procedure) of
|   PAYMENT transaction in TPC-C benchmark.

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

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

#define SQLTXT_ZERO "BEGIN
apayment.adopayment(:w_id, :d_id, :c_w_id, :c_d_id, \
  :c_id, 0, \
  :h_amount, :c_last, :w_street_1, :w_street_2, :w_city,
:w_state, \
  :w_zip, :d_street_1, :d_street_2, :d_city, :d_state,
:d_zip, :c_first, \
  :c_middle, :c_street_1, :c_street_2, :c_city, :c_state,
:c_zip, :c_phone, \
  :c_since, :c_credit, :c_credit_lim, :c_discount,
:c_balance, :c_data, \
  :h_date, :retry, :cr_date); END;"

#define SQLTXT_NONZERO "BEGIN
apayment.adopayment(:w_id, :d_id, :c_w_id, :c_d_id, \
  :c_id, 1, \
  :h_amount, :c_last, :w_street_1, :w_street_2, :w_city,
:w_state, \
  :w_zip, :d_street_1, :d_street_2, :d_city, :d_state,
:d_zip, :c_first, \
  :c_middle, :c_street_1, :c_street_2, :c_city, :c_state,
:c_zip, :c_phone, \
  :c_since, :c_credit, :c_credit_lim, :c_discount,
:c_balance, :c_data, \
  :h_date, :retry, :cr_date); END;"

#define SQLTXT_INIT "BEGIN pay.pay_init; END;"

paycтx *pctx;

plpayinit ()

{

  text stmbuf[SQL_BUF_SIZE];
  pctx = (paycтx *)malloc(sizeof(paycтx));
  memset(pctx,(char)0,sizeof(paycтx));

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

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

  /*  build the init statement  and execute it */

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

  /* customer id !=  0, go by last name */
#ifdef ATOMA
  sqlfile("../blocks/paynz_abort.sql",stmbuf);
  /* sprintf((char *)stmbuf, SQLTXT_NONZERO); */
#else
  sqlfile("../blocks/paynz.sql",stmbuf);
```

```
#endif
  OCIERROR(errhp,OCIStmtPrepare(pctx->curp0,
errhp, stmbuf,
        strlen((char *)stmbuf), OCI_NTV_SYNTAX,
OCI_DEFAULT));

  /* customer id ==  0, go by last name */
#ifdef ATOMA
  sqlfile("../blocks/payz_abort.sql",stmbuf);
  /* sprintf((char *)stmbuf, SQLTXT_ZERO); */
#else
  sqlfile("../blocks/payz.sql",stmbuf);
#endif
  OCIERROR(errhp,OCIStmtPrepare(pctx->curp1,
errhp, stmbuf,
        strlen((char *)stmbuf), OCI_NTV_SYNTAX,
OCI_DEFAULT));

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



  /* bind variables */


  OCIBNDR(pctx->curp0, pctx->w_id_bp,
errhp,":w_id",ADR(w_id),SIZ(int),
        SQLT_INT, &pctx->w_id_ind, NULL, NULL);
  OCIBNDR(pctx->curp0, pctx->d_id_bp,
errhp,":d_id",ADR(d_id),SIZ(int),
        SQLT_INT, &pctx->d_id_ind, NULL, NULL);
  OCIBND(pctx->curp0, pctx->c_w_id_bp,
errhp,":c_w_id",ADR(c_w_id),SIZ(int),
        SQLT_INT);
  OCIBND(pctx->curp0, pctx->c_d_id_bp,
errhp,":c_d_id",ADR(c_d_id),SIZ(int),
        SQLT_INT);
  OCIBND(pctx->curp0, pctx->c_id_bp,
errhp,":c_id",ADR(c_id),SIZ(int),
        SQLT_INT);
  OCIBNDR(pctx->curp0, pctx->h_amount_bp,
errhp,":h_amount",ADR(h_amount),
        SIZ(int),SQLT_INT, &pctx->h_amount_ind,
&pctx->h_amount_len,
        &pctx->h_amount_rc);
  OCIBNDR(pctx->curp0, pctx->c_last_bp,
errhp,":c_last",c_last,SIZ(c_last),
        SQLT_STR, &pctx->c_last_ind, &pctx-
>c_last_len, &pctx->c_last_rc);
  OCIBNDR(pctx->curp0, pctx->w_street_1_bp,
errhp,":w_street_1",w_street_1,
        SIZ(w_street_1),SQLT_STR, &pctx-
>w_street_1_ind,
        &pctx->w_street_1_len, &pctx->w_street_1_rc);
  OCIBNDR(pctx->curp0, pctx->w_street_2_bp,
errhp,":w_street_2",w_street_2,
        SIZ(w_street_2),SQLT_STR, &pctx-
>w_street_2_ind,
        &pctx->w_street_2_len, &pctx->w_street_2_rc);
  OCIBNDR(pctx->curp0, pctx->w_city_bp,
errhp,":w_city",w_city,SIZ(w_city),
        SQLT_STR, &pctx->w_city_ind, &pctx-
>w_city_len, &pctx->w_city_rc);
  OCIBNDR(pctx->curp0, pctx->w_state_bp,
errhp,":w_state",w_state,SIZ(w_state),
        SQLT_STR, &pctx->w_state_ind, &pctx-
>w_state_len, &pctx->w_state_rc);
  OCIBNDR(pctx->curp0, pctx->w_zip_bp,
errhp,":w_zip",w_zip,SIZ(w_zip),
        SQLT_STR, &pctx->w_zip_ind, &pctx-
>w_zip_len, &pctx->w_zip_rc);
  OCIBNDR(pctx->curp0, pctx->d_street_1_bp,
errhp,":d_street_1",d_street_1,
        SIZ(d_street_1),SQLT_STR, &pctx-
>d_street_1_ind,
        &pctx->d_street_1_len, &pctx->d_street_1_rc);
  OCIBNDR(pctx->curp0, pctx->d_street_2_bp,
errhp,":d_street_2",d_street_2,
        SIZ(d_street_2),SQLT_STR, &pctx-
>d_street_2_ind,
        &pctx->d_street_2_len, &pctx->d_street_2_rc);
  OCIBNDR(pctx->curp0, pctx->d_city_bp,
errhp,":d_city",d_city,SIZ(d_city),
        SQLT_STR, &pctx->d_city_ind, &pctx-
>d_city_len, &pctx->d_city_rc);
  OCIBNDR(pctx->curp0, pctx->d_state_bp,
errhp,":d_state",d_state,SIZ(d_state),
        SQLT_STR, &pctx->d_state_ind, &pctx-
>d_state_len, &pctx->d_state_rc);
  OCIBNDR(pctx->curp0, pctx->d_zip_bp,
errhp,":d_zip",d_zip,SIZ(d_zip),
        SQLT_STR, &pctx->d_zip_ind, &pctx-
>d_zip_len, &pctx->d_zip_rc);
  OCIBNDR(pctx->curp0, pctx->c_first_bp,
errhp,":c_first",c_first,SIZ(c_first),
        SQLT_STR, &pctx->c_first_ind, &pctx-
>c_first_len, &pctx->c_first_rc);
  OCIBNDR(pctx->curp0, pctx->c_middle_bp,
errhp,":c_middle",c_middle,2,
        SQLT_AFC, &pctx->c_middle_ind, &pctx-
>c_middle_len,
        &pctx->c_middle_rc);
  OCIBNDR(pctx->curp0, pctx->c_street_1_bp,
errhp,":c_street_1",c_street_1,
        SIZ(c_street_1),SQLT_STR, &pctx-
>c_street_1_ind,
        &pctx->c_street_1_len, &pctx->c_street_1_rc);
  OCIBNDR(pctx->curp0, pctx->c_street_2_bp,
errhp,":c_street_2",c_street_2,
        SIZ(c_street_2),SQLT_STR, &pctx-
>c_street_2_ind,
        &pctx->c_street_2_len, &pctx->c_street_2_rc);
  OCIBNDR(pctx->curp0, pctx->c_city_bp,
errhp,":c_city",c_city,SIZ(c_city),
        SQLT_STR, &pctx->c_city_ind, &pctx-
>c_city_len, &pctx->c_city_rc);
  OCIBNDR(pctx->curp0, pctx->c_state_bp,
errhp,":c_state",c_state,SIZ(c_state),
        SQLT_STR, &pctx->c_state_ind, &pctx-
>c_state_len, &pctx->c_state_rc);
  OCIBNDR(pctx->curp0, pctx->c_zip_bp,
errhp,":c_zip",c_zip,SIZ(c_zip),
        SQLT_STR, &pctx->c_zip_ind, &pctx-
>c_zip_len, &pctx->c_zip_rc);
  OCIBNDR(pctx->curp0, pctx->c_phone_bp,
errhp,":c_phone",c_phone,SIZ(c_phone),
SQLT_STR, &pctx->c_phone_ind, &pctx->c_phone_len,
&pctx->c_phone_rc);
  OCIBNDR(pctx->curp0, pctx->c_since_bp,
errhp,":c_since",c_since,7,
        SQLT_DAT, &pctx->c_since_ind, &pctx-
>c_since_len, &pctx->c_since_rc);
  OCIBNDR(pctx->curp0, pctx->c_credit_bp,
errhp,":c_credit",c_credit,
        SIZ(c_credit),SQLT_CHR, &pctx->c_credit_ind,
&pctx->c_credit_len,
        &pctx->c_credit_rc);
```

```
  OCIBNDR(pctx->curp0, pctx->c_credit_lim_bp,
errhp,":c_credit_lim",
        ADR(c_credit_lim),SIZ(int), SQLT_INT, &pctx-
>c_credit_lim_ind,
        &pctx->c_credit_lim_len, &pctx-
>c_credit_lim_rc);
  OCIBNDR(pctx->curp0, pctx->c_discount_bp,
errhp,":c_discount",
        ADR(c_discount),SIZ(int), SQLT_INT, &pctx-
>c_discount_ind,
        &pctx->c_discount_len, &pctx->c_discount_rc);
  OCIBNDR(pctx->curp0, pctx->c_balance_bp,
errhp,":c_balance",ADR(c_balance),
        SIZ(double),SQLT_FLT, &pctx->c_balance_ind,
&pctx->c_balance_len,
        &pctx->c_balance_rc);
  OCIBNDR(pctx->curp0, pctx->c_data_bp,
errhp,":c_data",c_data,SIZ(c_data),
        SQLT_STR, &pctx->c_data_ind, &pctx-
>c_data_len, &pctx->c_data_rc);

  OCIBNDR(pctx->curp0, pctx->h_date_bp,
errhp,":h_date",h_date,SIZ(h_date),
        SQLT_STR, &pctx->h_date_ind, &pctx-
>h_date_len, &pctx->h_date_rc);

  OCIBNDR(pctx->curp0, pctx->retries_bp,
errhp,":retry",ADR(retries),SIZ(int),
        SQLT_INT, &pctx->retries_ind, &pctx-
>retries_len, &pctx->retries_rc);
  OCIBNDR(pctx->curp0, pctx->cr_date_bp,
errhp,":cr_date",ADR(cr_date),
        SIZ(cr_date),SQLT_DAT, &pctx->cr_date_ind,
&pctx->cr_date_len,
        &pctx->cr_date_rc);

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


  OCIBNDR(pctx->curp1, pctx->w_id_bp1,
errhp,":w_id",ADR(w_id),SIZ(int),
        SQLT_INT, &pctx->w_id_ind, &pctx->w_id_len,
&pctx->w_id_rc);
  OCIBNDR(pctx->curp1, pctx->d_id_bp1,
errhp,":d_id",ADR(d_id),SIZ(int),
        SQLT_INT, &pctx->d_id_ind, &pctx->d_id_len,
&pctx->d_id_rc);
  OCIBND(pctx->curp1, pctx->c_w_id_bp1,
errhp,":c_w_id",ADR(c_w_id),SIZ(int),
        SQLT_INT);
  OCIBND(pctx->curp1, pctx->c_d_id_bp1,
errhp,":c_d_id",ADR(c_d_id),SIZ(int),
        SQLT_INT);
  OCIBNDR(pctx->curp1, pctx->c_id_bp1,
errhp,":c_id",ADR(c_id),SIZ(int),
        SQLT_INT, &pctx->c_id_ind, &pctx->c_id_len,
&pctx->c_id_rc);
  OCIBNDR(pctx->curp1, pctx->h_amount_bp1,
errhp,":h_amount",ADR(h_amount),
        SIZ(int),SQLT_INT, &pctx->h_amount_ind,
&pctx->h_amount_len,
        &pctx->h_amount_rc);
  OCIBND(pctx->curp1, pctx->c_last_bp1,
errhp,":c_last",c_last,SIZ(c_last),
        SQLT_STR);
  OCIBNDR(pctx->curp1, pctx->w_street_1_bp1,
errhp,":w_street_1",w_street_1,
        SIZ(w_street_1),SQLT_STR, &pctx-
>w_street_1_ind,
        &pctx->w_street_1_len, &pctx->w_street_1_rc);
  OCIBNDR(pctx->curp1, pctx->w_street_2_bp1,
errhp,":w_street_2",w_street_2,
        SIZ(w_street_2),SQLT_STR, &pctx-
>w_street_2_ind,
        &pctx->w_street_2_len, &pctx->w_street_2_rc);
  OCIBNDR(pctx->curp1, pctx->w_city_bp1,
errhp,":w_city",w_city,SIZ(w_city),
        SQLT_STR, &pctx->w_city_ind, &pctx-
>w_city_len, &pctx->w_city_rc);
  OCIBNDR(pctx->curp1, pctx->w_state_bp1,
errhp,":w_state",w_state,SIZ(w_state),
        SQLT_STR, &pctx->w_state_ind, &pctx-
>w_state_len, &pctx->w_state_rc);
  OCIBNDR(pctx->curp1, pctx->w_zip_bp1,
errhp,":w_zip",w_zip,SIZ(w_zip),
        SQLT_STR, &pctx->w_zip_ind, &pctx-
>w_zip_len, &pctx->w_zip_rc);
  OCIBNDR(pctx->curp1, pctx->d_street_1_bp1,
errhp,":d_street_1",d_street_1,
        SIZ(d_street_1),SQLT_STR, &pctx-
>d_street_1_ind,
        &pctx->d_street_1_len, &pctx->d_street_1_rc);
  OCIBNDR(pctx->curp1, pctx->d_street_2_bp1,
errhp,":d_street_2",d_street_2,
        SIZ(d_street_2),SQLT_STR, &pctx-
>d_street_2_ind,
        &pctx->d_street_2_len, &pctx->d_street_2_rc);
  OCIBNDR(pctx->curp1, pctx->d_city_bp1,
errhp,":d_city",d_city,SIZ(d_city),
        SQLT_STR, &pctx->d_city_ind, &pctx-
>d_city_len, &pctx->d_city_rc);
  OCIBNDR(pctx->curp1, pctx->d_state_bp1,
errhp,":d_state",d_state,
        SIZ(d_state), SQLT_STR, &pctx->d_state_ind,
&pctx->d_state_len,
        &pctx->d_state_rc);
  OCIBNDR(pctx->curp1, pctx->d_zip_bp1,
errhp,":d_zip",d_zip,SIZ(d_zip),
        SQLT_STR, &pctx->d_zip_ind, &pctx-
>d_zip_len, &pctx->d_zip_rc);
  OCIBNDR(pctx->curp1, pctx->c_first_bp1,
errhp,":c_first",c_first,
        SIZ(c_first), SQLT_STR, &pctx->c_first_ind,
&pctx->c_first_len,
        &pctx->c_first_rc);
  OCIBNDR(pctx->curp1, pctx->c_middle_bp1,
errhp,":c_middle",c_middle,2,
        SQLT_AFC, &pctx->c_middle_ind, &pctx-
>c_middle_len,
        &pctx->c_middle_rc);

  OCIBNDR(pctx->curp1, pctx->c_street_1_bp1,
errhp,":c_street_1",c_street_1,
        SIZ(c_street_1),SQLT_STR, &pctx-
>c_street_1_ind,
        &pctx->c_street_1_len, &pctx->c_street_1_rc);
  OCIBNDR(pctx->curp1, pctx->c_street_2_bp1,
errhp,":c_street_2",c_street_2,
        SIZ(c_street_2),SQLT_STR, &pctx-
>c_street_2_ind,
        &pctx->c_street_2_len, &pctx->c_street_2_rc);
  OCIBNDR(pctx->curp1, pctx->c_city_bp1,
errhp,":c_city",c_city,
                SIZ(c_city),SQLT_STR,
        &pctx->c_city_ind, &pctx->c_city_len,
&pctx->c_city_rc);
  OCIBNDR(pctx->curp1, pctx->c_state_bp1,
errhp,":c_state",c_state,
                SIZ(c_state),SQLT_STR, &pctx-
>c_state_ind, &pctx->c_state_len,
                &pctx->c_state_rc);
  OCIBNDR(pctx->curp1, pctx->c_zip_bp1,
errhp,":c_zip",c_zip,SIZ(c_zip),
        SQLT_STR, &pctx->c_zip_ind, &pctx-
>c_zip_len, &pctx->c_zip_rc);
  OCIBNDR(pctx->curp1, pctx->c_phone_bp1,
errhp,":c_phone",c_phone,
                SIZ(c_phone), SQLT_STR, &pctx-
>c_phone_ind, &pctx->c_phone_len,
                &pctx->c_phone_rc);
  OCIBNDR(pctx->curp1, pctx->c_since_bp1,
errhp,":c_since",c_since,7,
        SQLT_DAT, &pctx->c_since_ind, &pctx-
>c_since_len, &pctx->c_since_rc);
  OCIBNDR(pctx->curp1, pctx->c_credit_bp1,
errhp,":c_credit",c_credit,
        SIZ(c_credit),SQLT_CHR, &pctx->c_credit_ind,
&pctx->c_credit_len,
        &pctx->c_credit_rc);
  OCIBNDR(pctx->curp1, pctx->c_credit_lim_bp1,
errhp,":c_credit_lim",
        ADR(c_credit_lim),SIZ(int), SQLT_INT, &pctx-
>c_credit_lim_ind,
        &pctx->c_credit_lim_len, &pctx-
>c_credit_lim_rc);
  OCIBNDR(pctx->curp1, pctx->c_discount_bp1,
errhp,":c_discount",
        ADR(c_discount),SIZ(int), SQLT_INT, &pctx-
>c_discount_ind,
        &pctx->c_discount_len, &pctx->c_discount_rc);
  OCIBNDR(pctx->curp1, pctx->c_balance_bp1,
errhp,":c_balance",ADR(c_balance),
        SIZ(double),SQLT_FLT, &pctx->c_balance_ind,
&pctx->c_balance_len,
        &pctx->c_balance_rc);
  OCIBNDR(pctx->curp1, pctx->c_data_bp1,
errhp,":c_data",c_data,SIZ(c_data),
        SQLT_STR, &pctx->c_data_ind, &pctx-
>c_data_len, &pctx->c_data_rc);

  OCIBNDR(pctx->curp1, pctx->h_date_bp1,
errhp,":h_date",h_date,SIZ(h_date),
        SQLT_STR, &pctx->h_date_ind, &pctx-
>h_date_len, &pctx->h_date_rc);

  OCIBNDR(pctx->curp1, pctx->retries_bp1,
errhp,":retry",ADR(retries),SIZ(int),
        SQLT_INT, &pctx->retries_ind, &pctx-
>retries_len, &pctx->retries_rc);
  OCIBNDR(pctx->curp1, pctx->cr_date_bp1,
errhp,":cr_date",ADR(cr_date),
        SIZ(cr_date),SQLT_DAT, &pctx->cr_date_ind,
&pctx->cr_date_len,
        &pctx->cr_date_rc);

  return (0);

}



plpay ()
```

```
{

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

  if(bylastname) {

    execstatus=OCIStmtExecute(tpcsvc,pctx-
>curp1,errhp,1,0,0,0,OCI_DEFAULT);
  } else {
    execstatus=OCIStmtExecute(tpcsvc,pctx-
>curp0,errhp,1,0,0,0,OCI_DEFAULT);
  }
  if(execstatus != OCI_SUCCESS) {
     OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
     errcode = OCIERROR(errhp,execstatus);
     if(errcode == NOT_SERIALIZABLE) {
        retries++;
        goto retry;
     } else if (errcode == RECOVERR) {
        retries++;
        goto retry;
     } else {
        return -1;
     }
  }
  return (0);

}



void plpaydone ()

{
  if(pctx) {
    free(pctx);
  }
}
```

plsto.c

```
#ifdef RCSID

static char *RCSid =

   "$Header: plsto.c 7010000.3 95/02/14 12:48:03 plai
Generic<base> $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=========================================
========================+
|       Copyright (c) 1994  Oracle Corp, Redwood
Shores, CA      |
|                OPEN SYSTEMS PERFORMANCE
GROUP               |
|                    All Rights Reserved                    |

+=========================================
========================+
| FILENAME
|    plsto.c
| DESCRIPTION
|    OCI version of STOCK LEVEL transaction in TPC-C
benchmark.

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

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


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

#define SQLTXT1 "alter session set isolation_level =
read commited"
#define SQLTXT2 "alter session set isolation_level =
serializable"

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


struct stoctx {
  OCIStmt *curs;
  OCIBind *w_id_bp;
  OCIBind *d_id_bp;
  OCIBind *threshold_bp;
  OCIDefine *low_stock_bp;

  int norow;
};

typedef struct stoctx stoctx;

stoctx *sctx;
OCIStmt *curs1;
OCIStmt *curs2;


plstoinit ()

{

   text stmbuf[SQL_BUF_SIZE];
   sctx = (stoctx *)malloc(sizeof(stoctx));
   memset(sctx,(char)0,sizeof(stoctx));

   sctx->norow=0;

   OCIERROR(errhp,
      OCIHandleAlloc(tpcenv,(dvoid**)&sctx-
>curs,OCI_HTYPE_STMT,0,(dvoid**)0));
   sprintf ((char *) stmbuf, SQLTXT);
   OCIERROR(errhp,OCIStmtPrepare(sctx-
>curs,errhp,stmbuf,strlen((char *)stmbuf),

OCI_NTV_SYNTAX,OCI_DEFAULT));
   OCIERROR(errhp,
      OCIAttrSet(sctx-
>curs,OCI_HTYPE_STM,(dvoid*)&sctx->norow,0,
            OCI_ATTR_PREFETCH_ROWS,errhp));
#ifdef TECH
   OCIERROR(errhp,
```

```
OCIHandleAlloc(tpcenv,(dvoid**)&curs1,OCI_HTYPE_S
TMT,0,(dvoid**)0));
  sprintf ((char *) stmbuf, SQLTXT1);

OCIERROR(errhp,OCIStmtPrepare(curs1,errhp,stmbuf,
strlen((char *)stmbuf),

OCI_NTV_SYNTAX,OCI_DEFAULT));

  OCIERROR(errhp,

OCIHandleAlloc(tpcenv,(dvoid**)&curs2,OCI_HTYPE_S
TMT,0,(dvoid**)0));
  sprintf ((char *) stmbuf, SQLTXT2);

OCIERROR(errhp,OCIStmtPrepare(curs2,errhp,stmbuf,
strlen((char *)stmbuf),

OCI_NTV_SYNTAX,OCI_DEFAULT));
#endif


  /* bind variables */

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

  return (0);

}



plsto ()

{
retry:
#ifdef TECH
 execstatus=
   OCIStmtExecute(tpcsvc,curs1,errhp,1,0,0,0,
             OCI_COMMIT_ON_SUCCESS |
OCI_DEFAULT);
 if(execstatus != OCI_SUCCESS) {
   OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
   errcode = OCIERROR(errhp,execstatus);
   if(errcode == NOT_SERIALIZABLE) {
     retries++;
     goto retry;
   } else if (errcode == RECOVERR) {
     retries++;
     goto retry;
   } else {
     return -1;
   }
#endif
 execstatus=
   OCIStmtExecute(tpcsvc,sctx->curs,errhp,1,0,0,0,
             OCI_COMMIT_ON_SUCCESS |
OCI_DEFAULT);
 if(execstatus != OCI_SUCCESS) {
   OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
   errcode = OCIERROR(errhp,execstatus);
   if(errcode == NOT_SERIALIZABLE) {
     retries++;
     goto retry;
   } else if (errcode == RECOVERR) {
     retries++;
     goto retry;
   } else {
     return -1;
   }
 }

#ifdef TECH
 execstatus=
   OCIStmtExecute(tpcsvc,curs2,errhp,1,0,0,0,
             OCI_COMMIT_ON_SUCCESS |
OCI_DEFAULT);
 if(execstatus != OCI_SUCCESS) {
   OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
   errcode = OCIERROR(errhp,execstatus);
   if(errcode == NOT_SERIALIZABLE) {
     retries++;
     goto retry;
   } else if (errcode == RECOVERR) {
     retries++;
     goto retry;
   } else {
     return -1;
   }
#endif
 return (0);
}



void plstodone ()

{
 if(sctx) free(sctx);
#ifdef TECH
 if(cur1) free(cur1);
 if(cur2) free(cur2);
#endif
}
```

tpccpl.c

```
#ifdef RCSID

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


/*=========================================
========================+
|      Copyright (c) 1994  Oracle Corp, Redwood
Shores, CA      |
|                OPEN SYSTEMS PERFORMANCE
GROUP               |
|                   All Rights Reserved                   |

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

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

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

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

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;

int execstatus;
int errcode;

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


/* for stock-level transaction */

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

/* for delivery transaction */

int del_o_id[10];
int retries;

/* for order-status transaction */

int bylastname;
char c_last[17];
char c_first[17];
char c_middle[3];
double c_balance;
int o_id;
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];
int ol_amount[15];
char ol_delivery_d[15][11];

/* for payment transaction */

int c_w_id;
int c_d_id;
int h_amount;
char w_street_1[21];
char w_street_2[21];
char w_city[21];
char w_state[3];
char w_zip[10];
char d_street_1[21];
char d_street_2[21];
char d_city[21];
char d_state[3];
char d_zip[10];
char c_street_1[21];
char c_street_2[21];
char c_city[21];
char c_state[3];
char c_zip[10];
char c_phone[17];
char c_since_d[11];
int c_discount;
/*              TSL Hagino 970912
float c_discount;
*/
char c_credit[3];
int c_credit_lim;
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];
int nol_amount[15];
int o_all_local;
int w_tax;
int d_tax;
float total_amount;
char i_name[15][25];
int s_quantity[15];
char brand_gen[15];
int i_price[15];
int status;

unsigned char cr_date[7];
unsigned char c_since[7];
unsigned char o_entry_d_base[7];
unsigned char ol_d_base[15][7];
dvoid *xmem;

/* NewOrder Binding stuff */


errrpt (lda, cur)

ldadef *lda;
csrdef *cur;
{

  text msg[2048];

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

}


/* vmm313 void ocierror(fname, lineno, errhp, status)  */
int ocierror(fname, lineno, errhp, status)
char *fname;
int lineno;
OCIError *errhp;
sword status;
{
  text errbuf[512];
  ub4 buflen;
  sb4 errcode;

  switch (status) {
  case OCI_SUCCESS:
    break;
  case OCI_SUCCESS_WITH_INFO:
    (void) printf("Module %s Line %d\n", fname, lineno);
    (void) printf("Error -
OCI_SUCCESS_WITH_INFO\n");
  break;
  case OCI_NEED_DATA:
    (void) printf("Module %s Line %d\n", fname, lineno);
    (void) printf("Error - OCI_NEED_DATA\n");
  break;
  case OCI_NO_DATA:
    (void) printf("Module %s Line %d\n", fname, lineno);
    (void) printf("Error - OCI_NO_DATA\n");
    return IRRECERR;
                /* added 1998/01/20 */
  break;
  case OCI_ERROR:
    (void) OCIErrorGet (errhp, (ub4) 1,
                        (text *) NULL, &errcode,
errbuf,
                                    (ub4)
sizeof(errbuf), OCI_HTYPE_ERR);
    if (errcode != 8177)
    {
      (void) printf("Module %s Line %d\n", fname,
lineno);
      (void) printf("Error - %s\n", errbuf);
    }
    if ( errcode == SNAPSHOT_TOO_OLD ) {
      return RECOVERR;
    }
    return errcode;
/* vmm313    TPCexit(1); */
/* vmm313    exit(1); */
  break;
  case OCI_INVALID_HANDLE:
    (void) printf("Module %s Line %d\n", fname, lineno);
    (void) printf("Error - OCI_INVALID_HANDLE\n");
    TPCexit(1);
    exit(-1);
  break;
  case OCI_STILL_EXECUTING:
    (void) printf("Module %s Line %d\n", fname, lineno);
    (void) printf("Error - OCI_STILL_EXECUTE\n");
  break;
  case OCI_CONTINUE:
    (void) printf("Module %s Line %d\n", fname, lineno);
    (void) printf("Error - OCI_CONTINUE\n");
  break;
  default:
    break;
  }
  return RECOVERR;
}


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

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

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

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

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

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

/*
void
vgetdate(unsigned char *buf) {
```

```
  time_t tloc;
  char temp[5];
  int cen;

  if(time(&tloc) == (time_t)-1) {
    printf("Error getting date\n");
    exit(1);
  }
  cftime(temp,"%d",&tloc);
  buf[3] = (unsigned char)atoi(temp);
  cftime(temp,"%m",&tloc);
  buf[2] = (unsigned char)atoi(temp);
  cftime(temp,"%Y",&tloc);
  cen = atoi(temp);
  buf[0] = (unsigned char)((cen/100)+100);
  buf[1] = (unsigned char)((cen%100)+100);
  cftime(temp,"%H",&tloc);
  buf[4] = (unsigned char)(atoi(temp) + 1);
  cftime(temp,"%M",&tloc);
  buf[5] = (unsigned char)(atoi(temp) + 1);
  cftime(temp,"%S",&tloc);
  buf[6] = (unsigned char)(atoi(temp) + 1);
}
*/
void vgetdate (unsigned char *oradt)
{
 struct tm *loctime;
 time_t  int_time;

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

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

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

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

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

Date.century = (unsigned char)(century + 100);
if (Date.century < 119 || Date.century > 120) cnvrtOK =
0;
Date.year  = (unsigned char)(loctime->tm_year+100);
if (Date.year < 100 || Date.year > 199) cnvrtOK = 0;
Date.month = (unsigned char)(loctime->tm_mon + 1);
if (Date.month < 1 || Date.month > 12) cnvrtOK = 0;
Date.day   = (unsigned char)loctime->tm_mday;
if (Date.day < 1 || Date.day > 31) cnvrtOK = 0;
Date.hour  = (unsigned char)(loctime->tm_hour + 1);
if (Date.hour < 1 || Date.hour > 24) cnvrtOK = 0;
Date.minute= (unsigned char)(loctime->tm_min + 1);
if (Date.minute < 1 || Date.minute > 60) cnvrtOK = 0;
Date.second= (unsigned char)(loctime->tm_sec + 1);
if (Date.second < 1 || Date.second > 60) cnvrtOK = 0;

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

 return;

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

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


       int day,month,year;

       memcpy(&Date,oradt,7);

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

       return;
}

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

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


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

       memcpy(&Date,oradt,7);

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

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

       return;
}


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;
  }


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

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

#ifdef TUX
  userlog( "Something is wrong. Bye :-p\n" );
#else
  fprintf (stderr, "Something is wrong. Bye :-p\n" );
#endif

}



TPCinit (id, uid, pwd)

int id;
char *uid;
char *pwd;

{

  int i;
  char filename[40];
```

```
  text stmbuf[100];
char tmp[1024];

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

  OCIInitialize(OCI_DEFAULT,(dvoid *)0,0,0,0);
  OCIEnvInit(&tpcenv, OCI_DEFAULT, 0, (dvoid **)0);
  OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&tpcsrv,
OCI_HTYPE_SERVER, 0 , (dvoid **)0);
  OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&errhp,
OCI_HTYPE_ERROR, 0 , (dvoid **)0);
  OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&tpcsvc,
OCI_HTYPE_SVCCTX, 0 , (dvoid **)0);
  OCIServerAttach(tpcsrv, errhp, (text
*)0,0,OCI_DEFAULT);
  OCIAttrSet((dvoid *)tpcsvc, OCI_HTYPE_SVCCTX,
(dvoid *)tpcsrv, (ub4)0,OCI_ATTR_SRVRCTXT, errhp);
  OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&tpcusr,
OCI_HTYPE_SESSION, 0 , (dvoid **)0);
  OCIAttrSet((dvoid *)tpcusr, OCI_HTYPE_SESSION,
(dvoid *)uid, (ub4)strlen(uid),OCI_ATTR_USERNAME,
errhp);
  OCIAttrSet((dvoid *)tpcusr, OCI_HTYPE_SESSION,
(dvoid *)pwd, (ub4)strlen(pwd),
                OCI_ATTR_PASSWORD, errhp);
  OCIERROR(errhp, OCISessionBegin(tpcsvc, errhp,
tpcusr, OCI_CRED_RDBMS, OCI_DEFAULT));

  OCIAttrSet(tpcsvc, OCI_HTYPE_SVCCTX, tpcusr, 0,
OCI_ATTR_USERCTXT, errhp);

  /* run all transaction in serializable mode */

  OCIHandleAlloc(tpcenv, (dvoid **)&curi,
OCI_HTYPE_STMT, 0, (dvoid**)0);
  sprintf ((char *) stmbuf, SQLTXT);
  OCIStmtPrepare(curi, errhp, stmbuf, strlen((char
*)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT);
  OCIERROR(errhp,OCIStmtExecute(tpcsvc, curi,
errhp,1,0,0,0,OCI_DEFAULT));
  OCIHandleFree(curi, OCI_HTYPE_STMT);


#ifdef SQL_TRACE
  /* Turn on the SQL_TRACE */
  OCIHandleAlloc(tpcenv, (dvoid **)&curi,
OCI_HTYPE_STMT, 0, &xmem);
  sprintf ((char *) stmbuf, SQLTXT1);
  OCIStmtPrepare(curi, errhp, stmbuf, strlen((char
*)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT);
  OCIERROR(errhp, OCIStmtExecute(tpcsvc, curi,
errhp,1,0,0,0,OCI_DEFAULT));
  OCIHandleFree((dvoid *)curi, OCI_HTYPE_STMT);
#endif /* End SQL_TRACE */
  logon = 1;
  vgetdate(cr_date);


  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;
  vgetdate(cr_date);

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

  /* fill in date for o_entry_d from time in beginning of
txn*/
  cvtdmyhms(cr_date,o_entry_d);

  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 = (float)(c_discount)/10000;
  str->newout.w_tax = (float)(w_tax)/10000;
  str->newout.d_tax = (float)(d_tax)/10000;
  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] = (float)(i_price[i])/100;
    str->newout.ol_amount[i] =
(float)(nol_amount[i])/100;
  }
  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;

  vgetdate(cr_date);

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

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

```
  cvtdmyhms(cr_date,h_date);
  cvtdmy(c_since,c_since_d);


  str->payout.terror = NOERR;
  strncpy (str->payout.w_street_1, w_street_1, 21);
  strncpy (str->payout.w_street_2, w_street_2, 21);
  strncpy (str->payout.w_city, w_city, 21);
  strncpy (str->payout.w_state, w_state, 3);
  strncpy (str->payout.w_zip, w_zip, 10);
  strncpy (str->payout.d_street_1, d_street_1, 21);
  strncpy (str->payout.d_street_2, d_street_2, 21);
  strncpy (str->payout.d_city, d_city, 21);
  strncpy (str->payout.d_state, d_state, 3);
  strncpy (str->payout.d_zip, d_zip, 10);
  str->payout.c_id = c_id;
  strncpy (str->payout.c_first, c_first, 17);
  strncpy (str->payout.c_middle, c_middle, 3);
  strncpy (str->payout.c_last, c_last, 17);
  strncpy (str->payout.c_street_1, c_street_1, 21);
  strncpy (str->payout.c_street_2, c_street_2, 21);
  strncpy (str->payout.c_city, c_city, 21);
  strncpy (str->payout.c_state, c_state, 3);
  strncpy (str->payout.c_zip, c_zip, 10);
  strncpy (str->payout.c_phone, c_phone, 17);
  strncpy (str->payout.c_since, c_since_d, 11);
  strncpy (str->payout.c_credit, c_credit, 3);
  str->payout.c_credit_lim = (float)(c_credit_lim)/100;
  str->payout.c_discount = (float)(c_discount)/10000;
  str->payout.c_balance = (float)(c_balance)/100;
  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/100;
  str->ordout.o_id = o_id;
  strncpy (str->ordout.o_entry_d, o_entry_d, 20);
  if ( o_carrier_id == 11 )
     str->ordout.o_carrier_id = 0;
  else
     str->ordout.o_carrier_id = o_carrier_id;
  str->ordout.o_ol_cnt = o_ol_cnt;
  for (i = 0; i < o_ol_cnt; i++) {
    ol_delivery_d[i][10] = '\0';
    if ( !strcmp(ol_delivery_d[i],"15-09-1911") )
                strncpy(ol_delivery_d[i],"NOT
DELIVR",10);
    str->ordout.ol_supply_w_id[i] = ol_supply_w_id[i];
    str->ordout.ol_i_id[i] = ol_i_id[i];
    str->ordout.ol_quantity[i] = ol_quantity[i];
    str->ordout.ol_amount[i] = (float)(ol_amount[i])/100;
    strncpy (str->ordout.ol_delivery_d[i],
ol_delivery_d[i], 11);
  }
  str->ordout.retry = retries;
  return (0);

}



TPCdel (str)

struct delstruct *str;

{
/* [ADD] 1997.09.03 CREO)yamamoto */
  struct timeval tp,tp_e;
/* [END] */

  double tr_end;
  int i;

  w_id = str->delin.w_id;
  o_carrier_id = str->delin.o_carrier_id;
  retries = 0;
  vgetdate(cr_date);

  if (str->delout.terror = pldel ()) {
    if(str->delout.terror == DEL_ERROR)
      return DEL_ERROR;
    if (str->delout.terror != RECOVERR)
      str->delout.terror = IRRECERR;
    return (-1);
  }

/* update 1997.09.03 CREO)yamamoto
  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);
*/
  gettimeofday(&tp_e);
  fprintf (lfp, "%09d%03d %09d%03d %d %d", str-
>delin.qtime, str->delin.uqtime/1000, tp_e.tv_sec,
tp_e.tv_usec/1000, 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
    }
  }
/* update 1997.09.03 CREO)yamamoto
  fprintf (lfp, " %d\n", retries);
*/
  fprintf (lfp, " \n");

  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);

}
```

tpccsvr.c

```
#ifdef RCSID

static char *RCSid =

  "$Header: tpccsvr.c 7030100.1 95/07/19 15:39:28 plai
Generic<base> $ Copyr (c) 1995 Oracle";

#endif /* RCSID */



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

|      Copyright (c) 1995  Oracle Corp, Redwood
Shores, CA      |
|                OPEN SYSTEMS PERFORMANCE
GROUP              |
```

```
|                    All Rights Reserved                    |

+==========================================
=========================+
| FILENAME
|    tpccsvr.c
| DESCRIPTION
|    Tuxedo server for TPC-C.

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

#include <stdio.h>
#include "tpcc.h"
#include "tpcc_info.h"
#include <atmi.h>
#include <userlog.h>
#include <string.h>         /* for strcpy() (1997.10.08) */

/* [ADD] 1997.09.03 CREO)yamamoto */
#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
/* [END] */

    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;
   char *pwd;
   char         tmp[1024];

   if (argc >= 2) {
   /*
                sprintf( tmp, "init: %d %x\n", argc, argv );
                userlog( tmp );
                for ( id = 0; id < argc; id++ ) {
                        sprintf( tmp, "--> %d: '%s'\n",
id, argv[id] );
                        userlog( tmp );
                }
   */
      id = atoi (argv[argc-2]);
      strcpy( tmp, argv[argc-1] );
      pwd = uid = tmp;
      while ( *pwd++ != '/' )
         ;
      *( pwd-1 ) = '\0';
   /*
                sprintf( tmp, "args: %d, '%s', '%s'\n", id,  uid,
pwd );
                userlog( tmp );
   */
      return (TPCinit (id, uid, pwd));
   }
   else {
      userlog ("Error: not enough arguments to
tpsvrinit\n");
      return (-1);
   }
}



void tpsvrdone ()
{
   TPCexit ();
}


/* [ADD] 1997.09.03 CREO)yamamoto */
#ifdef FJ
/*    40 entries */

TPCC01(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;
    }
}

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, 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;
   }
}

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, 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;
  }
}

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, stoinfo, sizeof (struct
stostruct), 0);
    else
      tpreturn (TPSUCCESS, 0, 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 (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;
   }
}

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, stoinfo, sizeof (struct
stostruct), 0);
      else
         tpreturn (TPSUCCESS, 0, 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, stoinfo, sizeof (struct
stostruct), 0);
      else
         tpreturn (TPSUCCESS, 0, 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, stoinfo, sizeof (struct
stostruct), 0);
      else
         tpreturn (TPSUCCESS, 0, 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, stoinfo, sizeof (struct
stostruct), 0);
      else
         tpreturn (TPSUCCESS, 0, 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, stoinfo, sizeof (struct
stostruct), 0);
      else
         tpreturn (TPSUCCESS, 0, 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, stoinfo, sizeof (struct
stostruct), 0);
      else
         tpreturn (TPSUCCESS, 0, 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, stoinfo, sizeof (struct
stostruct), 0);
        else
            tpreturn (TPSUCCESS, 0, 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, stoinfo, sizeof (struct
stostruct), 0);
        else
            tpreturn (TPSUCCESS, 0, stoinfo, sizeof (struct
stostruct), 0);
        break;
    }
}
TPCC26(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;
    }
}
TPCC27(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;
    }
}
TPCC28(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;
   }
}
TPCC29(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;
   }
}
TPCC30(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;
   }
}
TPCC31(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;
   }
}
TPCC32(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;
   }
}
TPCC33(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;
   }
}
TPCC34(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;
   }
}
TPCC35(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;
      }
}
TPCC36(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;
   }
}
TPCC37(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;
   }
}
TPCC38(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;
   }
}
TPCC39(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;
   }
}
TPCC40(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;
   }
}
#else
/* [END] */

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
```

Appendix C: RTE Scripts

```
#
#  tpcC.conf : configration file for TPC-C
#
#

STARTGROUP = sync , 1
        STARTRTE
                RTEHOST    = rte02
                STARTSUT
                        SUTHOST
= cl02a,40
                        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,8501
#WAREHOUSE SCALE
        VAL             = U1I = 3000
#RAMP-UP TIME
        VAL             = U2I = 0
#MEASUERMENT TIME
        VAL             = U3I = 7200
#RAMP-DOWN TIME
        VAL             = U4I = 0
#NEW THINKTIME (msec)
        VAL             = U5I = 12040
#PAY THINKTIME (msec)
        VAL             = U6I = 12040
#
        VAL             = U7I = 0
        VAL             = U8I = 0
        VAL             = U9I = 0
#
#ORD THINKTIME (msec)
        VAL             = U10I = 10190
#DEL THINKTIME (msec)
        VAL             = U11I = 5040
#STK THINKTIME (msec)
        VAL             = U12I = 5040
#NURAND CONSTANT c_id
        VAL             = U13I = 511
#NURAND CONSTANT c_last
        VAL             = U14I = 87
#NURAND CONSTANT ol_i_id
        VAL             = U15I = 4093
#MSG  OFF:0, Each Term:1, Field:2
        VAL             = U16I= 0
#NEW KEYING-TIME (msec)
        VAL             = U17I = 18100
#PAY KEYING-TIME (msec)
        VAL             = U18I = 3050
#ORD KEYING-TIME (msec)
        VAL             = U19I = 2050
#DEL KEYING-TIME (msec)
        VAL             = U20I = 2050
#STK KEYING-TIME (msec)
        VAL             = U21I = 2050
ENDGROUP
```

Appendix D: System Tunables

```
###############
#### affs ####
###############
*ident "@(#)affs          4.3 93/01/07 16:41:12 -
EIWA/SCCS"
*
* AFFS( <-cp- 3b2/fs/vxfs/affs.cf  FLAG = orxj )
*
*FLAG       #VEC        PREFIX      SOFT
            #DEV        IPL
            DEPENDENCIES/VARIABLES
oxj         -           vx_         -           -
            -
            vx_name(%15c)           =
{VXFSNAME}
            vxfs_ninode(%l)         =
{VXFSNINODE}
            vxfs_trap_ibaddirerr(%i) =
{VXFS_TRAP_IBADDIRERR}
            VXFSNINODEcurve
(%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%
i) = {
                        32, 1024,
TV_LINEAR,
                        64, 2048,
TV_LINEAR,
                        128, 4096,
TV_LINEAR,
                        256, 8192,
TV_LINEAR,
                        512, 8192,
TV_LINEAR,
                        1024, 8192,
TV_LINEAR,
                        2048, 8192,
TV_LINEAR }
            VXFSNINODEcurvesz (%i) = { 7 }
$$$
VXFSNINODE          = 0
VXFS_TRAP_IBADDIRERR        = 0
VXFSNAME            = "affs"



###############
#### 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 = 4096
###############
#### 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%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 }
            usrdatarsvmem (%i) = {
USRDATARSVMEM }
            usrdatarsvmin (%i) = {
USRDATARSVMIN }
$$

************************************************************
**********
* 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 = 4096
NLWP = 6144
MAXUP = 4096
ARG_MAX = 1048576
FLCKREC = 16384

**********************************************************
**********************
* 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 = 0xFFFFFFFFFFFF
HDATLIM = 0xFFFFFFFFFFFF
SSTKLIM = 0x1000000
HSTKLIM = 0x1000000
SCORLIM = 0x7FFFFFFF
HCORLIM = 0x7FFFFFFF
SFNOLIM = 0x4000
HFNOLIM = 0x4000
SVMMLIM = 0xFFFFFFFFFFFF
HVMMLIM = 0xFFFFFFFFFFFF

**********************************************************
**********************
* 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 = 1
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 = 0

**********************************************************
**********************
* 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 = 6144
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 = 64
```

```
USRBSSRSVMEM = 32
BRKRSVMEM = 0
USRTXTRSVMIN = 4
USRBSSRSVMIN = 4
BRKRSVMIN = 4
SEGKMEM_LOCK_PAGE = 400
USRDATARSVMEM = 32
*USRDATARSVMEM = 4096
USRDATARSVMIN = 4
###############
#### sem ####
###############
* All Rights Reserved, Copyright (c) PFU &
FUJITSU LIMITED 1996
*#ident "@(#)sem.cf       6.1.2.1 96/10/22 DS"
*
* SEM
*
*FLAG      #VEC       PREFIX     SOFT
          #DEV       IPL
          DEPENDENCIES/VARIABLES
ox        -          sem        -          -
          -          ipc


          seminfo(%i%i%i%i%i%i%i%i%i)

                     ={SEMMAP,

                               SEMMNI,

                               SEMMNS,

                               SEMMNU,

                               SEMMSL,

                               SEMOPM,

                               SEMUME,


          16+8*SEMUME,

                               SEMVMX,

                               SEMAEM}

$$$

SEMMAP = 50
SEMMNI = 4096
SEMMNS = 256
SEMMNU = 40
SEMMSL = 512
SEMOPM = 512
SEMUME = 512
SEMVMX = 32767
SEMAEM = 16384
###############
#### sfs ####
###############
*      Copyright (c) 1993 UNIX System
Laboratories, Inc.
*      (a wholly-owned subsidiary of Novell, Inc.).
*      All Rights Reserved.
*      THIS IS UNPUBLISHED PROPRIETARY
SOURCE CODE OF UNIX SYSTEM
*      LABORATORIES, INC. (A WHOLLY-OWNED
SUBSIDIARY OF NOVELL, INC.).
*      The copyright notice above does not
evidence any actual or
*      intended publication of such source code.

*             All Rights Reserved, Copyright (c)
PFU & FUJITSU LIMITED 1993

*#ident      "@(#)sfs.cf  6.1.5.1 12/3/96 DS"
*
* SFS
*
*FLAG      #VEC       PREFIX     SOFT
          #DEV       IPL
          DEPENDENCIES/VARIABLES
orxj      -          sfs        -          -
          -


          sfs_ninode(%i) = {SFSNINODE}

          sfs_inode_lwm(%i) =
{SFSINODELWM}

          sfs_timelag(%i) = {SFSTIMELAG}

          sfs_tflush(%i) = {SFSFLUSH}

          sfs_dirlink(%i) = {SFSDIRLINK}

          sfs_ndquot(%i) = {NDQUOT}

          SFSNINODEcurve(%i%i%i%i%i%i%i%
i%i%i%i%i%i%i%i%i%i%i%i%i%i%i) = {

                     32, 1024, TV_LINEAR,

                     64, 2048, TV_LINEAR,

                     128, 4096, TV_LINEAR,

                     256, 8192, TV_LINEAR,

                     512, 16384, TV_LINEAR,

                     1024, 32768,
TV_LINEAR,

                     2048, 32768,
TV_LINEAR}

          SFSNINODEcurvesz(%i) = {7}
$$
SFSNINODE = 1024
SFSINODELWM = 200
SFSTIMELAG = 10000
SFSFLUSH = 60
SFSDIRLINK = 0
NDQUOT = 200
###############
#### 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(%ll%ll%i%i)

                          ={SHMMAX,

                                     SHMMIN,

                                     SHMMNI,

                                     SHMSEG}


          shmrsvmem(%i)={SHMRSVMEM}


          shmrsvmin(%i)={SHMRSVMIN}

                     shmzerothrtim(%i)


={SHMZEROTHRTIM}

$$$

SHMMAX = 0xFFFFFFFFFFFF
SHMMIN = 1
SHMMNI = 40000
SHMSEG = 40000
SHMRSVMEM = 14000
*SHMRSVMEM = 16544
*SHMRSVMEM = 10496
SHMRSVMIN = 16
SHMZEROTHRTIM = 10
###############
#### 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) = {
```

```
                        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}
  is_swapable(%i) = {ENABLE_SWAP}
$$
* 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
* if ENABLE_SWAP>0 then swapper works,
otherwise does not. (default:1)
ENABLE_SWAP = 1
###############
#### 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%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
###############
#### nfs ####
```

```
###############
*#ident	"@(#)nfs.cf	1.1 11 Mar 1994 DS"
*
*
*                               PROPRIETARY NOTICE
(Combined)
*
*  This source code is unpublished proprietary
information
*  constituting, or derived under license from
AT&T's Unix(r) System V.
*
*
*
*                               Copyright Notice
*
*  Notice of copyright on this source code product
does not indicate
*  publication.
*
*               (c) 1986,1987,1988,1989  Sun
Microsystems, Inc.
*               (c)
1983,1984,1985,1986,1987,1988,1989  AT&T.
*                       All rights reserved.
*
*  All Rights Reserved, Copyright (c) PFU &
FUJITSU LIMITED 1993,1994
*
* NFS
*
*FLAG       #VEC        PREFIX      SOFT
               #DEV        IPL
               DEPENDENCIES/VARIABLES
oxj            -           nfs         -           -
               -           KRPC

                           nrnode(%l) = {NRNODE}

                           nnfsmount(%l) =
{NNFSMOUNT}

                           klm_level(%l) =
{KLM_LEVEL}

                           nfs_async_max(%i) =
{NFS_ASYNC_MAX}

                           nfs_async_timeout(%i) =
{NFS_ASYNC_TIMEOUT}

                           nfs_mmap_timeout(%i) =
{NFS_MMAP_TIMEOUT}

                           nfs_maxclients(%i) =
{NFS_MAXCLIENTS}

                           nfsd_timeout(%i) =
{NFSD_TIMEOUT}

                           nfsd_max(%i) =
{NFSD_MAX}

                           nfsd_min(%i) =
{NFSD_MIN}

                           nfs_retries(%i) =
{NFS_RETRIES}

                           nfs_timeo(%i) =
{NFS_TIMEO}

                           nfs_nra(%i) =
{NFS_NRA}


                   chtable[NFS_MAXCLIENTS](%0xc)

                           nfs_cto(%i) = {1}

                           nfs_dnlc(%i) = {1}

                           nfsreadmap(%i) = {1}

                           nfssys(){nosys}
$$$

NRNODE = 300
NNFSMOUNT = 256
KLM_LEVEL = 3
NFS_ASYNC_MAX = 8
NFS_ASYNC_TIMEOUT = 4
NFS_MMAP_TIMEOUT = 30
NFS_MAXCLIENTS = 6
NFSD_TIMEOUT = 8
NFSD_MAX = 8
NFSD_MIN = 2
NFS_RETRIES = 5
NFS_TIMEO = 10
NFS_NRA = 1
###############
#### sv500l1.app ####
###############
[Apps]
;
; APP   Object Path                    Entry Point  Min
Max
; ===  ===========
========== ===    ===
TPCC026
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC027
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC028
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC029
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC030
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC031
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC032
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC033
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC034
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC035
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC036
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC037
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC038
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16

[AppDirs]
;
; Virtual Path         APP     Physical Path
; ============        ===    =============
/sample/wrbsdk/tpcc/tpcc026    TPCC026
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc027    TPCC027
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc028    TPCC028
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc029    TPCC029
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc030    TPCC030
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc031    TPCC031
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc032    TPCC032
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc033    TPCC033
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc034    TPCC034
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc035    TPCC035
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc036    TPCC036
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc037    TPCC037
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc038    TPCC038
/oracle/ows21

[AppProtection]
###############
#### sv500l2.app ####
###############
[Apps]
;
; APP   Object Path                    Entry Point  Min
Max
; ===  ===========
========== ===    ===
TPCC038
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC039
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC040
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC041
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
```

```
TPCC042
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC043
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC044
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC045
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC046
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC047
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC048
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC049
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16
TPCC050
/oracle/ows21/sample/wrbsdk/tpcc/tpcc.so
tpccentry 16 16

[AppDirs]
;
; Virtual Path        APP    Physical Path
; ============        ===    =============
/sample/wrbsdk/tpcc/tpcc038   TPCC038
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc039   TPCC039
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc040   TPCC040
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc041   TPCC041
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc042   TPCC042
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc043   TPCC043
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc044   TPCC044
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc045   TPCC045
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc046   TPCC046
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc047   TPCC047
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc048   TPCC048
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc049   TPCC049
/oracle/ows21
/sample/wrbsdk/tpcc/tpcc050   TPCC050
/oracle/ows21

[AppProtection]
###############
#### sv500l1.cfg ####
###############
[MultiPort]
ANY  8501  NONE  cl02  / /dev/null
ANY  8502  NONE  cl02  / /dev/null
ANY  8503  NONE  cl02  / /dev/null
ANY  8504  NONE  cl02  / /dev/null
ANY  8505  NONE  cl02  / /dev/null
ANY  8506  NONE  cl02  / /dev/null
ANY  8507  NONE  cl02  / /dev/null
ANY  8508  NONE  cl02  / /dev/null
ANY  8509  NONE  cl02  / /dev/null
ANY  8510  NONE  cl02  / /dev/null
ANY  8511  NONE  cl02  / /dev/null
ANY  8512  NONE  cl02  / /dev/null
ANY  8513  NONE  cl02  / /dev/null
[NetInfo]
;MaxConnectCount          = 300
DNSResolution             = NEVER
ServerPID                 =
/oracle/ows21/log/svora.pid
;
;
; Log Information
;
; Set values for ...
;    LogDir                ... Place for all log files
;    LogInfoFile           ... Information (Audit) Log
file
;    LogErrorFile          ... Error Log File
;
[Log]
LogTimeStyle              = GMT
LogErrorFile              =
/oracle/ows21/log/svora.err
;
;
; Server Parameters
;
; Set values for ...
;          RequestTimeout        ... Time out
for incoming requests
;          ResponseTimeout       ... Time out
for outgoing responses
;          ScriptTimeout         ... Time out
for Script response time
;
[Server]
InitialFile       = index.html
DefaultMIMEType   = application/octet-stream
; UserDir          = public_html
; UserDirInitialFile = myinitial.html
DefaultCharset    = iso-8859-1
PreferredLanguage  = en
ImageMap          = map
UseDirIndexing    = FALSE
;CGITimeout        = 1800
KeepAliveTimeout = 20
;
; Secure Information Parameters
;
; Set values for ...
;          UserID                ... Acquire
privileges for User ID
;          GroupID               ... Acquire
privileges for Group ID
;
[SecureInfo]
UserID                    = oracle
GroupID                   = dba
;
;
; Directory Mappings
;
; List each mapping from a physical directory to a
; virtual directory, including the indicator whether
; the directory contains scripts or whether the
; subdirectories should be mapped recursively (R)
or
; non-recursively (N).
;
[DirMaps]
; This first line which has ?/ows21/doc needs to be
here to keep
; the virtual path ordering correct.
/oracle/ows21/doc/    NR     /
/oracle/ows21/admdoc/ NR     /ows-adoc/
/oracle/ows21/admbin/ CR     /ows-abin/
/oracle/ows21/bin/    CR     /ows-bin/
/oracle/ows21/bin/    CN     /owa_dba/
/oracle/ows21/doc/    NR     /ows-doc/
/oracle/ows21/img/    NR     /ows-img/
/oracle/ows21/demo/img/    NR  /tr-img/
/oracle/ows21/sample/ NR     /sample/
/oracle/ows21/sample/bin CR  /sample/bin/
/oracle/ows21/sample/wrbsdk/tpcc/ NR
/sample/wrbsdk/tpcc/
;
;
; File Cache
;
; List the full virtual filename path or wild-card
; expression denoting one or more files in the
Servers
; virtual file system.
;
;[FileCache]
;/products/*
;/products/xr25/Version*.c
;/employees/list/phones
;
;
; Language Extensions
;
; List for each combination of language type
; and character set one or more unique
; extensions.
;
[LangExt]
en                        iso-8859-1
          eng
en                        unicode-1-1
          engU uc
fr-CA                     iso-8859-1
          frc
fr-FR                     iso-8859-1
          fr
jp-JP                     iso-2022-jp
          jp
jp-JP                     unicode-1-1-utf-8
          jpU
;
;
; MIME Types
;
; List for each combination of MIME Type
; and representation one or more unique
; extensions.
;
[MIMETypes]
text/html     htm html
image/jpeg    jpg jpeg
image/gif     gif
```

```
appl/text      doc
text/plain     txt ksh lst
application/postscript ps
application/pdf pdf

;
[Encodings]
compress       Z
gzip           gz
;
[Security]
Basic {
(Users)
admin: oracle
(Groups)
dba: admin
(Realms)
Admin Server: dba
}
;
;
[Protection]
/ows-abin/        Basic(Admin Server)
/       Basic(Admin Server)
/ows-adoc/       Basic(Admin Server)
/ows-bin/       Basic(Admin Server)
/owa_dba/       Basic(Admin Server)
/ows-doc/       Basic(Admin Server)
/ows-img/       Basic(Admin Server)
/tr-img/        Basic(Admin Server)
/sample/        Basic(Admin Server)
/sample/bin/    Basic(Admin Server)

;
[DynApps]
/oracle/ows21/lib/ndwfss.so oracle_adp_init

###############
#### sv500l2.cfg ####
###############
[MultiPort]
ANY  8517  NONE  cl02  / /dev/null
ANY  8518  NONE  cl02  / /dev/null
ANY  8519  NONE  cl02  / /dev/null
ANY  8520  NONE  cl02  / /dev/null
ANY  8521  NONE  cl02  / /dev/null
ANY  8522  NONE  cl02  / /dev/null
ANY  8523  NONE  cl02  / /dev/null
ANY  8524  NONE  cl02  / /dev/null
ANY  8525  NONE  cl02  / /dev/null
ANY  8526  NONE  cl02  / /dev/null
ANY  8527  NONE  cl02  / /dev/null
ANY  8528  NONE  cl02  / /dev/null
ANY  8529  NONE  cl02  / /dev/null
[NetInfo]
;MaxConnectCount              =  300
DNSResolution                 =  NEVER
ServerPID                  =
/oracle/ows21/log/svora.pid
;
;
; Log Information
;
; Set values for ...
;     LogDir                ... Place for all log files
;     LogInfoFile           ... Information (Audit) Log
file
;     LogErrorFile          ... Error Log File
;
[Log]
LogTimeStyle                  = GMT
LogErrorFile                  =
/oracle/ows21/log/svora.err
;
;
; Server Parameters
;
; Set values for ...
;               RequestTimeout          ... Time out
for incoming requests
;               ResponseTimeout         ... Time out
for outgoing responses
;               ScriptTimeout           ... Time out
for Script response time
;
[Server]
InitialFile       = index.html
DefaultMIMEType    = application/octet-stream
; UserDir          = public_html
; UserDirInitialFile = myinitial.html
DefaultCharset     = iso-8859-1
PreferredLanguage  = en
ImageMap          = map
UseDirIndexing     = FALSE
;CGITimeout        = 1800
KeepAliveTimeout = 20
;
; Secure Information Parameters
;
; Set values for ...
;               UserID                  ... Acquire
privileges for User ID
;               GroupID                 ... Acquire
privileges for Group ID
;
[SecureInfo]
UserID                     =  oracle
GroupID                    =  dba
;
;
; Directory Mappings
;
; List each mapping from a physical directory to a
; virtual directory, including the indicator whether
; the directory contains scripts or whether the
; subdirectories should be mapped recursively (R)
or
; non-recursively (N).
;
[DirMaps]
; This first line which has ?/ows21/doc needs to be
here to keep
; the virtual path ordering correct.
/oracle/ows21/doc/    NR     /
/oracle/ows21/admdoc/ NR     /ows-adoc/
/oracle/ows21/admbin/ CR     /ows-abin/
/oracle/ows21/bin/    CR     /ows-bin/
/oracle/ows21/bin/    CN     /owa_dba/
/oracle/ows21/doc/    NR     /ows-doc/
/oracle/ows21/img/    NR     /ows-img/
/oracle/ows21/demo/img/    NR  /tr-img/
/oracle/ows21/sample/ NR     /sample/
/oracle/ows21/sample/bin CR  /sample/bin/
/oracle/ows21/sample/wrbsdk/tpcc/ NR
/sample/wrbsdk/tpcc/
;
;
; File Cache
;
; List the full virtual filename path or wild-card
; expression denoting one or more files in the
Server's
; virtual file system.
;
;[FileCache]
;/products/*
;/products/xr25/Version*.c
;/employees/list/phones
;
;
; Language Extensions
;
; List for each combination of language type
; and character set one or more unique
; extensions.
;
[LangExt]
en                        iso-8859-1
              eng
en                        unicode-1-1
              engU uc
fr-CA                     iso-8859-1
              frc
fr-FR                     iso-8859-1
              fr
jp-JP                     iso-2022-jp
              jp
jp-JP                     unicode-1-1-utf-8
              jpU
;
;
; MIME Types
;
; List for each combination of MIME Type
; and representation one or more unique
; extensions.
;
[MIMETypes]
text/html      htm html
image/jpeg     jpg jpeg
image/gif      gif
appl/text      doc
text/plain     txt ksh lst
application/postscript ps
application/pdf pdf

;
[Encodings]
compress       Z
gzip           gz
;
[Security]
Basic {
(Users)
admin: oracle
(Groups)
dba: admin
(Realms)
Admin Server: dba
}
;
;
[Protection]
/ows-abin/        Basic(Admin Server)
```

```
/       Basic(Admin Server)
/ows-adoc/      Basic(Admin Server)
/ows-bin/       Basic(Admin Server)
/owa_dba/       Basic(Admin Server)
/ows-doc/       Basic(Admin Server)
/ows-img/       Basic(Admin Server)
/tr-img/        Basic(Admin Server)
/sample/        Basic(Admin Server)
/sample/bin/    Basic(Admin Server)

;
[DynApps]
/oracle/ows21/lib/ndwfss.so oracle_adp_init

###############
#### ubbweb ####
###############
*RESOURCES
IPCKEY                  80952
MASTER                  SITE1
UID                     102
GID                     101
PERM                    0660
MAXACCESSERS    1300
MAXSERVERS              50
MAXSERVICES             100
MODEL                   SHM
LDBAL                   Y
SCANUNIT        30
SANITYSCAN      5
BLOCKTIME               10

*MACHINES
cl01    LMID=SITE1
                TUXCONFIG="/oracle/client/tuxconfig
"
                ROOTDIR="/opt/uxptuxt"
#
                APPDIR="/oracle/bench/tpc/tpcc/TUX
_source"
                APPDIR="/oracle/bench/tpc/bin"
                ULOGPFX="/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
RQADDR=tpcc01 CLOPT ="-s TPCC01 1
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=2
RQADDR=tpcc02 CLOPT ="-s TPCC01 2
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=3
RQADDR=tpcc03 CLOPT ="-s TPCC01 3
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=4
RQADDR=tpcc04 CLOPT ="-s TPCC01 4
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=5
RQADDR=tpcc05 CLOPT ="-s TPCC01 5
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=6
RQADDR=tpcc06 CLOPT ="-s TPCC01 6
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=7
RQADDR=tpcc07 CLOPT ="-s TPCC01 7
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=8
RQADDR=tpcc08 CLOPT ="-s TPCC01 8
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=9
RQADDR=tpcc09 CLOPT ="-s TPCC01 9
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=10
RQADDR=tpcc10 CLOPT ="-s TPCC01 10
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=11
RQADDR=tpcc11 CLOPT ="-s TPCC01 11
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=12
RQADDR=tpcc12 CLOPT ="-s TPCC01 12
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=13
RQADDR=tpcc13 CLOPT ="-s TPCC01 13
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=14
RQADDR=tpcc14 CLOPT ="-s TPCC01 14
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=15
RQADDR=tpcc15 CLOPT ="-s TPCC01 15
tpcc/tpcc"
"tpccsvr.ott" SRVGRP=GROUP1 SRVID=16
RQADDR=tpcc16 CLOPT ="-s TPCC01 16
tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=17 RQADDR=tpcc17 CLOPT ="-s
TPCC01 17 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=18 RQADDR=tpcc18 CLOPT ="-s
TPCC01 18 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=19 RQADDR=tpcc19 CLOPT ="-s
TPCC01 19 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=20 RQADDR=tpcc20 CLOPT ="-s
TPCC01 20 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=21 RQADDR=tpcc21 CLOPT ="-s
TPCC01 21 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=22 RQADDR=tpcc22 CLOPT ="-s
TPCC01 22 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=23 RQADDR=tpcc23 CLOPT ="-s
TPCC01 23 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=24 RQADDR=tpcc24 CLOPT ="-s
TPCC01 24 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=25 RQADDR=tpcc25 CLOPT ="-s
TPCC01 25 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=26 RQADDR=tpcc26 CLOPT ="-s
TPCC01 26 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=27 RQADDR=tpcc27 CLOPT ="-s
TPCC01 27 tpcc/tpcc"
#"tpccsvr.ott"              SRVGRP=GROUP1
SRVID=28 RQADDR=tpcc28 CLOPT ="-s
TPCC01 28 tpcc/tpcc"

*SERVICES
TPCC01  TRANTIME=900

*ROUTING

# *NETWORK

###############
#### P_RUN.ora ####
###############
#
# $Header: p_run.ora 7030100.1 95/07/14
18:49:15 plai Generic<base> $ Copyr (c) 1993
Oracle
#
#=================================
============================+
#       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
dbwr_io_slaves                      = 6

cpu_count           = 18            # 18CPU
db_block_lru_latches     = 27                       #
18CPU
spin_count                  = 2000
parallel_max_servers = 90
db_name             = tpcc
db_files                    = 2000
db_file_multiblock_read_count = 64
db_block_buffers       = 5900000      # 11523.4MB
18CPU 16GB
_db_block_write_batch   = 3584
db_block_checkpoint_batch = 2048
hash_join_enabled         = FALSE
replication_dependency_tracking=FALSE
disk_asynch_io                      = TRUE
dml_locks           = 0
log_archive_start       = FALSE
log_archive_buffer_size = 32
log_checkpoint_interval = 10000000000
log_checkpoints_to_alert = TRUE
log_buffer          = 20971520
log_simultaneous_copies = 27
                #18CPU
log_small_entry_max_size= 0
gc_releasable_locks       = 0
max_rollback_segments = 700
open_cursors            = 512
processes           = 1024
sessions            = 1536
transactions            = 1024
distributed_transactions = 0
transactions_per_rollback_segment = 1
```

```
rollback_segments      =
(t1,t2,t3,t4,t5,t6,t7,t8,t9,t11,t12,t13,t14,t15,t16,t18,
t19,t20,t21,t22,t23,t24,t25,t26,t27,t28,t29,t30,t31,t
32,t33,t34,t35,t36,t37,t38,t39,t40,t41,t42,t43,t44,t4
5,t46,t48,t49,t50,t51,t52,t53,t54,t55,t56,t57,t58,t59
,t60,t61,t62,t63,t64,t65,t66,t67,t68,t69,t70,t71,t72,t
73,t74,t75,t76,t77,t78,t79,t80,t81,t82,t83,t84,t85,t8
6,t87,t88,t89,t90,t91,t92,t93,t94,t95,t96,t97,t98,t99
,t100,t101,t102,t103,t104,t105,t106,t107,t108,t109
,t110,t111,t112,t113,t114,t115,t116,t117,t118,t119
,t120,t121,t122,t123,t124,t125,t126,t127,t128,t129
,t130,t131,t132,t133,t134,t135,t136,t137,t138,t139
,t140,t141,t142,t143,t144,t145,t146,t147,t148,t149
,t150,t151,t152,t153,t154,t155,t156,t157,t158,t159
,t160,t161,t162,t163,t164,t165,t166,t167,t168,t169
,t170,t171,t172,t173,t174,t175,t176,t177,t178,t179
,t180,t181,t182,t183,t184,t185,t186,t187,t188,t189
,t190,t191,t192,t193,t194,t195,t196,t197,t198,t199
,t200,t201,t202,t203,t204,t205,t206,t207,t208,t209
)
shared_pool_size          = 30000000
discrete_transactions_enabled = FALSE
cursor_space_for_time   = TRUE
_controlfile_enqueue_timeout = 180000
enqueue_resources = 35000
recovery_parallelism = 90
buffer_pool_recycle =
(buffers:30000,lru_latches:4)
###############
#### rtcl_dm.sh ####
###############
#!/bin/sh

# rtcl_dm.sh -- move all oracle deamons to RT
class

# Version 2.0               Nov.13,1997
                get logs
# Version 2.1               Nov.15,1997
                change DATDIR (ps -> PS)
# Version 1.0dm             Nov.25,1997
                move only deamon to RT class
# Version 1.1dm Dec.09,1997          change
massage

PIDS=`ps -ef | grep ora_ | grep -v grep | nawk
'{print $2}'`

# echo $PIDS

echo "RTCL_DM: START == " `date`

for pid in $PIDS
do
                priocntl -s -c RT $pid
done

echo "RTCL_DM: END   == " `date`
###############
#### rtcl_lsnr.sh ####
###############
#!/bin/sh

# rtcl_lsnr.sh -- move all listener process to RT
class

# Version 2.0               Nov.13,1997
                get logs
# Version 2.1               Nov.15,1997
                change DATDIR (ps -> PS)
# Version 1.0b              Nov.25,1997
                move listener process to RT class
# Version 1.1b  Dec.09,1997          change
massage

echo "RTCL_LSNR: START == " `date`
PID=`ps -ef | grep lsnr | grep -v grep | nawk '{print
$2}'`
priocntl -s -c RT -i pid $PID
echo "RTCL_LSNR: END   == " `date`

###############
#### bind.sh ####
###############
#!/bin/sh

# bind.sh -- bind oracle processes
# No.30646-base, 6slave

# Version 2.0  Nov.13,1997          get log
# Version 2.1  Nov.14,1997          exec each
slave deamons
# Version 2.2  Dec.08,1997          for auto.sh
(add $2: outdir)
# Version 2.3  Dec.09,1997          add check
arguments
# Versoin 2.3b Dec.25,1997          for 3tier
# Version 2.4b Dec.27,1997          use run_id

PROG=`basename $0`
VERSION="2.4b"
SVR_SH_DIR=/home/oracle/tools
RRT=${SVR_SH_DIR}/.rrt/rroot

##
## function usage()
##
usage(){
                echo "$PROG Version $VERSION" >
/dev/stderr
                echo "usage: $PROG <run_id>
[outdir]" > /dev/stderr
                echo "       outdir: directory for data
[default: /output]" > /dev/stderr
}

#DATE=`date '+%y%m%d_%H%M'`

##
## check arguments
##
if [ $# -eq 1 ]
then
                if [ $1 = "-h" ]
                then
                                usage;
                                exit
                fi
                RUN_ID=$1
                DATDIR=/output/PS/${RUN_ID}
elif [ $# -eq 2 ]
then
                RUN_ID=$1
                DATDIR=$2/PS/${RUN_ID}
else
                usage;
                exit 1
fi

if [ ! -d $DATDIR ]
then
                mkdir -p $DATDIR
fi

INFOFILE=${DATDIR}/bind.info
CHKFILE=${DATDIR}/pbind.${RUN_ID}.rpt
PSFILE=${DATDIR}/ps.${RUN_ID}
PSFILE_RT=${DATDIR}/ps_rt.${RUN_ID}

DF=/dev/null

##
## MACROs
##

# cpu-id to pbind for oracle-deamon
PMON=9
SMON=9
CKPT=9
SLV1=10
SLV2=11
SLV3=12
SLV4=13
SLV5=14
SLV6=15
DBWR=16
LGWR=17

# number of oracle-shadows to each cpus
CPU_00_NUM=27
CPU_01_NUM=28
CPU_02_NUM=28
CPU_03_NUM=28
CPU_04_NUM=28
CPU_05_NUM=28
CPU_06_NUM=28
CPU_07_NUM=28
CPU_08_NUM=28
CPU_09_NUM=28
CPU_10_NUM=23
CPU_11_NUM=23
CPU_12_NUM=23
CPU_13_NUM=23
CPU_14_NUM=23
CPU_15_NUM=23
CPU_16_NUM=21
CPU_17_NUM=26

##
## function cut_shadow()
##
line=0
cut_shadow() {
                line=`expr $line + $1`
                tmp_pids=`head -${line} /tmp/p | tail -
$1 | awk '{print $2}'`
}

##
## MAIN
##
ps -ef | grep ora_ | grep -v grep > /tmp/pd
```

```
echo "BIND: oracle deamons == " `date` >>
$INFOFILE

#
# pbind pmon
#
PIDS=`grep ora_pmon /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $PMON $pid > $DF
done

#
# pbind smon
#
PIDS=`grep ora_smon /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $SMON $pid > $DF
done

#
# pbind ckpt
#
PIDS=`grep ora_ckpt /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $CKPT $pid > $DF
done

#
# pbind dbwr
#
PIDS=`grep ora_dbwr /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $DBWR $pid > $DF
done

#
# pbind slave(ora_i1??_tpcc)
#
PIDS=`grep ora_i101 /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $SLV1 $pid > $DF
done

PIDS=`grep ora_i102 /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $SLV2 $pid > $DF
done

PIDS=`grep ora_i103 /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $SLV3 $pid > $DF
done

PIDS=`grep ora_i104 /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $SLV4 $pid > $DF
done

PIDS=`grep ora_i105 /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $SLV5 $pid > $DF
done

PIDS=`grep ora_i106 /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $SLV6 $pid > $DF
done

#
# pbind lgwr
#
PIDS=`grep ora_lgwr /tmp/pd | grep -v grep | awk
'{print $2}'`
for pid in $PIDS
do
        pbind -b $LGWR $pid > $DF
done

#
# pbind all oracle shadows
#
echo "BIND: oracle shadows == " `date` >>
$INFOFILE

ps -ef | grep LOCAL | grep -v grep | grep oracle >
/tmp/p
cut_shadow $CPU_00_NUM
CPU_00=$tmp_pids
cut_shadow $CPU_01_NUM
CPU_01=$tmp_pids
cut_shadow $CPU_02_NUM
CPU_02=$tmp_pids
cut_shadow $CPU_03_NUM
CPU_03=$tmp_pids
cut_shadow $CPU_04_NUM
CPU_04=$tmp_pids
cut_shadow $CPU_05_NUM
CPU_05=$tmp_pids
cut_shadow $CPU_06_NUM
CPU_06=$tmp_pids
cut_shadow $CPU_07_NUM
CPU_07=$tmp_pids
cut_shadow $CPU_08_NUM
CPU_08=$tmp_pids
cut_shadow $CPU_09_NUM
CPU_09=$tmp_pids
cut_shadow $CPU_10_NUM
CPU_10=$tmp_pids
cut_shadow $CPU_11_NUM
CPU_11=$tmp_pids
cut_shadow $CPU_12_NUM
CPU_12=$tmp_pids
cut_shadow $CPU_13_NUM
CPU_13=$tmp_pids
cut_shadow $CPU_14_NUM
CPU_14=$tmp_pids
cut_shadow $CPU_15_NUM
CPU_15=$tmp_pids
cut_shadow $CPU_16_NUM
CPU_16=$tmp_pids
cut_shadow $CPU_17_NUM
CPU_17=$tmp_pids

for pid in $CPU_00
do
        pbind -b 0 $pid > $DF
done

for pid in $CPU_01
do
        pbind -b 1 $pid > $DF
done

for pid in $CPU_02
do
        pbind -b 2 $pid > $DF
done

for pid in $CPU_03
do
        pbind -b 3 $pid > $DF
done

for pid in $CPU_04
do
        pbind -b 4 $pid > $DF
done

for pid in $CPU_05
do
        pbind -b 5 $pid > $DF
done

for pid in $CPU_06
do
        pbind -b 6 $pid > $DF
done

for pid in $CPU_07
do
        pbind -b 7 $pid > $DF
done

for pid in $CPU_08
do
        pbind -b 8 $pid > $DF
done

for pid in $CPU_09
do
        pbind -b 9 $pid > $DF
done

for pid in $CPU_10
do
        pbind -b 10 $pid > $DF
done

for pid in $CPU_11
do
        pbind -b 11 $pid > $DF
done
```

```
for pid in $CPU_12
do
        pbind -b 12 $pid > $DF
done

for pid in $CPU_13
do
        pbind -b 13 $pid > $DF
done

for pid in $CPU_14
do
        pbind -b 14 $pid > $DF
done

for pid in $CPU_15
do
        pbind -b 15 $pid > $DF
done

for pid in $CPU_16
do
        pbind -b 16 $pid > $DF
done

for pid in $CPU_17
do
        pbind -b 17 $pid > $DF
done

echo "BIND: END    == " `date` >> $INFOFILE

echo "# get psx -e -Xc & pbind -q == " `date` >>
$INFOFILE
$RRT priocntl -e -c RT ${SVR_SH_DIR}/psx -e -
Xc > $PSFILE
$RRT priocntl -e -c RT pbind -q > $CHKFILE
#sleep 900
echo "# get ps -ef == " `date` >> $INFOFILE
$RRT priocntl -e -c RT ps -ef > ${PSFILE_RT}
```

Appendix E: Database Creation Code

```
##########################################
##
####
####   addfile.sh
####
####


#
# $Header: addfile.sh 7010000.2 94/10/12
13:34:31 plai Generic<base> $ Copyr (c) 1994
Oracle
#

#=================================
============================+
#       Copyright (c) 1994  Oracle Corp, Redwood
Shores, CA       |
#                OPEN SYSTEMS PERFORMANCE
GROUP                |
#                    All Rights Reserved
|
#=================================
============================+
# FILENAME
#     addfile.sh
# DESCRIPTION
#     Add datafile to a tablespace.
# USAGE
#     addfile.sh <tablespace> <data file> <size>
#=================================
============================*/

svrmgrl <<!
  connect internal
  alter tablespace $1 add datafile '$2' size $3
reuse;
  exit;
!

##########################################
##
####
####   alter.sh
####
####

#!/bin/ksh
undml.sh
sqlplus tpcc/tpcc <<!
  alter table history     storage (next 150M);
  alter cluster ccluster  storage (next  50M);
  alter cluster scluster  storage (next  50M);
  alter table orders      storage (next 60M);
  alter table order_line  storage (next 750M);
  alter table new_order   storage (next 50M);
  alter index iorders     storage (next 980M);
  alter index iorders2    storage (next 150M);
  alter index inew_order  storage (next 500M);
  alter index iorder_line storage (next 450M);
  alter index istock      storage (next 50M);
  alter index icustomer   storage (next 50M);
  alter index icustomer2  storage (next 50M);
 exit;
!
dml.sh

##########################################
##
####
####   benchdb.sh
####
####


#
# $Header: benchdb.sh 7030100.1 96/05/02
19:05:22 plai Generic<base> $ Copyr (c) 1995
Oracle
#

#
#=================================
============================+
#       Copyright (c) 1996  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=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES
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
  DATAFILE=`cat $PARTITION_DATA/SYSTEM`
  LOGCOUNT=`cat $PARTITION_DATA/LOG |
wc -l`

  if [ $LOGCOUNT -ne 2 ]
  then
    echo "Invalid number of Redo-LOG :
$LOGCOUNT"
    exit 1
  fi

  LOG1=`head -n 1 $PARTITION_DATA/LOG`
  LOG2=`tail -n -1 $PARTITION_DATA/LOG`

  svrmgrl <<!
            set echo on
            connect internal
            startup
pfile=$TPCC_ADMIN/p_create.ora  nomount
            create database tpcc controlfile reuse
maxdatafiles 1500
              datafile '/dev/rdsk/$DATAFILE' size
1501M reuse
              logfile  '/dev/rdsk/$LOG1'     size
1900M reuse,
                       '/dev/rdsk/$LOG2'
size 1900M reuse;
            exit
  !

#
# Create more rollback segments
#

svrmgrl <<!
  connect system/manager
  create rollback segment s1  storage (initial 600k
minextents 2 next 600k);
  create rollback segment s2  storage (initial 600k
minextents 2 next 600k);
  create rollback segment s3  storage (initial 600k
minextents 2 next 600k);
  create rollback segment s4  storage (initial 600k
minextents 2 next 600k);
  create rollback segment s5  storage (initial 600k
minextents 2 next 600k);
  create rollback segment s6  storage (initial 600k
minextents 2 next 600k);
  create rollback segment s7  storage (initial 600k
minextents 2 next 600k);
  create rollback segment s8  storage (initial 600k
minextents 2 next 600k);
  create rollback segment s9  storage (initial 600K
minextents 2 next 600k);
  create rollback segment s10 storage (initial 600k
minextents 2 next 600k);
  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);

!
fi



#  disconnect;
#  connect internal;
#  shutdown;
#  exit;

#
#  Startup database with params file that includes
new rollback segments
#

./online_roll_S.sh

crtroll.sh &                # x  5
crtware.sh &                              # x  1
crthist.sh &                # x  12
crtitem.sh &                # x  1
crtnord.sh &                # x  12
wait
crtord.sh &                 # x  25
crticust1.sh &                            # x  10
crticust2.sh &                            # x  30
crtistk.sh &                # x  6
crtiord1.sh &                             # x  20
wait                                      #      197
crtiord2.sh &                             # x  35
crtinord.sh &                             # x  10
crttemp.sh &                              # x  36
wait                                      #      119
crtcust.sh &                # x 100
crtiordl.sh &               # x  20
wait                                      #      140 +
140
crtstk.sh &                 # x 100
crtordl.sh &                # x  20
wait                                      #      200 +
200 + 100

# These disks are being used for iordl and iord2

#
# run catalog if NO_CAT unset
#

if [ "$NO_CAT" = "" ]
then
svrmgrl <<!
      set echo off;
      connect sys/change_on_install;
      @?/rdbms/admin/catalog;
      @?/rdbms/admin/catproc;
      @?/rdbms/admin/catparr;
      exit;
!
fi

##############################################
##
####
####  benchsetup.sh
####
####


#
# $Header: benchsetup.sh 7030100.2 96/05/16
17:59:17 plai Generic<base> $ Copyr (c) 1995
Oracle
#

#
#====================================
============================+
#      Copyright (c) 1996  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=$BENCH_HOME/tpcc/sql
TPCC_BLOCKS=$BENCH_HOME/tpcc/blocks
TPCC_SCRIPTS=$BENCH_HOME/tpcc/scripts
TPCC_UTILS=$TPCC_SCRIPTS/utils
AUDIT_SQL=$BENCH_HOME/tpcc/audit/sql
BUILD_SQL=sql
ADMIN=admin

OUTDIR=outdir
MULT=3000

PATH=${PATH}:${TPCC_SOURCE}:${TPCC_UTI
LS}
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)? [3000]" $C
  read MULT
  if [ "$MULT" = "" ]
  then
     MULT=3000
  fi
fi

if [ ! -d $OUTDIR ]
then
  mkdir $OUTDIR
fi

#
# Create data for load (pload data)
#
pre_pload.sh -mu $MULT &

####################################
```

```
# Create database.
####################################

if [ "$NO_DB" = "" ]
then

echo "[CALL] benchdb.sh"
echo ------ `date` -------
echo

benchdb.sh

echo
echo "[RETURENED] benchdb.sh"
echo ------ `date` --------
echo

fi

$TPCC_SCRIPTS/switchlog.sh

#
#  Create tables.
#

if [ "$NO_TAB" = "" ]
then
  sqlplus system/manager
@$BUILD_SQL/tpcc_tab
  sqlplus system/manager
@$BUILD_SQL/tpcc_rol
fi

# This step is required because some tables are
not recognized by tpcc
# userid (EX: district, warehouse and item)

sqldba <<!
  connect internal;
  shutdown;
  exit;
!

sqldba <<!
      set echo on
      connect internal;
      startup pfile=$ADMIN/p_build2.ora
      exit;
!

sleep 60

online_roll.sh

##########################################
################
#  Load history, new-order, order, order-line tables
##########################################
################

wait
pload_hist.sh    > ${OUTDIR}/pload_hist.out
2>&1 &
pload_neword.sh  >
${OUTDIR}/pload_neword.out  2>&1 &
pload_ord_ordl.sh >
${OUTDIR}/pload_ord_ordl.out 2>&1 &
wait

$TPCC_SCRIPTS/switchlog.sh

#
#  Create customer and stock tables.
#

if [ "$NO_TAB" = "" ]
then
  sqlplus tpcc/tpcc @$BUILD_SQL/tpcc_tab2 >
${OUTDIR}/tab2.out 2>&1 &
  sqlplus tpcc/tpcc @$BUILD_SQL/tpcc_tab3 >
${OUTDIR}/tab3.out 2>&1 &
fi

wait

$TPCC_SCRIPTS/switchlog.sh

#
#  Load warehouse, district, item tables
#

tpccload -M $MULT -w
tpccload -M $MULT -d
tpccload -M $MULT -i

# Loading customer table with 3000W

I=1
SW=1
EW=60
INC=60

echo "[BEGIN] Load customer tables"
echo ------- `date` -------
echo

while [ $I -le 50 ]
do
  tpccload -M $MULT -c -b $SW -e $EW >
${OUTDIR}/cust${I}.out 2>&1 &
  I=`expr $I + 1`
  SW=`expr $SW + $INC`
  EW=`expr $EW + $INC`
done

#
#  Load stock table (in parallel with loading
customer table)
#

I=1
SI=1
EI=2000
INC=2000
while [ $I -le 50 ]
do
  tpccload -M $MULT -S -j $SI -k $EI >
${OUTDIR}/stk${I}.out 2>&1 &
  I=`expr $I + 1`
  SI=`expr $SI + $INC`
  EI=`expr $EI + $INC`
done

wait

$TPCC_SCRIPTS/switchlog.sh

#
#  Create indexes
#

if [ "$NO_IND" = "" ]
then

sqlplus system/manager <<!
  alter user tpcc temporary tablespace temp;
  quit;
!

svrmgrl <<!
  connect internal
  alter tablespace temp
    default storage (initial 100M next 100M
                pctincrease 0 maxextents
unlimited);
  exit;
!

sqlplus tpcc/tpcc @$BUILD_SQL/tpcc_ix1
sqlplus tpcc/tpcc @$BUILD_SQL/tpcc_ix3
sqlplus tpcc/tpcc @$BUILD_SQL/tpcc_ix4
sqlplus tpcc/tpcc @$BUILD_SQL/tpcc_ix2

 svrmgrl <<!
  connect internal
  alter tablespace temp
    default storage (initial 20K next 20K
                pctincrease 50 maxextents
unlimited);
  exit;
!

fi

#
#  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 @$TPCC_BLOCKS/views
sqlplus tpcc/tpcc @$TPCC_BLOCKS/pay

#
#  Create cache views
#

$TPCC_SCRIPTS/create_cache_views.sh

#
```

```
# Get some statistics
#

$TPCC_SCRIPTS/utils/ext_all.sh >
${OUTDIR}/ext_all.out 2>&1

$TPCC_SCRIPTS/utils/space_init.sh
$TPCC_SCRIPTS/utils/space_get.sh 35000 3000
$TPCC_SCRIPTS/utils/space_rpt.sh
${OUTDIR}/space.rpt

sqlplus system/manager <<!
  alter user tpcc temporary tablespace system;
  quit;
!

sqlplus sys/change_on_install <<!
  grant execute on dbms_lock to public;
  grant execute on dbms_pipe to public;
  grant select on v_\$parameter to public;
  quit;
!

sqlplus tpcc/tpcc @$AUDIT_SQL/plsql_mon
sqlplus tpcc/tpcc @$AUDIT_SQL/cre_tab

svrmgrl <<!
  set echo off;
  connect sys/change_on_install;
  @?/rdbms/admin/catparr;
  exit;
!

echo
echo ======= `date` =======
echo "[End time]"

##########################################
##
####
####  crtcust.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

DATAFILE=`cat
$PARTITION_DATA/CUSTOMER`

I=1
SIZE=391M
for FILE in $DATAFILE
do

if [ $I -eq 1 ]
then
svrmgrl << !
  connect system/manager
  create tablespace cust datafile
    '/dev/rdsk/$FILE' size $SIZE reuse;
  exit;
!
else
  MOD=`expr $I % 100`
  if [ $MOD -eq 0 ]
  then
    wait
  fi
  sleep 2
  addfile.sh cust /dev/rdsk/${FILE} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "CUST END"


exit

##########################################
##
####
####  crthist.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

DATAFILE=`cat $PARTITION_DATA/HISTORY`

I=1
SIZE=910M
for FILE in $DATAFILE
do
if [ $I -eq 1 ]
then
svrmgrl << !
  connect system/manager
  create tablespace hist datafile
    '/dev/rdsk/$FILE' size $SIZE reuse;
  exit;
!
else
  addfile.sh hist /dev/rdsk/${FILE} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "Hist END"

exit

##########################################
##
####
####  crticust1.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

DATAFILE=`cat
$PARTITION_DATA/ICUSTOMER1`

I=1
SIZE=601M
for FILE in $DATAFILE
do
if [ $I -eq 1 ]
then
svrmgrl << !
  connect system/manager
  create tablespace icust1 datafile
    '/dev/rdsk/$FILE' size $SIZE reuse;
  exit;
!
else
  addfile.sh icust1 /dev/rdsk/${FILE} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "ICUST1 END"
exit

##########################################
##
####
####  crticust2.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

DATAFILE=`cat
$PARTITION_DATA/ICUSTOMER2`

I=1
SIZE=251M
for FILE in $DATAFILE
do
if [ $I -eq 1 ]
then
svrmgrl << !
  connect system/manager
  create tablespace icust2 datafile
    '/dev/rdsk/$FILE' size $SIZE reuse;
  exit;
!
else
  addfile.sh icust2 /dev/rdsk/${FILE} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "ICUST2 END"
exit

##########################################
##
####
####  crtinord.sh
####
####
```

```
#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

FILENUM=`cat $PARTITION_DATA/INEW-
ORDER | wc -l`

I=0
SIZE=307M
while [ $I -lt $FILENUM ]
do
if [ $I -eq 0 ]
then
svrmgrl << !
  connect system/manager
  create tablespace inord datafile
    '/dev/rvol/INORD_$I' size $SIZE reuse;
  exit;
!
else
  addfile.sh inord /dev/rvol/INORD_${I} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "INord END"
exit

########################################
##
####
####  crtiord1.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

FILENUM=`cat $PARTITION_DATA/IORDER1 |
wc -l`

I=0
SIZE=343M
while [ $I -lt $FILENUM ]
do
if [ $I -eq 0 ]
then
svrmgrl << !
  connect system/manager
  create tablespace iord1 datafile
    '/dev/rvol/IORD1_$I' size $SIZE reuse;
  exit;
!
else
  sleep 2
  addfile.sh iord1 /dev/rvol/IORD1_${I} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "IORD1 END"
exit

########################################
##
####
####  crtiord2.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

FILENUM=`cat $PARTITION_DATA/IORDER2 |
wc -l`

I=0
SIZE=151M
while [ $I -lt $FILENUM ]
do
if [ $I -eq 0 ]
then
svrmgrl << !
  connect system/manager
  create tablespace iord2 datafile
    '/dev/rvol/IORD2_$I' size $SIZE reuse;
  exit;
!
else
  addfile.sh iord2 /dev/rvol/IORD2_${I} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "IORD2 END"
exit

########################################
##
####
####  crtiordl.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

FILENUM=`cat $PARTITION_DATA/IORD_LINE |
wc -l`

I=0
SIZE=1885M
while [ $I -lt $FILENUM ]
do
if [ $I -eq 0 ]
then
svrmgrl << !
  connect system/manager
  create tablespace iordl datafile
    '/dev/rvol/IORDL_$I' size $SIZE reuse;
  exit;
!
else
  MOD=`expr $I % 10`
  if [ $MOD -eq 0 ]
  then
    wait
  fi
  sleep 2
  addfile.sh iordl /dev/rvol/IORDL_${I} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "IORDL END"
exit

########################################
##
####
####  crtistk.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

DATAFILE=`cat $PARTITION_DATA/ISTOCK`

I=1
SIZE=1697M
for FILE in $DATAFILE
do
if [ $I -eq 1 ]
then
svrmgrl << !
  connect system/manager
  create tablespace istk datafile
    '/dev/rdsk/$FILE' size $SIZE reuse;
  exit;
!
else
  addfile.sh istk /dev/rdsk/${FILE} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "ISTK END"
exit

########################################
##
####
####  crtitem.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES
```

```
DATAFILE=`cat $PARTITION_DATA/ITEM`

I=1
SIZE=19M
for FILE in $DATAFILE
do
if [ $I -eq 1 ]
then
svrmgrl << !
  connect system/manager
  create tablespace items datafile
    '/dev/rdsk/$FILE' size $SIZE reuse;
  exit;
!
else
  addfile.sh items /dev/rdsk/${FILE} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "ITEMS END"
exit

########################################
##
####
####  crtnord.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

FILENUM=`cat $PARTITION_DATA/NEW-
ORDER | wc -l`

I=0
SIZE=257M
while [ $I -lt $FILENUM ]
do
if [ $I -eq 0 ]
then
svrmgrl << !
  connect system/manager
  create tablespace nord datafile
    '/dev/rvol/NORD_$I' size $SIZE reuse;
  exit;
!
else
  addfile.sh nord /dev/rvol/NORD_${I} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "NORD END"
exit

########################################
##
####
####  crtord.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

FILENUM=`cat $PARTITION_DATA/ORDERS |
wc -l`

I=0
SIZE=361M
while [ $I -lt $FILENUM ]
do
if [ $I -eq 0 ]
then
svrmgrl << !
  connect system/manager
  create tablespace ord datafile
    '/dev/rvol/ORD_$I' size $SIZE reuse;
  exit;
!
else
  addfile.sh ord /dev/rvol/ORD_${I} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "ORD END"
exit

########################################
##
####
####  crtordl.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

FILENUM=`cat
$PARTITION_DATA/ORDER_LINE | wc -l`

I=0
SIZE=1864M
while [ $I -lt $FILENUM ]
do
if [ $I -eq 0 ]
then
svrmgrl << !
  connect system/manager
  create tablespace ordl datafile
    '/dev/rvol/ORDL_$I' size $SIZE reuse;
  exit;
!
else
  MOD=`expr $I % 20`
  if [ $MOD -eq 0 ]
  then
    wait
  fi
  sleep 2
  addfile.sh ordl /dev/rvol/ORDL_${I} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "ORDL END"
exit

########################################
##
####
####  crtroll.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

DATAFILE=`cat $PARTITION_DATA/ROLL`

I=1
SIZE=301M
for DISK in $DATAFILE
do
svrmgrl << DEND
  connect system/manager
  create tablespace roll$I datafile
    '/dev/rdsk/$DISK' size $SIZE reuse;
  exit;
DEND
I=`expr $I + 1`
done
wait
echo "ROLL END"
exit


########################################
##
####
####  crtstk.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

DATAFILE=`cat $PARTITION_DATA/STOCK`

rm /tmp/.stk_done

I=1
SIZE=259M
for FILE in $DATAFILE
do
if [ $I -eq 1 ]
then
svrmgrl << !
  connect system/manager
  create tablespace stocks datafile
    '/dev/rdsk/$FILE' size $SIZE reuse;
```

```
  exit;
!
else
  MOD=`expr $I % 100`
  if [ $MOD -eq 0 ]
  then
    wait
  fi
  sleep 2
  addfile.sh stocks /dev/rdsk/${FILE} $SIZE &
fi
I=`expr $I + 1`
done
wait

touch /tmp/.stk_done

echo "STOCKS END"

exit

##############################################
##
####
####   crttemp.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

DATAFILE=`cat $PARTITION_DATA/TEMP`

I=1
SIZE=1801M
for FILE in $DATAFILE
do
if [ $I -eq 1 ]
then
svrmgrl << !
  connect system/manager
  create tablespace temp datafile
    '/dev/rdsk/$FILE' size $SIZE reuse;
  exit;
!
else
  addfile.sh temp /dev/rdsk/${FILE} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "TEMP END"

exit

##############################################
##
####
####   crtware.sh
####
####

#! /bin/sh

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_ADMIN=admin
PARTITION_DATA=$BENCH_HOME/tpcc/scripts/
build/DATAFILES

DATAFILE=`cat
$PARTITION_DATA/WAREHOUSE`

I=1
SIZE=321M
for FILE in $DATAFILE
do
if [ $I -eq 1 ]
then
svrmgrl << !
  connect system/manager
  create tablespace ware datafile
    '/dev/rdsk/$FILE' size $SIZE reuse;
  exit;
!
else
  addfile.sh ware /dev/rdsk/${FILE} $SIZE &
fi
I=`expr $I + 1`
done
wait
echo "WARE END"
exit

##############################################
##
####
####   log1.sh
####
####

#
#! /bin/sh
#====================================
============================+
#          Copyright (c) 1993  Oracle Corp,
Belmont, CA         |
#               OPEN SYSTEMS PERFORMANCE
GROUP               |
#                   All Rights Reserved
|
#====================================
============================+
# FILENAME
#    dbup.sh
# DESCRIPTION
#               Usage: dbup.sh <# of dml_lock>
[SHARED]
#====================================
============================*/
#
svrmgrl <<!
  connect internal ;
  alter database add logfile '/dev/rvol/LOGx1' size
17000M reuse;
  exit;
!

#rm -rf $TPCC_ADMIN/p_tmp.ora

##############################################
##
####
####   log2.sh
####
####

#
#! /bin/sh
#====================================
============================+
#          Copyright (c) 1993  Oracle Corp,
Belmont, CA         |
#               OPEN SYSTEMS PERFORMANCE
GROUP               |
#                   All Rights Reserved
|
#====================================
============================+
# FILENAME
#    dbup.sh
# DESCRIPTION
#               Usage: dbup.sh <# of dml_lock>
[SHARED]
#====================================
============================*/
#
svrmgrl <<!
  connect internal ;
  alter database add logfile '/dev/rvol/LOGx2' size
17000M reuse;
  exit;
!

##############################################
##
####
####   p_build2.ora
####
####

#
# $Header: p_build2.ora 7030100.2 96/05/02
19:16:29 plai Generic<base> $ Copyr (c) 1995
Oracle
#
#====================================
============================+
#       Copyright (c) 1996  Oracle Corp, Redwood
Shores, CA      |
#               OPEN SYSTEMS PERFORMANCE
GROUP               |
#                   All Rights Reserved
|
#====================================
============================+
# FILENAME
#     p_build2.ora
# DESCRIPTION
#     Oracle parameter file for building TPC-C
database.
#====================================
============================
#

sort_area_size                  = 31457280
parallel_server_idle_time       = 30
parallel_max_servers            = 60
db_name                         = tpcc
db_files                        = 1500
db_file_multiblock_read_count   = 32
```

```
db_block_buffers                = 400000
_db_block_write_batch           = 512
db_block_checkpoint_batch       = 512
dml_locks                       = 500
hash_join_enabled               = FALSE
log_archive_start               = FALSE
log_archive_buffer_size         = 32
log_checkpoint_interval         = 1000000000
log_checkpoints_to_alert        = TRUE
log_buffer                      = 1048576
max_rollback_segments           = 500
processes                       = 1000
sessions                        = 1200
transactions                    = 1200
distributed_transactions        = 0
transactions_per_rollback_segment = 3
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,t42,t4
3,t44,t45,t46,t47,t48,t49,t50,t51,t52,t53,t54,t55,t56
,t57,t58,t59,t60,t61,t62,t63,t64,t65,t66,t67,t68,t69,t
70,t71,t72,t73,t74,t75,t76,t77,t78,t79,t80,t81,t82,t8
3,t84,t85,t86,t87,t88,t89,t90,t91,t92,t93,t94,t95,t96
,t97,t98,t99,t100)
shared_pool_size                = 20000000
discrete_transactions_enabled   = FALSE
cursor_space_for_time           = TRUE
event = "8103 trace name errorstack forever, level
3 : 8103 trace name context forever, level 10"

###########################################
##
####
####  p_create.ora
####
####

#
# $Header: p_create.ora 7030100.2 96/05/02
19:16:37 plai Generic<base> $ Copyr (c) 1995
Oracle
#
#===================================
============================+
#      Copyright (c) 1996  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.
#===================================
============================
#

db_name              = tpcc
db_files                    = 2000
db_block_buffers      = 2000
dml_locks            = 1000
log_checkpoint_interval = 999999999
log_buffer           = 32768
sessions             = 1000
processes            = 1000
transactions         = 1500

###########################################
##
####
####  pload.sh
####
####


#
# $Header: pload.sh 7030100.1 96/05/02 19:06:06
plai Generic<base> $ Copyr (c) 1995 Oracle
#

#
#===================================
============================+
#      Copyright (c) 1996  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=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_LOADER=$BENCH_HOME/tpcc/loader

OUTDIR=outdir
LDIR1=/data1
MULT=5

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


if [ ! -d $OUTDIR ]
then
  mkdir $OUTDIR
fi

#########################################
###################
#  Load history new-order order_line TO file
#########################################
###################

echo "[BEGIN] tpccload hist, new-order, order,
order_line"
echo ------- `date` -------
echo

tpccload -M $MULT -h -g -b   1 -e   5 >
${LDIR1}/hist1.dat &
tpccload -M $MULT -n -g > ${LDIR1}/neword1.dat
&
tpccload -M $MULT -o ${LDIR1}/ordline1.dat -g -b
1 -e 5 > ${LDIR1}/order1.dat &

wait

echo "[END] tpccload hist, new-order, order,
order_line"
echo ------- `date` -------
echo

#########################################
###################
#  Load history new-order order order_line TO
database
#########################################
###################

echo "[BEGIN] sqlldr hist, new-order, order,
order_line"
echo ------- `date` -------
echo

sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist1.log \
```

```
      bad=hist1.bad data=${LDIR1}/hist1.dat
discard=hist1.dsc \
      file=/dev/rdsk/hd1001 &

sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword1.log \
      bad=neword1.bad
data=${LDIR1}/neword1.dat discard=neword1.dsc
\
      file=/dev/rdsk/hd1006 &

sqlldr tpcc/tpcc control=$TPCC_LOADER/order.ctl
log=order1.log \
      bad=order1.bad data=${LDIR1}/order1.dat
discard=order1.dsc \
      file=/dev/rdsk/hd1005 &

sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl
log=ordline1.log \
      bad=ordline1.bad data=${LDIR1}/ordline1.dat
discard=ordline1.dsc \
      file=/dev/rdsk/hd1103 &

wait

echo "[END] tpccload hist, new-order, order,
order_line"
echo ------- `date` -------
echo

rm -rf ${LDIR1}/hist1.dat ${LDIR1}/neword1.dat
${LDIR1}/order1.dat ${LDIR1}/ordl
ine1.dat


#########################################
##
####
####   pload_hist.sh
####
####

#!/bin/sh

#
# $Header: pload.sh 7030100.1 96/05/02 19:06:06
plai Generic<base> $ Copyr (c) 1995 Oracle
#

#
#=================================
============================+
#       Copyright (c) 1996  Oracle Corp, Redwood
Shores, CA       |
#                OPEN SYSTEMS PERFORMANCE
GROUP                |
#                    All Rights Reserved
|
#=================================
============================+
# FILENAME
#      pload_hist.sh
# DESCRIPTION
#      Usage: pload_hist.sh [options]
#           -mu <multiplier>     (# of warehouses)
#=================================
============================
#

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_LOADER=$BENCH_HOME/tpcc/loader

PROG=$0
OUTDIR=outdir
LDIR=/data
MULT=3000

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)? [3000]" $C
  read MULT
  if [ "$MULT" = "" ]
  then
     MULT=3000
  fi
fi


if [ ! -d $OUTDIR ]
then
  mkdir $OUTDIR
fi

PROG=`basename $PROG`
#########################################
####################
#  Load history TO file
#########################################
####################
if [ $PROG = pre_pload_hist.sh ]
then
  echo "[BEGIN] tpccload history"
  echo ------- `date` -------
  echo

  rm ${LDIR}/hist*.dat ${LDIR}/hist*.data

  tpccload -M $MULT -h -g -b    1 -e   50 >
${LDIR}/hist01.data &
  tpccload -M $MULT -h -g -b   51 -e  100 >
${LDIR}/hist02.data &
  tpccload -M $MULT -h -g -b  101 -e  150 >
${LDIR}/hist03.data &
  tpccload -M $MULT -h -g -b  151 -e  200 >
${LDIR}/hist04.data &
  tpccload -M $MULT -h -g -b  201 -e  250 >
${LDIR}/hist05.data &
  tpccload -M $MULT -h -g -b  251 -e  300 >
${LDIR}/hist06.data &
  tpccload -M $MULT -h -g -b  301 -e  350 >
${LDIR}/hist07.data &
  tpccload -M $MULT -h -g -b  351 -e  400 >
${LDIR}/hist08.data &
  tpccload -M $MULT -h -g -b  401 -e  450 >
${LDIR}/hist09.data &
  tpccload -M $MULT -h -g -b  451 -e  500 >
${LDIR}/hist10.data &
  tpccload -M $MULT -h -g -b  501 -e  550 >
${LDIR}/hist11.data &
  tpccload -M $MULT -h -g -b  551 -e  600 >
${LDIR}/hist12.data &
  tpccload -M $MULT -h -g -b  601 -e  650 >
${LDIR}/hist13.data &
  tpccload -M $MULT -h -g -b  651 -e  700 >
${LDIR}/hist14.data &
  tpccload -M $MULT -h -g -b  701 -e  750 >
${LDIR}/hist15.data &
  tpccload -M $MULT -h -g -b  751 -e  800 >
${LDIR}/hist16.data &
  tpccload -M $MULT -h -g -b  801 -e  850 >
${LDIR}/hist17.data &
  tpccload -M $MULT -h -g -b  851 -e  900 >
${LDIR}/hist18.data &
  tpccload -M $MULT -h -g -b  901 -e  950 >
${LDIR}/hist19.data &
  tpccload -M $MULT -h -g -b  951 -e 1000 >
${LDIR}/hist20.data &
  tpccload -M $MULT -h -g -b 1001 -e 1050 >
${LDIR}/hist21.data &
  tpccload -M $MULT -h -g -b 1051 -e 1100 >
${LDIR}/hist22.data &
  tpccload -M $MULT -h -g -b 1101 -e 1150 >
${LDIR}/hist23.data &
  tpccload -M $MULT -h -g -b 1151 -e 1200 >
${LDIR}/hist24.data &
  tpccload -M $MULT -h -g -b 1201 -e 1250 >
${LDIR}/hist25.data &
  tpccload -M $MULT -h -g -b 1251 -e 1300 >
${LDIR}/hist26.data &
  tpccload -M $MULT -h -g -b 1301 -e 1350 >
${LDIR}/hist27.data &
  tpccload -M $MULT -h -g -b 1351 -e 1400 >
${LDIR}/hist28.data &
```

```
  tpccload -M $MULT -h -g -b 1401 -e 1450 >
${LDIR}/hist29.data &
  tpccload -M $MULT -h -g -b 1451 -e 1500 >
${LDIR}/hist30.data &
  tpccload -M $MULT -h -g -b 1501 -e 1550 >
${LDIR}/hist31.data &
  tpccload -M $MULT -h -g -b 1551 -e 1600 >
${LDIR}/hist32.data &
  tpccload -M $MULT -h -g -b 1601 -e 1650 >
${LDIR}/hist33.data &
  tpccload -M $MULT -h -g -b 1651 -e 1700 >
${LDIR}/hist34.data &
  tpccload -M $MULT -h -g -b 1701 -e 1750 >
${LDIR}/hist35.data &
  tpccload -M $MULT -h -g -b 1751 -e 1800 >
${LDIR}/hist36.data &
  tpccload -M $MULT -h -g -b 1801 -e 1850 >
${LDIR}/hist37.data &
  tpccload -M $MULT -h -g -b 1851 -e 1900 >
${LDIR}/hist38.data &
  tpccload -M $MULT -h -g -b 1901 -e 1950 >
${LDIR}/hist39.data &
  tpccload -M $MULT -h -g -b 1951 -e 2000 >
${LDIR}/hist40.data &
  tpccload -M $MULT -h -g -b 2001 -e 2050 >
${LDIR}/hist41.data &
  tpccload -M $MULT -h -g -b 2051 -e 2100 >
${LDIR}/hist42.data &
  tpccload -M $MULT -h -g -b 2101 -e 2150 >
${LDIR}/hist43.data &
  tpccload -M $MULT -h -g -b 2151 -e 2200 >
${LDIR}/hist44.data &
  tpccload -M $MULT -h -g -b 2201 -e 2250 >
${LDIR}/hist45.data &
  tpccload -M $MULT -h -g -b 2251 -e 2300 >
${LDIR}/hist46.data &
  tpccload -M $MULT -h -g -b 2301 -e 2350 >
${LDIR}/hist47.data &
  tpccload -M $MULT -h -g -b 2351 -e 2400 >
${LDIR}/hist48.data &
  tpccload -M $MULT -h -g -b 2401 -e 2450 >
${LDIR}/hist49.data &
  tpccload -M $MULT -h -g -b 2451 -e 2500 >
${LDIR}/hist50.data &
  tpccload -M $MULT -h -g -b 2501 -e 2550 >
${LDIR}/hist51.data &
  tpccload -M $MULT -h -g -b 2551 -e 2600 >
${LDIR}/hist52.data &
  tpccload -M $MULT -h -g -b 2601 -e 2650 >
${LDIR}/hist53.data &
  tpccload -M $MULT -h -g -b 2651 -e 2700 >
${LDIR}/hist54.data &
  tpccload -M $MULT -h -g -b 2701 -e 2750 >
${LDIR}/hist55.data &
  tpccload -M $MULT -h -g -b 2751 -e 2800 >
${LDIR}/hist56.data &
  tpccload -M $MULT -h -g -b 2801 -e 2850 >
${LDIR}/hist57.data &
  tpccload -M $MULT -h -g -b 2851 -e 2900 >
${LDIR}/hist58.data &
  tpccload -M $MULT -h -g -b 2901 -e 2950 >
${LDIR}/hist59.data &
  tpccload -M $MULT -h -g -b 2951 -e 3000 >
${LDIR}/hist60.data &
  wait

  cat ${LDIR}/hist01.data \
    ${LDIR}/hist02.data \
    ${LDIR}/hist03.data \
    ${LDIR}/hist04.data \
    ${LDIR}/hist05.data > ${LDIR}/hist1.dat &
  cat ${LDIR}/hist06.data \
    ${LDIR}/hist07.data \
    ${LDIR}/hist08.data \
    ${LDIR}/hist09.data \
    ${LDIR}/hist10.data > ${LDIR}/hist2.dat &
  cat ${LDIR}/hist11.data \
    ${LDIR}/hist12.data \
    ${LDIR}/hist13.data \
    ${LDIR}/hist14.data \
    ${LDIR}/hist15.data > ${LDIR}/hist3.dat &
  cat ${LDIR}/hist16.data \
    ${LDIR}/hist17.data \
    ${LDIR}/hist18.data \
    ${LDIR}/hist19.data \
    ${LDIR}/hist20.data > ${LDIR}/hist4.dat &
  cat ${LDIR}/hist21.data \
    ${LDIR}/hist22.data \
    ${LDIR}/hist23.data \
    ${LDIR}/hist24.data \
    ${LDIR}/hist25.data > ${LDIR}/hist5.dat &
  cat ${LDIR}/hist26.data \
    ${LDIR}/hist27.data \
    ${LDIR}/hist28.data \
    ${LDIR}/hist29.data \
    ${LDIR}/hist30.data > ${LDIR}/hist6.dat &
  cat ${LDIR}/hist31.data \
    ${LDIR}/hist32.data \
    ${LDIR}/hist33.data \
    ${LDIR}/hist34.data \
    ${LDIR}/hist35.data > ${LDIR}/hist7.dat &
  cat ${LDIR}/hist36.data \
    ${LDIR}/hist37.data \
    ${LDIR}/hist38.data \
    ${LDIR}/hist39.data \
    ${LDIR}/hist40.data > ${LDIR}/hist8.dat &
  cat ${LDIR}/hist41.data \
    ${LDIR}/hist42.data \
    ${LDIR}/hist43.data \
    ${LDIR}/hist44.data \
    ${LDIR}/hist45.data > ${LDIR}/hist9.dat &
  cat ${LDIR}/hist46.data \
    ${LDIR}/hist47.data \
    ${LDIR}/hist48.data \
    ${LDIR}/hist49.data \
    ${LDIR}/hist50.data > ${LDIR}/hist10.dat &
  cat ${LDIR}/hist51.data \
    ${LDIR}/hist52.data \
    ${LDIR}/hist53.data \
    ${LDIR}/hist54.data \
    ${LDIR}/hist55.data > ${LDIR}/hist11.dat &
  cat ${LDIR}/hist56.data \
    ${LDIR}/hist57.data \
    ${LDIR}/hist58.data \
    ${LDIR}/hist59.data \
    ${LDIR}/hist60.data > ${LDIR}/hist12.dat &
  wait

  rm ${LDIR}/hist*.data

  echo "[END] tpccload history"
  echo ------- `date` -------
  echo

  exit
fi

#########################################
###################
#  Load history TO database
#########################################
###################

echo "[BEGIN] sqlldr history"
echo ------- `date` -------
echo

sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist1.log \
    bad=hist1.bad data=${LDIR}/hist1.dat
discard=hist1.dsc \
    file=/dev/rdsk/hda1108 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist2.log \
    bad=hist2.bad data=${LDIR}/hist2.dat
discard=hist2.dsc \
    file=/dev/rdsk/hda1308 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist3.log \
    bad=hist3.bad data=${LDIR}/hist3.dat
discard=hist3.dsc \
    file=/dev/rdsk/hda2108 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist4.log \
    bad=hist4.bad data=${LDIR}/hist4.dat
discard=hist4.dsc \
    file=/dev/rdsk/hda2308 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist5.log \
    bad=hist5.bad data=${LDIR}/hist5.dat
discard=hist5.dsc \
    file=/dev/rdsk/hda3108 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist6.log \
    bad=hist6.bad data=${LDIR}/hist6.dat
discard=hist6.dsc \
    file=/dev/rdsk/hda3308 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist7.log \
    bad=hist7.bad data=${LDIR}/hist7.dat
discard=hist7.dsc \
    file=/dev/rdsk/hda4108 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist8.log \
    bad=hist8.bad data=${LDIR}/hist8.dat
discard=hist8.dsc \
    file=/dev/rdsk/hda4308 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist9.log \
    bad=hist9.bad data=${LDIR}/hist9.dat
discard=hist9.dsc \
    file=/dev/rdsk/hda5108 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist10.log \
    bad=hist10.bad data=${LDIR}/hist10.dat
discard=hist10.dsc \
    file=/dev/rdsk/hda5308 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist11.log \
    bad=hist11.bad data=${LDIR}/hist11.dat
discard=hist11.dsc \
    file=/dev/rdsk/hda6108 &
sqlldr tpcc/tpcc control=$TPCC_LOADER/hist.ctl
log=hist12.log \
```

```
    bad=hist12.bad data=${LDIR}/hist12.dat
discard=hist12.dsc \
    file=/dev/rdsk/hda6308 &
wait

echo "[END] sqlldr history"
echo ------- `date` -------
echo

#########################################
##
####
####   pload_neword.sh
####
####

#!/bin/sh

#
# $Header: pload.sh 7030100.1 96/05/02 19:06:06
plai Generic<base> $ Copyr (c) 1995 Oracle
#

#
#=====================================
============================+
#       Copyright (c) 1996  Oracle Corp, Redwood
Shores, CA       |
#                 OPEN SYSTEMS PERFORMANCE
GROUP                |
#                    All Rights Reserved
|
#=====================================
============================+
# FILENAME
#     pload_neword.sh
# DESCRIPTION
#     Usage: pload_neword.sh [options]
#           -mu <multiplier>     (# of warehouses)
#=====================================
==========================
#

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_LOADER=$BENCH_HOME/tpcc/loader

PROG=$0
OUTDIR=outdir
LDIR=/data
MULT=3000

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)? [3000]" $C
  read MULT
  if [ "$MULT" = "" ]
  then
     MULT=3000
  fi
fi


if [ ! -d $OUTDIR ]
then
  mkdir $OUTDIR
fi

PROG=`basename $PROG`
#########################################
####################
#  Load new-order TO file
#########################################
####################
if [ $PROG = pre_pload_neword.sh ]
then
  echo "[BEGIN] tpccload new-order"
  echo ------- `date` -------
  echo

  rm ${LDIR}/neword*.dat

  tpccload -M $MULT -n -g -b   01 -e   250 >
${LDIR}/neword1.dat &
  tpccload -M $MULT -n -g -b  251 -e   500 >
${LDIR}/neword2.dat &
  tpccload -M $MULT -n -g -b  501 -e   750 >
${LDIR}/neword3.dat &
  tpccload -M $MULT -n -g -b  751 -e  1000 >
${LDIR}/neword4.dat &
  tpccload -M $MULT -n -g -b 1001 -e  1250 >
${LDIR}/neword5.dat &
  tpccload -M $MULT -n -g -b 1251 -e  1500 >
${LDIR}/neword6.dat &
  tpccload -M $MULT -n -g -b 1501 -e  1750 >
${LDIR}/neword7.dat &
  tpccload -M $MULT -n -g -b 1751 -e  2000 >
${LDIR}/neword8.dat &
  tpccload -M $MULT -n -g -b 2001 -e  2250 >
${LDIR}/neword9.dat &
  tpccload -M $MULT -n -g -b 2251 -e  2500 >
${LDIR}/neword10.dat &
  tpccload -M $MULT -n -g -b 2501 -e  2750 >
${LDIR}/neword11.dat &
  tpccload -M $MULT -n -g -b 2751 -e  3000 >
${LDIR}/neword12.dat &
  wait

  echo "[END] tpccload new-order"
  echo ------- `date` -------
  echo

  exit
fi

#########################################
####################
#  Load new-order TO database
#########################################
####################

echo "[BEGIN] sqlldr new-order"
echo ------- `date` -------
echo

sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword1.log \
    bad=neword1.bad data=${LDIR}/neword1.dat
discard=neword1.dsc \
    file=/dev/rvol/NORD_0 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword2.log \
    bad=neword2.bad data=${LDIR}/neword2.dat
discard=neword2.dsc \
    file=/dev/rvol/NORD_1 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword3.log \
    bad=neword3.bad data=${LDIR}/neword3.dat
discard=neword3.dsc \
    file=/dev/rvol/NORD_2 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword4.log \
    bad=neword4.bad data=${LDIR}/neword4.dat
discard=neword4.dsc \
    file=/dev/rvol/NORD_3 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword5.log \
    bad=neword5.bad data=${LDIR}/neword5.dat
discard=neword5.dsc \
    file=/dev/rvol/NORD_4 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword6.log \
    bad=neword6.bad data=${LDIR}/neword6.dat
discard=neword6.dsc \
    file=/dev/rvol/NORD_5 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword7.log \
    bad=neword7.bad data=${LDIR}/neword7.dat
discard=neword7.dsc \
    file=/dev/rvol/NORD_6 &
```

```
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword8.log \
    bad=neword8.bad data=${LDIR}/neword8.dat
discard=neword8.dsc \
    file=/dev/rvol/NORD_7 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword9.log \
    bad=neword9.bad data=${LDIR}/neword9.dat
discard=neword9.dsc \
    file=/dev/rvol/NORD_8 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword10.log \
    bad=neword10.bad
data=${LDIR}/neword10.dat
discard=neword10.dsc \
    file=/dev/rvol/NORD_9 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword11.log \
    bad=neword11.bad
data=${LDIR}/neword11.dat
discard=neword11.dsc \
    file=/dev/rvol/NORD_10 &
sqlldr tpcc/tpcc
control=$TPCC_LOADER/neword.ctl
log=neword12.log \
    bad=neword12.bad
data=${LDIR}/neword12.dat
discard=neword12.dsc \
    file=/dev/rvol/NORD_11 &
wait

echo "[END] sqlldr new-order"
echo ------- `date` -------
echo

###########################################
##
####
####   pload_ord_ordl.sh
####
####

#!/bin/sh

#
# $Header: pload.sh 7030100.1 96/05/02 19:06:06
plai Generic<base> $ Copyr (c) 1995 Oracle
#

#
#=================================
=============================+
#       Copyright (c) 1996  Oracle Corp, Redwood
Shores, CA      |
#                OPEN SYSTEMS PERFORMANCE
GROUP                |
#                    All Rights Reserved
|
#=================================
=============================+
# FILENAME
#     pload_ord_ordl.sh
# DESCRIPTION
#     Usage: pload_ord_ordl.sh [options]
#            -mu <multiplier>     (# of warehouses)
#=================================
=============================
#

BENCH_HOME=$ORACLE_HOME/bench/tpc
TPCC_SOURCE=$BENCH_HOME/tpcc/source
TPCC_LOADER=$BENCH_HOME/tpcc/loader

PROG=$0
OUTDIR=outdir
LDIR=/data
MULT=3000

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)? [3000]" $C
  read MULT
  if [ "$MULT" = "" ]
  then
     MULT=3000
  fi
fi


if [ ! -d $OUTDIR ]
then
  mkdir $OUTDIR
fi

PROG=`basename $PROG`

#########################################
##################
#  Load order and order_line TO file
#########################################
##################
if [ $PROG = pre_pload_ord_ordl.sh ]
then
  echo "[BEGIN] tpccload order, order_line"
  echo ------- `date` -------
  echo

  rm ${LDIR}/ord*.dat ${LDIR}/ord*.data

  tpccload -M $MULT -o ${LDIR}/ordline1.dat  -g -
b   1 -e  40 > ${LDIR}/order01.data &
  tpccload -M $MULT -o ${LDIR}/ordline2.dat  -g -
b  41 -e  80 > ${LDIR}/order02.data &
  tpccload -M $MULT -o ${LDIR}/ordline3.dat  -g -
b  81 -e  120 > ${LDIR}/order03.data &
  tpccload -M $MULT -o ${LDIR}/ordline4.dat  -g -
b 121 -e  160 > ${LDIR}/order04.data &
  tpccload -M $MULT -o ${LDIR}/ordline5.dat  -g -
b 161 -e  200 > ${LDIR}/order05.data &
  tpccload -M $MULT -o ${LDIR}/ordline6.dat  -g -
b 201 -e  240 > ${LDIR}/order06.data &
  tpccload -M $MULT -o ${LDIR}/ordline7.dat  -g -
b 241 -e  280 > ${LDIR}/order07.data &
  tpccload -M $MULT -o ${LDIR}/ordline8.dat  -g -
b 281 -e  320 > ${LDIR}/order08.data &
  tpccload -M $MULT -o ${LDIR}/ordline9.dat  -g -
b 321 -e  360 > ${LDIR}/order09.data &
  tpccload -M $MULT -o ${LDIR}/ordline10.dat -g -
b 361 -e  400 > ${LDIR}/order10.data &
  tpccload -M $MULT -o ${LDIR}/ordline11.dat -g -
b 401 -e  440 > ${LDIR}/order11.data &
  tpccload -M $MULT -o ${LDIR}/ordline12.dat -g -
b 441 -e  480 > ${LDIR}/order12.data &
  tpccload -M $MULT -o ${LDIR}/ordline13.dat -g -
b 481 -e  520 > ${LDIR}/order13.data &
  tpccload -M $MULT -o ${LDIR}/ordline14.dat -g -
b 521 -e  560 > ${LDIR}/order14.data &
  tpccload -M $MULT -o ${LDIR}/ordline15.dat -g -
b 561 -e  600 > ${LDIR}/order15.data &
  tpccload -M $MULT -o ${LDIR}/ordline16.dat -g -
b 601 -e  640 > ${LDIR}/order16.data &
  tpccload -M $MULT -o ${LDIR}/ordline17.dat -g -
b 641 -e  680 > ${LDIR}/order17.data &
  tpccload -M $MULT -o ${LDIR}/ordline18.dat -g -
b 681 -e  720 > ${LDIR}/order18.data &
  tpccload -M $MULT -o ${LDIR}/ordline19.dat -g -
b 721 -e  760 > ${LDIR}/order19.data &
  tpccload -M $MULT -o ${LDIR}/ordline20.dat -g -
b 761 -e  800 > ${LDIR}/order20.data &
  tpccload -M $MULT -o ${LDIR}/ordline21.dat -g -
b 801 -e  840 > ${LDIR}/order21.data &
  tpccload -M $MULT -o ${LDIR}/ordline22.dat -g -
b 841 -e  880 > ${LDIR}/order22.data &
  tpccload -M $MULT -o ${LDIR}/ordline23.dat -g -
b 881 -e  920 > ${LDIR}/order23.data &
  tpccload -M $MULT -o ${LDIR}/ordline24.dat -g -
b 921 -e  960 > ${LDIR}/order24.data &
  tpccload -M $MULT -o ${LDIR}/ordline25.dat -g -
b 961 -e 1000 > ${LDIR}/order25.data &
  tpccload -M $MULT -o ${LDIR}/ordline26.dat -g -
b 1001 -e 1040 > ${LDIR}/order26.data &
  tpccload -M $MULT -o ${LDIR}/ordline27.dat -g -
b 1041 -e 1080 > ${LDIR}/order27.data &
```

```
    tpccload -M $MULT -o ${LDIR}/ordline28.dat -g -
b 1081 -e 1120 > ${LDIR}/order28.data &
    tpccload -M $MULT -o ${LDIR}/ordline29.dat -g -
b 1121 -e 1160 > ${LDIR}/order29.data &
    tpccload -M $MULT -o ${LDIR}/ordline30.dat -g -
b 1161 -e 1200 > ${LDIR}/order30.data &
    tpccload -M $MULT -o ${LDIR}/ordline31.dat -g -
b 1201 -e 1240 > ${LDIR}/order31.data &
    tpccload -M $MULT -o ${LDIR}/ordline32.dat -g -
b 1241 -e 1280 > ${LDIR}/order32.data &
    tpccload -M $MULT -o ${LDIR}/ordline33.dat -g -
b 1281 -e 1320 > ${LDIR}/order33.data &
    tpccload -M $MULT -o ${LDIR}/ordline34.dat -g -
b 1321 -e 1360 > ${LDIR}/order34.data &
    tpccload -M $MULT -o ${LDIR}/ordline35.dat -g -
b 1361 -e 1400 > ${LDIR}/order35.data &
    tpccload -M $MULT -o ${LDIR}/ordline36.dat -g -
b 1401 -e 1440 > ${LDIR}/order36.data &
    tpccload -M $MULT -o ${LDIR}/ordline37.dat -g -
b 1441 -e 1480 > ${LDIR}/order37.data &
    tpccload -M $MULT -o ${LDIR}/ordline38.dat -g -
b 1481 -e 1520 > ${LDIR}/order38.data &
    tpccload -M $MULT -o ${LDIR}/ordline39.dat -g -
b 1521 -e 1560 > ${LDIR}/order39.data &
    tpccload -M $MULT -o ${LDIR}/ordline40.dat -g -
b 1561 -e 1600 > ${LDIR}/order40.data &
wait
    tpccload -M $MULT -o ${LDIR}/ordline41.dat -g -
b 1601 -e 1640 > ${LDIR}/order41.data &
    tpccload -M $MULT -o ${LDIR}/ordline42.dat -g -
b 1641 -e 1680 > ${LDIR}/order42.data &
    tpccload -M $MULT -o ${LDIR}/ordline43.dat -g -
b 1681 -e 1720 > ${LDIR}/order43.data &
    tpccload -M $MULT -o ${LDIR}/ordline44.dat -g -
b 1721 -e 1760 > ${LDIR}/order44.data &
    tpccload -M $MULT -o ${LDIR}/ordline45.dat -g -
b 1761 -e 1800 > ${LDIR}/order45.data &
    tpccload -M $MULT -o ${LDIR}/ordline46.dat -g -
b 1801 -e 1840 > ${LDIR}/order46.data &
    tpccload -M $MULT -o ${LDIR}/ordline47.dat -g -
b 1841 -e 1880 > ${LDIR}/order47.data &
    tpccload -M $MULT -o ${LDIR}/ordline48.dat -g -
b 1881 -e 1920 > ${LDIR}/order48.data &
    tpccload -M $MULT -o ${LDIR}/ordline49.dat -g -
b 1921 -e 1960 > ${LDIR}/order49.data &
    tpccload -M $MULT -o ${LDIR}/ordline50.dat -g -
b 1961 -e 2000 > ${LDIR}/order50.data &
    tpccload -M $MULT -o ${LDIR}/ordline51.dat -g -
b 2001 -e 2040 > ${LDIR}/order51.data &
    tpccload -M $MULT -o ${LDIR}/ordline52.dat -g -
b 2041 -e 2080 > ${LDIR}/order52.data &
    tpccload -M $MULT -o ${LDIR}/ordline53.dat -g -
b 2081 -e 2120 > ${LDIR}/order53.data &
    tpccload -M $MULT -o ${LDIR}/ordline54.dat -g -
b 2121 -e 2160 > ${LDIR}/order54.data &
    tpccload -M $MULT -o ${LDIR}/ordline55.dat -g -
b 2161 -e 2200 > ${LDIR}/order55.data &
    tpccload -M $MULT -o ${LDIR}/ordline56.dat -g -
b 2201 -e 2240 > ${LDIR}/order56.data &
    tpccload -M $MULT -o ${LDIR}/ordline57.dat -g -
b 2241 -e 2280 > ${LDIR}/order57.data &
    tpccload -M $MULT -o ${LDIR}/ordline58.dat -g -
b 2281 -e 2320 > ${LDIR}/order58.data &
    tpccload -M $MULT -o ${LDIR}/ordline59.dat -g -
b 2321 -e 2360 > ${LDIR}/order59.data &
    tpccload -M $MULT -o ${LDIR}/ordline60.dat -g -
b 2361 -e 2400 > ${LDIR}/order60.data &
    tpccload -M $MULT -o ${LDIR}/ordline61.dat -g -
b 2401 -e 2440 > ${LDIR}/order61.data &
    tpccload -M $MULT -o ${LDIR}/ordline62.dat -g -
b 2441 -e 2480 > ${LDIR}/order62.data &
    tpccload -M $MULT -o ${LDIR}/ordline63.dat -g -
b 2481 -e 2520 > ${LDIR}/order63.data &
    tpccload -M $MULT -o ${LDIR}/ordline64.dat -g -
b 2521 -e 2560 > ${LDIR}/order64.data &
    tpccload -M $MULT -o ${LDIR}/ordline65.dat -g -
b 2561 -e 2600 > ${LDIR}/order65.data &
    tpccload -M $MULT -o ${LDIR}/ordline66.dat -g -
b 2601 -e 2640 > ${LDIR}/order66.data &
    tpccload -M $MULT -o ${LDIR}/ordline67.dat -g -
b 2641 -e 2680 > ${LDIR}/order67.data &
    tpccload -M $MULT -o ${LDIR}/ordline68.dat -g -
b 2681 -e 2720 > ${LDIR}/order68.data &
    tpccload -M $MULT -o ${LDIR}/ordline69.dat -g -
b 2721 -e 2760 > ${LDIR}/order69.data &
    tpccload -M $MULT -o ${LDIR}/ordline70.dat -g -
b 2761 -e 2800 > ${LDIR}/order70.data &
    tpccload -M $MULT -o ${LDIR}/ordline71.dat -g -
b 2801 -e 2840 > ${LDIR}/order71.data &
    tpccload -M $MULT -o ${LDIR}/ordline72.dat -g -
b 2841 -e 2880 > ${LDIR}/order72.data &
    tpccload -M $MULT -o ${LDIR}/ordline73.dat -g -
b 2881 -e 2920 > ${LDIR}/order73.data &
    tpccload -M $MULT -o ${LDIR}/ordline74.dat -g -
b 2921 -e 2960 > ${LDIR}/order74.data &
    tpccload -M $MULT -o ${LDIR}/ordline75.dat -g -
b 2961 -e 3000 > ${LDIR}/order75.data &
    wait

    cat ${LDIR}/order01.data \
        ${LDIR}/order02.data \
        ${LDIR}/order03.data > ${LDIR}/order1.dat &
    cat ${LDIR}/order04.data \
        ${LDIR}/order05.data \
        ${LDIR}/order06.data > ${LDIR}/order2.dat &
    cat ${LDIR}/order07.data \
        ${LDIR}/order08.data \
        ${LDIR}/order09.data > ${LDIR}/order3.dat &
    cat ${LDIR}/order10.data \
        ${LDIR}/order11.data \
        ${LDIR}/order12.data > ${LDIR}/order4.dat &
    cat ${LDIR}/order13.data \
        ${LDIR}/order14.data \
        ${LDIR}/order15.data > ${LDIR}/order5.dat &
    cat ${LDIR}/order16.data \
        ${LDIR}/order17.data \
        ${LDIR}/order18.data > ${LDIR}/order6.dat &
    cat ${LDIR}/order19.data \
        ${LDIR}/order20.data \
        ${LDIR}/order21.data > ${LDIR}/order7.dat &
    cat ${LDIR}/order22.data \
        ${LDIR}/order23.data \
        ${LDIR}/order24.data > ${LDIR}/order8.dat &
    cat ${LDIR}/order25.data \
        ${LDIR}/order26.data \
        ${LDIR}/order27.data > ${LDIR}/order9.dat &
    cat ${LDIR}/order28.data \
        ${LDIR}/order29.data \
        ${LDIR}/order30.data > ${LDIR}/order10.dat &
    cat ${LDIR}/order31.data \
        ${LDIR}/order32.data \
        ${LDIR}/order33.data > ${LDIR}/order11.dat &
    cat ${LDIR}/order34.data \
        ${LDIR}/order35.data \
        ${LDIR}/order36.data > ${LDIR}/order12.dat &
    cat ${LDIR}/order37.data \
        ${LDIR}/order38.data \
        ${LDIR}/order39.data > ${LDIR}/order13.dat &
    cat ${LDIR}/order40.data \
        ${LDIR}/order41.data \
        ${LDIR}/order42.data > ${LDIR}/order14.dat &
    cat ${LDIR}/order43.data \
        ${LDIR}/order44.data \
        ${LDIR}/order45.data > ${LDIR}/order15.dat &
    cat ${LDIR}/order46.data \
        ${LDIR}/order47.data \
        ${LDIR}/order48.data > ${LDIR}/order16.dat &
    cat ${LDIR}/order49.data \
        ${LDIR}/order50.data \
        ${LDIR}/order51.data > ${LDIR}/order17.dat &
    cat ${LDIR}/order52.data \
        ${LDIR}/order53.data \
        ${LDIR}/order54.data > ${LDIR}/order18.dat &
    cat ${LDIR}/order55.data \
        ${LDIR}/order56.data \
        ${LDIR}/order57.data > ${LDIR}/order19.dat &
    cat ${LDIR}/order58.data \
        ${LDIR}/order59.data \
        ${LDIR}/order60.data > ${LDIR}/order20.dat &
    cat ${LDIR}/order61.data \
        ${LDIR}/order62.data \
        ${LDIR}/order63.data > ${LDIR}/order21.dat &
    cat ${LDIR}/order64.data \
        ${LDIR}/order65.data \
        ${LDIR}/order66.data > ${LDIR}/order22.dat &
    cat ${LDIR}/order67.data \
        ${LDIR}/order68.data \
        ${LDIR}/order69.data > ${LDIR}/order23.dat &
    cat ${LDIR}/order70.data \
        ${LDIR}/order71.data \
        ${LDIR}/order72.data > ${LDIR}/order24.dat &
    cat ${LDIR}/order73.data \
        ${LDIR}/order74.data \
        ${LDIR}/order75.data > ${LDIR}/order25.dat &
    wait

    rm ${LDIR}/ord*.data

    echo "[END] tpccload hist, new-order, order,
order_line"
    echo ------- `date` -------
    echo

    exit
fi

#########################################
###################
#  Load order and order_line TO database
#########################################
###################

echo "[BEGIN] sqlldr order, order_line"
echo ------- `date` -------
echo

DATA=1
SUBD=0
while [ $DATA -le 25 ]
do
    sqlldr tpcc/tpcc
control=$TPCC_LOADER/order.ctl \
                log=order${DATA}.log \
```

```
            bad=order${DATA}.bad \
            data=${LDIR}/order${DATA}.dat \
            discard=order${DATA}.dsc \
            file=/dev/rvol/ORD_${SUBD} &
  DATA=`expr $DATA + 1`
  SUBD=`expr $SUBD + 1`
done

DATA=1
SUBD=0
while [ $DATA -le 75 ]
do
  sqlldr tpcc/tpcc
control=$TPCC_LOADER/ordline.ctl \
            log=ordline${DATA}.log \
            bad=ordline${DATA}.bad \
            data=${LDIR}/ordline${DATA}.dat \
            discard=ordline${DATA}.dsc \
            file=/dev/rvol/ORDL_${SUBD} &
  DATA=`expr $DATA + 1`
  SUBD=`expr $SUBD + 1`
done
wait

echo "[END] sqlldr, order_line"
echo ------- `date` -------
echo

##############################################
##
####
####  buf_pool.sql
####
####


rem
rem
=====================================
==========================+
rem        Copyright (c) By Fj permance group.
|
rem
=====================================
==========================+
rem  FILENAME
rem      buf_pool.sql
rem
=====================================
==========================
rem

set timing on;

alter cluster ccluster storage (buffer_pool recycle);

quit;

##############################################
##
####
####  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) 1996  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

drop index iwarehouse;
drop index idistrict;
drop index icustomer;
drop index icustomer2;
drop index iitem;

set timing on

create unique index iwarehouse on
warehouse(w_id)
      tablespace ware
      initrans 3
      storage (initial 300K next 20K pctincrease 0)
pctfree 1;

create unique index idistrict on district(d_w_id,
d_id)
      tablespace ware
      initrans 3
      storage (initial 300K 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 60M next 60M
                        minextents 5 maxextents
unlimited
                        pctincrease 0) pctfree 1;

create unique index icustomer2 on
customer(c_last, c_w_id, c_d_id, c_first, c_id)
      tablespace icust2
      initrans 3
      parallel 10
      storage (initial 50M next 50M
                        minextents 5 maxextents
unlimited
                        pctincrease 0) pctfree 1;

exit;


##############################################
##
####
####  tpcc_ix2.sql
####
####

rem
```

```
rem
=====================================
============================+
rem        Copyright (c) 1996  Oracle Corp,
Redwood Shores, CA      |
rem                 OPEN SYSTEMS
PERFORMANCE GROUP               |
rem                     All Rights Reserved
|
rem
=====================================
============================+
rem  FILENAME
rem       tpcc_ix2.sql
rem  DESCRIPTION
rem       Create indexes for TPC-C database.
rem
=====================================
============================
rem

drop index iorders;
drop index iorders2;
drop index inew_order;

set timing on

create unique index iorders on orders(o_w_id,
o_d_id, o_id)
      tablespace iord1
      initrans 3
      parallel 10
      pctfree  1
      storage (initial 57M next 57M pctincrease 0
            maxextents unlimited
            freelist groups 13 freelists 20);

create unique index iorders2 on orders(o_w_id,
o_d_id, o_c_id, o_id)
      tablespace iord2
      initrans 3
      parallel 10
      pctfree  1
      storage (initial 50M next 50M pctincrease 0
            maxextents unlimited
            freelist groups 13 freelists 20);

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 51M next 51M pctincrease 0
            maxextents unlimited
            freelist groups 13 freelists 20);

exit;


##########################################
##
####
####   tpcc_ix3.sql
####
####

rem
```

```
rem
=====================================
============================+
rem        Copyright (c) 1996  Oracle Corp,
Redwood Shores, CA      |
rem                 OPEN SYSTEMS
PERFORMANCE GROUP               |
rem                     All Rights Reserved
|
rem
=====================================
============================+
rem  FILENAME
rem       tpcc_ix4.sql
rem  DESCRIPTION
rem       Create indexes for TPC-C database.
rem
=====================================
============================
rem

drop index istock;

set timing on

create unique index istock on stock(s_i_id,
s_w_id)
      tablespace istk
      initrans 3
      parallel 10
      storage (initial 106M next 106M
            maxextents unlimited pctincrease 0)
pctfree 1;

exit;


##########################################
##
####
####   tpcc_ix4.sql
####
####

rem
rem
=====================================
============================+
rem        Copyright (c) 1996  Oracle Corp,
Redwood Shores, CA      |
rem                 OPEN SYSTEMS
PERFORMANCE GROUP               |
rem                     All Rights Reserved
|
rem
=====================================
============================+
rem  FILENAME
rem       tpcc_ix4.sql
rem  DESCRIPTION
rem       Create indexes for TPC-C database.
rem
=====================================
============================
rem

drop index iorder_line;
```

```
set timing on

create unique index iorder_line on
order_line(ol_w_id, ol_d_id, ol_o_id, ol_number)
      tablespace iordl
      initrans 4
      parallel 15
      pctfree  1
      storage (initial 600M next 200M pctincrease 0
                        minextents 10
maxextents unlimited
            freelist groups 13 freelists 20);

exit;


##########################################
##
####
####   tpcc_rol.sql
####
####

rem
rem
=====================================
============================+
rem        Copyright (c) 1996  Oracle Corp,
Redwood Shores, CA      |
rem                 OPEN SYSTEMS
PERFORMANCE GROUP               |
rem                     All Rights Reserved
|
rem
=====================================
============================+
rem  FILENAME
rem       tpcc_rol.sql
rem  DESCRIPTION
rem       Create rollback segments for TPCC
database.
rem
=====================================
============================
rem

  set timing on;

  host date;

  CREATE ROLLBACK SEGMENT t1
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t2
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t3
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t4
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t5
    TABLESPACE roll5
```

```
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t6
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t7
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t8
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t9
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t10
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t11
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t12
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t13
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t14
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t15
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t16
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t17
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t18
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t19
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t20
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t21
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t22
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t23
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t24
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t25
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t26
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t27
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t28
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t29
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t30
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t31
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t32
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t33
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t34
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t35
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t36
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t37
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t38
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t39
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t40
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t41
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t42
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t43
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t44
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t45
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t46
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t47
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t48
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t49
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t50
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t51
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t52
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t53
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t54
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t55
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t56
    TABLESPACE roll1
```

```
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t57
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t58
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t59
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t60
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t61
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t62
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t63
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t64
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t65
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t66
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t67
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t68
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t69
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t70
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t71
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t72
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t73
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t74
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t75
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t76
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t77
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t78
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t79
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t80
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t81
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t82
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t83
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t84
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t85
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t86
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t87
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t88
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t89
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t90
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t91
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t92
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t93
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t94
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t95
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t96
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t97
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t98
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t99
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t100
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t101
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t102
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t103
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t104
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t105
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t106
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t107
    TABLESPACE roll2
```

```
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t108
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t109
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t110
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t111
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t112
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t113
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t114
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t115
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t116
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t117
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t118
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t119
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t120
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t121
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t122
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t123
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t124
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t125
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t126
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t127
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t128
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t129
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t130
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t131
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t132
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t133
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t134
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t135
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t136
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t137
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t138
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t139
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t140
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t141
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t142
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t143
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t144
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t145
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t146
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t147
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t148
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t149
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t150
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t151
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t152
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t153
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t154
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t155
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t156
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t157
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t158
    TABLESPACE roll3
```

```
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t159
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t160
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t161
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t162
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t163
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t164
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t165
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t166
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t167
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t168
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t169
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t170
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t171
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t172
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t173
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t174
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t175
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t176
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t177
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t178
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t179
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t180
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t181
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t182
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t183
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t184
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t185
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t186
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t187
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t188
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t189
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t190
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t191
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t192
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t193
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t194
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t195
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t196
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t197
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t198
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t199
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t200
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t201
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t202
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t203
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t204
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t205
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t206
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t207
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t208
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t209
    TABLESPACE roll4
```

```
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t210
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t211
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t212
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t213
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t214
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t215
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t216
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t217
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t218
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t219
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t220
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t221
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t222
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t223
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t224
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t225
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t226
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t227
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t228
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t229
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t230
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t231
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t232
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t233
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t234
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t235
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t236
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t237
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t238
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t239
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t240
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t241
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t242
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t243
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t244
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t245
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t246
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t247
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t248
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t249
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t250
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t251
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t252
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t253
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t254
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t255
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t256
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t257
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t258
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t259
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t260
    TABLESPACE roll5
```

```
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t261
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t262
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t263
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t264
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t265
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t266
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t267
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t268
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t269
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t270
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t271
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t272
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t273
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t274
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t275
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t276
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t277
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t278
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t279
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t280
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t281
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t282
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t283
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t284
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t285
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t286
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t287
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t288
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t289
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t290
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t291
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t292
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t293
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t294
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t295
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t296
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t297
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t298
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t299
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t300
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t301
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t302
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t303
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t304
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t305
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t306
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t307
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t308
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t309
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t310
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t311
    TABLESPACE roll1
```

```
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t312
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t313
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t314
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t315
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t316
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t317
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t318
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t319
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t320
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t321
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t322
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t323
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t324
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t325
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t326
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t327
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t328
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t329
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t330
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t331
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t332
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t333
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t334
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t335
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t336
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t337
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t338
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t339
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t340
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t341
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t342
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t343
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t344
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t345
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t346
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t347
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t348
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t349
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t350
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t351
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t352
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t353
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t354
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t355
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t356
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t357
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t358
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t359
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t360
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t361
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t362
    TABLESPACE roll2
```

```
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t363
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t364
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t365
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t366
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t367
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t368
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t369
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t370
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t371
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t372
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t373
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t374
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t375
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t376
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t377
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t378
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t379
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t380
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t381
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t382
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t383
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t384
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t385
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t386
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t387
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t388
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t389
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t390
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t391
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t392
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t393
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t394
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t395
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t396
    TABLESPACE roll1
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t397
    TABLESPACE roll2
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t398
    TABLESPACE roll3
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t399
    TABLESPACE roll4
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);
  CREATE ROLLBACK SEGMENT t400
    TABLESPACE roll5
    STORAGE (initial 100K next 100K minextents
2 maxextents unlimited);

  host date;

  exit;

##########################################
##
####
####  tpcc_tab.sql
####
####

rem
rem
====================================
============================+
rem       Copyright (c) 1996  Oracle Corp,
Redwood Shores, CA     |
rem                OPEN SYSTEMS
PERFORMANCE GROUP              |
rem                    All Rights Reserved
|
rem
====================================
============================+
rem  FILENAME
rem      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_ytd number(12),
w_tax number(4),
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)
)
tablespace ware
initrans 4
pctfree 95 pctused 4
storage (initial 2M next 1M
minextents 10
maxextents unlimited
pctincrease 0);


rem
rem DISTRICT table
rem

create table district (
d_id number,
d_w_id number,
d_ytd number(12),
d_tax number(4),
d_next_o_id number,
d_name varchar2(10),
d_street_1 varchar2(20),
d_street_2 varchar2(20),
d_city varchar2(20),
d_state char(2),
d_zip char(9)
)
tablespace ware
initrans 4
pctfree 95 pctused 4
storage (initial 10M next 2M
minextents 20
maxextents unlimited
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(6),
h_data varchar2(24)
)
tablespace hist
initrans 3
pctfree 1
storage (initial 20K next 101M pctincrease 0
minextents 18
maxextents unlimited
freelist groups 40 freelists 12);


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 90M
minextents 10
maxextents unlimited pctincrease 0
freelist groups 13 freelists 20);


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 10K next 32M pctincrease 0
minextents 10
maxextents unlimited
freelist groups 13 freelists 20);


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_delivery_d date,
ol_i_id number,
ol_supply_w_id number,
ol_quantity number,
ol_amount number(6),
ol_dist_info char(24)
)
tablespace ordl
initrans 4
pctfree 5
storage (initial 20K next 621M pctincrease 0
maxextents unlimited
freelist groups 13 freelists 20);


rem
rem ITEM table
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(5,0),
i_data varchar2(50)
)
cluster icluster(i_id);


rem
rem done
rem

exit;

##########################################
##
####
####  tpcc_tab2.sql
####
####

rem
rem
====================================
==========================+
rem Copyright (c) 1996 Oracle Corp,
Redwood Shores, CA |
rem OPEN SYSTEMS
PERFORMANCE GROUP |
rem All Rights Reserved
|
rem
====================================
==========================+
rem FILENAME
```

```
rem      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      90000000
      hash is       (c_w_id * 30000 + c_d_id *
3000 + c_id)
      size          850
      initrans      3
      pctfree       0
      tablespace    cust
      storage (initial 390M next 390M
                        minextents 80
maxextents unlimited
                        pctincrease 0);

            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(12),
      c_discount    number(4),
      c_balance     number(12),
      c_ytd_payment  number(12),
      c_payment_cnt  number(8),
      c_delivery_cnt  number(8),
      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) 1996  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      300000000
      hash is       (s_i_id * 3000 + s_w_id)
      size          350
      initrans      3
      pctfree       0
      tablespace    stocks
      storage (initial 258M next 258M pctincrease
0
                        minextents 40
maxextents unlimited
                        freelist groups 10
freelists 20);

            create table stock (
      s_i_id        number(6,0),
      s_w_id        number(4,0),
      s_quantity    number(6,0),
      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(10,0),
      s_order_cnt   number(6,0),
      s_remote_cnt  number(6,0),
      s_data        varchar2(50)
      )
      cluster scluster (s_i_id, s_w_id);

rem
rem done
rem

      exit;

##########################################
##
####
####  views.sql
####
####


create or replace view wh_dist
(w_id, d_id, d_tax, d_next_o_id, w_tax )
as select w.w_id, d.d_id, d.d_tax, d.d_next_o_id,
w.w_tax
  from district d, warehouse w
  where w.w_id = d.d_w_id
/

create  or replace view stock_item
(i_id, s_w_id, i_price, i_name, i_data, s_data,
s_quantity,
 s_order_cnt, s_ytd, s_remote_cnt,
 s_dist_01, s_dist_02, s_dist_03, s_dist_04,
s_dist_05,
 s_dist_06, s_dist_07, s_dist_08, s_dist_09,
s_dist_10)
as
 select i.i_id, s_w_id, i.i_price, i.i_name, i.i_data,
s_data, s_quantity,
 s_order_cnt, s_ytd, s_remote_cnt,
 s_dist_01, s_dist_02, s_dist_03, s_dist_04,
s_dist_05,
 s_dist_06, s_dist_07, s_dist_08, s_dist_09,
s_dist_10
  from stock s, item i
 where i.i_id = s.s_i_id
/
quit;

##########################################
##
####
####  new.sql
####
####

  DECLARE   /* new order */
     not_serializable        EXCEPTION;
     PRAGMA EXCEPTION_INIT(not_serializable,-
8177);
     deadlock                EXCEPTION;
     PRAGMA EXCEPTION_INIT(deadlock,-60);
```

```
    snapshot_too_old        EXCEPTION;
    PRAGMA
EXCEPTION_INIT(snapshot_too_old,-1555);
  BEGIN
    LOOP BEGIN
      SELECT c_discount, c_last, c_credit
       INTO :c_discount, :c_last, :c_credit
       FROM customer
       WHERE c_id = :c_id
        AND c_d_id = :d_id
        AND c_w_id = :w_id;

      UPDATE wh_dist SET d_next_o_id =
d_next_o_id + 1, d_tax=d_tax+0
       WHERE  d_id = :d_id
        AND  w_id = :w_id
    RETURNING d_tax, d_next_o_id-1, w_tax
       INTO :d_tax, :o_id, :w_tax;

      INSERT INTO new_order  (no_o_id,
no_d_id, no_w_id)
          VALUES (:o_id , :d_id, :w_id);
      INSERT INTO orders (o_id, o_w_id, o_d_id,
o_c_id, o_carrier_id,
                  o_ol_cnt,
o_all_local,o_entry_d)
          VALUES (:o_id, :w_id, :d_id, :c_id, 11,
                :o_ol_cnt, :o_all_local, :cr_date);
      EXIT;

      EXCEPTION
        WHEN not_serializable OR deadlock OR
snapshot_too_old THEN
          ROLLBACK;
          :retry := :retry + 1;
      END;
    END LOOP;
  END;


##########################################
##
####
####  pay.sql
####
####

CREATE OR REPLACE PACKAGE pay
AS
 TYPE rowidarray IS TABLE OF ROWID INDEX
BY BINARY_INTEGER;
 row_id              rowidarray;
 cust_rowid          ROWID;
 dist_name           VARCHAR2(11);
 ware_name           VARCHAR2(11);
 c_num               BINARY_INTEGER;
 PROCEDURE pay_init;
END pay;
/
CREATE OR REPLACE PACKAGE BODY pay AS
  PROCEDURE pay_init IS
  BEGIN
    NULL;
  END pay_init;
END pay;
/

quit;
```

```
##########################################
##
####
####  paynz.sql
####
####

DECLARE /* paynz */
--    cust_rowid          ROWID;
--    dist_name           VARCHAR2(11);
--    ware_name           VARCHAR2(11);
    not_serializable      EXCEPTION;
    PRAGMA EXCEPTION_INIT(not_serializable,-
8177);
    deadlock              EXCEPTION;
    PRAGMA EXCEPTION_INIT(deadlock,-60);
    snapshot_too_old      EXCEPTION;
    PRAGMA
EXCEPTION_INIT(snapshot_too_old,-1555);
  BEGIN
    LOOP BEGIN
        UPDATE  customer
          SET  c_balance = c_balance -
:h_amount,
               c_ytd_payment = c_ytd_payment +
:h_amount,
               c_payment_cnt = c_payment_cnt+1
        WHERE    c_id = :c_id AND c_d_id =
:c_d_id AND
               c_w_id = :c_w_id
      RETURNING rowid, c_first, c_middle,
c_last, c_street_1,
               c_street_2, c_city, c_state, c_zip,
c_phone,
               c_since, c_credit, c_credit_lim,
               c_discount, c_balance
          INTO pay.cust_rowid,:c_first, :c_middle,
:c_last, :c_street_1,
               :c_street_2, :c_city, :c_state, :c_zip,
:c_phone,
               :c_since, :c_credit, :c_credit_lim,
               :c_discount, :c_balance;
        IF SQL%NOTFOUND THEN
          raise NO_DATA_FOUND;
        END IF;

--        :c_data := ' ';

       IF :c_credit = 'BC' THEN
         UPDATE customer
                SET c_data= substr ((to_char
(:c_id) || ' ' ||
                           to_char (:c_d_id) || ' ' ||
                           to_char (:c_w_id) || ' ' ||
                           to_char (:d_id) || ' ' ||
                           to_char (:w_id) || ' ' ||
                           to_char (:h_amount,
'9999.99') || ' | ')
                           || c_data, 1, 500)
          WHERE rowid = pay.cust_rowid
       RETURNING substr(c_data,1, 200)
          INTO :c_data;

       END IF;

         UPDATE district
           SET d_ytd = d_ytd + :h_amount
          WHERE d_id = :d_id
            AND d_w_id = :w_id
       RETURNING d_name, d_street_1,
d_street_2, d_city,d_state, d_zip
           INTO
pay.dist_name,:d_street_1,:d_street_2,:d_city,:d_
state,
                :d_zip;
        IF SQL%NOTFOUND THEN
         raise NO_DATA_FOUND;
        END IF;


         UPDATE warehouse
           SET w_ytd = w_ytd + :h_amount
          WHERE w_id = :w_id
       RETURNING w_name, w_street_1,
w_street_2, w_city, w_state, w_zip
           INTO pay.ware_name, :w_street_1,
:w_street_2, :w_city, :w_state,
                :w_zip;

       INSERT INTO history  (h_c_id, h_c_d_id,
h_c_w_id, h_d_id, h_w_id,
                    h_amount, h_date, h_data)
       VALUES
         (:c_id, :c_d_id, :c_w_id, :d_id, :w_id,
:h_amount,
          :cr_date, pay.ware_name || '    ' ||
pay.dist_name);
       COMMIT;
       :h_date := to_char (:cr_date, 'DD-MM-
YYYY.HH24:MI:SS');
       EXIT;

       EXCEPTION
         WHEN not_serializable OR deadlock OR
snapshot_too_old THEN
           ROLLBACK;
           :retry := :retry + 1;
       END;

    END LOOP;
  END;

##########################################
##
####
####  payz.sql
####
####

DECLARE /* payz */
--    TYPE rowidarray IS TABLE OF ROWID
INDEX BY BINARY_INTEGER;
--    cust_rowid          ROWID;
--    dist_name           VARCHAR2(11);
--    ware_name           VARCHAR2(11);
--    c_num               BINARY_INTEGER;
--    row_id              rowidarray;
    not_serializable      EXCEPTION;
    PRAGMA EXCEPTION_INIT(not_serializable,-
8177);
    deadlock              EXCEPTION;
    PRAGMA EXCEPTION_INIT(deadlock,-60);
    snapshot_too_old      EXCEPTION;
    PRAGMA
EXCEPTION_INIT(snapshot_too_old,-1555);
```

```
    CURSOR c_cur IS
      SELECT rowid
      FROM customer
      WHERE c_d_id = :c_d_id AND c_w_id =
:c_w_id AND c_last = :c_last
      ORDER BY c_w_id, c_d_id, c_last, c_first;
  BEGIN
    LOOP BEGIN

      pay.c_num := 0;
      FOR c_id_rec IN c_cur LOOP
        pay.c_num := pay.c_num + 1;
        pay.row_id(pay.c_num) := c_id_rec.rowid;
      END LOOP;
      pay.cust_rowid := pay.row_id ((pay.c_num +
1) / 2);
      UPDATE customer
       SET c_balance = c_balance - :h_amount,
         c_ytd_payment = c_ytd_payment+
:h_amount,
          c_payment_cnt = c_payment_cnt+1
      WHERE rowid = pay.cust_rowid
      RETURNING
           c_id, c_first, c_middle, c_last,
c_street_1, c_street_2,
           c_city, c_state, c_zip, c_phone,
           c_since, c_credit, c_credit_lim,
           c_discount, c_balance
        INTO :c_id, :c_first, :c_middle, :c_last,
          :c_street_1, :c_street_2, :c_city,
:c_state,
           :c_zip, :c_phone, :c_since, :c_credit,
           :c_credit_lim, :c_discount, :c_balance;

--       :c_data := ' ';
      IF :c_credit = 'BC' THEN
        UPDATE customer
         SET c_data = substr ((to_char (:c_id) || ' '
||
                          to_char (:c_d_id) || ' ' ||
                          to_char (:c_w_id) || ' ' ||
                          to_char (:d_id) || ' ' ||
                          to_char (:w_id) || ' ' ||
                          to_char (:h_amount/100,
'9999.99') || ' | ')
                          || c_data, 1, 500)
          WHERE rowid = pay.cust_rowid
          RETURNING substr(c_data,1, 200)
          INTO :c_data;

      END IF;

      UPDATE district
        SET d_ytd = d_ytd + :h_amount
        WHERE d_id = :d_id
          AND d_w_id = :w_id
        RETURNING  d_name, d_street_1,
d_street_2, d_city,
                   d_state, d_zip
        INTO pay.dist_name, :d_street_1,
:d_street_2, :d_city,
           :d_state, :d_zip;
       IF SQL%NOTFOUND THEN
        raise NO_DATA_FOUND;
       END IF;


      UPDATE warehouse
         SET w_ytd = w_ytd+:h_amount
         WHERE w_id = :w_id
         RETURNING w_name,
              w_street_1, w_street_2, w_city,
w_state, w_zip
          INTO pay.ware_name,
             :w_street_1, :w_street_2, :w_city,
:w_state, :w_zip;

       INSERT INTO history (h_c_id, h_c_d_id,
h_c_w_id, h_d_id, h_w_id,
                    h_amount, h_date, h_data)
        VALUES (:c_id, :c_d_id, :c_w_id, :d_id,
:w_id, :h_amount,
         :cr_date, pay.ware_name || '    ' ||
pay.dist_name);
       COMMIT;
       :h_date := to_char (:cr_date, 'DD-MM-
YYYY.HH24:MI:SS');
       EXIT;

       EXCEPTION
         WHEN not_serializable OR deadlock OR
snapshot_too_old THEN
          ROLLBACK;
          :retry := :retry + 1;
       END;

    END LOOP;
  END;

##########################################
##
####
####  CUSTOMER
####
####

hda0011
hda0012
hda0013
hda0014
hda0015
hda0111
hda0112
hda0113
hda0114
hda0211
hda0212
hda0213
hda0214
hda0311
hda0312
hda0313
hda0314
hda1011
hda1012
hda1013
hda1014
hda1015
hda1111
hda1112
hda1113
hda1114
hda1211
hda1212
hda1213
hda1214
hda1311
hda1312
hda1313
hda1314
hda2011
hda2012
hda2013
hda2014
hda2015
hda2111
hda2112
hda2113
hda2114
hda2211
hda2212
hda2213
hda2214
hda2311
hda2312
hda2313
hda2314
hda3011
hda3012
hda3013
hda3014
hda3015
hda3111
hda3112
hda3113
hda3114
hda3211
hda3212
hda3213
hda3214
hda3311
hda3312
hda3313
hda3314
hda4011
hda4012
hda4013
hda4014
hda4015
hda4111
hda4112
hda4113
hda4114
hda4211
hda4212
hda4213
hda4214
hda4311
hda4312
hda4313
hda4314
hda5011
hda5012
hda5013
hda5014
hda5015
hda5111
hda5112
hda5113
hda5114
hda5211
hda5212
hda5213
hda5214
```

hda5311
hda5312
hda5313
hda5314
hda6011
hda6012
hda6013
hda6014
hda6015
hda6111
hda6112
hda6113
hda6114
hda6211
hda6212
hda6213
hda6214
hda6311
hda6312
hda6313
hda6314
hda7011
hda7012
hda7013
hda7014
hda7015
hda7111
hda7112
hda7113
hda7114
hda7211
hda7212
hda7213
hda7214
hda7311
hda7312
hda7313
hda7314
hda8011
hda8012
hda8013
hda8014
hda8111
hda8112
hda8113
hda8114
hda8211
hda8212
hda8213
hda8214
hda8311
hda8312
hda8313
hda8314
hda9011
hda9012
hda9013
hda9014
hda9111
hda9112
hda9113
hda9114
hda9211
hda9212
hda9213
hda9214
hda9311
hda9312
hda9313
hda9314
hdb6006
hdb6007
hdb6106
hdb6107
hdb6206
hdb6207
hdb6306
hdb6307
hdb7006
hdb7007
hdb7106
hdb7107
hdb7206
hdb7207
hdb7306
hdb7307
hdb8006
hdb8007
hdb8106
hdb8107
hdb8206
hdb8207
hdb8306
hdb8307
hdb9006
hdb9007
hdb9106
hdb9107
hdb9206
hdb9207
hdb9306
hdb9307
hdc0006
hdc0007
hdc0106
hdc0107
hdc0206
hdc0207
hdc0306
hdc0307
hdc1006
hdc1007
hdc1106
hdc1107
hdc1206
hdc1207
hdc1306
hdc1307
hdc2006
hdc2007
hdc2106
hdc2107
hdc2206
hdc2207
hdc2306
hdc2307
hdc3006
hdc3007
hdc3106
hdc3107
hdc3206
hdc3207
hdc3306
hdc3307
hdc4006
hdc4007
hdc4106
hdc4107
hdc4206
hdc4207
hdc4306
hdc4307
hdc5006
hdc5007
hdc5106
hdc5107
hdc5206
hdc5207
hdc5306
hdc5307
hdc6006
hdc6007
hdc6106
hdc6107
hdc6206
hdc6207
hdc6306
hdc6307
hdc7006
hdc7007
hdc7106
hdc7107
hdc7206
hdc7207
hdc7306
hdc7307
hdc8006
hdc8007
hdc8106
hdc8107
hdc8206
hdc8207
hdc8306
hdc8307
hdc9006
hdc9007
hdc9106
hdc9107
hdc9206
hdc9207
hdc9306
hdc9307

##
##
####
HISTORY
####
####

hda1108
hda1308
hda2108
hda2308
hda3108
hda3308
hda4108
hda4308
hda5108
hda5308
hda6108
hda6308

```
########################################
##
####
####  ICUSTOMER1
####
####

hda0209
hda1209
hda2209
hda3209
hda4209
hda5209
hda6209
hda7209
hda8209
hda9209

########################################
##
####
####  ICUSTOMER2
####
####

hda0210
hda0310
hda1210
hda1310
hda2210
hda2310
hda3210
hda3310
hda4210
hda4310
hda5210
hda5310
hda6210
hda6310
hda7009
hda7210
hda7308
hda7310
hda8009
hda8010
hda8108
hda8210
hda8308
hda8310
hda9009
hda9010
hda9108
hda9210
hda9308
hda9310

########################################
##
####
####  INEW-ORDER
####
####

hda0110
hda1110
hda2110
hda3110
hda4110
hda5110
hda6110
hda7110
hda8110
hda9110

########################################
##
####
####  IORDER1
####
####

hda0115
hda0315
hda1115
hda1315
hda2115
hda2315
hda3115
hda3315
hda4115
hda4315
hda5115
hda5315
hda6115
hda6315
hda7115
hda7315
hda8115
hda8315
hda9115
hda9315

########################################
##
####
####  IORDER2
####
####

hda0109
hda0309
hda1109
hda1309
hda2109
hda2309
hda3109
hda3309
hda4109
hda4309
hda5109
hda5309
hda6109
hda6309
hda7108
hda7109
hda7309
hda8109
hda8309
hda9109
hda9309
hdb6005
hdb7005
hdb8005
hdb9005
hdc0005
hdc1005
hdc2005
hdc3005
hdc4005
hdc5005
hdc6005
hdc7005
hdc8005
hdc9005

########################################
##
####
####  IORD_LINE
####
####

hdb4001
hdb4002
hdb4003
hdb4004
hdb4005
hdb4101
hdb4102
hdb4103
hdb4104
hdb4105
hdb4201
hdb4202
hdb4203
hdb4204
hdb4205
hdb4301
hdb4302
hdb4303
hdb4304
hdb4305
hdb5001
hdb5002
hdb5003
hdb5004
hdb5005
hdb5101
hdb5102
hdb5103
hdb5104
hdb5105
hdb5201
hdb5202
hdb5203
hdb5204
hdb5205
hdb5301
hdb5302
hdb5303

########################################
##
####
####  ISTOCK
####
####

hda1009
hda2009
hda3009
hda4009
hda5009
hda6009
```

```
########################################
##
####
####  ITEM
####
####

hda0308

########################################
##
####
####  LOG
####
####

hdc9105
hdc9205

########################################
##
####
####  NEW-ORDER
####
####

hdc0104
hdc1104
hdc1304
hdc2104
hdc2304
hdc3104
hdc4104
hdc5104
hdc6104
hdc7104
hdc8104
hdc9104

########################################
##
####
####  ORDERS
####
####

hdb6305
hdb7305
hdb8305
hdb9305
hdc0105
hdc0305
hdc1105
hdc1305
hdc2105
hdc2305
hdc3105
hdc3304
hdc3305
hdc4304
hdc4305
hdc5304
hdc5305
hdc6304
hdc6305
hdc7304
hdc7305
hdc8304
hdc8305
hdc9304
hdc9305

########################################
##
####
####  ORDER_LINE
####
####

hdb0001
hdb0002
hdb0003
hdb0004
hdb0005
hdb0101
hdb0102
hdb0103
hdb0104
hdb0105
hdb0201
hdb0202
hdb0203
hdb0204
hdb0205
hdb0301
hdb0302
hdb0303
hdb0304
hdb0305
hdb1001
hdb1002
hdb1003
hdb1004
hdb1005
hdb1101
hdb1102
hdb1103
hdb1104
hdb1105
hdb1201
hdb1202
hdb1203
hdb1204
hdb1205
hdb1301
hdb1302
hdb1303
hdb1304
hdb1305
hdb2001
hdb2002
hdb2003
hdb2004
hdb2005
hdb2101
hdb2102
hdb2103
hdb2104
hdb2105
hdb2201
hdb2202
hdb2203
hdb2204
hdb2205
hdb2301
hdb2302
hdb2303
hdb2304
hdb2305
hdb3001
hdb3002
hdb3003
hdb3004
hdb3005
hdb3101
hdb3102
hdb3103
hdb3104
hdb3105
hdb3201
hdb3202
hdb3203
hdb3204
hdb3205

########################################
##
####
####  ROLL
####
####

hdb6304
hdb7304
hdb8304
hdb9304
hdc0304

########################################
##
####
####  STOCK
####
####

hda0001
hda0002
hda0003
hda0004
hda0005
hda0006
hda0007
hda0008
hda0101
hda0102
hda0103
hda0104
hda0105
hda0106
hda0107
hda0201
hda0202
hda0203
hda0204
hda0205
hda0206
hda0207
hda0208
hda0301
hda0302
hda0303
hda0304
hda0305
```

hda0306
hda0307
hda1001
hda1002
hda1003
hda1004
hda1005
hda1006
hda1007
hda1008
hda1101
hda1102
hda1103
hda1104
hda1105
hda1106
hda1107
hda1201
hda1202
hda1203
hda1204
hda1205
hda1206
hda1207
hda1208
hda1301
hda1302
hda1303
hda1304
hda1305
hda1306
hda1307
hda2001
hda2002
hda2003
hda2004
hda2005
hda2006
hda2007
hda2008
hda2101
hda2102
hda2103
hda2104
hda2105
hda2106
hda2107
hda2201
hda2202
hda2203
hda2204
hda2205
hda2206
hda2207
hda2208
hda2301
hda2302
hda2303
hda2304
hda2305
hda2306
hda2307
hda3001
hda3002
hda3003
hda3004
hda3005
hda3006
hda3007
hda3008
hda3101
hda3102
hda3103
hda3104
hda3105
hda3106
hda3107
hda3201
hda3202
hda3203
hda3204
hda3205
hda3206
hda3207
hda3208
hda3301
hda3302
hda3303
hda3304
hda3305
hda3306
hda3307
hda4001
hda4002
hda4003
hda4004
hda4005
hda4006
hda4007
hda4008
hda4101
hda4102
hda4103
hda4104
hda4105
hda4106
hda4107
hda4201
hda4202
hda4203
hda4204
hda4205
hda4206
hda4207
hda4208
hda4301
hda4302
hda4303
hda4304
hda4305
hda4306
hda4307
hda5001
hda5002
hda5003
hda5004
hda5005
hda5006
hda5007
hda5008
hda5101
hda5102
hda5103
hda5104
hda5105
hda5106
hda5107
hda5201
hda5202
hda5203
hda5204
hda5205
hda5206
hda5207
hda5208
hda5301
hda5302
hda5303
hda5304
hda5305
hda5306
hda5307
hda6001
hda6002
hda6003
hda6004
hda6005
hda6006
hda6007
hda6008
hda6101
hda6102
hda6103
hda6104
hda6105
hda6106
hda6107
hda6201
hda6202
hda6203
hda6204
hda6205
hda6206
hda6207
hda6208
hda6301
hda6302
hda6303
hda6304
hda6305
hda6306
hda6307
hda7001
hda7002
hda7003
hda7004
hda7005
hda7006
hda7007
hda7008
hda7101
hda7102
hda7103
hda7104
hda7105
hda7106
hda7107
hda7201
hda7202
hda7203
hda7204
hda7205
hda7206
hda7207

hda7208
hda7301
hda7302
hda7303
hda7304
hda7305
hda7306
hda7307
hda8001
hda8002
hda8003
hda8004
hda8005
hda8006
hda8007
hda8008
hda8101
hda8102
hda8103
hda8104
hda8105
hda8106
hda8107
hda8201
hda8202
hda8203
hda8204
hda8205
hda8206
hda8207
hda8208
hda8301
hda8302
hda8303
hda8304
hda8305
hda8306
hda8307
hda9001
hda9002
hda9003
hda9004
hda9005
hda9006
hda9007
hda9008
hda9101
hda9102
hda9103
hda9104
hda9105
hda9106
hda9107
hda9201
hda9202
hda9203
hda9204
hda9206
hda9207
hda9208
hda9301
hda9302
hda9303
hda9304
hda9305
hda9306
hda9307
hdb6001
hdb6002
hdb6003
hdb6004
hdb6101
hdb6102
hdb6103
hdb6104
hdb6201
hdb6202
hdb6203
hdb6204
hdb6301
hdb6302
hdb6303
hdb7001
hdb7002
hdb7003
hdb7004
hdb7101
hdb7102
hdb7103
hdb7104
hdb7201
hdb7202
hdb7203
hdb7204
hdb7301
hdb7302
hdb7303
hdb8001
hdb8002
hdb8003
hdb8004
hdb8101
hdb8102
hdb8103
hdb8104
hdb8201
hdb8202
hdb8203
hdb8204
hdb8301
hdb8302
hdb8303
hdb9001
hdb9002
hdb9003
hdb9004
hdb9101
hdb9102
hdb9103
hdb9104
hdb9201
hdb9202
hdb9203
hdb9204
hdb9301
hdb9302
hdb9303
hdc0001
hdc0002
hdc0003
hdc0004
hdc0101
hdc0102
hdc0103
hdc0201
hdc0202
hdc0203
hdc0204
hdc0301
hdc0302
hdc0303
hdc1001
hdc1002
hdc1003
hdc1004
hdc1101
hdc1102
hdc1103
hdc1201
hdc1202
hdc1203
hdc1204
hdc1301
hdc1302
hdc1303
hdc2001
hdc2002
hdc2003
hdc2004
hdc2101
hdc2102
hdc2103
hdc2201
hdc2202
hdc2203
hdc2204
hdc2301
hdc2302
hdc2303
hdc3001
hdc3002
hdc3003
hdc3004
hdc3101
hdc3102
hdc3103
hdc3201
hdc3202
hdc3203
hdc3204
hdc3301
hdc3302
hdc3303
hdc4001
hdc4002
hdc4003
hdc4004
hdc4101
hdc4102
hdc4103
hdc4201
hdc4202
hdc4203
hdc4204
hdc4301
hdc4302
hdc4303
hdc5001
hdc5002
hdc5003
hdc5004
hdc5101
hdc5102
hdc5103

```
hdc5201
hdc5202
hdc5203
hdc5204
hdc5301
hdc5302
hdc5303
hdc6001
hdc6002
hdc6003
hdc6004
hdc6101
hdc6102
hdc6103
hdc6201
hdc6202
hdc6203
hdc6204
hdc6301
hdc6302
hdc6303
hdc7001
hdc7002
hdc7003
hdc7004
hdc7101
hdc7102
hdc7103
hdc7201
hdc7202
hdc7203
hdc7204
hdc7301
hdc7302
hdc7303
hdc8001
hdc8002
hdc8003
hdc8004
hdc8101
hdc8102
hdc8103
hdc8201
hdc8202
hdc8203
hdc8204
hdc8301
hdc8302
hdc8303
hdc9001
hdc9002
hdc9003
hdc9004
hdc9101
hdc9102
hdc9103
hdc9201
hdc9202
hdc9203
hdc9204
hdc9301
hdc9302
hdc9303

########################################
##
####
####  SYSTEM
####
####

hda0009

########################################
##
####
####  TEMP
####
####

hda1010
hda2010
hda3010
hda4010
hda5010
hda6010
hda7010
hdb3301
hdb3302
hdb3303
hdb3304
hdb3305
hdb5304
hdb5305
hdb6105
hdb6205
hdb7105
hdb7205
hdb8105
hdb8205
hdb9105
hdb9205
hdc0205
hdc1205
hdc2205
hdc3205
hdc4105
hdc4205
hdc5105
hdc5205
hdc6105
hdc6205
hdc7105
hdc7205
hdc8105
hdc8205

########################################
##
####
####  WAREHOUSE
####
####

hda0108
```

Appendix F: Distribution of Tables and Logs

SCSI	Device		Table Name	Size (Mbytes)	Disk
WSA-0	/dev/rdsk/hda00	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		15	CUSTOMER	400	
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	SYSTEM	1510	4.0GB
		10	ORACLE_HOME	2048	
		18	STOCK <= hdb8302	265	
	/dev/rdsk/hda01	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	WAREHOUSE	330	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	NEW_ORDER_IX	315	
WSA-1	/dev/rdsk/hda02	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		08	STOCK	265	
		09	CUSTOMER_IX1	610	4.0GB
		10	CUSTOMER_IX2	260	
		15	STOCK <= hda7103	265	
	/dev/rdsk/hda03	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	ITEM	30	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2	260	
WSA-10	/dev/rdsk/hda10	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		15	CUSTOMER	400	
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	STOCK_IX	1705	4.0GB
		10	TEMP	1810	
	/dev/rdsk/hda11	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK => hdi7302	265	
		14	CUSTOMER	400	
		05	STOCK => hdi4301	265	4.0GB
		06	STOCK => hdi2101	265	
		07	STOCK	265	4.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	NEW_ORDER_IX	315	
WSA-11	/dev/rdsk/hda12	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	CUSTOMER_IX1	610	4.0GB
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda13	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2	260	
WSA-20	/dev/rdsk/hda20	01	STOCK => hdi5202	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		15	CUSTOMER	400	
		06	STOCK	265	
		07	STOCK	265	4.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		08	STOCK	265	
		09	STOCK_IX	1705	4.0GB
		10	TEMP	1810	
	/dev/rdsk/hda21	01	STOCK => hdi0205	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdc4211	265	
		12	CUSTOMER	400	
		03	STOCK => hdi6302	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK => hdi0105	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	NEW_ORDER_IX	315	
WSA-21	/dev/rdsk/hda22	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK => hdi1305	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK => hdb4612	265	
		07	STOCK => hdi2202	265	4.0GB
		08	STOCK => hdi7202	265	
		09	CUSTOMER_IX1	610	4.0GB
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda23	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdi0005	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK => hdi5002	265	
		14	CUSTOMER	400	
		05	STOCK => hdi0204	265	4.0GB
		06	STOCK => hdi1104	265	
		07	STOCK => hdi6301	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2 => hdc4309	260	
WSA-30	/dev/rdsk/hda30	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		15	CUSTOMER	400	
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	STOCK_IX	1705	4.0GB
		10	TEMP	1810	
	/dev/rdsk/hda31	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdi4201	265	
		12	CUSTOMER	400	
		03	STOCK => hdi2301	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	NEW_ORDER_IX	315	
WSA-31	/dev/rdsk/hda32	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	CUSTOMER_IX1	610	4.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda33	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK => hdi5003	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK => hdi4302	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2	260	
WSA-40	/dev/rdsk/hda40	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		15	CUSTOMER	400	
		06	STOCK =>hdb4712	265	
		07	STOCK => hdi4102	265	4.0GB
		08	STOCK	265	
		09	STOCK_IX	1705	4.0GB
		10	TEMP	1810	
	/dev/rdsk/hda41	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdi1203	265	
		12	CUSTOMER	400	
		03	STOCK => hdi0006	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK => hdi0305	265	
		14	CUSTOMER	400	
		05	STOCK => hdi6001	265	4.0GB
		06	STOCK => hdi5101	265	
		07	STOCK => hdi2001	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	

SCSI	Device		Table Name	Size (Mbytes)	Disk
		10	NEW_ORDER_IX	315	
WSA-41	/dev/rdsk/hda42	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdi7201	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	CUSTOMER_IX1	610	4.0GB
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda43	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdi1205	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK => hdi0104	265	4.0GB
		06	STOCK => hdi7002	265	
		07	STOCK	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2	260	
WSA-50	/dev/rdsk/hda50	01	STOCK => hdc8109	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		15	CUSTOMER	400	
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK => hdi4103	265	
		09	STOCK_IX	1705	4.0GB
		10	TEMP	1810	

SCSI	Device		Table Name	Size (Mbytes)	Disk
	/dev/rdsk/hda51	01	STOCK =>hdb7607	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdi5301	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK => hdi1003	265	4.0GB
		06	STOCK => hdi6202	265	
		07	STOCK => hdi3003	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	NEW_ORDER_IX	315	
WSA-51	/dev/rdsk/hda52	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK => hdi5201	265	
		14	CUSTOMER	400	
		05	STOCK => hdi7102	265	4.0GB
		06	STOCK => hdi3101	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	CUSTOMER_IX1	610	4.0GB
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda53	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2 => hdi4303	260	
WSA-2	/dev/rdsk/hda60	01	STOCK => hdc8208	265	4.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		15	CUSTOMER	400	
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK => hdi4203	265	
		09	STOCK_IX	1705	4.0GB
		10	TEMP	1810	
	/dev/rdsk/hda61	01	STOCK => hdi1004	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdi6101	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	NEW_ORDER_IX	315	
WSA-3	/dev/rdsk/hda62	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK => hdi5302	265	
		09	CUSTOMER_IX1	610	4.0GB
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda63	01	STOCK	265	4.0GB
		11	CUSTOMER	400	

SCSI	Device		Table Name	Size (Mbytes)	Disk
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK => hdc8110	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK => hdi1204	265	
		14	CUSTOMER	400	
		05	STOCK => hdi6201	265	4.0GB
		06	STOCK => hdi0206	265	
		07	STOCK => hdi6102	265	4.0GB
		08	HISTORY	920	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2 => hdc5209	260	
WSA-12	/dev/rdsk/hda70	01	STOCK =>hdic3312	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdi3201	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		15	CUSTOMER	400	
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	CUSTOMER_IX2	260	4.0GB
		10	TEMP	1810	
	/dev/rdsk/hda71	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK => hda0215	265	4.0GB
		13	CUSTOMER => hdc9010	400	
		04	STOCK => hdi5103	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK => hdi4002	265	
		07	STOCK => hdi4003	265	4.0GB
		08	ORDER_IX2	160	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	NEW_ORDER_IX	315	
WSA-13	/dev/rdsk/hda72	01	STOCK	265	4.0GB
		11	CUSTOMER	400	

SCSI	Device		Table Name	Size (Mbytes)	Disk
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK =>hdi7001	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	CUSTOMER_IX1	610	4.0GB
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda73	01	STOCK => hdi1304	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK => hdb7706	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK => hdb6208	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK => hdc9110	265	
		07	STOCK => hdi7101	265	4.0GB
		08	CUSTOMER_IX2	260	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2	260	
WSA-22	/dev/rdsk/hda80	01	STOCK => hdi4001	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK => hdi5001	265	4.0GB
		06	STOCK => hdi3001	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	CUSTOMER_IX2	260	4.0GB
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda81	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	

SCSI	Device		Table Name	Size (Mbytes)	Disk
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	ORDER_IX2	160	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	NEW_ORDER_IX	315	
WSA-23	/dev/rdsk/hda82	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK => hdc7208	265	
		07	STOCK	265	4.0GB
		08	STOCK	265	
		09	CUSTOMER_IX1	610	4.0GB
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda83	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK => hdi1005	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK => hdb7502	265	
		07	STOCK => hdi4202	265	4.0GB
		08	CUSTOMER_IX2	260	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2	260	
WSA-32	/dev/rdsk/hda90	01	STOCK	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK => hdi3301	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		13	CUSTOMER	400	
		04	STOCK => hdi5102	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK => hdi0004	265	4.0GB
		08	STOCK => hdi4101	265	
		09	CUSTOMER_IX2	260	4.0GB
		10	CUSTOMER_IX2 => hdc3311	260	
	/dev/rdsk/hda91	01	STOCK => hdi2201	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK =>hdb4512	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	CUSTOMER_IX2	260	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	NEW_ORDER_IX	315	
WSA-33	/dev/rdsk/hda92	01	STOCK => hdb7411	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK => hdi6002	265	
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK => hdi2102	265	4.0GB
		08	STOCK => hdi0304	265	
		09	CUSTOMER_IX1	610	4.0GB
		10	CUSTOMER_IX2	260	
	/dev/rdsk/hda93	01	STOCK => hdi7301	265	4.0GB
		11	CUSTOMER	400	
		02	STOCK	265	
		12	CUSTOMER	400	
		03	STOCK	265	4.0GB
		13	CUSTOMER	400	
		04	STOCK	265	

SCSI	Device		Table Name	Size (Mbytes)	Disk
		14	CUSTOMER	400	
		05	STOCK	265	4.0GB
		06	STOCK	265	
		07	STOCK	265	4.0GB
		08	CUSTOMER_IX2	260	
		09	ORDER_IX2	160	4.0GB
		15	ORDER_IX1	350	
		10	CUSTOMER_IX2	260	
WSA-62	/dev/rdsk/hdb00	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb01	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb02	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb03	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
WSA-63	/dev/rdsk/hdb10	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb11	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb12	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
	/dev/rdsk/hdb13	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
WSA-64	/dev/rdsk/hdb20	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb21	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb22	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb23	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
WSA-65	/dev/rdsk/hdb30	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb31	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb32	01	ORDER-LINE	1875	2.0GB
		02	ORDER-LINE	1875	2.0GB
		03	ORDER-LINE	1875	2.0GB
		04	ORDER-LINE	1875	2.0GB
		05	ORDER-LINE	1875	2.0GB
	/dev/rdsk/hdb33	01	TEMP	1809	2.0GB
		02	TEMP	1809	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		03	TEMP	1809	2.0GB
		04	TEMP	1809	2.0GB
		05	TEMP	1809	2.0GB
WSA-66	/dev/rdsk/hdb40	01	ORDER-LINE INDEX	1895	2.0GB
		02	ORDER-LINE INDEX	1895	2.0GB
		03	ORDER-LINE INDEX	1895	2.0GB
		04	ORDER-LINE INDEX	1895	2.0GB
		05	ORDER-LINE INDEX	1895	2.0GB
	/dev/rdsk/hdb41	01	ORDER-LINE INDEX	1895	2.0GB
		02	ORDER-LINE INDEX	1895	2.0GB
		03	ORDER-LINE INDEX	1895	2.0GB
		04	ORDER-LINE INDEX	1895	2.0GB
		05	ORDER-LINE INDEX	1895	2.0GB
	/dev/rdsk/hdb42	01	ORDER-LINE INDEX	1895	2.0GB
		02	ORDER-LINE INDEX	1895	2.0GB
		03	ORDER-LINE INDEX	1895	2.0GB
		04	ORDER-LINE INDEX	1895	2.0GB
		05	ORDER-LINE INDEX	1895	2.0GB
	/dev/rdsk/hdb43	01	ORDER-LINE INDEX	1895	2.0GB
		02	ORDER-LINE INDEX	1895	2.0GB
		03	ORDER-LINE INDEX	1895	2.0GB
		04	ORDER-LINE INDEX	1895	2.0GB
		05	ORDER-LINE INDEX	1895	2.0GB
WSA-56	/dev/rdsk/hdb44	12	STOCK <= hdc7103	265	2.0GB
	/dev/rdsk/hdb45	12	STOCK <= hda9104	265	2.0GB
	/dev/rdsk/hdb46	12	STOCK <= hda2206	265	2.0GB
	/dev/rdsk/hdb47	12	STOCK <= hda4006	265	2.0GB
WSA-67	/dev/rdsk/hdb50	01	ORDER-LINE INDEX	1895	2.0GB
		02	ORDER-LINE INDEX	1895	2.0GB
		03	ORDER-LINE INDEX	1895	2.0GB
		04	ORDER-LINE INDEX	1895	2.0GB
		05	ORDER-LINE INDEX	1895	2.0GB
	/dev/rdsk/hdb51	01	ORDER-LINE INDEX	1895	2.0GB
		02	ORDER-LINE INDEX	1895	2.0GB
		03	ORDER-LINE INDEX	1895	2.0GB
		04	ORDER-LINE INDEX	1895	2.0GB
		05	ORDER-LINE INDEX	1895	2.0GB
	/dev/rdsk/hdb52	01	ORDER-LINE INDEX	1895	2.0GB
		02	ORDER-LINE INDEX	1895	2.0GB
		03	ORDER-LINE INDEX	1895	2.0GB
		04	ORDER-LINE INDEX	1895	2.0GB
		05	ORDER-LINE INDEX	1895	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
	/dev/rdsk/hdb53	01	ORDER-LINE INDEX	1895	2.0GB
		02	ORDER-LINE INDEX	1895	2.0GB
		03	ORDER-LINE INDEX	1895	2.0GB
		04	TEMP	1895	2.0GB
		05	TEMP	1895	2.0GB
WSA-42	/dev/rdsk/hdb60	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdb61	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdb62	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03 08	STOCK STOCK <= hda7304	265 265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdb63	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	ROLLBACK	310	2.0GB
		05	ORDERS	370	2.0GB
WSA-52	/dev/rdsk/hdb70	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdb71	01 06	STOCK CUSTOMER	265 400	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdb72	01 06	STOCK CUSTOMER	265 400	2.0GB
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdb73	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	ROLLBACK	310	2.0GB
		05	ORDERS	370	2.0GB
WSA-57	/dev/rdsk/hdb74	11	STOCK <= hda9201	265	2.0GB
	/dev/rdsk/hdb75	02	STOCK <= hda8306	265	2.0GB
	/dev/rdsk/hdb76	07	STOCK <= hda5101	265	2.0GB
	/dev/rdsk/hdb77	06	STOCK <= hda7303	265	2.0GB
WSA-43	/dev/rdsk/hdb80	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdb81	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdb82	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdb83	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK => hda0018	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	ROLLBACK	310	2.0GB
		05	ORDERS	370	2.0GB
WSA-53	/dev/rdsk/hdb90	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdb91	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdb92	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdb93	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK => hdc1309	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	ROLLBACK	310	2.0GB
		05	ORDERS	370	2.0GB
WSA-4	/dev/rdsk/hdc00	01	STOCK => hdi3302	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
	/dev/rdsk/hdc01	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	NEW-ORDER	315	2.0GB
		05	ORDERS	370	2.0GB
	/dev/rdsk/hdc02	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdc03	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	ROLLBACK	310	2.0GB
		05	ORDERS	370	2.0GB
WSA-14	/dev/rdsk/hdc10	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdc11	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	NEW-ORDER	315	2.0GB
		05	ORDERS	370	2.0GB
	/dev/rdsk/hdc12	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdc13	01	STOCK	265	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		09	STOCK <= hdb9302	265	
		04	NEW-ORDER	265	2.0GB
		05	ORDERS	370	2.0GB
WSA-24	/dev/rdsk/hdc20	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdc21	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	NEW-ORDER	315	2.0GB
		05	ORDERS	370	2.0GB
	/dev/rdsk/hdc22	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdc23	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	NEW-ORDER	265	2.0GB
		05	ORDERS	370	2.0GB
WSA-34	/dev/rdsk/hdc30	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK => hdc8308	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK => hdc5311	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdc31	01	STOCK	265	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	NEW-ORDER	315	2.0GB
		05	ORDERS	370	2.0GB
	/dev/rdsk/hdc32	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		10	STOCK	265	
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdc33	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		09	STOCK	265	
		04	ORDERS	370	2.0GB
		11	CUSTOMER_IX2 <= hda9010	260	
		05	ORDERS	370	2.0GB
		12	CUSTOMER <= hda7001	265	
WSA-44	/dev/rdsk/hdc40	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		09	STOCK	265	
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdc41	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		09	STOCK	265	
		04	NEW-ORDER	315	2.0GB
		05	TEMP	370	2.0GB
	/dev/rdsk/hdc42	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		09	STOCK	265	
		04	STOCK	265	2.0GB
		11	STOCK <= hda2102	265	
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdc43	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		09	CUSTOMER_IX2 <= hda2310	260	
		04	ORDERS	370	2.0GB
		05	ORDERS	370	2.0GB
WSA-54	/dev/rdsk/hdc50	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdc51	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		09	STOCK	265	
		04	NEW-ORDER	315	2.0GB
		05	TEMP	370	2.0GB
	/dev/rdsk/hdc52	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		09	CUSTOMER_IX2 <= hda6310	260	
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdc53	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		11	STOCK <= hdc3003	265	

SCSI	Device		Table Name	Size (Mbytes)	Disk
		04	ORDERS	370	2.0GB
		05	ORDERS	370	2.0GB
WSA-5	/dev/rdsk/hdc60	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdc61	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	NEW-ORDER	315	2.0GB
		05	TEMP	370	2.0GB
	/dev/rdsk/hdc62	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdc63	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	ORDERS	370	2.0GB
		05	ORDERS	370	2.0GB
WSA-15	/dev/rdsk/hdc70	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK => hdi1103	265	2.0GB
		04	STOCK => hdi3002	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdc71	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03 08	STOCK => hdb4412 STOCK	265 265	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		04	NEW-ORDER	315	2.0GB
		05	TEMP	370	2.0GB
	/dev/rdsk/hdc72	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		08	STOCK <= hda8206	265	
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB
	/dev/rdsk/hdc73	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		04	ORDERS	370	2.0GB
		05	ORDERS	370	2.0GB
WSA-25	/dev/rdsk/hdc80	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		08	STOCK	265	
		04	STOCK	265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdc81	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		08	STOCK	265	
		04	NEW-ORDER	315	2.0GB
		09	STOCK <= hda5001	265	
		05	TEMP	370	2.0GB
		10	STOCK <= hda6303	265	
	/dev/rdsk/hdc82	01	STOCK	265	2.0GB
		06	CUSTOMER	400	
		02	STOCK	265	2.0GB
		07	CUSTOMER	400	
		03	STOCK	265	2.0GB
		08	STOCK <= hda6001	265	
		04	STOCK	265	2.0GB
		05	TEMP	1810	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
	/dev/rdsk/hdc83	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03 08	STOCK STOCK <= hdc3002	265 265	2.0GB
		04	ORDERS	370	2.0GB
		05	ORDERS	370	2.0GB
WSA-35	/dev/rdsk/hdc90	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03 08	STOCK STOCK	265 265	2.0GB
		04 10	STOCK STOCK <= hda7113	265 265	2.0GB
		05	ORD-INDX-2	160	2.0GB
	/dev/rdsk/hdc91	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03 08	STOCK NEW-ORDER	265 315	2.0GB
		04 10	NEW-ORDER STOCK <= hda7306	315 265	2.0GB
		05	LOG	370	2.0GB
	/dev/rdsk/hdc92	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	STOCK	265	2.0GB
		05	LOG	1810	2.0GB
	/dev/rdsk/hdc93	01 06	STOCK CUSTOMER	265 400	2.0GB
		02 07	STOCK CUSTOMER	265 400	2.0GB
		03	STOCK	265	2.0GB
		04	ORDERS	370	2.0GB
		05	ORDERS	370	2.0GB
WSA-6	/dev/rdsk/hdi00	01	WARE	17	2.0GB
		02	WARE	17	2.0GB
		03	WARE	17	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		04	STOCK <= hda9007	265	2.0GB
		06	STOCK <= hda4103	265	
		05	STOCK <= hda2302	265	2.0GB
	/dev/rdsk/hdi01	01	WARE	17	2.0GB
		02	WARE	17	2.0GB
		03	WARE	17	2.0GB
		04	STOCK <= hda4305	265	2.0GB
		06	WARE	17	
		05	STOCK <= hda2104	265	2.0GB
	/dev/rdsk/hdi02	01	WARE	17	2.0GB
		02	WARE	17	2.0GB
		03	WARE	17	2.0GB
		04	STOCK <= hda2305	265	2.0GB
		06	STOCK <= hda6306	265	
		05	STOCK <= hda2101	265	2.0GB
	/dev/rdsk/hdi03	01	WARE	17	2.0GB
		02	WARE	17	2.0GB
		03	WARE	17	2.0GB
		04	STOCK <= hda9208	265	2.0GB
		06	WARE	17	
		05	STOCK <= hda4104	265	2.0GB
WSA-7	/dev/rdsk/hdi10	01	WARE	17	2.0GB
		02	WARE	17	2.0GB
		03	STOCK <= hda5105	265	2.0GB
		04	STOCK <= hda6101	265	2.0GB
		05	STOCK <= hda8303	265	2.0GB
	/dev/rdsk/hdi11	01	WARE	17	2.0GB
		02	WARE	17	2.0GB
		03	STOCK <= hdc7003	265	2.0GB
		04	STOCK <= hda2306	265	2.0GB
	/dev/rdsk/hdi12	01	WARE	17	2.0GB
		02	WARE	17	2.0GB
		03	STOCK <= hda4102	265	2.0GB
		04	STOCK <= hda6304	265	2.0GB
		05	STOCK <= hda4302	265	2.0GB
	/dev/rdsk/hdi13	01	WARE	17	2.0GB
		02	WARE	17	2.0GB
		03	WARE	17	2.0GB
		04	STOCK <= hda7301	265	2.0GB
		05	STOCK <= hda2203	265	2.0GB
WSA-45	/dev/rdsk/hdi20	01	STOCK <= hda4107	265	2.0GB
		02	WARE	17	2.0GB
	/dev/rdsk/hdi21	01	STOCK <= hda1106	265	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		02	STOCK <= hda9207	265	2.0GB
	/dev/rdsk/hdi22	01	STOCK <= hda9101	265	2.0GB
		02	STOCK <= hda2207	265	2.0GB
	/dev/rdsk/hdi23	01	STOCK <= hda3103	265	2.0GB
		02	WARE	17	2.0GB
WSA-55	/dev/rdsk/hdi30	01	STOCK <= hda8006	265	2.0GB
		02	STOCK <= hdc7004	265	2.0GB
		03	STOCK <= hda5107	265	2.0GB
	/dev/rdsk/hdi31	01	STOCK <= hda5206	265	2.0GB
		02	WARE	17	2.0GB
		03	WARE	17	2.0GB
	/dev/rdsk/hdi32	01	STOCK <= hda7002	265	2.0GB
		02	WARE	17	2.0GB
		03	WARE	17	2.0GB
	/dev/rdsk/hdi33	01	STOCK <= hda9002	265	2.0GB
		02	STOCK <= hdc0001	265	2.0GB
		03	WARE	17	2.0GB
WSA-16	/dev/rdsk/hdi40	01	STOCK <= hda8001	265	2.0GB
		02	STOCK <= hda7106	265	2.0GB
		03	STOCK <= hda7107	265	2.0GB
	/dev/rdsk/hdi41	01	STOCK <= hda9008	265	2.0GB
		02	STOCK <= hda4007	265	2.0GB
		03	STOCK <= hda5008	265	2.0GB
	/dev/rdsk/hdi42	01	STOCK <= hda3102	265	2.0GB
		02	STOCK <= hda8307	265	2.0GB
		03	STOCK <= hda6008	265	2.0GB
	/dev/rdsk/hdi43	01	STOCK <= hda1105	265	2.0GB
		02	STOCK <= hda3304	265	2.0GB
		03	CUSTOMER_IX2 <= hda5301	260	2.0GB
WSA-26	/dev/rdsk/hdi50	01	STOCK <= hda8005	265	2.0GB
		02	STOCK <= hda2304	265	2.0GB
		03	STOCK <= hda3303	265	2.0GB
	/dev/rdsk/hdi51	01	STOCK <= hda4106	265	2.0GB
		02	STOCK <= hda9004	265	2.0GB
		03	STOCK <= hda7104	265	2.0GB
	/dev/rdsk/hdi52	01	STOCK <= hda5204	265	2.0GB
		02	STOCK <= hda2001	265	2.0GB
		03	WARE	17	2.0GB
	/dev/rdsk/hdi53	01	STOCK <= hda5102	265	2.0GB
		02	STOCK <= hda6208	265	2.0GB
		03	WARE	17	2.0GB
WSA-36	/dev/rdsk/hdi60	01	STOCK <= hda4105	265	2.0GB

SCSI	Device		Table Name	Size (Mbytes)	Disk
		02	STOCK <= hda9204	265	2.0GB
		03	WARE	17	2.0GB
	/dev/rdsk/hdi61	01	STOCK <= hda6102	265	2.0GB
		02	STOCK <= hda6307	265	2.0GB
		03	WARE	17	2.0GB
	/dev/rdsk/hdi62	01	STOCK <= hda6305	265	2.0GB
		02	STOCK <= hda5106	265	2.0GB
		03	WARE	17	2.0GB
	/dev/rdsk/hdi63	01	STOCK <= hda2307	265	2.0GB
		02	STOCK <= hda2103	265	2.0GB
		03	WARE	17	2.0GB
WSA-46	/dev/rdsk/hdi70	01	STOCK <= hda7203	265	2.0GB
		02	STOCK <= hda4306	265	2.0GB
		03	WARE	17	2.0GB
	/dev/rdsk/hdi71	01	STOCK <= hda7307	265	2.0GB
		02	STOCK <= hda5205	265	2.0GB
		03	WARE	17	2.0GB
	/dev/rdsk/hdi72	01	STOCK <= hda4202	265	2.0GB
		02	STOCK <= hda2208	265	2.0GB
		03	WARE	17	2.0GB
	/dev/rdsk/hdi73	01	STOCK <= hda9301	265	2.0GB
		02	STOCK <= hda1104	265	2.0GB
		03	WARE	17	2.0GB

SCSI	Device		Table Name	Size (GB)	Disk
WSA-60	/dev/rdsk/hdd00	01	LOG	7.7	7.7GB
WSA-61	/dev/rdsk/hdd01	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd02	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd03	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd10	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd11	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd12	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd13	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd20	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd21	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd22	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd23	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd30	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd31	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd32	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd33	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd40	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd41	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd42	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd43	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd50	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd51	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd52	01	LOG	7.7	7.7GB
	/dev/rdsk/hdd53	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh00	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh01	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh02	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh03	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh10	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh11	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh12	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh13	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh20	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh21	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh22	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh23	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh30	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh31	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh32	01	LOG	7.7	7.7GB
	/dev/rdsk/hdh33	01	LOG	7.7	7.7GB

Appendix G: 180 Day Space Calculations

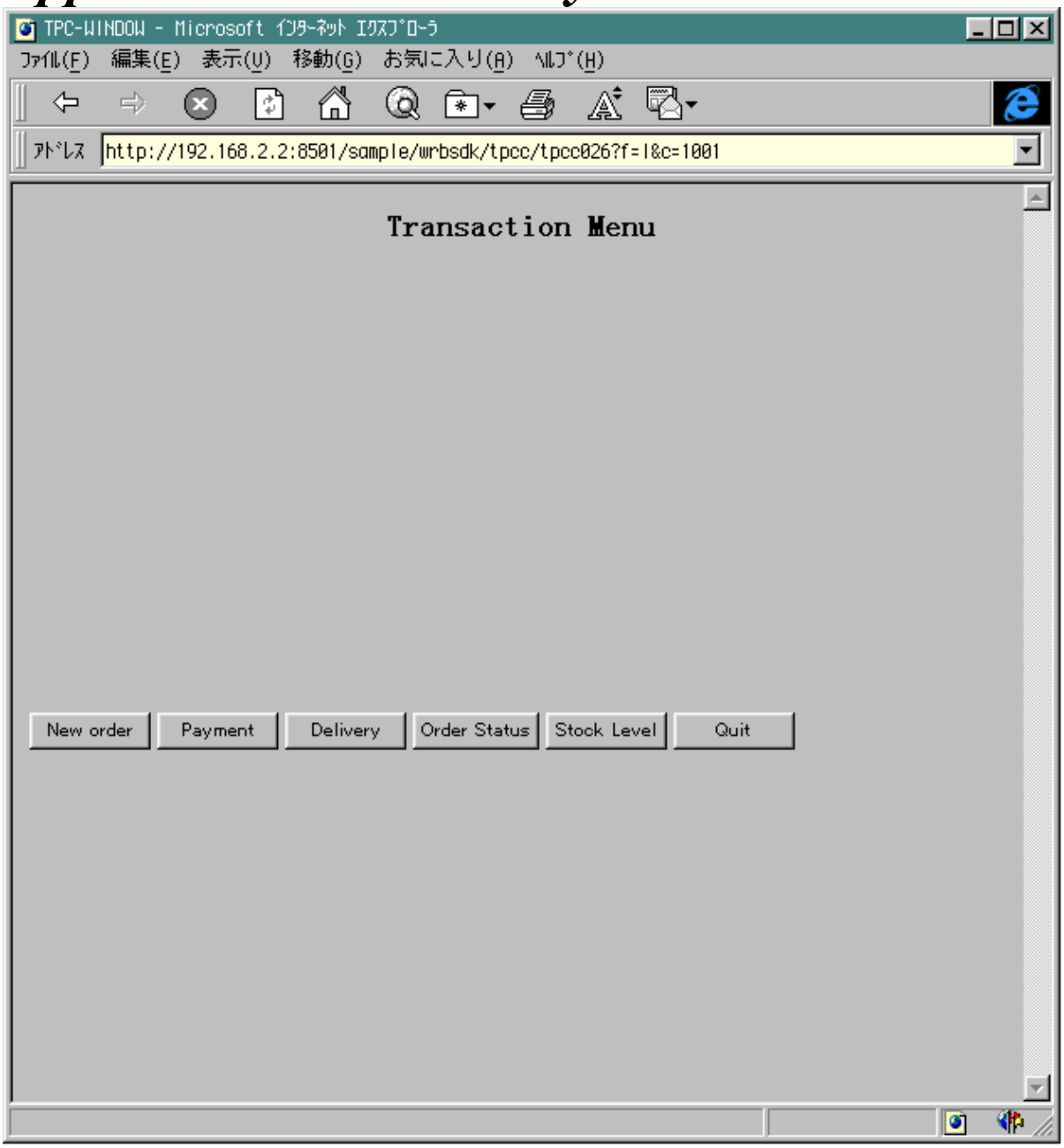
TPM	34,116.93				WAREHOUSES	3000
SEGMENT	TYPE	TSPACE	BLOCKS	FIVE_PCT	DAILY_GROW	TOTAL
CUSTOMER	TABLE	CUST	45,000,025	2,250,001	0	47,250,026
DISTRICT	TABLE	WARE	30,004	1,500	0	31,504
HISTORY	TABLE	HIST	2,552,013	0	464,357	3,016,370
ICUSTOMER	INDEX	ICUST1	1,133,501	56,675	0	1,190,176
ICUSTOMER2	INDEX	ICUST2	2,431,909	121,595	0	2,553,504
IDISTRICT	INDEX	WARE	300	15	0	315
IITEM	INDEX	ITEMS	1,000	50	0	1,050
INEW_ORDER	INDEX	INORD	363,852	18,193	0	382,045
IORDERS	INDEX	IORD1	1,070,447	53,522	0	1,123,969
IORDERS2	INDEX	IORD2	1,295,434	64,772	0	1,360,206
IORDER_LINE	INDEX	IORDL	12,668,753	633,438	0	13,302,191
ISTOCK	INDEX	ISTK	3,229,291	161,465	0	3,390,756
ITEM	TABLE	ITEMS	6,667	333	0	7,000
IWAREHOUSE	INDEX	WARE	150	8	0	158
NEW_ORDER	TABLE	NORD	240,494	12,025	0	252,519
ORDERS	TABLE	ORD	1,834,173	0	333,741	2,167,914
ORDER_LINE	TABLE	ORDL	33,730,571	0	6,137,512	39,868,083
ROLL_SEG	SYS	ROLL	770,560	0	0	770,560
STOCK	TABLE	STOCKS	60,000,002	3,000,000	0	63,000,002
SYSTEM	SYS	SYSTEM	768,512	0	0	768,512
WAREHOUSE	TABLE	WARE	3,004	150	0	3,154
TOTAL			**167,130,662**	**6,373,742**	**6,935,609**	**180,440,013**

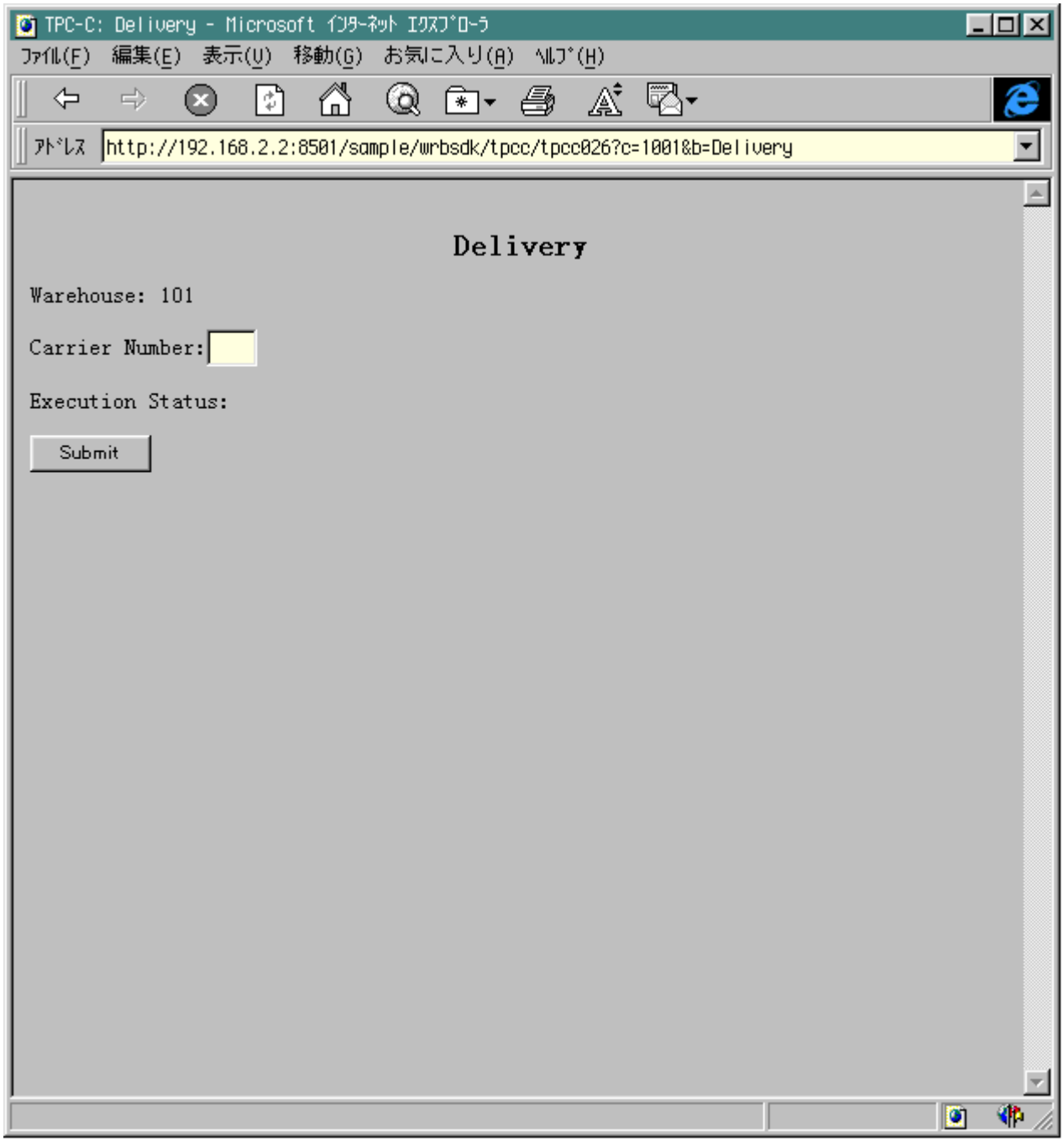
DYNAMIC SPACE	38,116,757	
STATIC SPACE	135,387,647	
FREE SPACE	6,935,609	
DAILY GROWTH	6,935,609	
DAILY SPREAD	0	ORACLE MAY BE CONFIGURED SUCH THAT DAILY SPREAD IS 0
180-DAY SPACE (BLK.)	1,383,797,308	
BLOCK SIZE (BYTES)	2,048	
180-DAY (GB)	2,639.38	
LOG BLOCK SIZE	512	
LOG BLOCKS/TPMC	31.90	NUMBER OF LOG BLOCKS USED PER NEW-ORDER
8-HOUR LOG (GB)	249.10	

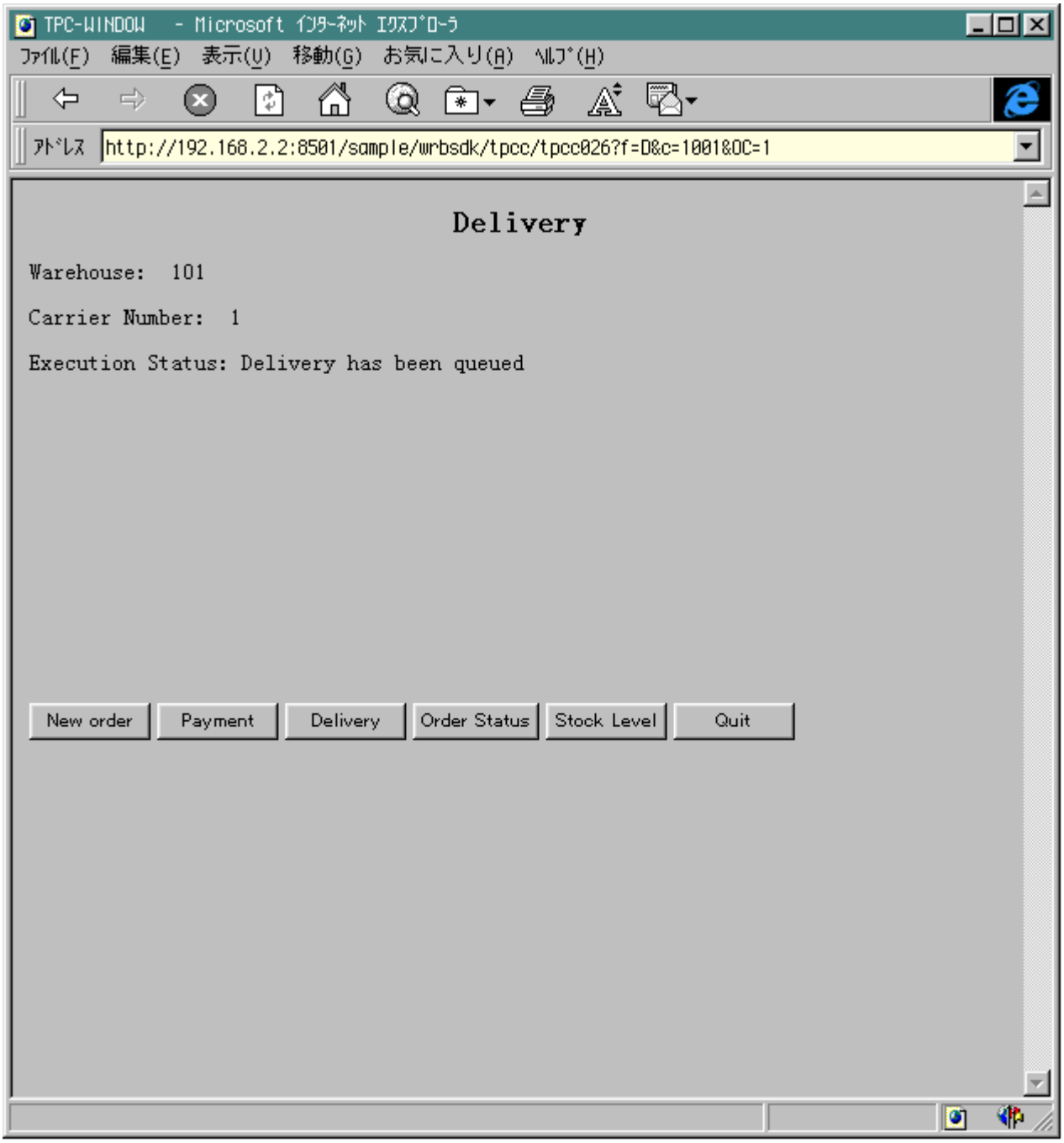
DATABASE DISKS	CAPACITY	NUMARRAY	EXTRAARRAY	TOTAL GB
	3.93	10	9	1,493.40
	1.95	30		1,170.00
				2,663.40

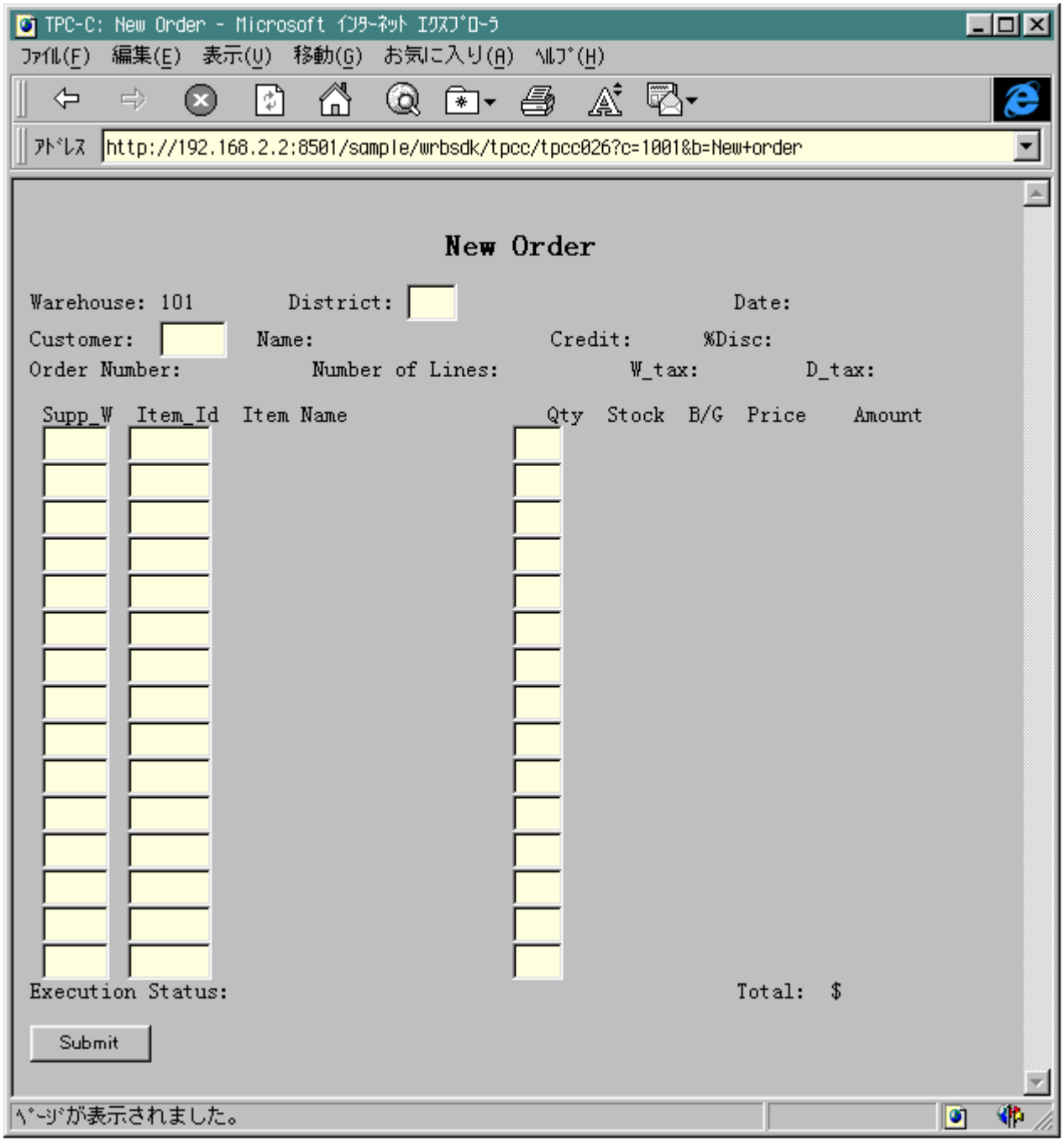
8-HR LOG DISKS	5@2GB CAPACITY	NUMARRAY		TOTAL GB
	7.81	8		249.92

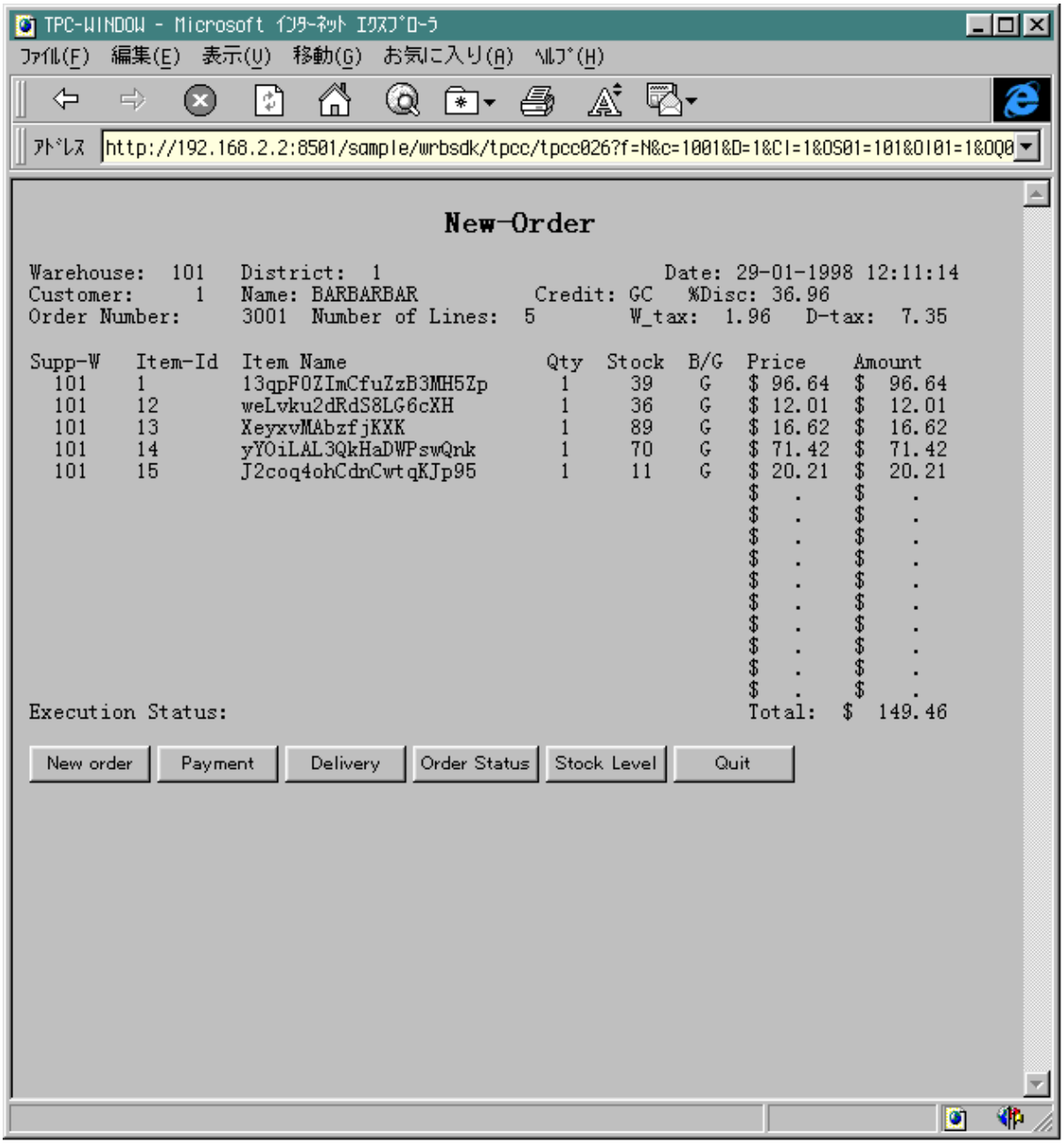
Appendix H: Screen Layouts

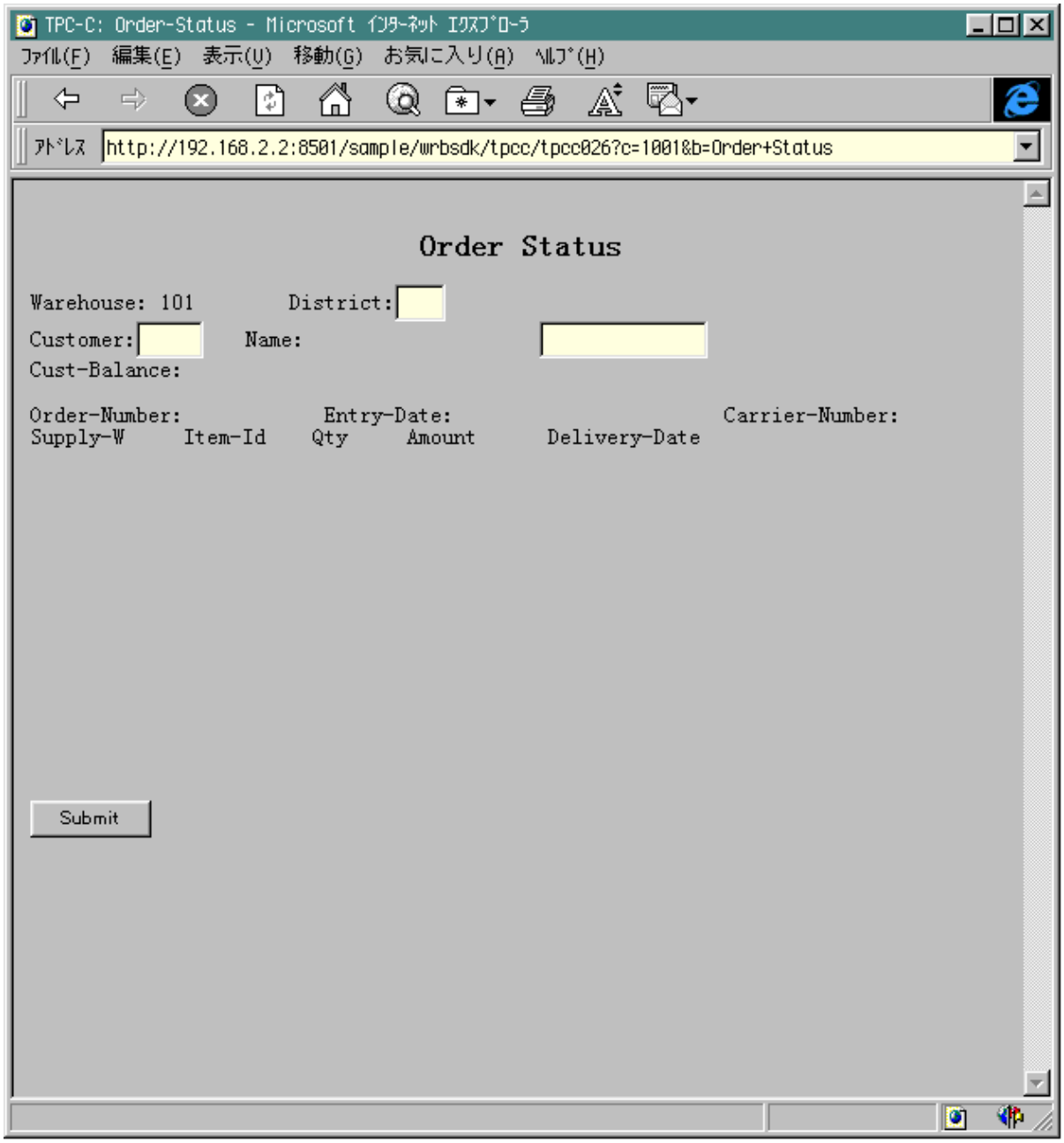


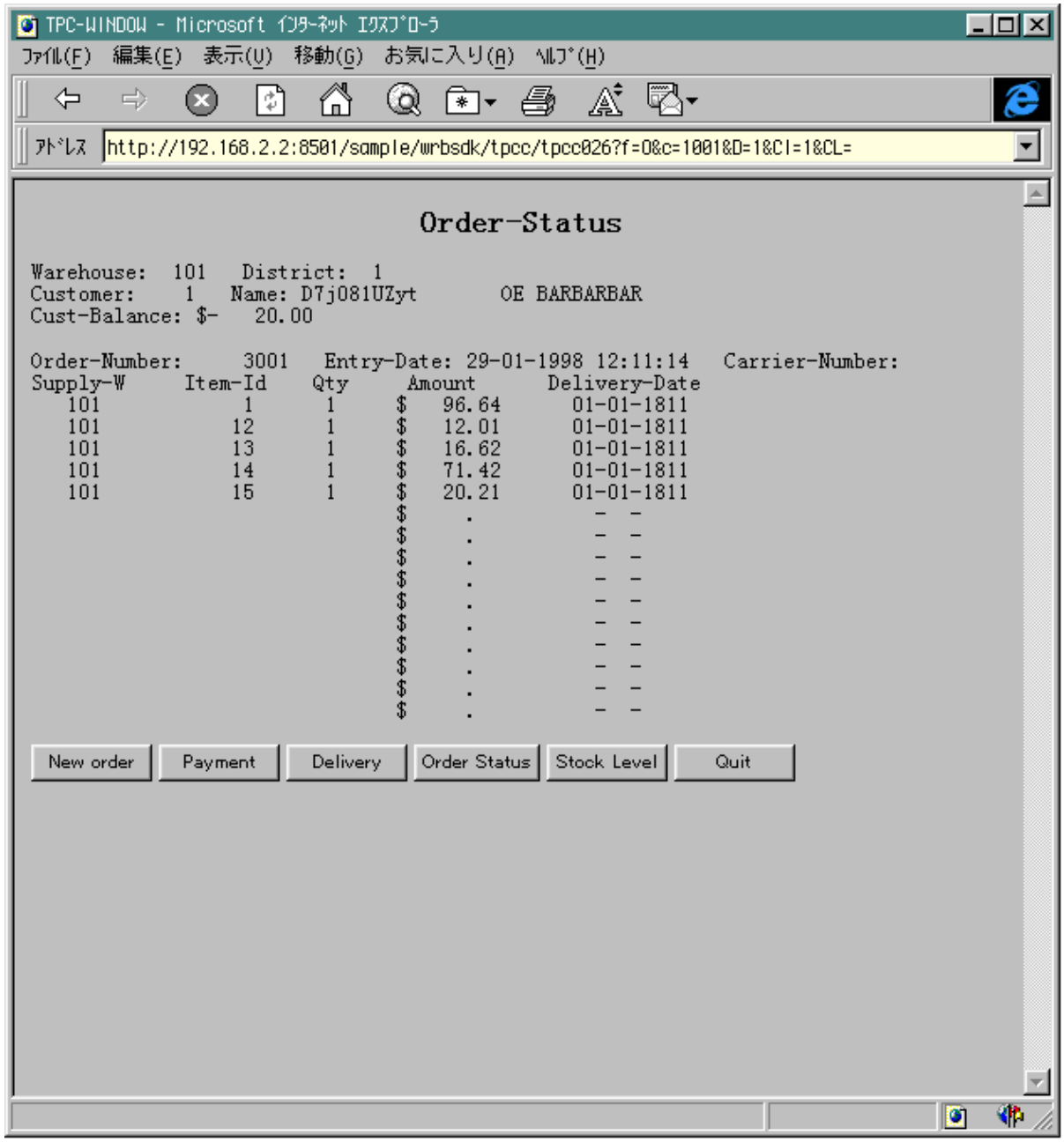

TPC-C: Delivery - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス
http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?c=1001&b=Delivery
Delivery
Warehouse: 101
Carrier Number:
Execution Status:
Submit

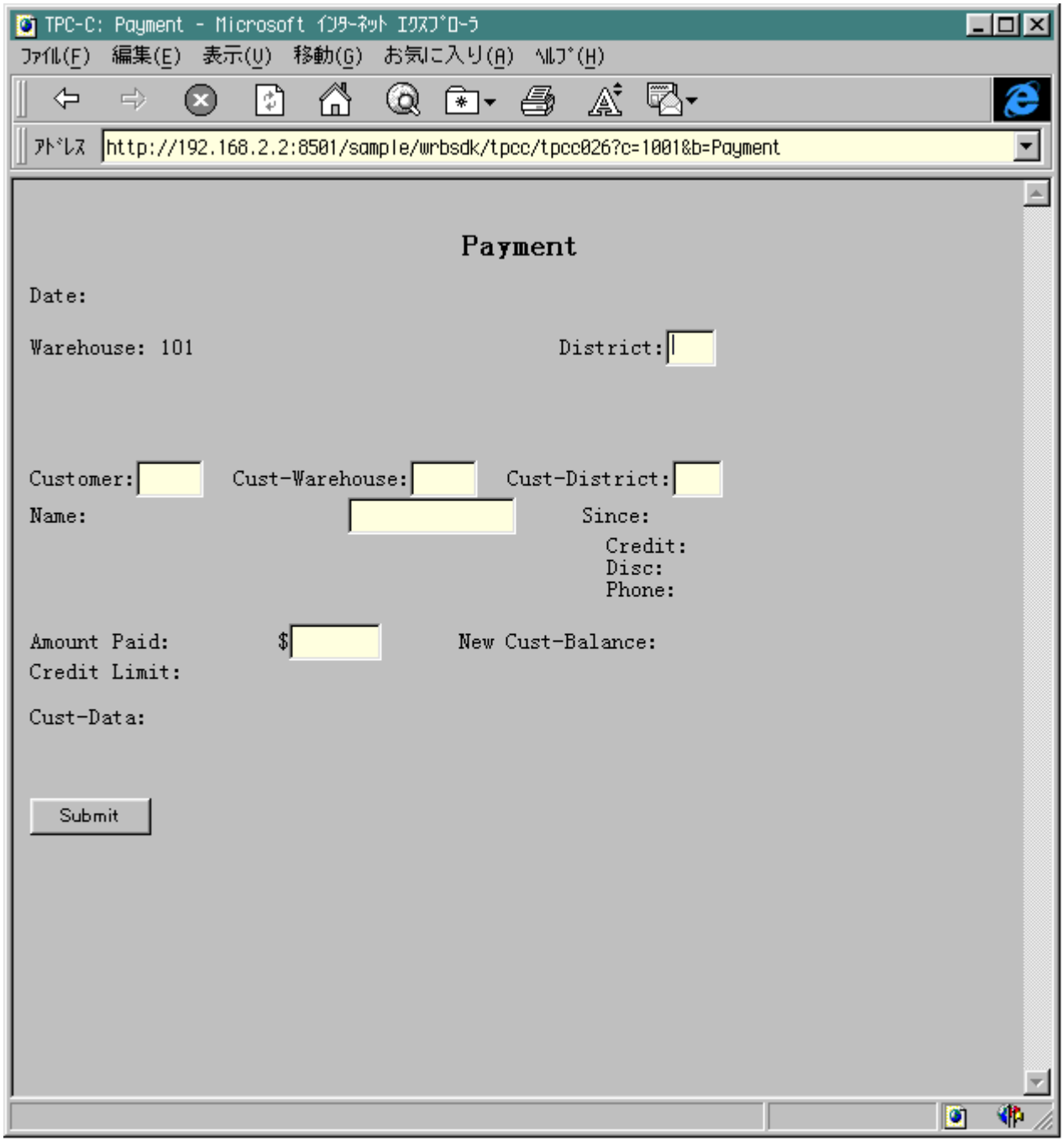

TPC-WINDOW - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?f=D&c=1001&OC=1
Delivery
Warehouse: 101
Carrier Number: 1
Execution Status: Delivery has been queued
New order
Payment
Delivery
Order Status
Stock Level
Quit

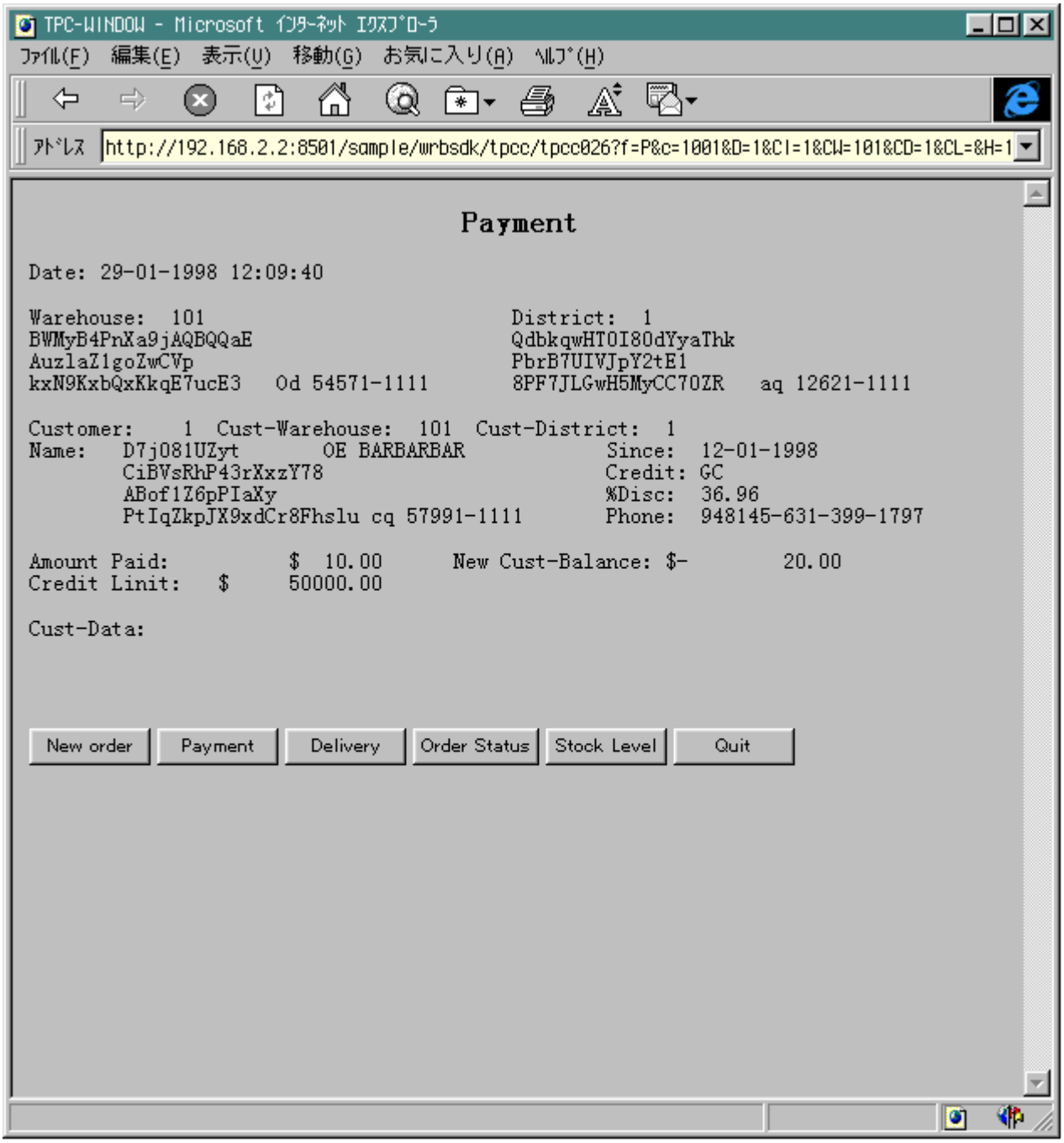

TPC-C: New Order - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?c=1001&b=New+order
New Order
Warehouse: 101 District:
Date:
Customer: Name: Credit: %Disc:
Order Number: Number of Lines: W_tax: D_tax:
Supp_W Item_Id Item Name Qty Stock B/G Price Amount
Execution Status:
Total: $
Submit
ページが表示されました。

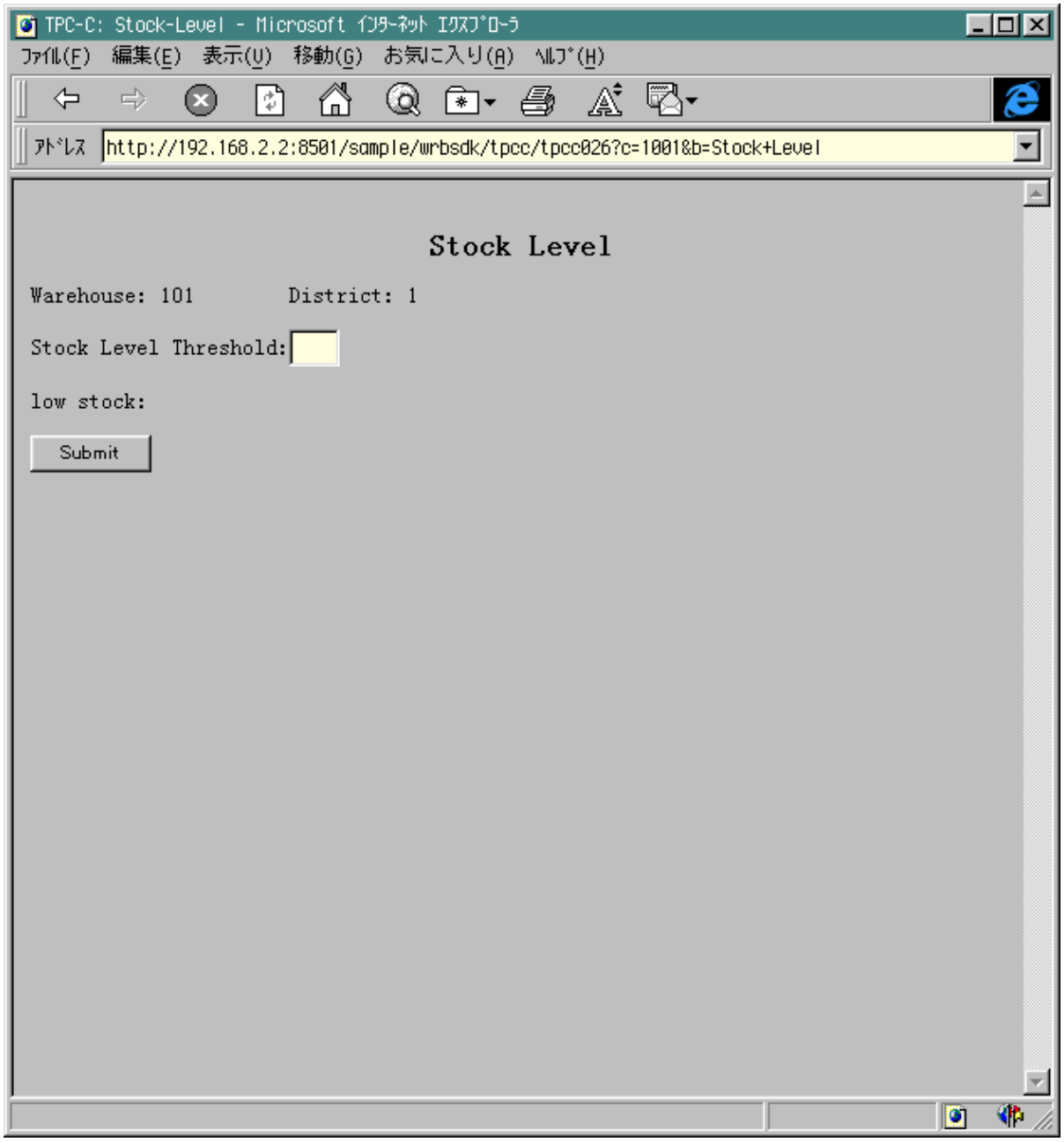

TPC-WINDOW - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?f=N&c=1001&D=1&CI=1&OS01=101&OI01=1&OQ0
New-Order
Warehouse: 101 District: 1 Date: 29-01-1998 12:11:14
Customer: 1 Name: BARBARBAR Credit: GC %Disc: 36.96
Order Number: 3001 Number of Lines: 5 W_tax: 1.96 D-tax: 7.35
Supp-W Item-Id Item Name Qty Stock B/G Price Amount
101 1 13qpFOZImCfuZzB3MH5Zp 1 39 G $ 96.64 $ 96.64
101 12 weLvku2dRdS8LG6cXH 1 36 G $ 12.01 $ 12.01
101 13 XeyxvMAbzfjKXK 1 89 G $ 16.62 $ 16.62
101 14 yYOiLAL3QkHaDWPswQnk 1 70 G $ 71.42 $ 71.42
101 15 J2coq4ohCdnCwtqKJp95 1 11 G $ 20.21 $ 20.21
Execution Status:
Total: $ 149.46
New order
Payment
Delivery
Order Status
Stock Level
Quit

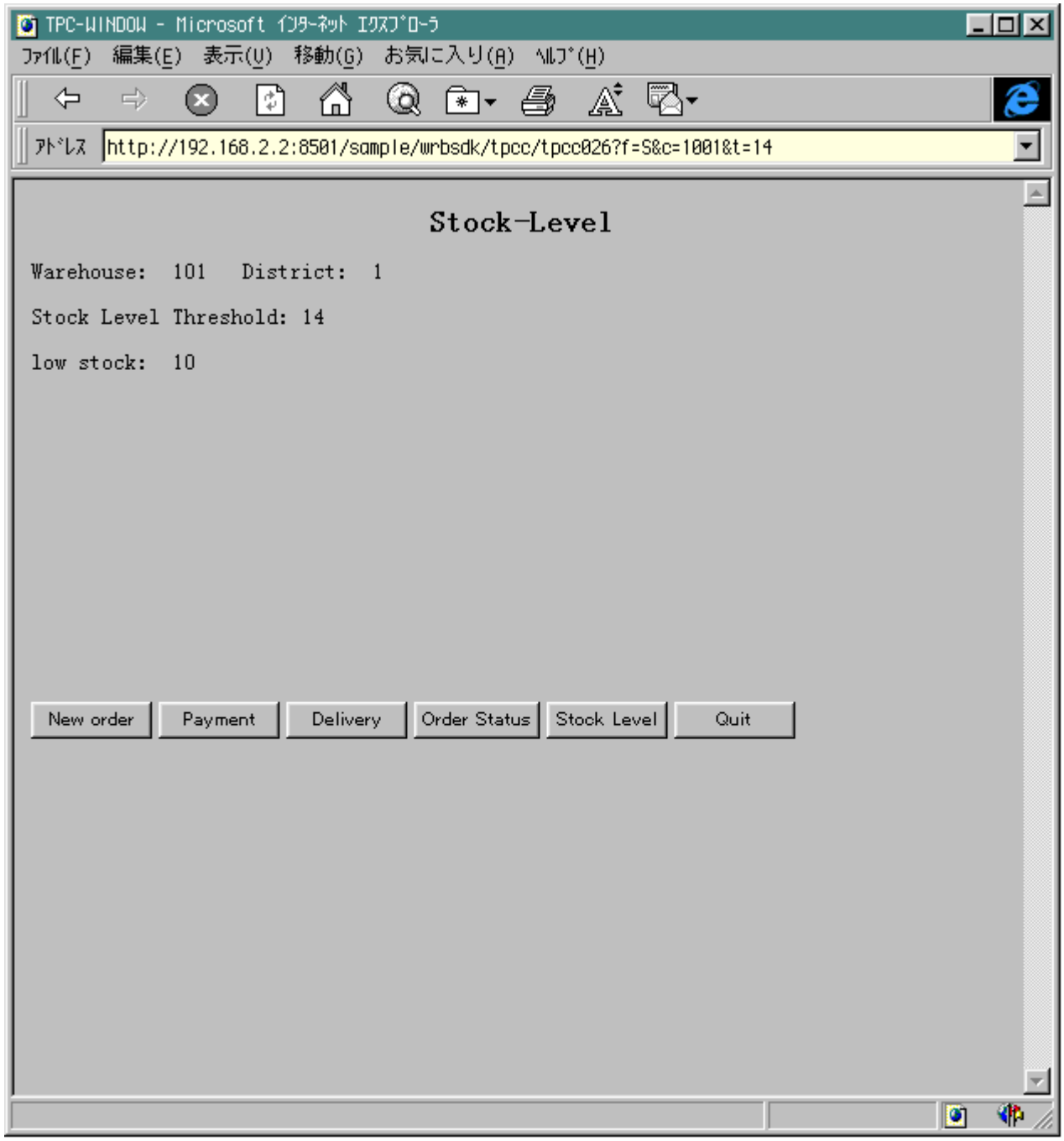

TPC-C: Order-Status - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?c=1001&b=Order+Status
Order Status
Warehouse: 101 District:
Customer: Name:
Cust-Balance:
Order-Number: Entry-Date: Carrier-Number:
Supply-W Item-Id Qty Amount Delivery-Date
Submit


TPC-WINDOW - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?f=0&c=1001&D=1&CI=1&CL=
Order-Status
Warehouse: 101 District: 1
Customer: 1 Name: D7jO81UZyt OE BARBARBAR
Cust-Balance: $- 20.00
Order-Number: 3001 Entry-Date: 29-01-1998 12:11:14 Carrier-Number:
Supply-W Item-Id Qty Amount Delivery-Date
101 1 1 $ 96.64 01-01-1811
101 12 1 $ 12.01 01-01-1811
101 13 1 $ 16.62 01-01-1811
101 14 1 $ 71.42 01-01-1811
101 15 1 $ 20.21 01-01-1811
New order
Payment
Delivery
Order Status
Stock Level
Quit


TPC-C: Payment - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?c=1001&b=Payment
Payment
Date:
Warehouse: 101
District:
Customer:
Cust-Warehouse:
Cust-District:
Name:
Since:
Credit:
Disc:
Phone:
Amount Paid:
$
New Cust-Balance:
Credit Limit:
Cust-Data:
Submit


TPC-WINDOW - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?f=P&c=1001&D=1&CI=1&CW=101&CD=1&CL=&H=1
Payment
Date: 29-01-1998 12:09:40
Warehouse: 101
BWMyB4PnXa9jAQBQQaE
AuzlaZ1goZwCVp
kxN9KxbQxKkqE7ucE3 Od 54571-1111
District: 1
QdbkqwHTOI8OdYyaThk
PbrB7UIVJpY2tE1
8PF7JLGwH5MyCC7OZR aq 12621-1111
Customer: 1 Cust-Warehouse: 101 Cust-District: 1
Name: D7j081UZyt OE BARBARBAR
CiBVsRhP43rXxzY78
ABof1Z6pPIaXy
PtIqZkpJX9xdCr8Fhslu cq 57991-1111
Since: 12-01-1998
Credit: GC
%Disc: 36.96
Phone: 948145-631-399-1797
Amount Paid: $ 10.00
Credit Linit: $ 50000.00
New Cust-Balance: $- 20.00
Cust-Data:
New order
Payment
Delivery
Order Status
Stock Level
Quit


TPC-C: Stock-Level - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス
http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?c=1001&b=Stock+Level
Stock Level
Warehouse: 101
District: 1
Stock Level Threshold:
low stock:
Submit


TPC-WINDOW - Microsoft インターネット エクスプローラ
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A) ヘルプ(H)
アドレス http://192.168.2.2:8501/sample/wrbsdk/tpcc/tpcc026?f=S&c=1001&t=14
Stock-Level
Warehouse: 101 District: 1
Stock Level Threshold: 14
low stock: 10
New order
Payment
Delivery
Order Status
Stock Level
Quit

Appendix I: Price Quotes

Allied Telesis, K. K.
Takanawadai Green Building
4-6-6 Higashi Gotanda,
Shinagawaku,tokyo 141 Japan
Tel:03-3443-9516,Fax:03-5488-6798

DATE 1/16/1998

Company	Fujitsu Limited
Address	3-9-18 Sinyokohama,Kouhokuku
City	Yokohama
Tel	045-471-1759
Fax	045-473-3744

QTY	DESCRIPTION	UNIT	TOTAL
1	CentreCOM 3024TR	¥89,800	¥89,800
1	Maintenance Fee		¥5,380/year

NOTE : ① Consumption tax not included.
② 1 year warranty
③ Valid for 60 days
④ The maintenance fee above is for send back repair.

Thank you for your consideration.

Best regards,

Kunihide Sakai

Kunihide Sakai
Chief,Sales Dept

Appendix J: Auditor's attestation letter

PERFORMANCE METRICS INC.
TPC Certified Auditors

January 27, 1998

Mr. Makoto Katoh
Manager, Performance Engineering Dept.
Development Promotion Division
Fujitsu Limited
3-9-18 Shinyokohama
Kohoku-ku, Yokohama 222-0030, Japan

Mr. Karl Haas
Director, OLTP Performance
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065
USA

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

Server: Fujitsu GRANPOWER 7000 Model 800
Database Manager: Oracle8 Enterprise Edition V8.0.3
Operating System: UXP/DS V20L30
Transaction Monitor: TP-Base V20
Web Server: Oracle WebServer V2.1

CPUs	Memory	Disks	New-Order Response Time @90%	tpmC
Server: Fujitsu GRANPOWER 7000 Model 800				
18 UltraSPARC @ 250 MHz	Main: 16 GB L2 Cache: 4 MB	1,320 @ 1.95 GB 205 @ 3.93 GB	2.82 sec.	34,116.93
28 Clients: Fujitsu GRANPOWER 7000 Model 200				
2 UltraSPARC @ 250 MHz	Main: 256 MB L2 Cache: 1 MB	C01-C25:1 @ 2.50 GB C26-C28:1 @ 4.53 GB	n/a	n/a
1 Client: Fujitsu GRANPOWER 7000 Model 200				
2 UltraSPARC @ 300 MHz	Main: 256 MB L2 Cache: 2 MB	C29: 1 @ 4.53 GB	n/a	n/a

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

The following attributes of the benchmark were given special attention:

- The transactions were correctly implemented.
- The database files were properly sized and populated.

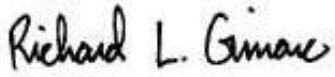
2229 Benita Dr., Suite 101, Rancho Cordova, CA 95670
(916) 635-2822 Fax: (916) 858-0109 e-mail: Richard@PerfMetrics.com
Page 1

PERFORMANCE METRICS INC.
TPC Certified Auditors

- The database was properly scaled with 3,000 warehouses. Only 2,900 warehouses were active during measurement. Rows for inactive warehouses were deleted from the WAREHOUSE table prior to measurement.
- The ACID properties were met.
- The durability data loss and log loss tests were performed on a 10-warehouse database.
- Input data was generated according to the specified percentages.
- Eight hours of log space was configured on the priced system. The measured system had more than 8 hours of configured log media. Forty 1.95 GB disks accounted for the over-configuration and were not included in the priced configuration.
- 520 of the 1.95 GB disks on the server contained backup data and were not included in the priced system. Performance data was collected during steady state to confirm that these disks were not active during the benchmark.
- The data for the 180-day space calculation was verified. 180 3.93 GB disks were added to the priced configuration to satisfy this requirement.
- A single 2.50 GB disk was substituted for the 4.53 GB disk on four of the client machines. Performance data collected during steady state demonstrated that this disk substitution was performance neutral.
- Measurement cycle times included a 0.1 second menu and 0.1 second response time delay for an emulated Web browser.
- The steady state portion of the test was 30 minutes.
- One checkpoint was taken during the steady state portion of the test.
- Checkpoints were verified to be clear of the guard zones.
- There were 29,000 user contexts present on the system.
- Each emulated user had a unique starting random number seed.
- The NURand constants used for database load and run-time were verified.
- System pricing was checked for major components and maintenance.

Additional Audit Notes: none

Regards,



Richard L. Gimarc
Auditor

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