

IBM eServer pSeries 690 Turbo c/s Model 7040-681

using

Oracle9i Enterprise Edition Release 2 , v9.2.0.1 for AIX 5L v5.2

IBM Websphere Application Server Enterprise Edition Version 3.0

TPC BenchmarkTM C

Full Disclosure Report

IBM System Performance and Evaluation Center

Submitted For Review

December 26, 2002



Special Notices

The following terms used in this publication are trademarks of **International Business Machines** Corporation in the United States and/or other countries:

eServer pSeries
RISC System/6000
AIX
IBM

The following terms used in this publication are trademarks of other companies as follows:

TPC Benchmark	Trademark of the Transaction Processing Performance Council
ORACLE, SQL*Loader	Trademark of Oracle, Inc.
Oracle9i, SQL*Net and SQL*Plus	Trademark of Oracle, Inc.

First Edition December 26, 2002

The information contained in this document is distributed on an AS IS basis without any warranty either expressed or implied. The use of this information or the implementation of any of these techniques is a customer's responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment.

While each item has been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used; any functionally equivalent program may be used.

It is possible that this material may contain references to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such products, programming, or services in your country.

All performance data contained in this publication was obtained in a controlled environment, and therefore the results which may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data in their specific environment.



Request for additional copies of this document should be sent to the following address:

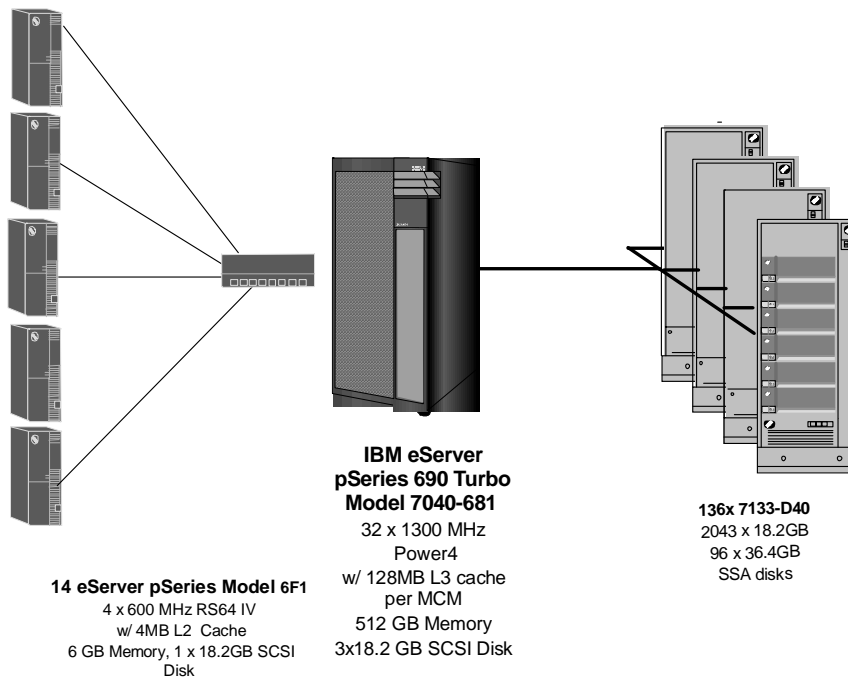
TPC Benchmark Administrator
IBM Commercial Performance
Mail Stop 9571
11501 Burnet Road
Austin, TX 78758
FAX Number (512) 838-1852

© **Copyright International Business Machines Corporation 2002. All rights reserved.**

Permission is hereby granted to reproduce this document in whole or in part, provided the copyright notice printed above is set forth in full text on the title page of each item reproduced.

NOTE: US. Government Users - Documentation related to restricted rights: Use, duplication, or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

		IBM eServer pSeries 690 Turbo Model 7040-681		TPC-C Rev. 5.0					
				Report Date Dec 26, 2002					
Total System Cost		TPC-C Throughput		Price/Performance		Availability Date			
\$7,591,208		427,760.83 tpmC		17.75\$/tpmC		May 31, 2003			
Processors		Database Manager		Operating System		Other Software		No. Users	
32 x database 56 x clients		Oracle9i Enterprise Edition Release 2 v9.2.0.1 for AIX 5L v5.2		AIX 5L V5.2		Websphere Application Server Enterprise Edition Version 3.0		340,000	



System Components	Clients		Server	
	Quantity	Description	Quantity	Description
Processor	14	4 x 600 MHz RS64 IV W/4MB L2 cache each	1	32 x 1300 MHz POWER4 w/ 128 MB L3 cache per MCM, 4 MCM in the SUT
Memory		6 GB		512 GB
Disk Controllers	1	SCSI-2 (Integrated)	1	SCSI-2 (Integrated)
Disk Drives	1	18.2 GB SCSI each client	34	SSA Adapters
			2,043	18.2 GB SSA disk
			96	36.4 GB SSA disk
			3	18.2 GB SCSI disk
Total Storage		16.93 GB each client		37,325.49
Terminals	1	System Console	1	System Console



ORACLE

IBM eServer pSeries 690 Turbo Model 7040-681

TPC-C Rev. 5.0

Report Date: December 26, 2002

Description	Part No.	Source	Unit Price	Qty	Ext Price	Maint Price
Server Hardware						
IBM eServer pSeries 690 (CD, cables, clock card)	7040-681	1	18,764	1	18,764	20,760
128MB (4x32) L3 Cache, 433MHz	4139	1	25,000	4	100,000	0
64GB H-Memory (inward facing)	4488	1	241,000	4	964,000	0
64GB H-Memory (outward facing)	4489	1	241,000	4	964,000	0
1.3GHz 8-way Processor MCM (1 thru 4 books)	5244	1	275,000	4	1,100,000	403,200
Std CEC Fan Pwr cblgrp, BPC(F&R), cable	6161	1	600	1	600	0
Standard CEC DC/DC Converter Assembly	6170	1	4,700	2	9,400	0
Power Cable Group, 1 thru 4 processor modules	6181	1	720	1	720	0
Additional DC/DC Converter Assembly, (DCA)	6189	1	4,200	3	12,600	0
Capacitor book (for 2 MCMs)	6198	1	1,800	2	3,600	0
Service Processor + Dual Rio Loop (GX), I/O loop adapter	6404	1	15,000	1	15,000	0
Regatta-H Planar (Mayfly)	6565	1	50,000	1	50,000	0
Media Drwr, 1U, Op Pnl/Dskt/4 SCSI Media Bays	8692	1	1,700	1	1,700	0
I/O Drawer	7040-61D	1	3,980	4	15,920	16,320
SCSI cable - B&C to Media Drawer	2122	1	275	1	275	0
Remote I/O Cable, 500MHz, 0.5m (RIO)	3145	1	310	4	1,240	0
Remote I/O Cable, 500MHz, 2m (RIO)	3149	1	350	8	2,800	0
18.2 GB Ultra3 10K RPM 1" U3 Module	3157	1	1,260	3	3,780	0
10/100 Mbps Ethernet adapter II	4962	1	275	5	1,375	0
Cable Grp, 4xUPIC/2xRIO, BPM(IO#1thru 4) to B&C(pos 4609)	6121	1	1,600	1	1,600	0
I/O Drwr DC/DC Converter assembly, (DCA)	6172	1	4,000	8	32,000	0
Power Cable, B&C to Media Drawer	6179	1	300	1	300	0
Ultra SCSI SE Adapter	6206	1	395	1	395	0
Advanced Serial RAID Adapter	6230	1	3,000	34	102,000	0
32 MB Fast Write Cache option card	6235	1	575	6	3,450	0
B&C Planar, 10 PCI Slots, 2x integrated SCSI	6563	1	8,000	8	64,000	0
Ultra3 SCSI 4-Pack	6564	1	500	16	8,000	0
Rack (Front primary and secondary racks)	7040-61R	1	9,050	1	9,050	3,960
Back Door, Slim Line (primary or secondary)	6074	1	750	2	1,500	0
Bulk Power Regulator (BPR)	6186	1	4,000	4	16,000	0
Bulk Power Controller (BPC), 4 Fans + 3 DCAs	6187	1	1,900	2	3,800	0
Bulk Power Distribution (BPD), 10 DCAs	6188	1	3,500	4	14,000	0
Integrated Battery Backup and Redundant + cable	6200	1	4,600	4	18,400	0
Line Cord, 60A, 14', IEC309 Plug, Chargeable	8678	1	1,000	2	2,000	0
Regatta-H Bulk Power Module (2x BPA, EPO PNL)	8690	1	5,000	1	5,000	0
Expansion Rack 24", 42 EIA	8691	1	8,000	1	8,000	0
System Console + cable	7316	1	4,400	1	4,400	0
Display, White P76/P77 Color Monitor	3627	1	675	1	675	0
				Sub total	3,560,344	444,240
Storage						
System Rack Model T42 (6089,6098)	7014-T42	1	4,150	14	58,100	12,432
Additional Power Distribution Unit	6171	1	1,000	41	41,000	0
SSA Disk Subsystem	7133-D40	1	12,750	136	1,734,000	600,576
50/60 Hz AC, 300 VDC Power Supply	8022	1	2,000	136	272,000	0
Raven Black Drawer Cover	8031	1	250	136	34,000	0
Dummy Drive	8399	1	30	2	60	0
10K/18.2 GB Advanced Disk Drive Module	8518	1	4,500	2,043	9,193,500	0
10K/36.4 GB Advanced Disk Drive Module	8536	1	5,900	96	566,400	0
SSA Cables (8801, 8802)	8801	1	125	136	17,000	0
				Sub total	11,916,060	613,008
Server Software						
AIX 5.1 (media only)	5765-E61	1	50	1	50	0
IBM Websphere Application Server	D5A LCLL	1	26,687	32	853,984	158,144
IBM C for AIX V5	D5A 1ELL	1	559	1	559	104
Oracle9i Database Enterprise Edition Release 2, v9.2.0.1 for AIX 5L v5.2, Processor 3 year term for 32 processors, Unlimited Users		2	640,000	1	640,000	0
Oracle Database Server Support Package for 3 years		2				6,000
				Sub total	1,494,593	164,248
Client Hardware and Software						
WorkGroup server pSeries 620 Model 6F1	7025-6F1	1	7,997	14	111,958	67,200
32x Speed CD-ROM Drive	2624	1	375	14	5,250	0
18.2 GB 10K RPM Ultra SCSI Enhanced Disk	3153	1	1,260	14	17,640	0
P76/p77 Color Monitor, captured cable + keyboard	3627	1	675	1	675	0
16 slot memory expansion card	4075	1	1,500	14	21,000	0
1024MB (2x512MB) 200PIN SDRAM DIMM Memory	4100	1	3,072	56	172,032	0
512MB (2x256MB) SDRAM DIMM Memory	4119	1	1,536	56	86,016	0
4-port 10/100 Mbps Ethernet adapter	4961	1	1,500	84	126,000	0
Proc card 4xRS64 IV/600MHz/4MB cache	5214	1	24,500	14	343,000	60,480
IPL Disk Mounting HW cables, terminator	6555	1	595	14	8,330	0
AIX 4. Unlimited Users	5765-C34	1	50	14	700	0
				Sub total	892,601	127,680
Third Party Hardware						
8-port 10/100 Ethernet Switch	FS108NA	3	85	4	340	0
				Sub total	340	0
Oracle Mandatory E-Business Discount on License and Support		2	(129,200)	1	(129,200)	0
				Discounts	(10,908,034)	(584,672)
				Total	6,826,704	764,504
Audited by: Brad Askins, Info Sizing						
Pricing Sources:						
Three-Year Cost of Ownership						\$7,591,208
1=IBM: Bill Casey, eServer pSeries Offering Manager, wrcasey@us.ibm.com, 512-838-1422						tpm c 427,760.83
2=Oracle: Herve Lejeune, herve.lejeune@oracle.com, 650-506-1894						\$/tpm c \$17.75
3=NETGEAR						

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

Numerical Quantities Summary for the IBM eServer pSeries 690 Turbo Model 7040-681

MQTH, computed Maximum Qualified Throughput: 427,760.83 tpmC

<u>Response Times (in seconds)</u>	<u>90th %</u>	<u>Average</u>	<u>Maximum</u>
New Order	0.90	0.45	97.30
Payment	0.85	0.42	96.37
Order-Status	0.88	0.44	48.76
Delivery (interactive)	0.19	0.11	48.16
Delivery (deferred)	0.19	0.11	3.73
Stock-Level	0.83	0.40	96.28
Menu	0.01	0.00	96.14

<u>Transaction Mix, in percent of total transactions</u>	<u>Percent</u>
New Order	44.93%
Payment	43.00%
Order-Status	4.01%
Delivery	4.01%
Stock-Level	4.01%

<u>Keying/Think Times (in seconds)</u>	<u>Min.</u>	<u>Average</u>	<u>Max.</u>
New Order	18.00/0.01	18.01/12.02	18.11/120.22
Payment	3.00/0.01	3.01/12.02	3.11/120.21
Order-Status	2.00/0.01	2.01/10.01	2.10/100.20
Delivery	2.00/0.01	2.01/5.02	2.10/50.21
Stock-Level	2.00/0.01	2.01/5.02	2.11/50.20

Test Duration

Ramp-up Time	1 hour 18 sec
Measurement interval	2 hours
Transactions during measurement interval (all types)	114237469
Ramp-down time	20 minutes

Checkpointing

Number of checkpoints	4
Checkpoint interval	29 min 51 sec

Abstract

This report documents the full disclosure information required by the TPC Benchmark™ C Standard Specification Revision 5.0 dated February 26, 2001 for measurements on the IBM eServer pseries 690 Turbo Model 7040-681. The phrase RS/6000 will be substituted for RISC System/6000 for the remainder of this document.

The software used on the IBM eServer pseries 690 Turbo Model 7040-681 includes AIX 5L Version 5.2 operating system, Oracle9i Server database manager, and Websphere Application Server Enterprise Edition Version 3.0 for AIX transaction manager.

IBM eServer pSeries 690 Turbo Model 7040-681

Company Name	System Name	Data Base Software	Operating System Software
IBM Corporation	IBM eServer pSeries 690 Turbo Model 7040-681		AIX 5L Version 5.2
Oracle Corporation		Oracle9i Enterprise Edition Release 2, v9.2.0.1 for AIX 5L v5.2	

Total System Cost	TPC-C Throughput	Price/Performance
<ul style="list-style-type: none">• Hardware• Software• 3 Years Maintenance	Sustained maximum throughput of system running TPC-C expressed in transactions per minute	Total system cost/tpmC
\$7,591,208	427,760.83	\$17.75per tpm-C

Preface

TPC Benchmark™ C Standard Specification was developed by the Transaction Processing Performance Council (TPC). It was released on August 13, 1992 and updated with revision 5.0 on February 26, 2001.

This is the full disclosure report for benchmark testing of the IBM eServer pSeries 690 Turbo Model 7040-681 according to the TPC Benchmark™ C Standard Specification.

TPC Benchmark™ C exercises the system components necessary to perform tasks associated with that class of on-line transaction processing (OLTP) environments emphasizing a mixture of read-only and update intensive transactions. This is a complex OLTP application environment exercising a breadth of system components associated by such environments characterized by:

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

This benchmark defines four on-line transactions and one deferred transaction, intended to emulate functions that are common to many OLTP applications. However, this benchmark does not reflect the entire range of OLTP requirements. The extent to which a customer can achieve the results reported by a vendor is highly dependent on how closely TPC-C approximates the customer application. The relative performance of systems derived from this benchmark does not necessarily hold for other workloads or environments. Extrapolations to any other environment are not recommended.

Benchmark results are highly dependent upon workload, specific application requirements, and 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 benchmarks when critical capacity planning and/or product evaluation decisions are contemplated.

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.

1. General Items

1.1 Application Code Disclosure

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 and output functions.

Appendix A contains the RS/6000 application code for the five TPC Benchmark™ C transactions. Appendix D contains the terminal functions and layouts.

1.2 Benchmark Sponsor

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

This benchmark was sponsored by **International Business Machines Corporation** and **Oracle Corporation**.

1.3 Parameter Settings

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

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

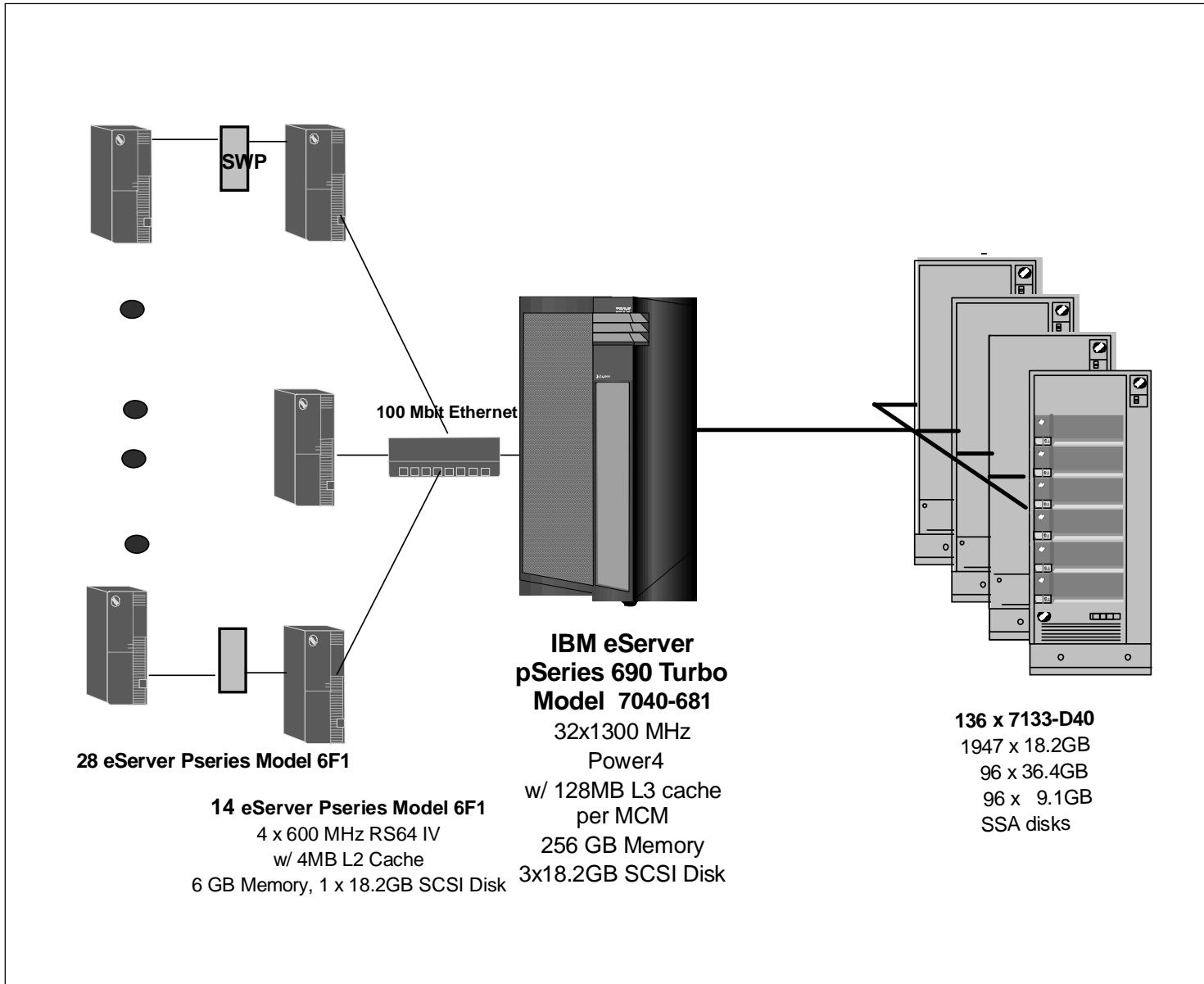
Appendix B contains the system, data base, and application parameters changed from their default values used in these TPC Benchmark™ C tests.

1.4 Configuration Diagrams

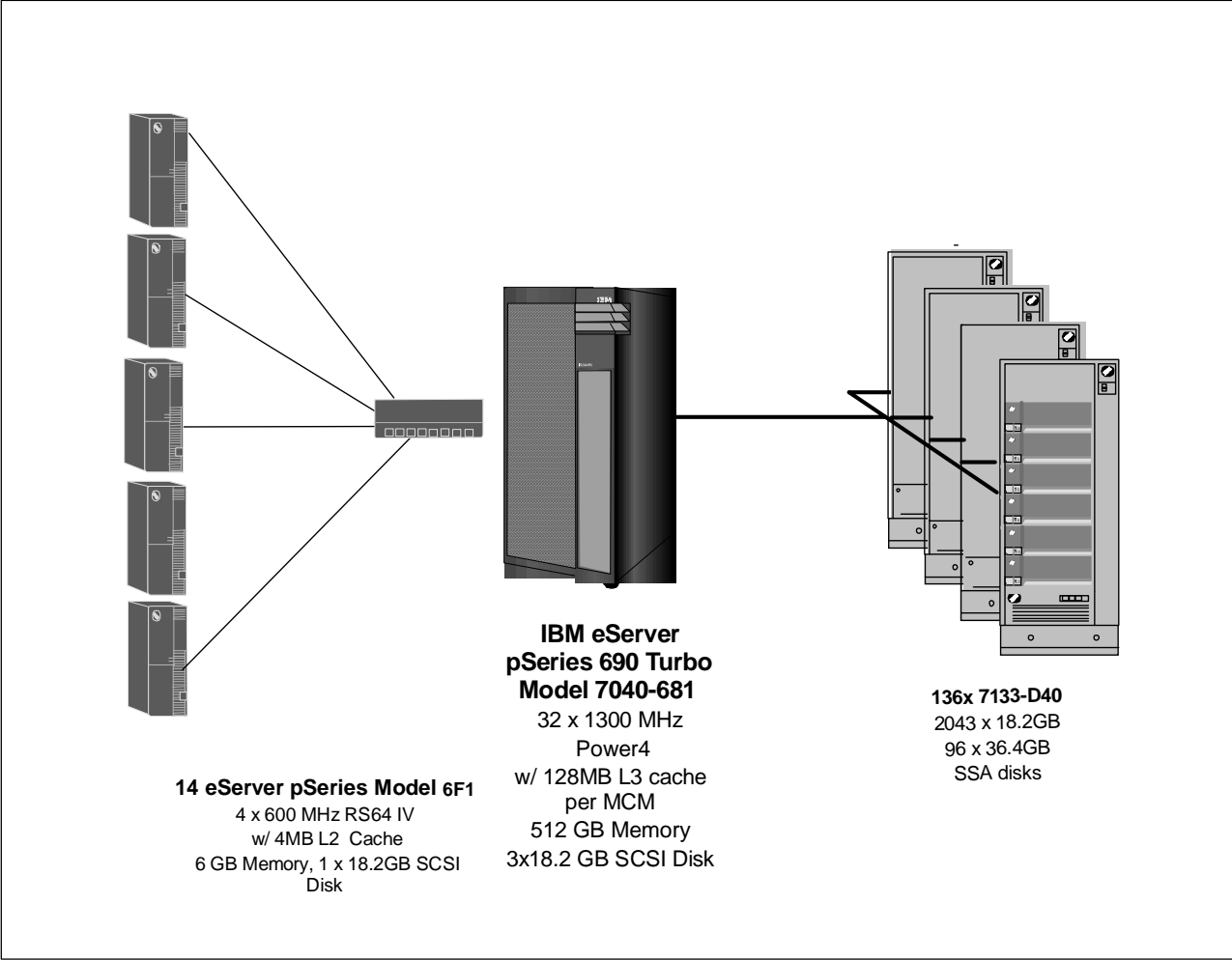
Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences. This includes, but is not limited to:

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

IBM eServer pseries 690 Turbo Model 7040-681 Benchmark Configuration



IBM eServer pseries 690 Turbo Model 7040-681 Priced Configuration



2. Clause 1: Logical Data Base Design Related Items

2.1 Table Definitions

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

Appendix C contains the table definitions and the database load programs used to build the data base.

2.2 Database Organization

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

Physical space was allocated to Oracle9i Server on the server disks according to the details provided in Appendix C. The size of the space segments on each disk was calculated to provide even distribution of data across the disk subsystem.

2.3 Insert and/or Delete Operations

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

There were no restrictions on insert and/or delete operations to any of the tables. The space required for an additional five percent of the initial table cardinality was allocated to Oracle9i Server and priced as static space.

2.4 Horizontal or Vertical Partitioning

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

Partitioning was not used for any of the measurement reported in this full disclosure.

3. Clause 2: Transaction and Terminal Profiles Related Items

3.1 Verification for the Random Number Generator

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

The srand(), getpid() and gettimeofday() functions are used to produce unique random seeds for each driver. The drivers use these seeds to seed the srand(), srandom() and srand48() functions. Random numbers are produced using wrappers around the standard system random number generators.

The negative exponential distribution uses the following function to generate the distribution. This function has the property of producing a negative exponential curve with a specified average and a maximum value 4 times the average.

```
const double RANDOM_4_Z = 0.89837799236185
const double RANDOM_4_K = 0.97249842407114

double neg_exp_4(double average {
    return - average * (1/RANDOM_4_Z * log (1 - RANDOM_4_K * drand48()));
}
```

The random functions used by the driver system and the data base generation program were verified. The C_LAST column was queried to verify the random values produced by the database generation program. After a measurement, the HISTORY, ORDER, and ORDER_LINE tables were queried to verify the randomness of values generated by the driver. The rows were counted and grouped by customer and item numbers.

Here is an example of one SQL query used to verify the random number generation functions:

```
vcreate table TEMP (W_ID int, D_ID, C_LAST char(16), CNTR int);
vinsert into TEMP select C_W_ID, C_D_ID, C_LAST, COUNT(*) from CUSTOMER group by C_W_ID,
    C_D_ID, C_LAST;
vselect CNTR, COUNT(*) from TEMP group by CNTR order by 1;
```

3.2 Input/Output Screens

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

The screen layouts corresponds exactly to the layout corresponding in clauses 2.4.3, 2.5.3, 2.6.3, 2.7.3 and 2.8.3 of the TPC-C specifications.

3.3 Priced Terminal Features

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 emulated workstations, IBM RS/6000 Model 44P-170, are commercially available and support all of the requirements in Clause 2.2.2.4.

3.4 Presentation Managers

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

The RS/6000 Model 44P-170 workstations did not involve screen presentations, message bundling or local storage of TPC-C rows. All screen processing was handled by the client system. All data manipulation was handled by the server system.

3.5 Home and Remote Order-lines

The percentage of home and remote order-lines in the New-Order transactions must be disclosed.

Table 3-1 show the percentage of home and remote transactions that occurred during the measurement period for the New-Order transactions.

3.6 New-Order Rollback Transactions

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

Table 3-1 show the percentage of New-Order transactions that were rolled back due to an illegal item being entered.

3.7 Number of Items per Order

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

Table 3-1 show the average number of items ordered per New-Order transaction.

3.8 Home and Remote Payment Transactions

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

Table 3-1 show the percentage of home and remote transactions that occurred during the measurement period for the Payment transactions.

3.9 Non-Primary Key Transactions

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

Table 3-1 show the percentage of non-primary key accesses to the data base by the Payment and Order-Status transactions.

3.10 Skipped Delivery Transactions

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

Table 3-1 show the percentage of Delivery transactions missed due to a shortage of supply of rows in the NEW-ORDER table.

3.11 Mix of Transaction Types

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

Table 3-1 show the mix percentage for each of the transaction types executed by the SUT.

3.12 Queueing Mechanism of Delivery

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

The Delivery transaction was submitted using an RPC call to an IBM Websphere Application Server Enterprise Edition Version 3.0, Encina interface transaction manager (TM). Websphere returns an immediate response to the calling program and schedules the work to be performed. This allows the Delivery transaction to be submitted, obtain an interactive response and queue the actual data base transaction for deferred execution. Please see the application code in Appendix A for details.

Table 3-1 Numerical Quantities for Transaction and Terminal Profiles

	IBM eServer pSeries 690 Turbo Model 7040-681
New Order	
Percentage of Home order lines	99.0%
Percentage of Remote order lines	1.00%
Percentage of Rolled Back Transactions	1.00%
Average Number of Items per order	10
Payment	
Percentage of Home transactions	85.01%
Percentage of Remote transactions	14.99%
Non-Primary Key Access	
Percentage of Payment using C_LAST	60.00%
Percentage of Order-Status using C_LAST	59.95%
Delivery	
Delivery transactions skipped	0
Transaction Mix	
New-Order	44.93%
Payment	43.00%
Order-Status	4.01%
Delivery	4.01%
Stock-Level	4.01%

4. Clause 3: Transaction and System Properties

The results of the ACID test must be disclosed along with a description of how the ACID requirements were met.

All ACID tests were conducted according to specification.

4.1 Atomicity Requirements

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

4.1.1 Atomicity of Completed Transaction

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

The following steps were performed to verify the Atomicity of completed transactions.

1. The balance was retrieved from the CUSTOMER table for a random Customer, District and Warehouse giving BALANCE_1.
2. The Payment transaction was executed for the Customer, District and Warehouse used in step 1.
3. The balance was retrieved again for the Customer used in step 1 and step 2 giving BALANCE_2. It was verified that BALANCE_1 was greater than BALANCE_2 by AMT.

4.1.2 Atomicity of Aborted Transactions

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

The following steps were performed to verify the Atomicity of the aborted Payment transaction:

1. The Payment application code was changed to execute a rollback of the transaction instead of performing the commit.
2. Using the balance, BALANCE_2, from the CUSTOMER table retrieved for the completed transaction, the Payment transaction was executed for the Customer, District, and Warehouse used in step 1 of the section 4.1.1, using a payment amount (AMT) of 410.00. The transaction rolled back due to the change in the application code from step 1.
3. The balance was retrieved again for the Customer used for step 2 giving BALANCE_3. It was verified that BALANCE_2 was equal to BALANCE_3.

4.2 Consistency Requirements

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

Verify that the data base is initially consistent by verifying that it meets the consistency conditions defined in Clauses 3.3.2.1 to 3.3.2.4. Describe the steps used to do this in sufficient detail so that the steps are independently repeatable.

4.2.1 Consistency Condition 1

Entries in the WAREHOUSE and DISTRICT tables must satisfy the relationship:

$$\forall W_YTD = \text{sum}(D_YTD)$$

for each warehouse defined by (W_ID = D_W_ID)

4.2.2 Consistency Condition 2

Entries in the *DISTRICT*, *ORDER*, and *NEW-ORDER* tables must satisfy the relationship:

$$\forall D_NEXT_O_ID - 1 = \max(O_ID) = \max(NO_O_ID)$$

for each district defined by $(D_W_ID = O_W_ID = NO_W_ID)$ and $(D_ID = O_D_ID = NO_D_ID)$. This condition does not apply to the *NEW-ORDER* table for any districts which have no outstanding new orders.

4.2.3 Consistency Condition 3

Entries in the *New-Order* table must satisfy the relationship:

$$\forall \max(NO_O_ID) - \min(NO_O_ID) + 1 = [\text{number of rows in the New-Order table for this district}]$$

for each district defined by *NO_W_ID* and *NO_D_ID*. This condition does not apply to any districts which have no outstanding new orders.

4.2.4 Consistency Condition 4

Entries in the *ORDER* and *ORDER-LINE* tables must satisfy the relationship:

$$\forall \text{sum}(O_OL_CNT) = [\text{number of rows in the ORDER-LINE table for this district}]$$

for each district defined by $(O_W_ID = OL_W_ID)$ and $(O_D_ID = OL_D_ID)$.

4.2.5 Consistency Tests

Verify that the data base is initially consistent by verifying that it meets the consistency conditions defined in Clauses 3.3.2.1 to 3.3.2.4. Describe the steps used to do this in sufficient detail so that the steps are independently repeatable.

The consistency conditions defined in 4.2.1 through 4.2.4 were tested using a shell script to issue queries to the database. All queries showed that the data base was in a consistent state.

After executing transactions at full load for approximately sixty minutes the shell script was executed again. All queries show that the database was still in a consistent state.

4.3 Isolation Requirements

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

4.3.1 Isolation Test 1

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

1. An Order status transaction T0 was executed for a randomly selected customer, and the order returned was as recorded. Transaction T0 was committed.
2. A new-order transaction T1 was started for the same customer used in T0. T1 was stopped immediately prior to commit.
3. An order-status transaction T2 was started for the same customer used in T1. Transaction T2 completed and was committed without being blocked by T1. T2 returned the same order that T0 had returned.
4. T1 completed and was committed.
5. An order-status transaction T3 was started for the same customer used in T1. T3 returned the order inserted by T1.

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

4.3.2 Isolation Test 2

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

The following steps were performed to satisfy the test of isolation for Order-Status and a rolled back New-Order transactions:

1. An Order status transaction T0 was executed for a randomly selected customer, and the order returned was recorded. Transaction T0 was committed.
2. A new-order transaction T1 with an invalid item was started for the same customer used in T0. Transaction T1 was stopped prior to rollback.
3. An order-status transaction T2 was started for the same customer used in T1. T2 completed and was committed without being blocked by T1. Transaction T2 returned the same order that T0 had returned.
4. T1 was rollback.
5. An order-status transaction T3 was started for the same customer used in T1. T3 returned the same order that T0 returned.

4.3.3 Isolation Test 3

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

The following steps were performed to verify isolation of two New-Order transactions:

1. The D_NEXT_O_ID of a randomly selected district was retrieved.
2. A new-order transaction T1 was started for a randomly selected customer within the district used in step1. T1 was stopped immediately prior to commit.
3. Another new-order transaction was started for the same customer used in T1. Transaction T2 waited.
4. T1 completed. T2 completed and was committed.
5. The order number returned by T1 was the same as the D_NEXT_O_ID retrieved in step 1. The order number returned by T2 was one greater than the order number returned by T1.
6. The D_NEXT_O_ID of the same district was retrieved again. It had been incremented by two (it was one greater than the order number returned by T2).

4.3.4 Isolation Test 4

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

The following steps were performed to verify the isolation of two New-Order transactions after one is rolled back:

1. The D_NEXT_O_ID of a randomly selected district was retrieved.
2. A new-order transaction T1 with an invalid item was started for a randomly selected customer with the district used in step1. T1 was stopped immediately prior to rollback.
3. Another new-order transaction was started for the same customer used in T1. T2 waited.
4. T1 was allowed to rollback. T2 completed and was committed.
5. The order number returned by T2 was the same as the D_NEXT_O_ID retrieved in step 1.
6. The D_NEXT_O_ID of the same district was retrieved again. It had been incremented by one (it was one greater than the order number returned by T2).

4.3.5 Isolation Test 5

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

The following steps were performed to successfully conduct this test:

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

4.3.6 Isolation Test 6

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

The following steps were performed to successfully conduct this test:

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

4.3.7 Isolation Test 7

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

The following steps were performed to successfully conduct this test:

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

4.3.8 Isolation Test 8

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

The following steps were performed to successfully conduct this test:

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

4.3.9 Isolation Test 9

This test demonstrates isolation for phantom protection between an Order-Status and a New-Order transaction.

The following steps were performed to successfully conduct this test:

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

4.4 Durability Requirements

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

4.4.1 Permanent Unrecoverable Failure of any Single Durable Medium

Permanent irrecoverable failure of any single durable medium containing TPC-C data base tables or recovery log data.

Failure of Durable Medium containing recovery log data and Instantaneous Interruption and Memory Failure.

This test was conducted on a fully scaled database. The following steps were performed successfully.

1. The current count of the total number of orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving SUM_1.
2. A test was started and allowed to run for twelve minutes.
3. One of the disks containing the transaction log data was powered off. Since the log was on a raid disk, Oracle9i continued to process the transactions successfully.
4. The test continued for another 1 1/2 minutes.
5. The system was immediately shut down by switching the Emergency Power Off , thereby removing system

power.

6. The disk from step 3 was powered back on.
7. The system was powered back on and rebooted.
8. Step 1 is performed returning the value for SUM_2. It was verified that SUM_2 was equal to SUM_1 plus the completed New_Order transactions recorded by the RTE and that no entries existed for rolled-back transactions.
9. Consistency condition 3 was verified.

Failure of Durable Medium containing TPC-C data base tables.

The following steps were successfully performed to pass the Durability test of failure of a disk unit with data base tables:

1. The contents of a disk containing a TPCC table was backed up by copying it to another disk.
2. The current count of the total number of orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving SUM_1.
3. A scaled-down test was started and allowed to run until steady state.
4. The disk containing the TPCC table was powered off.
5. The run was stopped.
6. The disk from step 4 was powered back on and was restored from the backup copy in step 1.
7. Oracle9i was restarted and its transaction log was used to roll forward through the transactions that had completed since the run had started.
8. Step 2 was performed returning SUM_2. It was verified that SUM_2 was equal to SUM_1 plus the completed New_Order transactions recorded by the RTE and that no entries existed for rolled-back transactions.
9. Consistency condition 3 was verified.

Failure of Durable Fast Write Cache on SSA Adapter for Redo Logs

The following steps were successfully performed to pass the Durability test for failure of a durable medium that contains transient redo log transactions:

1. The SSA adapter for the Redo logs contains a cache that is powered by an onboard battery which will retain its contents if the adapter fails, or system power goes off. This test was performed in two parts:
 - A) failure of the adapter/system power
 - B) failure of the onboard battery
2. Test (A) was performed with the power-off test of the log above. After the system was powered off, the cache was removed from its current adapter and re-inserted into the system before rebooting.
3. Data on the cache was recovered successfully by meeting the requirements listed in the power-off test of the logs above.
4. Test (B). The current count of the total number of orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table, giving SUM_1.
5. Test (B) was conducted by inducing a battery failure during a test run. Once the system had reached steady state, the battery was failed using a toggle switch. The system recorded the failure, flushed out its vram contents and quit using the cache. Error notices were posted into the system error log.
6. The run continued without the cache.
7. Step 4 was performed returning SUM_2. It was verified that SUM_2 was equal to SUM_1 plus the completed New_Order transactions recorded by the RTE and that no entries existed for rolled-back transactions.
8. Consistency condition 3 was verified..

5. Clause 4: Scaling and Data Base Population Related Items

5.1 Cardinality of Tables

The cardinality (e.g., the number of rows) of each table, as it existed at the start of the benchmark run, must be disclosed.

Table 5-1 portrays the TPC Benchmark™ C defined tables and the number of rows for each table as they were built initially. Table 5-1 Initial Cardinality of Tables (eServer pseries 690 Turbo Model 7040-681).

Table Name	Number of Rows
Warehouse	34,000
District	340,000
Customer	1,020,000,000
History	1,020,000,000
Order	1,020,000,000
New Order	306,000,000
Order Line	10,200,000,000
Stock	3,400,000,000
Item	100,000

5.2 Distribution of Tables and Logs

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

The following table depicts the data base configuration of the system tested.

Table 5-2. IBM eServer pseries 690 Turbo Model 7040-681 Data Distribution Benchmark Configuration

SSA	HDISKS	LV'S	DISK SIZE
ssa0	10,14,15,19,28,53,63,65,84,85	lvstk1 lvstk99 lvstk197 lvstk295 lvstk393	18GB
ssa0	100,17,40,43,51,59,66,68,75,81	lvstk47 lvstk145 lvstk243 lvstk341 lvstk439	18GB
ssa0	11,23,24,26,31,35,37,9,90,98	lvstk93 lvstk191 lvstk289 lvstk387 lvstk485	18GB
ssa0	12,27,39,41,44,46,62,77,91,95	lvstk70 lvstk168 lvstk266 lvstk364 lvstk462	18GB
ssa0	13,18,22,47,54,71,78,83,92,97	lvordl150 lvordl164 lvordl178 lvordl192 lvordl206 lvordl220 lvordl234 lvordl248 lvordl262 lvordl276 lvordl10 lvordl24	18GB
ssa0	16,25,49,50,61,67,80,86	lvhist2 lvhist3 lvhist16	18GB
ssa0	20,32,34,36,58,88,89,99	lvcust18 lvcust71 lvcust124 lvcust177 lvcust230 lvcust283 lvcust336 lvcust389 lvcust442 lvcust495 lvtemp18	18GB
ssa0	27,39,41,44,46,62,77,91,95	lvstk70 lvstk168 lvstk266 lvstk364 lvstk462	18GB
ssa0	29	lvhist2 lvhist3 lvhist16 lvitems1	18GB
ssa0	30	lvhist11 lvord8 lvhist18	18GB
ssa0	33,45,48,55,56,60,64,72,79,8	lvstk24 lvstk122 lvstk220 lvstk318 lvstk416	18GB
ssa0	38,42,52,57,69,70,73,74,93,96	lvcust41 lvcust94 lvcust147 lvcust200 lvcust253 lvcust306 lvcust359 lvcust412 lvcust465 lvcust518 lvtemp41	18GB
ssa0	7	lvcust18 lvcust71 lvcust124 lvcust177 lvcust230 lvcust283 lvcust336 lvcust389 lvcust442 lvcust495 lvtemp18	18GB
ssa0	76	lvhist2 lvhist3 lvitems1 lvhist16	18GB
ssa1	108,115,128,155,196	lvcust42 lvcust95 lvcust148 lvcust201 lvcust254 lvcust307 lvcust360 lvcust413 lvcust466 lvcust519 lvtemp42	18GB
ssa1	109,133,138,140,160	lvstk94 lvstk192 lvstk290 lvstk388 lvstk486	18GB
ssa1	111,125,162,167,183	lvord4 lvordl154 lvordl168 lvordl182 lvordl196 lvordl210 lvordl224 lvordl238 lvordl252 lvordl266 lvordl280 lvordl14	18GB
ssa1	116,139,157,159,182	lvstk25 lvstk123 lvstk221 lvstk319 lvstk417	18GB
ssa1	118,119,124,179,181	lvstk2 lvstk100 lvstk198 lvstk296 lvstk394	18GB
ssa1	122,123,156,165,168,1976	lvnord2 lvord15 lvhist1	18GB
ssa1	130,158,163,193,195	lvstk48 lvstk146 lvstk244 lvstk342 lvstk440	18GB
ssa1	134,135,148,161,184	lvstk71 lvstk169 lvstk267 lvstk365 lvstk463	18GB
ssa1	145,153,166,194,197	lvcust19 lvcust72 lvcust125 lvcust178 lvcust231 lvcust284 lvcust337 lvcust390 lvcust443 lvcust496 lvtemp19	18GB
ssa2	198,203,210,227,240,255,261,262,282,285	lvi ord10 lvi1ord3 lvi2ord28 lvi2ord41 lvi2ord10	18GB
ssa2	199,202,211,226,237,245,246,247,252,290	lvstk5 lvstk103 lvstk201 lvstk299 lvstk397	18GB
ssa2	200,204,205,216,221,225,269,271,277,291	lvcust45 lvcust98 lvcust151 lvcust204 lvcust257 lvcust310 lvcust363 lvcust416 lvcust469 lvcust522 lvtemp45	18GB
ssa2	201,208,213,222,238,253,260,272,274,289	lvcust22 lvcust75 lvcust128 lvcust181 lvcust234 lvcust287 lvcust340 lvcust393 lvcust446 lvcust499 lvtemp22	18GB
ssa2	206,207,209,218,228,235,248,256,268,293	lvstk97 lvstk195 lvstk293 lvstk391 lvstk489	18GB
ssa2	212,214,219,231,234,243,257,258,264,276	lvstk74 lvstk172 lvstk270 lvstk368 lvstk466	18GB
ssa2	215,236,239,242,267,270,275,278,283,287	lvstk28 lvstk126 lvstk224 lvstk322 lvstk420	18GB
ssa2	217,241,249,250,263,266,279,281,284,286	lvroll1 lvroll5 lvroll9 lvroll13	18GB
ssa2	220,224,229,230,251,265,273,280,288,292	lvstk51 lvstk149 lvstk247 lvstk345 lvstk443	18GB
ssa3	1793,310,311,337,342,344,360,362,374,385	lvstk95 lvstk193 lvstk291 lvstk389 lvstk487	18GB
ssa3	2257,303,313,324,341,352,358,364,365,388	lvstk26 lvstk124 lvstk222 lvstk320 lvstk418	18GB
ssa3	294,304,315,318,349,351,366,370,380,386,887	lvhist11 lvord8 lvhist18	18GB
ssa3	295,299,346,347,359,367,372,376,378,384	lvstk72 lvstk170 lvstk268 lvstk366 lvstk464	18GB
ssa3	296,301,309,312,317,321,339,353,363,373	lvcust43 lvcust96 lvcust149 lvcust202 lvcust255 lvcust308 lvcust361 lvcust414 lvcust467 lvcust520 lvtemp43	18GB
ssa3	297,302,305,306,331,333,335,348,375,377	lvord12 lvordl166 lvordl180 lvordl194 lvordl208 lvordl222 lvordl236 lvordl250 lvordl264 lvordl278 lvordl112 lvordl26	18GB
ssa3	298,319,322,325,327,336,338,357,382,383	lvstk3 lvstk101 lvstk199 lvstk297 lvstk395	18GB
ssa3	300,307,330,350,354,355,356,368,387	lvcust20 lvcust73 lvcust126 lvcust179 lvcust232 lvcust285 lvcust338 lvcust391 lvcust444 lvcust497 lvtemp20	18GB
ssa3	308,314,316,320,323,328,340,343,345,371	lvstk49 lvstk147 lvstk245 lvstk343 lvstk441	18GB
ssa4	389,391,397,402,410,420,466,471,481,484	lvhist9 lvordl153 lvordl167 lvordl181 lvordl195 lvordl209 lvordl223 lvordl237 lvordl251 lvordl265 lvordl279 lvordl13	18GB
ssa4	390,399,403,407,409,419,432,438,439,479	lvhist10 lvhist19	18GB
ssa4	393,394,396,405,412,424,451,465,474,475	lvcust44 lvcust97 lvcust150 lvcust203 lvcust256 lvcust309 lvcust362 lvcust415 lvcust468 lvcust521 lvtemp44	18GB
ssa4	395,401,418,428,435,437,440,443,478,482	lvcust21 lvcust74 lvcust127 lvcust180 lvcust233 lvcust286	18GB

		lvcust339 lvcust392 lvcust445 lvcust498 lvtemp21	
ssa4	398 ,404 ,413 ,417 ,421 ,459 ,462 ,469 ,473 ,476	lvstk4 lvstk102 lvstk200 lvstk298 lvstk396	18GB
ssa4	400 ,423 ,426 ,427 ,430 ,441 ,444 ,447 ,455 ,468	lvstk96 lvstk194 lvstk292 lvstk390 lvstk488	18GB
ssa4	406 ,408 ,414 ,433 ,436 ,446 ,454 ,470 ,477 ,480	lvstk50 lvstk148 lvstk246 lvstk344 lvstk442	18GB
ssa4	411 ,416 ,422 ,431 ,445 ,448 ,450 ,452 ,458 ,463	lvstk27 lvstk125 lvstk223 lvstk321 lvstk419	18GB
ssa4	429 ,434 ,449 ,453 ,456 ,460 ,461 ,464 ,467 ,472	lvstk73 lvstk171 lvstk269 lvstk367 lvstk465	18GB
ssa5	489 ,491 ,493 ,499 ,501 ,502 ,505 ,511 ,551 ,576	lvhist20 lvordl152 lvordl165 lvordl179 lvordl193 lvordl207 lvordl221 lvordl235 lvordl249 lvordl263 lvordl277 lvordl11	18GB
ssa5	492	lvstk66 lvstk164 lvstk262 lvstk360 lvstk458	18GB
ssa5	495 ,516 ,536 ,537 ,571	lvstk89 lvstk187 lvstk285 lvstk383 lvstk481	18GB
ssa5	496 ,500 ,515 ,544 ,575	lvcust14 lvcust67 lvcust120 lvcust173 lvcust226 lvcust279 lvcust332 lvcust385 lvcust438 lvcust491 lvtemp14	18GB
ssa5	497 ,514 ,531 ,540 ,549	lvcust37 lvcust90 lvcust143 lvcust196 lvcust249 lvcust302 lvcust355 lvcust408 lvcust461 lvcust514 lvtemp37	18GB
ssa5	503 ,517 ,522 ,527 ,557	lvware3 lvhist6 lvhist12 lvordl1	18GB
ssa5	504 ,512 ,550 ,577 ,578	lvstk20 lvstk118 lvstk216 lvstk314 lvstk412	18GB
ssa5	507 ,519 ,547 ,573 ,574	lvstk43 lvstk141 lvstk239 lvstk337 lvstk435	18GB
ssa5	532 ,533 ,538 ,552	lvstk66 lvstk164 lvstk262 lvstk360 lvstk458	18GB
ssa6	581 ,596 ,601 ,602 ,622 ,623 ,627 ,632 ,658 ,662	lvcust11 lvcust64 lvcust117 lvcust170 lvcust223 lvcust276 lvcust329 lvcust382 lvcust435 lvcust488 lvtemp11	18GB
ssa6	582 ,587 ,600 ,619 ,625 ,629 ,634 ,637 ,655 ,661	lvstk63 lvstk161 lvstk259 lvstk357 lvstk455	18GB
ssa6	583 ,584 ,621 ,631 ,648 ,652 ,656 ,657 ,663 ,667	lvstk40 lvstk138 lvstk236 lvstk334 lvstk432	18GB
ssa6	585 ,592 ,603 ,604 ,605 ,606 ,610 ,626 ,633 ,660	lvcust34 lvcust87 lvcust140 lvcust193 lvcust246 lvcust299 lvcust352 lvcust405 lvcust458 lvcust511 lvtemp34	18GB
ssa6	586 ,591 ,599 ,608 ,609 ,612 ,615 ,617 ,618 ,630 ,649 ,650 ,670 ,671 ,672	lvord7 lvordl157 lvordl171 lvordl185 lvordl199 lvordl213 lvordl227 lvordl241 lvordl255 lvordl269 lvordl3 lvordl17	18GB
ssa6	588 ,616 ,635 ,636 ,638 ,639 ,643 ,664 ,665 ,675	lvstk17 lvstk115 lvstk213 lvstk311 lvstk409	18GB
ssa6	589 ,613 ,640 ,642 ,654	lvhist9 lvordl153 lvordl167 lvordl181 lvordl195 lvordl209 lvordl223 lvordl237 lvordl251 lvordl265 lvordl279 lvordl13	18GB
ssa6	590 ,611 ,620 ,641 ,645 ,653 ,659 ,668,673 ,676	lvi1ord2 lvi2ord27 lvi2ord40 lvi2ord9	18GB
ssa6	594 ,595 ,597 ,598 ,614 ,628 ,646 ,647 ,651 ,669	lvstk86 lvstk184 lvstk282 lvstk380 lvstk478	18GB
ssa7	677 ,690 ,699 ,701 ,705 ,713 ,716 ,717 ,735 ,753	lvcust48 lvcust101 lvcust154 lvcust207 lvcust260 lvcust313 lvcust366 lvcust419 lvcust472 lvcust525 lvtemp48	18GB
ssa7	678 ,691 ,692 ,693 ,698 ,703 ,730 ,733 ,739 ,751	lvi1ord13 lvi1ord6 lvi2ord31 lvi2ord44 lvi2ord13	18GB
ssa7	679 ,682 ,685 ,686 ,700 ,704 ,712 ,725 ,763 ,771	lvstk77 lvstk175 lvstk273 lvstk371 lvstk469	18GB
ssa7	680 ,681 ,696 ,710 ,714 ,723 ,727 ,741 ,746 ,754	lvstk31 lvstk129 lvstk227 lvstk325 lvstk423	18GB
ssa7	683 ,684 ,687 ,688 ,689 ,736 ,764 ,765 ,767 ,768	lvcust2 lvcust55 lvcust108 lvcust161 lvcust214 lvcust267 lvcust320 lvcust373 lvcust426 lvcust479 lvtemp2	18GB
ssa7	694 ,708 ,711 ,718 ,722 ,728 ,731 ,732,738 ,755	lvroll4 lvroll8 lvroll12 lvroll16	18GB
ssa7	697 ,709 ,719 ,720 ,721,740,743 ,750 ,752 ,757	lvstk8 lvstk106 lvstk204 lvstk302 lvstk400	18GB
ssa7	702 ,707 ,742 ,747 ,748 ,749 ,759 ,762 ,769 ,770	lvstk54 lvstk152 lvstk250 lvstk348 lvstk446	18GB
ssa7	715 ,724 ,734 ,737 ,744 ,745 ,756 ,758 ,761 ,772	lvcust25 lvcust78 lvcust131 lvcust184 lvcust237 lvcust290 lvcust343 lvcust396 lvcust449 lvcust502 lvtemp25	18GB
ssa8	983 , 2269	log(RAID5)	2x218.7GB
ssa9	774 , 1051	log(RAID5)	2x218.7GB
ssa10	783 ,798 ,799 ,801 ,816 ,829 ,837 ,855,871 ,873	lvstk68 lvstk166 lvstk264 lvstk362 lvstk460	18GB
ssa10	784 ,789 ,807 ,811 ,819 ,823 ,827,839,846 ,870	lvstk22 lvstk120 lvstk218 lvstk316 lvstk414	18GB
ssa10	785 ,794 ,800 ,802 ,810 ,813 ,814 ,820 ,821 ,836	lvstk45 lvstk143 lvstk241 lvstk339 lvstk437	18GB
ssa10	787 ,804 ,812 ,833 ,838 ,848 ,854 ,858 ,863 ,864	lvstk91 lvstk189 lvstk287 lvstk385 lvstk483	18GB
ssa10	790 ,809 ,817 ,830 ,831 ,850 ,865 ,874 ,875	lvcust39 lvcust92 lvcust145 lvcust198 lvcust251 lvcust304 lvcust357 lvcust410 lvcust463 lvcust516 lvtemp39	18GB
ssa10	796 ,797 ,805 ,808 ,840 ,861 ,862 ,866 ,867 ,872	lvware5 lvhist4 lvnord4 lvhist14	18GB
ssa10	806 ,815 ,818 ,832 ,834 ,847 ,851 ,856 ,860 ,869	lvordl148 lvordl162 lvordl176 lvordl190 lvordl204 lvordl218 lvordl232 lvordl246 lvordl260 lvordl274 lvordl8 lvordl22	18GB
ssa10	822 ,825 ,826 ,828 ,841 ,852 ,853 ,857 ,859 ,868	lvcust16 lvcust69 lvcust122 lvcust175 lvcust228 lvcust281 lvcust334 lvcust387 lvcust440 lvcust493 lvtemp16	18GB
ssa11	876 ,903 ,904,921 ,933,935 ,936 ,943 ,951 ,959	lvstk90 lvstk188 lvstk286 lvstk384 lvstk482	18GB
ssa11	878 ,879 ,880 ,883 ,897 ,929 ,954 ,955 ,966 ,967	lvstk44 lvstk142 lvstk240 lvstk338 lvstk436	18GB
ssa11	881 ,882 ,906 ,912 ,918 ,949 ,950 ,952 ,956 ,970	lvware4 lvhist5 lvnord3 lvhist13 lvordl1	18GB
ssa11	884 ,892 ,893 ,895 ,902,923 ,927 ,932 ,940 ,964	lvstk67 lvstk165 lvstk263 lvstk361 lvstk459	18GB
ssa11	885 ,891 ,901 ,908 ,930 ,931 ,934 ,939 ,969 ,971	lvcust15 lvcust68 lvcust121 lvcust174 lvcust227 lvcust280	18GB

		lvcust333 lvcust386 lvcust439 lvcust492 lvtemp15	
ssa11	887 ,888 ,890 ,894 ,898 ,913 ,917 ,920 ,925 ,928 ,938 ,944 ,945 ,958 ,968	lvhist17 lvordl147 lvordl161 lvordl175 lvordl189 lvordl203 lvordl217 lvordl231 lvordl245 lvordl259 lvordl273 lvordl7	18GB
ssa11	896 ,900 ,905 ,907 ,909 ,911 ,914 ,916 ,924 ,960	lvcust38 lvcust91 lvcust144 lvcust197 lvcust250 lvcust303 lvcust356 lvcust409 lvcust462 lvcust515 lvtemp38	18GB
ssa11	910 ,915 ,922 ,937 ,941,94 ,947 ,962 ,963 ,965	lvstk21 lvstk119 lvstk217 lvstk315 lvstk413	18GB
ssa11	919 ,926 ,953 ,957	lvord12 lvordl166 lvordl180 lvordl194 lvordl208 lvordl222 lvordl236 lvordl250 lvordl264 lvordl278 lvordl112 lvordl26	18GB
ssa12	1000 ,1002 ,1010 ,1011 ,1012 ,1013 ,1015 ,1025 ,1030 ,974	lvstk75 lvstk173 lvstk271 lvstk369 lvstk467	18GB
ssa12	1001 ,1003 ,1021 ,1029 ,1042,1043,1048 ,1062,973 ,978	lvstk98 lvstk196 lvstk294 lvstk392 lvstk490	18GB
ssa12	1005 ,1028 ,1033 ,1035 ,1039 ,1053 ,986 ,991 ,994 ,997	lvstk29 lvstk127 lvstk225 lvstk323 lvstk421	18GB
ssa12	1006 ,1008 ,1027 ,1049 ,1052 ,1067,979 ,981 ,988 ,989	lvstk6 lvstk104 lvstk202 lvstk300 lvstk398	18GB
ssa12	1007 ,1026 ,1034 ,1040 ,1041 ,1044 ,972 ,980 ,984 ,992	lvstk52 lvstk150 lvstk248 lvstk346 lvstk444	18GB
ssa12	1014 ,1016 ,1018 ,1054 ,1055 ,1056 ,1061 ,985 ,990 ,998	lvcust23 lvcust76 lvcust129 lvcust182 lvcust235 lvcust288 lvcust341 lvcust394 lvcust447 lvcust500 lvtemp23	18GB
ssa12	1017 ,1019 ,1022 ,1038 ,1183 ,2047 ,976 ,977 ,982 ,993	lviord11 lviord4 lvi2ord29 lvi2ord42 lvi2ord11	18GB
ssa12	1024 ,1031 ,1045 ,1047 ,1050 ,1057 ,1065 ,1820 ,2052 ,975	lvroll2 lvroll6 lvroll10 lvroll14	18GB
ssa12	1032 ,1036 ,1037 ,1064 ,1066 ,2255 ,773 ,987 ,995 ,996	lvcust46 lvcust99 lvcust152 lvcust205 lvcust258 lvcust311 lvcust364 lvcust417 lvcust470 lvcust523 lvtemp46	18GB
ssa13	1068 ,1070 ,1071 ,1105 ,1112 ,1125 ,1129 ,1138 ,1139 ,1155	lvcust26 lvcust79 lvcust132 lvcust185 lvcust238 lvcust291 lvcust344 lvcust397 lvcust450 lvcust503 lvtemp26	18GB
ssa13	1069 ,1085 ,1087 ,1090 ,1110 ,1111 ,1117 ,1118 ,1131 ,1135	lvstk32 lvstk130 lvstk228 lvstk326 lvstk424	18GB
ssa13	1072 ,1083 ,1091 ,1100 ,1109 ,1113 ,1115 ,1121 ,1126 ,1130	lvstk55 lvstk153 lvstk251 lvstk349 lvstk447	18GB
ssa13	1073 ,1078 ,1082 ,1098 ,1123 ,1136 ,1137 ,1146 ,1147 ,1162	lviord14 lviord7 lvi2ord32 lvi2ord1 lvi2ord14	18GB
ssa13	1074 ,1080 ,1099 ,1102 ,1103 ,1132 ,1133 ,1140 ,1145 ,1153	lvstk78 lvstk176 lvstk274 lvstk372 lvstk470	18GB
ssa13	1076 ,1077 ,1084 ,1101 ,1114 ,1127 ,1143 ,1154 ,1156 ,1158	lvi1cust1 lvi1cust4 lvi1cust7 lvi1cust10 lvi2cust3 lvi2cust6 lvi2cust9 lvi2cust12 lvi2cust15 lvi2cust18	18GB
ssa13	1079 ,1092 ,1097 ,1104 ,1107 ,1108 ,1116 ,1120 ,1142 ,1149	lvcust3 lvcust56 lvcust109 lvcust162 lvcust215 lvcust268 lvcust321 lvcust374 lvcust427 lvcust480 lvtemp3	18GB
ssa13	1081 ,1086 ,1088 ,1124 ,1141 ,1148 ,1150 ,1151 ,1160 ,1161	lvstk9 lvstk107 lvstk205 lvstk303 lvstk401	18GB
ssa13	1089 ,1093 ,1095 ,1096 ,1106 ,1119 ,1122 ,1128 ,1144 ,1157	lvcust49 lvcust102 lvcust155 lvcust208 lvcust261 lvcust314 lvcust367 lvcust420 lvcust473 lvcust526 lvtemp49	18GB
ssa14	1004 ,1165 ,1179 ,1191 ,1220 ,1221 ,1223 ,1240 ,1244 ,1249	lvcust1 lvcust54 lvcust107 lvcust160 lvcust213 lvcust266 lvcust319 lvcust372 lvcust425 lvcust478 lvtemp1	18GB
ssa14	1023 ,1169 ,1171 ,1172 ,1196 ,1212 ,1228 ,1237 ,1242 ,1251	lvstk7 lvstk105 lvstk203 lvstk301 lvstk399	18GB
ssa14	1164 ,1168 ,1176 ,1187 ,1189 ,1190 ,1202 ,1210 ,1217 ,1229 ,1418	lviord12 lviord5 lvi2ord30 lvi2ord43 lvi2ord12	18GB
ssa14	1166 ,1167 ,1199 ,1203 ,1224 ,1225 ,1239 ,1243 ,1253 ,1254	lvstk53 lvstk151 lvstk249 lvstk347 lvstk445	18GB
ssa14	1170 ,1182 ,1186 ,1200 ,1204 ,1208 ,1211 ,1234 ,1241 ,1256 ,1256	lvstk76 lvstk174 lvstk272 lvstk370 lvstk468	18GB
ssa14	1173 ,1209 ,1255	lvssystem1 lvroll3 lvroll7 lvroll11 lvroll15	18GB
ssa14	1174 ,1188 ,1201 ,1219 ,1233 ,1245 ,1246 ,1248 ,1257 ,775	lvcust24 lvcust77 lvcust130 lvcust183 lvcust236 lvcust289 lvcust342 lvcust395 lvcust448 lvcust501 lvtemp24	18GB
ssa14	1175 ,1185 ,1205 ,1227 ,1231,1238 ,1258	lvroll3 lvroll7 lvroll11 lvroll15 lvssystem1	18GB
ssa14	1177 ,1178 ,1184 ,1192 ,1193 ,1194 ,1214 ,1218 ,1226 ,1236	lvcust47 lvcust100 lvcust153 lvcust206 lvcust259 lvcust312 lvcust365 lvcust418 lvcust471 lvcust524 lvtemp47	18GB
ssa14	1180 ,1195 ,1197 ,1198 ,1216 ,1247 ,1250 ,1252 ,1259 ,2051	lvstk30 lvstk128 lvstk226 lvstk324 lvstk422	18GB
ssa15	1260 ,1287 ,1288 ,1299 ,1311 ,1315 ,1317 ,1318	lvcust6 lvcust59 lvcust112 lvcust165 lvcust218 lvcust271	18GB

	,1322 ,1324	lvcust324 lvcust377 lvcust430 lvcust483 lvtemp6	
ssa15	1261 ,1274 ,1278,1281 ,1305 ,1307 ,1312 ,1325 ,1328 ,2256	lvstk58 lvstk156 lvstk254 lvstk352 lvstk450	18GB
ssa15	1263 ,1277 ,1282 ,1283 ,1294 ,1308 ,1320 ,1349 ,1350 ,1351	lvcust29 lvcust82 lvcust135 lvcust188 lvcust241 lvcust294 lvcust347 lvcust400 lvcust453 lvcust506 lvtemp29	18GB
ssa15	1264 ,1265 ,1266 ,1295 ,1300 ,1314 ,1316 ,1332 ,1338 ,1345	lvstk12 lvstk110 lvstk208 lvstk306 lvstk404	18GB
ssa15	1267 ,1268 ,1269 ,1290 ,1296 ,1298 ,1331 ,1333 ,1336 ,1343	lvstk1 lvstk2 lvstk3 lvstk4 lvtools	18GB
ssa15	1270 ,1273 ,1301 ,1303 ,1313 ,1330 ,1342 ,1346 ,1347 ,1353	lvi1ord17 lvi2ord22 lvi2ord35 lvi2ord17 lvi2ord4	18GB
ssa15	1271 ,1284 ,1285 ,1286 ,1289 ,1306 ,1309 ,1310 ,1321 ,1323	lvstk81 lvstk179 lvstk277 lvstk375 lvstk473	18GB
ssa15	1272 ,1276 ,1276 ,1279 ,1280 ,1292 ,1293 ,1302 ,1304 ,1339	lvcust52 lvcust105 lvcust158 lvcust211 lvcust264 lvcust317 lvcust370 lvcust423 lvcust476 lvcust529 lvtemp52	18GB
ssa15	1275 ,1297 ,1327 ,1329 ,1334 ,1340 ,1341 ,1344 ,1348 ,1352	lvstk35 lvstk133 lvstk231 lvstk329 lvstk427	18GB
ssa16	1355 ,1363 ,1368 ,1400,1402 ,1404,1413 ,1417 ,1437 ,1440	lvstk36 lvstk134 lvstk232 lvstk330 lvstk428	18GB
ssa16	1356 ,1362 ,1396 ,1403 ,1406 ,1407 ,1416 ,1426 ,1442 ,1449	lvcust53 lvcust106 lvcust159 lvcust212 lvcust265 lvcust318 lvcust371 lvcust424 lvcust477 lvcust530 lvtemp53	18GB
ssa16	1357 ,1360 ,1371 ,1375 ,1389 ,1390 ,1398 ,1409 ,1410 ,1422	lvcust30 lvcust83 lvcust136 lvcust189 lvcust242 lvcust295 lvcust348 lvcust401 lvcust454 lvcust507 lvtemp30	18GB
ssa16	1358 ,1372 ,1374 ,1378 ,1380 ,1382 ,1384 ,1387 ,1408 ,1448	lvcust7 lvcust60 lvcust113 lvcust166 lvcust219 lvcust272 lvcust325 lvcust378 lvcust431 lvcust484 lvtemp7	18GB
ssa16	1361 ,1370 ,1373 ,1376 ,1383 ,1399 ,1412 ,1414 ,1415 ,1428	lvstk82 lvstk180 lvstk278 lvstk376 lvstk474	18GB
ssa16	1364 ,1369 ,1385 ,1393 ,1395 ,1401 ,1421 ,1432 ,1441 ,1447	lvi1ord18 lvi2ord23 lvi2ord36 lvi2ord5 lvi2ord18	18GB
ssa16	1365 ,1366 ,1419 ,1420 ,1423 ,1430,1435 ,1439 ,1443 ,1450	lvstk13 lvstk111 lvstk209 lvstk307 lvstk405	18GB
ssa16	1379 ,1386 ,1392 ,1405 ,1411 ,1427,1434 ,1436 ,1444 ,1446 ,1446	lvstk59 lvstk157 lvstk255 lvstk353 lvstk451	18GB
ssa17	1451 ,1461 ,1464 ,1465 ,1481 ,1506 ,1511 ,1522 ,1523 ,1540	lvi1ord15 lvi1ord8 lvi2ord33 lvi2ord2 lvi2ord15	18GB
ssa17	1452 ,1470 ,1472 ,1479 ,1489,1490 ,1498,1513,1520 ,1543	lvstk79 lvstk177 lvstk275 lvstk373 lvstk471	18GB
ssa17	1453 ,1454 ,1467 ,1478 ,1492 ,1496 ,1497 ,1507 ,1525	lvstk56 lvstk154 lvstk252 lvstk350 lvstk448	18GB
ssa17	1455 ,1458 ,1494 ,1499 ,1512,1526 ,1531 ,1538 ,1542 ,1546	lvstk33 lvstk131 lvstk229 lvstk327 lvstk425	18GB
ssa17	1456 ,1457 ,1460 ,1486 ,1487 ,1517 ,1518 ,1519 ,1521 ,1541	lvstk10 lvstk108 lvstk206 lvstk304 lvstk402	18GB
ssa17	1459 ,1468 ,1477 ,1509 ,1524 ,1530 ,1534 ,1537 ,1539	lvi1cust2 lvi1cust5 lvi1cust8 lvi2cust1 lvi2cust4 lvi2cust7 lvi2cust10 lvi2cust13 lvi2cust16 lvi2cust19	18GB
ssa17	1462 ,1474 ,1475 ,1495 ,1500 ,1504 ,1508 ,1529 ,1535 ,1545	lvcust50 lvcust103 lvcust156 lvcust209 lvcust262 lvcust315 lvcust368 lvcust421 lvcust474 lvcust527 lvtemp50	18GB
ssa17	1463 ,1469 ,1476 ,1484 ,1485 ,1503 ,1510 ,1515 ,1528 ,1532	lvcust27 lvcust80 lvcust133 lvcust186 lvcust239 lvcust292 lvcust345 lvcust398 lvcust451 lvcust504 lvtemp27	18GB
ssa17	1471 ,1473 ,1480 ,1483 ,1493 ,1501 ,1502 ,1527 ,1533 ,1544	lvcust4 lvcust57 lvcust110 lvcust163 lvcust216 lvcust269 lvcust322 lvcust375 lvcust428 lvcust481 lvtemp4	18GB
ssa18	776 , 777	log(RAID5)	2x218.7GB
ssa19	778 , 1235	log(RAID5)	2x218.7GB
ssa20	1548 ,1575 ,1587 ,1589 ,1590 ,1597 ,1612 ,1633 ,1637 ,1643	lvcust5 lvcust58 lvcust111 lvcust164 lvcust217 lvcust270 lvcust323 lvcust376 lvcust429 lvcust482 lvtemp5	18GB
ssa20	1549 ,1558 ,1571 ,1573 ,1583 ,1586 ,1638 ,1640 ,1641 ,1642	lvstk11 lvstk109 lvstk207 lvstk305 lvstk403	18GB
ssa20	1550 ,1551 ,1555 ,1563 ,1579 ,1588 ,1591 ,1594 ,1615 ,1621	lvcust28 lvcust81 lvcust134 lvcust187 lvcust240 lvcust293 lvcust346 lvcust399 lvcust452 lvcust505 lvtemp28	18GB
ssa20	1552 ,1570 ,1580 ,1598 ,1611 ,1625 ,1626 ,1628 ,1629 ,1636	lvstk80 lvstk178 lvstk276 lvstk374 lvstk472	18GB
ssa20	1553 ,1557 ,1562 ,1566 ,1567 ,1576 ,1617,1630 ,1634 ,1635	lvstk34 lvstk132 lvstk230 lvstk328 lvstk426	18GB

ssa20	1556 ,1565 ,1568 ,1600 ,1620 ,1622 ,1623 ,1627 ,1631 ,1632	lvi1cust3 lvi1cust6 lvi1cust9 lvi2cust2 lvi2cust5 lvi2cust8 lvi2cust11 lvi2cust14 lvi2cust17 lvi2cust20	18GB
ssa20	1559 ,1560 ,1564 ,1592 ,1593 ,1599 ,1602,1608 ,1609,1610	lvstk57 lvstk155 lvstk253 lvstk351 lvstk449	18GB
ssa20	1561 ,1569 ,1572 ,1584 ,1596 ,1601 ,1607 ,1618 ,1619 ,1639	lvi1ord16 lvi1ord9 lvi2ord34 lvi2ord3 lvi2ord16	18GB
ssa20	1574 ,1577 ,1582 ,1585 ,1603 ,1605 ,1613 ,1614 ,1616 ,1624	lvcust51 lvcust104 lvcust157 lvcust210 lvcust263 lvcust316 lvcust369 lvcust422 lvcust475 lvcust528 lvtemp51	18GB
ssa21	1644 ,1651 ,1681 ,1690 ,1700 ,1702 ,1718 ,1726 ,1728 ,1734	lvi1ord1 lvi2ord26 lvi2ord39 lvi2ord8 lvi2ord21	18GB
ssa21	1646 ,1671 ,1673 ,1676 ,1682 ,1684 ,1693 ,1704 ,1719 ,1723	lvstk85 lvstk183 lvstk281 lvstk379 lvstk477	18GB
ssa21	1647 ,1686 ,1689 ,1692 ,1712 ,1729 ,1730 ,1731 ,1732 ,1733	lvstk16 lvstk114 lvstk212 lvstk310 lvstk408	18GB
ssa21	1649 ,1650 ,1653 ,1655 ,1658 ,1679 ,1688 ,1708 ,1714 ,1715	lvstk39 lvstk137 lvstk235 lvstk333 lvstk431	18GB
ssa21	1654 ,1663 ,1667 ,1691 ,1694 ,1696 ,1710 ,1717 ,1722 ,1739	lvcust33 lvcust86 lvcust139 lvcust192 lvcust245 lvcust298 lvcust351 lvcust404 lvcust457 lvcust510 lvtemp33	18GB
ssa21	1656 ,1668 ,1674 ,1675 ,1683 ,1698 ,1698 ,1703 ,1705 ,1707 ,1721	lvcust10 lvcust63 lvcust116 lvcust169 lvcust222 lvcust275 lvcust328 lvcust381 lvcust434 lvcust487 lvtemp10	18GB
ssa21	1657 ,1662 ,1666 ,1670 ,1677 ,1678 ,1685 ,1687 ,1695 ,1697 ,1699 ,1701 ,1706 ,1711 ,1727 ,1645 ,1648 ,1652 ,1735 ,1736 ,1738	lvordl142 lvordl156 lvordl170 lvordl184 lvordl198 lvordl212 lvordl226 lvordl240 lvordl254 lvordl268 lvordl2 lvordl16	18GB
ssa21	1659 ,1660 ,1661 ,1664 ,1669 ,1672 ,1680 ,1720 ,1724 ,1725	lvstk62 lvstk160 lvstk258 lvstk356 lvstk454	18GB
ssa22	1740 ,1741 ,1744 ,1749 ,1767 ,1770,1771 ,1776 ,1789 ,1817	lvi1ord19 lvi2ord24 lvi2ord37 lvi2ord6 lvi2ord19	18GB
ssa22	1742 ,1743 ,1753 ,1754 ,1768 ,1769 ,1773 ,1783 ,1784 ,1795	lvstk14 lvstk112 lvstk210 lvstk308 lvstk406	18GB
ssa22	1745 ,1748 ,1751 ,1785 ,1790 ,1794 ,1807 ,1812 ,1831 ,2254	lvcust8 lvcust61 lvcust114 lvcust167 lvcust220 lvcust273 lvcust326 lvcust379 lvcust432 lvcust485 lvtemp8	18GB
ssa22	1746 ,1747 ,1755 ,1772 ,1775 ,1777 ,1806 ,1833 ,1835 ,2086 ,2137	lvstk5 lvstk6 lvstk7 lvstk8 lvstk9 lvstk10	18GB
ssa22	1750 ,1759 ,1761 ,1764 ,1765 ,1787,1801 ,1803,1826 ,1830	lvstk60 lvstk158 lvstk256 lvstk354 lvstk452	18GB
ssa22	1752 ,1757 ,1758 ,1779 ,1782 ,1791,1797 ,1810 ,1832 ,369	lvstk37 lvstk135 lvstk233 lvstk331 lvstk429	18GB
ssa22	1760 ,1762 ,1786 ,1792 ,1809 ,1813 ,1819 ,1822 ,1823 ,2271	lvord7_n lvordl146 lvordl160 lvordl174 lvordl188 lvordl202 lvordl216 lvordl230 lvordl244 lvordl258 lvordl272 lvordl6	18GB
ssa22	1766 ,1798 ,1800 ,1802 ,1805 ,1814 ,1821 ,1824 ,1827 ,1829	lvstk83 lvstk181 lvstk279 lvstk377 lvstk475	18GB
ssa22	1774 ,1780 ,1781 ,1799 ,1804 ,1811 ,1815 ,1818 ,1825 ,1828	lvcust31 lvcust84 lvcust137 lvcust190 lvcust243 lvcust296 lvcust349 lvcust402 lvcust455 lvcust508 lvtemp31	18GB
ssa23	1836 ,1839 ,1851 ,1859 ,1862 ,1866 ,1888 ,1891 ,1903,1924	lvstk15 lvstk113 lvstk211 lvstk309 lvstk407	18GB
ssa23	1837 ,1842 ,1847 ,1854 ,1880 ,1892 ,1904 ,1908 ,1914 ,1922	lvcust32 lvcust85 lvcust138 lvcust191 lvcust244 lvcust297 lvcust350 lvcust403 lvcust456 lvcust509 lvtemp32	18GB
ssa23	1838 ,1863 ,1864 ,1870 ,1874 ,1879 ,1896 ,1919 ,1920 ,1927	lvcust9 lvcust62 lvcust115 lvcust168 lvcust221 lvcust274 lvcust327 lvcust380 lvcust433 lvcust486 lvtemp9	18GB
ssa23	1840 ,1841 ,1844 ,1852 ,1858 ,1867 ,1875 ,1876 ,1884 ,1895 ,1907 ,1909 ,1912 ,1917 ,1923	lvord14 lvordl155 lvordl169 lvordl183 lvordl197 lvordl211 lvordl225 lvordl239 lvordl253 lvordl267 lvordl1 lvordl15	18GB
ssa23	1845 ,1868 ,1873 ,1887 ,1899 ,1901 ,1906 ,1916 ,1926 ,1928	lvstk61 lvstk159 lvstk257 lvstk355 lvstk453	18GB
ssa23	1846 ,1855 ,1872 ,1877 ,1883 ,1893 ,1894 ,1900 ,1911 ,1913	lvstk38 lvstk136 lvstk234 lvstk332 lvstk430	18GB
ssa23	1849 ,1853 ,1861 ,1865 ,1871 ,1878 ,1889 ,1905 ,1910 ,1921	lvi1ord20 lvi2ord25 lvi2ord38 lvi2ord7 lvi2ord20	18GB
ssa23	1850 ,1860 ,1881 ,1885 ,1915 ,1918 ,1925 ,1929 ,1930 ,1931	lvstk84 lvstk182 lvstk280 lvstk378 lvstk476	18GB
ssa23	1856 ,1857 ,1890 ,1897 ,1902	lvord4 lvordl154 lvordl168 lvordl182 lvordl196 lvordl210 lvordl224 lvordl238 lvordl252 lvordl266 lvordl280 lvordl14	18GB
ssa24	1935 ,1959 ,1973 ,197 ,1994	lvstk92 lvstk190 lvstk288 lvstk386 lvstk484	18GB
ssa24	1938 ,1945 ,1946 ,1971 ,1972	lvcust17 lvcust70 lvcust123 lvcust176 lvcust229 lvcust282	18GB

		lvcust335 lvcust388 lvcust441 lvcust494 lvtemp17	
ssa24	,1951 ,1981 ,1996 ,2007 ,2020	lvstk69 lvstk167 lvstk265 lvstk363 lvstk461	18GB
ssa24	,1958 ,1966 ,1975 ,1998 ,2004	lvstk23 lvstk121 lvstk219 lvstk317 lvstk415	18GB
ssa24	,1974 ,1997 ,2005 ,2023 ,2027 ,1963 ,1965 ,1982 ,1984 ,2002	lvcust40 lvcust93 lvcust146 lvcust199 lvcust252 lvcust305 lvcust358 lvcust411 lvcust464 lvcust517 lvtemp40	18GB
ssa24	,1976 ,1999 ,2008 ,2012,2014	lvord10 lvord1163 lvord1177 lvord1191 lvord1205 lvord1219 lvord1233 lvord1247 lvord1261 lvord1275 lvord19 lvord123	18GB
ssa24	1987 ,1992 ,2000 ,2024 ,2025	lvstk46 lvstk144 lvstk242 lvstk340 lvstk438	18GB
ssa24	1989 ,2013 ,2018 ,2021 ,2026	lvware6 lvhist15 lvord13	18GB
ssa25	1207 ,2037 ,2084 ,2262 ,779	lvstk88 lvstk186 lvstk284 lvstk382 lvstk480	18GB
ssa25	2028 ,2029 ,2030 ,2069 ,2280	lvstk42 lvstk140 lvstk238 lvstk336 lvstk434	18GB
ssa25	2033 ,2045 ,2066 ,2067 ,2094	lvord10 lvord1163 lvord1177 lvord1191 lvord1205 lvord1219 lvord1233 lvord1247 lvord1261 lvord1275 lvord19 lvord123	18GB
ssa25	2034 ,2048 ,2062 ,2074 ,2082	lvstk19 lvstk117 lvstk215 lvstk313 lvstk411	18GB
ssa25	2035 ,2112 ,2113 ,2118 ,2119	lvcust36 lvcust89 lvcust142 lvcust195 lvcust248 lvcust301 lvcust354 lvcust407 lvcust460 lvcust513 lvtemp36	18GB
ssa25	2039 ,2040 ,2058 ,2108	lvstk65 lvstk163 lvstk261 lvstk359 lvstk457	18GB
ssa25	2046 ,2055 ,2070 ,2090 ,2116	lvware2 lvhist7 lvord5 lvord2 lvord9	18GB
ssa25	2080 ,2087 ,2100 ,2122	lvord1145 lvord1159 lvord1173 lvord1187 lvord1201 lvord1215 lvord1229 lvord1243 lvord1257 lvord1271 lvord15 lvord119	18GB
ssa25	2083 ,2085 ,2107 ,2117 ,2272 ,1788 ,2064 ,2076 ,2105 ,2115	lvcust13 lvcust66 lvcust119 lvcust172 lvcust225 lvcust278 lvcust331 lvcust384 lvcust437 lvcust490 lvtemp13	18GB
ssa26	2124 ,2125 ,2130 ,2142 ,2150 ,2155 ,2156 ,2157 ,2201 ,2203	lvstk87 lvstk185 lvstk283 lvstk381 lvstk479	18GB
ssa26	2126 ,2133 ,2139 ,2141 ,2186 ,2200 ,2202,2206 ,2209 ,2214	lvstk18 lvstk116 lvstk214 lvstk312 lvstk410	18GB
ssa26	2129 ,2131 ,2135 ,2136 ,2159 ,2164 ,2172 ,2188 ,2215 ,2216	lvstk41 lvstk139 lvstk237 lvstk335 lvstk433	18GB
ssa26	2132 ,2140 ,2148 ,2149 ,2178 ,2182 ,2211 ,2217 ,2218 ,2219	lvstk64 lvstk162 lvstk260 lvstk358 lvstk456	18GB
ssa26	2134 ,2146 ,2166 ,2174 ,2175 ,2179 ,2187 ,2189 ,2198 ,2199	lvware1 lvhist8 lvord3	18GB
ssa26	2137 ,2138 ,2143 ,2154 ,2162 ,2163 ,2165 ,2170 ,2171 ,2173 ,2180 ,2190 ,2192 ,2197 ,2204	lvord1 lvord1144 lvord1158 lvord1172 lvord1186 lvord1200 lvord1214 lvord1228 lvord1242 lvord1256 lvord1270 lvord14	18GB
ssa26	2144 ,2153 ,2160 ,2167 ,2168 ,2169 ,2176 ,2183 ,2184 ,2210	lvcust35 lvcust88 lvcust141 lvcust194 lvcust247 lvcust300 lvcust353 lvcust406 lvcust459 lvcust512 lvtemp35	18GB
ssa26	2145 ,2185 ,2191 ,2205,2212	lvord1148 lvord1162 lvord1176 lvord1190 lvord1204 lvord1218 lvord1232 lvord1246 lvord1260 lvord1274 lvord18 lvord122	18GB
ssa26	2147 ,2151 ,2152 ,2158 ,2161 ,2177 ,2193 ,2195 ,2196 ,2208	lvcust12 lvcust65 lvcust118 lvcust171 lvcust224 lvcust277 lvcust330 lvcust383 lvcust436 lvcust489 lvtemp12	18GB
ssa28	1359 ,1547	log(RAID5)	2x218.7GB
ssa29	1796 , 2253	log(RAID5)	2x218.7GB
ssa30	1933 ,1934 ,1978 ,1991 ,2015	lvware6 lvhist15 lvord13	18GB
ssa30	1939 ,1940 ,1941 ,1947 ,1957	lvstk23 lvstk121 lvstk219 lvstk317 lvstk415	18GB
ssa30	1942 ,1948 ,1949 ,1964 ,1985	lvstk46 lvstk144 lvstk242 lvstk340 lvstk438	18GB
ssa30	1944 ,1956 ,1961 ,1977 ,1986	lvord10 lvord1163 lvord1177 lvord1191 lvord1205 lvord1219 lvord1233 lvord1247 lvord1261 lvord1275 lvord19 lvord123	18GB
ssa30	1952 ,1955 ,1960 ,1962 ,2009	lvstk69 lvstk167 lvstk265 lvstk363 lvstk461	18GB
ssa30	1953 ,1954 ,2003 ,2006 ,2016	lvstk92 lvstk190 lvstk288 lvstk386 lvstk484	18GB
ssa30	1968 ,1969 ,1980 ,1983 ,2010	lvcust17 lvcust70 lvcust123 lvcust176 lvcust229 lvcust282 lvcust335 lvcust388 lvcust441 lvcust494 lvtemp17	18GB
ssa31	1206 ,2044 ,2092 ,2096 ,2097	lvord1145 lvord1159 lvord1173 lvord1187 lvord1201 lvord1215 lvord1229 lvord1243 lvord1257 lvord1271 lvord15 lvord119	18GB
ssa31	1215 ,2050 ,2104 ,2109 ,2110	lvstk19 lvstk117 lvstk215 lvstk313 lvstk411	18GB
ssa31	1763 ,2036 ,2053 ,2065 ,2075	lvstk65 lvstk163 lvstk261 lvstk359 lvstk457	18GB
ssa31	2032 ,2056 ,2057 ,2073 ,2114	lvstk88 lvstk186 lvstk284 lvstk382 lvstk480	18GB
ssa31	2038 ,2054 ,2078 ,2088 ,2120	lvord7_n lvord1146 lvord1160 lvord1174 lvord1188 lvord1202 lvord1216 lvord1230 lvord1244 lvord1258 lvord1272 lvord16	18GB
ssa31	2043 ,2049 ,2089 ,2091 ,2095	lvware2 lvhist7 lvord5 lvord2 lvord9	18GB
ssa31	2063 ,2068 ,2072 ,2101 ,2102	lvcust36 lvcust89 lvcust142 lvcust195 lvcust248 lvcust301 lvcust354 lvcust407 lvcust460 lvcust513 lvtemp36	18GB

ssa31	2077 ,2081 ,2093 ,2111 ,2121	lvstk42 lvstk140 lvstk238 lvstk336 lvstk434	18GB
ssa32	485 ,554 ,562 ,565 ,570	lvware3 lvhist6 lvhist12 lvord1	18GB
ssa32	486 ,524 ,546 ,561 ,568	lvstk66 lvstk164 lvstk262 lvstk360 lvstk458	18GB
ssa32	487 ,508 ,509 ,528 ,529	lvstk89 lvstk187 lvstk285 lvstk383 lvstk481	18GB
ssa32	488 ,526 ,545 ,569	lvhist20 lvordl152 lvordl165 lvordl179 lvordl193 lvordl207 lvordl221 lvordl235 lvordl249 lvordl263 lvordl277 lvordl11	18GB
ssa32	498 ,548 ,553 ,556,572	lvordl150 lvordl164 lvordl178 lvordl192 lvordl206 lvordl220 lvordl234 lvordl248 lvordl262 lvordl276 lvordl110 lvordl24	18GB
ssa32	506 ,510 ,518 ,521 ,539	lvcust14 lvcust67 lvcust120 lvcust173 lvcust226 lvcust279 lvcust332 lvcust385 lvcust438 lvcust491 lvtemp14	18GB
ssa32	513 ,530 ,543 ,566 ,567	lvstk20 lvstk118 lvstk216 lvstk314 lvstk412	18GB
ssa32	520 ,523 ,525 ,535 ,580	lvcust37 lvcust90 lvcust143 lvcust196 lvcust249 lvcust302 lvcust355 lvcust408 lvcust461 lvcust514 lvtemp37	18GB
ssa32	541 ,542 ,558 ,560 ,563	lvstk43 lvstk141 lvstk239 lvstk337 lvstk435	18GB
ssa34	147 ,142 ,154 ,188 ,187	stkv2	18GB
ssa34	169 ,150	stkv25	18GB
ssa34	104 ,185 ,106	stkv25	18GB
ssa34	107 ,189 ,103 ,173	stkv48	18GB
ssa34	176	ustvg19	18GB
ssa34	105 ,141 ,191 ,131 ,132	custvg4	18GB
ssa34	170 ,192 ,144 ,164 ,113	ordlvg	18GB
ssa34	175 ,172 ,126 ,174 ,127	tblvg8	18GB
ssa34	114	stkv48	18GB
ssa34	180 ,143 ,152 ,151 ,146	stkv71	18GB
ssa34	120 ,121 ,137 ,110 ,178	stkv94	18GB
ssa34	190 ,112 ,102 ,186	custvg19	18GB

5.3 Data Base Model Implemented

A statement must be provided that describes the data base model implemented by the DBMS used.

The database manager used for this testing was Oracle9i Enterprise Edition Release 2, v9.2.0.1 for AIX 5L v5.2 from Oracle Corp. Oracle9i Enterprise Edition Release 2 is a relational DBMS.

5.4 Partitions/Replications Mapping

The mapping of data base partitions/replications must be explicitly described.

IBM did not implement horizontal or vertical partitioning for this TPC-C test.

5.5 60 day space calculations

IBM eServer pseries 690 Turbo Model 7040-681

TPM		427760							
Warehouses		34000							
SEGMENT	TYPE	TSPACE	BLOCKS	BLOCK SIZE	KBYTES	FIVE_PCT(BYTES)	DAILY_GROW	TOTAL	
CUSTOMER	TABLE	CUST	255000003	4096	1020000012	51000001	0	1071000012.60	
DISTRICT	TABLE	WARE	170004	4096	680016	34001	0	714016.80	
HISTORY	TABLE	HIST	14936120	4096	59744480	0	12026494	71770973.54	
ICUSTOMER	INDEX	ICUST1	6346481	4096	25385924	1269296	0	26655220.20	
ICUSTOMER2	INDEX	ICUST2	12091256	4096	48365024	2418251	0	50783275.20	
IDISTRICT	INDEX	WARE	1642	4096	6568	328	0	6896.40	
IITEM	INDEX	ITEMS	411	4096	1644	82	0	1726.20	
INORD	INDEX	NORD	1250765	4096	5003060	250153	0	5253213.00	
IORDERS	INDEX	IORD1	6503076	4096	26012304	1300615	0	27312919.20	
IORDERS2	INDEX	IORD2	10114447	4096	40457788	2022889	0	42480677.40	
IORDL	INDEX	ORDL	44624561	16384	713992976	0	143725946	857718922.08	
ISTOCK	INDEX	ISTR	18921379	4096	75685516	3784276	0	79469791.80	
IITEM	TABLE	ITEMS	3031	4096	12124	606	0	12730.20	
IWAREHOUSE	INDEX	WARE	131	4096	524	26	0	550.20	
ORDERS	TABLE	ORD	10327908	4096	41311632	0	8315983	49627614.92	
ROLL_SEG	SYS	ROLL	3275776	16384	52412416	0	0	52412416.00	
STOCK	TABLE	STOCKS	309090911	4096	1236363644	61818182	0	1298181826.20	
SYSTEM	SYS	SYSTEM	1064704	4096	4258816	0	0	4258816.00	
WAREHOUSE	TABLE	WARE	17010	4096	68040	3402	0	71442.00	
Total			693,739,616		3,349,762,508	123,902,109	164,068,423	3,637,733,039.93	
Bytes									
Dynamic space in Kbytes			815,049,088						
Static space in Kbytes			2,658,615,529						
Free space in Kbytes			164,068,423						
Daily growth in Kbytes			164,068,423						
Daily spread			0 Oracle may be configured such that daily spread is 0						
60-day space (in Kbytes)			12,502,720,881						
60-day (GB)			11,923.52						
			new_order		84470030				
Log block size			512 Redo blocks written		2126045396				
Log blocks/tpmC			25.17 Number of log blocks used in one tpmC						
8-hour log (GB)			2,464.23						
Disk	Disk	SUT	SUT	Priced	Priced	Space usage (GB)			
Type	Formatted Capacity	# of disks	Capacity(GB)	# of disks	Capacity(GB)	DB	34667.16		
						RAID	2624.40		
						OS	33.94		
9.1	8672	96	813.00	0	0.00	Total Space	37325.49		
18.2	17376	1947	33038.16	2043	34667.16				
RAID(6-36.4GB)	223948.8	12	2,624.40	12	2,624.40				

6. Clause 5: Performance Metrics and Response Time Related Items

6.1 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 6-1 list the response times and the ninetieth percentiles for each of the transaction types for the measured system.

6.2 Keying and Think Times

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

Table 6-1 list the TPC-C keying and think times for the measured system.

Table 6-1. IBM eServer pseries 690 Turbo Model 7040-681 Response, Think and Keying Times

Response Times	New Order	Payment	Order Status	Delivery (int./def.)	Stock Level	Menus
90 %	0.9	0.85	0.88	0.19/0.19	0.83	0.01
Average	0.45	0.42	0.44	0.11/0.11	0.4	0.01
Maximum	97.3	96.37	48.76	48.16/3.73	96.28	96.14
			Think Times			
Minimum	0.01	0.01	0.01	0.01	0.01	N/A
Average	12.02	12.02	10.01	5.02	5.02	N/A
Maximum	120.22	120.21	100.20	50.21	50.20	N/A
			Keying Times			
Minimum	18.00	3.00	2.00	2.00	2.00	N/A
Average	18.01	3.01	2.01	2.01	2.01	N/A
Maximum	18.11	3.11	2.10	2.10	2.11	N/A

6.3 Response Time Frequency Distribution

Response time frequency distribution curves must be reported for each transaction type.

Figure 6-3-1. IBM eServer pseries 690 Turbo Model 7040-681 New-Order Response Time Distribution

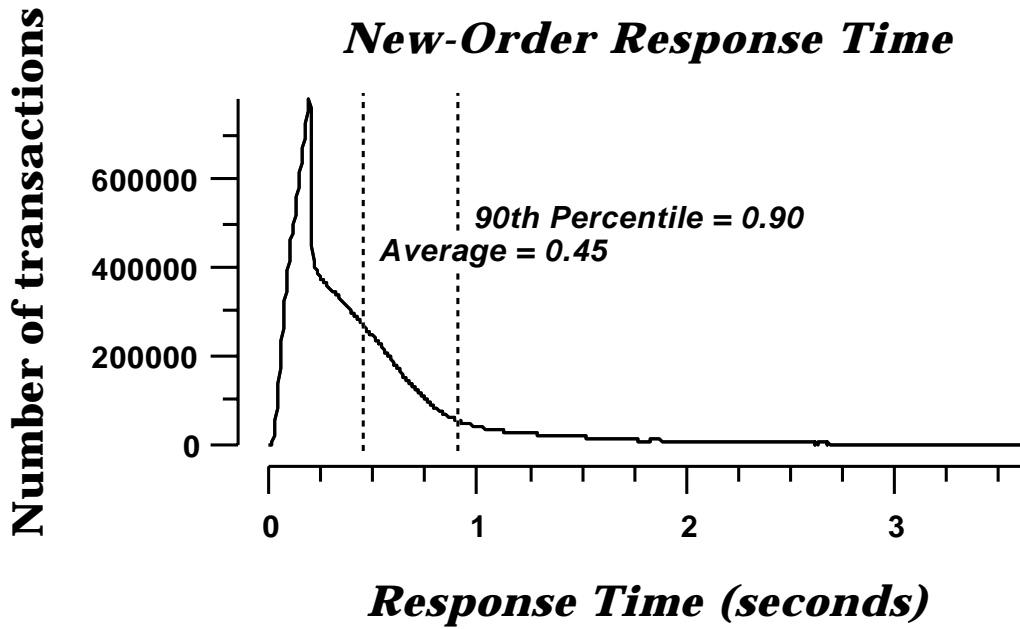


Figure 6-3-2. IBM eServer pseries 690 Turbo Model 7040-681 Payment Response Time Distribution

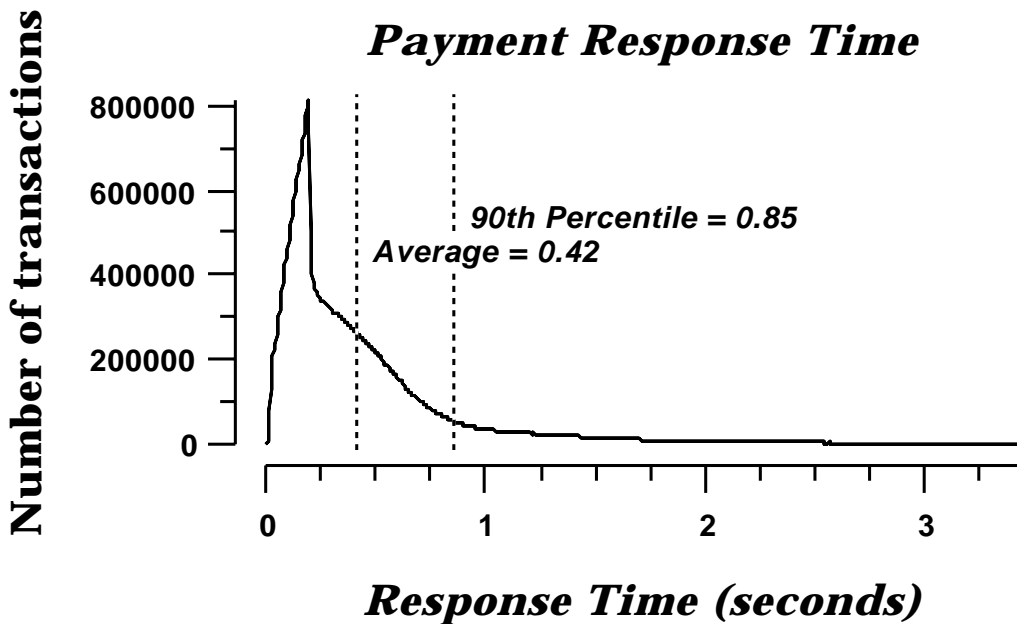


Figure 6-3-3. IBM eServer pseries 690 Turbo Model 7040-681 Order-Status Response Time Distribution

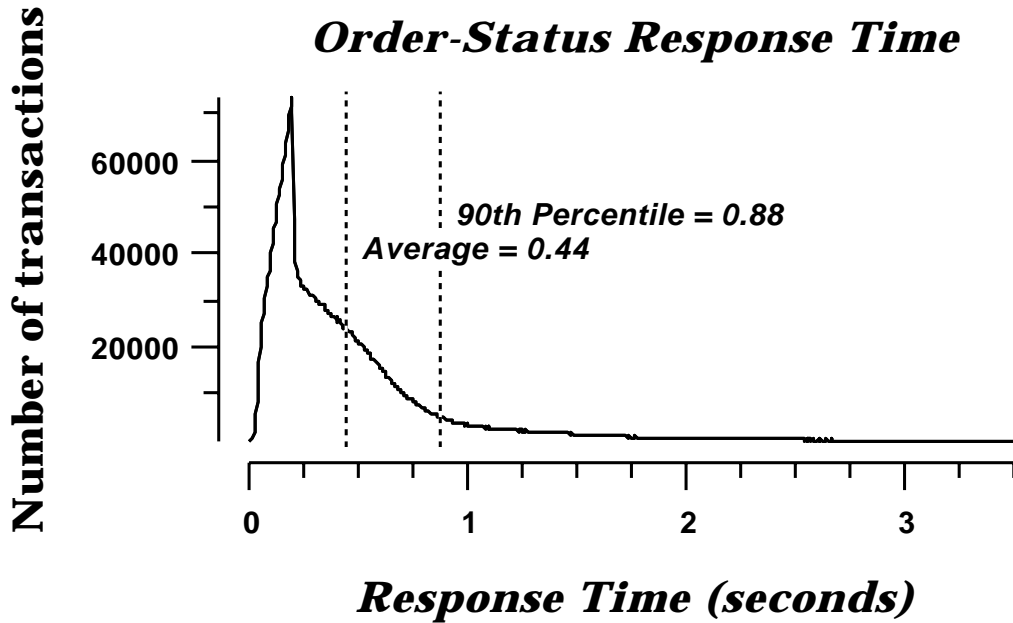


Figure 6-3-4. IBM eServer pseries 690 Turbo Model 7040-681 Delivery (Interactive) Response Time Distribution

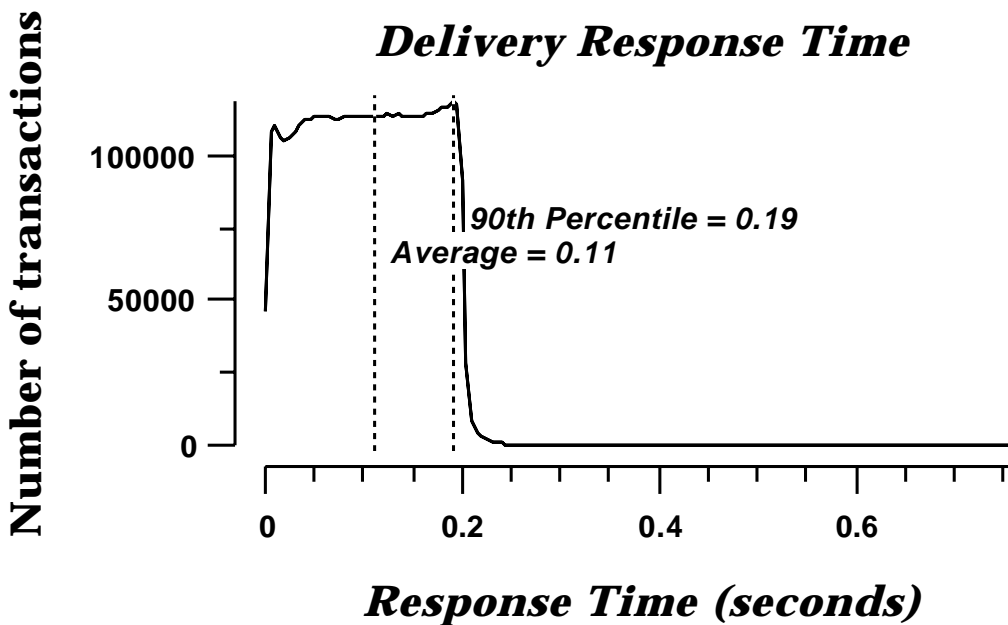


Figure 6-3-5. IBM eServer pseries 690 Turbo Model 7040-681 Delivery (Deferred) Response Time Distribution

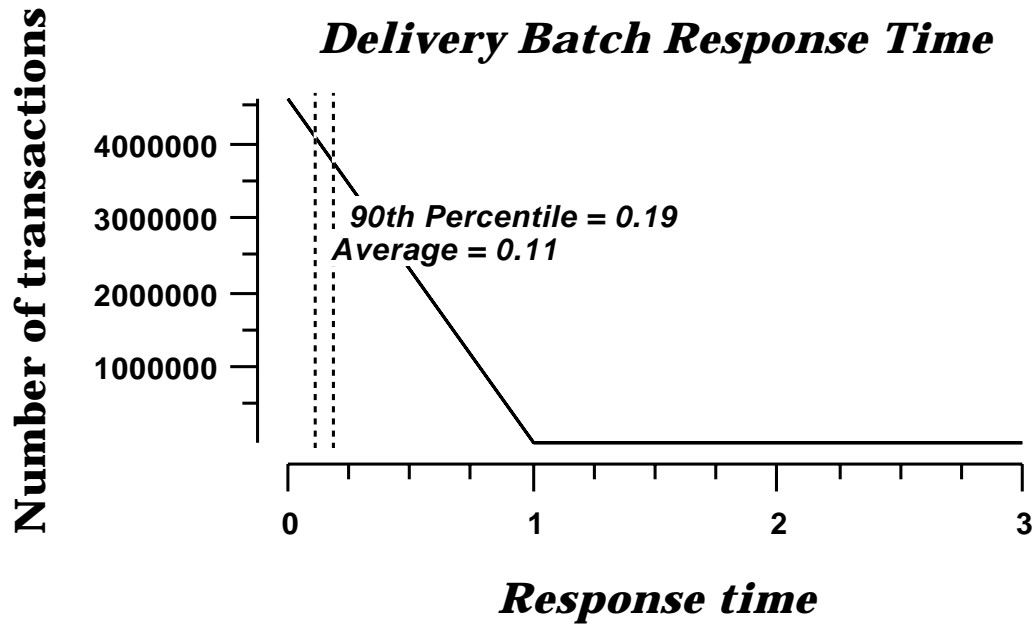
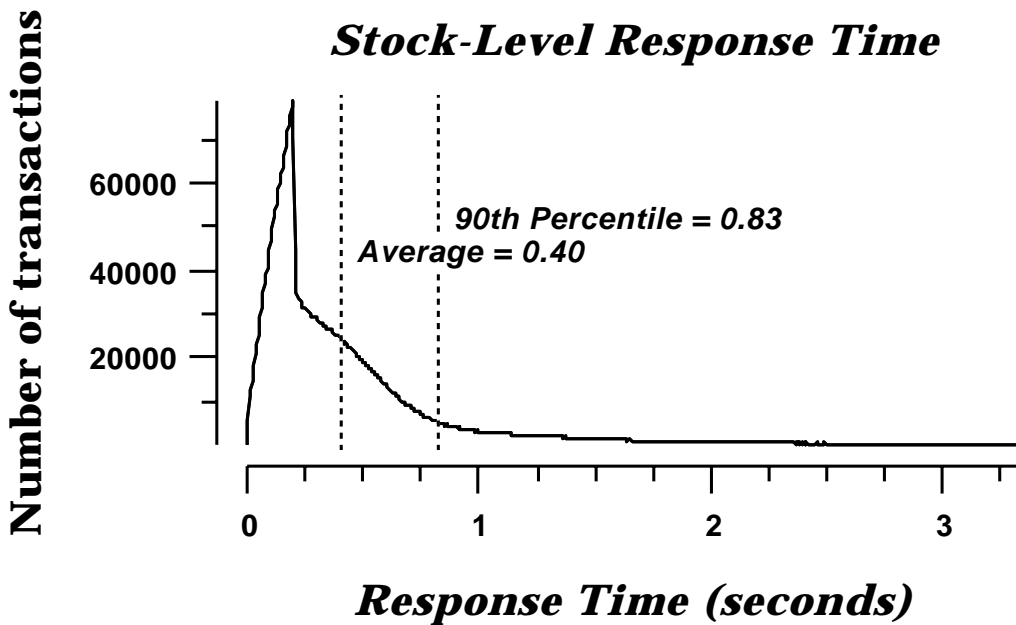


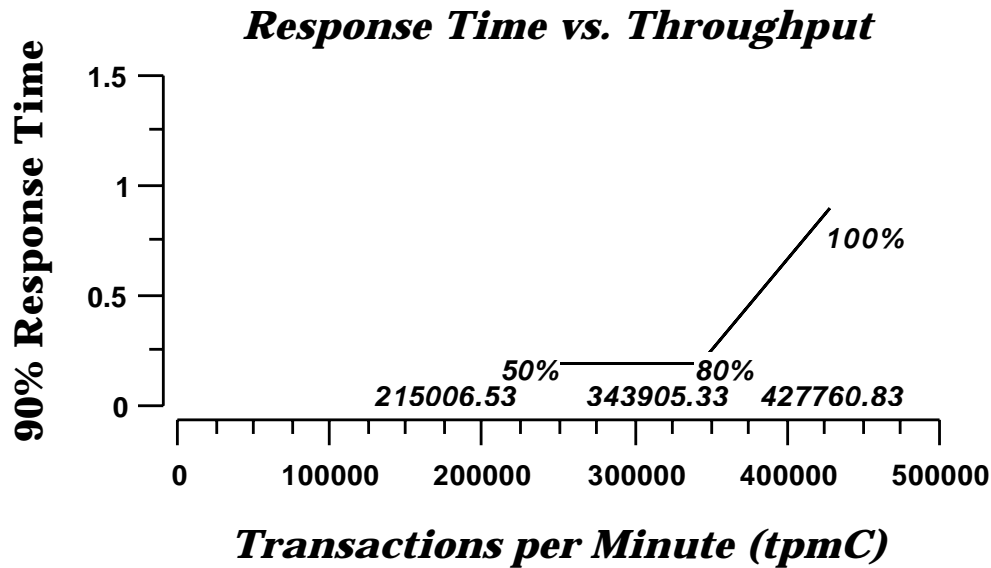
Figure 6-3-6. IBM eServer pseries 690 Turbo Model 7040-681 Stock Level Response Time Distribution



6.4 Performance Curve for Response Time versus Throughput

The performance curve for response times versus throughput must be reported for the New-Order transaction.

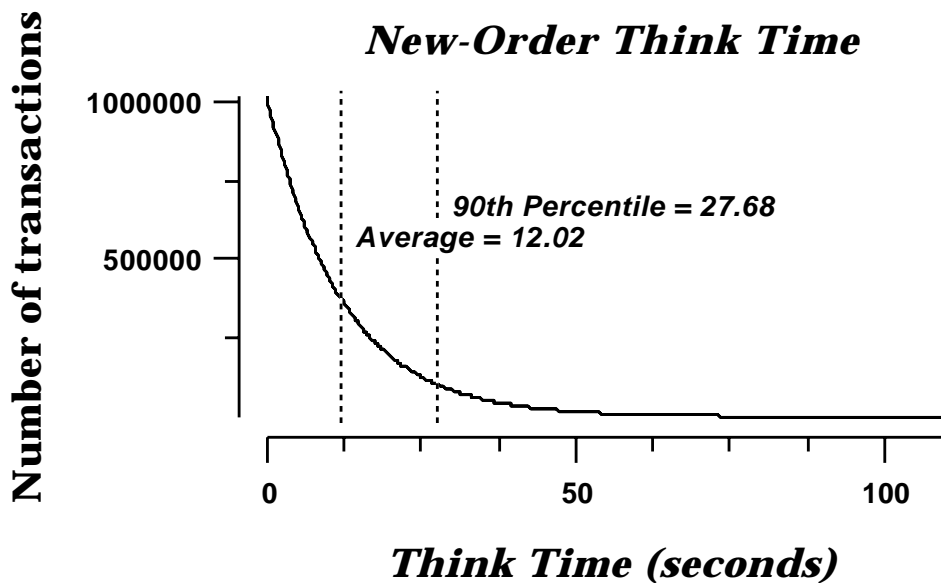
Figure 6-4-1. IBM eServer pseries 690 Turbo Model 7040-681 New-Order Response Time vs. Throughput



6.5 Think Time Frequency Distribution

A graph of the think time frequency distribution must be reported for the New-Order transaction.

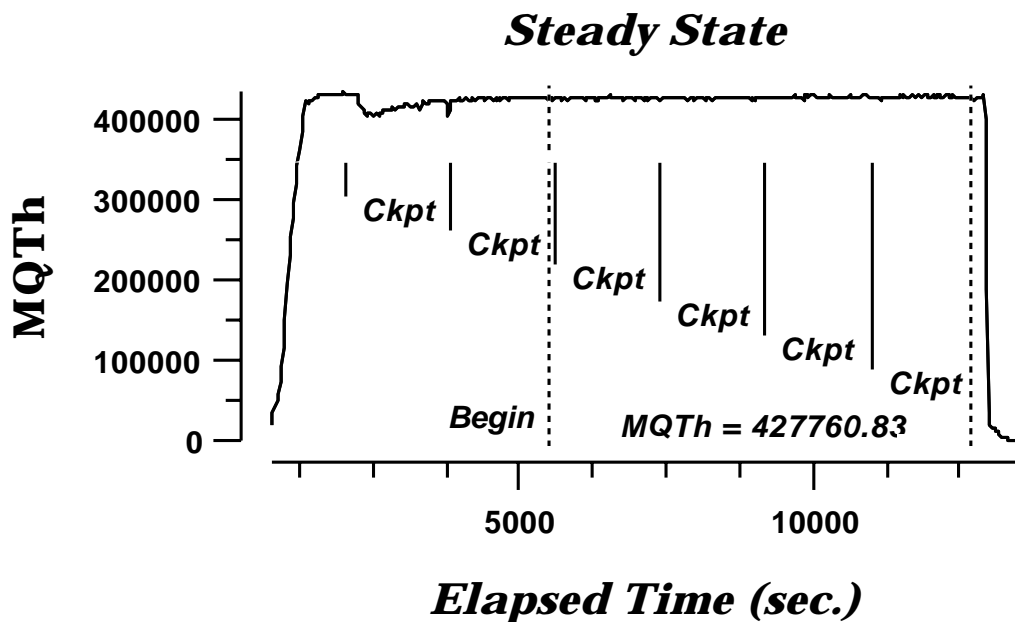
Figure 6-5-1. IBM eServer pseries 690 Turbo Model 7040-681 New-Order Think Time Distribution



6.6 Throughput versus Elapsed Time

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

Figure 6-6-1. New-Order Throughput vs. Elapsed Time



6.7 Steady State Determination

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

All the emulated users were allowed to logon and do transactions. The time stamping interval was set to start after several minutes of rampup. Refer to the Numerical Quantities Summary pages for the rampup time. Figure 6.6.1 New-Order throughput versus Elapsed Time graph shows that the system was in steady state at the beginning of the Measurement Interval.

6.8 Work Performed During Steady State

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

6.8.1 Transaction Flow

For each of the TPC Benchmark™C transaction types, the following steps are executed:

IBM Websphere Application Server Enterprise Edition Version 3.0, Encina interface, was used as a transaction manager (TM). Each transaction was divided into three programs. The front end program handled all screen I/O , a database client program which connected to the database and served as a Websphere Server (a back end program), and a database server program which handled all database operations at the SUT. Both the front end and back end programs ran on the client system. The front end program communicates with the database client program through DCE RPCs. The database client program communicates with the Server system over Ethernet using SQL*Net calls. Besides calling Websphere Application Server Enterprise Edition Encina initialization code during startup, all other functions are transparent to the application code. Encina routes the transaction and balances the load according to the options defined in the configuration file in appendix B.2, The transaction flow is described below.

- Each client machine is a node in an Encina Cell.

- Two servers are configured in each node: one processes the delivery transactions and one all other transaction.
- The delivery server is configured with one processing agent with 2 server manager DCE threads, and 2 background threads to process deferred deliveries. Each background thread has one connection to the database.
- The other server is configured with 50 processing agents. Each processing agent has 1 server manager DCE threads. Each thread has one connection to the database.
- When the Encina clients are started, they connect to Encina cell.
- When terminals are started, each terminal connects to the Encina client. The client spawns a thread for each connection to handle that connection. The thread executes the 'process_terminal' routine. The process_terminal displays the TPC-C transaction menu on the user terminal.
- The TPC-C user chooses the transaction type and proceeds to fill the screen fields required for transaction.
- The process_terminal accepts all values entered by the user and transmits those values to one of the TPC_C backend programs. The transaction is performed through a DCE RPC. There is an interface for each TPC-C transaction type and each TPC-C backend program exports one or more of these interfaces. (The delivery servers export only the delivery interface, the other servers export the other four interfaces, and only those). Encina transparently routes the RPC to one of the servers exporting the corresponding interface.
- A TPC-C backend server program receives an RPC and proceeds to execute all database operations related to the request. All information entered on the user terminal is contained in the RPC.
- Once the transaction is committed, the server program fills in the output parameters. The RPC is then sent back to the client program.
- When the RPC returns to the client, the process_terminal routine writes the transaction out on the user terminal.

6.8.2 Database Transaction

All database operations are performed by the TPC-C back-end programs. The process is described below:

Using SQL*Net calls, the TPC-C back-end program interacts with Oracle9i Server to perform SQL data manipulations such as update, select, delete and insert, as required by the transaction. After all database operations are performed for a transaction, the transaction is committed.

Oracle9i Server proceeds to update the database as follows:

When Oracle9i Server changes a database table with an update, insert, or delete operation, the change is initially made in memory, not on disk. When there is not enough space in the memory buffer to read in or write additional data pages, Oracle9i Server will make space by flushing some modified pages to disk. Modified pages are also written to disk when a checkpoint occurs. Before a change is made to the database, it is first recorded in the transaction log. This ensures that the database can be recovered completely in the event of a failure. Using the transaction log, transactions that started but did not complete prior to a failure can be undone, and transactions recorded as complete in the transaction log but not yet written to disk can be redone.

6.8.3 Checkpoints

A checkpoint is the process of writing all modified data pages to disk. The TPC-C benchmark was setup to automatically checkpoint every 30 minutes. Two checkpoint occur during the rampup period, with four other occurring during the measurement interval. The checkpoints duration were 29min 50sec, 25min 43 sec, 25min 52 sec, 25min 52 sec, 25min 42 sec and 25min 36.

6.9 Measurement Interval

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

be included.

A two hour Measurement Interval was used. Further, the measurement interval is a multiple of the checkpoint interval. This demonstrates that a different measurement interval over the eight hour period would yield similar throughput results. No connections were lost during the run.

7. Clause 6: SUT, Driver, and Communication Definition Related Items

7.1 RTE Availability

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

IBM used an internally developed RTE for these tests. Appendix D contains the scripts used in the testing.

7.2 Functionality and Performance of Emulated Components

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

In the benchmark configuration the Remote Terminal Emulator (RTE) communicates with the client system over Ethernet. The 28 eServer Pseries 6F1 emulates a network of 340,000 RS/6000 Model 44P-170 workstations. The communications mechanism used in the benchmark and priced configurations are the same. In the benchmark configuration a separate Ethernet LAN was used to connect two driver systems to a eServer Pseries Model 6F1 client system. In other words, there were several separate LAN segments between every one driver to one client. Each LAN segment in the priced configuration is used to connect 868 workstations.

7.3 Network Bandwidth

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

The Ethernet used in the LAN complies with the IEEE 802.3 standard and has a bandwidth of 10 Megabits per second Half Duplex. Each LAN segment in the IBM eServer pseries 690 Turbo Model 7040-681 configuration connected 868 workstations.

7.4 Operator Intervention

If the configuration requires operator intervention, the mechanism and the frequency of this intervention must be disclosed.

No operator intervention is required to sustain the reported throughput during the eight hour period.

8. Clause 7: Pricing Related Items

8.1 Hardware and Programs Used

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

The detailed list of all hardware and programs for the priced configuration is listed in the pricing sheets (please refer to Section 8.2 for details) for each system reported. The prices for all products and features that are provided by IBM are available the same day as product or feature availability.

Pricing for IBM Websphere Application Server Enterprise Edition Version 3.0 is for Txseries License only.

8.2 Three Year Cost of System Configuration

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

The price sheets for the eServer pseries 690 Turbo Model 7040-681 are contained on the following page. The basis for the discounts used are:

3-year Term Maintenance Contract Discount

This discount is available for customers who sign a 3-year maintenance agreement on the hardware. A discount of 3% is available for customers when they sign a 3-year maintenance agreement.

Scope Incentive

A 2% discount is applied for the ServiceElect contract that combines hardware maintenance with one or more services, which in this pricing report the selected service is SupportLine.

3-year Maintenance Prepay Discount

This is a discount for prepayment of maintenance costs. A discount of 10.36% is available for this configuration based on payment for three years maintenance at the time of purchase. This discount is applied to the balance after the 3-year term maintenance contract discount and Scope discounts are applied.

Hardware Volume Revenue Discount

The IBM-supplied hardware and software is discounted by 61% from the list price, based on the dollar value of this configuration only

8.3 Availability Dates

The committed delivery date for general availability (availability date) of products used in the price calculations 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 products are generally available today except the following:

Product	Availability Date
64GB memory	May 31, 2003

8.4 Statement of tpmC and Price/Performance

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

System	tpmC	3-year System Cost	\$/tpmC	Availability Date
IBM eServer pseries 690 Turbo Model 7040-681	\$7,591,208	\$7,591,208	\$17.75	All HS/SW available as shown in Section 8.3

IBM eServer pseries 690 Turbo Model 7040-681 Three Year System Price Configuration

Description	Part No.	Source	Unit Price	Qty	Ext Price	Maint Price
Server Hardware						
IBM eServer pSeries 690 (CD, cables, clock card)	7040-681	1	18,764	1	18,764	20,760
128MB (4x32) L3 Cache, 433MHz	4139	1	25,000	4	100,000	0
64GB H-Memory (inward facing)	4488	1	241,000	4	964,000	0
64GB H-Memory (outward facing)	4489	1	241,000	4	964,000	0
1.3GHz 8-way Processor MCM (1 thru 4 books)	5244	1	275,000	4	1,100,000	403,200
Std CEC Fan Power cblgrp, BPC(F&R), cable	6161	1	600	1	600	0
Standard CEC DC/DC Converter Assembly	6170	1	4,700	2	9,400	0
Power Cable Group, 1 thru 4 processor modules	6181	1	720	1	720	0
Additional DC/DC Converter Assembly, (DCA)	6189	1	4,200	3	12,600	0
Capacitor book (for 2 MCMs)	6198	1	1,800	2	3,600	0
Service Processor + Dual Rio Loop (GX), I/O loop adapter	6404	1	15,000	1	15,000	0
Regatta-H Planar (Mayfly)	6565	1	50,000	1	50,000	0
Media Drawer, 1U, Op Pnl/Dskt/4 SCSI Media Bays	8692	1	1,700	1	1,700	0
I/O Drawer	7040-61D	1	3,980	4	15,920	16,320
SCSI cable - B&C to Media Drawer	2122	1	275	1	275	0
Remote I/O Cable, 500MHz, 0.5m (RIO)	3145	1	310	4	1,240	0
Remote I/O Cable, 500MHz, 2m (RIO)	3149	1	350	8	2,800	0
18.2 GB Ultra3 10K RPM 1" U3 Module	3157	1	1,260	3	3,780	0
10/100 Mbps Ethernet adapter II	4962	1	275	5	1,375	0
Cable Grp, 4xUPIC/2xRIO, BPM (IO#1 thru 4) to B&C(pos 4609)	6121	1	1,600	1	1,600	0
I/O Drawer DC/DC Converter assembly, (DCA)	6172	1	4,000	8	32,000	0
Power Cable, B&C to Media Drawer	6179	1	300	1	300	0
Ultra SCSI SE Adapter	6206	1	395	1	395	0
Advanced Serial RAID Adapter	6230	1	3,000	34	102,000	0
32 MB Fast_Write Cache option card	6235	1	575	6	3,450	0
B&C Planar, 10 PCI Slots, 2x integrated SCSI	6563	1	8,000	8	64,000	0
Ultra3 SCSI 4-Pack	6564	1	500	16	8,000	0
Rack (Front primary and secondary racks)	7040-61R	1	9,050	1	9,050	3,960
Bak Door, Slim Line (primary or secondary)	6074	1	750	2	1,500	0
Bulk Power Regulator (BPR)	6186	1	4,000	4	16,000	0
Bulk Power Controller (BPC), 4 Fans + 3 DCAs	6187	1	1,900	2	3,800	0
Bulk Power Distribution (BPD), 10 DCAs	6188	1	3,500	4	14,000	0
Integrated Battery Backup and Redundant + cable	6200	1	4,600	4	18,400	0
Line Cord, 60A, 14', IEC309 Plug, Chargeable	8678	1	1,000	2	2,000	0
Regatta-H Bulk Power Module (2x BPA, EPO PNL)	8690	1	5,000	1	5,000	0
Expansion Rack 24", 42 EIA	8691	1	8,000	1	8,000	0
System Console + cable	7316	1	4,400	1	4,400	0
Display, White P76/P77 Color Monitor	3627	1	675	1	675	0
				Subtotal	3,560,344	444,240
Storage						
System Rack Model T42 (6089,6098)	7014-T42	1	4,150	14	58,100	12,432
Additional Power Distribution Unit	6171	1	1,000	41	41,000	0
SSA Disk Subsystem	7133-D40	1	12,750	136	1,734,000	600,576
50/60 Hz AC, 300 VDC Power Supply	8022	1	2,000	136	272,000	0
Raven Black Drawer Cover	8031	1	250	136	34,000	0
Dummy Drive	8399	1	30	2	60	0
10K/18.2 GB Advanced Disk Drive Module	8518	1	4,500	2,043	9,193,500	0
10K/36.4 GB Advanced Disk Drive Module	8536	1	5,900	96	566,400	0
SSA Cables (8801, 8802)	8801	1	125	136	17,000	0
				Subtotal	11,916,060	613,008
Server Software						
AIX 5.1 (media only)	5765-E61	1	50	1	50	0
IBM Websphere Application Server	D5ALCLL	1	26,687	32	853,984	158,144
IBM C for AIX V5	D5A1ELL	1	559	1	559	104
Oracle9i Database Enterprise Edition Release 2, v9.2.0.1 for AIX 5L v5.2, Processor 3 year term for 32 processors, Unlimited Users		2	640,000	1	640,000	0
Oracle Database Server Support Package for 3 years		2				6,000
				Subtotal	1,494,593	164,248
Client Hardware and Software						
WorkGroup server pSeries 620 Model 6F1	7025-6F1	1	7,997	14	111,958	67,200
32x Speed CD-ROM Drive	2624	1	375	14	5,250	0
18.2 GB 10K RPM Ultra SCSI Enhanced Disk	3153	1	1,260	14	17,640	0
P76/p77 Color Monitor, captured cable + keyboard	3627	1	675	1	675	0
16 slot memory expansion card	4075	1	1,500	14	21,000	0
1024MB(2x512MB)200PIN SDRAM DIMM Memory	4100	1	3,072	56	172,032	0
512MB (2x256MB) SDRAM DIMM Memory	4119	1	1,536	56	86,016	0
4-port 10/100 Mbps Ethernet adapter	4961	1	1,500	84	126,000	0
Proc card 4xRS64 IV/600MHz/4MB cache	5214	1	24,500	14	343,000	60,480
IPL Disk Mounting HW cables, terminator	6555	1	595	14	8,330	0
AIX 4. Unlimited Users	5765-C34	1	50	14	700	0
				Subtotal	892,601	127,680
Third Party Hardware						
8-port 10/100 Ethernet Switch	FS108NA	3	85	4	340	0
				Subtotal	340	0
Oracle Mandatory E-Business Discount on License and Support		2	(129,200)	1	(129,200)	
				Discounts	(10,908,034)	(584,672)
				Total	6,826,704	764,504
Audited by: Brad Askins, Info Sizing						
Pricing Sources:					Three-Year Cost of Ownership	\$7,591,208
1=IBM: Bill Casey, eServer pSeries Offering Manager, wrcasey@us.ibm.com, 512-838-1422					tpm C	427,760.83
2=Oracle: Hervé Lejeune, herve.lejeune@oracle.com, 650-506-1894					\$/tpm C	\$17.75
3=NETGEAR						

9. Clause 9: Audit Related Items

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

The auditor's attestation letter is included in this section of this report



Benchmark Sponsor:	Carole Gottleib	Ravi Bhatnagar
	IBM eServer Performance	Oracle Corporation
	11501 Burnet Road	500 Oracle Parkway
	Austin, Texas 78758	Redwood Shores, CA 94065

December 24, 2002

I remotely verified the TPC Benchmark™ C performance of the following Client/Server configuration:

Platform:	IBM eServer pSeries 690 Turbo Model 7040-681
Operating system:	AIX 5L V5.2
Database Manager:	Oracle9i Enterprise Edition Release 2v9.2.0.1 for AIX 5L v5.2
Transaction Manager:	Websphere Application Server Enterprise Edition 3.0

The results were:

CPU's Speed	Memory	Disks	NewOrder 90% Response Time	tpmC
Server: IBM eServer pSeries 660 Model 7040-681				
32 x POWER4 (1300 MHz)	512 GB Main (128MB L3 Cache per MCM)	2046 x 18.2 GB 96 x 36.4 GB	0.90 Seconds	427,760.83
Fourteen (14) Clients: eServer pSeries Model 6F1 (Specification for each)				
4 x RS64IV (600 MHz)	6 GB Main (4 MB L2 Cache per processor)	1 x 18.2 GB	n/a	n/a

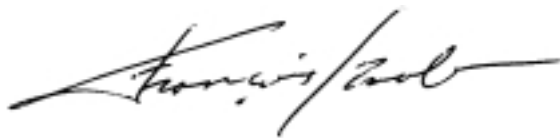
In my opinion, these performance results were produced in compliance with the TPC's requirements for the benchmark. The following verification items were given special attention:

- The database records were the proper size
- The database was properly scaled and populated
- The required ACID properties were met
- The transactions were correctly implemented
- Input data was generated according to the specified percentages
- The transaction cycle times included the required keying and think times
- The reported response times were correctly measured.
- All 90% response times were under the specified maximums
- At least 90% of all delivery transactions met the 80 Second completion time limit
- The reported measurement interval was 120 minutes (7200 seconds)
- The reported measurement interval was representative of steady state conditions
- Four checkpoints were taken during the reported measurement interval
- The 60 day storage requirement was correctly computed
- The system pricing was verified for major components and maintenance

Additional Audit Notes:

The measured system included 96 IBM 10K rpm drives (9.1GB disks) that were substituted by 96 IBM 10K rpm drives (18.2 GB disks) in the priced configuration. Based on the specifications of these disks and on additional performance data collected on these disks, it is my opinion that this substitution does not have a material effect on the reported performance.

Respectfully Yours,



François Raab, President



Bradley J. Askins, Auditor

Appendix A: TPC-C Application

Source

A.1 Client/Terminal Handler code

callora.c

```
/*
 * callora.c
 *
 * $Revision: 1.3 $
 * $Date: 1999/05/06 21:28:29 $
 * $Log: callora.c,v $
 *
 * $TALog: callora.c,v $
 * Revision 1.3 1999/05/06 21:28:29 oz
 * - Removed all the .. from the includes
 * - Added -I.. to the makefiles instead
 * - Moved all the thread related code and connection
 *   selection to serverMon.c
 *
 * - get_db_ready now does not take the number of connections
 * - Export create_connection() and clean_connection(void *)
 * - All the transactions take a connection pointer as a first param
 * [from r1.2 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5]
 *
 * Revision 1.2 1999/04/19 20:14:48 oz
 * - Moved all the simulated code to server.c
 * - Created nulldb.c for compilation with no DB
 * [from r1.1 by delta oz-24331-TPCC-move-sim-code-to-common-file,r1.1]
 *
 * Revision 1.1 1999/04/19 14:37:27 oz
 * [added by delta wenjian-23742-TPCC-update-with-Raliegth-code,r1.3]
 *
 * Revision 1.15 1998/10/22 20:51:00wenjian
 * [merge of changes from 1.6 to 1.14 into 1.12]
 *
 * Revision 1.14 1998/10/08 14:17:57dongfeng
 * Add codes for doing web-based tpcc.
 * [from r1.6 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.1]
 *
 * Revision 1.12 1998/09/04 19:17:54wenjian
 * Add new variables: more_srv_work, period_to_add_rt, and
 * period_to_check_tran to replace the original constants in
 * order to control the increment of server RT.
 * [from r1.11 by delta wenjian-23183-TPCC-sync-AIX-code-with-Austin,r1.5]
 *
 * Revision 1.11 1998/08/28 18:29:56wenjian
 * This delta sync the TPCC code with Austin.
 *
 * Modify get_wait_time():
 * - add rt_increment so that the wait time is increased in a certain time;
 * - rt_increment is reset to 0 at the beginning of each run
 * - the waiting time is different for differenttran type.
 * [from r1.8 by delta wenjian-23183-TPCC-sync-AIX-code-with-Austin,r1.1]
 *
 * Revision 1.8 1998/08/18 14:38:37wenjian
 * Change the wait time for NewOrder to 0.23 second
 * [from r1.6 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.4]
 *
 * Revision 1.6 1998/06/17 15:28:50wenjian
 * - Reduce matrix size
 * - In get_wait_time(), the waiting time is decided by transaction type.
 * [from r1.5 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.2]
 *
 * Revision 1.5 1998/02/17 22:06:58wenjian
 * Define macro RANDOM as rand on NT and random on other platforms
 * [from r1.4 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.1]
 *
 * Revision 1.4 1998/01/23 15:07:42 oz
 * - Updated the SP TPCC directory to the latest files used
 *   during the SP tpcc audit.
 * [from r1.3 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
 *
 * Revision 1.1 1997/07/22 21:17:14radha
 * [added by delta radha-20360-TPCC-integrate-with-Oracle-7322-driversr1.1]
 *
 */

#include <stdio.h>
#include <time.h>
#include <string.h>
#include "serverDebug.h"

#ifdef MULTIPLE_INTERFACE
#include "common/neworder.h"
#include "common/payment.h"

```

```
#include "common/stocklevel.h"
#include "common/orderstatus.h"
#else
#include "common/tpcc_trans.h"
#endif
#include "common/databuf.h"
#include "server.h"

#ifdef WIN32
#include <winsock.h>
#endif

#include "tpcc_info.h"

#ifdef WIN32
#define RANDOM rand
#else
#define RANDOM random
#endif

extern int server_null_test;
extern void *create_ora_connection();

#ifdef DEBUG_SERVER
#define PRINT_NEW_IN(a, b) fprintf(stderr, "%s\n", b); print_new_in(a)
#define PRINT_NEW_ORDER(a, b) fprintf(stderr, "%s\n", b); print_new_order(a)
#define PRINT_NEW_RES(rc, a) \
    fprintf(stderr, "<R do_new_order, rc=%d, transtatus=%d, duplicates=%d, all_local=%d\n", \
        rc, (a)->s_transtatus, (a)->s_all_local, (a)->duplicate_items)
#else
#define PRINT_NEW_RES(rc, a)
#define PRINT_NEW_ORDER(a, b)
#define PRINT_NEW_IN(a, b)
#define PRINT_DIST_NEW_ORDER(a, b)
#endif

#define TPCC_RET_SCP(a,b,len) \
    strncpy((char *)dataP->b, (char *)oraStruct.a, len); \
    (char *)dataP->b[(len)-1] = '\0'
#define TPCC_CP(a,b) oraStruct.a = dataP->b
#define TPCC_SCP(a,b,len) strncpy((char *)oraStruct.a, (char *)dataP->b, len)
#define TPCC_RET_CP(a,b) dataP->b = oraStruct.a

#define TPCCP_RET_SCP(a,b,len) \
    strncpy((char *)dataP->b, (char *)oraStructP->a, len); \
    dataP->b[(len)-1] = '\0'
#define TPCCP_CP(a,b) oraStructP->a = dataP->b
#define TPCCP_SCP(a,b,len) strncpy((char *)oraStructP->a, (char *)dataP->b, len)
#define TPCCP_RET_CP(a,b) dataP->b = oraStructP->a

/*
 * Talk to Oracle
 */
int get_db_ready(char *dbName, int flag)
{
    int rc;
    char dvryFileName[100];
    extern char *tpcc_serverName;

    AUDITLOG(">get_db_ready to %s flag %d\n", dbName, flag);
    if (server_null_test) return(0);

    fprintf(stderr, ">> get_db_ready, db: %s, flag %d\n", dbName, flag);

    sprintf(dvryFileName, "/home/encina/runs/deliveries/%s",
        tpcc_serverName);
    rc = TPCinit(serverIdNumber, "tpcc", "tpcc", dvryFileName);
    err_printf("TPCinit(%d,tpcc, tpcc, %s) returned %d\n",
        serverIdNumber, dvryFileName, rc);
    if (rc) {
        fprintf(stderr, "TPCinit(%d,tpcc, tpcc, %s) returned %d\n",
            serverIdNumber, dvryFileName, rc);
    }

    AUDITLOG("<get_db_ready rc %d\n", rc);
    return(rc);
}

void *create_connection() {
    return create_ora_connection();
}

void do_delivery(cnP, dataP)
void *cnP;
delivery_data_t *dataP;
{
    struct delstruct oraStruct;
    int rc;

    AUDITLOG(">do_delivery\n");

    TPCC_CP(delin.w_id, w_id);
    TPCC_CP(delin.o_carrier_id, o_carrier_id);
    TPCC_CP(delin.qtime, start_queue);
    TPCC_CP(delin.in_timing_int, queued_time);

```

```

DPRINT(("CallingTPCdel: w_id %d, o_carrier_id %d, %f qtime, %d in_timing_int\n",
oraStruct.delin.w_id, oraStruct.delin.o_carrier_id,
oraStruct.delin.qtime, oraStruct.delin.in_timing_int));

rc = TPCdel(cnP, &oraStruct);
if ((rc != 0) && (rc != -666)) {
err_printf("Error TPCdel: terror %d, rc %d, retry %d, w_id %d, o_carrier_id %d, %f qtime,
%d in_timing_int\n",
oraStruct.delout.terror, rc, oraStruct.delout.retry,
oraStruct.delin.w_id, oraStruct.delin.o_carrier_id,
oraStruct.delin.qtime, oraStruct.delin.in_timing_int);
}
dataP->header.returncode = rc == 0 ? TPCC_SUCCESS : oraStruct.delout.terror;
AUDITLOG(("<do_delivery rc %d\n", rc));
}

void copyout_order_status(orderStatus_data_t *dataP,
struct ordstruct *oraStructP)
{
int i;
TPCCP_RET_CP(ordout.c_balance, c_balance);
TPCCP_RET_CP(ordout.o_id, o_id);
TPCCP_RET_CP(ordout.o_carrier_id, o_carrier_id);
TPCCP_RET_CP(ordout.o_ol_cnt, o_ol_cnt);
TPCCP_RET_CP(ordout.c_id, c_id);
#define I_CP(ind, a,b) dataP->item[ind].b = oraStructP->ordout.a[ind]
#define I_SCP(ind, a, b, len) \
strcpy((char *)dataP->item[ind].b, (char *)oraStructP->ordout.a[ind], len); \
dataP->item[ind].b[(len) - 1] = '\0'
for (i=0; i<oraStructP->ordout.o_ol_cnt && i < 15; i++) {
I_CP(i, ol_amount, ol_amount);
I_CP(i, ol_i_id, ol_i_id);
I_CP(i, ol_supply_w_id, ol_supply_w_id);
I_CP(i, ol_quantity, ol_quantity);
I_SCP(i, ol_delivery_d, delivery_date, 11);
}
#undef I_CP
#undef I_SCP
TPCCP_RET_SCP(ordout.c_first, c_first, 17);
TPCCP_RET_SCP(ordout.c_middle, c_middle, 3);
TPCCP_RET_SCP(ordout.c_last, c_last, 17);
TPCCP_RET_SCP(ordout.o_entry_d, entry_date, 20);
}

void do_order_status(cnP, dataP)
void *cnP;
orderStatus_data_t *dataP;
{
struct ordstruct oraStruct;
int i, rc;

AUDITLOG((">do_order_status\n"));

TPCC_CP(ordin.w_id, w_id);
TPCC_CP(ordin.d_id, d_id);
TPCC_CP(ordin.c_id, c_id);
oraStruct.ordin.bylastname = ((dataP->c_id == 0) ? 1 : 0);
TPCC_SCP(ordin.c_last, c_last, 17);

DEBUGP(("CallingTPCord: w_id %d, d_id %d, c_id %d, bylastname %d, c_last %s\n",
oraStruct.ordin.w_id, oraStruct.ordin.d_id, oraStruct.ordin.c_id, oraStruct.ordin.bylastname,
oraStruct.ordin.c_last));

rc = TPCord(cnP, &oraStruct);
if (rc != 0) {
err_printf("Error TPCord: terror %d, rc %d, retry %d, w_id %d, d_id %d, c_id %d, bylastname %d,
c_last %s\n ",
oraStruct.ordout.terror, rc, oraStruct.ordout.retry,
oraStruct.ordin.w_id, oraStruct.ordin.d_id, oraStruct.ordin.c_id,
oraStruct.ordin.bylastname, oraStruct.ordin.c_last);
}

copyout_order_status(dataP, &oraStruct);

dataP->header.returncode = rc == 0 ? TPCC_SUCCESS : oraStruct.ordout.terror;
AUDITLOG(("<do_order_stats rc %d\n", dataP->header.returncode));
}

void do_stock_level(cnP, dataP)
void *cnP;
stockLevel_data_t *dataP;
{
struct stostruct oraStruct;
/* What's this comment?? -- srs: i only did this one to check the links */
int rc;

AUDITLOG((">do_stock_level\n"));

TPCC_CP(stoin.w_id, w_id);
TPCC_CP(stoin.d_id, d_id);
TPCC_CP(stoin.threshold, threshold);

DEBUGP(("CallingTPCsto: w_id %d, d_id %d, threshold %d\n",
oraStruct.stoin.w_id, oraStruct.stoin.d_id,
oraStruct.stoin.threshold));

rc = TPCsto(cnP, &oraStruct);
if (rc != 0) {
err_printf("Error TPCsto: terror %d, rc %d, retry %d, w_id %d, d_id %d, threshold %d\n",
oraStruct.stout.terror, rc, oraStruct.stout.retry,
oraStruct.stoin.w_id, oraStruct.stoin.d_id,
oraStruct.stoin.threshold);
}

TPCC_RET_CP(stout.low_stock, stock_count);

dataP->header.returncode = rc == 0 ? TPCC_SUCCESS : oraStruct.stout.terror;

DEBUGP(("do_stock_lev returning %d\n", dataP->header.returncode));
AUDITLOG(("<do_stock_level rc %d\n", dataP->header.returncode));
}

void copyin_payment(dataP, oraStructP)
payment_data_t *dataP;
struct paystruct *oraStructP;
{
TPCCP_CP(payin.w_id, w_id);
TPCCP_CP(payin.d_id, d_id);
TPCCP_CP(payin.c_w_id, c_w_id);
TPCCP_CP(payin.c_d_id, c_d_id);
TPCCP_CP(payin.c_id, c_id);
oraStructP->payin.bylastname = ((dataP->c_id == 0) ? 1 : 0);
TPCCP_CP(payin.h_amount, h_amount);
TPCCP_SCP(payin.c_last, c_last, 17);
}

void copyout_payment(dataP, oraStructP)
payment_data_t *dataP;
struct paystruct *oraStructP;
{
TPCCP_RET_SCP(payout.w_street_1, w_street_1, 21);
TPCCP_RET_SCP(payout.w_street_2, w_street_2, 21);
TPCCP_RET_SCP(payout.w_city, w_city, 21);
TPCCP_RET_SCP(payout.w_state, w_state, 3);
TPCCP_RET_SCP(payout.w_zip, w_zip, 10);
TPCCP_RET_SCP(payout.d_street_1, d_street_1, 21);
TPCCP_RET_SCP(payout.d_street_2, d_street_2, 21);
TPCCP_RET_SCP(payout.d_city, d_city, 21);
TPCCP_RET_SCP(payout.d_state, d_state, 3);
TPCCP_RET_SCP(payout.d_zip, d_zip, 10);
TPCCP_RET_CP(payout.c_id, c_id);
TPCCP_RET_SCP(payout.c_first, c_first, 17);
TPCCP_RET_SCP(payout.c_middle, c_middle, 3);
TPCCP_RET_SCP(payout.c_last, c_last, 17);
TPCCP_RET_SCP(payout.c_street_1, c_street_1, 21);
TPCCP_RET_SCP(payout.c_street_2, c_street_2, 21);
TPCCP_RET_SCP(payout.c_city, c_city, 21);
TPCCP_RET_SCP(payout.c_state, c_state, 3);
TPCCP_RET_SCP(payout.c_zip, c_zip, 10);
TPCCP_RET_SCP(payout.c_phone, c_phone, 17);
TPCCP_RET_SCP(payout.c_since, c_date, 11);
TPCCP_RET_SCP(payout.c_credit, c_credit, 3);
TPCCP_RET_CP(payout.c_credit_lim, c_credit_lim);
TPCCP_RET_CP(payout.c_discount, c_discount);
TPCCP_RET_CP(payout.c_balance, c_balance);
TPCCP_RET_SCP(payout.c_data, c_data, 201);
TPCCP_RET_SCP(payout.h_date, pay_date, 20);
strcpy((char *)dataP->w_name, "W_NAME");
strcpy((char *)dataP->d_name, "D_NAME");
/* Ignore c_ytd_payment, c_payment_cnt */
}

void do_payment(cnP, dataP)
void *cnP;
payment_data_t *dataP;
{
struct paystruct oraStruct;
int firstWh, secondWh;
int rc;

AUDITLOG((">do_payment\n"));

copyin_payment(dataP, &oraStruct);

#if 0
err_printf("TPCpay: w_id %d, D_id %d, C_w_id %d, c_id %d, bylastname %d, amount %.2f, c_last %s
(%s)\n",
oraStruct.payin.w_id,
oraStruct.payin.d_id,
oraStruct.payin.c_w_id,
oraStruct.payin.c_id,
oraStruct.payin.bylastname,
oraStruct.payin.h_amount,
oraStruct.payin.c_last,
dataP->c_last);
#endif

rc = TPCpay(cnP, &oraStruct);

#if 0
err_printf("< TPCpay terror %d, rc %d, retry %d\n",
oraStruct.payout.terror, rc, oraStruct.payout.retry);
#endif
}

```


<pre> dataP->header.num_rms = 1; if (rc != 0) { err_printf("Error TPCpay: terror %d, rc %d, retry %d, w_id %d, D_id %d, C_w_id %d, c_id %d, bylastname %d, amount %.2f, c_last %s (%s)\n", oraStruct.payout.terror, rc, oraStruct.payout.retry, oraStruct.payin.w_id, oraStruct.payin.d_id, oraStruct.payin.c_w_id, oraStruct.payin.c_id, oraStruct.payin.bylastname, oraStruct.payin.h_amount, oraStruct.payin.c_last ? oraStruct.payin.c_last : "-NULL-", (char *)dataP->c_last ? (char *)dataP->c_last : "-NULL-"); } copyout_payment(dataP, &oraStruct); dataP->header.returncode = rc == 0 ? TPCC_SUCCESS : oraStruct.payout.terror; AUDITLOG(("<do_payment rc %d\n", dataP->header.returncode)); } static void copyin_new_order(dataP, oraStructP) newOrder_data_t *dataP; struct newstruct *oraStructP; { int i; TPCCP_CP(newin.w_id, w_id); TPCCP_CP(newin.d_id, d_id); TPCCP_CP(newin.c_id, c_id); #define NO_I_CP(ind,a,b) oraStructP->a[ind] = dataP->item[ind].b #define NO_I_SCP(ind,a,b,len) strncpy((char *)oraStructP->a[ind], (char *)dataP->item[ind].b, len) /* tpccpl.c loops over 15 items, we do the same */ for (i=0; i<15; i++) { NO_I_CP(i, newin.ol_i_id, ol_i_id); NO_I_CP(i, newin.ol_supply_w_id, ol_supply_w_id); NO_I_CP(i, newin.ol_quantity, ol_quantity); } #define DEBUG_SERVER fprintf(stderr, "NewOrder: Item %d, supplyWh %d (local %d)\n", i, oraStructP->newin.ol_supply_w_id[i], oraStructP->newin.w_id); } #endif /* Ignore all_local field, total_items, * tpccpl.c doesnt use them */ #undef NO_I_CP #undef NO_I_SCP } void copyout_new_order(dataP, oraStructP) newOrder_data_t *dataP; struct newstruct *oraStructP; { int i; TPCCP_RET_CP(newout.o_id, o_id); TPCCP_RET_CP(newout.o_ol_cnt, o_ol_cnt); TPCCP_RET_SCP(newout.c_last, c_last, 17); TPCCP_RET_SCP(newout.c_credit, c_credit, 3); TPCCP_RET_CP(newout.c_discount, c_discount); TPCCP_RET_CP(newout.w_tax, w_tax); TPCCP_RET_CP(newout.d_tax, d_tax); TPCCP_RET_SCP(newout.o_entry_d, entry_date, 20); TPCCP_RET_CP(newout.total_amount, total); TPCCP_RET_SCP(newout.status, statusline, 26); #define NO_RET_CP(ind,a,b) dataP->item[ind].b = oraStructP->newout.a[ind] #define NO_RET_SCP(ind,a,b,len) strncpy((char *)dataP->item[ind].b, (char *)oraStructP->newout.a[ind], len) for (i=0; i<oraStructP->newout.o_ol_cnt && i<15; i++) { NO_RET_SCP(i, i_name, name_i, 25); NO_RET_CP(i, s_quantity, s_quantity); dataP->item[i].brand_generic[0] = oraStructP->newout.brand_generic[i]; dataP->item[i].brand_generic[1] = '0'; NO_RET_CP(i, i_price, price); NO_RET_CP(i, ol_amount, ol_amount); /* Ignore s_idx and s_dist */ } if (oraStructP->newout.status[0] != '0') { DEBUGP(("TPCnew:status -- %s\n", oraStructP->newout.status)); dataP->items_valid = 0; } else { dataP->items_valid = 1; } } #undef NO_RET_CP #undef NO_RET_SCP } void do_new_order(cnP, dataP) void *cnP; newOrder_data_t *dataP; { </pre>	<pre> static int num_calls = 0; int i; struct newstruct oraStruct; int rc; AUDITLOG((">do_new_order\n")); /* Copy the structure into the TPCC structure. */ copyin_new_order(dataP, &oraStruct); DEBUGP((">TPCnew %d items to wh %d\n", dataP->o_ol_cnt, dataP->w_id)); dataP->header.num_rms = 1; #if 0 err_printf("Error TPCnew : w_id %d, d_id %d, c_id %d, o_ol_cnt %d (out cnt %d)\n", oraStruct.newin.w_id, oraStruct.newin.d_id, oraStruct.newin.c_id, dataP->o_ol_cnt, oraStruct.newout.o_ol_cnt); for (i=0; i<15; i++) { err_printf("ol_i_id %d, ol_supply_w_id %d, ol_quantity %d\n", oraStruct.newin.ol_i_id[i], oraStruct.newin.ol_supply_w_id[i], oraStruct.newin.ol_quantity[i]); } #endif rc = TPCnew(cnP, &oraStruct); #if 0 err_printf("<TPCnew terror %d, rc %d, retry %d\n", oraStruct.newout.terror, rc, oraStruct.newout.retry); #endif if (rc != 0) { err_printf("Error TPCnew : terror %d, rc %d, retry %d, w_id %d, d_id %d, c_id %d, o_ol_cnt %d (out cnt %d)\n", oraStruct.newout.terror, rc, oraStruct.newout.retry, oraStruct.newin.w_id, oraStruct.newin.d_id, oraStruct.newin.c_id, dataP->o_ol_cnt, oraStruct.newout.o_ol_cnt); for (i=0; i<15; i++) { err_printf("ol_i_id %d, ol_supply_w_id %d, ol_quantity %d\n", oraStruct.newin.ol_i_id[i], oraStruct.newin.ol_supply_w_id[i], oraStruct.newin.ol_quantity[i]); } } DEBUGP(("<TPCnew %d\n", rc)); /* copy out results */ copyout_new_order(dataP, &oraStruct); if (rc == 0) { dataP->header.returncode = dataP->items_valid ? TPCC_SUCCESS : INVALID_NEWO; } #if 0 if (dataP->items_valid && (++num_calls % 500) == 0) { int i; err_printf("TPCnew Success: w_id %d, d_id %d, c_id %d, o_ol_cnt %d, Oid %d\n", oraStruct.newin.w_id, oraStruct.newin.d_id, oraStruct.newin.c_id, oraStruct.newout.o_ol_cnt, oraStruct.newout.o_id); for (i=0; i<15 && i<oraStruct.newout.o_ol_cnt; i++) { err_printf(" %2d: i_id %15d, sw_id %4d, qty %d, price %.2f amt %.2f\n", i, oraStruct.newin.ol_i_id[i], oraStruct.newin.ol_supply_w_id[i], oraStruct.newin.ol_quantity[i], oraStruct.newout.i_price[i], oraStruct.newout.ol_amount[i]); } } #endif } else { dataP->header.returncode = oraStruct.newout.terror; } AUDITLOG(("<do_new_order rc %d\n", dataP->header.returncode)); } client.C /* (C)1997 IBM Corporation */ #include <unistd.h> #include <stdlib.h> #include <stdio.h> #include <sys/types.h> #include <ctype.h> #include <string.h> #include <math.h> #include "screen.h" #include "encina.h" extern "C" void set_client_debug_state(void *contextP, int state, int tran); Encina encina; extern "C" int client_login(int infd, int outfd, int *w_idP, int *d_idP) { </pre>
---	---

<pre> /* * client_bg_thread.c * * \$Revision: 1.17 \$ * \$Date: 1999/05/06 21:28:25 \$ * \$Log: \$ * * * \$TALog:client_bg_thread.c,v \$ * Revision 1.17 1999/05/06 21:28:25 oz * - Removed all the .. from the includes * - Added -L. to the makefiles instead * - Moved all the thread related code and connection * selection to serverMon.c * [from r1.13 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5] * * Revision 1.13 1999/01/29 20:16:31wenjian * Small changes to make check_threads more robust * [from r1.12 by delta wenjian-23787-TPCC-integrate-code-for-AIX-and-NT,r1.7] * * Revision 1.12 1998/12/28 20:07:11wenjian * - Change client_info to a pointer pClientInfo for flexibility. * [from r1.11 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.5] * * Revision 1.11 1998/12/14 20:27:53wenjian * Made corresponding changes due to data structure change of tran_info_t. * [from r1.10 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.3] * * Revision 1.10 1998/12/11 16:14:18wenjian * Add code for checking statistic data in a single variable and collecting * statistic data based on iStatsFrequency. * * - Add new check_threads() in order to use the single statistic var. * [from r1.9 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.1] * * Revision 1.9 1998/12/08 23:03:48wenjian * Add (or rename) Makefile for each platform (AIX and NT). Reorganize the * files a little bit. * * - Change path for tran_stat.h * - Add ifdef before getStatsForm since it is only used by NT * [from r1.8 by delta wenjian-23787-TPCC-integrate-code-for-AIX-and-NT,r1.1] * * Revision 1.8 1998/12/07 20:04:11wenjian * Clean up * [from r1.7 by delta wenjian-23742-TPCC-update-with-Raleigh-code,r1.2] * * Revision 1.7 1998/11/24 21:45:58wenjian * - Add #ifdef MULTIPLE_INTERFACE * - Take care special case forcheck_threads * [from r1.6 by delta wenjian-23742-TPCC-update-with-Raleigh-code,r1.1] * * Revision 1.6 1998/11/09 16:59:35wenjian * In this revision, most of the changes are related to the directory of header * files after directory reorganization. Other changes include adding or removing * files to put them in the right directories. Makefiles are written for NT * platform so that nmake is working on NT now. Need a top levelMakefile for all * the directories. * [from r1.5 by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.2] * * Revision 1.5 1998/11/09 14:48:14wenjian * In an effort to make a new directory structure for TPCC, this delta * creates two directories: tpcc/client and tpcc/server. All the files * for this revision are copied fromtpcc/sp-tpcc without any change. * Further change may be needed for some files due to the change of * the directory structure. * [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1] * * Revision 1.29 1998/11/06 16:10:54wenjian * - Move gettimeofday() in check_threads out of the for loop * - Minor change for cleanup * [from r1.28 by delta wenjian-23646-TPCC-clean-up-source-code,r1.1] * * Revision 1.28 1998/10/26 15:17:53dongfeng * remove #include <sys/times.h> from NT tests * [from r1.27 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.4] * * Revision 1.27 1998/10/22 21:05:37wenjian * [merge of changes from 1.12 to 1.26 into 1.25] * * Revision 1.26 1998/10/22 19:18:31dongfeng * [from r1.24 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.2] * * Revision 1.24 1998/10/08 14:17:59dongfeng * Add codes for doing web-based tpcc. * [from r1.12 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.1] * * Revision 1.25 1998/10/08 18:03:00gerstl * Changes to allow configurations where some servers only service * specific transaction types. Split transaction interfaces by type. * [from r1.23 by delta gerstl-23515-TPCC-allow-separate-online-transaction-interfaces,r1.1] * * Revision 1.23 1998/09/03 16:07:11wenjian * Change GET_USER_SYS_TIME to GET_CLIENT_USAGE * [from r1.19 by delta wenjian-23183-TPCC-sync-AIX-code-with-Austin,r1.3] * * Revision 1.19 1998/08/18 13:35:41wenjian * - Clean up including header files </pre>	<pre> * - Remove client_report() and socket_print_rt_avg() * - Use #ifdef for the call of getUserSysTime() * [from r1.16 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.7] * * Revision 1.16 1998/07/08 18:15:42wenjian * Add getUserSysTime(). * [from r1.15 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.6] * * Revision 1.15 1998/07/02 18:28:51wenjian * Change client_status_report to send more information of the * client process. These changes are matched with changes in * tpcc_monitor.c. * [from r1.9 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.5] * * Revision 1.7 1998/04/29 19:47:40wenjian * - Add client_status_report to communicate with tpcc_monitor * - Add socket_print_rt_avg * [from r1.6 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.1] * * Revision 1.6 1998/02/17 22:12:40wenjian * [merge of changes from 1.3 to 1.4 into 1.5] * * Revision 1.4 1998/02/17 16:04:40 oz * - Split the login into two parts to allow for special logins * - If the warehouse ID is 0, this is a speciallogin to * query the client for status * * - check_threads: Return the number of threads * - New function: client_report * [from r1.3 by delta oz-21864-TPCC-split-client-login-screen,r1.1] * * Revision 1.5 1998/02/17 22:06:59wenjian * Add necessary head files for win32 * [from r1.3 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.1] * * Revision 1.3 1998/01/29 22:53:34 oz * - Use pthread delay instead of sleep * [from r1.2 by delta oz-21749-TPCC-use-pthread-delay-for-bg-thread,r1.1] * * Revision 1.2 1998/01/26 20:37:34 oz * - Remove all the code associated with explicit binding * * - Removed include of mon_client_utils.h * [from r1.1 by delta oz-21697-TPCC-remove-explicit-binding-code,r1.1] * * Revision 1.1 1998/01/26 16:19:22 oz * - moved all the code pertaining to the background * thread to its own file and all the data structures * to client_utils.h * [added by delta oz-21689-TPCC-move-client-bg-thread-to-separate-file,r1.1] * */ */ /* * client_bg_thread * * A file used for debug purposes only. * * It implements a background thread that once a minute checks the * state of all the threads and reports the state of the client. * */ #include <stdio.h> #include <stdlib.h> #include <string.h> #include <stdarg.h> #include <time.h> #ifdef WIN32 #include <sys/times.h> /* for getUserSysTime */ #endif #if defined (solaris) #include <dce/pthread.h> #else #include <pthread.h> #endif /* solaris */ #include <tpm/mon/mon.h> #include <utils/trace.h> #include "common/delivery.h" #ifdef MULTIPLE_INTERFACE #include "common/neworder.h" #include "common/payment.h" #include "common/stocklevel.h" #include "common/orderstatus.h" #else #include "common/tpcc_trans.h" #endif #include "common/utilities.h" #include "client_utils.h" #include "common/do_tpcc.h" #include "client.h" #include "encina_client.h" #include "tools/tran_stat.h" #include "common/get_local_time.h" #ifdef WIN32 #define read(A,B,C) recv(A,B,C,0) #define write(A,B,C) send(A,B,C,0) </pre>
---	--

<pre> #endif #if 1 #define PRINT_AV(total, num, str) \ { \ if((num) > 0) { \ fprintf(ERROUT, " %s %.0f.", str, (double)(total)/(num)); \ } \ } #else #define PRINT_AV(a,b,c) #endif static void check_threads(total_tran_count_t *tran_ctP, int *numP, int *numInitP); static struct timeval *client_last_time(thread_descr_t *descrP); void getUserSysTime(struct timeval *user_time, struct timeval *sys_time); /* * client_last_time * * Each thread maintains the current state it is in and the time * it entered this state. * This routine returns a pointer to the structure in the thread * data that contains the time corresponding to the threads current * state. * Typical use: * - Set the state, then call gettimeofday on the pointer * returned by this function. */ static struct timeval *client_last_time(thread_descr_t *descrP) { struct timeval *lastTimeP = &descrP->done; switch (descrP->state) { case thread_state_init: /* Thread is initializing - notrans yet */ lastTimeP = &descrP->init; break; case thread_state_called: /* Tran type was sent by the RTE */ lastTimeP = &descrP->called; break; case thread_state_returned: /* Final screen sent to RTE */ lastTimeP = &descrP->returned; break; case thread_state_sent: /* Sent to server */ lastTimeP = &descrP->sent; break; case thread_state_received: /* Received reply from server */ lastTimeP = &descrP->received; break; case thread_state_done: /* The thread exited */ lastTimeP = &descrP->done; break; default: err_printf("client_last_time:bad state: %d\n", descrP->state); lastTimeP = &descrP->done; break; } return(lastTimeP); } void set_client_debug_state(void *contextP, int state, int tran) { thread_info_t *thread_context = (thread_info_t *)contextP; struct timezone tz; thread_descr_t *descrP = &thread_context->descr; descrP->state = state; gettimeofday(client_last_time(descrP), &tz); if (state == thread_state_called) descrP->tran = tran; } /* How often to report the state of a thread: * If it is in the thread_state_init phase: report if it has been in * that state for more than 5 minutes. * Report if it takes the terminal more than 3 minutes to generate the next * transaction. Otherwise, report if anything takes longer than 60 seconds. */ #define THREAD_STATE_REPORT_DELTA(state) \ ((state) == thread_state_init ? 300 : \ (state) == thread_state_returned ? 180 : 60) static char *thread_state_to_str(int state) { char *ret_val = "-Unknown-"; switch(state) { case thread_state_init: ret_val = "state_init"; break; case thread_state_called: ret_val = "state_called"; break; case thread_state_sent: ret_val = "state_sent"; break; case thread_state_received: ret_val = "state_received"; break; case thread_state_done: ret_val = "state_done"; break; case thread_state_returned: ret_val = "state_returned"; break; } return(ret_val); } static void print_rt_avg(total_tran_count_t *curP, total_tran_count_t *prevP, </pre>	<pre> int type) { int i; static char *names[] = {"0", "no", "pa", "os", "dl", "sl"}; err_printf("%s RT avg:", type ? "server" : "client"); for (i=1; i<=MAX_TRAN_TYPE; i++) { int num_trans = curP->tran[i].RTcount - prevP->tran[i].RTcount; double rt_diff = curP->tran[i].RTtotal[type] - prevP->tran[i].RTtotal[type]; PRINT_AV(rt_diff, num_trans, names[i]); } fprintf(ERROUT, "\n"); } /* * A background thread that keeps tabs on the state of all the * threads of the client. (For Debug) */ static void *bg_thread(void *argP) { static int sleep_time = 60000; /* in ms */ static struct timespec time_wait = {60, 0}; struct timespec sleep_end; total_tran_count_t tran_ct, tran_reported[2]; int total_newo, total_tran_err; struct timeval cur_time; struct timezone tz; struct timeval time_reported[2]; gettimeofday(&time_reported[0], &tz); time_reported[1] = time_reported[0]; memset(&tran_reported[0], '\0', 2 * sizeof(tran_reported[0])); while (1) { double time_diff1, time_diff2; double tran_diff1, tran_diff2; double prev_newo1, prev_newo2; double err_diff1, err_diff2; check_threads(&tran_ct, NULL, NULL); total_tran_err = tran_ct.errors; total_newo = tran_ct.tran[NEWO_TRANS].num - tran_ct.tran[NEWO_TRANS].errs; gettimeofday(&cur_time, &tz); time_diff1 = time_diff_ms(&cur_time, &time_reported[0]); prev_newo1 = tran_reported[0].tran[NEWO_TRANS].num - tran_reported[0].tran[NEWO_TRANS].errs; tran_diff1 = total_newo - prev_newo1; err_diff1 = total_tran_err - tran_reported[0].errors; time_diff2 = time_diff_ms(&cur_time, &time_reported[1]); prev_newo2 = tran_reported[1].tran[NEWO_TRANS].num - tran_reported[1].tran[NEWO_TRANS].errs; tran_diff2 = total_newo - prev_newo2; err_diff2 = total_tran_err - tran_reported[1].errors; if (total_newo != 0 && tran_diff2 > 0) { err_printf("bg_thread: TPM: %.0f (last %.0f sec), %.0f (last %.0f sec)\n", tran_diff1 / time_diff1 * 60000, time_diff1 / 1000., tran_diff2 / time_diff2 * 60000, time_diff2 / 1000.); /* print av server response time for all transactions */ print_rt_avg(&tran_ct, &tran_reported[1], 0); print_rt_avg(&tran_ct, &tran_reported[1], 1); } if (err_diff2 != 0) { err_printf("bg_thread: errPM %.1f (last %.0f sec)\n", err_diff2 / time_diff2 * 60000, time_diff2 / 1000.); } tran_reported[0] = tran_reported[1]; tran_reported[1] = tran_ct; time_reported[0] = time_reported[1]; time_reported[1] = cur_time; pthread_delay_np(&time_wait); } } #ifdef KEEP_TERMINAL_INFO static void check_threads(total_tran_count_t *tran_ctP, int *num_threadsP, int *num_threadsInitP) { struct timezone tz; int num_per_state[NUM_STATES]; int total_stuck = 0; static int init_printed = 0; int total_tran_err; int num_active = 0; MUTEX_LOCK(&init_lock); if (info_list == NULL (info_list_len < 1)) { if (num_threadsP) *num_threadsP = 0; if (num_threadsInitP) *num_threadsInitP = 0; memset(tran_ctP, '\0', sizeof(*tran_ctP)); } } </pre>
--	--

```

} else {
int i,j;
struct timeval cur_time;
int num_init = 0, num_done = 0;

for (i=0; i<NUM_STATES; i++) num_per_state[i] = 0;

gettimeofday(&cur_time, &tz);
memset(tran_ctP, '0', sizeof(*tran_ctP));
for (i=0; i<info_list_len; i++) {
struct timeval *client_timeP;
int time_diff;
thread_descr_t *descrP;
int delta;
if(info_list[i] == NULL || !info_list[i]->initialized){
continue;
}
if(!info_list[i]->done) num_active++;
descrP = &info_list[i]->descr;
delta = THREAD_STATE_REPORT_DELTA(descrP->state);
client_timeP = client_last_time(descrP);

for (j=1; j<=MAX_TRAN_TYPE; j++) {
tran_ctP->tran[j].num += info_list[i]->tran[j].num;
tran_ctP->tran[j].errs += info_list[i]->tran[j].errs;
tran_ctP->tran[j].RTcount += info_list[i]->tran[j].RTcount;
tran_ctP->tran[j].RTtotal[0] += info_list[i]->tran[j].RTtotal[0];
tran_ctP->tran[j].RTtotal[1] += info_list[i]->tran[j].RTtotal[1];
tran_ctP->errors += info_list[i]->tran[j].errs;
}

time_diff = cur_time.tv_sec - client_timeP->tv_sec;
DPRINT("bg_thread: thread %d (index %d) state %s tran %d for %d sec\n",
info_list[i]->thread_id, i,
thread_state_to_str(descrP->state),
descrP->tran,
time_diff);
if (descrP->state == thread_state_init) {
num_init++;
} else if (descrP->state == thread_state_done) {
num_done++;
} else if (time_diff > delta) {
num_per_state[descrP->state]++;
total_stuck++;
if (!descrP->printed) {
err_printf("bg_thread: thread %d (index %d) state %s tran %d stuck for %d
sec\n",
info_list[i]->thread_id, i,
thread_state_to_str(descrP->state),
descrP->tran,
time_diff);
descrP->printed = 1;
}
} else if (descrP->printed) {
err_printf("bg_thread: thread %d (index %d) state %s tran %d unstuck.\n",
info_list[i]->thread_id, i,
thread_state_to_str(descrP->state),
descrP->tran);
descrP->printed = 0;
}
}
}

if (num_threadsP)
*num_threadsP = num_active;
if (num_threadsInitP)
*num_threadsInitP = num_init;

if (num_init > 0) {
err_printf("bg_thread: %d threads still in the init state\n",
num_init);
} else if (!init_printed) {
err_printf("bg_thread: All %d threads are running\n",
info_list_len);
init_printed = 1;
}
if (num_active != info_list_len)
err_printf("%d threads of %d are still active\n",
num_active, info_list_len);

if (num_done > 0) {
err_printf("bg_thread: %d threads done so far.\n", num_done);
}
if (total_stuck > 0) {
err_printf("bg_thread: Summary %d stuck: ", total_stuck);
for (i=0; i<NUM_STATES; i++) {
if (num_per_state[i] > 0) {
fprintf(ERROROUT, "%d %s, ",
num_per_state[i], thread_state_to_str(i));
}
}
fprintf(ERROROUT, "\n");
}
total_tran_err = 0;
for (i=0; i<=MAX_TRAN_TYPE; i++)
total_tran_err += tran_ctP->tran[i].errs;
if (total_tran_err > 0) {
err_printf("bg_thread: %d errs: %d no, %d pa, %d os, %d sl\n",
total_tran_err,
tran_ctP->tran[NEWO_TRANS].errs,
tran_ctP->tran[PAYMENT_TRANS].errs,
tran_ctP->tran[ORDER_STAT_TRANS].errs,
tran_ctP->tran[STOCK_TRANS].errs);
}
}
MUTEX_UNLOCK(&init_lock);
}

#else
static void check_threads(total_tran_count_t *tran_ctP, int *num_threadsP, int *num_threadsInitP)
{
int i;
extern total_tran_count_t *pClientInfo;

if (num_threadsP != NULL)
*num_threadsP = 0;
if (num_threadsInitP != NULL)
*num_threadsInitP = 0;

memcpy(tran_ctP, pClientInfo, sizeof(total_tran_count_t));

/* report error info */
if (pClientInfo->errors > 0) {
err_printf("bg_thread: %d errs: %d no, %d pa, %d os, %d sl\n",
pClientInfo->errors,
pClientInfo->tran[NEWO_TRANS].errs,
pClientInfo->tran[PAYMENT_TRANS].errs,
pClientInfo->tran[ORDER_STAT_TRANS].errs,
pClientInfo->tran[STOCK_TRANS].errs);
}
}

#endif

void start_bg_debug_thread()
{
int rc;
pthread_attr_t attr;
pthread_t thread;

if (rc = pthread_attr_create(&attr)) {
err_printf("start_bg_debug_thread: pthread_attr_create failed: %d\n", rc);
return;
}
if ((rc = pthread_create(&thread,
attr,
bg_thread,
(pthread_addr_t) NULL)) != 0) {
err_printf("start_bg_debug_thread: pthread_create failed: %d\n", rc);
return;
}
if (rc = pthread_detach(&thread) != 0) {
err_printf("start_bg_debug_thread: pthread_detach failed %d\n", rc);
return;
}
}

/* client_status_report:
* mainly copied from bg_thread
*/
void *client_status_report(int fileno)
{
static struct timespec time_wait = {60, 0};

total_tran_count_t tran_ct;
tran_info_t *curP;
struct timeval cur_time;
struct timezone tz;
char buf[1024], cmd='\0';
int i, cnt=0;

/* a loop for communication with tpcc_monitor */
while (cmd != 'q') {
struct timeval cur_time;
struct timeval user_time={0,0}, sys_time={0,0};
struct timezone tz;
int num_threads, num_threadsInit;

memset(&tran_ct, 0, sizeof(tran_ct));

/* read next cmd from the socket */
read(fileno, buf, 1);
cmd = buf[0];
/* DPRINT("%c\n", cmd); */
if (cmd=='q') {
break;
}

check_threads(&tran_ct, &num_threads, &num_threadsInit);
gettimeofday(&cur_time, &tz);
#ifdef GET_CLIENT_USAGAGE
getUserSysTime(&user_time, &sys_time);
#endif

/* The tpcc_monitor has to read the data in the same order

```

```

* to get the correct data.
*/
    prefix_sprintf(buf,"n");
    write(fileno,buf,strlen(buf));
    sprintf(buf,"%d %d %d %d %d %d\n",
        cur_time.tv_sec, cur_time.tv_usec,
        tran_ct.tran[NEWO_TRANS].num-tran_ct.tran[NEWO_TRANS].errs,
        tran_ct.errors, num_threads, num_threadsInit);
    write(fileno,buf, strlen(buf));
    sprintf(buf,"%d %d %d %d %d %d\n", user_time.tv_sec,user_time.tv_usec,
        sys_time.tv_sec, sys_time.tv_usec);
    write(fileno,buf, strlen(buf));
    for (i=0,curP=tran_ct.tran; i<=MAX_TRAN_TYPE;i++, curP++) {
        if (i==0) continue;
        sprintf(buf,"%d %d %d %d %d %d\n",
            i, curP->RTtotal, curP->errs, curP->RTtotal[0],
            curP->RTtotal[1]);
        write(fileno,buf, strlen(buf));
    }
    write(fileno,ENDMSG, strlen(ENDMSG));
}
}
}
/* for AIX only */
void getUserSysTime(struct timeval *user_time, struct timeval *sys_time)
{
#ifdef defined(AIX)
    struct rusage rubuff;

    if (getrusage(RUSAGE_SELF,&rubuff) == 0) {
        user_time->tv_sec = rubuff.ru_utime.tv_sec;
        user_time->tv_usec = rubuff.ru_utime.tv_usec;

        sys_time->tv_sec = rubuff.ru_stime.tv_sec;
        sys_time->tv_usec = rubuff.ru_stime.tv_usec;
    } else {
        user_time->tv_sec = user_time->tv_sec = 0;
        sys_time->tv_sec = sys_time->tv_sec = 0;
    }
#endif
#ifdef defined(solaris)
    /* WARNING: not test it yet */
    struct tms t;
    static long ticks = 0;
    register long n;

    if (ticks == 0) ticks =sysconf(_SC_CLK_TCK);

    (void)times(&t);

    user_time->tv_sec = t.tms_utime / ticks;
    user_time->tv_usec = (t.tms_utime % ticks) * 1000000 / ticks;

    sys_time->tv_sec = t.tms_stime / ticks;
    sys_time->tv_usec = (t.tms_stime % ticks) * 1000000 / ticks;
#else
    user_time->tv_sec = 0;
    user_time->tv_usec = 1000;

    sys_time->tv_sec = 0;
    sys_time->tv_usec = 1000;
#endif
}
}

#ifdef WEB_TPCC_CLIENT
extern void getStatsForm(char* outBufP, char* rawStatsP, int interval);

int web_client_status(char* szFormP, int cmd, int interval)
{
    total_tran_count_t tran_ct;
    tran_info_t *curP;
    struct timeval cur_time;
    struct timezone tz;
    int i, cnt=0;
    int num_threads, num_threadsInit;
    char *statusP;
    char tempP[512];

    if(cmd==1)
        statusP = szFormP;
    else
        statusP = tempP;

    *statusP = '\0';
    memset(&tran_ct, 0, sizeof(tran_ct));

    if(cmd == 2) /* quit */
        return 1;

    check_threads(&tran_ct, &num_threads, &num_threadsInit);
    gettimeofday(&cur_time, &tz);

    prefix_sprintf(statusP,"n");
    sprintf(statusP+strlen(statusP), "%d %d %d %d %d %d\n", cur_time.tv_sec,
        cur_time.tv_usec,
        tran_ct.tran[NEWO_TRANS].num-tran_ct.tran[NEWO_TRANS].errs,
        tran_ct.errors,

```

```

        num_threads, num_threadsInit);
    sprintf(statusP+strlen(statusP), "0 0 0 0\n");
    for (i=0,curP=tran_ct.tran; i<=MAX_TRAN_TYPE;i++, curP++) {
        if (i==0) continue;
        sprintf(statusP+strlen(statusP), "%d %d %d %d %d %d\n",
            i, curP->RTcount, curP->errs, curP->RTtotal[0],
            curP->RTtotal[1]);
    }

    if(cmd != 1)
        getStatsForm(szFormP, statusP, interval);

    return 0;
}
#endif

client_listen.c

/*
 * client_listen.c
 *
 * $Revision: 1.8 $
 * $Date: 1999/05/06 21:28:26 $
 * $Lo: $
 *
 *
 * $TALog: client_listen.c,v $
 * Revision 1.8 1999/05/06 21:28:26 oz
 * - Removed all the .. from the includes
 * - Added -I. to the makefiles instead
 * - Moved all the thread related code and connection
 * selection to serverMon.c
 * [from r1.7 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5]
 *
 * Revision 1.7 1999/01/29 20:16:32wenjian
 * Change logprintf to err_printf
 * [from r1.6 by delta wenjian-23787-TPCC-integrate-code-for-AIX-and-NT,r1.7]
 *
 * Revision 1.6 1998/11/09 16:59:36wenjian
 * In this revision, most of the changes are related to the directory of header
 * files after directory reorganization. Other changes include adding or removing
 * files to put them in the right directories. Makefiles are written for NT
 * platform so that nmake is working on NT now. Need a top level Makefile for all
 * the directories.
 * [from r1.5 by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.2]
 *
 * Revision 1.5 1998/11/09 14:48:14wenjian
 * In an effort to make a new directory structure for TPCC, this delta
 * creates two directories: tpcc/client and tpcc/server. All the files
 * for this revision are copied from tpcc/sp-tpcc without any change.
 * Further change may be needed for some files due to the change of
 * the directory structure.
 * [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1]
 *
 * Revision 1.30 1998/09/26 10:56:25 oz
 * - renamed thread_init and thread_done to clnt_thread_init and
 * clnt_thread_done respectively because of name conflicts on AIX4.3
 * [from r1.22 by delta oz-23339-TPCC-update-for-NT,r1.2]
 *
 * Revision 1.22 1998/08/18 14:38:38wenjian
 * Minor change
 * [from r1.18 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.4]
 *
 * Revision 1.18 1998/04/29 19:47:41wenjian
 * - Use fd instead of stream on NT
 * - Add code to consider tpcc_monitor as a special client login
 * - Use TRY and CATCH_ALL to deal with exceptions
 * [from r1.17 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.1]
 *
 * Revision 1.17 1998/02/17 22:13:28wenjian
 * [merge of changes from 1.14 to 1.15 into 1.16]
 *
 * Revision 1.15 1998/02/17 16:04:41 oz
 * - Split the login into two parts to allow for special logins
 * - If the warehouse ID is 0, this is a speciallogin to
 * query the client for status
 *
 * - First, login
 * If the w_id is bigger than 0: normal thread.
 * Otherwise, callclient_report.
 * [from r1.14 by delta oz-21864-TPCC-split-client-login-screen,r1.1]
 *
 * Revision 1.16 1998/02/17 22:06:59wenjian
 * Add head files and define macros for win32
 * [from r1.14 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.1]
 *
 * Revision 1.14 1998/01/28 22:24:48 oz
 * [from r1.13 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.4]
 *
 * Revision 1.13 1998/01/26 16:19:22 oz
 * - moved all the code pertaining to the background
 * thread to its own file and all the data structures
 * to client_utils.h
 * [from r1.12 by delta oz-21689-TPCC-move-client-bg-thread-to-separate-file,r1.1]
 *
 * Revision 1.12 1998/01/24 14:17:04 oz
 * - User server name to identify server and name delivery file
 * - Use env variable HOME instead of /home/encina if HOME is set
 *

```

```

* - Print the thread ID on thread exit as well
* [from r1.11 by delta oz-21687-TPCC-use-server-name-to-identify-process,r1.1]
*
* Revision 1.11 1998/01/23 15:07:44 oz
* - Updated the SP TPCC directory to the latest files used
* during the SP tpcc audit.
* [from r1.10 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
*
*
* Exported functions:
*     make_connections
*
* Private functions:
*     process_terminal
*
*/

/* client_listen.c
* Code in the client that listens for requests from the
* terminal processes and submits them for processing.
*
* There is one listening function:make_connection.
* That function calls cnm_ManageConnection which never returns
* and so it is best to call it in its own independent thread.
*
* As soon as cnm_ManageConnections receives a connection it
* starts a new thread and calls process_terminal in that
* thread passing in the file descriptor for the new connection.
*
* Note that the client does not need to know in advance how many
* terminals it will talk to.
*
* The function process_terminal reads initializes the thread
* and then calls client_init to process all the requests from
* that terminal.
*/

#include <stdlib.h>
#ifdef WIN32
#include <io.h>
#else
#include <stdio.h>
#include <sys/types.h>
#include <tc/tc.h>
#include "common/do_tpcc.h"
#include "common/tpcc_type.h"

#ifdef (solaris)
#include <dce/pthread.h>
#else /* solaris */
#include <pthread.h>
#endif /* solaris */

#include "client_utils.h"
#ifdef WIN32
#include "cnm.h"
#else
#include <cnm/cnm.h>
#endif

#ifdef WIN32
#define close      _close
#define fileno     _fileno
#endif

/*
* State about the terminal stored by the terminal thread
* work_entry: The work entry to be used by this terminal thread.
*/
typedef struct {
    int profiling;
    int terminal_id;
    void *handle_contextP;
} terminal_context_t;

/**
** Function Prototypes
**/
static void process_terminal(cnm_arg_t *argP);

extern void client_init(int, int, int, void *);
extern void client_login(int, int, int *, int *);

/*
* process_terminal
*
* The argument we get is a file descriptor for a terminal
* process. We read from that file to receive input and send
* output back to that file.
*/
static void process_terminal(cnm_arg_t *argP)
{
    int w_id, d_id;
    terminal_context_t terminal_context;
    tpcc_data_t tran_data;
    int fdIn;
    pthread_t thread = pthread_self();
    int thread_id = pthread_getunique_np(&thread);

```

```

struct timespec rand_sleep;
#ifdef _AIX
tid_t tid = thread_self();
#else
int tid = thread_id;
#endif

#ifdef WIN32
fdIn = argP->fd;
#else
fdIn = fileno(argP->stream);
#endif /* WIN32 */

/*
* Default terminal context
* This may be updated later by the terminal
*/
terminal_context.terminal_id = -1;
terminal_context.profilng = 0;

TRY {
    client_login(fdIn, fdIn, &w_id, &d_id);
    if(w_id > 0) {
        /* Initialize the server handle and other thread structures */
        terminal_context.handle_contextP = (void *)clnt_thread_init();

        err_printf("Tid: %d (0x%x) w_id %d, d_id %d\n", tid, tid, w_id, d_id);
        client_init(fdIn, fdIn, w_id, d_id, terminal_context.handle_contextP);
        err_printf("Thread done - Tid %d (0x%x)\n", tid, tid);
        clnt_thread_done(terminal_context.handle_contextP);
    } else {
        err_printf("Starting Auxiliary Thread, Tid %d (0x%x)\n", tid, tid);
        client_status_report(fdIn);
        err_printf("End of Auxiliary Thread, Tid %d (0x%x)\n", tid, tid);
    }
} CATCH_ALL {
    err_printf("An exception happened\n");
    logprintf("End of Auxiliary Thread, Tid %d (0x%x)\n", tid, tid);
}
ENDTRY

close(fdIn);
}

/*
* make_connections
*
* Listen for connections on a socket.
* Whenever a connection is made, start a thread to talk
* to the terminal.
* This functions is spawned on its own thread.
*/
void make_connections(argP)
void *argP;
{
    int port = (int)argP;
    char port_descr[28];
    int rc;

    DPRINT(("Using socket %d\n", port));
    err_printf("Using thread stack size default\n");
    sprintf(port_descr, "ncacn_ip_tcp[%d]", port);
    rc = cnm_ManageConnections(port_descr,
                                (cnm_userRoutine_t)process_terminal,
                                NULL,
                                0, /* Max Connections */
                                1); /* Spawn threads */
    err_printf("cnm_ManageConnections returned %d\n", rc);
}

```

client_listen.h

```

/*
* client_listen.h
*
* $Revision: 1.5 $
* $Date: 1998/11/09 14:48:14 $
* $Log: $
*
* $TALog: client_listen.h,v $
* Revision 1.5 1998/11/09 14:48:14 wenjian
* In an effort to make a new directory structure for TPCC, this delta
* creates two directories: tpcc/client and tpcc/server. All the files
* for this revision are copied from tpcc/sp-tpcc without any change.
* Further change may be needed for some files due to the change of
* the directory structure.
* [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1]
*
* Revision 1.1 1997/04/20 11:57:55 oz
* - This is the code base modified at IBM Poughkeepsie
* by Ofer Zajicek and Radha Sivaramakrishnan for the
* SP scaling test for TPCC.
* [added by delta oz-19782-TPCC-add-ibm-sp-code,r1.1]
*
* Revision 1.1 1995/07/09 18:12:10 oz
* - Modified the client side of the TPCC benchmark to have multithreaded
* clients. There is a terminal process for each terminal -- when
* not using the terminal emulator each terminal process emulates one

```

```

* terminal. The terminal processes communication with the client
* process using a unix socket.
*
* On the client side there is a thread for each terminal process.
* That thread receives the request from the terminal and puts it on
* a queue. There is one processing thread that dequeues the requests
* and sends them to the server for processing.
* [added by delta oz-15875-TPCC-reduce-the-number-of-clients,r1.1]
*
*/

```

```

/* client_listen.h
*/

```

```

#ifndef TPCC_CLIENT_LISTEN_H
#define TPCC_CLIENT_LISTEN_H

```

```

void make_connections(void *argP);

```

```

#endif /* TPCC_CLIENT_LISTEN_H */

```

client main.c

```

#include "string.h"
#include "tpcc.h"

```

```

extern void client_init(int infd, int outfd, int w_id, int d_id, void *conP);
extern void client_login(int infd, int outfd, int *w_idP, int *d_idP);

```

```

main()
{
    int w_id, d_id;
    client_login(0, 1, &w_id, &d_id);
    client_init(0, 1, w_id, d_id, (void *)0);
}

```

```

int send_new_order(void *contextP, NewOrder_data *data) {
    int i;

```

```

    data->s_W_ID = 11;
    data->s_D_ID = 22;
    data->s_C_ID = 3333;
    strcpy((char *)data->s_C_LAST, "1234567890123456");
    strcpy((char *)data->s_C_CREDIT, "BC");
    data->s_C_DISCOUNT = 0.1556;
    data->s_O_ID = 4444;
    strcpy((char *)data->s_O_ENTRY_D, "1992-10-2 12:33:11");
    strcpy((char *)data->s_status_line, "123456789012345678901234");
    data->s_total_amount = 12.98;
    data->s_transtatus = 0;
    data->s_W_TAX = 0.1234;
    data->s_D_TAX = 0.5678;

```

```

    for (i=0; i < data->s_O_OL_CNT; i++) {
        strcpy((char *)data->item[i].s_I_NAME, "123456789012345678901234");
        data->item[i].s_S_QUANTITY = i + 1;
        data->item[i].s_brand_generic[0] = 'B';
        data->item[i].s_I_PRICE = i + 1;
        data->item[i].s_OL_AMOUNT = i + 1;
    }
    return 0;
}

```

```

int send_payment(void *contextP, Payment_data *data) {

```

```

    data->s_W_ID = 11;
    data->s_D_ID = 22;
    data->s_C_ID = 3333;
    data->s_C_W_ID = 44;
    data->s_C_D_ID = 55;
    data->s_H_AMOUNT = 9.55;
    strcpy((char *)data->s_W_STREET_1, "12345678901234567890");
    strcpy((char *)data->s_W_STREET_2, "12345678901234567890");
    strcpy((char *)data->s_W_CITY, "12345678901234567890");
    strcpy((char *)data->s_W_STATE, "PR");
    strcpy((char *)data->s_W_ZIP, "123456789");
    strcpy((char *)data->s_D_STREET_1, "12345678901234567890");
    strcpy((char *)data->s_D_STREET_2, "12345678901234567890");
    strcpy((char *)data->s_D_CITY, "12345678901234567890");
    strcpy((char *)data->s_D_STATE, "PR");
    strcpy((char *)data->s_D_ZIP, "123456789");
    strcpy((char *)data->s_C_FIRST, "1234567890123456");
    strcpy((char *)data->s_C_MIDDLE, "12");
    strcpy((char *)data->s_C_LAST, "1234567890123456");
    strcpy((char *)data->s_C_STREET_1, "12345678901234567890");
    strcpy((char *)data->s_C_STREET_2, "12345678901234567890");
    strcpy((char *)data->s_C_CITY, "12345678901234567890");
    strcpy((char *)data->s_C_STATE, "PR");
    strcpy((char *)data->s_C_ZIP, "123456789");
    strcpy((char *)data->s_C_PHONE, "1234567890123456");
    strcpy((char *)data->s_C_SINCE, "1992-23-22 21:11:11");
    strcpy((char *)data->s_H_DATE, "1992-10-2 12:33:11");
    strcpy((char *)data->s_C_CREDIT, "BC");
    data->s_C_CREDIT_LIM = 5000;
    data->s_C_DISCOUNT = 0.10;

```

```

    data->s_C_BALANCE = 122.10;
    strcpy((char *)data->s_C_DATA,
"1234567890123456789012345678901234567890123456789012345678901234567890");
    return 0;
}

```

```

int send_order_status(void *contextP, OrderStatus_data *data) {
    int i;

```

```

    data->s_W_ID = 11;
    data->s_D_ID = 22;
    data->s_C_ID = 3333;
    strcpy((char *)data->s_C_FIRST, "1234567890123456");
    strcpy((char *)data->s_C_MIDDLE, "12");
    strcpy((char *)data->s_C_LAST, "1234567890123456");
    data->s_C_BALANCE = 122.10;
    data->s_O_ID = 44;
    strcpy((char *)data->s_O_ENTRY_D, "1992-10-2 12:33:11");
    data->s_O_CARRIER_ID = 55;
    data->s_ol_cnt = 10;

```

```

    for (i=0; i < data->s_ol_cnt; i++) {
        data->item[i].s_OL_SUPPLY_W_ID = i + 1;
        data->item[i].s_OL_I_ID = i + 1;
        data->item[i].s_OL_QUANTITY = i + 1;
        data->item[i].s_OL_AMOUNT = i + 1;
        strcpy((char *)data->item[i].s_OL_DELIVERY_D, "1992-10-2 12:33:11");
    }
    return 0;
}

```

```

int send_delivery(void *contextP, Delivery_data *data) {
    strcpy((char *)data->s_exec_status, "Delivery has been queued");
    return 0;
}

```

```

int send_stock_level(void *contextP, StockLevel_data *data) {
    data->s_low_stock = 22;
    return 0;
}

```

client utils.c

```

/*
*
* client_utils.c
*

```

```

* $Revision: 1.9 $
* $Date: 1999/05/06 21:28:26 $
* $Log: $
*

```

```

* $TALog: client_utils.c.v$
* Revision 1.9 1999/05/06 21:28:26 oz
* - Removed all the .. from the includes
* - Added -I.. to the makefiles instead
* - Moved all the thread related code and connection
* selection to serverMon.c
* [from r1.7 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5]
*

```

```

* Revision 1.7 1998/12/11 16:37:57wenjian
* Move some common functions from client/client_utils.c to common/tpcc_utils.c.
* In this version, we only move time_diff_ms(). Need some work in order to
* move other functions like ERROUT.
*

```

```

* - Move time_diff_ms() to common/tpcc_utils.c
* [from r1.6 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.2]
*

```

```

* Revision 1.6 1998/11/09 16:59:36wenjian
* In this revision, most of the changes are related to the directory of header
* files after directory reorganization. Other changes include adding or removing
* files to put them in the right directories. Makefiles are written for NT
* platform so that nmake is working on NT now. Need a top level Makefile for all
* the directories.
* [from r1.5 by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.2]
*

```

```

* Revision 1.5 1998/11/09 14:48:14wenjian
* In an effort to make a new directory structure for TPCC, this delta
* creates two directories: tpcc/client and tpcc/server. All the files
* for this revision are copied from tpcc/sp-tpcc without any change.
* Further change may be needed for some files due to the change of
* the directory structure.
* [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1]
*

```

```

* Revision 1.9 1998/10/08 14:18:00dongfeng
* Add codes for doing web-based tpcc.
* [from r1.7 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.1]
*

```

```

* Revision 1.7 1998/04/29 19:47:42wenjian
* - Add prefix_sprintf
* - Remove ENCINA_C_CALLING_CONVENTION from err_printf
* [from r1.6 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.1]
*

```

```

* Revision 1.6 1998/02/17 22:07:00wenjian
* Minor changes for NT

```



```

* [from r1.5 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.1]
*
* Revision 1.5 1998/01/24 14:17:04 oz
* - User server name to identify server and name delivery file
* - Use env variable HOME instead of /home/encina if HOME is set
*
* - Flush the logfile after each write
* [from r1.4 by delta oz-21687-TPCC-use-server-name-to-identify-process,r1.1]
*
* Revision 1.4 1998/01/23 15:07:46 oz
* - Updated the SP TPCC directory to the latest files used
* during the SP tpcc audit.
* [from r1.3 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
*
*
* client_utils.c
* Generic utilities used by the client processes
*/

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

#ifdef solaris
#include <dce/pthread.h>
#else /* solaris */
#include <pthread.h>
#endif

#include "common/databuf.h"
#include "client_utils.h"
#include "common/do_tpcc.h"
#include "common/tpcc_type.h"

#define CASE(a) case a: retVal = #a; break

int print_thread_id = 1;
extern int user_id;
extern char *user_code;
/*
 * Translate the tpcc return code to a string value
 */
static char *TpccRcToStr(rc)
tpcc_rc_t rc;
{
    char *retVal;
    switch (rc) {
        CASE(INVALID_NEWO);
        CASE(INVALID_HANDLE);
        CASE(SQL_ERROR);
        CASE(TRPC_ERROR);
        CASE(DCE_ERROR);
        CASE(NO_SUCH_LAST_NAME);
        CASE(INVALID_TRAN_TYPE);
        CASE(TPCC_ERROR_BEGIN_NEWO);
        CASE(TPCC_ERROR_DECL_NEWO_SEL_ITEM);
        CASE(TPCC_ERROR_OPEN_NEWO_SEL_ITEM);
        CASE(TPCC_ERROR_OPEN_DIST_NEWO_SEL_ITEM);
        CASE(TPCC_ERROR_FETCH_NEWO_SEL_ITEM);
        CASE(TPCC_ERROR_FETCH_DIST_NEWO_SEL_ITEM);
        CASE(TPCC_ERROR_PREP_NEWO_SEL_STCK);
        CASE(TPCC_ERROR_DECL_NEWO_SEL_STCK);
        CASE(TPCC_ERROR_OPEN_NEWO_SEL_STCK);
        CASE(TPCC_ERROR_OPEN_DIST_NEWO_SEL_STCK);
        CASE(TPCC_ERROR_FETCH_NEWO_SEL_STCK);
        CASE(TPCC_ERROR_FETCH_DIST_NEWO_SEL_STCK);
        CASE(TPCC_ERROR_NEWO_SELECT);
        CASE(TPCC_ERROR_NEWO_UPD_STCK);
        CASE(TPCC_ERROR_DIST_NEWO_UPD_STCK);
        CASE(TPCC_ERROR_NEWO_SELECT_2);
        CASE(TPCC_ERROR_DECL_NEWO_SEL_CUST);
        CASE(TPCC_ERROR_OPEN_NEWO_SEL_CUST);
        CASE(TPCC_ERROR_OPEN_DIST_NEWO_SEL_CUST);
        CASE(TPCC_ERROR_FETCH_NEWO_SEL_CUST);
        CASE(TPCC_ERROR_FETCH_DIST_NEWO_SEL_CUST);
        CASE(TPCC_ERROR_DECL_NEWO_SEL_DIST);
        CASE(TPCC_ERROR_OPEN_NEWO_SEL_DIST);
        CASE(TPCC_ERROR_OPEN_DIST_NEWO_SEL_DIST);
        CASE(TPCC_ERROR_FETCH_NEWO_SEL_DIST);
        CASE(TPCC_ERROR_FETCH_DIST_NEWO_SEL_DIST);
        CASE(TPCC_ERROR_FETCH_DIST_NEWO_SEL_DIST);
        CASE(TPCC_ERROR_PREP_NEWO_INS_OL);
        CASE(TPCC_ERROR_DECL_NEWO_INS_OL);
        CASE(TPCC_ERROR_OPEN_NEWO_INS_OL);
        CASE(TPCC_ERROR_OPEN_DIST_NEWO_INS_OL);
        CASE(TPCC_ERROR_PUT_NEWO_INS_OL);
        CASE(TPCC_ERROR_PUT_DIST_NEWO_INS_OL);
        CASE(TPCC_ERROR_DECL_NEWO_SEL_WARE);
        CASE(TPCC_ERROR_OPEN_NEWO_SEL_WARE);
        CASE(TPCC_ERROR_OPEN_DIST_NEWO_SEL_WARE);
        CASE(TPCC_ERROR_FETCH_NEWO_SEL_WARE);
        CASE(TPCC_ERROR_FETCH_DIST_NEWO_SEL_WARE);
        CASE(TPCC_ERROR_EXECUTE_NEWO_UPD_INS);
        CASE(TPCC_ERROR_UPDATE_NEWO_NEXT_OID);
        CASE(TPCC_ERROR_PREP_NEWO_INS);
        CASE(TPCC_ERROR_EXECUTE_DIST_NEWO_INS);
        CASE(TPCC_ERROR_EXECUTE_NEWO_COMMIT);
        CASE(TPCC_ERROR_ROLLBACK_NEWO);
    }
}

```

```

CASE(TPCC_ERROR_REMOTE_OL_SELECT);
CASE(TPCC_ERROR_REMOTE_OL_UPDATE);
CASE(TPCC_ERROR_OPEN_ORDS_CNT_CID);
CASE(TPCC_ERROR_FETCH_ORDS_CNT_CID);
CASE(TPCC_ERROR_OPEN_ORDS_SEL_CLAST);
CASE(TPCC_ERROR_FETCH_ORDS_SEL_CLAST);
CASE(TPCC_ERROR_OPEN_ORDS_SEL_CID);
CASE(TPCC_ERROR_FETCH_ORDS_SEL_CID);
CASE(TPCC_ERROR_OPEN_ORDS_SEL_OLDORD);
CASE(TPCC_ERROR_FETCH_ORDS_OLDORD);
CASE(TPCC_ERROR_OPEN_ORDS_SEL_OL);
CASE(TPCC_ERROR_FETCH_ORDS_SEL_OL);
CASE(TPCC_ERROR_EXECUTE_ORDS_COMMIT);
CASE(TPCC_ERROR_OPEN_DELIVERY_OLDEST_OID);
CASE(TPCC_ERROR_FETCH_DELIVERY_OLDEST_OID);
CASE(TPCC_ERROR_EXECUTE_DELIVERY_COMMIT);
CASE(TPCC_ERROR_OPEN_DELIVERY_SEL_ORD);
CASE(TPCC_ERROR_FETCH_DELIVERY_SEL_ORD);
CASE(TPCC_ERROR_OPEN_DELIVERY_SEL_SUM_OL);
CASE(TPCC_ERROR_FETCH_DELIVERY_SEL_SUM_OL);
CASE(TPCC_ERROR_EXECUTE_DELIVERY_EXEC_DVRY);
CASE(TPCC_ERROR_SELECT_DELIVERY_ORDER_ID);
CASE(TPCC_ERROR_SELECT_DELIVERY_CARRIER_ID);
CASE(TPCC_ERROR_SELECT_DELIVERY_BALANCE);
CASE(TPCC_ERROR_OPEN_STOCKLEVEL_SEL_OID);
CASE(TPCC_ERROR_FETCH_STOCKLEVEL_SEL_OID);
CASE(TPCC_ERROR_OPEN_STOCKLEVEL_CNT_SID);
CASE(TPCC_ERROR_FETCH_STOCKLEVEL_CNT_SID);
CASE(TPCC_ERROR_OPEN_STOCKLEVEL_FIND);
CASE(TPCC_ERROR_FETCH_STOCKLEVEL_FIND);
CASE(TPCC_ERROR_EXECUTE_STOCKLEVEL_COMMIT);
CASE(TPCC_ERROR_OPEN_PAYMENT_CNT_CID);
CASE(TPCC_ERROR_FETCH_PAYMENT_CNT_CID);
CASE(TPCC_ERROR_OPEN_PAYMENT_SEL_CLAST);
CASE(TPCC_ERROR_FETCH_PAYMENT_SEL_CLAST);
CASE(TPCC_ERROR_OPEN_PAYMENT_SEL_CID);
CASE(TPCC_ERROR_FETCH_PAYMENT_SEL_CID);
CASE(TPCC_ERROR_DECL_PAYMENT_SEL_DIST);
CASE(TPCC_ERROR_OPEN_PAYMENT_SEL_DIST);
CASE(TPCC_ERROR_OPEN_DIST_PAYMENT_SEL_DIST);
CASE(TPCC_ERROR_FETCH_PAYMENT_SEL_DIST);
CASE(TPCC_ERROR_FETCH_DIST_PAYMENT_SEL_DIST);
CASE(TPCC_ERROR_DECL_PAYMENT_SEL_WARE);
CASE(TPCC_ERROR_OPEN_PAYMENT_SEL_WARE);
CASE(TPCC_ERROR_OPEN_DIST_PAYMENT_SEL_WARE);
CASE(TPCC_ERROR_FETCH_PAYMENT_SEL_WARE);
CASE(TPCC_ERROR_FETCH_DIST_PAYMENT_SEL_WARE);
CASE(TPCC_ERROR_EXECUTE_PAYMENT_UPD_CUST_LAST);
CASE(TPCC_ERROR_EXECUTE_PAYMENT_UPD_CUST_ID);
CASE(TPCC_ERROR_COMMIT_PAYMENT_UPD_CUST);
CASE(TPCC_ERROR_SELECT_PAYMENT_W_YTD);
CASE(TPCC_ERROR_SELECT_PAYMENT_D_YTD);
CASE(TPCC_ERROR_BEGIN_PAYMENT);
CASE(TPCC_ERROR_EXECUTE_PAYMENT_COMMIT);

    default: retVal = "-Unknown-"; break;
}
return(retVal);
}

/*
 * get_thread_id
 * A function that returns the thread ID of the current thread
 */
int get_thread_id()
{
#ifdef WEB_TPCC_CLIENT
    return(GetCurrentThreadId());
#else
    pthread_t thread = pthread_self();
    int thread_id = pthread_getunique_np(&thread);
    return(thread_id);
#endif
}

#define A_CASE(a,b) case a: retVal = b; break
/*
 * Translate the transaction code to its name - for formatting
 */
char *clientUtils_TranCodeToName(type)
int type;
{
    char *retVal = "-Unknown-";
    switch (type) {
        A_CASE(NEWO_TRANS, "NEWO");
        A_CASE(PAYMENT_TRANS, "PAYMN");
        A_CASE(ORDER_STAT_TRANS, "ORDER");
        A_CASE(DELIVERY_TRANS, "DELIV");
        A_CASE(STOCK_TRANS, "STOCK");
    }
    return(retVal);
}

/*
 * Print the return status of a TPCC transaction
 * and the corresponding SQL codes and ISAM codes
 */
void clientUtils_ReportReturn(msg, statusP)
char *msg;
data_header *statusP;

```



```

*
*
* $TALog:client_utils.h,v$
* Revision 1.11 1999/05/06 21:28:26 oz
* - Removed all the .. from the includes
* - Added -L. to the makefiles instead
* - Moved all the thread related code and connection
* selection to serverMon.c
* [from r1.8 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5]
*
* Revision 1.8 1998/12/14 20:27:54wenjian
* Made corresponding changes due to data structure change of tran_info_t.
*
* - Change data structure tran_info_t
* [from r1.7 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server,r1.3]
*
* Revision 1.7 1998/12/11 16:14:19wenjian
* Add code for checking statistic data in a single variable and collecting
* statistic data based on iStatsFrequency.
*
* - Add total_num_trans to tran_info_t;
* [from r1.6 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server,r1.1]
*
* Revision 1.6 1998/11/09 16:59:37wenjian
* In this revision, most of the changes are related to the directory of header
* files after directory reorganization. Other changes include adding or removing
* files to put them in the right directories. Makefiles are written for NT
* platform so that nmake is working on NT now. Need a top level Makefile for all
* the directories.
* [from r1.5 by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.2]
*
* Revision 1.5 1998/11/09 14:48:15wenjian
* In an effort to make a new directory structure for TPCC, this delta
* creates two directories: tpcc/client and tpcc/server. All the files
* for this revision are copied from tpcc/sp-tpcc without any change.
* Further change may be needed for some files due to the change of
* the directory structure.
* [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1]
*
* Revision 1.4 1998/11/09 14:26:51wenjian
* Change enc_status to a data structure that has fields:
* - Status code
* - Line Number
* - File Name
* - Encina Error Code
* - Error Msg
* Remove statusMsgs in web_tpcc.c
*
* Add definition of enc_status_t
* [from r1.19 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.6]
*
* Revision 1.19 1998/10/22 21:13:07wenjian
* [merge of changes from 1.11 to 1.18 into 1.14]
*
* Revision 1.18 1998/10/22 19:18:32dongfeng
* [from r1.17 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.2]
*
* Revision 1.17 1998/10/08 14:18:00dongfeng
* Add codes for doing web-based tpcc.
* [from r1.11 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.1]
*
* Revision 1.14 1998/08/18 14:38:39wenjian
* Minor change
* [from r1.13 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.4]
*
* Revision 1.13 1998/08/18 13:35:42wenjian
* Remove NUM_NEXT_REPORTS since it is no use.
* [from r1.11 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.7]
*
* Revision 1.11 1998/06/17 15:28:51wenjian
* Add 'double time' in struct total_tran_count_t.
* [from r1.10 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.2]
*
* Revision 1.10 1998/04/29 19:47:43wenjian
* - Define ENDMMSG marking the end of socket message between tpcc_client
* and tpcc_monitor
* - Remove ENCINA_C_CALLING_CONVENTION from err_printf
* [from r1.9 by delta wenjian-22495-TPCC-add-new-feature-to-monitor-tpcc-clients,r1.1]
*
* Revision 1.9 1998/02/17 22:13:41wenjian
* [merge of changes from 1.6 to 1.7 into 1.8]
*
* Revision 1.7 1998/02/17 16:04:41 oz
* - Split the login into two parts to allow for special logins
* - If the warehouse ID is 0, this is a speciallogin to
* query the client for status
* [from r1.6 by delta oz-21864-TPCC-split-client-login-screen,r1.1]
*
* Revision 1.8 1998/02/17 22:07:00wenjian
* Minor changes for NT
* [from r1.6 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.1]
*
* Revision 1.6 1998/01/26 16:43:32 oz
* - Removed the code for collecting stats in the client
* and dumping them before exit.
*
* - Removed timeP and time_allocated from thread_info_t
* [from r1.5 by delta oz-21691-TPCC-remove-client-stats-code,r1.1]

```

```

* Revision 1.5 1998/01/26 16:19:23 oz
* - moved all the code pertaining to the background
* thread to its own file and all the data structures
* to client_utils.h
* [from r1.4 by delta oz-21689-TPCC-move-client-bg-thread-to-separate-file,r1.1]
*
* Revision 1.4 1998/01/23 15:07:47 oz
* - Updated the SP TPCC directory to the latest files used
* during the SP tpcc audit.
* [from r1.3 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
*
*
* client_utils.h
* Generic utilities used by the client processes
*/

#ifndef TPCC_CLIENT_UTILS_H
#define TPCC_CLIENT_UTILS_H

#include "common/tpcc_type.h"
#include <stdio.h>
#include <time.h>
#include <encina/encina.h>
#include "client.h"
#ifdef WIN32
#include <winsock.h>
#endif

/*
* err_printf
* Print a string to stderr after prefixing it with the client
* info and the current time.
* logprintf
* Prints as above to the log file.
*/

#ifdef WEB_TPCC_CLIENT
extern FILE * errtpcc;
#endif
extern FILE * logtpcc;
extern char log_file_name[];
extern void logprintf( char *format, ...);
extern void err_printf( char *format, ...);
extern void prefix_sprintf( char *buf, char *format, ...);

/* tran_timing_t: for debug:
* Keep track of the timestamps of all the transactions
* and dump it out upon exit. There is an array of timestamps
* per thread and each thread dumps it when it exits.
*/
typedef struct {
int server;
int terminal;
int tran;
int sub_tran; /* Subclass: for NewOrder and payment: 1=>hasRemote */
struct timeval start; /* Time received from terminal */
struct timeval send; /* Time the RPC was made (explicit only) */
struct timeval svr_start; /* Time received by server */
struct timeval svr_done; /* Time sent by server */
struct timeval end; /* Time sent to terminal */
int num_rms; /* Number of RMs the tran involved */
int tran_failed;
} tran_timing_t;

typedef enum {
thread_state_init = 0,
thread_state_called,
thread_state_sent,
thread_state_received,
thread_state_returned,
thread_state_done
} thread_state_t;

#define NUM_STATES thread_state_done
#define ENDMMSG "...." /* a special string to mark the end of a message */

typedef struct {
thread_state_t state;
int tran;
struct timeval init, called, sent, received, returned, done;
int printed, done_printed;
} thread_descr_t;

typedef struct {
int num;
int errs;
double RTtotal[2];
int RTcount;
} tran_info_t;
/*
* total_tran_count_t
*
* structure that holds the total count of transaction of each type
* as well as the reposne times.
*/
typedef struct {
tran_info_t tran[MAX_TRAN_TYPE + 1];
int errors;

```

```

double time; /* used for tools/tpcc_monitor.c */
} total_tran_count_t;

/* enc_status_t
 * structure that holds error information
 */
typedef struct {
    int status;
    int line;
    char file[268];
    unsigned long encinaError;
    char errorMsg[ENCINA_MAX_STATUS_STRING_SIZE];
} enc_status_t;

/*
 * thread_info_t
 *
 * per thread information kept by this module
 */
typedef struct {
    int thread_index;
    int thread_id;
    int initialized;
    tran_timing_t last_tran;
    int num_trans;
    int consecutive_errors;
    thread_descr_t descr;
    tran_info_t tran[MAX_TRAN_TYPE + 1];
    int done;
} thread_info_t;

int time_diff_ms(struct timeval *t2, struct timeval *t1);

extern int debug;
#define DPRINT(args) if (debug) err_printf args

extern MUTEX_Tinit_lock;
extern int info_list_len;
extern thread_info_t **info_list; /* List of all the thread info */

/*
 * A global variable by which the process would like to
 * identify itself in the prefix to output
 */
extern int user_id;

/*
 * clientUtils_ReportReturn
 * Called when a transaction is returned in order to error codes
 */
extern void clientUtils_ReportReturn(char *msg, data_header *statusP);

#define CHECK_ENVIRON(str, var) if (str == NULL) { fprintf(ERROR, \
    "%s environment variable is not defined.\n", var); }

char *clientUtils_TranCodeToName(int type);

#endif /* TPCC_CLIENT_UTILS_H */

datbuf.h

/*
 * datbuf.h
 *
 * $Revision: 1.1 $
 * $Date: 1998/11/06 21:10:11 $
 * $Log: datbuf.h,v $
 * Revision 4.2 95/05/16 10:55:31 10:55:31tpcc (TPCC Benchmark)
 * Added necessary RCS ident strings
 *
 * Revision 4.1 95/05/09 15:21:02 15:21:02true (Scott Truesdale)
 * New code from Transarc - initial version
 *
 * Revision 3.2 95/04/03 17:43:09 17:43:09true (Scott Truesdale)
 * Changes from Transarc - added sql error handling in client; cleaned up debug handling with macros; added
 * check on db paramters via call to server.
 *
 * Revision 3.1 95/04/03 15:10:30 15:10:30true (Scott Truesdale)
 * Base of rev 3 - shipped to transarc
 *
 *
 *
 * $TALog: datbuf.h,v $
 * Revision 1.1 1998/11/06 21:10:11dongfeng
 * - Move all files common to client and server totpcc/common
 * directory
 * [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
 *
 * Revision 1.3 1998/10/22 15:33:04wenjian
 * Make changes to Encina server code to connect with SQL server and add
 * callsql.c and sql directory.
 *
 * Add ERR_BAD_ITEM_ID, which is returned by SQLnew and same as INVALID_NEWO
 * [from r1.2 by delta wenjian-23529-TPCC-integrate-with-SQL-server,r1.1]
 *
 * Revision 1.2 1998/01/23 15:07:47 oz
 * - Updated the SP TPCC directory to the latest files used
 *
 * during the SP tpcc audit.
 * [from r1.1 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
 *
 * Revision 1.1 1997/04/20 11:57:57 oz
 * - This is the code base modified at IBM Poughkeepsie
 * by Ofer Zajicek and Radha Sivaramakrishnan for the
 * SP scaling test for TPCC.
 * [added by delta oz-19782-TPCC-add-ibm-sp-code,r1.1]
 *
 * Revision 1.31 1995/10/30 19:10:54 oz
 * [merge of changes from 1.29 to 1.30 into 1.27]
 *
 * Revision 1.30 1995/10/27 15:41:30 oz
 * - Modified the tpc-c code to work with the new informix
 * sql code that is in ex_trans.ec
 * [from r1.29 by delta oz-16761-TPCC-modify-code-to-work-with-oracle,r1.1]
 *
 * Revision 1.27 1995/10/20 18:44:30ctipper
 * [merge of changes from 1.17 to 1.25 into 1.22]
 *
 * Revision 1.25 1995/10/20 18:15:34ctipper
 * Incorporate changes per code review.
 *
 * - add DISTRIBUTED_TRAN_FAILED,TPCC_DB_INFO_PARTIAL, and
 * TPCC_DB_INFO_FAILED error codes to tpcc_rc_t
 * - got rid of MAX_NUM_SERVERS variables
 * [from r1.23 by delta ctipper-16547-TPCC-more-distributed-trans,r1.2]
 *
 * Revision 1.23 1995/10/13 17:00:26ctipper
 * This delta encompasses all changes necessary to do distributed, XA
 * transactions with the TPCC benchmark. This includes the changes
 * necessary to build with Informix version 6.
 *
 * Each client still talks to only one server, however, if a distributed
 * transaction is necessary, the client sends the request to a different
 * interface of that server which then forwards all or part of the
 * request on to the appropriate remote server.
 *
 * - added new error codes to the tpcc_rc_t enumeration.
 * - defined MAX_NUM_SERVERS to be 10
 * [from r1.19 by delta ctipper-16547-TPCC-more-distributed-trans,r1.1]
 *
 * Revision 1.19 1995/09/20 21:02:39 oz
 * -Corrected code for the payment transaction
 * - The distributed case now no longer uses
 * stored procedures
 * [from r1.18 by delta oz-16547-TPCC-add-distributed-transactions,r1.2]
 *
 * Revision 1.18 1995/09/20 17:51:10 oz
 * - Added distributed transactions for the new order and
 * payment transaction
 *
 * - Added new error codes
 * [from r1.17 by delta oz-16547-TPCC-add-distributed-transactions,r1.1]
 *
 * Revision 1.22 1995/10/02 20:31:07 oz
 * - Corrected definition of ERROR()
 * [from r1.21 by delta oz-16638-tpcc-modify-terminal-for-RTE,r1.3]
 *
 * Revision 1.21 1995/10/02 18:51:45 oz
 * - Added definitions needed for utils.c and liberty.c
 * [from r1.20 by delta oz-16638-tpcc-modify-terminal-for-RTE,r1.2]
 *
 * Revision 1.20 1995/10/02 15:52:35 oz
 * - Modified the TPC-C benchmark to be compatible with the RTE.
 * - There are now 3 terminal processes:
 * emulator: the old terminal process with a built in
 * simple emulator
 * curses: An interactive terminal process using curses
 * liberty: An interactive terminal process to be used with
 * the RTE compatible with the liberty freedom terminal.
 *
 * - Define TRUE and FALSE only if they are not already defined.
 * (curses.h defines TRUE)
 * - Removed READ_TO_DATE and YEAR_TO_SECOND
 * - Added term_type_t
 * - Added
 * GOOD_INPUT (0)
 * WRONG_INPUT (10)
 * [from r1.17 by delta oz-16638-tpcc-modify-terminal-for-RTE,r1.1]
 *
 * Revision 1.17 1995/07/28 15:28:23 oz
 * - Added a -null and -no_marshallling option to TPCC
 *
 * - Added INVALID_TRAN_TYPE return code
 * [from r1.16 by delta oz-16070-TPCC-add-null-and-marshalling-test,r1.1]
 *
 * Revision 1.16 1995/07/18 17:02:38 oz
 * - Added a DCE_ERROR error code
 * [from r1.15 by delta oz-15938-TPCC-add-dce-only-client,r1.1]
 *
 * Revision 1.15 1995/05/22 19:50:48shl
 * [merge of changes from 1.12 to 1.13 into 1.14]
 *
 * Revision 1.13 1995/05/18 15:11:27 oz
 * [from r1.12 by delta oz-15290-TPCC-incorporate-hp-drop-of-05-16-95,r1.1]
 *
 * Revision 1.14 1995/05/22 17:26:35ctipper
 * [merge of changes from 1.5 to 1.9 into 1.11]

```

```

*
* [*** log entries omitted ***]
*
*/

#ifndef __TPCC_DATABUF_H__
#define __TPCC_DATABUF_H__

#define I_NAME_LEN 24
#define I_DATA 50
#define W_NAME_LEN 10
#define ADDR_LEN 20
#define STATE_LEN 2
#define ZIP_LEN 9
#define DIST_INFO_LEN 24
#define S_DATA_LEN 50
#define D_NAME_LEN 10
#define H_DATA_LEN 24
#define CARRIER_LEN 2
#define C_LAST_LEN 17
#define C_MID_LEN 2
#define PHONE_LEN 16
#define CREDIT_LEN 2
#define C_DATA_LEN 500
#define BC_DTA_LEN 23

#define YEAR_TO_DATE 1
#define YEAR_TO_SECOND 2

#define ERROR(x) fprintf(stderr, "Error: %s\n", #x), exit(11)

#define MAX_STR_LEN 255
#define MAX_OL 15

#ifndef TRUE
#define TRUE 1
#endif
#ifndef FALSE
#define FALSE 0
#endif

#define CANCEL -1

#define DATETIME_LEN 19

#define D_PER_W 10

#define COLLECTOR 1 /* ctipper 5/3/95 */

#define ERR_BAD_ITEM_ID 1 /* copied from sql/tpcc.h */
#define RPC_ERROR -2
#define SUCCESS_CODE 0

#define CHAR_NULL '\0' /* strue 1/23/95 */

typedef enum {
liberty_term,
curses_term,
emulator_term
} term_type_t;

typedef enum {
TPCC_SUCCESS = 0,
GOOD_INPUT = 0,

INVALID_NEWO = 100,
SQL_ERROR = 2,
TRPC_ERROR = 3,
DCE_ERROR = 4,
NO_SUCH_LAST_NAME = 5,
INVALID_TRAN_TYPE = 6,
INVALID_HANDLE = 7,

WRONG_INPUT = 10,

DISTRIBUTED_TRAN_FAILED = 15,

TPCC_DB_INFO_PARTIAL = 20,
TPCC_DB_INFO_FAILED,

TPCC_ERROR_BEGIN_NEWO = 110,

TPCC_ERROR_DECL_NEWO_SEL_ITEM,
TPCC_ERROR_OPEN_NEWO_SEL_ITEM,
TPCC_ERROR_OPEN_DIST_NEWO_SEL_ITEM,
TPCC_ERROR_FETCH_NEWO_SEL_ITEM,
TPCC_ERROR_FETCH_DIST_NEWO_SEL_ITEM,
TPCC_ERROR_PREP_NEWO_SEL_STCK,
TPCC_ERROR_DECL_NEWO_SEL_STCK,
TPCC_ERROR_OPEN_NEWO_SEL_STCK,
TPCC_ERROR_OPEN_DIST_NEWO_SEL_STCK,
TPCC_ERROR_FETCH_NEWO_SEL_STCK,
TPCC_ERROR_FETCH_DIST_NEWO_SEL_STCK,
TPCC_ERROR_NEWO_SELECT,
TPCC_ERROR_NEWO_UPD_STCK,
TPCC_ERROR_DIST_NEWO_UPD_STCK,
TPCC_ERROR_NEWO_SELECT_2,
TPCC_ERROR_DECL_NEWO_SEL_CUST,
TPCC_ERROR_OPEN_NEWO_SEL_CUST,

TPCC_ERROR_OPEN_DIST_NEWO_SEL_CUST,
TPCC_ERROR_FETCH_NEWO_SEL_CUST,
TPCC_ERROR_FETCH_DIST_NEWO_SEL_CUST,
TPCC_ERROR_DECL_NEWO_SEL_DIST,
TPCC_ERROR_OPEN_NEWO_SEL_DIST,
TPCC_ERROR_FETCH_NEWO_SEL_DIST,
TPCC_ERROR_FETCH_DIST_NEWO_SEL_DIST,
TPCC_ERROR_PREP_NEWO_INS_OL,
TPCC_ERROR_DECL_NEWO_INS_OL,
TPCC_ERROR_OPEN_NEWO_INS_OL,
TPCC_ERROR_OPEN_DIST_NEWO_INS_OL,
TPCC_ERROR_PUT_NEWO_INS_OL,
TPCC_ERROR_PUT_DIST_NEWO_INS_OL,
TPCC_ERROR_DECL_NEWO_SEL_WARE,
TPCC_ERROR_OPEN_NEWO_SEL_WARE,
TPCC_ERROR_OPEN_DIST_NEWO_SEL_WARE,
TPCC_ERROR_FETCH_NEWO_SEL_WARE,
TPCC_ERROR_FETCH_DIST_NEWO_SEL_WARE,
TPCC_ERROR_EXECUTE_NEWO_UPD_INS,
TPCC_ERROR_UPDATE_NEWO_NEXT_OID,
TPCC_ERROR_PREP_NEWO_INS,
TPCC_ERROR_EXECUTE_DIST_NEWO_INS,
TPCC_ERROR_EXECUTE_NEWO_COMMIT,
TPCC_ERROR_ROLLBACK_NEWO,
TPCC_ERROR_REMOTE_OL_SELECT,
TPCC_ERROR_REMOTE_OL_UPDATE,

TPCC_ERROR_OPEN_ORDS_CNT_CID = 200,
TPCC_ERROR_FETCH_ORDS_CNT_CID,
TPCC_ERROR_OPEN_ORDS_SEL_CLAST,
TPCC_ERROR_FETCH_ORDS_SEL_CLAST,
TPCC_ERROR_OPEN_ORDS_SEL_CID,
TPCC_ERROR_FETCH_ORDS_SEL_CID,
TPCC_ERROR_OPEN_ORDS_SEL_OLDORD,
TPCC_ERROR_FETCH_ORDS_OLDORD,
TPCC_ERROR_OPEN_ORDS_SEL_OL,
TPCC_ERROR_FETCH_ORDS_SEL_OL,
TPCC_ERROR_EXECUTE_ORDS_COMMIT,

TPCC_ERROR_OPEN_DELIVERY_OLDEST_OID = 300,
TPCC_ERROR_FETCH_DELIVERY_OLDEST_OID,
TPCC_ERROR_EXECUTE_DELIVERY_COMMIT,
TPCC_ERROR_OPEN_DELIVERY_SEL_ORD,
TPCC_ERROR_FETCH_DELIVERY_SEL_ORD,
TPCC_ERROR_OPEN_DELIVERY_SEL_SUM_OL,
TPCC_ERROR_FETCH_DELIVERY_SEL_SUM_OL,
TPCC_ERROR_EXECUTE_DELIVERY_EXEC_DVRY,
TPCC_ERROR_SELECT_DELIVERY_ORDER_ID,
TPCC_ERROR_SELECT_DELIVERY_CARRIER_ID,
TPCC_ERROR_SELECT_DELIVERY_BALANCE,

TPCC_ERROR_OPEN_STOCKLEVEL_SEL_OID = 400,
TPCC_ERROR_FETCH_STOCKLEVEL_SEL_OID,
TPCC_ERROR_OPEN_STOCKLEVEL_CNT_SID,
TPCC_ERROR_FETCH_STOCKLEVEL_CNT_SID,
TPCC_ERROR_OPEN_STOCKLEVEL_FIND,
TPCC_ERROR_FETCH_STOCKLEVEL_FIND,
TPCC_ERROR_EXECUTE_STOCKLEVEL_COMMIT,

TPCC_ERROR_OPEN_PAYMENT_CNT_CID = 500,
TPCC_ERROR_FETCH_PAYMENT_CNT_CID,
TPCC_ERROR_OPEN_PAYMENT_SEL_CLAST,
TPCC_ERROR_FETCH_PAYMENT_SEL_CLAST,
TPCC_ERROR_OPEN_PAYMENT_SEL_CID,
TPCC_ERROR_FETCH_PAYMENT_SEL_CID,
TPCC_ERROR_DECL_PAYMENT_SEL_DIST,
TPCC_ERROR_OPEN_PAYMENT_SEL_DIST,
TPCC_ERROR_OPEN_DIST_PAYMENT_SEL_DIST,
TPCC_ERROR_FETCH_PAYMENT_SEL_DIST,
TPCC_ERROR_FETCH_DIST_PAYMENT_SEL_DIST,
TPCC_ERROR_DECL_PAYMENT_SEL_WARE,
TPCC_ERROR_OPEN_PAYMENT_SEL_WARE,
TPCC_ERROR_OPEN_DIST_PAYMENT_SEL_WARE,
TPCC_ERROR_FETCH_PAYMENT_SEL_WARE,
TPCC_ERROR_FETCH_DIST_PAYMENT_SEL_WARE,
TPCC_ERROR_EXECUTE_PAYMENT_UPD_CUST_LAST,
TPCC_ERROR_EXECUTE_PAYMENT_UPD_CUST_ID,
TPCC_ERROR_COMMIT_PAYMENT_UPD_CUST,
TPCC_ERROR_SELECT_PAYMENT_W_YTD,
TPCC_ERROR_SELECT_PAYMENT_D_YTD,
TPCC_ERROR_BEGIN_PAYMENT,
TPCC_ERROR_EXECUTE_PAYMENT_COMMIT,
TPCC_ERROR_PAYMENT_UPD_CUST_BY_NAME,
TPCC_ERROR_PAYMENT_UPD_CUST_BY_ID,
TPCC_ERROR_PAYMENT_UPDATE_DIST,
TPCC_ERROR_PAYMENT_UPDATE_WH,
TPCC_ERROR_PAYMENT_INSERT_HISTORY,
TPCC_ERROR_EXECUTE_PAYMENT_WH_DIST
} tpcc_re_t;

typedef enum {
TPCC_DEADLOCK_MSG = 10,
TPCC_RETRY_MSG
} tpcc_msg_t;

#endif /* __TPCC_DATABUF_H__ */

```

debug.c

```
#include <stdio.h>
#include <string.h>
#include <stdarg.h>
#include <sys/stat.h>
#include <errno.h>
#include <math.h>
#include <time.h>
#include <fcntl.h>
#include <unistd.h>
#ifdef WIN32
#include <process.h>
#else
#include <termio.h>
#endif

int print_thread_id = 0;
int user_id = 1;
char *user_code = "C";

int get_thread_id()
{
    return(0);
}

/*
 * get_prefix
 * Format the output prefix for printing:
 * It contains the user_id, 'C' or 'T' depending on whether it
 * is a terminal or a client and optional a thread identifier
 * The prefix is written in the buffer passed in by the caller.
 */
void get_prefix(buffer)
char *buffer;
{
    if(print_thread_id) {
        int thread_id = get_thread_id();
        sprintf(buffer, "%s(%d-%s-%d)%s",
            user_id < 10 ? " " : "user_id < 100 ? " : "",
            user_id,
            user_code,
            thread_id,
            thread_id < 10 ? " " : "");
    } else {
        sprintf(buffer, "%s(%2d-%s)",
            user_id < 10 ? " " : "", user_id, user_code);
    }
}

/*
 * err_printf
 * A var-arg function that appends the current time and
 * other data to the print request and sends it to stderr
 */
void err_printf(char *format, ...)
{
    static int initialized = 0;
    static FILE *debug_f = NULL;
    time_t cur_time;
    char time_str[30];
    char line_prefix[50];
    va_list ap;

    va_start(ap, format);

    if (!initialized) {
        char fileName[45];
        initialized = 1;
        sprintf(fileName, "DebugFile.%d", getpid());
        debug_f = fopen(fileName, "w");
    }

    cur_time = time(&cur_time);
    strftime(time_str, 29, "%X", localtime(&cur_time));

    get_prefix(line_prefix);

    if (debug_f) {
        fprintf(debug_f, "%s %s - ", line_prefix, time_str);
        vfprintf(debug_f, format, ap);
        fflush(debug_f);
    }

    va_end(ap);
}

void set_client_debug_state(void *contextp, int state, int tran)
{
}

```

delivery.tacf

```
/*
 * Copyright (C) 1991, 1990 Transarc Corporation
 * All Rights Reserved

```

```
/*
/*
 * neworder.tacf -- attribute configuration file fortpcc server.
 * used for transparent binding
 *
 * $Revision: 1.1 $
 * $Date: 1998/11/06 21:10:11 $
 * $Log: tpcc.tacf,v $
 *
 * $TALog: delivery.tacf,v $
 * Revision 1.1 1998/11/06 21:10:11dongfeng
 * - Move all files common to client and server totppc/common
 * directory
 * [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
 *
 * Revision 1.1 1997/04/20 11:57:57 oz
 * - This is the code base modified at IBM Poughkeepsie
 * by Ofer Zajicek and Radha Sivaramakrishnan for the
 * SP scaling test for TPCC.
 * [added by delta oz-19782-TPCC-add-ibm-sp-code,r1.1]
 *
 * Revision 1.3 1996/01/12 16:06:44 oz
 * - Added transaction specific servers: there are 5 different interfaces
 * one for each transaction type.
 * [added by delta oz-16955-TPCC-add-transaction-specific-servers,r1.1]
 */

```

```
[implicit_handle(mon_handle_t handle)]
```

```
interface delivery
```

```
{
```

```
}
```

delivery.tidl

```
/*
 * id: Sid: $
 *
 * component_name: encina benchmarks
 *
 * the following functions list may not be complete.
 * functions defined by/via macros may not be included.
 *
 * functions:
 * <fill_me_in>
 *
 * origins: transarc corp.
 *
 * (c) copyright transarc corp. 1995, 1993
 * all rights reserved
 * licensed materials - property of transarc
 *
 * us government users restricted rights - use, duplication or
 * disclosure restricted by gsa adp schedule contract with transarc corp
 */
/*
 * history
 * $talog: $
 */
/*
 * delivery.tidl -- interface definition file fortpccserver.
 *
 * $Revision: 1.11 $
 * $Date: 1995/10/20 21:55:05 $
 * $Log: tpcc.tidl,v $
 */

```

```
[uuid(d714d8f8-2105-11cf-830f-0800093b9834)version(1.0)]
```

```
interface delivery
```

```
{
```

```
import "tpm/mon/mon_handle.idl";
```

```
import "tpcc_type.idl";
```

```
[nontransactional] void
```

```
impTPCCDelivery([in,out]delivery_data_t *dataP,
                [out] trpc_status_t *trpcStatus);
```

```
}
```

do_tpcc.c

```
/*
 * do_tpcc.c
 *
 * $Revision: 1.12 $
 * $Date: 1999/05/06 21:28:26 $
 * $Log: do_tpcc.c,v $
 *
 * $TALog: do_tpcc.c,v $
 * Revision 1.12 1999/05/06 21:28:26 oz
 * - Removed all the .. from the includes
 * - Added -I. to the makefiles instead
 * - Moved all the thread related code and connection
 * selection to serverMon.c
 * [from r1.8 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5]

```



```

int argc;
char *argv[];
{
    check_parms(argc,argv); /* Read and parse the command line parameters */

    err_printf("Client %d starting.\n", user_id);

    init_encina_client(user_id);
    enroll_client(user_id); /* enroll as a client */

    /*
     * Open log file
     */
    logtpcc = fopen(log_file_name, "w");
    print_header(argc, argv); /* Print a test header to the logfile */

    /*
     * Start the listening thread:
     * This call will not return
     */
    make_connections((void *)user_port);

    exit_program(0);
    return(0); /* to satisfy lint */
} /****** end of main *****/

/*=====*/
/*
 * User must supply user_id as a parm and all other parameters
 * as environment variables.
 */
/*-----*/
/*
 * Check Parameters
 * Check the parameters passed in.
 *
 * Not all the parameters are relevant for this executable.
 * This code is shared between the regular Encina Monitor
 * based TPC-C client and other test clients that do not
 * use the Encina Monitor. The type of this executable is
 * in client_type and is set to mon_client for the TPCC
 * Monitor based client (the audited client).
 */
/*-----*/
static void check_parms (argc,argv)
int argc;
char *argv[];
{
    char *host_name = getenv("HOST");
    char *home_dir = getenv("HOME");
    int next_arg = 1;
    int errors = 0;
    char *progName;
    int print_help = 0;

    user_id = -1;
    result_dir = "";

    while (next_arg < argc) {
        if (!STRCMP("-debug", argv[next_arg])) {
            /* Enable debug mode (for testing) */
            debug = 1;
        } else if (!STRCMP("-dir", argv[next_arg])) {
            /* The directory for the client output */
            result_dir = argv[next_arg];
        } else if (!STRCMP("-log", argv[next_arg])) {
            /* A less intrusive form of debug mode */
            logtrans = 1;
        } else if (!STRCMP("-id", argv[next_arg])) {
            /* The id of this client */
            user_id = atoi(argv[next_arg]);
        } else if (!STRCMP("-port", argv[next_arg])) {
            /* The id of this client */
            user_port = atoi(argv[next_arg]);
            if (user_id < 0) user_id = user_port;
        } else if (!STRCMP("-security", argv[next_arg])) {
            /* Enable security between the client and the server.
             * This is enabled by default
             */
            useSecurity = TRUE;
        } else if (!STRCMP("-noSecurity", argv[next_arg])) {
            /* Disable security between the client and the server.
             * This is enabled by default
             */
            useSecurity = FALSE;
        } else if (!STRCMP("-null", argv[next_arg])) {
            /* For testing: do not access the data in the DB */
            logprintf("Performing NULL test\n");
            null_test = 1;
        } else if (!STRCMP("-lock", argv[next_arg])) {
            logprintf("Locking longterm handles\n");
            client_lock_handles = atoi(argv[next_arg]);
        } else {
            printf("invalid parameter: %s\n", argv[next_arg]);
            print_help = 1;
        }
        next_arg++;
    }

    if (user_id < 0) {
        printf(" Missing UserId\n");
        print_help = 1;
    }

    if (print_help) {
        progName = strchr(argv[0], '/');
        progName = (progName ? progName + 1 : argv[0]);

        printf("\nusage:\n You can specify the following in any order\n");
        printf(" You must specify the Id\n");

        printf(" -id <num> The user ID for this client\n");
        printf(" -dir <dir> Directory for output (default '\")

```

do tpcc.h

```

/*
 * do_tpcc.h
 *
 * $Revision: 1.1 $
 * $Date: 1998/11/09 16:00:05 $
 * $Log: do_tpcc.h,v $
 *
 * $ALog:do_tpcc.h,v $
 * Revision 1.1 1998/11/09 16:00:05dongfeng
 * Move do_tpcc.h to common directory
 * [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.4]
 *
 * Revision 1.7 1998/01/23 15:07:49 oz
 * - Updated the SP TPCC directory to the latest files used
 * during the SP tpcc audit.
 * [from r1.6 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
 *
 *
 */

#ifndef _DO_TPCC_H_INCLUDED_
#define _DO_TPCC_H_INCLUDED_

#include <dce/rpc.h>
#include <trpc/trpc.h>
#include "databuf.h"

#define WRONG_INPUT 0
#define NEW_ORDER 1
#define PAYMENT 2
#define ORDER_STATUS 3
#define DELIVERY 4
#define STOCK_LEVEL 5
#define QUIT 9
#define MIN_OL 5
#define MAX_FLDS 200 /* Maximum fields in a TPC-C form */

#define THRESHOLD_LEN 2

#define ON 1
#define OFF 0

```



```

#define YES 1
#define NO 0

#define INSIZE 1024

#define DO_ROLLBACK 1
#define DONT_ROLLBACK 0

/** The response time requirements for the transactions in seconds.
** 90% of the transactions are required to have a response time less
** than or equal to the value below.
**/
#define NEWORD_90RT 5
#define PAYMENT_90RT 5
#define ORDSTAT_90RT 5
#define DELIVERY_90RT 5 /* 5 for interactive or 80 for background*/
#define STOCKLEV_90RT 20

/*
 * What type of client is this?
 */
typedef enum {
    tk_client,
    dce_client,
    mon_client,
    db_client
} client_type_t;

extern client_type_t client_type;

typedef enum {
    transparent, explicit, longTerm, noReservation
} binding_t;

/* Handle from client to PA is now described using both the paHandle
and the mondHandle. */
#define NUM_TRANS 5
#define NEWO_ERR 6
#define PAYMENT_ERR 7
#define ORD_STAT_ERR 8
#define DELIVERY_ERR 9
#define STOCK_ERR 10
#define NEWO_ROLLBACK 11
#define END_OF_WINDOW 0xff
#define BEGIN_WINDOW 0xaa
#ifdef SHORT_WAITS
#define NEWO_MEAN_THINK_TIME 122
#define PAYMENT_MEAN_THINK_TIME 122
#define ORDER_STAT_MEAN_THINK_TIME 102
#define DELIVERY_MEAN_THINK_TIME 51
#define STOCK_MEAN_THINK_TIME 51
#define NEWO_MIN_KEY_TIME 185
#define PAYMENT_MIN_KEY_TIME 31
#define ORDER_STAT_MIN_KEY_TIME 21
#define DELIVERY_MIN_KEY_TIME 21
#define STOCK_MIN_KEY_TIME 21
#else
#define NEWO_MEAN_THINK_TIME 61
#define PAYMENT_MEAN_THINK_TIME 61
#define ORDER_STAT_MEAN_THINK_TIME 51
#define DELIVERY_MEAN_THINK_TIME 26
#define STOCK_MEAN_THINK_TIME 26
#define NEWO_MIN_KEY_TIME 93
#define PAYMENT_MIN_KEY_TIME 16
#define ORDER_STAT_MIN_KEY_TIME 11
#define DELIVERY_MIN_KEY_TIME 11
#define STOCK_MIN_KEY_TIME 11
#endif

#endif /* _DO_TPCC_H_INCLUDED_ */

                encina.C

/* (C)1997 IBM Corporation */
/*****
*/
*/
*/
File: tuxclient.h
*/
/*****

#include <stdlib.h>
#include "inout.h"
#include "encina.h"

extern "C" {
}

extern "C" send_new_order(void *contextP, NewOrder_data *dataP);
extern "C" send_payment(void *contextP, Payment_data *dataP);
extern "C" send_stock_level(void *contextP, StockLevel_data *dataP);
extern "C" send_order_status(void *contextP, OrderStatus_data *dataP);
extern "C" send_delivery(void *contextP, Delivery_data *dataP);

void Encina::cleanup() {
}

Encina::Encina() {
    return;
}

Encina::~Encina() {
    return;
}

int Encina::tran(NewOrder_data *dataP, void *contextP, char *servname) {
    send_new_order(contextP, dataP);
    return 0;
}

int Encina::tran(Payment_data *dataP, void *contextP, char *servname) {
    send_payment(contextP, dataP);
    return 0;
}

int Encina::tran(OrderStatus_data *dataP, void *contextP, char *servname) {
    send_order_status(contextP, dataP);
    return 0;
}

int Encina::tran(StockLevel_data *dataP, void *contextP, char *servname) {
    send_stock_level(contextP, dataP);
    return 0;
}

int Encina::tran(Delivery_data *dataP, void *contextP, char *servname) {
    send_delivery(contextP, dataP);
    return 0;
}

int Encina::tran(char *servname) {
    return -1;
}

int Encina::atran(char *servname) {
    return 0;
}

                encina.h

/* (C)1997 IBM Corporation */
/*****
*/
*/
File: tuxclient.h
*/
/*****

#ifdef ENCINA_H
#define ENCINA_H

const int TMINBUFSIZE= 1536;

class Encina {
public:
    static void cleanup();
    int tran(char *servname);
    int tran(NewOrder_data *dataP, void *contextP, char *servname);
    int tran(Payment_data *dataP, void *contextP, char *servname);
    int tran(StockLevel_data *dataP, void *contextP, char *servname);
    int tran(OrderStatus_data *dataP, void *contextP, char *servname);
    int tran(Delivery_data *dataP, void *contextP, char *servname);
    int atran(char *servname);
    Encina();
    ~Encina();
};

extern Encina encina;

#endif

                encina_client.c

/*
 *
 * encina_client.c
 *
 * $Revision: 1.7 $
 * $Date: 1999/05/06 21:28:26 $
 * $Log: $
 *
 *
 * $TALog: encina_client.c,v $
 * Revision 1.7 1999/05/06 21:28:26 oz
 * - Removed all the .. from the includes
 * - Added -I.. to the makefiles instead

```

```

* - Moved all the thread related code and connection
* selection to serverMon.c
* [from r1.6 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5]
*
* Revision 1.6 1998/11/09 16:59:37wenjian
* In this revision, most of the changes are related to the directory of header
* files after directory reorganization. Other changes include adding or removing
* files to put them in the right directories. Makefiles are written for NT
* platform so that nmake is working on NT now. Need a top level Makefile for all
* the directories.
* [from r1.5 by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.2]
*
* Revision 1.5 1998/11/09 14:48:16wenjian
* In an effort to make a new directory structure for TPCC, this delta
* creates two directories: tpcc/client and tpcc/server. All the files
* for this revision are copied from tpcc/sp-tpcc without any change.
* Further change may be needed for some files due to the change of
* the directory structure.
* [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1]
*
* Revision 1.5 1998/01/23 15:07:51 oz
* - Updated the SP TPCC directory to the latest files used
* during the SP tpcc audit.
* [from r1.4 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
*
*
*/
/*
* encina_client.c
*
* The Encina related code in the client that is common to both
* the monitor client and the toolkit client.
*/
#include <stdio.h>
#include <string.h>
#include <stdarg.h>
#include <trpc/trpc.h>
#include <encina/encina.h>
#include "common/utilities.h"
#include "client_utils.h"
#include "encina_client.h"

static trpc_handle_t bind_to_server(char *name);

/*
* encina_error_message
*
* Report an encina error message by interpreting it and writing
* it to both the logfile (if any) and to standard error
*/
void encina_error_message(msg, n)
char *msg;
unsigned long n;
{
    char errorMsg[ENCINA_MAX_STATUS_STRING_SIZE];
    encina_StatusToString(n, ENCINA_MAX_STATUS_STRING_SIZE, errorMsg);
    err_printf("ERROR: %s. Error code = %s (%d 0x%x)\n", msg, errorMsg, n, n);
}

/*
* encina_error
*
* This is called for FATAL errors. It reports the error and exits.
*/
void encina_error(funcName, n)
char *funcName;
unsigned long n;
{
    char msg[128];
    sprintf("%s failed", funcName);
    encina_error_message(msg, n);
    exit_program(1);
}

/*
* secure_handle
*
* Secure a handle to an encina server.
* This can be called with either a PA handle or with
* a trpc handle to a toolkit server.
*/
void secure_handle(trpc_handle_t handle, int use_security)
{
    trpc_binding_handle_t rpcHandle;
    unsigned long status = 0;
    unsigned char *serverPrincipal;

    ENCINA_CALL("trpc_GetRpcHandleFromBinding",
        trpc_GetRpcHandleFromBinding(handle, &rpcHandle));

    rpc_mgmt_inq_server_princ_name(rpcHandle, rpc_c_authn_default,
        &serverPrincipal, &status);
}

```

```

if (use_security) {
    DPRINT(("rpc_binding_set_auth_info -> principal %s, protect %d, authn %d, authz %d\n",
        serverPrincipal, rpc_c_protect_level_connect,
        rpc_c_authn_default, rpc_c_authz_dce));

    rpc_binding_set_auth_info(rpcHandle, serverPrincipal,
        rpc_c_protect_level_connect,
        rpc_c_authn_default,
        NULL,
        rpc_c_authz_dce,
        &status);
} else {
    DPRINT(("rpc_binding_set_auth_info -> principal %s, protect %d, authn %d, authz %d\n",
        serverPrincipal, rpc_c_protect_level_none,
        rpc_c_authn_default, rpc_c_authz_dce));

    rpc_binding_set_auth_info(rpcHandle, serverPrincipal,
        rpc_c_protect_level_none,
        rpc_c_authn_default,
        NULL,
        rpc_c_authz_dce,
        &status);
}

if (status != rpc_s_ok) {
    switch (status) {
        case rpc_s_invalid_binding :
            printf("rpc binding invalid***** \n");
            break;
        case rpc_s_wrong_kind_of_binding :
            printf("rpc binding is the wrong kind \n");
            break;
        case rpc_s_unknown_authn_service :
            printf("rpc authn service unknown \n");
            break;
    } /* switch */
    bde_Exit(1);
}
}

```

encina_client.h

```

/*
* encina_client.h
*
* $Revision: 1.5 $
* $Date: 1998/11/09 14:48:16 $
* $Log: $
*
* $STALog: encina_client.h,v $
* Revision 1.5 1998/11/09 14:48:16wenjian
* In an effort to make a new directory structure for TPCC, this delta
* creates two directories: tpcc/client and tpcc/server. All the files
* for this revision are copied from tpcc/sp-tpcc without any change.
* Further change may be needed for some files due to the change of
* the directory structure.
* [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1]
*
* Revision 1.5 1998/01/23 15:07:52 oz
* - Updated the SP TPCC directory to the latest files used
* during the SP tpcc audit.
* [from r1.4 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
*
* Declarations common to monitor version and toolkit version
*/

#ifndef ENCINA_CLIENT_H
#define ENCINA_CLIENT_H

#include <trpc/trpc.h>

void encina_error_message(char *msg, unsigned long n);
void encina_error(char *funcName, unsigned long n);
void secure_handle(trpc_handle_t handle, int use_security);

#endif /* ENCINA_CLIENT_H */

/* (C)1997 IBM Corporation */
#include <stdio.h>
#include "field.h"
#include "inout.h"
#include "format.h"
#if 0
#endif
#if USE_ALLOCA
#include <alloca.h>
#endif
#endif

extern char const * const blanks;

```

field.C

```

extern char const * const underscores;
extern char const * const backspaces;
extern int position(InOut *ioP, int x, int y);

Field *genfield(InOut *ioP, int x, int y, int len, int *ptr) {
    return new IntField(ioP, x, y, len, ptr);
}
Field *genfield(InOut *ioP, int x, int y, int len, short *ptr) {
    return new ShortField(ioP, x, y, len, ptr);
}
Field *genfield(InOut *ioP, int x, int y, int len, long *ptr) {
    return new LongField(ioP, x, y, len, ptr);
}
Field *genfield(InOut *ioP, int x, int y, int len, char *ptr) {
    return new TextField(ioP, x, y, len, ptr);
}
Field *genfield(InOut *ioP, int x, int y, int len, double *ptr) {
    return new MoneyField(ioP, x, y, len, ptr);
}
Field *genfield(InOut *ioP, int x, int y, int len, unsigned char *ptr) {
    return new Int8Field(ioP, x, y, len, ptr);
}

/*****
Field
*****/
Field::Field(InOut *inoutP, int size, char *str)
: ioP(inoutP), len(size), pos(0), changed(0), need_redisplay(0)
{
    need_free_string = need_free = 0;
    if (str == NULL) {
        string = new char[len+1];
        need_free_string = 1;
    } else {
        string = str;
    }
    ok_func = NULL;
    ok_data = NULL;
    string[0] = 0;
}

Field::Field(InOut *ioP, int inx, int iny, int size, char *str)
: ioP(ioP), x(inx), y(iny), len(size), pos(0), changed(0), need_redisplay(0)
{
    need_free_string = need_free = 0;
    if (str == NULL) {
        string = new char[len+1];
        need_free_string = 1;
    } else {
        string = str;
    }
    ok_func = NULL;
    ok_data = NULL;
    string[0] = 0;
}

int Field::reset() {
    pos=0;
    changed=0;
    return 0;
}

Field::~Field() {
    if (need_free_string)
        delete [] string;
}

int Field::finalize_field() {
    changed = 0;
    string[pos] = 0;
    return 0;
}

int Field::display_field(int use_underscores) {
    position(ioP, x, y);
    ioP->write(string);
    if (use_underscores) {
        ioP->write(underscores, len-pos);
    } else {
        ioP->write(blanks, len-pos);
    }
    return 0;
}

int Field::get_key() {
    char key;
    int cc;
    cc = ioP->read(&key, 1);

    return (cc == 0) ? EOF : key ;
}

int Field::add_char(int key) {
    if (pos >= len || (!isprint(key) && key != ' ')) {
        ioP->write("\a", 1);
        return 1;
    }
    changed = 1;
    string[pos] = key;
    ioP->write(&string[pos+1], 1);
    return 0;
}

int Field::backspace() {
    ioP->write("\b\b", 3);
    changed = 1;
    pos--;
    return 0;
}

int Field::start_position() {
    position(ioP, x, y);
    return 0;
}

int Field::get_field(int need_pos) {
    int key;

    if (need_pos)
        position(ioP, x, y);
    if (pos != 0) {
        need_redisplay = 1;
        ioP->write(string, pos);
        ioP->write(underscores, len-pos);
        if (len-pos < 6)
            ioP->write(backspaces, len-pos);
        else
            position(ioP, x+pos, y);
    }

    ioP->mark();
    while (1) {
        key = get_key();
        switch(key) {
            case EOF:
                return EOF;

            case '\r': /* Carriage Return */
            case '\n': /* Newline */
                ioP->hold();
                if (changed) {
                    finalize_field();
                }
                ioP->pop();
                display_field(1);
                return ENTER;
                break;

            case '\t': /* Tab */
            case '\006': /* Ctrl-F */
            case '\016': /* Ctrl-N */
                if (changed) {
                    finalize_field();
                }
                ioP->pop();
                display_field(1);
                return NEXT_FIELD;
                break;

            case '\002': /* Ctrl-B */
            case '\020': /* Ctrl-P */
                if (changed) {
                    finalize_field();
                }
                ioP->pop();
                display_field(1);
                return PREV_FIELD;
                break;

            case '\b': /* Backspace */
            case '\177': /* Del */
                if (pos > 0) {
                    backspace();
                } else
                    ioP->write("\a", 1);
                break;

            case '\014': /* Ctrl-L */
                ioP->pop();
                return REDISPLAY;

            case '\030': /* Ctrl-X */
            case '\003': /* Ctrl-C */
                ioP->unmark();
                return ABORT;

            default:
                add_char(key);
        }
    }

/*****
IntField
*****/
IntField::IntField(InOut *ioP, int inx, int iny, int size, int *val) : Field(ioP, inx, iny, size), value(val) {
    if (value==NULL) {
        value = new int;
        need_free=1;
    }
}

IntField::IntField(InOut *ioP, int size, int *val) : Field(ioP, size), value(val) {
    if (value==NULL) {
        value = new int;
        need_free=1;
    }
}

```

```

IntField::~IntField() {
    if (need_free)
        delete value;
}

int IntField::add_char(int key) {
    if (pos < len && isdigit(key)) {
        changed = 1;
        string[pos] = key;
        ioP->write(&string[pos++], 1);
        return 0;
    }
    ioP->write("a", 1);
    return 1;
}

int IntField::display_field(intuse_underscores) {
    int firstchar;
#ifdef USE_ALLOCA
    char *buf = (char *)alloca(len+1);
#else
    char *buf = new char[len+1];
#endif
    memset(buf, 'x', len);
    if (pos)
        firstchar = format_int(buf, len+1, *value);
    else
        firstchar = len;
    position(ioP, x, y);
    if (use_underscores) {
        ioP->write(underscores, firstchar);
        ioP->write(buf+firstchar, len-firstchar);
    } else {
        ioP->write(buf, len);
    }
    return 0;
}

int IntField::finalize_field() {
    changed = 0;
    string[pos] = 0;
    if (value != NULL)
        *value = atoi(string);
    return 0;
}

ShortField
ShortField::ShortField(InOut *ioP, int inx, int iny, int size, short *val) : Field(ioP, inx, iny, size), value(val) {
    if (value==NULL) {
        value = new short;
        need_free=1;
    }
}

ShortField::ShortField(InOut *ioP, int size, short *val) : Field(ioP, size), value(val) {
    if (value==NULL) {
        value = new short;
        need_free=1;
    }
}

ShortField::~ShortField() {
    if (need_free)
        delete value;
}

int ShortField::add_char(int key) {
    if (pos < len && isdigit(key)) {
        changed = 1;
        string[pos] = key;
        ioP->write(&string[pos++], 1);
        return 0;
    }
    ioP->write("a", 1);
    return 1;
}

int ShortField::display_field(intuse_underscores) {
    int firstchar;
#ifdef USE_ALLOCA
    char *buf = (char *)alloca(len+1);
#else
    char *buf = new char[len+1];
#endif
    if (pos)
        firstchar = format_short(buf, len+1, *value);
    else
        firstchar = len;
    position(ioP, x, y);
    if (use_underscores) {
        ioP->write(underscores, firstchar);
        ioP->write(buf+firstchar, len-firstchar);
    } else {
        ioP->write(buf, len);
    }
    return 0;
}

int ShortField::finalize_field() {
    changed = 0;
    string[pos] = 0;
    if (value != NULL)
        *value = atoi(string);
}

return 0;
}

ShortField
ShortField::ShortField(InOut *ioP, int inx, int iny, int size, unsigned char *val) : Field(ioP, inx, iny, size),
value(val) {
    if (value==NULL) {
        value = new unsigned char;
        need_free=1;
    }
}

Int8Field::Int8Field(InOut *ioP, int size, unsigned char *val) : Field(ioP, size), value(val) {
    if (value==NULL) {
        value = new unsigned char;
        need_free=1;
    }
}

Int8Field::~Int8Field() {
    if (need_free)
        delete value;
}

int Int8Field::add_char(int key) {
    if (pos < len && isdigit(key)) {
        changed = 1;
        string[pos] = key;
        ioP->write(&string[pos++], 1);
        return 0;
    }
    ioP->write("a", 1);
    return 1;
}

int Int8Field::display_field(intuse_underscores) {
    int firstchar;
#ifdef USE_ALLOCA
    char *buf = (char *)alloca(len+1);
#else
    char *buf = new char[len+1];
#endif
    if (pos)
        firstchar = format_char(buf, len+1, *value);
    else
        firstchar = len;
    position(ioP, x, y);
    if (use_underscores) {
        ioP->write(underscores, firstchar);
        ioP->write(buf+firstchar, len-firstchar);
    } else {
        ioP->write(buf, len);
    }
    return 0;
}

int Int8Field::finalize_field() {
    changed = 0;
    string[pos] = 0;
    if (value != NULL)
        *value = atoi(string);
    return 0;
}

LongField
LongField::LongField(InOut *ioP, int inx, int iny, int size, long *val) : Field(ioP, inx, iny, size), value(val) {
    if (value==NULL) {
        value = new long;
        need_free=1;
    }
}

LongField::LongField(InOut *ioP, int size, long *val) : Field(ioP, size), value(val) {
    if (value==NULL) {
        value = new long;
        need_free=1;
    }
}

LongField::~LongField() {
    if (need_free)
        delete value;
}

int LongField::add_char(int key) {
    if (pos < len && isdigit(key)) {
        changed = 1;
        string[pos] = key;
        ioP->write(&string[pos++], 1);
        return 0;
    }
    ioP->write("a", 1);
    return 1;
}

int LongField::display_field(intuse_underscores) {
    int firstchar;
#ifdef USE_ALLOCA
    char *buf = (char *)alloca(len+1);
#else
    char *buf = new char[len+1];
#endif
    if (pos)
        firstchar = format_int(buf, len+1, *value);
    else
        firstchar = len;
    position(ioP, x, y);
    if (use_underscores) {
        ioP->write(underscores, firstchar);
        ioP->write(buf+firstchar, len-firstchar);
    } else {
        ioP->write(buf, len);
    }
    return 0;
}

int LongField::finalize_field() {
    changed = 0;
    string[pos] = 0;
    if (value != NULL)
        *value = atoi(string);
}

```

```

else
    firstchar = format_long(buf, len+1, *value);
firstchar = len;
position(ioP, x, y);
if (use_underscores) {
    ioP->write(underscores, firstchar);
    ioP->write(buf+firstchar, len-firstchar);
} else {
    ioP->write(buf, len);
}
return 0;
}
int LongField::finalize_field() {
    changed = 0;
    string[pos] = 0;
    if (value != NULL)
        *value = atoi(string);
    return 0;
}
/*****
MoneyField
*****/
MoneyField::MoneyField(InOut *ioP, int inx, int iny, int size, double *val) : Field(ioP, inx, iny, size),
value(val) {
    seen_dollar = seen_sign = seen_dot = seen_digit = 0;
    if (value == NULL) {
        value = new double;
        need_free = 1;
    }
}
MoneyField::MoneyField(InOut *ioP, int size, double *val) : Field(ioP, size), value(val) {
    seen_dollar = seen_sign = seen_dot = seen_digit = 0;
    if (value == NULL) {
        value = new double;
        need_free = 1;
    }
}
MoneyField::~MoneyField() {
    if (need_free)
        delete value;
}
int MoneyField::add_char(int key) {
    do {
        if (pos >= len)
            break;
        if (key == '$') {
            if (!(pos == 0 || (pos == 1 && seen_sign))) break;
            seen_dollar = 1;
        } else if (key == '.') {
            if (!(pos == 0 || (pos == 1 && seen_dollar))) break;
            seen_sign = 1;
        } else if (key == ',') {
            if (seen_dot) break;
            seen_dot = 1;
        } else if (!isdigit(key))
            break;
        if (seen_dot) {
            if (seen_dot >= 4)
                break;
            seen_dot++;
        }
        changed = 1;
        string[pos] = key;
        ioP->write(&string[pos++], 1);
        return 0;
    } while (0);
    ioP->write("a", 1);
    return 1;
}
int MoneyField::backspace() {
    ioP->write("b\b", 3);
    changed = 1;
    pos--;
    if (seen_dot)
        seen_dot--;
    if (string[pos] == '-')
        seen_sign = 0;
    if (string[pos] == '$')
        seen_dollar = 0;
    if (string[pos] == '.')
        seen_dot = 0;
    return 0;
}
int MoneyField::display_field(int use_underscores) {
    int firstchar;
    #if USE_ALLOCA
    char *buf = (char *)alloca(len+1);
    #else
    char *buf = new char[len+1];
    #endif
    if (pos)
        firstchar = format_money(buf, len+1, *value);
    else
        firstchar = len;
    position(ioP, x, y);
    if (use_underscores) {
        ioP->write(underscores, firstchar);
        ioP->write(buf+firstchar, len-firstchar);
    } else {
        ioP->write(buf, len);
    }
    return 0;
}
}
int MoneyField::finalize_field() {
    changed = 0;
    string[pos] = 0;
    if (value != NULL) {
        *value = atof(string + seen_dollar + seen_sign);
        if (seen_sign)
            *value = -*value;
    }
    return 0;
}
int MoneyField::reset() {
    Field::reset();
    seen_dollar = seen_sign = seen_dot = seen_digit = 0;
    return 0;
}
/*****
TextField
*****/
TextField::TextField(InOut *ioP, int inx, int iny, int size, char *str) : Field(ioP, inx, iny, size, str) {
    value = TextField::string;
}
TextField::TextField(InOut *ioP, int size, char *str) : Field(ioP, size, str) {
    value = TextField::string;
}
int TextField::add_char(int key) {
    if (pos >= len || (!isalnum(key) && key != '.' && key != ',')) {
        ioP->write("a", 1);
        return 1;
    }
    changed = 1;
    string[pos] = key;
    ioP->write(&string[pos++], 1);
    return 0;
}
}

```

field.h

```

/* (C)1997 IBM Corporation */
#ifndef INCLUDE_FIELD_H
#define INCLUDE_FIELD_H

#include "inout.h"

class Field {
public:
    enum return_codes { INVALID, ENTER, NEXT_FIELD, PREV_FIELD, ABORT, REDISPLAY };
    InOut *ioP;
    int x, y;
    const int len;
    int pos;
    int changed;
    int need_redisplay;
    char *string;
    int (*ok_func)(void *data);
    int need_free;
    int need_free_string;
    void *ok_data;
    Field(InOut *ioP, int size, char *string=NULL);
    Field(InOut *ioP, int x, int y, int size, char *string=NULL);
    virtual ~Field();
    virtual int get_field(int need_pos=1);
    int get_key();
    virtual int backspace();
    virtual int reset();
    virtual int start_position();
    virtual int add_char(int key);
    virtual int display_field(int use_underscores=0);
    virtual int finalize_field();

    class Error {
    public:
        enum { USER_ABORT };
    };
};

class Int8Field : public Field {
public:
    unsigned char *value;
    int add_char(int key);
    int display_field(int use_underscores=0);
    int finalize_field();

    Int8Field(InOut *ioP, int x, int y, int size, unsigned char *value=NULL);
    Int8Field(InOut *ioP, int size, unsigned char *value=NULL);
    virtual ~Int8Field();
};

class ShortField : public Field {
public:
    short *value;
    int add_char(int key);
};

```

```

int display_field(int use_underscores=0);
int finalize_field();

ShortField(InOut *ioP, int x, int y, int size, short *value=NULL);
ShortField(InOut *ioP, int size, short *value=NULL);
virtual ~ShortField();
};

class IntField : public Field {
public:
    int *value;
    int add_char(int key);
    int display_field(int use_underscores=0);
    int finalize_field();

    IntField(InOut *ioP, int x, int y, int size, int *value=NULL);
    IntField(InOut *ioP, int size, int *value=NULL);
    virtual ~IntField();
};

class LongField : public Field {
public:
    long *value;
    int add_char(int key);
    int display_field(int use_underscores=0);
    int finalize_field();

    LongField(InOut *ioP, int x, int y, int size, long *value=NULL);
    LongField(InOut *ioP, int size, long *value=NULL);
    virtual ~LongField();
};

class MoneyField : public Field {
public:
    int seen_dollar, seen_sign, seen_dot, seen_digit;
    double *value;
    int add_char(int key);
    int reset();
    int backspace();
    int display_field(int use_underscores=0);
    int finalize_field();
    MoneyField(InOut *ioP, int x, int y, int size, double *value=NULL);
    MoneyField(InOut *ioP, int size, double *value=NULL);
    virtual ~MoneyField();
};

class TextField : public Field {
public:
    char *value;
    int add_char(int key);
    TextField(InOut *ioP, int x, int y, int size, char *value=NULL);
    TextField(InOut *ioP, int size, char *value=NULL);
};

Field *genfield(InOut *ioP, int x, int y, int len, int *ptr);
Field *genfield(InOut *ioP, int x, int y, int len, short *ptr);
Field *genfield(InOut *ioP, int x, int y, int len, long *ptr);
Field *genfield(InOut *ioP, int x, int y, int len, char *ptr);
Field *genfield(InOut *ioP, int x, int y, int len, unsigned char *ptr);
Field *genfield(InOut *ioP, int x, int y, int len, double *ptr);

#endif /* INCLUDE_FIELD_H */

```

format.C

```

/* (C)1997 IBM Corporation */
#include <string.h>
#include <math.h>

int format_char(char *buf, int size, char val) {
    int neg, pos;
    pos = size;
    buf[--pos] = 0;
    if (val == 0 && pos > 0) {
        buf[--pos] = '0';
        neg = 0;
    } else {
        neg = (val < 0) ? 1 : 0;
        if (neg) val = -val;
        while (val && pos > 0) {
            buf[--pos] = (val % 10) + '0';
            val /= 10;
        }
    }
    /* Too long */
    if (!pos && (val || neg)) {
        memset(buf, '*', size);
        return -1;
    }
    if (neg)
        buf[--pos] = '-';
    if (pos)
        memset(buf, ' ', pos);
    return pos;
}

int format_short(char *buf, int size, short val) {

```

```

    int neg, pos;
    pos = size;
    buf[--pos] = 0;
    if (val == 0 && pos > 0) {
        buf[--pos] = '0';
        neg = 0;
    } else {
        neg = (val < 0) ? 1 : 0;
        if (neg) val = -val;
        while (val && pos > 0) {
            buf[--pos] = (val % 10) + '0';
            val /= 10;
        }
    }
    /* Too long */
    if (!pos && (val || neg)) {
        memset(buf, '*', size);
        return -1;
    }
    if (neg)
        buf[--pos] = '-';
    if (pos)
        memset(buf, ' ', pos);
    return pos;
}

int format_int(char *buf, int size, int val) {
    int neg, pos;
    pos = size;
    buf[--pos] = 0;
    if (val == 0 && pos > 0) {
        buf[--pos] = '0';
        neg = 0;
    } else {
        neg = (val < 0) ? 1 : 0;
        if (neg) val = -val;
        while (val && pos > 0) {
            buf[--pos] = (val % 10) + '0';
            val /= 10;
        }
    }
    /* Too long */
    if (!pos && (val || neg)) {
        memset(buf, '*', size);
        return -1;
    }
    if (neg)
        buf[--pos] = '-';
    if (pos)
        memset(buf, ' ', pos);
    return pos;
}

int format_long(char *buf, int size, long val) {
    int neg, pos;
    pos = size;
    buf[--pos] = 0;
    if (val == 0 && pos > 0) {
        buf[--pos] = '0';
        neg = 0;
    } else {
        neg = (val < 0) ? 1 : 0;
        if (neg) val = -val;
        while (val && pos > 0) {
            buf[--pos] = (val % 10) + '0';
            val /= 10;
        }
    }
    /* Too long */
    if (!pos && (val || neg)) {
        memset(buf, '*', size);
        return -1;
    }
    if (neg)
        buf[--pos] = '-';
    if (pos)
        memset(buf, ' ', pos);
    return pos;
}

int format_float(char *buf, int size, int dec, double val) {
    static double pow10[] = { 1, 10, 100, 1000, 10000, 100000, 1000000 };
    int neg, pos;
    pos = size;
    buf[--pos] = 0;
    #if defined WIN32
        val = rint(val * pow10[dec]);
    #else /* there is no rint on NT. Use floor instead */
        val = floor(val * pow10[dec] + 0.5);
    #endif
    #endif
    neg = (val < 0) ? 1 : 0;
    if (neg) val = -val;

    while (val >= 1 && pos > 0) {
        if (!dec--) {
            buf[--pos] = '-';
            continue;
        }
        buf[--pos] = (int)fmod(val, 10) + '0';
        val /= 10;
    }
}

```

```

if(dec >= 0) {
    while (dec >= 0 && pos > 0) {
        if(!dec--) {
            buf[--pos] = '.';
        } else {
            buf[--pos] = '0';
        }
    }
    if(pos > 0)
        buf[--pos] = '0';
}
/* Too long */
if(!pos && (val >= 1 || neg)) {
    memset(buf, '*', size);
    return -1;
}
if(neg)
    buf[--pos] = '-';
if(pos)
    memset(buf, '*', pos);
return pos;
}

int format_money(char *buf, int size, double val) {
    int pos;
    pos = format_float(buf, size, 2, val);
    if(pos > 0)
        buf[--pos] = '$';
    return pos;
}

int format_date(char *buf, int size, unsigned char* val) {
    memcpy(buf, val, size);
    buf[size]=0;
    return 0;
}

int format_phone(char *buf, int size, unsigned char *phone) {
    buf[0] = phone[0];
    buf[1] = phone[1];
    buf[2] = phone[2];
    buf[3] = phone[3];
    buf[4] = phone[4];
    buf[5] = phone[5];
    buf[6] = '-';
    buf[7] = phone[6];
    buf[8] = phone[7];
    buf[9] = phone[8];
    buf[10] = '-';
    buf[11] = phone[9];
    buf[12] = phone[10];
    buf[13] = phone[11];
    buf[14] = '-';
    buf[15] = phone[12];
    buf[16] = phone[13];
    buf[17] = phone[14];
    buf[18] = phone[15];
    buf[19] = '\0';
    return size;
}

int format_zip(char *buf, int size, unsigned char *zip) {
    buf[0] = zip[0];
    buf[1] = zip[1];
    buf[2] = zip[2];
    buf[3] = zip[3];
    buf[4] = zip[4];
    buf[5] = '-';
    buf[6] = zip[5];
    buf[7] = zip[6];
    buf[8] = zip[7];
    buf[9] = zip[8];
    buf[10] = '\0';
    return size;
}

```

format.h

```

/* (C)1997 IBM Corporation*/
#ifndef INCLUDE_FORMAT_H
#define INCLUDE_FORMAT_H

int format_char (char *buf, int size, char val);
int format_int (char *buf, int size, int val);
int format_long (char *buf, int size, long val);
int format_short(char *buf, int size, short val);
int format_float(char *buf, int size, int dec, double val);
int format_money(char *buf, int size, double val);
int format_date (char *buf, int size, unsigned char *val);
int format_phone(char *buf, int size, unsigned char *phone);
int format_zip (char *buf, int size, unsigned char *zip);

#endif /* INCLUDE_FORMAT_H */

```

get local time.c

```

/*

```

```

*
* get_local_time.c
*
* $Revision: 1.2 $
* $Date: 1998/11/06 21:42:02 $
* $Log: $
*
*
* $TALog: get_local_time.c,v $
* Revision 1.2 1998/11/06 21:42:02dongfeng
* - Add makefile-nt
*
* - cast cur_t from double to long to get rid of some warnings.
* [from r1.1 by delta dongfeng-23677-TPCC-new-directory-structures,r1.2]
*
* Revision 1.1 1998/11/06 21:10:12dongfeng
* - Move all files common to client and server totppc/common
* directory
* [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
*
* Revision 1.2 1998/10/22 19:18:33dongfeng
* [added by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.2]
*
*/

#ifdef WIN32
#include "get_local_time.h"
#else
#include <sys/time.h>
#endif
#include <stdio.h>

#define Li2Double(x) ((double)(x).HighPart) * 4.294967296E9 +(double)(x).LowPart)

#ifdef WIN32
LARGE_INTEGER pFreq;
double sFreq;

get_time_init()
{
    QueryPerformanceFrequency(&pFreq);
    sFreq=Li2Double(pFreq);
}

get_local_time(struct timeval *timeP)
{
    double cur_t;
    LARGE_INTEGER counter;

    QueryPerformanceCounter(&counter);
    cur_t = Li2Double(counter) / sFreq;
    timeP->tv_sec = (long)cur_t;
    timeP->tv_usec = ((long)cur_t - timeP->tv_sec) * 1000000;
}

int gettimeofday(struct timeval *curTimeP, struct timezone *timezoneP)
{
    get_local_time(curTimeP);
    return 1;
}

#else
get_time_init()
{
}

get_local_time(struct timeval *timeP)
{
    struct timezone tz;

    gettimeofday(timeP, &tz);
}

#endif

#ifdef _GET_LOCAL_TIME_H
#define _GET_LOCAL_TIME_H

#ifdef WIN32
#include <windows.h>
#include <winsock.h>

/*
* gettimeofday is not available in the MicrosoftC/C++ Run Time
* and the Win32 API.
*/

*
* It is not used and just for unix compatibility.
*/
struct timezone {
    char a;
};

get_time_init();

```

get local time.h

```

#endif

```

```
int gettimeofday(struct timeval *curTimeP, struct timezone *timezoneP);
```

```
#endif
```

```
#endif
```

inout.C

```
/* (C)1997 IBM Corporation */
```

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

```
#ifndef WIN32
```

```
#include <strings.h>
```

```
#endif
```

```
#include <unistd.h>
```

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

```
#ifdef WIN32
```

```
#include <i.o.h>
```

```
#include <winsock.h>
```

```
#endif
```

```
#include <stdio.h>
```

```
#include <ctype.h>
```

```
#include <errno.h>
```

```
#include "screen.h"
```

```
extern char *sys_errlist[];
```

```
#if 1
```

```
void InOut::write(const void *buf, size_t size) {
```

```
    if (IOError) return;
```

```
    debug("write(%*s, %d)\n", size, size, buf, size);
```

```
    output.queue(buf, size);
```

```
    if (!Hold && input.len() == 0) { /* Don't write anything until there is no input */
```

```
        flush();
```

```
    }
```

```
}
```

```
ssize_t InOut::read(void *buf, size_t size) {
```

```
    int rc;
```

```
    if (IOError) return(0);
```

```
    while (input.len() < size) {
```

```
#ifdef WIN32
```

```
        rc = recv(in_fd, (char *)input.ptr(), input.free(), 0);
```

```
#else
```

```
        rc = ::read(in_fd, input.ptr(), input.free());
```

```
#endif
```

```
        debug("::read(%*s, %d) = %d\n", rc, rc, input.ptr(), input.free(), rc);
```

```
        if (inlog) {
```

```
            fwrite(input.ptr(), rc, 1, inlog);
```

```
            fflush(inlog);
```

```
        }
```

```
        if (rc > 0) {
```

```
            input.queue(rc);
```

```
        } else if (rc <= 0) {
```

```
            IOError = 1;
```

```
            return(0);
```

```
        }
```

```
    }
```

```
    memcpy(buf, input.ptr(), size);
```

```
    input.dequeue(size);
```

```
    debug("read(%*s, %d) = %d\n", size, size, buf, size, size);
```

```
    return size;
```

```
}
```

```
#else
```

```
void InOut::write(const void *buf, size_t size) {
```

```
    debug("write(%s, %d)\n", buf, size);
```

```
#ifdef WIN32
```

```
    send(out_fd, (char *)buf, size, 0);
```

```
#else
```

```
    ::write(out_fd, buf, size);
```

```
#endif
```

```
}
```

```
ssize_t InOut::read(void *buf, size_t size) {
```

```
    int rc;
```

```
    rc = ::read(in_fd, buf, size);
```

```
    debug("read(%s, %d) = %d\n", buf, size, rc);
```

```
    return rc;
```

```
}
```

```
#endif
```

```
void InOut::flush() {
```

```
    debug("flush():\n");
```

```
    Hold = 0;
```

```
    if (IOError) return;
```

```
    while (output.len()) {
```

```
        debug("::write(%*s, %d)\n", output.len(), output.len(), output.ptr(), output.len());
```

```
#ifdef WIN32
```

```
            int rc = send(out_fd, (char *)output.ptr(), output.len(), 0);
```

```
#else
```

```
            int rc = ::write(out_fd, output.ptr(), output.len());
```

```
#endif
```

```
            if (outlog) {
```

```
                fwrite(output.ptr(), rc, 1, outlog);
```

```
                fflush(outlog);
```

```
            }
```

```
            if (rc > 0) {
```

```
                output.dequeue(rc);
```

```
            } else if (rc < 0) {
```

```
                err_printf("Error writing data!\n");
```

```
                IOError = 1;
```

```
                return;
```

```
            }
```

```
        }
```

```
    }
```

```
void InOut::write(const void *buf) {
```

```
    write(buf, strlen((const char *)buf));
```

```
}
```

```
InOut::InOut(int in, int out) : input(256), output(2048) {
```

```
#ifdef WIN32
```

```
    struct termios buf;
```

```
#endif
```

```
#ifdef DEBUG
```

```
{
```

```
    char buf[256];
```

```
    sprintf(buf, "logs/debug.%d", getpid());
```

```
    debugfile = fopen(buf, "w");
```

```
    sprintf(buf, "logs/in.%d", getpid());
```

```
    inlog = fopen(buf, "w");
```

```
    sprintf(buf, "logs/out.%d", getpid());
```

```
    outlog = fopen(buf, "w");
```

```
}
```

```
#endif
```

```
int rc;
```

```
Hold = 0;
```

```
debugfile = inlog = outlog = (FILE *)0;
```

```
IOError = 0;
```

```
in_fd = in;
```

```
if (out < 0)
```

```
    out_fd = in;
```

```
else
```

```
    out_fd = out;
```

```
#ifndef WIN32
```

```
    if ((rc = tcgetattr(in_fd, &save_term)) < 0) {
```

```
        return;
```

```
    }
```

```
buf = save_term;
```

```
buf.c_lflag &= ~(ECHO | ICANON); /* echo off, canonical mode off */
```

```
buf.c_cc[VMIN] = 1; /* Case B: 1 byte at a time, no timer */
```

```
buf.c_cc[VTIME] = 0;
```

```
err_printf("echo off - tcsetattr on %d\n", in_fd);
```

```
if (tcsetattr(in_fd, TCSAFLUSH, &buf) < 0)
```

```
    return;
```

```
#endif
```

```
}
```

```
InOut::~InOut() {
```

```
#ifdef WIN32
```

```
    return;
```

```
#else
```

```
    if (tcsetattr(in_fd, TCSAFLUSH, &save_term) < 0)
```

```
        return;
```

```
#endif
```

```
}
```

inout.h

```
/* (C)1997 IBM Corporation */
```

```
#ifndef INOUT_H
```

```
#define INOUT_H
```

```
#include <unistd.h>
```

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

```
#include <stdio.h>
```

```
#include <ctype.h>
```

```
#ifdef WIN32
```

```
#include <termios.h>
```

```
#endif
```

```
#include <stdarg.h>
```

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

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

```
/* This is for a VT100 */
```

```
#if 1
```

```
#define ESC "\033"
```

```
#define ESCc "\033'
```

```
#else
```

```
#define ESCc '^'
```

```
#define ESC '^'
```

```
#endif
```

```
#define TRIGGER "\021"
```

```
#define TRIGGERc '\021'
```

```
extern "C" err_printf(...);
```



```

#define POS(x,y) ESC "[" #y ";" #x "H"
#define CLEAR_EOS ESC "[J"

#ifdef WIN32
typedef int ssize_t;
#endif

class InOut {
private:
    class Buffer {
    private:
        int BufSize;
        enum { NUMMARKS=8 };
        char *buffer;
        int marks[NUMMARKS];

    public:
        int Pos;
        int Start;

        int num_marks;
        Buffer(int size) {
            BufSize = size;
            buffer = new char [BufSize];
            Pos = Start = 0;
            num_marks = 0;
        }
        int pos() { return Pos; };
        void pos(int P) { Pos = P; };
        int start() { return Start; };
        void start(int S) { Start = S; };
        int len() { return Pos-Start; };
        int free() { return BufSize-Pos-1; };
        void *ptr() { return &buffer[Start]; };
        int lastmark() { if (num_marks) return marks[num_marks-1]; return 999; };

        void mark() {
            if (num_marks < NUMMARKS)
                marks[num_marks++] = Pos;
            else {
                fprintf(stderr, "Buffer mark overflow\n");
                exit (1);
            }
        }
        void unmark() {
            if (num_marks <= 0)
                return;
            num_marks--;
        }
        void pop() {
            if (num_marks <= 0)
                return;
            if (marks[num_marks-1] >= Start) {
                Pos=marks[--num_marks];
            } else {
                num_marks=0;
            }
        }
    }
    void queue(int size) {
        Pos += size;
    }
    void queue(const void *buf, int size) {
        /* If this is too big see if we can move what we have over*/
        if (size+Pos >= BufSize) {
            if (size + len() >= BufSize) {
                fprintf(stderr, "Buffer overflow\n");
                exit (1);
            }
            /* This requires memcpy to be "safe" */
            if (Start + len() >= BufSize) {
                fprintf(stderr, "Strange Error: Start %d + len %d >= size %d\n",
                    Start , len(), BufSize);
                exit(1);
            }
            memcpy(buffer, &buffer[Start], len());
            Pos = Start;

            /* Fix up our marks*/
            int count = 0;
            for (int i = 0; i < num_marks; i++) {
                if (marks[i] - Start >= 0)
                    marks[count++] = marks[i] - Start;
            }
            num_marks = count;
            Start = 0;

            memcpy(&buffer[Pos], buf, size);
            Pos += size;
        }
    }
    void dequeue(int size) {
        Start += size;
        if (Start >= Pos) {
            /* Fix up our marks*/
            int count = 0;
            for (int i = 0; i < num_marks; i++) {
                if (marks[i] - Start >= 0)
                    marks[count++] = marks[i] - Start;
            }
            num_marks = count;
        }
    }
};

```

```

        Start = Pos = 0;
    }
};
int in_fd, out_fd;
int Hold;
#ifdef WIN32
    struct termios save_term;
#endif
public:
    Buffer input;
    Buffer output;
    FILE *debugfile;
    FILE *inlog, *outlog;
public:
    int IOError;
    ssize_t read(void *buf, size_t size);
    void write(const void *buf, size_t size);
    void write(const void *buf);
    void flush();
    void mark() { debug("mark()\n"); output.mark(); };
    void unmark() { debug("unmark()\n"); output.unmark(); };
    void pop() { debug("pop()\n"); output.pop(); };
    void hold() { debug("hold()\n"); Hold = 1; };
#ifdef DEBUG
    void debug(char *fmt, ...) {
        va_list args;

        fprintf(debugfile, "Start=%2d, Pos=%2d, Marks=%2d(%03d): ", output.Start, output.Pos,
            output.num_marks, output.lastmark());
        va_start(args,fmt);
        vfprintf (debugfile, fmt, args);
        va_end (args);
        ::fflush(debugfile);
    }
#else
    void debug(char *fmt, ...) {};
#endif
    InOut(int in=0, int out=1);
    ~InOut();
};

extern char const * const blanks;
extern char const * const underscores;
extern char const * const backspaces;

int format_int(char *buf, int size, int val);
int format_float(char *buf, int size, int dec, double val);
int format_money(char *buf, int size, double val);

#endif /* INOUT_H */

```

mon_client.c

```

/*
 *      mon_client.c
 *
 * $Revision: 1.27 $
 * $Date: 1999/05/26 16:29:52 $
 * $Log: $
 *
 * $TALog: mon_client.c,v $
 * Revision 1.27 1999/05/26 16:29:52 wenjian
 * Sync with Austin code, and sync code for Oracle DB and SQL server.
 * [from r1.26 by delta wenjian-24433-TPCC-clean-up-and-update,r1.2]
 *
 * Revision 1.26 1999/05/06 21:28:26 oz
 * - Removed all the .. from the includes
 * - Added -L. to the makefiles instead
 * - Moved all the thread related code and connection
 *   selection to serverMon.c
 * [from r1.16 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5]
 *
 * Revision 1.16 1999/01/29 20:16:33 wenjian
 * - Rename client_init to init_encina_client because we have another
 *   client_init in screen/client.C
 * - Add code to read StatsFrequency from .tpccrc (UNIX only)
 * [from r1.15 by delta wenjian-23787-TPCC-integrate-code-for-AIX-and-NT,r1.7]
 *
 * Revision 1.15 1999/01/12 20:52:55 wenjian
 * Call initialization function to create the shared file mapping between
 * it and the corresponding dll.
 * [from r1.14 by delta wenjian-23856-TPCC-integrate-with-NT-performance-monitor,r1.1]
 *
 * Revision 1.14 1998/12/28 20:07:12 wenjian
 * - Change client_info to a pointer pClientInfo for flexibility.
 * [from r1.13 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server,r1.5]
 *
 * Revision 1.13 1998/12/16 17:17:41 wenjian
 * - Change (iStatsFrequency <= 1) to (iStatsFrequency < 1) in pre_rpc.
 * [from r1.12 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server,r1.4]
 *
 * Revision 1.12 1998/12/14 20:27:54 wenjian
 * Made corresponding changes due to data structure change of tran_info_t.
 * [from r1.11 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server,r1.3]
 *
 * Revision 1.11 1998/12/11 16:14:19 wenjian

```

```

* Add code for checking statistic data in a single variable and collecting
* statistic data based on iStatsFrequency.
*
* - Add code to store statistic data in a single var
* - Collect statistic data once every iStatsFrequency transactions
* [from r1.10 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server,r1.1]
*
* Revision 1.10 1998/12/08 23:03:49wenjian
* Add (or rename) Makefile for each platform (AIX and NT). Reorganize the
* files a little bit.
*
* - Define variable iStatsFrequency for AIX
* [from r1.9 by delta wenjian-23787-TPCC-integrate-code-for-AIX-and-NT,r1.1]
*
* Revision 1.9 1998/12/08 18:55:19wenjian
* In pre_rpc, set headerP->stats to iStatsFrequency
* [from r1.8 by delta wenjian-23785-TPCC-pass-statsFrequency-from-client-to-server,r1.1]
*
* Revision 1.8 1998/12/07 20:04:12wenjian
* Clean up
* [from r1.7 by delta wenjian-23742-TPCC-update-with-Raleigh-code,r1.2]
*
* Revision 1.7 1998/11/24 21:45:59wenjian
* - Add #ifdef MULTIPLE_INTERFACE
* - Check if we need to collect statistics for response time
* - Do mutex_lock for terminal_context_init
* [from r1.6 by delta wenjian-23742-TPCC-update-with-Raleigh-code,r1.1]
*
* Revision 1.6 1998/11/09 16:59:38wenjian
* In this revision, most of the changes are related to the directory of header
* files after directory reorganization. Other changes include adding or removing
* files to put them in the right directories. Makefiles are written for NT
* platform so that nmake is working on NT now. Need a top level Makefile for all
* the directories.
* [from r1.5 by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.2]
*
* Revision 1.5 1998/11/09 14:48:16wenjian
* In an effort to make a new directory structure for TPCC, this delta
* creates two directories: tpcc/client and tpcc/server. All the files
* for this revision are copied from tpcc/sp-tpcc without any change.
* Further change may be needed for some files due to the change of
* the directory structure.
* [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1]
*
* Revision 1.43 1998/11/06 16:10:55wenjian
* - Minor change to reduce the print statement
* [from r1.42 by delta wenjian-23646-TPCC-clean-up-source-code,r1.1]
*
* Revision 1.42 1998/10/27 14:57:51dongfeng
* Change enc_status to a data structure that has fields:
* - Status code
* - Line Number
* - File Name
* - Encina Error Code
* - Error Msg
* Remove statusMsgs in web_tpcc.c
* [from r1.41 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.6]
*
* Revision 1.41 1998/10/26 14:41:34dongfeng
* Add Init command in web client so when something bad happens during
* initialization web client sends back error information and allows
* reinitialization instead of killing IIS server.
*
* Define Macro CHK_STATUS instead of using #ifdef
* [from r1.40 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.3]
*
* Revision 1.40 1998/10/22 21:24:11wenjian
* [merge of changes from 1.23 to 1.39 into 1.38]
*
* Revision 1.39 1998/10/22 19:18:34dongfeng
* [from r1.37 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.2]
*
* Revision 1.37 1998/10/08 14:18:01dongfeng
* Add codes for doing web-based tpcc.
* [from r1.23 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.1]
*
* Revision 1.38 1998/10/08 18:03:01gerstl
* Changes to allow configurations where some servers only service
* specific transaction types. Split transaction interfaces by type.
* [from r1.36 by delta gerstl-23515-TPCC-allow-separate-online-transaction-interfaces,r1.1]
*
* Revision 1.36 1998/10/07 15:14:22gerstl
* [merge of changes from 1.26 to 1.31 into 1.34]
*
* Revision 1.31 1998/09/04 19:17:56wenjian
* Remove log_file_handle and related code.
* [from r1.29 by delta wenjian-23183-TPCC-sync-AIX-code-with-Austin,r1.5]
*
* Revision 1.29 1998/08/28 18:30:00wenjian
* This delta sync the TPCC code with Austin.
*
* Remove UNCOND_EVENT in CALLTPCC, pre_rpc and post_rpc.
* [from r1.26 by delta wenjian-23183-TPCC-sync-AIX-code-with-Austin,r1.1]
*
* Revision 1.34 1998/09/26 10:56:26 oz
* - renamed thread_init and thread_done to clnt_thread_init and
* clnt_thread_done respectively because of name conflicts on AIX4.3
* [from r1.26 by delta oz-23339-TPCC-update-for-NT,r1.2]

```

```

* Revision 1.26 1998/08/18 14:38:41wenjian
* Remove adl.h from this file since it is not ported on NT. Use corresponding
* rpc protect levels and authz levels to replace ADL_..... macros defined in
* adl.h
* [from r1.23 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.4]
*
* Revision 1.23 1998/06/17 15:05:45wenjian
* Somehow, read and write didn't work for socket on NT, although they
* are supposed to work. As a work-around way, use recv and send for
* NT in this revision. We may change them back if the problem is gone.
*
* Define SKIP_RPC and add code to do tests without calling DCE.
* This addition is for test purpose only.
* [from r1.22 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.2]
*
* Revision 1.22 1998/02/17 22:13:49wenjian
* [merge of changes from 1.19 to 1.20 into 1.21]
*
* Revision 1.20 1998/02/17 16:04:42 oz
* - Split the login into two parts to allow for special logins
* - If the warehouse ID is 0, this is a special login to
* query the client for status
*
* - Keep track of threads that have been initialized and also
* threads that are done.
* [from r1.19 by delta oz-21864-TPCC-split-client-login-screen,r1.1]
*
* Revision 1.21 1998/02/17 22:07:03wenjian
* Minor changes for NT
* [from r1.19 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.1]
*
* Revision 1.19 1998/01/26 20:37:35 oz
* - Remove all the code associated with explicit binding
*
* - Removed GET_SERVER_INDEX
* - Removed bindingType
* - Removed explicit binding from CALLTPCC
* - Removed calls to cancel_all_reservations and to init_handles
* [from r1.18 by delta oz-21697-TPCC-remove-explicit-binding-code,r1.1]
*
* Revision 1.18 1998/01/26 16:43:32 oz
* - Removed the code for collecting stats in the client
* and dumping them before exit.
*
* - Removed pre_rpc_stats and post_rpc_stats
* - Removed code to write the stats out
* [from r1.17 by delta oz-21691-TPCC-remove-client-stats-code,r1.1]
*
* Revision 1.17 1998/01/26 16:19:23 oz
* - moved all the code pertaining to the background
* thread to its own file and all the data structures
* to client_utils.h
* [from r1.16 by delta oz-21689-TPCC-move-client-bg-thread-to-separate-file,r1.1]
*
* Revision 1.16 1998/01/26 15:33:32 oz
* - call impTPCCNOInfo to make sure there is a server out there
* [from r1.15 by delta oz-21671-TPCC-merge-online-transaction-interfaces,r1.2]
*
* Revision 1.15 1998/01/23 21:58:51 oz
* - In order to simplify the Encina TPCC code: Merge the four
* online transactions into 1 interface
* - Moved all the scripts to a scripts subdirectory
* - Removed unused files
* [from r1.14 by delta oz-21671-TPCC-merge-online-transaction-interfaces,r1.1]
*
* Revision 1.14 1998/01/23 15:07:53 oz
* - Updated the SP TPCC directory to the latest files used
* during the SP tpcc audit.
* [from r1.13 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
*
*
*
*/

#include "common/get_local_time.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdarg.h>
#include <time.h>
#if defined (solaris)
#include <dce/pthread.h>
#else /* solaris */
#include <pthread.h>
#endif /* solaris */
#include <tpm/mon/mon.h>
#include <utils/trace.h>
#include "common/delivery.h"
#ifdef MULTIPLE_INTERFACE
#include "common/neworder.h"
#include "common/payment.h"
#include "common/stocklevel.h"
#include "common/orderstatus.h"
#else
#include "common/tpcc_trans.h"
#endif
#include "common/utilities.h"
#include "client_utils.h"
#include "common/do_tpcc.h"

```



```

if (headerP->stats && tran_type <= MAX_TRAN_TYPE && tran_type > 0
    && !curP->tran_failed) {
    set_client_debug_state((void *)thread_infoP, thread_state_received, 0);

    /* update total server round trip response time */
    time_diff_s = time_diff_ms(&(curP->srvr_done), &(curP->srvr_start));
    thread_infoP->tran[tran_type].RTtotal[1] += time_diff_s;

    /* update total client round trip response time */
    time_diff_c = time_diff_ms(&(curP->end), &(curP->start));
    thread_infoP->tran[tran_type].RTtotal[0] += time_diff_c;

    /* update num for the number of trans which have RT measured */
    thread_infoP->tran[tran_type].RTcount ++;
}
#else
/* Store the info for each client.
 * Note: since we don't use mutex for performance reason, pClientInfo
 * may not be accurate if more than one thread work on the same
 * data at a same time. But this can reduce the overhead caused
 * by scanning info_list for each terminal.
 */
if ((headerP->returncode == TPCC_SUCCESS) ||
    (headerP->returncode == INVALID_NEWO)) {
    curP->tran_failed = 0;
    if (headerP->returncode == INVALID_NEWO) {
        curP->sub_tran |= 0x100;
    }
} else {
    pClientInfo->tran[tran_type].errs ++;
    pClientInfo->errors ++;
    curP->tran_failed = 1;
}
if (headerP->stats && tran_type <= MAX_TRAN_TYPE && tran_type > 0
    && !curP->tran_failed) {
    /* update total server round trip response time */
    time_diff_s = time_diff_ms(&(curP->srvr_done), &(curP->srvr_start));
    pClientInfo->tran[tran_type].RTtotal[1] += time_diff_s;

    /* update total client round trip response time */
    time_diff_c = time_diff_ms(&(curP->end), &(curP->start));
    pClientInfo->tran[tran_type].RTtotal[0] += time_diff_c;

    /* update num for the number of trans which have RT measured */
    pClientInfo->tran[tran_type].RTcount ++;
}
#endif
/*
 * exit_program - restores original terminal attributes before leaving the
 * program.
 */
void exit_program( err )
short int err;
{
    if ( err )
        fprintf(ERROUT, "exit_program: Error Code = %d\n", err );

    MUTEX_LOCK(&init_lock);
    /** Cancel all the longterm reservations (if any)
     * and write out the time-stamps
     */
    if (info_list && (info_list_len > 0) ) {
        int i;

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

            if (info_list[i] && info_list[i]->initialized)
                info_list[i]->initialized= 0;

        }
    }

    MUTEX_UNLOCK(&init_lock);

    if (logtpcc) {
        fclose(logtpcc);
    } else {
        if (logtpcc = fopen(log_file_name, "w") ) {
            fprintf(logtpcc, "ERROR: Client exiting before SYNC with error%d\n",
                err);
            fclose(logtpcc);
        }
    }

    mon_ExitClient( err );
}
#endif WEB_TPCC_CLIENT
exit( err );
#endif
/*
 * clnt_thread_init
 */

```

```

* This function must be called by each work thread
* It returns a pointer to a context that must be passed
* on calls back to this module.
* There is 1 threadInfo entry in an array for each executor thread.
* When an executor thread is started the first thing it does is call
* this clnt_thread_init function. This function creates a context for the
* thread and if longterm reservations are used this function
* initializes the pa handle.
*/
void *clnt_thread_init(void)
{
    int thread_index;
    struct timezone tz;
    thread_info_t *thread_infoP;

    if (iStatsFrequency < 1)
        return(NULL);

    thread_infoP = (thread_info_t *)calloc(1, sizeof(thread_info_t));

    thread_infoP->descr.state = thread_state_init;
    gettimeofday(&thread_infoP->descr.init, &tz);
    thread_infoP->initialized= 1;

    MUTEX_LOCK(&init_lock);
    thread_index = info_list_len++;
    thread_infoP->thread_index = thread_index;
    thread_infoP->thread_id = get_thread_id();

    num_active_threads++;
    info_list =
        (thread_info_t **)realloc((void *)info_list,
            sizeof(thread_info_t *) * info_list_len);

    info_list[thread_index] = thread_infoP;

    MUTEX_UNLOCK(&init_lock);

    if (num_active_threads % 200 == 0)
        err_printf("Thread %d Initialized (currently %d are active).\n",
            thread_index, num_active_threads);

    return(thread_infoP);
}
/*
 * clnt_thread_done
 *
 * Called before a thread exits.
 * Perform some cleanup.
 */
void clnt_thread_done(contextP)
void *contextP;
{
    int all_done = 0;
    int j;
    thread_info_t *infoP = (thread_info_t *)contextP;

    if (!infoP) return;

    MUTEX_LOCK(&init_lock);

    num_active_threads--;

    #if 0
    err_printf("> thread_done, %d active\n", num_active_threads);
    #endif

    set_client_debug_state((void *)infoP, thread_state_done, 0);
    infoP->done = 1;

    if (num_active_threads == 0) {
        all_done = 1;
    }

    if (info_list[infoP->thread_index] != infoP) {
        fprintf(ERROUT, "Strange error: expected to find %d in info_list[%d] and found %d instead\n",
            infoP, infoP->thread_index,
            info_list[infoP->thread_index]);
    }

    MUTEX_UNLOCK(&init_lock);
    if (all_done) {
        int i;
        thread_info_t **curP;

    #if 0
        fprintf(ERROUT, "All Done - exiting\n");
    #endif

        MUTEX_LOCK(&init_lock);
        for (i=0, curP=info_list; i<info_list_len; i++, curP++) {
            free(*curP);
        }
        free(info_list);
        info_list = NULL;
        info_list_len = 0;
        MUTEX_UNLOCK(&init_lock);
    }
    #if 0

```

```

        exit(0);
#endif
}
}
*/
* The following send_*** functions are called from the screen
* module after the transaction data is received in order to
* send the data to the server for processing.
*/
*/
* send_new_order
* Send a new order request to the server
*/
void send_new_order(contextP, dataP)
void *contextP;
newOrder_data_t *dataP;
{
    thread_info_t *thread_context = (thread_info_t *)contextP;
    trpc_status_t trpcStatus;

    DPRINT(("New Order, w_id %d, %d orders\n", dataP->w_id, dataP->o_ol_cnt));
    PRE_RPC_WORK(thread_context, dataP, NEWO_TRANS, dataP->o_all_local == 0);
    CALLTPCC(NewOrder, thread_context, dataP, &trpcStatus);
    POST_RPC_WORK(thread_context, dataP, NEWO_TRANS);
}
*/
* send_payment
* Send a payment request to the server
*/
void send_payment(contextP, dataP)
void *contextP;
payment_data_t *dataP;
{
    trpc_status_t trpcStatus;
    thread_info_t *thread_context = (thread_info_t *)contextP;

    PRE_RPC_WORK(thread_context, dataP, PAYMENT_TRANS,
        dataP->w_id != dataP->c_w_id);
    CALLTPCC(Payment, thread_context, dataP, &trpcStatus);
    POST_RPC_WORK(thread_context, dataP, PAYMENT_TRANS);
}
*/
* send_order_status
* Send an order status request to the server
*/
void send_order_status(contextP, dataP)
void *contextP;
orderStatus_data_t *dataP;
{
    trpc_status_t trpcStatus;
    thread_info_t *thread_context = (thread_info_t *)contextP;

    PRE_RPC_WORK(thread_context, dataP, ORDER_STAT_TRANS, 0);
    CALLTPCC(OrderStatus, thread_context, dataP, &trpcStatus);
    POST_RPC_WORK(thread_context, dataP, ORDER_STAT_TRANS);
}
*/
* send_delivery
* Send a delivery request to the server
*/
void send_delivery(contextP, dataP)
void *contextP;
delivery_data_t *dataP;
{
    trpc_status_t trpcStatus;
    thread_info_t *thread_context = (thread_info_t *)contextP;

    PRE_RPC_WORK(thread_context, dataP, DELIVERY_TRANS, 0);
    CALLTPCC(Delivery, thread_context, dataP, &trpcStatus);
    POST_RPC_WORK(thread_context, dataP, DELIVERY_TRANS);
}
*/
* send_stock_level
* Send a stock level request to the server
*/
void send_stock_level(contextP, dataP)
void *contextP;
stockLevel_data_t *dataP;
{
    trpc_status_t trpcStatus;
    thread_info_t *thread_context = (thread_info_t *)contextP;

    PRE_RPC_WORK(thread_context, dataP, STOCK_TRANS, 0);
    CALLTPCC(StockLevel, thread_context, dataP, &trpcStatus);
    POST_RPC_WORK(thread_context, dataP, STOCK_TRANS);
}
int too_many_errors(contextP)
void *contextP;
{
    thread_info_t *thread_context = (thread_info_t *)contextP;

```

```

        return (thread_context->consecutive_errors > MAX_CONSECUTIVE_ERRORS);
    }
}
/*
* Enroll the client:
* Perform the needed initialization
*/
void init_encina_client(user_id)
int user_id;
{
    int i;
    mon_status_t monStatus;
    char *env_str;
    char serverName[48];
    struct timezone tz;
    struct timeval a_time;
    unsigned long status;
    FILE *rcFile;

#ifdef WIN32
    get_time_init();
    pClientInfo = perfCntDataInit();
#endif
    if (pClientInfo == NULL)
        pClientInfo = malloc(sizeof(total_tran_count_t));
    memset(pClientInfo, 0, sizeof(total_tran_count_t));

    read_mon_environment();

    if (!cellName)
        CHK_STATUS(30, CELL_NAME_UNAVAILABLE,
            "ENCINA_TPM_CELL is not set!");

    MUTEX_INIT(&init_lock);

    info_list = NULL;
    info_list_len = 0;

#ifdef WEB_TPCC_CLIENT
    /* initialize iStatsFrequency */
    iStatsFrequency = 1;
    rcFile = fopen("~/tpccrc", "r");
    if (rcFile != NULL) {
        char buf[100];
        int num = 1;
        while (1) { /* read the whole rcFile */
            num = fscanf(rcFile, "%s", buf);
            if (num <= 0) break;
            if (strcmp(buf, "StatsFrequency") == 0) {
                fscanf(rcFile, "%d", &iStatsFrequency);
                break;
            }
        }
    }
    err_printf("iStatsFrequency=%d\n", iStatsFrequency);
#endif

    gettimeofday(&a_time, &tz);
#ifdef WIN32
    srand(a_time.tv_sec ^ a_time.tv_usec);
#else
    srand48(a_time.tv_sec ^ a_time.tv_usec);
#endif
}

/*
* Enroll the client:
* get the necessary handles.
*/
void enroll_client(user_id)
int user_id;
{
    int i;
    mon_status_t monStatus;
    char *env_str;
    char serverName[48];
    static char *clientName = "tpcc_client";
    unsigned long status;
    static int client_enrolled = 0;

    MUTEX_LOCK(&init_lock);
    if (client_enrolled) {
        MUTEX_UNLOCK(&init_lock);
        return;
    }
    if (useSecurity) {
        client_authnLevel = rpc_c_protect_level_connect;
        client_authzSvc = rpc_c_authz_dce;
    } else {
        client_authnLevel = rpc_c_protect_level_none;
        client_authzSvc = rpc_c_authz_none;
    }

    if (envRetrieval == 0) {
        ENCINA_CALL_RC("mon_RetrieveEnable", mon_RetrieveEnable(FALSE), status);
        CHK_STATUS(status, MON_RETRIEVEENABLE_FAILED,
            "mon_RetrieveEnable failed");
    }
}

```

```

DPRINT(("Cellname: %s\n", cellName));

ENCINA_CALL_RC("mon_InitClient",mon_InitClient(clientName,cellName),
status);
CHK_STATUS(status,MON_INITCLIENT_FAILED,
"mon_InitClient failed");

DPRINT(("mon_SecuritySetDefaults->authn %d, authz %d\n",
client_authnLevel, client_authzSvc);
ENCINA_CALL_RC("mon_SecuritySetDefaults",
mon_SecuritySetDefaults(client_authnLevel,client_authzSvc),
status);
CHK_STATUS(status,MON_SECURITYSET_FAILED,
"mon_SecuritySetDefaults failed");

ENCINA_CALL_RC("mon_SetHandleCacheRefreshInterval",
mon_SetHandleCacheRefreshInterval(300), status);
CHK_STATUS(status,MON_SETREFRESHINTERVAL_FAILED,
"mon_SetHandleCacheRefreshInterval failed");

{
dbInfo_data_t data;
trpc_status_t trpcStatus;
/* Get DB Info -- currently id does not do anything
but it will tell us if there is a server out there.
Better to know instead of when all the terminals
are up and ready
*/
impTPCCNOInfo(&data, &trpcStatus);
if(trpcStatus) {
char msg[100];
sprintf(msg, "TRPC error during db info at init.");
encina_error_message(msg, trpcStatus);
CHK_STATUS(33,NOINFO_TRPC_ERROR,
"TRPC error during db info at init");
}
}

/* Start bg_thread for debug purpose and performance tuning.
* In the final test, we do not start it in order to get the
* best performance.
* On NT, bg_thread may use lots of CPU. But we need to verify it.
*/
if (1)
start_bg_debug_thread();

client_enrolled = 1;
MUTEX_UNLOCK(&init_lock);
}

/*-----*/
/* Read environment parameters */
/*-----*/
static void read_mon_environment()
{
char *env_str;

cellName = getenv("ENCINA_TPM_CELL");
CHECK_ENVIRON(cellName, "ENCINA_TPM_CELL");

if (env_str = getenv("TPCC_ENV_RETRIEVE")) {
envRetrieval = atoi(env_str);
}
}

/*
* dump_pa_ring_buffer() -- For Debugging --
* Dump the ring buffer in the PA we are talking to
* Only works if we are using long term reservation
*/
static void dump_pa_ring_buffer(pa_handle)
trpc_handle_t pa_handle;
{
err_printf("Dumping Ring Buffer of server\n");
admin_trace_DumpRingBuffer((handle_t)pa_handle, "stderr");
}

/* terminal_context_init:
* The same function as thread_init in the thread-pool version.
*
* This function must be called by each terminal when using
* thread pool. After a terminal is logged on, the first thing
* it does is to call this function.
* This function creates a context for the terminal.
* It returns a pointer to a context that must be passed
* on calls back to this module.
*/
void *terminal_context_init(int fdIn)
{
int thread_index;
struct timezone tz;
thread_info_t *thread_infoP;

if (iStatsFrequency < 1)
return(NULL);

```

```

thread_infoP = (thread_info_t *)calloc(1, sizeof(thread_info_t));

thread_infoP->descr.state = thread_state_init;
gettimeofday(&thread_infoP->descr.init, &tz);
thread_infoP->initialized = 1;

MUTEX_LOCK(&init_lock);
thread_index = info_list_len++;
thread_infoP->thread_index = thread_index;
thread_infoP->thread_id = fdIn;

num_active_threads++;
info_list =
(thread_info_t **)realloc((void *)info_list, \
sizeof(thread_info_t *) * \
info_list_len);
info_list[thread_index] = thread_infoP;

MUTEX_UNLOCK(&init_lock);

if (num_active_threads % 200 == 0)
err_printf("Terminal%d Initialized (currently %dare\
active).\n", \
thread_index, num_active_threads);

return(thread_infoP);
}

```

neworder.tacf

```

/*
* Copyright (C) 1991, 1990Transarc Corporation
* All Rights Reserved
*/
/*
* neworder.tacf -- attribute configuration file fortppc server.
* used for transparent binding
*
* $Revision: 1.1 $
* $Date: 1998/11/06 21:10:13 $
* $Log: neworder.tacf,v $
*
* $TALog: neworder.tacf,v $
* Revision 1.1 1998/11/06 21:10:13dongfeng
* - Move all files common to client and server totppc/common
* directory
* [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
*
* Revision 1.2 1998/10/08 18:03:01gerstl
* Changes to allow configurations where some servers only service
* specific transaction types. Split transaction interfaces by type.
* [added by delta gerstl-23515-TPCC-allow-separate-online-transaction-interfacesr1.1]
*
*
*/

[implicit_handle(mon_handle_t handle)]
interface neworder
{
}

```

neworder.tidl

```

/*
* id: $id: $
*
* component_name: encina benchmarks
*
* the following functions list may not be complete.
* functions defined by/via macros may not be included.
*
* functions:
* <fill_me_in>
*
* origins: transarc corp.
*
* (c) copyright transarc corp. 1995, 1993
* all rights reserved
* licensed materials - property of transarc
*
* us government users restricted rights - use, duplication or
* disclosure restricted by gsa adp schedule contract with transarc corp
*/
/*
* history
* $talog: $
*/

/*
* neworder.tidl -- interface definition file fortppcserver.
*

```

```

* $Revision: 1.0 $
* $Date: 1995/10/20 21:55:05 $
* $Log:      tpcc.tidl,v $
*/

[
  uuid(f7065094-5e04-11d2-b351-9e621208aa77),
  version(1.0)
]
interface neworder
{
import "tpm/mon/mon_handle.idl";
import "tpcc_type.idl";

[nontransactional] void
  impTPCCNewOrder([in,out] newOrder_data_t *dataP,
                  [out] trpc_status_t *trpcStatus);

[nontransactional] void
  impTPCCNOInfo([out] dbInfo_data_t *dataP,
               [out]trpc_status_t *trpcStatus);
}

orderstatus.tacf

/*
* Copyright (C) 1991, 1990Transarc Corporation
* All Rights Reserved
*/
/*
* orderstatus.tacf -- attribute configuration file fortppc server.
* used for transparent binding
*
* $Revision: 1.1 $
* $Date: 1998/11/06 21:10:14 $
* $Log:      $
*
* $TALog: orderstatus.tacf,v $
* Revision 1.1 1998/11/06 21:10:14dongfeng
* - Move all files common to client and server totppc/common
*   directory
* [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
*
* Revision 1.2 1998/10/08 18:03:02gerstl
* Changes to allow configurations where some servers only service
* specific transaction types. Split transaction interfaces by type.
* [added by delta gerstl-23515-TPCC-allow-separate-online-transaction-interfacesr1.1]
*
*/

[implicit_handle (mon_handle_t handle)]
interface orderstatus
{
}

orderstatus.tidl

/*
* id: Sid: $
*
* component_name: encina benchmarks
*
* the following functions list may not be complete.
* functions defined by/via macros may not be included.
*
* functions:
* <fill_me_in>
*
* origins: transarc corp.
*
* (c) copyright transarc corp. 1995, 1993
* all rights reserved
* licensed materials - property of transarc
*
* us government users restricted rights - use, duplication or
* disclosure restricted by gsa adp schedule contract with transarc corp
*/
/*
* history
* $Stalog: $
*/

/*
* orderstatus.tidl -- interface definition file fortppcserver.
*
* $Revision: 1.0 $
* $Date: 1995/10/20 21:55:05 $
* $Log:      tpcc.tidl,v $
*/
[
  uuid(06287200-5e05-11d2-8984-9e621208aa77),
  version(1.0)
]
interface orderstatus
{

```

```

import "tpm/mon/mon_handle.idl";
import "tpcc_type.idl";

[nontransactional] void
  impTPCCOrderStatus([in,out] orderStatus_data_t *dataP,
                    [out] trpc_status_t *trpcStatus);
}

payment.tacf

/*
* Copyright (C) 1991, 1990Transarc Corporation
* All Rights Reserved
*/
/*
* payment.tacf -- attribute configuration file fortppc server.
* used for transparent binding
*
* $Revision: 1.1 $
*
*/

[implicit_handle (mon_handle_t handle)]
interface payment
{
}

payment.tidl

/*
* payment.tidl -- interface definition file fortppcserver.
*
* $Revision: 1.0 $
* $Date: 1995/10/20 21:55:05 $
* $Log:      tpcc.tidl,v $
*/
[
  uuid(1341a902-5e05-11d2-bb70-9e621208aa77),
  version(1.0)
]
interface payment
{
import "tpm/mon/mon_handle.idl";
import "tpcc_type.idl";

[nontransactional] void
  impTPCCPayment([in,out] payment_data_t *dataP,
                 [out] trpc_status_t *trpcStatus);
}

screen.C

/* (C)1997 IBM Corporation */
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <ctype.h>
#include <string.h>
#include <math.h>

#include "screen.h"
#include "format.h"
#include "encina.h"

#define USE_INSULTS
#define LOCAL_SESSION_DATA

extern "C" err_printf(...);

extern char const * const blanks;
extern char const * const underscores;
extern char const * const backspaces;

static int clear_eos(InOut *ioP);
static int clear_eos(char *buf);
static int string_empty(char const *text);
static int pos_zero(int const *val);
static int pos_nonzeros(int const **val);

/*****
Screen
*****/
int Screen::reset() {
  int has_data=0;
  pos=0;
  if (dataptr) memset(dataptr, 0, data_len);
  for (int i = 0; fields[i] != NULL; i++) {
    fields[i]->reset();
  }
  return 0;
};

```

```

int Screen::present_empty_fields() {
    if (empty_fields)
        threadP->write(empty_fields, empty_fields_len);
    // threadP->write(end_str, end_str_len);
    return 0;
}
int Screen::present() {
    threadP->write(screen, screen_len);
    threadP->write(session_data, session_data_len);
    if (has_data) {
        for (int i = 0; fields[i] != NULL; i++) {
            fields[i]->display_field(1);
        }
        threadP->write(end_str, end_str_len);
    } else {
        present_empty_fields();
    }
    return 0;
};
int Screen::user_input() {
    int key;
    has_data = 1;
    fields[pos]->start_position();
    threadP->flush();
    // threadP->mark();
    key = fields[pos]->get_field(0);
    do {
        switch (key) {
            case EOF:
                return 0;
                break;
            case Field::NEXT_FIELD:
                if (fields[++pos] == NULL) {
                    pos = 0;
                }
                break;
            case Field::PREV_FIELD:
                if (--pos < 0) {
                    while (fields[++pos] != NULL);
                    pos--;
                }
                break;
            case Field::REDISPLAY:
                present();
                break;
            case Field::ABORT:
                position(1, 2);
                threadP->write(end_str, end_str_len);
                return 0;
            case Field::ENTER:
                if (validate()) {
                    // threadP->pop();
                    return 1;
                }
                break;
        }
        key = fields[pos]->get_field(0);
    } while (1);
    return 0;
}
Screen::~Screen() {
    if (fields != NULL) {
        for (int lpos = 0; fields[lpos] != NULL; lpos++) {
            delete fields[lpos];
        }
        delete [] fields;
    }
    fields=NULL;
}
int Screen::display_status(int status) {
    position(threadP, status_x, status_y);
    threadP->write("Execution Status: ");
    if (status == TRAN_OK) {
        threadP->write("Transaction Committed");
    } else if (status == INVALID_ITEM) {
        threadP->write("Item number is not valid");
    } else {
        threadP->write("ERROR: Rollback -- ");
        threadP->write("Rollback -- ");
        char buf[6];
        format_int(buf, 6, status);
        threadP->write(buf, 5);
    }
    return 0;
}
int Screen::handle() {
    threadP->debug("%s - reset\n", tran_type);
    reset();

    threadP->debug("%s - present\n", tran_type);
    threadP->hold();
    present();
    threadP->write(TRIGGER, 1);
    threadP->debug("%s - user_input\n", tran_type);
    if (user_input()) {
        threadP->write(end_str, end_str_len);
        threadP->write(TRIGGER, 1);
        return -1;
    }
    threadP->flush();
    threadP->hold();
    threadP->debug("%s - process\n", tran_type);
    if (process()) {
        threadP->write(end_str, end_str_len);
        threadP->write(TRIGGER, 1);
        return -1;
    }
    threadP->debug("%s - respond\n", tran_type);
    respond();
    // position(threadP, 1, 2);
    threadP->write(end_str, end_str_len);
    threadP->write(TRIGGER, 1);
    threadP->flush();
    return 0;
}
}
/******
NewOrder
*****
int NewOrder::reset() {
    Screen::reset();
    pos=start_field;
    memset(dataptr, 0, sizeof(*data));
    return 0;
};
NewOrder::NewOrder(User_data *udP, Thread_data *threadP) : Screen(udP, threadP) {
    tran_type = NEWORDER_SERVICE;
    dataptr = data = new NewOrder_data;
    data_len = sizeof(NewOrder_data);

    status_x = 1;
    status_y = 24;

    screen = static_screen;
    empty_fields = static_empty_fields;
#ifdef LOCAL_SESSION_DATA
    session_data = new char[static_session_data_len+1];
    sprintf(session_data, "%s%5d", POS(12,4), user_dataP->warehouse);
#else
    session_data = static_session_data;
    sprintf(session_data, "%s%5d", POS(12,4), user_dataP->warehouse);
#endif
    screen_len = static_screen_len;
    empty_fields_len = static_empty_fields_len;
    session_data_len = static_session_data_len;

    int lpos = 0;
    fields = new Field*[2+MAX_ITEMS*3+1];
    for (int i = 0; i < MAX_ITEMS; i++) {
        fields[lpos++] = genfield(threadP, 3, 9+i, 5, &data->item[i].s_OL_SUPPLY_W_ID);
        fields[lpos++] = genfield(threadP, 10, 9+i, 6, &data->item[i].s_OL_I_ID);
        fields[lpos++] = genfield(threadP, 45, 9+i, 2, &data->item[i].s_OL_QUANTITY);
    }
#ifdef USE_SMART_FIELDS
    if (i > 0) {
        int **tmp = new int *[4];
        tmp[0] = &fields[lpos-6]->pos;
        tmp[1] = &fields[lpos-5]->pos;
        tmp[2] = &fields[lpos-4]->pos;
        tmp[3] = NULL;
        fields[pos-3]->ok_func = (int (*)(void*))pos_nonzeros;
        fields[pos-3]->ok_data = tmp;
        fields[pos-2]->ok_func = (int (*)(void*))pos_nonzeros;
        fields[pos-2]->ok_data = tmp;
        fields[pos-1]->ok_func = (int (*)(void*))pos_nonzeros;
        fields[pos-1]->ok_data = tmp;
    }
#endif
}
#endif
}
start_field = lpos;
fields[lpos++] = genfield(threadP, 29, 4, 4, &data->s_D_ID); /* District */
fields[lpos++] = genfield(threadP, 12, 5, 4, &data->s_C_ID); /* Customer */
fields[lpos++] = NULL;
reset();
};
int NewOrder::validate() {
    if (!fields[start_field]->pos) {
        pos=start_field;
        message(threadP, "District ID is a required field");
        return 0;
    }
    if (!fields[start_field+1]->pos) {
        pos=start_field+1;
        message(threadP, "Customer ID is a required field");
        return 0;
    }
}
int last=-1;
data->s_O_OL_CNT = 0;
data->s_all_local = 1;
data->s_W_ID = user_dataP->warehouse;
for (int i = 0; i < MAX_ITEMS*3; i+=3) {
    if (fields[i]->pos || fields[i+1]->pos || fields[i+2]->pos) {
        if (fields[i]->pos) {
            pos=i;
        }
    }
}
}

```



```

#if defined(USE_INSULTS)
    message(threadP, "Yeah, I think this is a bogus field too.");
#else
    message(threadP, "Warehouse ID is a required field");
#endif
    return 0;
    if(!fields[i+1]->pos) {
        pos=i+1;
#if defined(USE_INSULTS)
        message(threadP, "Umm, WHAT did you want?");
#else
        message(threadP, "Item ID is a required field");
#endif
        return 0;
    }
    if(data->item[i/3].s_OL_QUANTITY<= 0) {
        pos=i+2;
#if defined(USE_INSULTS)
        message(threadP, "So something plus nothing is...");
#else
        message(threadP, "Please enter a quantity greater than 0");
#endif
        return 0;
    }
    if(data->item[i/3].s_OL_SUPPLY_W_ID!=data->s_W_ID) {
        data->s_all_local=0;
    }
    data->s_O_OL_CNT++;
} else if (last < 0) {
    last = i;
}
}
if(data->s_O_OL_CNT <= 0) {
    pos=0;
#if defined(USE_INSULTS)
    message(threadP, "It's kind of pointless without ordering something isn't it?");
#else
    message(threadP, "Please enter an item to order");
#endif
    return 0;
}
// Compress the order lines: some of them may be empty
int ind;
for (i=0,ind=0; ind<data->s_O_OL_CNT ; i++) {
    if(fields[i*3]->pos) {
        if(i > ind) {
            data->item[ind] = data->item[i];
        }
        ind ++;
    }
}
if(i > ind) {
    int j;
    for (j=ind; j<i; j++) {
        /* At least one empty line was skipped */
        data->item[j].s_OL_SUPPLY_W_ID=0;
        data->item[j].s_OL_I_ID=0;
        data->item[j].s_OL_QUANTITY=0;
    }
}
return 1;
}

int NewOrder::respond() {
    int i;
    double amount, total_amount, cost;
    char buf[32];
    position(threadP, 1, 9); clear_eos(threadP);
    position(threadP, 25, 5); threadP->write(data->s_C_LAST);
    position(threadP, 52, 5); threadP->write(data->s_C_CREDIT);
    position(threadP, 15, 6); format_int(buf,10, data->s_O_ID); threadP->write(buf, 9);

    display_status(data->s_transtatus);
    if(data->s_transtatus != TRAN_OK) {
        return -1;
    }

    position(threadP, 25, 5); threadP->write( data->s_C_LAST);
    position(threadP, 52, 5); threadP->write( data->s_C_CREDIT);
    position(threadP, 15, 6); format_int( buf, 10, data->s_O_ID); threadP->write(buf, 9);
    position(threadP, 48, 6); format_int( buf, 3, data->s_O_OL_CNT); threadP->write(buf, 2);
    position(threadP, 61, 4); format_date(buf, 20, data->s_O_ENTRY_D); threadP->write(buf, 19);
    position(threadP, 64, 5); format_float(buf, 6, 2, data->s_C_DISCOUNT/100); threadP->write(buf, 5);
    position(threadP, 59, 6); format_float(buf, 6, 2, data->s_W_TAX/100); threadP->write(buf, 5);
    position(threadP, 74, 6); format_float(buf, 6, 2, data->s_D_TAX/100); threadP->write(buf, 5);
    total_amount = 0;
    for (i=0; i < data->s_O_OL_CNT; i++) {
        position(threadP, 3, 9+i); format_int(buf, 6, data->item[i].s_OL_SUPPLY_W_ID);
threadP->write(buf, 5);
        position(threadP, 10, 9+i); format_int(buf, 7, data->item[i].s_OL_I_ID); threadP->write(
buf, 6);
        position(threadP, 19, 9+i); threadP->write( data->item[i].s_I_NAME);
        position(threadP, 45, 9+i); format_int(buf, 3, data->item[i].s_OL_QUANTITY);
threadP->write(buf, 2);
        position(threadP, 51, 9+i); format_int(buf, 4, data->item[i].s_S_QUANTITY);
threadP->write(buf, 3);
        position(threadP, 58, 9+i); threadP->write(&data->item[i].s_brand_generic, 1);
        position(threadP, 62, 9+i); format_money(buf, 8, data->item[i].s_I_PRICE);
threadP->write(buf, 7);
        position(threadP, 71, 9+i); format_money(buf, 10, data->item[i].s_OL_AMOUNT);
threadP->write(buf, 9);
    }
    /* Clear the screen of any empty input fields */
    position(threadP, 63, 24); threadP->write( "Total:");
    position(threadP, 70, 24); format_money( buf, 10, data->s_total_amount ); threadP->write( buf, 9 );
    return 0;
}

/*****
Payment
*****/
Payment::Payment(User_data *udP, Thread_data *threadP) : Screen(udP, threadP) {
    tran_type = PAYMENT_SERVICE;
    dataptr = data = new Payment_data;
    data_len = sizeof(Payment_data);

    int lpos = 0;
    screen = static_screen;
    empty_fields = static_empty_fields;
#ifdef LOCAL_SESSION_DATA
    session_data = new char[static_session_data_len+1];
    sprintf(session_data, "%s%5d", POS(12,6), user_dataP->warehouse);
#else
    session_data = static_session_data;
    sprintf(session_data, "%s%5d", POS(12,6), user_dataP->warehouse);
#endif
    screen_len = static_screen_len;
    empty_fields_len = static_empty_fields_len;
    session_data_len = static_session_data_len;

    fields = new Field*[7];
    fields[lpos++] = genfield(threadP, 52, 6, 2, &data->s_D_ID); /* District */
    fields[lpos++] = genfield(threadP, 11, 11, 4, &data->s_C_ID); /* Customer # */
    fields[lpos++] = genfield(threadP, 29, 12, 16, (char*)data->s_C_LAST); /* Name */
    fields[lpos++] = genfield(threadP, 33, 11, 5, &data->s_C_W_ID); /* Cust-Warehouse */
    fields[lpos++] = genfield(threadP, 54, 11, 2, &data->s_C_D_ID); /* Cust-District */
    fields[lpos++] = genfield(threadP, 23, 17, 8, &data->s_H_AMOUNT); /* Amount Paid */
    fields[lpos++] = NULL;
#ifdef USE_SMART_FIELDS
    fields[1]->ok_func = (int(*) (void*))pos_zero;
    fields[1]->ok_data = &fields[2]->pos;
    fields[2]->ok_func = (int(*) (void*))pos_zero;
    fields[2]->ok_data = &fields[1]->pos;
#endif
}

int Payment::validate() {
    if(!fields[0]->pos) {
        pos=0;
        message(threadP, "District ID is a required field");
        return 0;
    }
    if(fields[1]->pos) {
#if defined(USE_BYNAME)
        data->s_byname = 0;
#endif
    } else if(fields[2]->pos) {
#if defined(USE_BYNAME)
        data->s_byname = 1;
#endif
    } else {
        pos=1;
        message(threadP, "Customer ID or Name is required");
        return 0;
    }
    if(!fields[3]->pos) {
        pos=3;
        message(threadP, "Customer Warehouse is a required field");
        return 0;
    }
    if(!fields[4]->pos) {
        pos=4;
        message(threadP, "Customer District is a required field");
        return 0;
    }
    if(data->s_H_AMOUNT <= 0) {
        pos=5;
        message(threadP, "Enter a positive amount");
        return 0;
    }
    data->s_W_ID = user_dataP->warehouse;
    return 1;
}

int Payment::respond() {
    if(data->s_transtatus != TRAN_OK) {
        display_status(data->s_transtatus);
        return -1;
    }
}

```

```

char buf[32];
position(threadP, 52, 6);format_int(buf, 3, data->s_D_ID); threadP->write(buf, 2);
position(threadP, 33,11);format_int(buf, 6, data->s_C_W_ID); threadP->write(buf, 4);
position(threadP, 54,11);format_int(buf, 3, data->s_C_D_ID); threadP->write(buf, 2);
position(threadP, 7, 4);threadP->write( data->s_H_DATE );
position(threadP, 1, 7);threadP->write( data->s_W_STREET_1);
position(threadP, 42, 7);threadP->write( data->s_D_STREET_1);
position(threadP, 1, 8);threadP->write( data->s_W_STREET_2);
position(threadP, 42, 8);threadP->write( data->s_D_STREET_2);
position(threadP, 1, 9);threadP->write( data->s_W_CITY);
position(threadP, 22, 9);threadP->write( data->s_W_STATE);
position(threadP, 25, 9);format_zip(buf, 10, data->s_W_ZIP); threadP->write(buf, 10);
position(threadP, 42, 9);threadP->write( data->s_D_CITY);
position(threadP, 63, 9);threadP->write( data->s_D_STATE);
position(threadP, 66, 9);format_zip(buf, 10, data->s_D_ZIP); threadP->write(buf, 10);
position(threadP, 11,11);format_int( buf, 6, data->s_C_ID); threadP->write(buf, 4);
position(threadP, 9,12);threadP->write( data->s_C_FIRST);
position(threadP, 26,12);threadP->write( data->s_C_MIDDLE);
position(threadP, 29,12);threadP->write( data->s_C_LAST);
position(threadP, 58,12);format_date(buf, 10, data->s_C_SINCE); threadP->write( buf, 10);
position(threadP, 9,13);threadP->write( data->s_C_STREET_1);
position(threadP, 58,13);threadP->write( data->s_C_CREDIT);
position(threadP, 9,14);threadP->write( data->s_C_STREET_2);
position(threadP, 58,14);format_float(buf, 6, 2, data->s_C_DISCOUNT/100);threadP->write(buf, 6);
position(threadP, 9,15);threadP->write( data->s_C_CITY);
position(threadP, 30,15);threadP->write( data->s_C_STATE);
position(threadP, 33,15);format_zip(buf, 10, data->s_C_ZIP); threadP->write(buf, 10);
position(threadP, 58,15);format_phone(buf, 18, data->s_C_PHONE ); threadP->write(buf, 18);
position(threadP, 17,17);format_money( buf, 15, data->s_H_AMOUNT);threadP->write(buf, 14);
position(threadP, 55,17);format_money( buf, 16, data->s_C_BALANCE); threadP->write(buf, 15);
position(threadP, 17,18);format_money( buf, 15, data->s_C_CREDIT_LIM); threadP->write(buf, 14);

if(data->s_C_CREDIT[0] == 'B' && data->s_C_CREDIT[1] == 'C') {
    int i, size = strlen((char *)data->s_C_DATA);
    for (i = 0; i < 4; i++) {
        position(threadP, 12,20+i);
        threadP->write(data->s_C_DATA, (size > 50)?50:size);
        size -= 50;
        if (size <= 0) break;
    }
}

return 0;
}

/*****
OrderStatus
*****
OrderStatus::OrderStatus(User_data *udP, Thread_data *threadP) : Screen(udP, threadP) {
    tran_type = ORDERSTATUS_SERVICE;
    dataptr = data = new OrderStatus_data;
    data_len = sizeof(OrderStatus_data);

    status_x=1;
    status_y=25;

    int pos = 0;
    screen = static_screen;
    empty_fields = static_empty_fields;
#ifdef LOCAL_SESSION_DATA
    session_data = new char[static_session_data_len+1];
    sprintf(session_data, "%s%5d", POS(12,4), user_dataP->warehouse);
#else
    session_data = static_session_data;
    sprintf(session_data, "%s%5d", POS(12,4), user_dataP->warehouse);
#endif
    screen_len = static_screen_len;
    empty_fields_len = static_empty_fields_len;
    session_data_len = static_session_data_len;

    fields = new Field*[4];
    fields[pos++] = genfield(threadP, 29, 4, 2,&data->s_D_ID ); /* District */
    fields[pos++] = genfield(threadP, 11, 5, 4,&data->s_C_ID ); /* Customer ID */
    fields[pos++] = genfield(threadP, 44, 5, 16, (char*)data->s_C_LAST ); /* Customer Name */
    fields[pos++] = NULL;
#ifdef USE_SMART_FIELDS
    fields[1]->ok_func = (int (*)(void*))pos_zero;
    fields[1]->ok_data = &fields[2]->pos;
    fields[2]->ok_func = (int (*)(void*))pos_zero;
    fields[2]->ok_data = &fields[1]->pos;
#endif
};

int OrderStatus::validate() {
    if (!fields[0]->pos) {
        pos=0;
        message(threadP, "District ID is a required field");
        return 0;
    }
    if (fields[1]->pos) {
#ifdef defined(USE_BYNAME)
        data->s_byname = 0;
#endif
    } else if (fields[2]->pos) {
#ifdef defined(USE_BYNAME)
        data->s_byname = 1;
#endif
    }
}

} else {
    pos=1;
    message(threadP, "Customer ID or Name is required");
    return 0;
}

data->s_W_ID = user_dataP->warehouse;

return 1;
}

int OrderStatus::respond() {
    display_status(data->s_transtatus);
    if (data->s_transtatus != TRAN_OK)
        return -1;
}

char buf[32];

position(threadP, 11, 5);format_int(buf, 6, data->s_C_ID); threadP->write(buf, 4);
position(threadP, 24, 5);threadP->write(data->s_C_FIRST);
position(threadP, 41, 5);threadP->write(data->s_C_MIDDLE);
position(threadP, 44, 5);threadP->write(data->s_C_LAST);
position(threadP, 15, 6);format_money(buf, 11, data->s_C_BALANCE); threadP->write(buf, 10);
position(threadP, 15, 8);format_int(buf, 10, data->s_O_ID); threadP->write(buf, 9);
position(threadP, 38, 8);format_date(buf, 19, data->s_O_ENTRY_D); threadP->write(buf);
if (data->s_O_CARRIER_ID > 0) {
    position(threadP, 76, 8);
    format_int(buf, 3, data->s_O_CARRIER_ID);
    threadP->write(buf, 2);
}

for (int i=0; i < data->s_ol_cnt; i++) {
    position(threadP, 3, i+10);
    format_int(buf, 6, data->item[i].s_OL_SUPPLY_W_ID);
    threadP->write(buf, 5);

    position(threadP, 14, i+10);
    format_int(buf, 7, data->item[i].s_OL_I_ID);
    threadP->write(buf, 6);

    position(threadP, 25, i+10);
    format_int(buf, 3, data->item[i].s_OL_QUANTITY);
    threadP->write(buf, 2);

    position (threadP, 32, i+10);
    format_money(buf, 10, data->item[i].s_OL_AMOUNT);
    threadP->write(buf, 9);

    position (threadP, 47, i+10);
    format_date(buf, 20, data->item[i].s_OL_DELIVERY_D);
    threadP->write(buf, 19);
}

return 0;
}

/*****
Delivery
*****
Delivery::Delivery(User_data*udP, Thread_data *threadP) : Screen(udP, threadP) {
    tran_type = DELIVERY_SERVICE;
    dataptr = data = new Delivery_data;
    data_len = sizeof(Delivery_data);

    status_x = 1;
    status_y = 8;

    int pos = 0;
    screen = static_screen;
    empty_fields = static_empty_fields;
#ifdef LOCAL_SESSION_DATA
    session_data = new char[static_session_data_len+1];
    sprintf(session_data, "%s%5d", POS(12,4), user_dataP->warehouse);
#else
    session_data = static_session_data;
    sprintf(session_data, "%s%5d", POS(12,4), user_dataP->warehouse);
#endif
    screen_len = static_screen_len;
    empty_fields_len = static_empty_fields_len;
    session_data_len = static_session_data_len;

    fields = new Field*[2];
    fields[pos++] = genfield(threadP, 17, 6, 2,&data->s_O_CARRIER_ID ); /* Carrier Number */

    fields[pos++] = NULL;
};

int Delivery::validate() {
    if (!fields[0]->pos) {
        pos=0;
        message(threadP, "Carrier ID is a required field");
        return 0;
    }
}

time((time_t *)&(data->s_queued_time));

data->s_W_ID = user_dataP->warehouse;

return 1;
}

```

```

int Delivery::respond() {
    if(data->s_transtatus == TRAN_OK {
        position(threadP, status_x, status_y);
        threadP->write("Execution Status: Delivery has been queued");
    } else {
        display_status(data->s_transtatus);
        return -1;
    }
    return 0;
}

/*****
StockLevel
*****/
StockLevel::StockLevel(User_data*udP, Thread_data *threadP) : Screen(udP, threadP) {
    tran_type = STOCKLEVEL_SERVICE;
    dataptr = data = new StockLevel_data;
    data_len = sizeof(StockLevel_data);

    status_x = 1;
    status_y = 10;

    int pos = 0;
    screen = static_screen;
    empty_fields = static_empty_fields;
    session_data = static_session_data;
#ifdef LOCAL_SESSION_DATA
    session_data = new char[static_session_data_len+1];
    sprintf(session_data, "%s%5d%s%2d", POS(12,4), user_dataP->warehouse,
                                                    POS(29,4),
user_dataP->district);
#else
    session_data = static_session_data;
    sprintf(session_data, "%s%5d%s%2d", POS(12,4), user_dataP->warehouse,
                                                    POS(29,4),
user_dataP->district);
#endif
    screen_len = static_screen_len;
    empty_fields_len = static_empty_fields_len;
    session_data_len = static_session_data_len;

    fields = new Field*[2];
    fields[pos++] = genfield(threadP, 24, 6, 2,&data->s_threshold ); /* Threshold */
    fields[pos++] = NULL;
};

int StockLevel::validate(){
    if(data->s_threshold <= 0) {
        pos=0;
        message(threadP, "A positive non-zero threshold is required");
        return 0;
    }
    data->s_W_ID = user_dataP->warehouse;
    data->s_D_ID = user_dataP->district;

    return 1;
}

int StockLevel::respond() {
    display_status(data->s_transtatus);
    if(data->s_transtatus != TRAN_OK)
        return -1;

    position(threadP, 12, 8);
    char buf[5];
    format_int(buf, 4, data->s_low_stock);
    threadP->write(buf, 4);

    return 0;
}

/*****
perform
*****/
int NewOrder::process() {
    if(tran_type == NULL)
        return 0;

    if(encina.tran(data, threadP->contextP, tran_type) < 0) {
        return -1;
    }
    return 0;
}

int Payment::process() {
    if(tran_type == NULL)
        return 0;

    if(encina.tran(data, threadP->contextP, tran_type) < 0) {
        return -1;
    }
    return 0;
}

int StockLevel::process() {
    if(tran_type == NULL)
        return 0;

    if(encina.tran(data, threadP->contextP, tran_type) < 0) {
        return -1;
    }
    return 0;
}

int OrderStatus::process() {
    if(tran_type == NULL)
        return 0;

    if(encina.tran(data, threadP->contextP, tran_type) < 0) {
        return -1;
    }
    return 0;
}

int Delivery::process() {
    if(tran_type == NULL)
        return 0;

    if(encina.tran(data, threadP->contextP, tran_type) < 0) {
        return -1;
    }
    return 0;
}

int Screen::process() {
    if(tran_type == NULL)
        return 0;

    return 0;
}

/*****
Login
*****/
Login::Login(User_data *udP, Thread_data *threadP) : Screen(udP, threadP) {
    tran_type = NULL;
    status_x=1;
    status_y=24;

    dataptr = NULL;
    data_len = 0;

    int pos = 0;
    screen = static_screen;
    screen_len = static_screen_len;
    empty_fields = static_empty_fields;
    empty_fields_len = static_empty_fields_len;

    fields = new Field*[3];
    fields[pos++] = genfield(threadP, 16, 5, 5,&(udP->warehouse) ); //Warehouse
    fields[pos++] = genfield(threadP, 34, 5, 2,&(udP->district) ); //District
    fields[pos++] = NULL;
};

int Login::validate(){
    if(!fields[0]->pos) {
        pos=0;
        message(threadP, "Warehouse ID is a required field");
        return 0;
    }
    if(!fields[1]->pos) {
        pos=1;
        message(threadP, "District ID is a required field");
        return 0;
    }

    return 1;
}

/*****
Menu
*****/
Menu::Menu(User_data *udP, Thread_data *threadP) : Screen(udP, threadP) {
    tran_type = NULL;
    status_x=1;
    status_y=24;

    int pos = 0;
    screen = static_screen;
    screen_len = static_screen_len;
    empty_fields = NULL;
    empty_fields_len = 0;

    fields = NULL;
};

/*****
Static data
*****/
char const * const blanks = " ";
char const * const underscores = "_____";
char const * const backspaces = "\b\b\b\b\b\b\b\b\b\b";

/*****
Utility Functions
*****/

```

```

*****/
static int string_empty(char const *data) {
    return data[0] == 0;
}
static int pos_zero(int const *val) {
    return *val == 0;
}
static int pos_nonzeros(int const **val) {
    int const **ptr;
    for (ptr = val; *ptr; ptr++) {
        if (**ptr == 0)
            return 0;
    }
    return 1;
}
int position(int x, int y, char *buf) {
    int pos = 0;
    buf[pos++] = ESCc;
    buf[pos++] = 'l';
    if (y >= 10) buf[pos++] = (y / 10) + '0';
    buf[pos++] = (y % 10) + '0';
    buf[pos++] = ':';
    if (x >= 10) buf[pos++] = (x / 10) + '0';
    buf[pos++] = (x % 10) + '0';
    buf[pos++] = 'H';
    buf[pos++] = 0;
    return 0;
}
int position(InOut *threadP, int x, int y) {
    char buf[16];
    position(x, y, buf);
    threadP->write(buf);
    return 0;
}
static int clear_eos(InOut *threadP) {
    threadP->write(ESC "J");
    return 0;
}
int message(InOut *threadP, char const *text, int need_flush) {
    position(threadP, 1,25);
    threadP->write(text);
    clear_eos(threadP);
    if (need_flush)
        threadP->flush();
    return 0;
}
static int clear_eos(char *buf) {
    buf[0] = ESCc;
    buf[1] = 'l';
    buf[2] = 'J';
    return 0;
}

```

screen.h

```

/* (C)1997 IBM Corporation */
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <ctype.h>
#ifdef WIN32
#include <termios.h>
#else
#include <time.h>
#endif
#include "field.h"
#include "inout.h"
#include "tpcc.h"
extern int position(int x, int y, char *buf);
extern int position(InOut *ioP, int x, int y);
extern int message(InOut *ioP, char const *text, int need_flush=1);
class User_data {
public:
    int warehouse;
    int district;
};
class Thread_data : public InOut {
public:
    void *contextP;
    Thread_data(int infd, int outfd, void *conP) : InOut(infd, outfd), contextP(conP) {};
};
class Screen {
protected:
    static char const end_str[];
    static int end_str_len;
    int has_data;
    void *dataptr;

```

```

char *tran_type;
char const *screen;
char const *empty_fields;
char *session_data;
int screen_len;
int session_data_len;
int empty_fields_len;
int pos;
int status_x, status_y;
int data_len;
Thread_data *threadP;
public:
    User_data *user_dataP;
    Field **fields;
    virtual char const *isa() { return "Screen"; };
    virtual int reset();
    virtual int present();
    virtual int present_empty_fields();
    virtual int process();
    virtual int user_input();
    virtual int validate() { return 1; };
    virtual int respond() { return 0; };
    int handle();
    int display_status(int status);
    Screen(User_data *udP, Thread_data *thrP) {
        user_dataP = udP;
        threadP = thrP;
        has_data = 0;
        pos = 0;
        fields = NULL;
        screen = empty_fields = session_data = NULL;
        screen_len = session_data_len = empty_fields_len = 0;
    };
    virtual ~Screen();
};
class Login : public Screen {
protected:
    static char const static_screen[];
    static char const static_empty_fields[];
    static char static_session_data[];
    static int static_screen_len;
    static int static_empty_fields_len;
    static int static_session_data_len;
public:
    int validate();
    Login::Login(User_data *udP, Thread_data *thrP);
};
class NewOrder : public Screen {
protected:
    static char const static_screen[];
    static char const static_empty_fields[];
    static char static_session_data[];
    static int static_screen_len;
    static int static_empty_fields_len;
    static int static_session_data_len;
    int start_field;
    void swap_fields(int i, int j);
public:
    NewOrder_data *data;
    int reset();
    NewOrder::NewOrder(User_data *udP, Thread_data *thrP);
    int validate();
    int process();
    int respond();
};
class Payment : public Screen {
protected:
    static char const static_screen[];
    static char const static_empty_fields[];
    static char static_session_data[];
    static int static_screen_len;
    static int static_empty_fields_len;
    static int static_session_data_len;
public:
    Payment_data *data;
    int validate();
    int process();
    int respond();
    Payment(User_data *udP, Thread_data *thrP);
};
class OrderStatus : public Screen {
protected:
    static char const static_screen[];
    static char const static_empty_fields[];
    static char static_session_data[];
    static int static_screen_len;
    static int static_empty_fields_len;
    static int static_session_data_len;
public:
    OrderStatus_data *data;
    int validate();
    int process();
    int respond();
};

```

```

OrderStatus(User_data *udP, Thread_data *thrP);
};

class Delivery : public Screen {
protected:
static char const static_screen[];
static char const static_empty_fields[];
static char static_session_data[];
static int static_screen_len;
static int static_empty_fields_len;
static int static_session_data_len;
public:
Delivery_data *data;
int validate();
int process();
int respond();

Delivery(User_data *udP, Thread_data *thrP);
};

class StockLevel : public Screen {
protected:
static char const static_screen[];
static char const static_empty_fields[];
static char static_session_data[];
static int static_screen_len;
static int static_empty_fields_len;
static int static_session_data_len;
public:
StockLevel_data *data;
int validate();
int process();
int respond();

StockLevel(User_data *udP, Thread_data *thrP);
};

class Menu : public Screen {
protected:
static char const static_screen[];
static char const static_empty_fields[];
static char static_session_data[];
static int static_screen_len;
static int static_empty_fields_len;
static int static_session_data_len;
public:
Menu(User_data *udP, Thread_data *thrP);
};

/* (C)1997 IBM Corporation */
#include "screen.h"

char const NewOrder::static_screen[] =
POS( 1, 3) CLEAR_EOS
POS(36, 3) "New Order"
POS( 1, 4) "Warehouse"
POS(19, 4) "District:"
POS(55, 4) "Date:"
POS( 1, 5) "Customer:"

POS(19, 5) "Name:"
POS(44, 5) "Credit:"
POS(57, 5) "Disc.:"
POS( 1, 6) "Order Number:"
POS(25, 6) "Number of Lines:"
POS(52, 6) "W_Tax:"
POS(67, 6) "D_Tax:"
POS( 2, 8) "Supp_W Item_Num Item_Name"
POS(44, 8) "Qty Stock B/G Price Amount"
;

char const NewOrder::static_empty_fields[] =
POS(29, 4) "____" /* District */
POS(12, 5) "____" /* Customer */

POS( 3, 9) "____"
POS(10, 9) "____"
POS(45, 9) "____"
POS( 3,10) "____"
POS(10,10) "____"
POS(45,10) "____"
POS( 3,11) "____"
POS(10,11) "____"
POS(45,11) "____"
POS( 3,12) "____"
POS(10,12) "____"
POS(45,12) "____"
POS( 3,13) "____"
POS(10,13) "____"
POS(45,13) "____"
POS( 3,14) "____"
POS(10,14) "____"
POS(45,14) "____"

POS( 3,15) "____"
POS(10,15) "____"
POS(45,15) "____"
POS( 3,16) "____"
POS(10,16) "____"
POS(45,16) "____"
POS( 3,17) "____"
POS(10,17) "____"
POS(45,17) "____"
POS( 3,18) "____"
POS(10,18) "____"
POS(45,18) "____"
POS( 3,19) "____"
POS(10,19) "____"
POS(45,19) "____"
POS( 3,20) "____"
POS(10,20) "____"
POS(45,20) "____"
POS( 3,21) "____"
POS(10,21) "____"
POS(45,21) "____"
POS( 3,22) "____"
POS(10,22) "____"
POS(45,22) "____"
POS( 3,23) "____"
POS(10,23) "____"
POS(45,23) "____"
;

char NewOrder::static_session_data[] =
POS(12,4) "#####" /* Warehouse Id */
;

int NewOrder::static_screen_len = sizeof(NewOrder::static_screen) - 1;
int NewOrder::static_empty_fields_len = sizeof(NewOrder::static_empty_fields) - 1;
int NewOrder::static_session_data_len = sizeof(NewOrder::static_session_data) - 1;

/* Payment */
char const Payment::static_screen[] =
POS( 1, 3) CLEAR_EOS
POS(38,3) "Payment"
POS( 1, 4) "Date:"
POS( 1, 6) "Warehouse:"
POS(42, 6) "District:"
POS( 1,11) "Customer:"
POS(17,11) "Cust-Warehouse:"
POS(39,11) "Cust-District:"
POS( 1,12) "Name:"
POS(50,12) "Since:"
POS(50,13) "Credit:"
POS(50,14) "%Disc:"
POS(50,15) "Phone:"
POS( 1,17) "Amount Paid:"
POS(37,17) "New Cust-Balance:"
POS( 1,18) "Credit Limit:"
POS( 1,20) "Cust-Data:"
;

char const Payment::static_empty_fields[] =
POS( 52, 6) "____" /* District */
POS(11,11) "____" /* Customer # */
POS(33,11) "____" /* Cust-Warehouse */
POS(54,11) "____" /* Cust-District */
POS(29,12) "____" /* Name */
POS(23,17) "____" /* Amount Paid */
;

char Payment::static_session_data[] =
POS(12,6) "#####" /* Warehouse */
;

int Payment::static_screen_len = sizeof(Payment::static_screen) - 1;
int Payment::static_empty_fields_len = sizeof(Payment::static_empty_fields) - 1;
int Payment::static_session_data_len = sizeof(Payment::static_session_data) - 1;

/* Order Status */
char const OrderStatus::static_screen[] =
POS( 1, 3) CLEAR_EOS
POS(35,3) "Order-Status"
POS( 1, 4) "Warehouse:"
POS(19, 4) "District:"
POS( 1, 5) "Customer:"
POS(18,5) "Name:"
POS( 1, 6) "Cust-Balance:"
POS( 1, 8) "Order-Number"
POS(26, 8) "Entry-Date:"
POS(60, 8) "Carrier-Number:"
POS( 1, 9) "Supply-W"
POS(14, 9) "Item-Num"
POS(25, 9) "Qty"
POS(33, 9) "Amount"
POS(45, 9) "Delivery-Date"
;

char const OrderStatus::static_empty_fields[] =
POS(29, 4) "____" /* District */
POS(11, 5) "____" /* Customer ID */
POS(44, 5) "____" /* Customer Name */
;

char OrderStatus::static_session_data[] =

```

screen data.C

```

POS(12,4) "#####" /* Warehouse */
;
int OrderStatus::static_screen_len = sizeof(OrderStatus::static_screen) - 1;
int OrderStatus::static_empty_fields_len = sizeof(OrderStatus::static_empty_fields) - 1;
int OrderStatus::static_session_data_len = sizeof(OrderStatus::static_session_data) - 1;

/* Delivery */
char const Delivery::static_screen[]=
  POS(1,3) CLEAR_EOS
  POS(38,3) "Delivery"
  POS(1,4) "Warehouse:"
  POS(1,6) "Carrier Number:"
;
char const Delivery::static_empty_fields[]=
  POS(17,6) "___" /* Carrier Number */
;
char Delivery::static_session_data[]=
  POS(12,4) "#####" /* Warehouse */
;
int Delivery::static_screen_len = sizeof(Delivery::static_screen) - 1;
int Delivery::static_empty_fields_len = sizeof(Delivery::static_empty_fields) - 1;
int Delivery::static_session_data_len = sizeof(Delivery::static_session_data) - 1;

/* Stock level */
char const StockLevel::static_screen[]=
  POS(1,3) CLEAR_EOS
  POS(35,3) "Stock-Level"
  POS(1,4) "Warehouse:"
  POS(19,4) "District:"
  POS(1,6) "Stock Level Threshold:"
  POS(1,8) "Low Stock:"
;
char const StockLevel::static_empty_fields[]=
  POS(24,6) "___" /* Threshold */
;
char StockLevel::static_session_data[]=
  POS(12,4) "#####" /* Warehouse */
  POS(29,4) "##" /* District */
;
int StockLevel::static_screen_len = sizeof(StockLevel::static_screen) - 1;
int StockLevel::static_empty_fields_len = sizeof(StockLevel::static_empty_fields) - 1;
int StockLevel::static_session_data_len = sizeof(StockLevel::static_session_data) - 1;

/* Login */
char const Login::static_screen[]=
  POS(1,1) CLEAR_EOS
  POS(30,3) "Please login."
  POS(5,5) "Warehouse:"
  POS(24,5) "District:"
;
char const Login::static_empty_fields[]=
  POS(16,5) "_____" /* Warehouse */
  POS(34,5) "_____" /* District */
;
int Login::static_screen_len = sizeof(Login::static_screen) - 1;
int Login::static_empty_fields_len = sizeof(Login::static_empty_fields) - 1;

/* Menu */
char const Menu::static_screen[]=
  POS(1,1) CLEAR_EOS
  "(1)New-Order(2)Payment (3)Order-Status (4)Delivery (5)StockLevel (9)Exit"
;
int Menu::static_screen_len = sizeof(Menu::static_screen) - 1;

/* end string */
char const Screen::end_str[] = "033[H\n";
int Screen::end_str_len = sizeof(Screen::end_str) - 1;

server.c

/*
 * server.c
 */
 * $Revision: 1.11 $
 * $Date: 1999/05/06 21:28:30 $
 * $Log: server.c,v $
 *
 * $TALog: server.c,v $
 * Revision 1.11 1999/05/06 21:28:30 oz
 * - Removed all the .. from the includes
 * - Added -I. to the makefiles instead
 * - Moved all the thread related code and connection
 * selection to serverMon.c
 * [from r1.10 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5]
 *
 * Revision 1.10 1999/04/19 20:14:48 oz
 * - Moved all the simulated code to server.c
 * - Created nulldb.c for compilation with no DB
 * [from r1.8 by delta oz-24331-TPCC-move-sim-code-to-common-file,r1.1]
 *
 * Revision 1.9 1999/04/14 18:11:56wenjian
 * Make changes so that the web client data structures for transactions
 * are same as the data structures used in SQL server. It is an important
 * change to integrate with MS TPCC kit. It will also avoidcopyin/copyout
 * for each transaction.

```

```

* [from r1.8 by delta wenjian-24134-TPCC-make-client-data-structure-same-as-server,r1.1]
*
* Revision 1.8 1998/12/11 16:37:58wenjian
* Move some common functions from client/client_utils.c to common/tpcc_utils.c.
* In this version, we only move time_diff_ms(). Need some work in order to
* move other functions like ERROUT.
*
* - Move time_diff_ms() to common/tpcc_utils.c
* [from r1.7 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.2]
*
* Revision 1.7 1998/12/11 16:14:20wenjian
* Add code for checking statistic data in a single variable and collecting
* statistic data based on iStatsFrequency.
*
* - Add time_diff_ms()
* [from r1.6 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.1]
*
* Revision 1.6 1998/11/09 16:59:47wenjian
* In this revision, most of the changes are related to the directory of header
* files after directory reorganization. Other changes include adding or removing
* files to put them in the right directories. Makefiles are written for NT
* platform so that nmake is working on NT now. Need a top level Makefile for all
* the directories.
* [from r1.5 by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.2]
*
* Revision 1.5 1998/11/09 14:48:24wenjian
* In an effort to make a new directory structure for TPCC, this delta
* creates two directories: tpcc/client and tpcc/server. All the files
* for this revision are copied from tpcc/sp-tpcc without any change.
* Further change may be needed for some files due to the change of
* the directory structure.
* [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1]
*
* Revision 1.13 1998/11/06 16:10:56wenjian
* - Change num_mults from 5 to 20
* [from r1.12 by delta wenjian-23646-TPCC-clean-up-source-code,r1.1]
*
* Revision 1.12 1998/10/22 16:25:12wenjian
* Multi-threaded version.
*
* - Define deliLog for the output of delivery server
* - Stop printing to stderr in err_print() since it seems to be too
* expensive on NT. Print to file instead.
* [from r1.11 by delta wenjian-23529-TPCC-integrate-with-SQL-server,r1.2]
*
* Revision 1.11 1998/06/17 15:05:48wenjian
* Somehow, read and write didn't work for socket on NT, although they
* are supposed to work. As a work-around way, use recv and send for
* NT in this revision. We may change them back if the problem is gone.
*
* Use recv and send for socket read and write.
* [from r1.10 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.2]
*
* Revision 1.10 1998/02/17 22:07:06wenjian
* Define macros to deal with the different function names on NT
* [from r1.9 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.1]
*
* Revision 1.9 1998/01/23 15:08:48 oz
* - Updated the SP TPCC directory to the latest files used
* during the SP tpcc audit.
* [from r1.8 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
*
*/
/*
 *
 * TPCC Server
 *
 * There are currently three versions of the TPCC benchmark
 * implemented here: An Encina monitor based benchmark,
 * an Encina Toolkit based benchmark and a DCE only benchmark.
 *
 * This file, server.c, contains all the code that is common to
 * all the versions. Each server has its own main file:
 * serverMon.c for the monitor server, serverTK.c for the toolkit
 * server and serverDce.c for the dce server.
 *
 * Each server is comprised of three main modules: the server specific
 * one (mentioned above), the common one, in this file, and the
 * server part, which is in the SQL files DBInfo.ec, dbInit.ec,
 * delivery.ec, newOrder.ec, orderStatus.ec, payment.ec, stockLevel.ec.
 */
#include <sys/types.h>
#ifdef WIN32
#include <sys/socket.h>
#include <sys/errno.h>
#else
#include <winsock.h>
#include <io.h>
#endif
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#ifdef SOLARIS
#include <dce/pthread.h>
#else /* solaris */
#include <pthread.h>

```

<pre> #endif #include <utils/trace.h> #include <tpm/mon.h> #include "common/utilities.h" #include "server.h" #include "common/tpcc_type.h" #include "common/do_tpcc.h" #include <time.h> #define DEFINE_SERVER_DEBUG #include "serverDebug.h" #if defined(solaris) extern int errno; #endif #ifdef WIN32 #define O_RDONLY _O_RDONLY #define read(A,B,C) recv(A,B,C,0) #define open _open #define close _close #endif #define SIM_ERROR_CODE TPCC_SUCCESS #ifdef WIN32 #define RANDOM rand #else #define RANDOM random #endif #define TPCC_HOME "/tmp" #define TIME_PREFIX_LEN 50 extern char sys_errlist[]; extern time_diff_ms(struct timeval *, struct timeval *); void dprint(char *format, ...); /* * Global variables common to all types of servers */ FILE *server_logtrans = NULL; FILE *deliLog = NULL; int logtrans = -1; FILE *dvry_log = NULL; /* FILE structure for delivery log*/ int dvry_log_fd = -1; /* File descriptor for delivery log*/ int status_log = -1; /* File descriptor for status log */ FILE *deliveryLog = NULL; FILE *deliveryOut = NULL; int serverIdNumber = 0; /* The ID of the server * This is used to identify output */ int serverPid = 0; int num_mults = 80; /* The number of times the matrices are * multiplied (in order to spend some time) */ int server_null_test = 0; int server_init = 0; /* The time (in seconds) the test started * This is used by the deferred delivery * which reports its times as elapsed time * since start time */ int null_with_sleep = 1; /* Sleep for some time when simulating trans */ void err_print(char *format, ...); void logprint(char *format, ...); void open_log_files() { /* open DVRY_LOG to keep delivery transactions logs*/ char logname[MAX_STR_LEN], fname[MAX_STR_LEN]; char buffer[MAX_STR_LEN]; char *tpcc_home; char *log_dir; int bytes; int current_fp; int current; log_dir = getenv("DELIVERY_LOGS"); if (log_dir == NULL) { fprintf(stderr, "DELIVERY_LOGS not specified, using %s\n", TPCC_HOME); log_dir = TPCC_HOME; } /* sprintf(buffer, "%s/status.%d", log_dir, getpid()); status_log = creat(buffer, 0666); */ tpcc_home = getenv("TPCC_HOME"); if (tpcc_home == NULL) { fprintf(stderr, "TPCC_HOME not specified, using /tmp\n"); tpcc_home = "/tmp"; } sprintf(fname, "%s/CURRENT", tpcc_home); current_fp = open(fname, O_RDONLY); </pre>	<pre> bytes = read(current_fp, buffer, MAX_STR_LEN); if (bytes == -1) { fprintf(stderr, "Could not read CURRENT file.\n"); exit(1); } buffer[bytes] = '\0'; current = atoi(buffer); close(current_fp); dvry_log = NULL; } /* * logprintf() -- variable argument function used to print error * and debug statements. Function is called when * any of the debug macros (defined in serverDebug.h) * are used. */ /* * get_thread_id * A function that returns the thread ID of the current thread */ int get_thread_id() { pthread_t thread = pthread_self(); int thread_id = pthread_getunique_np(&thread); return(thread_id); } void print_time_prefix(FILE *file) { time_t cur_timet; char time_str[30]; cur_timet = time(&cur_timet); strftime(time_str, 29, "%X", localtime(&cur_timet)); fprintf(file, "%4d %5d %4d %s - ", serverIdNumber, serverPid, get_thread_id(), time_str); } char *get_time_prefix(char *buffer) { time_t cur_timet; char time_str[30]; int len; cur_timet = time(&cur_timet); strftime(time_str, 29, "%X", localtime(&cur_timet)); len = sprintf(buffer, "%4d %5d %4d %s - ", serverIdNumber, serverPid, get_thread_id(), time_str); if (len >= TIME_PREFIX_LEN) { fprintf(stderr, "TIME_PREFIX_LEN (%d) too small: %d\n", TIME_PREFIX_LEN, len); exit(12); } return(buffer); } void logprintf(char *format, ...) { char formatBuffer[200]; char *fmt = formatBuffer; int fmtLen; va_list ap; va_start(ap, format); fmtLen = TIME_PREFIX_LEN + strlen(format) + 2; if (fmtLen > sizeof(formatBuffer)) { fmt = (char *)malloc(fmtLen); } get_time_prefix(fmt); strcat(fmt, format); if (server_logtrans) { vfprintf(server_logtrans, fmt, ap); fflush(server_logtrans); } else { vfprintf(stderr, fmt, ap); } if (fmt != formatBuffer) free(fmt); va_end(ap); } void err_print(char *format, ...) { char formatBuffer[200]; char *fmt = formatBuffer; int fmtLen; char timeBuffer[128]; va_list ap; va_start(ap, format); fmtLen = TIME_PREFIX_LEN + strlen(format) + 2; </pre>
---	--

```

if(fmtLen > sizeof(formatBuffer)) {
    fmt = (char *)malloc(fmtLen);
}
get_time_prefix(fmt);
strcat(fmt, format);
if(server_logtrans) {
    fprintf(server_logtrans, fmt, ap);
    fflush(server_logtrans);
} else {
    fprintf(stderr, fmt, ap);
}

if (fmt != formatBuffer) free(fmt);
va_end(ap);
}

/*
 * dprint() -- variable argument function used to print debug
 * statements; for use with DPRINT macro.
 */

void dprint(char *format, ...)
{
    va_list ap;
    va_start(ap, format);

    print_time_prefix(stderr);
    fprintf(stderr, format, ap);

    va_end(ap);
}

/** Code that has to do with null servers and simulated DBs */

void mat_mult(int);

#define ROWS 5
#define COLS 5

double matrix_a[ROWS][COLS] = {
    {1.2, 3.4, 2.3, 4.6, 5.2},
    {2.3, 4.5, 1.2, 9.4, 3.1},
    {3.4, 5.2, 3.8, 6.5, 1.6},
    {1.2, 5.3, 6.1, 2.9, 3.8},
    {2.4, 1.2, 3.4, 7.2, 1.0}
};

double matrix_b[ROWS][COLS] = {
    {3.4, 5.9, 2.8, 3.4, 5.6},
    {7.2, 9.3, 4.6, 5.2, 1.3},
    {6.4, 5.2, 8.3, 9.4, 2.3},
    {7.2, 3.4, 6.9, 8.1, 2.3},
    {2.3, 4.5, 7.2, 3.4, 5.8}
};

/* Num of ms to add to RT */
static int rt_increment = 0;

/* Num of ms to add to rt_increment after a certain time.*/
static int more_srv_work = 0;

/* how often (in second) to add more_srv_work to rt_increment*/
static int period_to_add_rt = 7*60;

/* how often (in second) to check if there is transaction */
static int period_to_check_tran = 10;

static struct timespec *get_wait_time(struct timespec *timeP, int tran)
{
    int ran = RANDOM() % 1000;
    int wait;

    if (0) {
        if (ran > 998) {
            timeP->tv_sec = 10;
        } else if (ran > 990) {
            timeP->tv_sec = 5;
        } else if (ran > 970) {
            timeP->tv_sec = 1;
        } else {
            timeP->tv_sec = 0;
        }
        timeP->tv_nsec = 50000000;
        if (tran == NEWO_TRANS) {
            timeP->tv_nsec *= 2;
            timeP->tv_sec *= 2;
        }
    } else {
        int time_ms = 0;
        if (tran == NEWO_TRANS) {
            time_ms = 195;
        } else if (tran == PAYMENT_TRANS) {
            time_ms = 50;
        } else if (tran == ORDER_STAT_TRANS) {
            time_ms = 115;
        } else if (tran == STOCK_TRANS) {
            time_ms = 10;
        } else if (tran == DELIVERY_TRANS) {
            time_ms = 0;
        }
        time_ms += rt_increment;
        timeP->tv_sec = 0;
        timeP->tv_nsec = time_ms * 1000000;
    }
    return(timeP);
}

/** A simulated new order transaction */
void sim_new_order(dataP)
newOrder_data_t *dataP;
{
    int i;
    static int next_id = 100;
    struct timespec wait_time;
    static int lasttime = 0;
    struct timeval now;
    static int periods = 0;

    get_local_time(&now);
    if (now.tv_sec - lasttime > period_to_check_tran) {
        static trans[3]; /* Keep the counts for the last 5 periods */
        lasttime = now.tv_sec;

        if ((trans[1] - trans[0] < 2) &&
            (trans[2] - trans[1] < 2)) {
            rt_increment = 0;
            periods = 0;
            more_srv_work = getenv("TPCC_MORE_SERVER_WORK") ?
                atoi(getenv("TPCC_MORE_SERVER_WORK")) : 0;
            err_printf("Nothing much happening - resetting test\n");
        } else {
            periods++;
            if (periods % (period_to_add_rt / period_to_check_tran) == 0) {
                rt_increment += more_srv_work;
                err_printf("rt_increment now %d\n", rt_increment);
            }
        }
        trans[0] = trans[1];
        trans[1] = trans[2];
        trans[2] = next_id;
    }

    if (null_with_sleep)
        pthread_delay_np(get_wait_time(&wait_time, NEWO_TRANS));

    mat_mult(num_mults);
    printf((char *)dataP->c_last, "BARBARBAR");
    printf((char *)dataP->c_credit, "GC");
    dataP->c_discount = 0.33;
    dataP->o_id = next_id++;
    printf((char *)dataP->entry_date, "17-12-1995.12.33:56");
    dataP->total = 99.1;
    dataP->w_tax = 0.729;
    dataP->d_tax = 0.15;
    for (i=0; i<dataP->o_ol_cnt; i++) {
        dataP->item[i].price = dataP->item[i].ol_i_id > 1000;
        printf((char *)dataP->item[i].name_i, "item %d", i);
        dataP->item[i].s_quantity = i;
        dataP->item[i].brand_generic[0] = i%2 ? 'O' : 'E';
        dataP->item[i].brand_generic[1] = '\0';
        dataP->item[i].ol_amount =
            dataP->item[i].price * dataP->item[i].ol_quantity;
    }

    if ((dataP->item[dataP->o_ol_cnt - 1].ol_i_id < 1) ||
        (dataP->item[dataP->o_ol_cnt - 1].ol_i_id > 100000)) {
        dataP->header.returncode = INVALID_NEWO;
    } else if (RANDOM() % 90 == 0) {
        dataP->header.returncode = SIM_ERROR_CODE;
    } else {
        dataP->header.returncode = TPCC_SUCCESS;
    }
    return;
}

/** A simulated payment transaction */
void sim_payment(dataP)
payment_data_t *dataP;
{
    struct timespec wait_time;

    if (null_with_sleep)
        pthread_delay_np(get_wait_time(&wait_time, PAYMENT_TRANS));
    mat_mult(num_mults);

    dataP->c_id = 1;
    dataP->c_credit_lim = 100.9;
    dataP->c_discount = 0.2;
    dataP->c_balance = 11.1;

    printf((char *)dataP->c_first, "%-16s", "c_first");
    printf((char *)dataP->c_middle, "%-2s", "MI");
    printf((char *)dataP->c_last, "%-16s", "c_last");
    printf((char *)dataP->c_street_1, "%-20s", "c_street_1");
    printf((char *)dataP->c_street_2, "%-20s", "c_street_2");
    printf((char *)dataP->c_city, "%-20s", "c_city");
    printf((char *)dataP->c_state, "%-2s", "PA");
}

```



```

sprintf((char *)dataP->c_zip, "%-9s", "152111111");
sprintf((char *)dataP->c_phone, "%-16s", "6522573904218222");
sprintf((char *)dataP->c_date, "%-19s", "28-11-1995");
sprintf((char *)dataP->c_credit, "%-2s", "GC");
sprintf((char *)dataP->pay_date, "%-19s", "17-12-1995.12:39:13");
sprintf((char *)dataP->d_street_1, "%-20s", "d_street_1");
sprintf((char *)dataP->d_street_2, "%-20s", "d_street_2");
sprintf((char *)dataP->d_city, "%-20s", "d_city");
sprintf((char *)dataP->d_state, "%-2s", "PA");
sprintf((char *)dataP->d_zip, "%-9s", "152111111");
sprintf((char *)dataP->w_street_1, "%-20s", "w_street_1");
sprintf((char *)dataP->w_street_2, "%-20s", "w_street_2");
sprintf((char *)dataP->w_city, "%-20s", "w_city");
sprintf((char *)dataP->w_state, "%-2s", "OH");
sprintf((char *)dataP->w_zip, "%-9s", "142411111");

if (RANDOM() % 70 == 0) {
    dataP->header.returncode = SIM_ERROR_CODE;
} else {
    dataP->header.returncode = TPCC_SUCCESS;
}
}

/** A simulated stock level transaction */
void sim_stock_level(dataP)
stockLevel_data_t *dataP;
{
    struct timespec wait_time;

    if (null_with_sleep)
        pthread_delay_np(&get_wait_time(&wait_time, STOCK_TRANS));

    mat_mult(num_mults);

    dataP->stock_count = 12;
    if (RANDOM() % 80 == 0) {
        dataP->header.returncode = SIM_ERROR_CODE;
    } else {
        dataP->header.returncode = TPCC_SUCCESS;
    }
}

/** A simulated delivery transaction */
void sim_delivery(dataP)
delivery_data_t *dataP;
{
    struct timespec wait_time;

    if (null_with_sleep)
        pthread_delay_np(&get_wait_time(&wait_time, DELIVERY_TRANS));

    dataP->start_queue = 2.2;
    dataP->header.returncode = TPCC_SUCCESS;
}

/** A simulated order status transaction */
void sim_order_status(dataP)
orderStatus_data_t *dataP;
{
    int i;
    struct timespec wait_time;

    if (null_with_sleep)
        pthread_delay_np(&get_wait_time(&wait_time, ORDER_STAT_TRANS));

    mat_mult(num_mults);

    dataP->c_id = dataP->c_id ? dataP->c_id : 99;
    strcpy((char *)dataP->c_first, "Jerome");
    strcpy((char *)dataP->c_middle, "LB");
    strcpy((char *)dataP->c_last, "Trevoe");
    dataP->c_balance = 90.78;
    dataP->o_id = 99;
    strcpy((char *)dataP->entry_date, "06-12-1995.16:42:28");
    dataP->o_carrier_id = 9;
    dataP->o_ol_cnt = 7;

    for (i=0; i<dataP->o_ol_cnt; i++) {
        dataP->item[i].ol_supply_w_id = 1;
        dataP->item[i].ol_i_id = dataP->w_id * 10 + dataP->d_id;
        dataP->item[i].ol_quantity = 10 * (i+1);
        dataP->item[i].ol_amount = dataP->item[i].ol_quantity * 10.1;
        strcpy((char *)dataP->item[i].delivery_date, "NOT DELIVR");
    }

    if (RANDOM() % 90 == 0) {
        dataP->header.returncode = SIM_ERROR_CODE;
    } else {
        dataP->header.returncode = 0;
    }
}

/*
 * mat_mult
 * Multiply the above two matrices
 */

```

```

static void mat_mult(iter)
int iter;
{
    float res[ROWS][COLS];
    int i, j, k;
    int a_num_rows = ROWS;
    int a_num_columns = COLS;
    int b_num_rows = ROWS;
    int b_num_columns = COLS;

    for (; iter>0; iter--) {
        for (i=0; i<a_num_rows; i++) {
            for (j=0; j<b_num_columns; j++) {
                res[i][j] = 0;
                for (k=0; k<b_num_rows; k++) {
                    res[i][j] += matrix_a[i][k] * matrix_b[k][j];
                }
                matrix_a[i][j] = res[i][j];
            }
        }
        pthread_yield();
    }
}

```

server.h

```

/*
 * server.h
 *
 * $Revision: 1.11 $
 * $Date: 1999/05/06 21:28:31 $
 * $Log: $
 *
 * $STALog: server.h,v $
 * Revision 1.11 1999/05/06 21:28:31 oz
 * - Removed all the .. from the includes
 * - Added -I. to the makefiles instead
 * - Moved all the thread related code and connection
 * selection to serverMon.c
 * [from r1.9 by delta oz-24309-TPCC-add-oracle8.1-code.r1.5]
 *
 * Revision 1.9 1999/01/12 20:52:59wenjian
 * Define MAPOBJNAMEFORMAT so that the server processes and dll can communicate
 * via the shared file mappings.
 * [from r1.8 by delta wenjian-23856-TPCC-integrate-with-NT-performance-monitor.r1.1]
 *
 * Revision 1.8 1998/12/14 20:27:57wenjian
 * Made corresponding changes due to data structure change of tran_info_t.
 *
 * - change server_tran_t
 * [from r1.7 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.3]
 *
 * Revision 1.7 1998/12/11 16:14:20wenjian
 * Add code for checking statistic data in a single variable and collecting
 * statistic data based on iStatsFrequency.
 *
 * - Add server_tran_t and server_info_t
 * [from r1.6 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.1]
 *
 * Revision 1.6 1998/11/09 16:59:48wenjian
 * In this revision, most of the changes are related to the directory of header
 * files after directory reorganization. Other changes include adding or removing
 * files to put them in the right directories. Makefiles are written for NT
 * platform so that nmake is working on NT now. Need a top level Makefile for all
 * the directories.
 * [from r1.5 by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.2]
 *
 * Revision 1.5 1998/11/09 14:48:25wenjian
 * In an effort to make a new directory structure for TPCC, this delta
 * creates two directories: tpcc/client and tpcc/server. All the files
 * for this revision are copied from tpcc/sp-tpcc without any change.
 * Further change may be needed for some files due to the change of
 * the directory structure.
 * [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1]
 *
 * Revision 1.9 1998/10/22 15:33:05wenjian
 * Make changes to Encina server code to connect with SQL server and add
 * callsql.c and sql directory.
 *
 * Add delivery_sql_t to deal with SYSTEMTIME struct used in SQL
 * [from r1.7 by delta wenjian-23529-TPCC-integrate-with-SQL-server,r1.1]
 *
 * Revision 1.7 1998/01/23 15:08:50 oz
 * - Updated the SP TPCC directory to the latest files used
 * during the SP tpcc audit.
 * [from r1.6 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27.r1.1]
 *
 *
 */

/** server.h */

/** Declarations common to all the server modules */

```

```

#ifdef TPCC_SERVER_H
#define TPCC_SERVER_H

#include "common/tpcc_type.h"

#define get_dbname_from_id(i) rmList[i].dbName
#define MAPOBJNAMEFORMAT "srv_%s_PA%d"

#ifdef WIN32
typedef struct {
    delivery_data_t data;
    SYSTEMTIME queue_time;
} delivery_sql_t;
#endif

typedef enum {
    mon_server = 11
} server_type_t;

typedef struct {
    int num;
    double RTtotal;
    int RTcount;
} server_tran_t;

typedef struct {
    server_tran_t tran[MAX_TRAN_TYPE + 1];
    int total_trans;
} server_info_t;

extern int server_no_db;
extern int serverIdNumber;
extern int server_init;
extern server_type_t server_type;
extern int get_db_for_wh(int);

#endif /* TPCC_SERVER_H */

serverDebug.h

/*
 * serverDebug.h
 *
 * $Revision: 1.5 $
 * $Date: 1998/11/09 14:48:25 $
 * $Log: serverDebug.h,v $
 * Revision 4.4 95/05/16 10:55:40tpcc (TPCC Benchmark)
 * Added necessary RCS ident strings
 *
 */
#ifdef SERVER_DEBUG
#define SERVER_DEBUG

#include <utils/trace.h>

#ifdef DEFINE_SERVER_DEBUG
long serverDebug = 0;
#else
extern long serverDebug;
#endif

#ifdef TRACE_TRANS
#define TRACETRAN(list)logprintf list
#else
#define TRACETRAN(list)
#endif

#ifdef DEBUG_SERVER
#define AUDITLOG(list)if (serverDebug & AUDIT_TRANS)UNCOND_EVENT list
#define NEWOLOG(list)if (serverDebug & DBG_NEWO) err_printf list
#define PAYLOG(list)if (serverDebug & DBG_PAY) err_printf list
#define OSLOG(list)if (serverDebug & DBG_OS) err_printf list
#define STKLOG(list)if (serverDebug & DBG_STK) err_printf list
#define DEBUGP(list)if (serverDebug) err_printf list
#else
#define AUDITLOG(list)
#define NEWOLOG(list)
#define PAYLOG(list)
#define OSLOG(list)
#define STKLOG(list)
#define DELLOG(list)
#define DEBUGP(list)
#endif

#define ERRLOG(list)err_printf list
#define SQL_RET_CODE(var, code) var = (code)

/* Fix DPRINT to write on a debugging unit that can get set differently
for delivery */

#ifdef UNIT_TEST
#define DPRINT(list)dprint list
#define DELPRINT(list)delprint list
#else
#define DPRINT(list)
#define DELPRINT(list)
#endif

```

serverDebug.h

```

#endif

#define DBG_NEWO      0x0001
#define DBG_PAY      0x0002
#define DBG_OS       0x0004
#define DBG_STK      0x0008
#define DBG_DEL      0x0010
#define DBG_ERR      0x0020
#define AUDIT_TRANS  0x0100

#endif /* SERVER_DEBUG */

serverMon.c

/*
 * serverMon.c
 *
 * $Revision: 1.23 $
 * $Date: 1999/05/28 19:44:17 $
 * $Log: serverEncina.c,v $
 *
 * $TALog: serverMon.c,v $
 * Revision 1.23 1999/05/28 19:44:17wenjian
 * Add create_null_connection and clean_null_connection so that
 * we can run with NULL DB.
 * [from r1.22 by delta wenjian-24433-TPCC-clean-up-and-update,r1.4]
 *
 * Revision 1.22 1999/05/28 14:30:22wenjian
 * - Fix a bug for callingget_thread_data()
 * [from r1.21 by delta wenjian-24433-TPCC-clean-up-and-update,r1.3]
 *
 * Revision 1.21 1999/05/26 16:29:59wenjian
 * Sync with Austin code, and sync code for Oracle DB and SQL server.
 *
 * - Fix some minor bugs
 * [from r1.20 by delta wenjian-24433-TPCC-clean-up-and-update,r1.2]
 *
 * Revision 1.20 1999/05/06 21:28:31 oz
 * - Removed all the .. from the includes
 * - Added -.I. to the makefiles instead
 * - Moved all the thread related code and connection
 * selection to serverMon.c
 *
 * - Added connection and thread related code.
 * - pre_DB and post_DB always get called.
 * - get_db_ready has only 2 parameters
 * - added tran_info_t, total_tran_count_t, and thread_info_t
 * - Added clean_thread_data and get_thread_data
 * - Preallocate all the connections if necessary after
 * initializing theDB.
 * - A connection is created by calling theDB specific
 * create_connection() which returns a void* handle to
 * be passed to all the transactions.
 * [from r1.18 by delta oz-24309-TPCC-add-oracle8.1-code,r1.5]
 *
 * Revision 1.18 1999/04/20 15:11:28 oz
 * [merge of changes from 1.13 to 1.17 into 1.12]
 *
 * Revision 1.17 1999/04/19 20:14:49 oz
 * - Moved all the simulated code to server.c
 * - Created nulldb.c for compilation with no DB
 * [from r1.13 by delta oz-24331-TPCC-move-sim-code-to-common-file,r1.1]
 *
 * Revision 1.12 1999/01/12 20:53:00wenjian
 * - Call initialization functionperfSrvDataInit to create the shared file
 * mapping for this server process
 * - Change server_info to pServerInfo
 * [from r1.11 by delta wenjian-23856-TPCC-integrate-with-NT-performance-monitor,r1.1]
 *
 * Revision 1.11 1998/12/14 20:27:58wenjian
 * Made corresponding changes due to data structure change of tran_info_t.
 *
 * - Made changes for server_tran_t
 * - Add tran_type to FUNCTION_BEGIN,FUNCTION_END,pre_DB, and post_DB
 * [from r1.10 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server,r1.3]
 *
 * Revision 1.10 1998/12/11 16:14:20wenjian
 * Add code for checking statistic data in a single variable and collecting
 * statistic data based on iStatsFrequency.
 *
 * - Change pre_oracle to pre_DB, post_oracle to post_DB
 * - Add code to collect server RT in a global var
 * [from r1.9 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server,r1.1]
 *
 * Revision 1.9 1998/12/08 18:55:22wenjian
 * - Remove "statsFrequency=" command line argument.
 * - Check rpc header for stats
 * [from r1.8 by delta wenjian-23785-TPCC-pass-statsFrequency-from-client-to-server,r1.1]
 *
 * Revision 1.8 1998/12/07 20:04:16wenjian
 * Remove interfaces for explicit bindings
 * [from r1.7 by delta wenjian-23742-TPCC-update-with-Raliegh-code,r1.2]
 *
 * Revision 1.7 1998/11/24 21:46:03wenjian
 * - Remove COLLECT_TIMESTAMPS;use command line argument iStatsFrequency
 * instead
 * - Take care ofMULTIPLE_INTERFACEand SINGLE_INTERFACE
 * [from r1.6 by delta wenjian-23742-TPCC-update-with-Raliegh-code,r1.1]

```

serverMon.c

<pre> * Revision 1.6 1998/11/09 16:59:48wenjian * In this revision, most of the changes are related to the directory of header * files after directory reorganization. Other changes include adding or removing * files to put them in the right directories. Makefiles are written for NT * platform so that nmake is working on NT now. Need a top level Makefile for all * the directories. * [from r1.5 by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.2] * * Revision 1.5 1998/11/09 14:48:25wenjian * In an effort to make a new directory structure for TPCC, this delta * creates two directories: tpcc/client and tpcc/server. All the files * for this revision are copied from tpcc/sp-tpcc without any change. * Further change may be needed for some files due to the change of * the directory structure. * [added by delta wenjian-23677-TPCC-reorganize-directory-structure,r1.1] * * Revision 1.36 1998/11/06 16:10:56wenjian * - Increase the range for the number of threads communicating to the DB. * - Print warning if the number of threads is out of the range. * [from r1.35 by delta wenjian-23646-TPCC-clean-up-source-code,r1.1] * * Revision 1.35 1998/10/22 21:30:47wenjian * [merge of changes from 1.20 to 1.29 into 1.34] * * Revision 1.29 1998/10/08 14:18:02dongfeng * Add codes for doing web-based tpcc. * [from r1.20 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.1] * * Revision 1.34 1998/10/22 19:43:38wenjian * [merge of changes from 1.28 to 1.32 into 1.30] * * Revision 1.32 1998/10/22 16:25:13wenjian * Multi-threaded version. * * - Open server_logtrans for err_printf() and deliLog for delivery server * - Call get_time_init() for get_local_time * - Changes for multi-threaded version * [from r1.31 by delta wenjian-23529-TPCC-integrate-with-SQL-server,r1.2] * * Revision 1.31 1998/10/22 15:33:05wenjian * Make changes to Encina server code to connect with SQL server and add * callsql.c and sql directory. * * - Add SYSTEMTIME*queue_time to deferred_dvry_t for NT * - Allocate space and set value for queue_time * - Pass delivery_sql_t instead of delivery_data_t for SQLdel on NT * [from r1.28 by delta wenjian-23529-TPCC-integrate-with-SQL-server,r1.1] * * Revision 1.30 1998/10/08 18:03:03gerstl * Changes to allow configurations where some servers only service * specific transaction types. Split transaction interfaces by type. * * Transaction interface support is based upon a bitmap passed to the * server. * [from r1.28 by delta gerstl-23515-TPCC-allow-separate-online-transaction-interfaces,r1.1] * * Revision 1.28 1998/10/07 15:49:49gerstl * [merge of changes from 1.22 to 1.26 into 1.27] * * Revision 1.26 1998/09/03 20:22:02wenjian * Sync with Austin code: mostly use servrMon.c in Austin. * [from r1.25 by delta wenjian-23183-TPCC-sync-AIX-code-with-Austin,r1.4] * * Revision 1.25 1998/09/03 16:07:12wenjian * Remove UNCOND_EVENT which is not in austin code. * [from r1.24 by delta wenjian-23183-TPCC-sync-AIX-code-with-Austin,r1.3] * * Revision 1.24 1998/09/02 15:43:30wenjian * Define num_worker_threads. * [from r1.23 by delta wenjian-23183-TPCC-sync-AIX-code-with-Austin,r1.2] * * Revision 1.23 1998/08/28 18:30:02wenjian * This delta sync the TPCC code with Austin. * * - Take care of 1 thread per PA * - Update with Austin code * - Remove the old code starting from #ifdef THIS_WAS_THE_OLD_CODE * - Remove impTPCCDvryInfo() * [from r1.22 by delta wenjian-23183-TPCC-sync-AIX-code-with-Austin,r1.1] * * Revision 1.27 1998/09/26 10:27:33 oz * Changes for NT. * [from r1.22 by delta oz-23339-TPCC-update-for-NT,r1.1] * * Revision 1.22 1998/08/18 14:38:43wenjian * Minor change * [from r1.20 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.4] * * Revision 1.20 1998/02/17 22:07:06wenjian * Minor changes for NT * [from r1.19 by delta wenjian-21750-TPCC-changes-for-porting-on-NT,r1.1] * * Revision 1.19 1998/01/30 15:12:28 oz * - Remove the explicit binding functions from the tidl * files and from serverMon.c * [from r1.18 by delta oz-21697-TPCC-remove-explicit-binding-code,r1.2] * * Revision 1.18 1998/01/24 14:17:06 oz </pre>	<pre> * - User server name to identify server and name delivery file * - Use env variable HOME instead of/home/encina if HOME is set * * - Removed the machine list * The server ID is computed from the first number found in the host name * [from r1.17 by delta oz-21687-TPCC-use-server-name-to-identify-process,r1.1] * * Revision 1.17 1998/01/23 21:59:00 oz * - In order to simplify the Encina TPCC code: Merge the four * online transactions into 1 interface * - Moved all the scripts to a scripts subdirectory * - Removed unused files * [from r1.16 by delta oz-21671-TPCC-merge-online-transaction-interfaces,r1.1] * * Revision 1.16 1998/01/23 15:08:53 oz * - Updated the SP TPCC directory to the latest files used * during the SP tpcc audit. * [from r1.15 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1] * * * serverMon.c * * Code that is monitor specific. * */ #include <sys/types.h> #ifdef WIN32 #include <unistd.h> #else #include <process.h> #include <winsock.h> #endif #include <stdio.h> #include <stdarg.h> #include <time.h> #include <tmx/tmxa.h> #include <tc/tc.h> #include <tpm/mon/mon_server.h> #ifdef solaris #include <dce/pthread.h> #else /* solaris */ #include <pthread.h> #endif /* solaris */ #include <utils/trace.h> #include "common/databuf.h" #include "common/utilities.h" #include "serverDebug.h" #include "common/delivery.h" #ifdef MULTIPLE_INTERFACE #include "common/neworder.h" #include "common/payment.h" #include "common/stocklevel.h" #include "common/orderstatus.h" #else #include "common/tpcc_trans.h" #endif #include "server.h" #define FUNCTION_BEGIN(name, dataP, tran_type, infoP) \ pre_DB(name, &(dataP)->header, tran_type, infoP); #define FUNCTION_END(name, dataP, tran_type, infoP) \ post_DB(name, &(dataP)->header, tran_type, infoP); #ifdef WIN32 #define CASECMP(x,y) _stricmp(x,y) == 0 #define getpid _getpid extern void TPCExit(); #else #define CASECMP(x,y) streacmp(x,y) == 0 extern void TPCExit(); #endif extern void *create_connection(); extern void clean_connection(void *); extern int get_db_ready(char *, int); inModule("serverMon"); static void start_deferred_delivery_threads(); static void queue_delivery(delivery_data_t *dataP); static void *create_null_connection(); static void clean_null_connection(void *ptr); extern int server_null_test; extern void err_printf(char *format, ...); extern int get_db_ready(char *, int); extern void logprintf(char *format, ...); extern server_info_t *perfSrvDataInit(char *, int); extern void *start_bg_thread(); static void get_mon_server_env(); server_type_t server_type = mon_server; </pre>
--	---

```

char *tpcc_serverName = NULL;
char *dbName = NULL;
int total_num_warehouses;
int num_deferred_dvry_threads = 1;
int num_worker_threads = 1;
int dvry_queue_size = 3000;
server_info_t *pServerInfo = NULL;
char oracle_home[256]; /* will be used in tpccpl.c */

typedef struct {
    pthread_mutex_t    lock;
    pthread_cond_t    q_cond;
    pthread_cond_t    work_cond;

    int                num_waiters; /* Number of new requests waiting */

    int                head, tail;
    int                allocated; /* Total size of the queue */
    int                size; /* Num elements currently there */
#ifdef WIN32
    SYSTEMTIME        *queue_time;
#endif
    delivery_data_t    *data;
} deferred_dvry_t;

static deferred_dvry_t deferred_dvry_data;
#define MAX_DVRY_QUEUE deferred_dvry_data.allocated

/*
 * Information about one transaction type
 */
typedef struct {
    int num;
    int errs;
    double RT;
} tran_info_t;

/*
 * total_tran_count_t
 *
 * structure that holds the total count of transaction of each type
 * as well as the reposne times.
 */
typedef struct {
    tran_info_t tran[MAX_TRAN_TYPE + 1];
    int errors;
} total_tran_count_t;

typedef struct {
    void *cnP; /* DB specific connection to be used by this thread */

    int calls; /* Number of times it was used */
    int errors; /* Total number of errors on this connection */
    int calls_last_err; /* Number of calls when the last error occurred */
    int consecutive_errs; /* Number of consecutive errs */
    int connect_time; /* Time (seconds) connections was created */

    /* For debug */
    int state; /* State of the connection */
    struct timeval tran_time; /* Time this tran started */
    int cur_tran_type;
    void *cur_tran_dataP;
    total_tran_count_t stat;
    int printed;
} thread_info_t;

#define SVR_STATE_NONE            0
#define SVR_STATE_SENT           1
#define SVR_STATE_REPLIED       2
#define SVR_STATE_ERR           3

/* Connection related data structures */
static void clean_thread_data(void *ptr);

pthread_key_t thread_key;
pthread_mutex_t init_lock;
thread_info_t *info_array = NULL; /* Array of thread data */
int num_threads = 0; /* number of threads that have already been init */
int next_thread = 0; /* next thread id: next entry in the array */

int preallocate_cn = 1; /* Should all connections be preallocated */
int num_connections = 0;
int num_allocated = 0;

static thread_info_t *get_thread_data();

static void display_mon_env()
{
    char *env_str;
    char envMsg[64];
#define DISPLAY_ENV_VAR(var) \
    if((env_str = getenv(var)) != NULL) { \
        UNCOND_EVENT("%s == %s\n", var, env_str); \
    } else { \
        UNCOND_EVENT("%s not set\n", var); \
    }

    UNCOND_EVENT("TPCCServer display env. ID: %d\n", serverIdNumber);

    /*
     * For debugging purpose: have the first PA
     * display the following information
     */

    if((serverIdNumber & 0xff) == 0) {
        DISPLAY_ENV_VAR("RPC_SUPPORTED_PROTSEQS");
        DISPLAY_ENV_VAR("RPC_UNSUPPORTED_NETADDRS");
        DISPLAY_ENV_VAR("ENCINA_BINDING_TIMEOUT");
        DISPLAY_ENV_VAR("ENCINA_THREAD_POOL_QUEUE_LENGTH");
        DISPLAY_ENV_VAR("ENCINA_THREAD_POOL_QUEUE_LENGTH");
        DISPLAY_ENV_VAR("ENCINA_TPOOL_SIZE");
        DISPLAY_ENV_VAR("ENCINA_RPC_THREAD_STACK_SIZE");
    }

    /* get_server_index() -- This is used for debug purposes only
     *
     * Return the server index for this server.
     * By convention, all the client machines have similar
     * names with different numbers, as in client1 client2, ...
     * If the convention is followed the server index is the first
     * number found. Otherwise, it is 0.
     */
    static int get_server_index()
    {
        int i, ind;
        char host_name[128];
        if (0 == gethostname(host_name, sizeof(host_name))) {
            err_printf("Machine is on host %s\n", host_name);
            ind = strcspn(host_name, "0123456789");
            return(atol(host_name + ind));
        }

        return(0);
    }

    static parse_cmd_line(int argc, char *argv[], char **scheduling, int *interface_type)
    {
        int nextInd = 1;
        char usageStr[128];
        int envRetrieval;
        if((nextInd + 3) > argc) {
            sprintf(usageStr,
                "Not enough parameters. Usage: %s[-no_db] interfaces schedulingPolicy envRetrievalFlag\n",
                argv[0]);
            fprintf(stderr, "%s\n", usageStr);
            mon_TerminateServer(usageStr);
        } else {
            if(strncmp(argv[nextInd], "-no_db") == 0) {
                server_null_test = 1;
                fprintf(stderr, " ---= NULL test ==---\n");
                nextInd++;
            }
            *interface_type = strtol(argv[nextInd++], NULL, 0);
            *scheduling = argv[nextInd++];
            envRetrieval = atoi(argv[nextInd++]);

            while(nextInd < argc) {
                if(strncmp(argv[nextInd], "db:", 3) == 0) {
                    dbName = argv[nextInd] + 3;
                    nextInd++;
                } else if(strncmp(argv[nextInd], "dvry=", 5) == 0) {
                    num_deferred_dvry_threads = atol(argv[nextInd] + 5);
                    if(num_deferred_dvry_threads < 0 || num_deferred_dvry_threads > 200) {
                        err_printf("num_deferred_dvry_threads was %d (>200), reset to 10\n",
                            num_deferred_dvry_threads);
                        num_deferred_dvry_threads = 10;
                    }
                    nextInd++;
                } else if(strncmp(argv[nextInd], "dvryQ=", 6) == 0) {
                    dvry_queue_size = atol(argv[nextInd] + 6);
                    if(dvry_queue_size < 1 || dvry_queue_size > 200000)
                        dvry_queue_size = 10;
                    nextInd++;
                } else {
                    serverDebug = atol(argv[nextInd++]);
                }
            }
        }
    }

    static void set_scheduling(char *scheduling)

```

<pre> mon_paAccess_t paAccess; UNCOND_EVENT("Setting Scheduling Policy: %s\n", scheduling); if(CASECMP(scheduling, "MON_CONCURRENT_SHARED")) { paAccess = MON_CONCURRENT_SHARED; } else if(CASECMP(scheduling, "MON_EXCLUSIVE")) { num_deferred_dvry_threads = 1; paAccess = MON_EXCLUSIVE; } else if(CASECMP(scheduling, "MON_SHARED")) { num_deferred_dvry_threads = 1; paAccess = MON_SHARED; } else { err_printf("Invalid Policy: %s\n", scheduling); mon_TerminateServer("Invalid scheduling policy specified."); } ENCINA_CALL("mon_SetSchedulingPolicy", mon_SetSchedulingPolicy(paAccess)); } static void register_interfaces(int interface_type) { extern FILE *deliLog; char *env_str; int env_val; UNCOND_EVENT("Registering interfaces\n"); num_worker_threads = 0; /* interface_type is a bitmap of the interfaces this * server needs to support. */ #ifdef MULTIPLE_INTERFACE if (interface_type & NEWO_INTERFACE) { ENCINA_CALL("mon_InitServerInterface", mon_InitServerInterface(MON_SERVER_INTERFACE(neworder, 1.0)); } if (interface_type & PAYMENT_INTERFACE) { ENCINA_CALL("mon_InitServerInterface", mon_InitServerInterface(MON_SERVER_INTERFACE(payment, 1.0)); } if (interface_type & ORDER_STAT_INTERFACE) { ENCINA_CALL("mon_InitServerInterface", mon_InitServerInterface(MON_SERVER_INTERFACE(orderstatus, 1.0)); } if (interface_type & STOCK_INTERFACE) { ENCINA_CALL("mon_InitServerInterface", mon_InitServerInterface(MON_SERVER_INTERFACE(stocklevel, 1.0)); } #else if (interface_type & ONLINE_INTERFACES) { ENCINA_CALL("mon_InitServerInterface", mon_InitServerInterface(MON_SERVER_INTERFACE(tpccTrans, 1.0)); } #endif if (interface_type & DELIVERY_INTERFACE) { #ifdef WIN32 deliLog = fopen("deliLog.out", "w"); #endif if (num_deferred_dvry_threads > 0) { start_deferred_delivery_threads(); } ENCINA_CALL("mon_InitServerInterface", mon_InitServerInterface(MON_SERVER_INTERFACE(delivery, 1.0)); } else { num_deferred_dvry_threads = 0; } /* ENCINA_TPOOL_SIZE and ENCINA_APPL_TPOOL_SIZE * are set in tpccCommon.tcl for each * server started. If we are delivery only, we don't care * about it, otherwise we need to adjust num_worker_threads */ if (interface_type & ONLINE_INTERFACES) { if ((env_str = getenv("ENCINA_APPL_TPOOL_SIZE")) != NULL) { env_val = atoi(env_str); if (env_val >= 0 && env_val < 1000) num_worker_threads += env_val; else { err_printf("ENCINA_APPL_TPOOL_SIZE was %d, reset to 10\n", env_val); num_worker_threads += 10; } } } if ((env_str = getenv("ENCINA_TPOOL_SIZE")) != NULL) { env_val = atoi(env_str); if (env_val >= 0 && env_val < 1000) num_worker_threads += env_val; else { err_printf("ENCINA_TPOOL_SIZE was %d, reset to 10\n", env_val); num_worker_threads += 5; } } if (num_worker_threads < 1) num_worker_threads = 1; } </pre>	<pre> void main(argc, argv) int argc; char *argv[]; { extern FILE *server_logtrans; int rc; int pa_num; char *scheduling = ""; int rmlid; char intermediary[256]; extern int serverPid; int interface_type = ALL_INTERFACE; int status; inFunction("server_Init"); /* hard code first for a quick test */ /* getenv didn't work, though we have ORACLE_HOME defined */ /* strcpy(oracle_home, getenv("ORACLE_HOME")); */ /* strcpy(oracle_home, "/home/oracle817/app/oracle/product/8.1.7"); server_logtrans = fopen("server_print.out", "w"); get_time_init(); serverPid = getpid(); UNCOND_EVENT("TPCC Server Starting\n"); /* Use the top 8 bits of the serverIdNumber to store the server index */ serverIdNumber = (get_server_index() & 0xff) * 1000; parse_cmd_line(argc, argv, &scheduling, &interface_type); display_mon_env(); DEBUGP("Debug level set at %d\n", serverDebug); DEBUGP("Creating thread data key"); if (status = pthread_keycreate(&thread_key, clean_thread_data)) { fprintf(stderr, "init_global_data : pthread_keycreate failed: %d\n", status); mon_TerminateServer("Cannot create a key for the thread data"); } mon_RetrieveEnable(FALSE); err_printf("Setting scheduling %s.\n", scheduling); set_scheduling(scheduling); err_printf(" Registering interfaces \n"); register_interfaces(interface_type); err_printf("Calling mon_init\n"); ENCINA_CALL("mon_InitServer", mon_InitServer()); ENCINA_CALL("mon_SetHandleCacheRefreshInterval", mon_SetHandleCacheRefreshInterval(300)); pa_num = mon_RetrievePaNum(); tpcc_serverName = mon_RetrieveServerId(); if (pa_num > 0) serverIdNumber += pa_num; err_printf("PA Number %d, serverId %d (%s)\n", pa_num, serverIdNumber, tpcc_serverName); num_connections = num_deferred_dvry_threads + num_worker_threads; if ((rc = get_db_ready(dbName, 0)) != 0) { char msg[128]; sprintf(msg, "failed to open database tpcc/tpcc: %d", rc); WARNING("%s\n", msg); err_printf("%s\n", msg); mon_TerminateServer(msg); } if (preallocate_cn num_connections == 1) { int i; thread_info_t *curP; /* Preallocate all the desired connections */ logprintf("Preallocating %d connections to the DB\n", num_connections); info_array = (thread_info_t *) calloc(num_connections, sizeof(*info_array)); for (i=0, curP = info_array; i < num_connections; i++, curP++) { if (server_null_test) curP->cnP = create_null_connection(); else curP->cnP = create_connection(); } num_allocated = num_connections; } /* initialize pServerInfo */ #ifdef WIN32 pServerInfo = perfSrvDataInit(tpcc_serverName, pa_num); #endif if (pServerInfo == NULL) pServerInfo = malloc(sizeof(server_info_t)); memset(pServerInfo, 0, sizeof(server_info_t)); #ifdef WIN32 start_bg_thread(); </pre>
--	--

```

#endif
err_printf(">> Calling mon_BeginService()\n");
ENCINA_CALL("mon_BeginService", mon_BeginService());

fprintf(stderr, "mon_BeginService returned ... terminating\n");
TPCexit();
}

static void clean_thread_data(void *ptr) {
    thread_info_t *threadP = (thread_info_t *)ptr;
    if (server_null_test)
        clean_null_connection(threadP->cnP);
    else
        clean_connection(threadP->cnP);
    err_printf("Closing connection 0x%p. Called %d, %d errors\n",
        threadP->cnP, threadP->calls, threadP->errors);
}

/*
 * The routine executed by the deferred delivery thread
 *
 * Logic:
 *
 * Wait until there is a valid request in the deferred delivery data.
 * After processing the request data_valid is set to FALSE
 * (allowing new requests to be queued).
 * This is a simple fixed size queue implemented in a cyclic array
 */
static void deferred_delivery()
{
    thread_info_t *infoP;
    pthread_mutex_lock(&deferred_dvry_data.lock);

    while (1) {
        if (deferred_dvry_data.size > 0) {
            /*
             * There is a request to be processed
             */
            int ind = deferred_dvry_data.head % MAX_DVRY_QUEUE;
#ifdef WIN32
            delivery_sql_t dbData;
            dbData.data = deferred_dvry_data.data[ind];
            dbData.queue_time = deferred_dvry_data.queue_time[ind];
#else
            delivery_data_t data = deferred_dvry_data.data[ind];
#endif

            deferred_dvry_data.head ++;
            deferred_dvry_data.size --;

            if (deferred_dvry_data.num_waiters > 0)
                pthread_cond_signal(&deferred_dvry_data.q_cond);

            if (deferred_dvry_data.head % 1000 == 0) {
                err_printf("Processed %d deferred deliveries so far, queue size %d\n",
                    deferred_dvry_data.head,
                    deferred_dvry_data.size);
            }

            if (deferred_dvry_data.head > deferred_dvry_data.tail) {
                err_printf("Error: Deferred Queue: head %d > tail %d\n",
                    deferred_dvry_data.head,
                    deferred_dvry_data.tail);
                continue;
            }
            pthread_mutex_unlock(&deferred_dvry_data.lock);
#ifdef WIN32
            if (server_null_test) {
                sim_delivery(&dbData);
            } else {
                infoP = get_thread_data();
                do_delivery(infoP->cnP, &dbData);
            }
#else
            if (server_null_test) {
                sim_delivery(&data);
            } else {
                infoP = get_thread_data();
                do_delivery(infoP->cnP, &data);
            }
#endif
        }

        DPRINT("Deferred: Locking\n");
        pthread_mutex_lock(&deferred_dvry_data.lock);
    } else {
        /*
         * Wait for a request to be queued
         */
        DPRINT("Deferred delivery waiting\n");
        pthread_cond_wait(&deferred_dvry_data.work_cond,
            &deferred_dvry_data.lock);
        DPRINT("Deferred: Awake\n");
    }
}

/*
 * queue_delivery
 *
 * Queue a delivery request to be processed in the background
 * The queue is implemented as a simple queue of size 1.
 * If data_valid is true: there is already a request waiting in the queue
 * Sleep on a condition variable until the queue is empty.
 * Once the queue is empty put the request in the queue, wake up the
 * background thread and leave.
 */
static void queue_delivery(dataP)
    delivery_data_t *dataP;
{
    struct timeval now;
    int waited = 0;
    static int last_report_time = 0;
#ifdef WIN32
    SYSTEMTIME queue_time;
#endif

    DPRINT("queue: Locking\n");
    pthread_mutex_lock(&deferred_dvry_data.lock);
    DPRINT("queue: Locked\n");

    while (deferred_dvry_data.size >= MAX_DVRY_QUEUE) {
        /* The request queue is full
         * Wait until a request is processed and removed from the queue.
         */
        deferred_dvry_data.num_waiters ++;
        DPRINT(">> queue_delivery: %d waiters, size %d\n",
            deferred_dvry_data.num_waiters, deferred_dvry_data.size);
        DPRINT("Queue Delivery waiting. %d waiters\n",
            deferred_dvry_data.num_waiters);
        pthread_cond_wait(&deferred_dvry_data.q_cond,
            &deferred_dvry_data.lock);
        deferred_dvry_data.num_waiters --;
        waited ++;
    }
    DPRINT("Queueing delivery\n");
    /*
     * There is room in the queue.
     * Enter the request and wake up the background thread
     */
#ifdef WIN32
    GetLocalTime(&queue_time);
    deferred_dvry_data.size ++;
    deferred_dvry_data.data[deferred_dvry_data.tail % MAX_DVRY_QUEUE] = *dataP;
    deferred_dvry_data.queue_time[deferred_dvry_data.tail % MAX_DVRY_QUEUE] = queue_time;
#else
    get_local_time(&now);
    dataP->start_queue = (double)now.tv_sec + (now.tv_usec / 1000000.0);
    deferred_dvry_data.size ++;
    deferred_dvry_data.data[deferred_dvry_data.tail % MAX_DVRY_QUEUE] = *dataP;
    if (now.tv_sec - last_report_time > 29) {
        err_printf("queue_delivery - %d waiters, size %d\n",
            deferred_dvry_data.num_waiters, deferred_dvry_data.size);
        last_report_time = now.tv_sec;
    }
#endif
}

/*
 * start_deferred_delivery_threads
 *
 * Initialize the deferred delivery data structure and start
 * a background thread to process the delivery requests
 */
static void start_deferred_delivery_threads()
{
    pthread_t thread;
    int i;
    int rc;

    pthread_mutex_init(&deferred_dvry_data.lock, pthread_mutexattr_default);
    pthread_cond_init(&deferred_dvry_data.work_cond, pthread_condattr_default);
    pthread_cond_init(&deferred_dvry_data.q_cond, pthread_condattr_default);
    deferred_dvry_data.num_waiters = 0;
    deferred_dvry_data.head = 0;
    deferred_dvry_data.tail = 0;
    deferred_dvry_data.size = 0;
    deferred_dvry_data.allocated = dvry_queue_size;
    deferred_dvry_data.data =
        (delivery_data_t *) malloc(dvry_queue_size * sizeof(delivery_data_t));
#ifdef WIN32
    deferred_dvry_data.queue_time =

```

```

        (SYSTEMTIME*)malloc(dvry_queue_size * sizeof(SYSTEMTIME));
#endif

/*
 * Create the background delivery thread.
 */
err_printf("Starting %d deferred delivery threads, queue size %d\n",
          num_deferred_dvry_threads,
          dvry_queue_size);
for (i=0;i<num_deferred_dvry_threads; i++) {
    if((rc = pthread_create(&thread,
                          pthread_attr_default,
                          (pthread_startroutine_t)deferred_delivery,
                          (pthread_addr_t0) != 0) {
        WARNING("Failed to create delivery thread rc=%d\n", rc);
        exit(1);
    }
    (void)pthread_detach(&thread);
}

void exit_program(code)
int code;
{
char errMsg[55];
sprintf(errMsg, "exit_program called with code %d", code);
fprintf(stderr, "%s\n", errMsg);

TPCexit();

mon_TerminateServer(errMsg);
}

static char *thread_state_to_str(int state)
{
char *retval;
switch(state) {
    case SVR_STATE_NONE: retval = "None"; break;
    case SVR_STATE_SENT: retval = "Sent"; break;
    case SVR_STATE_REPLIED: retval = "Replied"; break;
    case SVR_STATE_ERR: retval = "Err"; break;
    default: retval = "unknown"; break;
}
return retval;
}

static thread_info_t *get_thread_data() {
thread_info_t *dataP;
struct timeval cur_time;

/* Get a thread structure.
 * Each thread always uses the same connection.
 * The first time the thread tries to talk to theDB it creates
 * a connection, initializes it and stores it in a thread global
 * data structure.
 *
 * There is a special case for the single connection case: If there
 * is exactly one connection then it is global and not per thread.
 * There may be many threads but it is assumed that the application is
 * responsible for synchronizing the threads so that no two threads
 * ever use the connection at the same time.
 */
if(num_connections == 1) {
    dataP = &info_array[0];
} else {
    pthread_getspecific(thread_key, (pthread_addr_t *)&dataP);
}
if(dataP == NULL) { /* No connection assigned to this thread */
    pthread_mutex_lock(&init_lock); /* Initialize a connection */
    get_local_time(&cur_time);

    fprintf(stderr, "get_cn> initializing threads\n");

    if(preallocate_cn) {
        if(next_thread >= num_allocated) {
            fprintf(stderr, "Too many threads, not enough connections\n");
            mon_TerminateServer("Too many threads, not enough connections");
        }
        dataP = &info_array[next_thread++];
    } else {
        dataP = (thread_info_t *)malloc(sizeof(thread_info_t));
        memset(dataP, (char)0, sizeof(*dataP));
        if(server_null_test)
            dataP->cnP = create_null_connection();
        else
            dataP->cnP = create_connection();
    }
    pthread_setspecific(thread_key, dataP); /* Store it */

    fprintf(stderr, "get_cn> initialized connection 0x%x\n", dataP);
    pthread_mutex_unlock(&init_lock);
}
return dataP;
}

static void pre_DB(char *name, data_header *headerP,
                  int tran_type, thread_info_t *infoP)
{
    struct timeval tp;
    DPRINT("> %s", name);
    get_local_time(&tp);
    if(infoP != NULL) {
        infoP->cur_tran_type = tran_type;
        infoP->calls++;
        infoP->state = SVR_STATE_SENT;
        infoP->tran_time = tp;
    }

    headerP->start_time.sec = tp.tv_sec;
    headerP->start_time.usec = tp.tv_usec;
}

static void post_DB(char *name, data_header *headerP,
                  int tran_type, thread_info_t *infoP)
{
    struct timeval tp;
    DPRINT("< %s\n", name);
    get_local_time(&tp);
    headerP->end_time.sec = tp.tv_sec;
    headerP->end_time.usec = tp.tv_usec;
    headerP->dtype = serverIdNumber;

    if(infoP != NULL){
        infoP->tran_time = tp;
        infoP->state = SVR_STATE_REPLIED;
    }

    pServerInfo->tran[tran_type].num++;
    /* store the RT info for this server */
    if (tran_type <= MAX_TRAN_TYPE && tran_type > 0) {
        pServerInfo->tran[tran_type].RTtotal +=
            time_diff_ms(&(headerP->end_time), &(headerP->start_time));
        pServerInfo->tran[tran_type].RTcount ++;
    }
}

/*
 * ----- The following are the entry points
 * for the RPCs arriving at the Server
 */

void impTPCCDbInfo(dataP, trpcStatus)
dbInfo_data_t *dataP;
trpc_status_t *trpcStatus;
{
    UNCOND_EVENT("> impTPCCDbInfo");
    err_printf("> impTPCCDbInfo");
    dataP->server_id = serverIdNumber;
    err_printf("< impTPCCDbInfo");
}

void impTPCCNOInfo(dataP, trpcStatus)
dbInfo_data_t *dataP;
trpc_status_t *trpcStatus;
{
    impTPCCDbInfo(dataP, trpcStatus);
}

void impTPCCNewOrder(dataP, trpcStatus)
newOrder_data_t *dataP;
trpc_status_t *trpcStatus;
{
    static int numCalls = 0;
    thread_info_t *infoP = get_thread_data();
    FUNCTION_BEGIN("NewOrder", dataP, NEWO_TRANS, infoP);
    if(server_null_test) {
        sim_new_order(dataP);
    } else {
        do_new_order(infoP->cnP, dataP);
    }

    if((dataP->header.returncode != TPCC_SUCCESS) &&
       (dataP->header.returncode != INVALID_NEWO)) {
        logprintf("< impTPCCNewOrder; rc=%d, sql=%d, isam=%d\n",
                dataP->header.returncode,
                dataP->header.sql_code,
                dataP->header.isam_code);
    } else if(dataP->header.returncode == INVALID_NEWO) {
        DPRINT("< impTPCCNewOrder INVALID_NEWO\n");
    }
    if(++numCalls % 10000 == 0) {
        err_printf("impTPCCNewOrder so far %d\n", numCalls);
    }
    FUNCTION_END("NewOrder", dataP, NEWO_TRANS, infoP);
}

void impTPCCPayment(dataP, trpcStatus)
payment_data_t *dataP;
trpc_status_t *trpcStatus;
{

```

```

static int numCalls = 0;
thread_info_t *infoP = get_thread_data();
FUNCTION_BEGIN("Payment", dataP, PAYMENT_TRANS, infoP);
if (server_null_test) {
    sim_payment(dataP);
} else {
    do_payment(infoP->cnP, dataP);
}

if (dataP->header.returncode != TPCC_SUCCESS) {
    logprintf("< impTPCCPayment; rc=%d, sql=%d, isam=%d\n",
        dataP->header.returncode,
        dataP->header.sql_code,
        dataP->header.isam_code);
}
if (++numCalls % 10000 == 0) {
    err_printf("impTPCCPayment so far %d\n", numCalls);
}
FUNCTION_END("Payment", dataP, PAYMENT_TRANS, infoP);
}

void impTPCCOrderStatus(dataP, trpcStatus)
orderStatus_data_t *dataP;
trpc_status_t *trpcStatus;
{
    thread_info_t *infoP = get_thread_data();
    FUNCTION_BEGIN("OrderStatus", dataP, ORDER_STAT_TRANS, infoP);
    if (server_null_test) {
        sim_order_status(dataP);
    } else {
        do_order_status(infoP->cnP, dataP);
    }

    if (dataP->header.returncode != TPCC_SUCCESS) {
        logprintf("< impTPCCOrderStatus; rc=%d, sql=%d, isam=%d\n",
            dataP->header.returncode,
            dataP->header.sql_code,
            dataP->header.isam_code);
    }
    FUNCTION_END("OrderStatus", dataP, ORDER_STAT_TRANS, infoP);
}

void impTPCCStockLevel(dataP, trpcStatus)
stockLevel_data_t *dataP;
trpc_status_t *trpcStatus;
{
    thread_info_t *infoP = get_thread_data();
    FUNCTION_BEGIN("StockLevel", dataP, STOCK_TRANS, infoP);
    if (server_null_test) {
        sim_stock_level(dataP);
    } else {
        do_stock_level(infoP->cnP, dataP);
    }

    if (dataP->header.returncode != TPCC_SUCCESS) {
        logprintf("< impTPCCStockLevel; rc=%d, sql=%d, isam=%d\n",
            dataP->header.returncode,
            dataP->header.sql_code,
            dataP->header.isam_code);
    }
    FUNCTION_END("StockLevel", dataP, STOCK_TRANS, infoP);
}

void impTPCCDelivery(dataP, trpcStatus)
delivery_data_t *dataP;
trpc_status_t *trpcStatus;
{
#ifdef WIN32
    delivery_sql_t dbData;
#endif

    thread_info_t *infoP = NULL;
    FUNCTION_BEGIN("DELIVERY", dataP, DELIVERY_TRANS, infoP);
    if (num_deferred_dvry_threads > 0) {
        queue_delivery(dataP);
    } else {
#ifdef WIN32
        if (server_null_test) {
            sim_delivery(&dbData);
        } else {
            infoP = get_thread_data();
            do_delivery(infoP->cnP, &dbData);
        }
    }
#else
        if (server_null_test) {
            sim_delivery(dataP);
        } else {
            infoP = get_thread_data();
            do_delivery(infoP->cnP, dataP);
        }
    }
#endif

    if (dataP->header.returncode != TPCC_SUCCESS) {
        logprintf("< impTPCCDelivery; rc=%d, sql=%d, isam=%d\n",
            dataP->header.returncode,
            dataP->header.sql_code,

```

```

        dataP->header.isam_code);
    }
    FUNCTION_END("DELIVERY", dataP, DELIVERY_TRANS, infoP);
}

/* functions in order to run with NULL database */
static void *create_null_connection() {
    static cn_num = 0;
    int *id = (int *)malloc(sizeof(int));
    *id = cn_num++;
    return id;
}

static void clean_null_connection(void *ptr) {
    free(ptr);
    return;
}

```

stocklevel.tacf

```

/*
 * Copyright (C) 1991, 1990Transarc Corporation
 * All Rights Reserved
 */
/*
 * stocklevel.tacf -- attribute configuration file fortpcc server.
 * used for transparent binding
 *
 * $Revision: 1.1 $
 * $Date: 1998/11/06 21:10:16 $
 * $Log: tpcc.tacf,v $
 *
 * $TALog: stocklevel.tacf,v$
 * Revision 1.1 1998/11/06 21:10:16dongfeng
 * - Move all files common to client and server totpcc/common
 * directory
 * [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
 *
 * Revision 1.2 1998/10/08 18:03:04gerstl
 * Changes to allow configurations where some servers only service
 * specific transaction types. Split transaction interfaces by type.
 * [added by delta gerstl-23515-TPCC-allow-separate-online-transaction-interfacesr1.1]
 *
 *
 */

```

```

[implicit_handle (mon_handle_t handle)]
interface stocklevel
{
}

```

stocklevel.tidl

```

/*
 * id: Sid: $
 *
 * component_name: encina benchmarks
 *
 * the following functions list may not be complete.
 * functions defined by/via macros may not be included.
 *
 * functions:
 * <fill_me_in>
 *
 * origins: transarc corp.
 *
 * (c) copyright transarc corp. 1995, 1993
 * all rights reserved
 * licensed materials - property of transarc
 *
 * us government users restricted rights - use, duplication or
 * disclosure restricted by gsa adp schedule contract with transarc corp
 */
/*
 * history
 * $talog: $
 */
/*
 * stocklevel.tidl -- interface definition file fortpccserver.
 *
 * $Revision: 1.0 $
 * $Date: 1995/10/20 21:55:05 $
 * $Log: tpcc.tidl,v $
 */
[
    uuid(1dda58c8-5e05-11d2-bd18-9e621208aa77),
    version(1.0)
]
interface stocklevel
{
import "tpm/mon/mon_handle.idl";
import "tpcc_type.idl";

[nontransactional] void

```



```

impTPCCStockLevel([in,out]stockLevel_data_t *dataP,
                  [out] trpc_status_t *trpcStatus);
}

                                tpcc.h

#ifndef TPCC_H_INCLUDED
#define TPCC_H_INCLUDED
/*****
*/
/* File:tpcc.h
/* created: 8-26-91
*/
/* program description:
*/
/* This module contains global variables and data definitions
/* for the tpcc application.
*/
/*****
*/
#include "../common/tpcc_type.h"

#define TPCC_H

/-----*/
/* Global numbers, constants,...
*/
/-----*/

#define INVALID_ITEM 100
#define TRAN_OK 0
#define REMOTE_WAREHOUSE 17

#define FORM_DATE 1
#define FORM_DATETIME 2

#define MAX_ITEMS 15

/-----*/
/* transaction structures
*/
/-----*/

typedef orderStatus_data_t OrderStatus_data;
typedef newOrder_data_t NewOrder_data;
typedef stockLevel_data_t StockLevel_data;
typedef delivery_data_t Delivery_data;
typedef payment_data_t Payment_data;

/*****
Compatibility for older .sqc files
*****/
#define s_C_BALANCE c_balance
#define s_C_CITY c_city
#define s_C_CREDIT c_credit
#define s_C_CREDIT_LIM c_credit_lim
#define s_C_DATA c_data
#define s_C_DISCOUNT c_discount
#define s_C_D_ID c_d_id
#define s_C_FIRST c_first
#define s_C_ID c_id
#define s_C_LAST c_last
#define s_C_MIDDLE c_middle
#define s_C_PHONE c_phone
#define s_C_SINCE c_date
#define s_C_STATE c_state
#define s_C_STREET_1 c_street_1
#define s_C_STREET_2 c_street_2
#define s_C_W_ID c_w_id
#define s_C_ZIP c_zip
#define s_D_CITY d_city
#define s_D_ID d_id
#define s_D_STATE d_state
#define s_D_STREET_1 d_street_1
#define s_D_STREET_2 d_street_2
#define s_D_TAX d_tax
#define s_D_ZIP d_zip
#define s_H_AMOUNT h_amount
#define s_H_DATE pay_date
#define s_I_NAME name_i
#define s_I_PRICE price
#define s_OL_AMOUNT ol_amount
#define s_OL_DELIVERY_D delivery_date
#define s_OL_I_ID ol_i_id
#define s_OL_QUANTITY ol_quantity
#define s_OL_SUPPLY_W_ID ol_supply_w_id
#define s_O_CARRIER_ID o_carrier_id
#define s_O_ENTRY_D entry_date
#define s_O_ID o_id
#define s_O_OL_CNT o_ol_cnt
#define s_S_QUANTITY s_quantity
#define S_QUANTITY quantity
#define s_W_CITY w_city
#define s_W_ID w_id
#define s_W_STATE w_state
#define s_W_STREET_1 w_street_1
#define s_W_STREET_2 w_street_2
#define s_W_TAX w_tax

```

```

#define s_W_ZIP w_zip
#define s_all_local_o_all_local
#define s_brand_generic brand_generic
#define s_exec_status exec_status
#define s_low_stock stock_count
#define s_ol_cnt o_ol_cnt
#define s_queued_time queued_time
#define s_status_line statusline
#define s_threshold threshold
#define s_total_amount total
#define s_transtatus header.returncode

#if 0
#define NEWORDER_SERVICE "NEWORD"
#define PAYMENT_SERVICE "PAYMENT"
#define DELIVERY_SERVICE "DELIVERY"
#define STOCKLEVEL_SERVICE "STOCKLEV"
#define ORDERSTATUS_SERVICE "ORDSTAT"
#else
#define NEWORDER_SERVICE "neword_sql"
#define PAYMENT_SERVICE "payment_sql"
#define DELIVERY_SERVICE "delivery_sql"
#define STOCKLEVEL_SERVICE "stocklev_sql"
#define ORDERSTATUS_SERVICE "ordstat_sql"
#endif

#endif /* TPCC_H_INCLUDED */

                                tpcc.tacf

/*
* Copyright (C) 1991, 1990Transarc Corporation
* All Rights Reserved
*/
/*
* tpcc.tacf -- attribute configuration file fortgcc server.
* used for transparent binding
*
* $Revision: 1.1 $
* $Date: 1998/11/06 21:10:17 $
* $Log: tpcc.tacf,v $
Revision 4.2 95/05/16 10:55:49 10:55:49tpcc (TPCC Benchmark)
Added necessary RCS ident strings

*/

[implicit_handle(mon_handle_t handle)]
interface tpccTransactions
{
}

                                tpcc_trans.tacf

/*
* Copyright (C) 1991, 1990Transarc Corporation
* All Rights Reserved
*/
/*
* neworder.tacf -- attribute configuration file fortgcc server.
* used for transparent binding
*
* $Revision: 1.1 $
* $Date: 1998/11/06 21:10:17 $
* $Log: tpcc.tacf,v $
*
* $TALog:tpcc_trans.tacf,v $
* Revision 1.1 1998/11/06 21:10:17dongfeng
* - Move all files common to client and server totgcc/common
* directory
* [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
*
* Revision 1.1 1997/06/16 22:04:48 oz
* - Integration with Data Dependent Routing: Phase 1
* Separated the all the binding related code into its own files.
* - Added mon_client_utils.[ch] that export binding related calls.
* - Added a TPCC_USE_DDR compile time switch
* - Added tpcc_trans.tidl: All the functions in one interface.
* [added by delta oz-20170-TPCC-add-data-dependent-routing, r1.1]
*
*/

[implicit_handle(mon_handle_t handle)]
interface tpccTrans
{
}

                                tpcc_trans.tidl

/*
* id:Sid:$
*
* component_name: encina benchmarks

```

```

*
* the following functions list may not be complete.
* functions defined by/via macros may not be included.
*
* functions:
* <fill_me_in>
*
* origins: transarc corp.
*
* (c) copyright transarc corp. 1995, 1993
* all rights reserved
* licensed materials - property of transarc
*
* us government users restricted rights - use, duplication or
* disclosure restricted by gsa adp schedule contract with transarc corp
*/
/*
* history
* Stalog: $
*/
/*
* tpcc_trans.tidl -- interface definition file fortppcserver.
*
* $Revision: 1.11 $
* $Date: 1995/10/20 21:55:05 $
* $Log: tpcc.tidl,v $
*/

[uuid(955d7288-e672-11d0-bcef-9e621234aa77)version(1.0)]

interface tpccTrans
{
import "tpm/mon/mon_handle.idl";
import "tpcc_type.idl";

[nontransactional] void
    impTPCCNewOrder([in,out] newOrder_data_t *dataP,
                    [out] trpc_status_t *trpcStatus);

[nontransactional] void
    impTPCCPayment([in,out] payment_data_t *dataP,
                  [out] trpc_status_t *trpcStatus);

[nontransactional] void
    impTPCCOrderStatus([in,out] orderStatus_data_t *dataP,
                       [out] trpc_status_t *trpcStatus);

[nontransactional] void
    impTPCCStockLevel([in,out] stockLevel_data_t *dataP,
                      [out] trpc_status_t *trpcStatus);

[nontransactional] void
    impTPCCNOInfo([out] dbInfo_data_t *dataP,
                  [out]trpc_status_t *trpcStatus);
}

                tpcc_type.idl

/*
*
* tpcc_type.idl
*
* $Revision: 1.2 $
* $Date: 1998/12/08 18:55:21 $
* $Log: $
*
*
* $TALog: tpcc_type.idl,v $
* Revision 1.2 1998/12/08 18:55:21 wenjian
* Add "int stats" to data_header structure
* [from r1.1 by delta wenjian-23785-TPCC-pass-statsFrequency-from-client-to-server,r1.1]
*
* 2001/11/25 klavs: Changed warehouses references to long int
*
* Revision 1.1 1998/11/06 21:10:17 dongfeng
* - Move all files common to client and server to tpcc/common
* directory
* [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
*
* Revision 1.11 1998/01/24 14:17:07 oz
* - User server name to identify server and name delivery file
* - Use env variable HOME instead of/home/encina if HOME is set
*
* - Added const ONLINE_INTERFACES
* [from r1.10 by delta oz-21687-TPCC-use-server-name-to-identify-process,r1.1]
*
* Revision 1.10 1998/01/23 15:09:11 oz
* - Updated the SP TPCC directory to the latest files used
* during the SP tpcc audit.
* [from r1.9 by delta oz-20774-TPCC-update-to-latest-SP-version-11-27,r1.1]
*
*
*/
[

uid(008c6338-2b0a-1001-a9ab-02608c2f015a)version(1)
]
interface tpcc_types {

const long NAME_LENGTH = 32;

const long NEWO_INTERFACE = 0x01;
const long PAYMENT_INTERFACE= 0x02;
const long ORDER_STAT_INTERFACE= 0x04;
const long DELIVERY_INTERFACE= 0x08;
const long STOCK_INTERFACE = 0x10;
const long ONLINE_INTERFACES = NEWO_INTERFACE | PAYMENT_INTERFACE|
ORDER_STAT_INTERFACE | STOCK_INTERFACE;
const long ALL_INTERFACE = 0xffff;

const long NEWO_TRANS = 1;
const long PAYMENT_TRANS = 2;
const long ORDER_STAT_TRANS = 3;
const long DELIVERY_TRANS = 4;
const long STOCK_TRANS = 5;
const long MAX_TRAN_TYPE = 5;

typedef struct {
        long int sec;
        long int usec;
} time_type;

typedef struct {
short int dtype;
short int returncode;
long int sql_code;
long int isam_code;
long int num_rms;

short int stats; /* For instrument only */
time_type start_time; /* For Debug Purposes only */
time_type end_time; /* For Debug Purposes only */
} data_header;

/* Definitions for payment transaction
*
* payment_data_t
*
* An in-out structure for payment transaction.
* It contains all the input parameters as well as the output parameters.
*/
typedef struct {
data_header header;
long int w_id;
short int d_id;
short int c_id;
long int c_w_id;
short int c_d_id;
short int byname;
double h_amount;
char pay_date[20];

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

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

char c_first[17]; /* was C_LAST_LEN already includes +1 */
char c_middle[3];
char c_last[17];
char c_phone[17];
char c_credit[3];
char c_street_1[21];
char c_street_2[21];
char c_city[21];
char c_state[3];
char c_zip[10];
double c_credit_lim;
double c_balance;
double c_discount;
double c_ytd_payment;
short int c_payment_cnt;
char c_date[20];
char c_data[201];
} payment_data_t;

/* Definitions for new order transaction */

typedef struct {
long int ol_supply_w_id;
short int ol_quantity;
short int s_quantity;
long int ol_i_id;
char name_i[25];
char brand_generic[2];
}

```

```

double price;
double ol_amount;
long int s_idx;
char s_dist[25];
} OL_TABLE,newOrder_item_t;

typedef struct {
    data_header header;
    long int w_id;
    short int d_id;
    short int c_id;
    short int o_ol_cnt;
    short int o_all_local;
    short int items_valid; /* true if all valid*/
    short int total_items;
    long int o_id;
    double w_tax;
    double d_tax;
    double total;
    double c_discount;
    char entry_date[20];
    char c_last[17];
    char c_credit[3];
    char statusline[26];
    OL_TABLE item[15];
} newOrder_data_t;

/* Definitions for order status transaction */

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

typedef struct {
    data_header header;
    long int w_id;
    short int d_id;
    short int c_id;
    short int o_id;
    short int o_ol_cnt;
    short int byname;
    short o_carrier_id;
    char c_last[17];
    char c_first[17];
    char c_middle[3];
    char entry_date[20];
    double c_balance;
    orderStatusItem_t item[15];
} orderStatus_data_t;

/* Definitions for stock level transaction */

typedef struct {
    data_header header;
    long int w_id;
    short int d_id;
    short int threshold;
    long int stock_count;
} stockLevel_data_t;

/* Definitions for delivery transaction */

typedef struct {
    data_header header;
    long int w_id;
    short int o_carrier_id;
    long int queued_time;
    short status;
    char exec_status[50];
    double start_queue;
} delivery_data_t;

typedef struct {
    long int first_wh;
    long int last_wh;
    long int server_id;
} dbInfo_data_t;

/*
 * A union of all the transactions
 */
typedef union switch(long int tran_type) data {
    case NEWO_TRANS: newOrder_data_t new_order;
    case PAYMENT_TRANS: payment_data_t payment;
    case ORDER_STAT_TRANS: orderStatus_data_t order_status;
    case DELIVERY_TRANS: delivery_data_t delivery;
    case STOCK_TRANS: stockLevel_data_t stock_level;
} tpc_data_t;
}

}

tpcc_utils.c
*/
*/
*/
tpcc_utils.c
*/
*/
*/
$Revision: 1.2 $
$Date: 1998/12/14 20:27:57 $
$Log: $
*/
*/
*/
$TALog: tpcc_utils.c,v $
$Revision: 1.2 1998/12/14 20:27:57wenjian
$ Made corresponding changes due to data structure change of tran_info_t.
*/
*/
- Add header file winsock.h for NT platform
[from r1.1 by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.3]
*/
*/
Revision 1.1 1998/12/11 16:37:58wenjian
Move some common functions from client/client_utils.c to common/tpcc_utils.c.
In this version, we only move time_diff_ms(). Need some work in order to
move other functions like ERROUT.
*/
*/
- A file including utility functions for both client and server
[added by delta wenjian-23788-TPCC-use-single-stats-var-for-each-client-and-server, r1.2]
*/
*/
*/
tpcc_utils.c
Generic utilities used by the client and server processes
*/
*/
#include <stdio.h>
#include <time.h>
#include <string.h>
#include <stdarg.h>

#ifdef (solaris)
#include <dce/pthread.h>
#else /* solaris */
#include <pthread.h>
#endif

#include "databuf.h"
#include "do_tpcc.h"
#include "tpcc_type.h"

#ifdef WIN32
#include <winsock.h>
#endif

/*
 * time_diff_ms
 * Return the difference in milliseconds between two times
 */
int time_diff_ms(t2, t1)
struct timeval *t2, *t1;
{
    int t_diff;

    t_diff = (t2->tv_usec + 1000000 - t1->tv_usec + 500) / 1000 +
        (t2->tv_sec - t1->tv_sec - 1) * 1000;

    return(t_diff);
}

util.h

#ifdef LOCAL_UTIL_H
#define LOCAL_UTIL_H

#include "util_token.h"

#define UTIL_ALLOC(ptr, type, size) \
    ptr = (type)malloc(size); \
    if (ptr==NULL) { \
        fprintf(stderr, "UTIL_ALLOC failed\n"); \
        exit(1); \
    }

#endif

util_alloc.h

/*
 * util_alloc.h
 */
$Revision: 1.1 $
$Date: 1998/11/06 21:10:18 $
$Log: util_alloc.h,v $
$Revision: 4.2 95/05/16 10:55:43tpcc (TPCC Benchmark)
Added necessary RCS ident strings
*/
*/
*/

```

```

#endif TRANSARC_UTIL_ALLOC_H
#define TRANSARC_UTIL_ALLOC_H

/*
 * UTIL_[ALLOC,REALLOC,NEW,FREE] -- macros that wrap calls to
 * malloc, realloc, free. The allocation macros check the return
 * value, a NULL pointer is converted into a fatal error.
 */
#define UTIL_ALLOC_ROBUST(ptr,type, size) \
((ptr) = (type) malloc(size))

#define UTIL_ALLOC(ptr, type, size) \
do { \
    if(UTIL_ALLOC_ROBUST(ptr,type, size) == 0) \
        util_MemoryError("UTIL_ALLOC",__FILE__, __LINE__); \
} while (0)

#define UTIL_REALLOC_ROBUST(ptr,type, size) \
(ptr = (type) realloc((void *) ptr, size))

#define UTIL_REALLOC(ptr, type, size) \
do { \
    if(UTIL_REALLOC_ROBUST(ptr,type, size) == 0) \
        util_MemoryError("UTIL_REALLOC",__FILE__, __LINE__); \
} while (0)

#define UTIL_FREE(ptr) \
do { \
    if(!ptr) { \
        util_MemoryError("UTIL_FREE",__FILE__, __LINE__); \
    } \
    free((void *) (ptr)); \
    ptr = 0; /* Make all free'd pointers zero. */ \
} while (0)

#define UTIL_ALLOC_ARRAY_ROBUST(ptr,type, number) \
((ptr) = (type *)malloc(sizeof(type) * (number)))

#define UTIL_ALLOC_ARRAY(ptr, type, number) \
do { \
    if(UTIL_ALLOC_ARRAY_ROBUST(ptr,type, number) == 0) \
        util_MemoryError("UTIL_ALLOC_ARRAY",__FILE__, __LINE__); \
} while (0)

#define UTIL_COPY_STRING_ROBUST(to,from) \
(((to) = (char *)malloc(strlen(char *(from))+1)) ? \
strcpy(char *(to), (char *(from)) : 0)

#define UTIL_COPY_STRING(to,from) \
do { \
    if(UTIL_COPY_STRING_ROBUST(to,from) == 0) \
        util_MemoryError("UTIL_COPY_STRING",__FILE__, __LINE__); \
} while (0)

#endif /* TRANSARC_UTIL_ALLOC_H */

                util token.h

/*
 * ID: $Id: util_token.h,v 1.1 1998/11/06 21:10:18dongfeng Exp $
 *
 * COMPONENT_NAME: Encina Toolkit Executive
 *
 * The following functions list may not be complete.
 * Functions defined by/via macros may not be included.
 *
 * FUNCTIONS:
 *
 * ORIGINS: Transarc Corp.
 *
 * (C) COPYRIGHT Transarc Corp. 1995, 1994, 1993, 1990
 * All Rights Reserved
 * Licensed Materials - Property of Transarc
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with Transarc Corp
 */
/*
 * HISTORY
 * $TALog: util_token.h,v $
 * Revision 1.1 1998/11/06 21:10:18dongfeng
 * - Move all files common to client and server totpcc/common
 * directory
 * [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
 *
 * Revision 1.7 1995/01/30 12:50:13barry
 * Update copyrights for Encina 1.2.
 * [from r1.6 by delta barry-0000-update-copyrights-for-1.2,r1.1]
 */
                *
                * Revision 1.6 1995/01/13 14:12:51psu
                * fix comment to conform to coding standards.
                * [from r1.5 by delta psu-13196-client-interoperate-with-oracle-pro-c-2,r1.3]
                *
                * Revision 1.5 1995/01/12 20:34:22psu
                * put comment on ## fix in the code
                *
                * try to make sure people don't change the use of ##
                * [from r1.4 by delta psu-13196-client-interoperate-with-oracle-pro-c-2,r1.2]
                *
                * Revision 1.4 1995/01/12 15:48:28psu
                * fix use of ## to make pro*c happy
                *
                * pro*c 2.0 can't deal with a ## b ## c, change to a ## b##c.
                * [from r1.3 by delta psu-13196-client-interoperate-with-oracle-pro-c-2,r1.1]
                *
                * Revision 1.3 1994/02/04 17:22:22pinaki
                * Update copyright.
                * [from r1.2 by delta pinaki-0000-update-copyright-for-1.1,r1.1]
                *
                * Revision 1.2 1993/12/18 22:06:57mwyong
                * [from r1.1 by delta mwyong-10043-util-always-offer-UTIL_IDENT,r1.1]
                *
                * Revision 1.1 1993/12/03 22:00:04mwyong
                * Split the various features into separate files, so that they can
                * be included separately.
                * [added by delta mwyong-9848-utils-split-util.h-into-separately-usable-parts,r1.1]
                *
                */
                #ifndef TRANSARC_UTIL_TOKEN_H
                #define TRANSARC_UTIL_TOKEN_H

                #include <encina/c_prologue.h>

                /* UTIL_IDENT-- the identity function */
                #define UTIL_IDENT(a)          a

                /*
                * UTIL_[STRING,CONCAT,CONCAT3] -- macros for converting into, and
                * concatenating together, strings.
                */

                /* Note, the a ## b##c is needed to make some broken cpp's work correctly.
                * This was originally put here for OraclePro*C 2.0, but other compilers
                * may have similar problems.
                */

                #if          ENCINA_C_ANSI_STRING_TOKEN_SUPPORT
                #define UTIL_STRING(a)  # a
                #define UTIL_CONCAT(a,b)          a ## b
                #define UTIL_CONCAT3(a,b,c)  a ## b##c
                #else          /* ENCINA_C_ANSI_STRING_TOKEN_SUPPORT */
                #define UTIL_STRING(a)          "a"
                #define UTIL_CONCAT(a,b)          UTIL_IDENT(a)b
                #define UTIL_CONCAT3(a,b,c)      UTIL_CONCAT(a,b)c
                #endif          /* ENCINA_C_ANSI_STRING_TOKEN_SUPPORT */

                #include <encina/c_epilogue.h>
                #endif /* TRANSARC_UTIL_TOKEN_H */

                utilities.h

                /*
                * ID: $Id: utilities.h,v 1.1 1998/11/06 21:10:19dongfeng Exp $
                *
                * COMPONENT_NAME: Encina Toolkit Server Core
                *
                * ORIGINS: Transarc Corp.
                *
                * (C) COPYRIGHT Transarc Corp. 1995
                * All Rights Reserved
                * Licensed Materials - Property of Transarc
                *
                * US Government Users Restricted Rights - Use, duplication or
                * disclosure restricted by GSA ADP Schedule Contract with Transarc Corp
                * $Revision: 1.1 $
                * $Log: utilities.h,v $
                *
                * $TALog: utilities.h,v $
                * Revision 1.1 1998/11/06 21:10:19dongfeng
                * - Move all files common to client and server totpcc/common
                * directory
                * [added by delta dongfeng-23677-TPCC-new-directory-structures,r1.1]
                *
                * Revision 1.7 1998/10/27 14:57:52dongfeng
                * Change enc_status to a data structure that has fields:
                * - Status code
                * - Line Number
                * - File Name
                * - Encina Error Code
                * - Error Msg
                * Remove statusMsgs in web_tpsc.c
                * [from r1.6 by delta dongfeng-23067-TPCC-add-web-based-tpcc-client,r1.6]
                *
                * Revision 1.6 1998/10/22 19:18:37dongfeng
                */

```



```

cust_rowid      ROWID;
dist_name       VARCHAR2(11);
ware_name       VARCHAR2(11);
c_num           BINARY_INTEGER;
PROCEDURE pay_init;
END initpay;
/
CREATE OR REPLACE PACKAGE BODY initpay AS
PROCEDURE pay_init IS
BEGIN
NULL;
END pay_init;
END initpay;
/
exit

                                payz.sql

DECLARE /* payz */
not_serializable EXCEPTION;
PRAGMA EXCEPTION_INIT(not_serializable,-8177);
deadlock          EXCEPTION;
PRAGMA EXCEPTION_INIT(deadlock,-60);
snapshot_too_old  EXCEPTION;
PRAGMA EXCEPTION_INIT(snapshot_too_old,-1555);
BEGIN
LOOP BEGIN
UPDATE 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 initpay.ware_name,
:w_street_1, :w_street_2, :w_city, :w_state, :w_zip;

--Bulk fetch
SELECT rowid
BULK COLLECT INTO initpay.row_id
FROM customer
WHERE c_d_id = :c_d_id AND c_w_id = :c_w_id AND c_last = :c_last
ORDER BY c_last, c_d_id, c_w_id, c_first;

--Store number of rows processed
initpay.c_num := sql%rowcount;
initpay.cust_rowid := initpay.row_id((initpay.c_num) / 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 = initpay.cust_rowid
RETURNING
c_id, c_first, c_middle, c_last, c_street_1, c_street_2,
c_city, c_state, c_zip, c_phone,
c_since, c_credit, c_credit_lim,
c_discount, c_balance
INTO :c_id, :c_first, :c_middle, :c_last,
:c_street_1, :c_street_2, :c_city, :c_state,
:c_zip, :c_phone, :c_since, :c_credit,
:c_credit_lim, :c_discount, :c_balance;

:c_data := '';
IF :c_credit = 'BC' THEN
UPDATE 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') || '' )
|k_data, 1, 500)
WHERE rowid = initpay.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 initpay.dist_name, :d_street_1, :d_street_2, :d_city,
:d_state, :d_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,
:c_date, initpay.ware_name || ' ' || initpay.dist_name);

--Sanjay-No commit needed iff Commit on Success done
-- COMMIT;
EXIT;

EXCEPTION

```

```

WHEN not_serializable OR deadlock OR snapshot_too_old THEN
ROLLBACK;
:retry := :retry + 1;
END;

END LOOP;
END;

```

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 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 initpay.ware_name, :w_street_1, :w_street_2, :w_city,
:w_state, :w_zip;

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 initpay.cust_rowid, :c_first, :c_middle, :c_last, :c_street_1,
:c_street_2, :c_city, :c_state, :c_zip, :c_phone,
:c_since, :c_credit, :c_credit_lim,
:c_discount, :c_balance;
IF SQL%NOTFOUND THEN
raise NO_DATA_FOUND;
END IF;

-- :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') || '' )
|k_data, 1, 500)
WHERE rowid = initpay.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 initpay.dist_name, :d_street_1, :d_street_2, :d_city, :d_state,
:d_zip;
IF SQL%NOTFOUND THEN
raise NO_DATA_FOUND;
END IF;

INSERT INTO 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,
:c_date, initpay.ware_name || ' ' || initpay.dist_name);
-- COMMIT;
-- :h_date := to_char (:c_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;

```

pldel.c

```

#ifdef RCSID
static char *RCSid =
  "$Header: /afs/transarc.com/project/encina/rcs/test/src/benchmarks/tpcc/server/ora8.1_nt/RCS/pldel.c,v
1.2 1999/04/15 12:16:51 oExp S Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*
=====+
|   Copyright (c) 1996 OracleCorp, 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 "plora.h"
#ifdef TUX
#include <userlog.h>
#endif

#include "tpccflags.h"

#if defined(ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
#define SQLTXT0 "SELECT substr(value,1,5) FROM v$parameter \
  WHERE name = 'instance_number'"
#endif

#ifdef PLSQDEL
#define SQLTXT "BEGIN delivery.deliver(:w_id, :carrier_id, :order_id, \
:retry); END;"
#else
#ifdef DMLRETDEL
#define SQLTXT1 "DELETE FROM new_order WHERE no_d_id = :d_id \
  AND no_w_id = :w_id AND rownum <= 1 \
  RETURNING no_o_id INTO :o_id"
#else
#define SQLTXT1A "\
SELECT /*+ USE_NL(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 \
"

#define SQLTXT1B "\
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 \
"

#define SQLTXT1C "\
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 \
"

#define SQLTXT1D "\
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 \
"

#define SQLTXT1E "\
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 \
"

#define SQLTXT1F "\
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 \
"

#define SQLTXT1G "\
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 \
"

#define SQLTXT1H "\
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 \
"

#define SQLTXT2 "\
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 \
"

#define SQLTXT3 "\
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 SQLTXT2 "DELETE FROM new_order WHERE rowid = :no_rowid"
#endif

#ifdef DMLRETDEL
#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 \
  RETURNING o_c_id INTO :o_c_id"
#else
#define SQLTXT3 "UPDATE orders SET o_carrier_id = :carrier_id \
  WHERE rowid = :o_rowid"
#endif

#ifdef DMLRETDEL
#define SQLTXT4 "UPDATE /*+ buffer */ 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 \
  RETURNING ol_amount INTO :ol_amount"
#else
#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 SQLTXT5A "\
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 \
"

#define SQLTXT5B "\
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 \
"

#define SQLTXT5C "\
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 \
"

#define SQLTXT5D "\
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 \
"

#define SQLTXT5E "\
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"

#endif
#endif /* PLSQDEL */

#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
#endif

```

```

#ifdef DMLRETDEL
ub4 del_o_id_len[NDISTS];
ub4 c_id_len[NDISTS];
int oid_ctx;
int cid_ctx;
OCIBind *olamt_bp;
#else
ub2 del_o_id_len[NDISTS];
ub2 c_id_len[NDISTS];
#endif

ub2 cons_len[NDISTS];
ub2 w_id_len[NDISTS];
ub2 d_id_len[NDISTS];
ub2 del_date_len[NDISTS];
ub2 carrier_id_len[NDISTS];
ub2 amt_len[NDISTS];
ub2 no_rowid_len[NDISTS];
ub2 no_rowid_ptr_len[NDISTS];
ub2 o_rowid_len[NDISTS];
ub2 o_rowid_ptr_len[NDISTS];
#ifdef (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];
#ifdef (ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
ub2 inum_rcode;
#endif

int del_o_id[NDISTS];
int cons[NDISTS];
int w_id[NDISTS];
int d_id[NDISTS];
int c_id[NDISTS];
int carrier_id[NDISTS];
int amt[NDISTS];
ub4 del_o_id_rcnt;
int retry;
OCIRowid *no_rowid_ptr[NDISTS];
OCIRowid *o_rowid_ptr[NDISTS];
OCIDate del_date[NDISTS];
#ifdef (ISO) || defined(ISO5) || defined(ISO6) || defined(ISO8)
char inum[10];
#endif

OCISmt *curd0;
OCISmt *curd1;
OCISmt *curd2;
OCISmt *curd3;
OCISmt *curd4;
OCISmt *curd5;
OCISmt *curd6;
OCISmt *curdtest;

OCIBind *w_id_bp;
OCIBind *w_id_bp3;
OCIBind *w_id_bp4;
OCIBind *w_id_bp5;
OCIBind *w_id_bp6;
OCIBind *d_id_bp;
OCIBind *d_id_bp3;
OCIBind *d_id_bp4;
OCIBind *d_id_bp6;
OCIBind *o_id_bp;
OCIBind *cr_date_bp;
OCIBind *c_id_bp;
OCIBind *c_id_bp3;
OCIBind *no_rowid_bp;
OCIBind *carrier_id_bp;
OCIBind *o_rowid_bp;
OCIBind *del_o_id_bp;
OCIBind *del_o_id_bp3;
OCIBind *amt_bp;
OCIBind *bstr1_bp[10];
OCIBind *bstr2_bp[10];
OCIBind *retry_bp;
OCIDefine *inum_dp;
OCIDefine *d_id_dp;
OCIDefine *del_o_id_dp;
OCIDefine *no_rowid_dp;
OCIDefine *c_id_dp;
OCIDefine *o_rowid_dp;
OCIDefine *cons_dp;
OCIDefine *amt_dp;

int norow;
};

typedef struct delctx delctx;

/* delctx *dctx; */

#ifdef DMLRETDEL
struct amtctx {
int o_amt[NDISTS][NITEMS];
sb2 ol_amt_ind[NDISTS][NITEMS];
ub4 ol_amt_len[NDISTS][NITEMS];
ub2 ol_amt_rcode[NDISTS][NITEMS];
int ol_cnt[NDISTS];
};
typedef struct amtctx amtctx;
/* amtctx *actx; */

extern sb4 no_data();

sb4 TPC_oid_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
dvoid **bufpp, ub4 **alenp, ub1 *piecep,
dvoid **indpp, ub2 **rcodepp)
{
delctx *dctx = (delctx *)ctxp;
*bufpp = &dctx->del_o_id[iter];
*indpp = &dctx->del_o_id_ind[iter];
dctx->del_o_id_len[iter]=sizeof(dctx->del_o_id[0]);
*alenp = &dctx->del_o_id_len[iter];
*rcodepp = &dctx->del_o_id_rcode[iter];
*piecep = OCI_ONE_PIECE;
return (OCI_CONTINUE);
}

sb4 cid_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
dvoid **bufpp, ub4 **alenp, ub1 *piecep,
dvoid **indpp, ub2 **rcodepp)
{
delctx *dctx = (delctx *)ctxp;
*bufpp = &dctx->c_id[iter];
*indpp = &dctx->c_id_ind[iter];
dctx->c_id_len[iter]=sizeof(dctx->c_id[0]);
*alenp = &dctx->c_id_len[iter];
*rcodepp = &dctx->c_id_rcode[iter];
*piecep = OCI_ONE_PIECE;
return (OCI_CONTINUE);
}

sb4 amt_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
dvoid **bufpp, ub4 **alenp, ub1 *piecep,
dvoid **indpp, ub2 **rcodepp)
{
amtctx *actx;
actx =(amtctx *)ctxp;
actx->ol_cnt[iter]=actx->ol_cnt[iter]+1;
*bufpp = &actx->ol_amt[iter][index];
*indpp = &actx->ol_amt_ind[iter][index];
actx->ol_amt_len[iter][index]=sizeof(actx->ol_amt[0][0]);
*alenp = &actx->ol_amt_len[iter][index];
*rcodepp = &actx->ol_amt_rcode[iter][index];
*piecep = OCI_ONE_PIECE;
return (OCI_CONTINUE);
}

#endif

tkvcinit(ora_cn_data_t *ora_SlotDataP)
{
int i,j;
char bstr1[10];
char bstr2[10];
text stmbuf[SQL_BUF_SIZE];

delctx *dctx;
amtctx *actx;
global_delivery_t *delP;
OCISvc *tpcenv = ora_SlotDataP->tpcenv;
OCIServer *tpcsrv = ora_SlotDataP->tpcsrv;
OCISession *tpcusr = ora_SlotDataP->tpcusr;
OCISmt *curi = ora_SlotDataP->curi;

dctx = (delctx *) malloc (sizeof(delctx));
memset(dctx,(char)0,sizeof(delctx));
dctx->norow = 0;

ora_SlotDataP->dctx = (void *)dctx;
delP = (global_delivery_t *)malloc(sizeof(global_delivery_t));
memset(delP, (char)0, sizeof(global_delivery_t));
ora_SlotDataP->delP = delP;

#ifdef DMLRETDEL
actx = (amtctx *) malloc (sizeof(amtctx));
memset(actx,(char)0,sizeof(amtctx));
ora_SlotDataP->actx = (void *)actx;
#else
#endif

```


<pre> for(i=0;i<NDISTS;i++){ OCIERROR(errhp, OCIDescriptorAlloc(tpcenv,(dvoid*)&dctx->o_rowid_ptr[i], OCI_DTYPE_ROWID,0,(dvoid**0)); OCIERROR(errhp, OCIDescriptorAlloc(tpcenv,(dvoid*)&dctx->no_rowid_ptr[i], OCI_DTYPE_ROWID,0,(dvoid**0)); } #endif #if defined(ISO) defined(ISO5) defined(ISO6) defined(ISO8) OCIHandleAlloc(tpcenv,(dvoid **>(&dctx->curd0), OCI_HTYPE_STMT,0,(dvoid**0); sprintf((char *) stmbuf, SQLTXTO); OCIStmtPrepare(dctx->curd0, errhp, stmbuf, strlen((char *)stmbuf),OCI_NTV_SYNTAX, OCI_DEFAULT); OCIDFNRA(dctx->curd0, dctx->inum_dp,errhp,1,dctx->inum,SIZ(dctx->inum),SQLT_STR, &(dctx->inum_ind),&(dctx->inum_len),&(dctx->inum_rcode)); #endif /* If PLSQDEL and ISO? are both defined, then they both try to use curd0! This could cause a problem. Will try to fix later - VMM 12/30/97*/ #ifdef PLSQDEL OCIHandleAlloc(tpcenv,(dvoid **>(&dctx->curd0), OCI_HTYPE_STMT, 0,(dvoid**0)); sprintf((char *) stmbuf, SQLTXT); OCIStmtPrepare(dctx->curd0, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX,OCI_DEFAULT); OCIBND(dctx->curd0, dctx->w_id_bp , errhp,":w_id",ADR(delP->w_id),SIZ(int), SQLT_INT); OCIBND(dctx->curd0, dctx->carrier_id_bp , errhp,":carrier_id", ADR(dctx->carrier_id),SIZ(int),SQLT_INT); OCIBNDRAA(dctx->curd0, dctx->o_id_bp, errhp, "order_id", dctx->del_o_id,SIZ(int),SQLT_INT,dctx->del_o_id_ind, dctx->del_o_id_len,dctx->del_o_id_rcode,NDISTS, &dctx->del_o_id_rcnt); OCIBND(dctx->curd0, dctx->retry_bp , errhp,":retry",ADR(dctx->retry), SIZ(int),SQLT_INT); #else #ifdef DMLRETDDEL OCIHandleAlloc(tpcenv,(dvoid **>(&dctx->curd1), OCI_HTYPE_STMT,0,(dvoid**0); sprintf((char *) stmbuf, "%s", SQLTXT1); OCIStmtPrepare(dctx->curd1, errhp, stmbuf, strlen((char *)stmbuf),OCI_NTV_SYNTAX, OCI_DEFAULT); OCIBND(dctx->curd1, dctx->w_id_bp,errhp,":w_id",dctx->w_id,SIZ(int), SQLT_INT); OCIBNDRA(dctx->curd1, dctx->d_id_bp,errhp,":d_id",dctx->d_id,SIZ(int), SQLT_INT,NULL,NULL,NULL); OCIBNDRAD(dctx->curd1, dctx->del_o_id_bp, errhp, ":o_id", SIZ(int),SQLT_INT,NULL, (dvoid *)dctx->no_data,TPC_oid_data); #else OCIHandleAlloc(tpcenv,(dvoid **>(&dctx->curd1), OCI_HTYPE_STMT,0,(dvoid**0); sprintf((char *) stmbuf, "%s%s%s%s%s%s%s%s", SQLTXT1A, SQLTXT1B, SQLTXT1C, SQLTXT1D, SQLTXT1E, SQLTXT1F, SQLTXT1G, SQLTXT1H, SQLTXT1I, SQLTXT1J); OCIStmtPrepare(dctx->curd1, errhp, stmbuf, strlen((char *)stmbuf),OCI_NTV_SYNTAX, OCI_DEFAULT); OCIERROR(errhp, OCIAttrSet(dctx->curd1,OCI_HTYPE_STMT,(dvoid*)&dctx->norow,0, OCI_ATTR_PREFETCH_ROWS,errhp)); /* bind variables */ OCIBND(dctx->curd1, dctx->w_id_bp,errhp,":w_id",ADR(delP->w_id),SIZ(int),SQLT_INT); OCIDFNRA(dctx->curd1, dctx->d_id_dp,errhp,1,dctx->d_id,SIZ(int), SQLT_INT,dctx->d_id_ind,dctx->d_id_len,dctx->d_id_rcode); OCIDFNRA(dctx->curd1, dctx->del_o_id_dp,errhp,2,dctx->del_o_id, SIZ(int),SQLT_INT,dctx->del_o_id_ind, dctx->del_o_id_len, dctx->del_o_id_rcode); OCIDFNRA(dctx->curd1, dctx->no_rowid_dp,errhp,3,dctx->no_rowid_ptr, SIZ(OCIROWID *),SQLT_RDD,dctx->no_rowid_ind, dctx->no_rowid_len, dctx->no_rowid_rcode); OCIDFNRA(dctx->curd1, dctx->c_id_dp,errhp,4,dctx->c_id,SIZ(dctx->c_id[0]), SQLT_INT,dctx->c_id_ind,dctx->c_id_len,dctx->c_id_rcode); </pre>	<pre> OCIDFNRA(dctx->curd1, dctx->o_rowid_dp,errhp,5,dctx->o_rowid_ptr, SIZ(OCIROWID *),SQLT_RDD,dctx->o_rowid_ind, dctx->o_rowid_len, dctx->o_rowid_rcode); /* open second cursor */ OCIHandleAlloc(tpcenv,(dvoid **>(&dctx->curd2), OCI_HTYPE_STMT,0,(dvoid**0); sprintf((char *) 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); #endif /*DMLRETDDEL*/ /* open third cursor */ OCIHandleAlloc(tpcenv,(dvoid **>(&dctx->curd3), OCI_HTYPE_STMT,0,(dvoid**0); sprintf((char *) stmbuf, SQLTXT3); OCIStmtPrepare(dctx->curd3, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX,OCI_DEFAULT); /* bind variables */ OCIBNDRA(dctx->curd3, dctx->carrier_id_bp,errhp,":carrier_id",dctx->carrier_id, SIZ(dctx->carrier_id[0]),SQLT_INT,dctx->carrier_id_ind, dctx->carrier_id_len,dctx->carrier_id_rcode); #ifdef DMLRETDDEL OCIBNDRA(dctx->curd3, dctx->w_id_bp3, errhp, ":w_id", dctx->w_id,SIZ(int), SQLT_INT,NULL,NULL,NULL); OCIBNDRA(dctx->curd3, dctx->d_id_bp3, errhp, ":d_id", dctx->d_id,SIZ(int), SQLT_INT,NULL,NULL,NULL); OCIBNDRA(dctx->curd3, dctx->del_o_id_bp3, errhp, ":o_id", dctx->del_o_id, SIZ(int),SQLT_INT,NULL,NULL,NULL); OCIBNDRAD(dctx->curd3, dctx->c_id_bp3, errhp, ":o_c_id", SIZ(int), SQLT_INT,NULL,(dvoid *)dctx->no_data, cid_data); #else OCIBNDRA(dctx->curd3, dctx->o_rowid_bp,errhp,":o_rowid",&(dctx->o_rowid_ptr[0]), SIZ(dctx->o_rowid_ptr[0]),SQLT_RDD,dctx->o_rowid_ind, dctx->o_rowid_ptr_len,dctx->o_rowid_rcode); #endif /* open fourth cursor */ OCIHandleAlloc(tpcenv,(dvoid **>(&dctx->curd4), OCI_HTYPE_STMT,0,(dvoid**0); sprintf((char *) stmbuf, SQLTXT4); OCIStmtPrepare(dctx->curd4, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX,OCI_DEFAULT); /* bind variables */ OCIBND(dctx->curd4, dctx->w_id_bp4,errhp,":w_id",dctx->w_id, SIZ(int),SQLT_INT); OCIBND(dctx->curd4, dctx->d_id_bp4,errhp,":d_id",dctx->d_id, SIZ(int),SQLT_INT); OCIBND(dctx->curd4, dctx->o_id_bp,errhp,":o_id",dctx->del_o_id, SIZ(int),SQLT_INT); OCIBND(dctx->curd4, dctx->cr_date_bp,errhp,":cr_date", dctx->del_date, SIZ(OCIDATE),SQLT_ODT); #ifdef DMLRETDDEL OCIBNDRAD(dctx->curd4, dctx->olam_bp, errhp, ":ol_amount", SIZ(int),SQLT_INT,NULL,actx->no_data,amt_data); #else /* open fifth cursor */ OCIHandleAlloc(tpcenv,(dvoid **>(&dctx->curd5), OCI_HTYPE_STMT,0,(dvoid**0); sprintf((char *) stmbuf, "%s%s%s%s%s", SQLTXT5A, SQLTXT5B, SQLTXT5C, SQLTXT5D, SQLTXT5E); OCIStmtPrepare(dctx->curd5, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX,OCI_DEFAULT); OCIERROR(errhp, OCIAttrSet(dctx->curd5,OCI_HTYPE_STMT,(dvoid*)&dctx->norow,0, OCI_ATTR_PREFETCH_ROWS,errhp)); /* bind variables */ OCIBND(dctx->curd5,dctx->w_id_bp,errhp,":w_id",ADR(delP->w_id),SIZ(delP->w_id),SQLT_INT); for (i = 0; i < NDISTS; i++) { sprintf(bstr1,":d_id%d", i + 1); sprintf(bstr2,":o_id%d", i + 1); OCIBNDRA(dctx->curd5,dctx->bstr1_bp[i],errhp,bstr1,ADR(dctx->d_id[i]), SIZ(dctx->d_id[0]),SQLT_INT,&(dctx->d_id_ind[i]), &(dctx->d_id_len[i]),&(dctx->d_id_rcode[i])); OCIBNDRA(dctx->curd5,dctx->bstr2_bp[i],errhp,bstr2,ADR(dctx->del_o_id[i]), SIZ(dctx->del_o_id[0]),SQLT_INT,&(dctx->del_o_id_ind[i]), </pre>
---	--

<pre> &(dctx->del_o_id_len[i]),&(dctx->del_o_id_rcode[i])); } OCIDFNRA(dctx->curd5,dctx->cons_dp,errhp,1,dctx->cons,SIZ(dctx->cons[0]),SQLT_INT, dctx->cons_ind,dctx->cons_len,dctx->cons_rcode); OCIDFNRA(dctx->curd5,dctx->amt_dp,errhp,2,dctx->amt,SIZ(dctx->amt[0]),SQLT_INT, dctx->amt_ind,dctx->amt_len,dctx->amt_rcode); #endif /* open sixth cursor */ OCIHandleAlloc(tpcenv, (dvoid **)&dctx->curd6, OCI_HTYPE_STMT, 0, (dvoid**)0); sprintf ((char *) stmbuf, SQLTXT6); OCIStmtPrepare(dctx->curd6, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT); /* bind variables */ OCIBND(dctx->curd6,dctx->amt_bp,errhp,"amt",dctx->amt,SIZ(int), SQLT_INT); OCIBND(dctx->curd6,dctx->w_id_bp,errhp,"w_id",dctx->w_id,SIZ(int), SQLT_INT); OCIBND(dctx->curd6,dctx->d_id_bp,errhp,"d_id",dctx->d_id,SIZ(int), SQLT_INT); OCIBND(dctx->curd6,dctx->c_id_bp,errhp,"c_id",dctx->c_id,SIZ(int), SQLT_INT); #endif return (0); } void shiftdata(delctx *dctx,int from) { int i; for (i=from;i<NDISTS-1;i++) { dctx->del_o_id_ind[i] = dctx->del_o_id_ind[i+1]; dctx->del_o_id_rcode[i] = dctx->del_o_id_rcode[i+1]; dctx->w_id[i] = dctx->w_id[i+1]; dctx->d_id[i] = dctx->d_id[i+1]; dctx->carrier_id[i] = dctx->carrier_id[i+1]; } } tkvcd (ora_cn_data_t *ora_SlotDataP) { int i,j,v; int rpc,rcount,count; int invalid; int tmp_id; int tmp_amt; delctx *dctx = (delctx *)ora_SlotDataP->dctx; #ifdef DMLRETDDEL /* VMM 1/13/98 */ amtctx *actx = (amtctx *)ora_SlotDataP->actx; #endif /* DMLRETDDEL */ global_delivery_t *delP = ora_SlotDataP->delP; OCIEnc *tpcenv = ora_SlotDataP->tpcenv; OCIServer *tpcsrv = ora_SlotDataP->tpcsrv; OCIErr *errhp = ora_SlotDataP->errhp; OCISvcCtx *tpcsvc = ora_SlotDataP->tpcsvc; OCISession *tpcusr = ora_SlotDataP->tpcusr; OCISmt *curi = ora_SlotDataP->curi; #ifdef ISO defined(ISO5) defined(ISO6) defined(ISO8) int hasno; int reread; char sdate[30]; OCIStmtExecute(tpcsvc,dctx->curd0,errhp,1,0,0,OCI_DEFAULT); sysdate (sdate); printf ("Delivery started at %s on node %s\n", sdate, dctx->inum); #endif #ifdef PLSQLDEL for (i = 0; i < NDISTS; i++) { dctx->del_o_id_ind[i] = TRUE; dctx->del_o_id_len[i] = sizeof(int); } OCIERROR(errhp, OCIStmtExecute(tpcsvc,dctx->curd0,errhp,1,0,0,OCI_DEFAULT)); for (i = 0; i < NDISTS; i++) { delP->del_o_id[i] = 0; if (dctx->del_o_id_ind[i] == 0) { delP->del_o_id[i] = dctx->del_o_id[i]; } } #else retry: </pre>	<pre> #ifdef ISO defined(ISO5) defined(ISO6) defined(ISO8) reread = 1; #endif #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] = delP->w_id; dctx->d_id[i] = i+1; dctx->carrier_id[i] = delP->o_carrier_id; memcpy(&dctx->del_date[i],&delP->cr_date,sizeof(OCIDate)); } #ifdef DMLRETDDEL /* VMM 1/13/98 */ memset(actx,(char)0,sizeof(amtctx)); #endif /* DMLRETDDEL */ /* array select from new_order and orders tables */ delP->execstatus=OCIStmtExecute(tpscvc,dctx->curd1,errhp,NDISTS,0,0,OCI_DEFAULT); if((delP->execstatus != OCI_SUCCESS) && (delP->execstatus != OCI_NO_DATA)) { OCITransRollback(tpscvc,errhp,OCI_DEFAULT); delP->errcode = OCIERROR(errhp,delP->execstatus); if(delP->errcode == NOT_SERIALIZABLE) { delP->retries++; goto retry; } else if(delP->errcode == RECOVER) { delP->retries++; goto retry; } else { return -1; } } /* mark districts with no new order */ OCIAttrGet(dctx->curd1,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,errhp); rpc = rcount; #ifdef DMLRETDDEL /* we have to compress the array here */ if (rcount != NDISTS) { int j = 0; for (i=0;i<NDISTS;i++) { if (dctx->del_o_id_ind[j] == 0) /* there is data here */ j++; else shiftdata(dctx, j); } } #else invalid = NDISTS - rcount; for (i = rpc; i < NDISTS; i++) { dctx->del_o_id_ind[i] = NA; dctx->w_id_ind[i] = NA; dctx->d_id_ind[i] = NA; dctx->c_id_ind[i] = NA; dctx->carrier_id_ind[i] = NA; dctx->no_rowid_ind[i] = NA; dctx->o_rowid_ind[i] = NA; } #endif #ifdef 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; } } } } </pre>
---	--

```

if(!hasno)
    printf ("Delivery[dist %d] found no new order at %s\n", i, sdate);
}
if (reread) {
    sleep (60);
    sysdate (sdate);
    printf ("Delivery wake up at %s\n", sdate);
    reread = 0;
    goto iso;
}
}
#endif

#ifndef DMLRETDEL
/* array delete of new_order table */
delP->execstatus=OCISmtExecute(tpscvc,dctx->curd2,errhp,rpc,0,0,OCI_DEFAULT);
if(delP->execstatus != OCI_SUCCESS) {
    OCITransRollback(tpscvc,errhp,OCI_DEFAULT);
    delP->errcode = OCIErrror(errhp,delP->execstatus);
    if(delP->errcode == NOT_SERIALIZABLE){
        delP->retries++;
        goto retry;
    } else if(delP->errcode == RECOVERR) {
        delP->retries++;
        goto retry;
    } else {
        return -1;
    }
}

/* mark districts with no new order */
OCIAttrGet(dctx->curd2,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,errhp);

if(rcount != rpc) {
#ifndef TUX
    userlog ("Error in TPC-C server %d: %d rows selected, %d rows deleted\n",
        proc_no, rpc, dctx->curd2.rpc);
#else
    fprintf(stderr,
        "Error in TPC-C server %d: %d rows selected, %d rows deleted\n",
        proc_no, rpc, rcount);
#endif /* TUX */
    OCITransRollback(tpscvc,errhp,OCI_DEFAULT);
    return (DEL_ERROR);
}

#ifndef DMLRETDEL */
delP->execstatus=OCISmtExecute(tpscvc,dctx->curd3,errhp,rpc,0,0,OCI_DEFAULT);
if(delP->execstatus != OCI_SUCCESS) {
    OCITransRollback(tpscvc,errhp,OCI_DEFAULT);
    delP->errcode = OCIErrror(errhp,delP->execstatus);
    if(delP->errcode == NOT_SERIALIZABLE){
        delP->retries++;
        goto retry;
    } else if(delP->errcode == RECOVERR) {
        delP->retries++;
        goto retry;
    } else {
        return -1;
    }
}

OCIAttrGet(dctx->curd3,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,errhp);

if(rcount != rpc) {
#ifndef 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(tpscvc,errhp,OCI_DEFAULT);
return (-1);
}

/* array update of order_line table */
delP->execstatus=OCISmtExecute(tpscvc,dctx->curd4,errhp,rpc,0,0,OCI_DEFAULT);
if(delP->execstatus != OCI_SUCCESS) {
    OCITransRollback(tpscvc,errhp,OCI_DEFAULT);
    delP->errcode = OCIErrror(errhp,delP->execstatus);
    if(delP->errcode == NOT_SERIALIZABLE){
        delP->retries++;
        goto retry;
    } else if(delP->errcode == RECOVERR) {
        delP->retries++;
        goto retry;
    } else {
        return -1;
    }
}

#ifndef DMLRETDEL
OCIAttrGet(dctx->curd4,OCI_HTYPE_STMT,&rcount,NULL,OCI_ATTR_ROW_COUNT,errhp);
/* add up amounts */
count=0;
for (i=0;i<rpc;i++)
{
    dctx->amt[i]=0;
    for (j=0;j<dctx->ol_cnt[i];j++)
        if (dctx->ol_amt_rcode[i][j] == 0)
        {
            dctx->amt[i] = dctx->amt[i] + dctx->ol_amt[i][j];
            count = count+1;
        }
    }
if(rcount > rpc*NITEMS) {
    userlog ("Error in TPC-C server %d: %d ordns updated, %d ordl updated\n",
        proc_no, rpc, rcount);
}
#else
/* array select from order_line table */
delP->execstatus=OCISmtExecute(tpscvc,dctx->curd5,errhp,rpc,0,0,OCI_DEFAULT);
if((delP->execstatus != OCI_SUCCESS) && (delP->execstatus != OCI_NO_DATA)) {
    OCITransRollback(tpscvc,errhp,OCI_DEFAULT);
    delP->errcode = OCIErrror(errhp,delP->execstatus);
    if(delP->errcode == NOT_SERIALIZABLE){
        delP->retries++;
        goto retry;
    } else if(delP->errcode == RECOVERR) {
        delP->retries++;
        goto retry;
    } else {
        return -1;
    }
}

OCIAttrGet(dctx->curd5,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,errhp);
if(rcount != rpc) {
#ifndef 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(tpscvc,errhp,OCI_DEFAULT);
return (-1);
}

/* reorder amount selected if necessary */
for (i = 0; i < rpc; i++) {
    if(dctx->cons[i] != dctx->d_id[i]) {
#ifndef 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) {
#ifndef 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(tpscvc,errhp,OCI_DEFAULT);
return (-1);
}

}

#ifndef 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 */
#ifndef defined(ISO5) || defined(ISO6)
    execstatus=OCISmtExecute(tpscvc,dctx->curd6,errhp,rpc,0,0,OCI_DEFAULT);
#else
    delP->execstatus=OCISmtExecute(tpscvc,dctx->curd6,errhp,rpc,0,0,OCI_COMMIT_ON_SUCCESS | OCI_DEFAULT);
#endif

if(delP->execstatus != OCI_SUCCESS) {
    OCITransRollback(tpscvc,errhp,OCI_DEFAULT);
    delP->errcode = OCIErrror(errhp,delP->execstatus);
    if(delP->errcode == NOT_SERIALIZABLE){
        delP->retries++;
        goto retry;
    } else if(delP->errcode == RECOVERR) {
        delP->retries++;
        goto retry;
    }
}

```

```

goto retry;
} else {
return -1;
}
}

OCIAttrGet(dctx->curd6,OCI_HTYPE_STMT,&rcount,0,OCI_ATTR_ROW_COUNT,errhp);

if (rcount != rpc) {
#ifdef TUX
userlog ("Error in TPC-C server %d: %d rows selected, %d cust updated\n",
proc_no, rpc, rcount);
#else
fprintf (stderr,
"Error in TPC-C server %d: %d rows selected, %d cust updated\n",
proc_no, rpc, rcount);
#endif
}

OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
return (-1);
}

#ifdef ISO5 || defined(ISO6)
sysdate (sdate);
#endif
#ifdef ISO5
printf ("Delivery sleep before commit at %s\n", sdate);
#else
printf ("Delivery sleep before abort at %s\n", sdate);
#endif
#ifdef ISO5
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
#ifdef 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++)
delP->del_o_id[i] = 0;
for (i = 0; i < rpc; i++)
delP->del_o_id[dctx->d_id[i] - 1] = dctx->del_o_id[i];
#endif

return (0);
}

void tkvcddone (ora_cn_data_t *ora_SlotDataP)
{
delctx *dctx = (delctx *)ora_SlotDataP->dctx;
global_delivery_t *delP = ora_SlotDataP->delP;

if (dctx)
{
#ifdef ISO || defined(ISO5) || defined(ISO6) || defined(ISO8)
OCIHandleFree((dvoid *)dctx->curd1,OCI_HTYPE_STMT);
#endif
#ifdef PLSQDEL
OCIHandleFree((dvoid *)dctx->curd0,OCI_HTYPE_STMT);
#else
/* Again the above will cause a problem if both PLSQDEL and ISO are
defined - VMM 12/30/97 */
OCIHandleFree((dvoid *)dctx->curd1,OCI_HTYPE_STMT);
OCIHandleFree((dvoid *)dctx->curd2,OCI_HTYPE_STMT);
OCIHandleFree((dvoid *)dctx->curd3,OCI_HTYPE_STMT);
OCIHandleFree((dvoid *)dctx->curd4,OCI_HTYPE_STMT);
OCIHandleFree((dvoid *)dctx->curd5,OCI_HTYPE_STMT);
OCIHandleFree((dvoid *)dctx->curd6,OCI_HTYPE_STMT);
#endif
free (dctx);
ora_SlotDataP->dctx = NULL;
}

if (delP) {
free(delP);
ora_SlotDataP->delP = NULL;
}
}

#ifdef RCSID
static char *RCSid =

```

plnew.c

```

"Header: /afs/transarc.com/project/encina/rcs/test/src/benchmarks/tpcc/server/ora8.1_mt/RCS/plnew.c,v
1.3 1999/05/26 16:29:56wenjian Exp $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=====
| Copyright (c) 1996, 1997, 1998 OracleCorp, 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 "plora.h"
#ifdef TUX
#include <userlog.h>
#endif
#include "tpccflags.h"

extern void err_printf(char *format, ...);

#define PLSQLNO

#ifdef PLSQLNO
#define SQLTXT2 "BEGIN initnew.new_init(:idx1arr);END;"
#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"

#define SQLTXT3 "
SELECT 0,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :10 AND s_w_id = :30 AND s_i_id = i_id UNION ALL \
SELECT 1,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :11 AND s_w_id = :31 AND s_i_id = i_id UNION ALL \
SELECT 2,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :12 AND s_w_id = :32 AND s_i_id = i_id UNION ALL \
SELECT 3,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :13 AND s_w_id = :33 AND s_i_id = i_id UNION ALL \
SELECT 4,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :14 AND s_w_id = :34 AND s_i_id = i_id UNION ALL \
SELECT 5,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :15 AND s_w_id = :35 AND s_i_id = i_id UNION ALL \
SELECT 6,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :16 AND s_w_id = :36 AND s_i_id = i_id UNION ALL \
SELECT 7,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :17 AND s_w_id = :37 AND s_i_id = i_id UNION ALL \
SELECT 8,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :18 AND s_w_id = :38 AND s_i_id = i_id UNION ALL \
SELECT 9,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :19 AND s_w_id = :39 AND s_i_id = i_id UNION ALL \
SELECT 10,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :20 AND s_w_id = :40 AND s_i_id = i_id UNION ALL \
SELECT 11,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :21 AND s_w_id = :41 AND s_i_id = i_id UNION ALL \
SELECT 12,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :22 AND s_w_id = :42 AND s_i_id = i_id UNION ALL \
SELECT 13,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :23 AND s_w_id = :43 AND s_i_id = i_id UNION ALL \
SELECT 14,stock.rowid,i_price,i_name,i_data,s_dist%02d,s_data,s_quantity \
FROM item,stock WHERE i_id = :24 AND s_w_id = :44 AND s_i_id = i_id"

#define SQLTXT4 "INSERT INTO order_line \
(ol_o_id,ol_d_id,ol_w_id,ol_number,ol_delivery_d,ol_i_id, \
ol_supply_w_id,ol_quantity,ol_amount,ol_dist_info) \
VALUES (:ol_o_id, :ol_d_id, \
:ol_w_id, :ol_number, :null_date, :ol_i_id, :ol_supply_w_id, :ol_quantity, \
:ol_amount, :ol_dist_info)"
#endif /* PLSQLNO */

#define NITEMS 15
#define ROWIDLEN 20
#define OCICROWLEN 20

sb4 no_data(dvoid *ctxp, OCIBind *bp, ub4 iter, ub4 index,
dvoid **bufpp, ub4 *alenp, ub1 *piecep,
dvoid **indpp)
{
*bufpp = (dvoid*)0;
*alenp = 0;
*indpp = (dvoid*)0;
*piecep = OCI_ONE_PIECE;
return (OCI_CONTINUE);
}

struct newctx {
sb2 no_l_i_id_ind[NITEMS];
sb2 no_l_supply_w_id_ind[NITEMS];
sb2 no_l_quantity_ind[NITEMS];
sb2 no_l_amount_ind[NITEMS];
sb2 l_name_ind[NITEMS];
sb2 s_quantity_ind[NITEMS];
sb2 i_price_ind[NITEMS];
sb2 ol_w_id_ind[NITEMS];

```

<pre> sb2 ol_d_id_ind[NITEMS]; sb2 ol_o_id_ind[NITEMS]; sb2 ol_number_ind[NITEMS]; sb2 cons_ind[NITEMS]; sb2 s_rowid_ind[NITEMS]; sb2 s_remote_ind[NITEMS]; sb2 s_quant_ind[NITEMS]; sb2 i_data_ind[NITEMS]; sb2 s_data_ind[NITEMS]; sb2 s_dist_info_ind[NITEMS]; sb2 ol_dist_info_ind[NITEMS]; sb2 null_date_ind[NITEMS]; #ifdef PLSQLNO sb2 s_bg_ind[NITEMS]; #endif ub2 nol_i_id_len[NITEMS]; ub2 nol_supply_w_id_len[NITEMS]; ub2 nol_quantity_len[NITEMS]; ub2 nol_amount_len[NITEMS]; ub2 s_quantity_len[NITEMS]; ub2 i_name_len[NITEMS]; ub2 i_price_len[NITEMS]; ub2 i_data_len[NITEMS]; ub2 s_dist_info_len[NITEMS]; ub2 s_data_len[NITEMS]; ub2 ol_w_id_len[NITEMS]; ub2 ol_d_id_len[NITEMS]; ub2 ol_o_id_len[NITEMS]; ub2 ol_number_len[NITEMS]; ub2 cons_len[NITEMS]; ub2 s_rowid_len[NITEMS]; ub2 s_remote_len[NITEMS]; ub2 s_quant_len[NITEMS]; ub2 ol_dist_info_len[NITEMS]; ub2 null_date_len[NITEMS]; #ifdef PLSQLNO ub2 s_bg_len[NITEMS]; #endif ub2 nol_i_id_rcode[NITEMS]; ub2 nol_supply_w_id_rcode[NITEMS]; ub2 nol_quantity_rcode[NITEMS]; ub2 nol_amount_rcode[NITEMS]; ub2 i_name_rcode[NITEMS]; ub2 s_quantity_rcode[NITEMS]; ub2 i_price_rcode[NITEMS]; ub2 ol_w_id_rcode[NITEMS]; ub2 ol_d_id_rcode[NITEMS]; ub2 ol_o_id_rcode[NITEMS]; ub2 ol_number_rcode[NITEMS]; ub2 cons_rcode[NITEMS]; ub2 s_rowid_rcode[NITEMS]; ub2 s_remote_rcode[NITEMS]; ub2 s_quant_rcode[NITEMS]; ub2 i_data_rcode[NITEMS]; ub2 s_data_rcode[NITEMS]; ub2 s_dist_info_rcode[NITEMS]; ub2 ol_dist_info_rcode[NITEMS]; ub2 null_date_rcode[NITEMS]; #ifdef PLSQLNO ub2 s_bg_rcode[NITEMS]; #endif int ol_w_id[NITEMS]; int ol_d_id[NITEMS]; int ol_o_id[NITEMS]; int ol_number[NITEMS]; int cons[NITEMS]; OCIRowid *s_rowid_ptr[NITEMS]; int s_remote[NITEMS]; char i_data[NITEMS][51]; char s_data[NITEMS][51]; char s_dist_info[NITEMS][25]; OCIDate null_date[NITEMS]; /* base date for null date entry */ OCISmt *curn1; #ifdef PLSQLNO OCIBind *ol_i_id_bp; OCIBind *ol_supply_w_id_bp; OCIBind *i_price_bp; OCIBind *i_name_bp; OCIBind *s_bg_bp; OCIBind *s_data_bp; OCIBind *i_data_bp; ub4 nol_i_count; ub4 nol_s_count; ub4 nol_q_count; ub4 nol_item_count; ub4 nol_name_count; ub4 nol_qty_count; ub4 nol_bg_count; ub4 nol_am_count; ub4 s_remote_count; ub4 s_data_count; ub4 i_data_count; #endif OCISmt *curn2; </pre>	<pre> OCISmt *curn3[10]; OCIBind *ol_i_id_bp4; OCIBind *ol_supply_w_id_bp4; OCIBind *ol_quantity_bp; OCIBind *ol_quantity_bp4; OCIBind *s_remote_bp; OCIBind *s_quantity_bp; OCISmt *curn4; OCIBind *w_id_bp; OCIBind *d_id_bp; OCIBind *c_id_bp; OCIBind *o_all_local_bp; OCIBind *o_all_cnt_bp; OCIBind *w_tax_bp; OCIBind *d_tax_bp; OCIBind *o_id_bp; OCIBind *c_discount_bp; OCIBind *c_credit_bp; OCIBind *c_last_bp; OCIBind *retries_bp; OCIBind *cr_date_bp; OCIBind *s_rowid_bp; OCIBind *id_bp[10][15]; OCIBind *sd_bp[10][15]; OCIDefine *Dcons[10]; OCIDefine *Ds_rowid[10]; OCIDefine *Di_price[10]; OCIDefine *Di_data[10]; OCIDefine *Ds_dist_info[10]; OCIDefine *Ds_data[10]; OCIDefine *Ds_quantity[10]; OCIDefine *Di_name[10]; OCIBind *ol_o_id_bp; OCIBind *ol_d_id_bp; OCIBind *ol_w_id_bp; OCIBind *ol_number_bp; OCIBind *ol_amount_bp; OCIBind *ol_dist_info_bp; OCIBind *null_date_bp; sb2 w_id_ind; ub2 w_id_len; ub2 w_id_rc; sb2 d_id_ind; ub2 d_id_len; ub2 d_id_rc; sb2 c_id_ind; ub2 c_id_len; ub2 c_id_rc; sb2 o_all_local_ind; ub2 o_all_local_len; ub2 o_all_local_rc; sb2 o_all_cnt_ind; ub2 o_all_cnt_len; ub2 o_all_cnt_rc; sb2 w_tax_ind; ub2 w_tax_len; ub2 w_tax_rc; sb2 d_tax_ind; ub2 d_tax_len; ub2 d_tax_rc; sb2 o_id_ind; ub2 o_id_len; ub2 o_id_rc; sb2 c_discount_ind; ub2 c_discount_len; ub2 c_discount_rc; sb2 c_credit_ind; ub2 c_credit_len; ub2 c_credit_rc; sb2 c_last_ind; ub2 c_last_len; ub2 c_last_rc; sb2 retries_ind; ub2 retries_len; ub2 retries_rc; sb2 cr_date_ind; ub2 cr_date_len; ub2 cr_date_rc; int cs; int norow; /* context holders */ int i_name_ctx; int i_data_ctx; int i_price_ctx; </pre>
--	--

```

int s_data_ctx;
int s_dist_info_ctx;
int s_quantity_ctx;
);
typedef struct newctx newctx;

/* newctx *nctx; */

tkvcninit (ora_cn_data_t *ora_SlotDataP)
{
int i, j;
text stmbuf[16*1024];
char id[4];
char sid[4];

newctx *nctx;
OCIEnv *tpcenv = ora_SlotDataP->tpcenv;
OCIServer *tpcsrv = ora_SlotDataP->tpcsrv;
OCIErr *errhp = ora_SlotDataP->errhp;
OCISvcCtx *tpscvc = ora_SlotDataP->tpscvc;
OCISession *tpcsr = ora_SlotDataP->tpcsr;
OCISmt *curi = ora_SlotDataP->curi;
global_newOrder_t *newP;

nctx = (newctx *) malloc (sizeof(newctx));
memset(nctx, (char)0, sizeof(newctx));
ora_SlotDataP->nctx = (void *)nctx;

ora_SlotDataP->globals = (global_newOrder_t *) malloc (sizeof(global_newOrder_t));
memset(ora_SlotDataP->globals, (char)0, sizeof(global_newOrder_t));
newP = ora_SlotDataP->globals;

nctx->cs = 1;
nctx->norow=0;
for(i=0; i<NITEMS; i++){
OCIERROR(errhp, OCIDescriptorAlloc(tpcenv, (dvoid **)&nctx->s_rowid_ptr[i],
OCL_DTYPE_ROWID, 0, (dvoid **)&0));
}
nctx->w_id_ind = TRUE;
nctx->w_id_len = sizeof(newP->w_id);
nctx->d_id_ind = TRUE;
nctx->d_id_len = sizeof(newP->d_id);
nctx->c_id_ind = TRUE;
nctx->c_id_len = sizeof(newP->c_id);
nctx->o_all_local_ind = TRUE;
nctx->o_all_local_len = sizeof(newP->o_all_local);
nctx->o_of_cnt_ind = TRUE;
nctx->o_of_cnt_len = sizeof(newP->o_of_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(newP->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(newP->retries);
nctx->cr_date_ind = TRUE;
nctx->cr_date_len = sizeof(newP->cr_date);

/* open first cursor */
OCIERROR(errhp, OCIHandleAlloc(tpcenv, (dvoid **)&nctx->cur1,
OCI_HTYPE_STMT, 0, (dvoid **)&0));
#ifdef PLSQLNO
sqlfile("tkvcnew.sql", stmbuf);
#else
sqlfile("tkvcbnew.sql", stmbuf);
#endif
OCIERROR(errhp, OCISmtPrepare(nctx->cur1, errhp, stmbuf, strlen((char *)stmbuf),
OCI_NTV_SYNTAX, OCI_DEFAULT));

/* bind variables */
OCIBNDR(nctx->cur1, nctx->w_id_bp, errhp, "w_id", ADR(newP->w_id), SIZ(newP->w_id),
SQLT_INT, &nctx->w_id_ind, &nctx->w_id_len, &nctx->w_id_rc);
OCIBNDR(nctx->cur1, nctx->d_id_bp, errhp, "d_id", ADR(newP->d_id), SIZ(newP->d_id),
SQLT_INT, &nctx->d_id_ind, &nctx->d_id_len, &nctx->d_id_rc);
OCIBNDR(nctx->cur1, nctx->c_id_bp, errhp, "c_id", ADR(newP->c_id), SIZ(newP->c_id),
SQLT_INT, &nctx->c_id_ind, &nctx->c_id_len, &nctx->c_id_rc);
OCIBNDR(nctx->cur1, nctx->o_all_local_bp, errhp, "o_all_local",
ADR(newP->o_all_local), SIZ(newP->o_all_local), SQLT_INT, &nctx->o_all_local_ind,
&nctx->o_all_local_len, &nctx->o_all_local_rc);
OCIBNDR(nctx->cur1, nctx->o_of_cnt_bp, errhp, "o_of_cnt", ADR(newP->o_of_cnt),
SIZ(newP->o_of_cnt), SQLT_INT, &nctx->o_of_cnt_ind, &nctx->o_of_cnt_len,
&nctx->o_of_cnt_rc);
OCIBNDR(nctx->cur1, nctx->w_tax_bp, errhp, "w_tax", ADR(newP->w_tax), SIZ(newP->w_tax),
SQLT_FLT, &nctx->w_tax_ind, &nctx->w_tax_len, &nctx->w_tax_rc);
OCIBNDR(nctx->cur1, nctx->d_tax_bp, errhp, "d_tax", ADR(newP->d_tax), SIZ(newP->d_tax),
SQLT_FLT, &nctx->d_tax_ind, &nctx->d_tax_len, &nctx->d_tax_rc);
OCIBNDR(nctx->cur1, nctx->o_id_bp, errhp, "o_id", ADR(newP->o_id), SIZ(newP->o_id),
SQLT_INT, &nctx->o_id_ind, &nctx->o_id_len, &nctx->o_id_rc);
OCIBNDR(nctx->cur1, nctx->c_discount_bp, errhp, "c_discount",
ADR(newP->c_discount), SIZ(newP->c_discount), SQLT_FLT,
&nctx->c_discount_ind, &nctx->c_discount_len, &nctx->c_discount_rc);
OCIBNDR(nctx->cur1, nctx->c_credit_bp, errhp, "c_credit", newP->c_credit,
SIZ(newP->c_credit), SQLT_CHR,
&nctx->c_credit_ind, &nctx->c_credit_len, &nctx->c_credit_rc);
OCIBNDR(nctx->cur1, nctx->c_last_bp, errhp, "c_last", newP->c_last, SIZ(newP->c_last),
SQLT_STR, &nctx->c_last_ind, &nctx->c_last_len, &nctx->c_last_rc);
OCIBNDR(nctx->cur1, nctx->retries_bp, errhp, "retries", ADR(newP->retries),
SIZ(newP->retries), SQLT_INT,
&nctx->retries_ind, &nctx->retries_len, &nctx->retries_rc);
OCIBNDR(nctx->cur1, nctx->cr_date_bp, errhp, "cr_date", &nctx->cr_date, SIZ(OCIDate),
SQLT_ODT, &nctx->cr_date_ind, &nctx->cr_date_len, &nctx->cr_date_rc);

#ifdef PLSQLNO
OCIBNDRAA(nctx->cur1, nctx->ol_i_id_bp, errhp, "ol_i_id", newP->ol_i_id,
SIZ(int), SQLT_INT, nctx->ol_i_id_ind, nctx->ol_i_id_len,
nctx->ol_i_id_rcode, NITEMS, &nctx->ol_i_id_count);
OCIBNDRAA(nctx->cur1, nctx->ol_supply_w_id_bp, errhp, "ol_supply_w_id",
newP->ol_supply_w_id, SIZ(int), SQLT_INT, nctx->ol_supply_w_id_ind,
nctx->ol_supply_w_id_len, nctx->ol_supply_w_id_rcode,
NITEMS, &nctx->ol_s_count);
OCIBNDRAA(nctx->cur1, nctx->ol_quantity_bp, errhp, "ol_quantity", newP->ol_quantity,
SIZ(int), SQLT_INT, nctx->ol_quantity_ind, nctx->ol_quantity_len,
nctx->ol_quantity_rcode, NITEMS, &nctx->ol_q_count);
OCIBNDRAA(nctx->cur1, nctx->i_price_bp, errhp, "i_price", newP->i_price, SIZ(float),
SQLT_FLT, nctx->i_price_ind, nctx->i_price_len, nctx->i_price_rcode,
NITEMS, &nctx->ol_item_count);
OCIBNDRAA(nctx->cur1, nctx->i_name_bp, errhp, "i_name", newP->i_name,
SIZ(newP->i_name[0]), SQLT_STR, nctx->i_name_ind, nctx->i_name_len,
nctx->i_name_rcode, NITEMS, &nctx->ol_name_count);
OCIBNDRAA(nctx->cur1, nctx->s_quantity_bp, errhp, "s_quantity", newP->s_quantity,
SIZ(int), SQLT_INT, nctx->s_quant_ind, nctx->s_quant_len,
nctx->s_quant_rcode, NITEMS, &nctx->ol_qty_count);
OCIBNDRAA(nctx->cur1, nctx->s_bg_bp, errhp, "brand_generic", newP->brand_generic,
SIZ(char), SQLT_CHR, nctx->s_bg_ind, nctx->s_bg_len,
nctx->s_bg_rcode, NITEMS, &nctx->ol_bg_count);
OCIBNDRAA(nctx->cur1, nctx->ol_amount_bp, errhp, "ol_amount", newP->ol_amount,
SIZ(int), SQLT_INT, nctx->ol_amount_ind, nctx->ol_amount_len,
nctx->ol_amount_rcode, NITEMS, &nctx->ol_am_count);
OCIBNDRAA(nctx->cur1, nctx->s_remote_bp, errhp, "s_remote", nctx->s_remote,
SIZ(int), SQLT_INT, nctx->s_remote_ind, nctx->s_remote_len,
nctx->s_remote_rcode, NITEMS, &nctx->s_remote_count);

/* open second cursor */
OCIERROR(errhp, OCIHandleAlloc(tpcenv, (dvoid **)&nctx->cur2, OCI_HTYPE_STMT,
0, (dvoid **)&0));
sprintf((char *) stmbuf, SQLTEXT2);
OCIERROR(errhp, OCISmtPrepare(nctx->cur2, errhp, stmbuf,
strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));

/* execute second cursor to init newinit package */
{
int idx1arr[NITEMS];
OCIBind *idx1arr_bp;
ub2 idx1arr_len[NITEMS];
ub2 idx1arr_rcode[NITEMS];
sb2 idx1arr_ind[NITEMS];
ub4 idx1arr_count;
ub2 idx;

for (idx = 0; idx < NITEMS; idx++) {
idx1arr[idx] = idx + 1;
idx1arr_ind[idx] = TRUE;
idx1arr_len[idx] = sizeof(int);
}
idx1arr_count = NITEMS;
newP->o_of_cnt = NITEMS;

/* Bind array */
OCIBNDRAA(nctx->cur2, idx1arr_bp, errhp, "idx1arr", idx1arr,
SIZ(int), SQLT_INT, idx1arr_ind, idx1arr_len,
idx1arr_rcode, NITEMS, &idx1arr_count);

newP->execstatus = OCISmtExecute(tpscvc, nctx->cur2, errhp, 1, 0, 0, OCI_DEFAULT);
if (newP->execstatus != OCI_SUCCESS) {
OCITransRollback(tpscvc, errhp, OCI_DEFAULT);
newP->errcode = OCIERROR(errhp, newP->execstatus);
return -1;
}
}
#else
/* open second cursor */
OCIERROR(errhp, OCIHandleAlloc(tpcenv, (dvoid **)&nctx->cur2, OCI_HTYPE_STMT,
0, (dvoid **)&0));
sprintf((char *) stmbuf, SQLTEXT2);
OCIERROR(errhp, OCISmtPrepare(nctx->cur2, errhp, stmbuf,
strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT));

/* bind variables */
OCIBNDRA(nctx->cur2, nctx->s_quantity_bp, errhp, "s_quantity", newP->s_quantity,
SIZ(int), SQLT_INT, nctx->s_quant_ind, nctx->s_quant_len,
nctx->s_quant_rcode);
OCIBNDRA(nctx->cur2, nctx->s_rowid_bp, errhp, "s_rowid", nctx->s_rowid_ptr,

```



```

/* initialization for array operations*/
for (i = 0; i < newP->o_ol_cnt; i++) {
    nctx->ol_w_id[i] = newP->w_id;
    nctx->ol_d_id[i] = newP->d_id;
    nctx->ol_number[i] = i + 1;
    nctx->null_date_ind[i] = TRUE;
    nctx->nol_i_id_ind[i] = 0;
    nctx->nol_supply_w_id_ind[i] = TRUE;
    nctx->nol_quantity_ind[i] = TRUE;
    nctx->nol_amount_ind[i] = TRUE;
    nctx->ol_w_id_ind[i] = TRUE;
    nctx->ol_d_id_ind[i] = TRUE;
    nctx->ol_o_id_ind[i] = TRUE;
    nctx->ol_number_ind[i] = TRUE;
    nctx->ol_dist_info_ind[i] = TRUE;
    nctx->s_remote_ind[i] = TRUE;
    nctx->s_data_ind[i] = TRUE;
    nctx->i_data_ind[i] = TRUE;
    nctx->s_quant_ind[i] = TRUE;
    nctx->s_bg_ind[i] = TRUE;
    nctx->cons_ind[i] = TRUE;
    nctx->s_rowid_ind[i] = TRUE;
    nctx->nol_i_id_len[i] = sizeof(int);
    nctx->nol_supply_w_id_len[i] = sizeof(int);
    nctx->nol_quantity_len[i] = sizeof(int);
    nctx->nol_amount_len[i] = sizeof(int);
    nctx->ol_w_id_len[i] = sizeof(int);
    nctx->ol_d_id_len[i] = sizeof(int);
    nctx->ol_o_id_len[i] = sizeof(int);
    nctx->ol_number_len[i] = sizeof(int);
    nctx->ol_dist_info_len[i] = nctx->s_dist_info_len[i];
    nctx->null_date_len[i] = sizeof(OCIDate);
    nctx->s_remote_len[i] = sizeof(int);
    nctx->s_data_len[i] = sizeof(int);
    nctx->i_data_len[i] = sizeof(int);
    nctx->s_quant_len[i] = sizeof(int);
    nctx->s_rowid_len[i] = sizeof(nctx->s_rowid_ptr[0]);
    nctx->cons_len[i] = sizeof(int);
    nctx->i_name_len[i] = 0;
    nctx->s_bg_len[i] = 0;
}

for (i = newP->o_ol_cnt; i < NITEMS; i++) {
    nctx->nol_i_id_ind[i] = NA;
    nctx->nol_supply_w_id_ind[i] = NA;
    nctx->nol_quantity_ind[i] = NA;
    nctx->nol_amount_ind[i] = NA;
    nctx->ol_w_id_ind[i] = NA;
    nctx->ol_d_id_ind[i] = NA;
    nctx->ol_o_id_ind[i] = NA;
    nctx->ol_number_ind[i] = NA;
    nctx->ol_dist_info_ind[i] = NA;
    nctx->null_date_ind[i] = NA;
    nctx->s_remote_ind[i] = NA;
    nctx->s_data_ind[i] = NA;
    nctx->i_data_ind[i] = NA;
    nctx->s_quant_ind[i] = NA;
    nctx->s_bg_ind[i] = NA;
    nctx->cons_ind[i] = NA;
    nctx->s_rowid_ind[i] = NA;

    nctx->nol_i_id_len[i] = 0;
    nctx->nol_supply_w_id_len[i] = 0;
    nctx->nol_quantity_len[i] = 0;
    nctx->nol_amount_len[i] = 0;
    nctx->ol_w_id_len[i] = 0;
    nctx->ol_d_id_len[i] = 0;
    nctx->ol_o_id_len[i] = 0;
    nctx->ol_number_len[i] = 0;
    nctx->ol_dist_info_len[i] = 0;
    nctx->null_date_len[i] = 0;
    nctx->s_remote_len[i] = 0;
    nctx->i_data_len[i] = 0;
    nctx->s_data_len[i] = 0;
    nctx->s_quant_len[i] = 0;
    nctx->s_bg_len[i] = 0;
    nctx->cons_len[i] = 0;
    nctx->i_name_len[i] = 0;
    nctx->s_bg_len[i] = 0;
}

newP->execstatus = OCISmtExecute(tpcsvc, nctx->curr1, errhp, 1, 0, 0,
                                OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);

#else
newP->execstatus = OCISmtExecute(tpcsvc, nctx->curr1, errhp, 1, 0, 0, OCI_DEFAULT);
#endif

if(newP->execstatus != OCI_SUCCESS) {
    OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
    newP->errcode = OCIERROR(errhp, newP->execstatus);
    if(newP->errcode == NOT_SERIALIZABLE) {
        newP->retries++;
        goto retry;
    } else if(newP->errcode == RECOVER) {
        newP->retries++;
        goto retry;
    }
}

```

```

else {
    return -1;
}

#endif PLSQLNO
/* did the txn succeed ? */
if (rcount != newP->o_ol_cnt)
{
    newP->status = rcount - newP->o_ol_cnt;
    newP->o_ol_cnt = rcount;
}
#endif

#ifdef DEBUG
err_printf("tkvcn (NO): w_id = %d, d_id = %d, c_id = %d\n", w_id, d_id, c_id);
#endif

#endif PLSQLNO
/* initialization for array operations*/

for (i = 0; i < o_ol_cnt; i++) {
    nctx->ol_w_id[i] = w_id;
    nctx->ol_d_id[i] = d_id;
    nctx->ol_number[i] = i + 1;
    nctx->null_date_ind[i] = TRUE;
    nctx->nol_i_id_ind[i] = TRUE;
    nctx->nol_supply_w_id_ind[i] = TRUE;
    nctx->nol_quantity_ind[i] = TRUE;
    nctx->nol_amount_ind[i] = TRUE;
    nctx->ol_w_id_ind[i] = TRUE;
    nctx->ol_d_id_ind[i] = TRUE;
    nctx->ol_o_id_ind[i] = TRUE;
    nctx->ol_number_ind[i] = TRUE;
    nctx->ol_dist_info_ind[i] = TRUE;
    nctx->s_remote_ind[i] = TRUE;
    nctx->s_data_ind[i] = TRUE;
    nctx->i_data_ind[i] = TRUE;
    nctx->s_quant_ind[i] = TRUE;
    nctx->cons_ind[i] = TRUE;
    nctx->s_rowid_ind[i] = TRUE;

    nctx->nol_i_id_len[i] = sizeof(int);
    nctx->nol_supply_w_id_len[i] = sizeof(int);
    nctx->nol_quantity_len[i] = sizeof(int);
    nctx->nol_amount_len[i] = sizeof(int);
    nctx->ol_w_id_len[i] = sizeof(int);
    nctx->ol_d_id_len[i] = sizeof(int);
    nctx->ol_o_id_len[i] = sizeof(int);
    nctx->ol_number_len[i] = sizeof(int);
    nctx->ol_dist_info_len[i] = nctx->s_dist_info_len[i];
    nctx->null_date_len[i] = sizeof(OCIDate);
    nctx->s_remote_len[i] = sizeof(int);
    nctx->s_data_len[i] = sizeof(int);
    nctx->i_data_len[i] = sizeof(int);
    nctx->s_quant_len[i] = sizeof(int);
    nctx->s_rowid_len[i] = sizeof(nctx->s_rowid_ptr[0]);
    nctx->cons_len[i] = sizeof(int);
}

for (i = o_ol_cnt; i < NITEMS; i++) {
    nctx->nol_i_id_ind[i] = NA;
    nctx->nol_supply_w_id_ind[i] = NA;
    nctx->nol_quantity_ind[i] = NA;
    nctx->nol_amount_ind[i] = NA;
    nctx->ol_w_id_ind[i] = NA;
    nctx->ol_d_id_ind[i] = NA;
    nctx->ol_o_id_ind[i] = NA;
    nctx->ol_number_ind[i] = NA;
    nctx->ol_dist_info_ind[i] = NA;
    nctx->null_date_ind[i] = NA;
    nctx->s_remote_ind[i] = NA;
    nctx->s_data_ind[i] = NA;
    nctx->i_data_ind[i] = NA;
    nctx->s_quant_ind[i] = NA;
    nctx->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_data_len[i] = 0;
    nctx->s_quant_len[i] = 0;
    nctx->cons_len[i] = 0;
}

rpc3 = SellItemStk (nctx, newP, tpcsvc, errhp);
if (rpc3 == -2)
    goto retry;
else if (rpc3 == -1)
    return (-1);

/* compute order line amounts, total amount and stock quantities */

total_amount = 0.0;
for (i = 0; i < newP->o_ol_cnt; i++)
{

```



```

nctx->ol_o_id[i] = newP->o_id;
if (nctx->no_l_i_id_ind[i] != NA) {
    newP->s_quantity[i] -= newP->no_l_quantity[i];
    if (newP->s_quantity[i] < 10)
        newP->s_quantity[i] += 91;
    newP->no_l_amount[i] = (newP->no_l_quantity[i] * newP->i_price[i]);
    newP->total_amount += newP->no_l_amount[i];
    if (strchr(nctx->i_data[i], "ORIGINAL") &&
        strchr(nctx->s_data[i], "ORIGINAL"))
        newP->brand_gen[i] = 'B';
    else
        newP->brand_gen[i] = 'G';
}
}
total_amount *= ((float)(10000 - c_discount)/10000) * (1.0 + ((float)(d_tax)/10000) +
((float)(w_tax)/10000));
newP->total_amount = newP->total_amount/100;

rpc = UpdStk2 (nctx, newP, tpcsvc, errhp);
if (rpc == -2)
    goto retry;
else if (rpc == -1)
    return (-1);

/* error processing - will keep it separated for readability */
/* number of items selected != number of stock updated */

if (rpc3 != rpc) {
    userlog ("Error in TPC-C server %d: %d rows of item read, ",
            newP->proc_no, rpc3);
    userlog ("          but %d rows of stock updated\n", rpc);
    /* rollback */
    OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
    return (-1);
}

/* common code for insert into order_line */
for (i=0; i<newP->o_ol_cnt; i++) /* move district info in place */
{
    nctx->ol_dist_info_len[i] = nctx->s_dist_info_len[i];
}

/* array insert into order line table */
flags = (newP->status ? OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
if ((newP->o_ol_cnt - newP->status) > 0)
{
    newP->execstatus = OCISmtExecute(tpcsvc, nctx->curm4, errhp, newP->o_ol_cnt - newP->status,
    0, 0, flags);
    if (newP->execstatus != OCI_SUCCESS) {
        OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
        newP->errcode = OCIErrror(errhp, execstatus);
        if (newP->errcode == NOT_SERIALIZABLE) {
            newP->retries++;
            goto retry;
        } else if (newP->errcode == RECOVER) {
            newP->retries++;
            goto retry;
        } else {
            return -1;
        }
    }
    OCIAttrGet(nctx->curm4, OCI_HTYPE_STMT, &rcount, NULL,
    OCI_ATTR_ROW_COUNT, errhp);
    if (rcount != (newP->o_ol_cnt - newP->status))
    {
        userlog ("Error in TPC-C server %d: array insert failed\n",
            newP->proc_no);
        /* rollback */
        OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
        return (-1);
    }
}

/* commit if no invalid item */

if (newP->status) {
    OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
    fflush(stdout);
}

#else
newP->total_amount = 0.0;
for (i = 0; i < newP->o_ol_cnt; i++)
{
    if (nctx->no_l_i_id_ind[i] != NA) {
        newP->total_amount += newP->no_l_amount[i];
    }
}
newP->total_amount *= ((float)(10000 - newP->c_discount)/10000) * (1.0 + ((float)(newP->d_tax)/
10000) + ((float)(newP->w_tax)/10000));
newP->total_amount = newP->total_amount/100;
#endif
return (0);
}

```

```

void tkvcndone (ora_cn_data_t *ora_SlotDataP)
{
    int i;
    newctx *nctx = (newctx *)ora_SlotDataP->nctx;
    global_newOrder_t *newP = ora_SlotDataP->globals;

    if (nctx)
    {
        OCIHandleFree((dvoid *)nctx->curm1, OCI_HTYPE_STMT);
        OCIHandleFree((dvoid *)nctx->curm2, OCI_HTYPE_STMT);
        for (i = 0; i < 10; i++)
            OCIHandleFree((dvoid *)nctx->curm3[i], OCI_HTYPE_STMT);
        OCIHandleFree((dvoid *)nctx->curm4, OCI_HTYPE_STMT);
        free (nctx);
    }
    if (newP) {
        err_printf("free_handles> newP: 0x%x\n", newP);
        free(newP);
        ora_SlotDataP->globals = NULL;
    }
}

/* 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, nctx, newP)

int i, j;
newctx *nctx;
global_newOrder_t *newP;
{
    /* shift up the values for the stock table */
    nctx->s_remote[i] = nctx->s_remote[j];

    /* shift up the order_line values */

    nctx->no_l_i_id_ind[i] = nctx->no_l_i_id_ind[j];
    newP->no_l_i_id[i] = newP->no_l_i_id[j];

    nctx->no_l_quantity_ind[i] = nctx->no_l_quantity_ind[j];
    newP->no_l_quantity[i] = newP->no_l_quantity[j];

    nctx->no_l_supply_w_id_ind [i] = nctx->no_l_supply_w_id_ind[j];
    newP->no_l_supply_w_id[i] = newP->no_l_supply_w_id[j];
}

#if 0
/* TODO - this routine is not ever called. So, no changes for now */

swapitemstock (i, j)

int i, j;

{
    int k;
    int tempi;
    int tempf;
    char tempstr[52];
    ub2 tempub2;
    sb2 tempub2;
    OCIRowid *tmpRID;

    tempub2 = nctx->cons_ind[i];
    nctx->cons_ind[i] = nctx->cons_ind[j];
    nctx->cons_ind[j] = tempub2;
    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;

    tempub2 = nctx->s_rowid_ind[i];
    nctx->s_rowid_ind[i] = nctx->s_rowid_ind[j];
    nctx->s_rowid_ind[j] = tempub2;
    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;

    tempub2 = nctx->i_price_ind[i];
    nctx->i_price_ind[i] = nctx->i_price_ind[j];
    nctx->i_price_ind[j] = tempub2;
    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;

tempub2 = nctx->i_name_ind[i];
nctx->i_name_ind[i] = nctx->i_name_ind[j];
nctx->i_name_ind[j] = tempub2;
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);

tempub2 = nctx->i_data_ind[i];
nctx->i_data_ind[i] = nctx->i_data_ind[j];
nctx->i_data_ind[j] = tempub2;
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);

tempub2 = nctx->s_quantity_ind[i];
nctx->s_quantity_ind[i] = nctx->s_quantity_ind[j];
nctx->s_quantity_ind[j] = tempub2;
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;

tempub2 = nctx->s_dist_info_ind[i];
nctx->s_dist_info_ind[i] = nctx->s_dist_info_ind[j];
nctx->s_dist_info_ind[j] = tempub2;
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);

tempub2 = nctx->s_data_ind[i];
nctx->s_data_ind[i] = nctx->s_data_ind[j];
nctx->s_data_ind[j] = tempub2;
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);
}
#endif

SellItemStk (nctx, newP, tpcsvc, errhp)
newctx *nctx;
global_newOrder_t *newP;
OCISvcCtx *tpcsvc;
OCLError *errhp;
{
    int i, j, rpc3, rcount;

    /* array select from item and stock tables */
    newP->execstatus = OCISmtExecute(tpcsvc, (nctx->curr3)[newP->d_id-1], errhp, newP->o_ol_cnt,
        0, 0, OCI_DEFAULT);
    if (newP->execstatus != OCI_SUCCESS) && (newP->execstatus != OCI_NO_DATA) {
        newP->errcode = OCIERROR(errhp, newP->execstatus);
        if (newP->errcode == NOT_SERIALIZABLE) {
            newP->retries++;
            OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
            return (-2);
        } else if (newP->errcode == RECOVER) {
            /* In case of NO_DATA this should NOT return, but simply fall through */
            OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
            newP->retries++;
            return (-2);
        }
    }
}

} else {
    OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
    return (-1);
}
}
/* mark invalid items */
OCIAttrGet((nctx->curr3)[newP->d_id-1], OCI_HTYPE_STMT, &rcount, NULL,
    OCI_ATTR_ROW_COUNT, errhp);
rpc3 = rcount;

/* the result is in order, so we have to shift up to fill */
/* the slot for the line with the invalid item. */
/* If more than one item is wrong, this is not an simulated */
/* error and we'll blow off */

if ((newP->status = newP->o_ol_cnt - rcount) > 1)
{
    userlog ("TPC-C server %d: more than 1 invaliditem?\n", proc_no);
    return (rpc3);
}
if (newP->status == 0) return (rpc3);

/* find the invalid item, transfer the rowid information */

for (i = 0; i < newP->o_ol_cnt; i++) {
    if (nctx->cons[i] != i) break; /* this item is invalid */
}

    userlog ("TPC-C server %d: reordering items and stocks\n",
        proc_no);

/* not the last item - shift up */

for (j = i; j < newP->o_ol_cnt-1; j++)
{
    shiftitemstock (j, j+1, nctx, newP);
}
/* zero the last item */
i = newP->o_ol_cnt-1;
nctx->noI_id_ind[i] = NA;
nctx->noI_supply_w_id_ind[i] = NA;
nctx->noI_quantity_ind[i] = NA;
nctx->noI_amount_ind[i] = NA;
nctx->oI_w_id_ind[i] = NA;
nctx->oI_d_id_ind[i] = NA;
nctx->oI_o_id_ind[i] = NA;
nctx->null_date_ind[i] = NA;
nctx->oI_number_ind[i] = NA;
nctx->oI_dist_info_ind[i] = NA;
nctx->s_remote_ind[i] = NA;
nctx->s_quant_ind[i] = NA;

nctx->noI_i_id_len[i] = 0;
nctx->noI_supply_w_id_len[i] = 0;
nctx->noI_quantity_len[i] = 0;
nctx->noI_amount_len[i] = 0;
nctx->oI_w_id_len[i] = 0;
nctx->oI_d_id_len[i] = 0;
nctx->oI_o_id_len[i] = 0;
nctx->oI_number_len[i] = 0;
nctx->oI_dist_info_len[i] = 0;
nctx->null_date_ind[i] = 0;
nctx->s_remote_len[i] = 0;
nctx->s_quant_len[i] = 0;

return (rpc3);
}

UpdStk2 (nctx, newP, tpcsvc, errhp)
newctx *nctx;
global_newOrder_t *newP;
OCISvcCtx *tpcsvc;
OCLError *errhp;
{
    int rpc, rowoff, iters, rcount;

    /* array update of stock table */

    newP->execstatus = OCISmtExecute(tpcsvc, nctx->curr2, errhp, newP->o_ol_cnt - newP->status, 0, 0, 0,
        OCI_DEFAULT);
    if (newP->execstatus != OCI_SUCCESS) {
        OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
        newP->errcode = OCIERROR(errhp, newP->execstatus);
        if (newP->errcode == NOT_SERIALIZABLE) {
            newP->retries++;
            return (-2);
        } else if (newP->errcode == RECOVER) {
            newP->retries++;
            return (-2);
        } else {
            return -1;
        }
    }
    OCIAttrGet(nctx->curr2, OCI_HTYPE_STMT, &rcount, NULL, OCI_ATTR_ROW_COUNT, errhp);
}

```

```

rpc = rcount;

if (rpc != (newP->o_ol_cnt - newP->status)) {
    userlog ("Error in TPC-C server %d: array update failed\n",
            newP->proc_no);
    OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);
    return (-1);
}

return (rpc);
}

```

plora.h

```

#ifndef TPCC_PLORA_H
#define TPCC_PLORA_H

```

```

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

```

```

#define NEWO_TRANS (1)
#define PAYMENT_TRANS (2)
#define ORDER_STAT_TRANS (3)
#define DELIVERY_TRANS (4)
#define STOCK_TRANS (5)
#define MAX_TRAN_TYPE (5)

```

```

/* struct to copy-in/out neworder vars */

```

```

struct global_newOrder_t {
    int w_id;
    int d_id;
    int c_id;
    int nol_i_id[15];
    int nol_supply_w_id[15];
    int nol_quantity[15];
    int retries;
    ub4 datelen;
    text o_entry_d[20];
    int o_id;
    int o_ol_cnt;
    char c_last[17];
    char c_credit[3];
    float c_discount;
    float w_tax;
    float d_tax;
    float total_amount;
    char i_name[15][25];
    int s_quantity[15];
    char brand_gen[15];
    float i_price[15];
    int nol_amount[15];
    int status;
    int o_all_local;
    int errcode;
    int execstatus;
    int proc_no;
    char brand_generic[15][1];
    int tracelevel;
    OCIDate cr_date;
    OCIDate c_since;
    OCIDate o_entry_d_base;
    OCIDate ol_d_base[15];
};

```

```

typedef struct global_newOrder_t global_newOrder_t;

```

```

struct global_payment_t {
    int w_id;
    int d_id;
    int c_id;
    char c_last[17];
    char c_first[17];
    char c_middle[3];
    double c_balance;
    int retries;
    int bylastname;
    OCIDate c_since;
    int execstatus;
    int errcode;
    int c_w_id;
    int c_d_id;
    int h_amount;
    char w_street_1[21];
    char w_street_2[21];
    char w_city[21];
    char w_state[3];
    char w_zip[10];
    char d_street_1[21];
    char d_street_2[21];
    char d_city[21];
    char d_state[3];
    char d_zip[10];
    char c_street_1[21];
    char c_street_2[21];
    char c_city[21];
    char c_state[3];
    char c_zip[10];
};

```

```

char c_phone[17];
ub4 sincelen;
text c_since_d[11];
float c_discount;
char c_credit[3];
int c_credit_lim;
char c_data[201];
ub4 hlen;
text h_date[20];
OCIDate cr_date;
};

```

```

typedef struct global_payment_t global_payment_t;

```

```

struct global_order_t {
    OCIDate ol_d_base[15];
    int w_id;
    int d_id;
    int c_id;
    char c_last[17];
    char c_first[17];
    char c_middle[3];
    double c_balance;
    int o_id;
    int o_carrier_id;
    int o_ol_cnt;
    int ol_supply_w_id[15];
    int ol_i_id[15];
    unsigned char o_entry_d_base[7];
    int ol_quantity[15];
    ub4 ol_del_len[15];
    text ol_delivery_d[15][11];
    int ol_amount[15];
    int errcode;
    int execstatus;
    int retries;
    int bylastname;
    char o_entry_d[20];
};

```

```

typedef struct global_order_t global_order_t;

```

```

struct global_delivery_t {
    int w_id;
    int o_carrier_id;
    int retries;
    int del_o_id[10];
    int errcode;
    int execstatus;
    int proc_no;
    OCIDate cr_date;
};

```

```

typedef struct global_delivery_t global_delivery_t;

```

```

struct global_stock_t {
    int w_id;
    int d_id;
    int threshold;
    int retries;
    int low_stock;
    int errcode;
    int execstatus;
};

```

```

typedef struct global_stock_t global_stock_t;

```

```

/* Oracle handles and rest of thread specific vars(thread slot data) */

```

```

struct ora_cn_data_t {
    OCIEnv *tpcenv;
    OCIServer *tpcsrv;
    OCIError *errhp;
    OCISvcCtx *tpcsvc;
    OCISession *tpcusr;
    OCISmt *curi;
    dvoid *xmem;
};

```

```

global_newOrder_t *globals;
global_payment_t *payP;
global_order_t *ordP;
global_delivery_t *delP;
global_stock_t *stoP;
void *nctx;
void *pctx;
void *octx;
void *setx;
void *dctx;
void *actx; /* for #ifdef DMLRETDL */
void *cbctx; /* for orderstatus */
void *ctxp_octx; /* for orderstatus */
};

```

```

typedef struct ora_cn_data_t ora_cn_data_t;

```

```

#endif /* TPCC_PLORA_H */

```

plord.c

```
#ifndef RCSID
static char *RCSID =
    "$Header: /afs/transarc.com/project/encina/rcs/test/src/benchmarks/tpcc/server/oras.1_mt/RCS/plord.c,v
1.2 1999/04/15 12:16:51 oExp $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=====+
|   Copyright (c) 1995 OracleCorp, Redwood Shores, CA   |
|   OPEN SYSTEMS PERFORMANCE GROUP                       |
|   All Rights Reserved                                   |
+=====+
| FILENAME
|   plord.c
| DESCRIPTION
|   OCI version (using PL/SQL anonymous block) of
|   ORDER STATUS transaction in TPC-C benchmark.
+=====*/

#include "tpcc.h"
#include "plora.h" /* */
#include "tpccflags.h"

#ifndef PLSQLORD
#define SQLTXT "BEGIN orderstatus.getstatus (:w_id, :d_id, :c_id, :byln, \
:c_last, :c_first, :c_middle, :c_balance, :o_id, :o_entry_d, :o_cr_id, \
:o_ol_cnt, :ol_s_w_id, :ol_i_id, :ol_quantity, :ol_amount, :ol_d_d); END;"
#else

#define SQLCUR0 "SELECT rowid FROM customer \
WHERE c_d_id = :d_id AND c_w_id = :w_id AND c_last = :c_last \
ORDER BY c_last, c_d_id, c_w_id, c_first"

#define SQLCUR1 "SELECT c_id, c_balance, c_first, c_middle, c_last, \
o_id, o_entry_d, o_carrier_id, o_ol_cnt \
FROM 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_c_id, o_d_id, o_w_id, o_id DESC"

#define SQLCUR2 "SELECT c_balance, c_first, c_middle, c_last, \
o_id, o_entry_d, o_carrier_id, o_ol_cnt \
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_c_id, o_d_id, o_w_id, o_id DESC"

#define SQLCUR3 "SELECT ol_i_id, ol_supply_w_id, ol_quantity, ol_amount, \
ol_delivery_d \
FROM order_line \
WHERE ol_d_id = :d_id AND ol_w_id = :w_id AND ol_o_id = :o_id"

#define SQLCUR4 "SELECT count(c_last) FROM customer \
WHERE c_d_id = :d_id AND c_w_id = :w_id AND c_last = :c_last"
#endif

struct ordctx {
    sb2 c_rowid_ind[100];
    sb2 ol_supply_w_id_ind[NITEMS];
    sb2 ol_i_id_ind[NITEMS];
    sb2 ol_quantity_ind[NITEMS];
    sb2 ol_amount_ind[NITEMS];
    sb2 ol_delivery_d_ind[NITEMS];
    sb2 ol_w_id_ind;
    sb2 ol_d_id_ind;
    sb2 ol_o_id_ind;
    sb2 c_id_ind;
    sb2 c_first_ind;
    sb2 c_middle_ind;
    sb2 c_balance_ind;
    sb2 c_last_ind;
    sb2 o_id_ind;
    sb2 o_entry_d_ind;
    sb2 o_carrier_id_ind;
    sb2 o_ol_cnt_ind;

    ub4 c_rowid_len[100];
    ub2 ol_supply_w_id_len[NITEMS];
    ub2 ol_i_id_len[NITEMS];
    ub2 ol_quantity_len[NITEMS];
    ub2 ol_amount_len[NITEMS];
    ub2 ol_delivery_d_len[NITEMS];
    ub2 ol_w_id_len;
    ub2 ol_d_id_len;
    ub2 ol_o_id_len;

    ub2 c_rowid_rcode[100];
    ub2 ol_supply_w_id_rcode[NITEMS];
    ub2 ol_i_id_rcode[NITEMS];
    ub2 ol_quantity_rcode[NITEMS];
    ub2 ol_amount_rcode[NITEMS];
    ub2 ol_delivery_d_rcode[NITEMS];
    ub2 ol_w_id_rcode;

    ub2 ol_d_id_rcode;
    ub2 ol_o_id_rcode;

    ub4 ol_supply_w_id_csize;
    ub4 ol_i_id_csize;
    ub4 ol_quantity_csize;
    ub4 ol_amount_csize;
    ub4 ol_delivery_d_csize;
    ub4 ol_w_id_csize;
    ub4 ol_d_id_csize;
    ub4 ol_o_id_csize;

    OCISmt *curo0;
    OCIBind *w_id_bp0;
    OCIBind *d_id_bp0;
    OCIBind *c_id_bp;
    OCIBind *c_last_bp;
    #ifdef PLSQLORD
    OCIBind *byln_bp;
    OCIBind *c_first_bp;
    OCIBind *c_middle_bp;
    OCIBind *c_balance_bp;
    OCIBind *o_entry_d_bp;
    OCIBind *o_cr_id_bp;
    OCIBind *o_ol_cnt_bp;
    OCIBind *ol_i_id_bp;
    OCIBind *ol_supply_w_id_bp;
    OCIBind *ol_quantity_bp;
    OCIBind *ol_amount_bp;
    OCIBind *ol_d_base_bp;
    ub4 ol_i_id_cnt;
    ub4 ol_sup_cnt;
    ub4 ol_qty_cnt;
    ub4 ol_amt_cnt;
    ub4 ol_del_d_cnt;
    #else
    OCISmt *curo1;
    OCISmt *curo2;
    OCISmt *curo3;
    OCISmt *curo4;
    OCIBind *w_id_bp2;
    OCIBind *w_id_bp3;
    OCIBind *w_id_bp4;
    OCIBind *d_id_bp2;
    OCIBind *d_id_bp3;
    OCIBind *d_id_bp4;
    OCIBind *c_last_bp4;
    OCIBind *o_id_bp;
    OCIBind *c_rowid_bp;
    OCIDefine *c_rowid_dp;
    OCIDefine *c_last_dp;
    OCIDefine *c_last_dp1;
    OCIDefine *c_id_dp;
    OCIDefine *c_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_ol_cnt_dp1;
    OCIDefine *o_ol_cnt_dp2;
    OCIDefine *ol_d_dp;
    OCIDefine *ol_i_id_dp;
    OCIDefine *ol_supply_w_id_dp;
    OCIDefine *ol_quantity_dp;
    OCIDefine *ol_amount_dp;
    OCIDefine *ol_d_base_dp;
    OCIDefine *c_count_dp;
    OCIRowid *c_rowid_ptr[100];
    int cs;
    int cust_idx;
    int norow;
    int rcount;
    int somerows;
    #endif
};

typedef struct ordctx ordctx;

struct defctx
{
    boolean reexec;
    ub4 count;
};

typedef struct defctx defctx;

struct defctx_ordctx {
    defctx *ctxp;
    ordctx *octx;
};

typedef struct defctx_ordctx defctx_ordctx;

/* ordctx *octx; */
```

```

/* defctx cbctx; */

#ifdef PLSQLORD
sb4 rid_data(dvoid *ctxp_octx, OCIDefine *dp, ub4 iter,
             dvoid **bufpp, ub4 **alenp, ub1 *piecep,
             dvoid **indpp, ub2 **rcodepp)
{
    ub4 i;
    defctx *ctxp = ((defctx_ordctx *)ctxp_octx)->ctxp;
    ordctx *octx = ((defctx_ordctx *)ctxp_octx)->octx;
    if (((defctx *)ctxp)->reexec) /* if this is the second execute - use entry 0 */
    {
        i = 0;
        ((defctx *)ctxp)->count--; /* count down */
    }
    else
        i = iter;
    *bufpp = octx->c_rowid_ptr[i];
    *indpp = &octx->c_rowid_ind[i];
    *alenp = &octx->c_rowid_len[i];
    *rcodepp = &octx->c_rowid_rcode[i];
    *piecep = OCI_ONE_PIECE;
    return (OCI_CONTINUE);
}
#endif

tkvcoint(ora_cn_data_t *ora_SlotDataP)
{
    int i;
    text stmbuff[SQL_BUF_SIZE];

    ordctx *octx;
    defctx *cbctx;
    global_order_t *ordP;
    OCIEEnv *tpcenv = ora_SlotDataP->tpcenv;
    OCIServer *tpcsrv = ora_SlotDataP->tpcsrv;
    OCIErr *errhp = ora_SlotDataP->errhp;
    OCISvcCtx *tpscvc = ora_SlotDataP->tpscvc;
    OCISession *tpcusr = ora_SlotDataP->tpcusr;
    OCISmt *curi = ora_SlotDataP->curi;
    defctx_ordctx *ctxp_octx;

    octx = (ordctx *) malloc(sizeof(ordctx));
    memset(octx, (char)0, sizeof(ordctx));
    ora_SlotDataP->octx = (void *)octx; /* */

    cbctx = (defctx *) malloc(sizeof(defctx));
    memset(cbctx, (char)0, sizeof(defctx));
    ora_SlotDataP->cbctx = (void *)cbctx; /* */

    /* allocate the space */
    ctxp_octx = (defctx_ordctx *) malloc(sizeof(defctx_ordctx));
    ora_SlotDataP->ctxp_octx = (void *)ctxp_octx;

    ora_SlotDataP->ordP = (global_order_t *) malloc(sizeof(global_order_t));
    memset(ora_SlotDataP->ordP, (char)0, sizeof(global_order_t));
    ordP = ora_SlotDataP->ordP;

#ifdef PLSQLORD
    octx->cs = 1;
    octx->norow = 0;
    octx->somerows = 10;
    /* get the rowid handles */
    for(i=0; i<100; i++) {
        OCIErr *errhp, OCIDescriptorAlloc(tpcenv, (dvoid **)&octx->c_rowid_ptr[i],
        OCI_DTYPE_ROWID, 0, (dvoid **)0);
    }
#endif

    OCIErr *errhp,
    OCIHandleAlloc(tpcenv, (dvoid **)&octx->uro0, OCI_HTYPE_STMT, 0, (dvoid **)0);
    OCIErr *errhp,
    OCIHandleAlloc(tpcenv, (dvoid **)&octx->uro1, OCI_HTYPE_STMT, 0, (dvoid **)0);
#ifdef PLSQLORD
    OCIErr *errhp,
    OCIHandleAlloc(tpcenv, (dvoid **)&octx->uro2, OCI_HTYPE_STMT, 0, (dvoid **)0);
    OCIErr *errhp,
    OCIHandleAlloc(tpcenv, (dvoid **)&octx->uro3, OCI_HTYPE_STMT, 0, (dvoid **)0);
    OCIErr *errhp,
    OCIHandleAlloc(tpcenv, (dvoid **)&octx->uro4, OCI_HTYPE_STMT, 0, (dvoid **)0);
#endif

#ifdef PLSQLORD
    sprintf((char *) stmbuff, SQLTXT);
    OCIErr *errhp,
    OCISmtPrepare(octx->uro0, errhp, stmbuff, strlen((char *) stmbuff),
    OCI_NTV_SYNTAX, OCI_DEFAULT);
#else
    /* c_id = 0, use find customer by lastname. Get an array or rowid's back */
    sprintf((char *) stmbuff, SQLCUR0);
    OCIErr *errhp,
    OCISmtPrepare(octx->uro0, errhp, stmbuff, strlen((char *) stmbuff),
    OCI_NTV_SYNTAX, OCI_DEFAULT);

```

```

OCIError *errhp,
    OCIAttrSet(octx->uro0, OCI_HTYPE_STMT, (dvoid **)&octx->norow, 0,
    OCI_ATTR_PREFETCH_ROWS, errhp);
/* get order/customer info back based on rowid */
sprintf((char *) stmbuff, SQLCUR1);
OCIError *errhp,
    OCISmtPrepare(octx->uro1, errhp, stmbuff, strlen((char *) stmbuff),
    OCI_NTV_SYNTAX, OCI_DEFAULT);

OCIError *errhp,
    OCIAttrSet(octx->uro1, OCI_HTYPE_STMT, (dvoid **)&octx->norow, 0,
    OCI_ATTR_PREFETCH_ROWS, errhp);

/* c_id == 0, use lastname to find customer */
sprintf((char *) stmbuff, SQLCUR2);
OCIError *errhp,
    OCISmtPrepare(octx->uro2, errhp, stmbuff, strlen((char *) stmbuff),
    OCI_NTV_SYNTAX, OCI_DEFAULT);

OCIError *errhp,
    OCIAttrSet(octx->uro2, OCI_HTYPE_STMT, (dvoid **)&octx->norow, 0,
    OCI_ATTR_PREFETCH_ROWS, errhp);

sprintf((char *) stmbuff, SQLCUR3);
OCIError *errhp,
    OCISmtPrepare(octx->uro3, errhp, stmbuff, strlen((char *) stmbuff),
    OCI_NTV_SYNTAX, OCI_DEFAULT);

OCIError *errhp,
    OCIAttrSet(octx->uro3, OCI_HTYPE_STMT, (dvoid **)&octx->norow, 0,
    OCI_ATTR_PREFETCH_ROWS, errhp);

sprintf((char *) stmbuff, SQLCUR4);
OCIError *errhp,
    OCISmtPrepare(octx->uro4, errhp, stmbuff, strlen((char *) stmbuff),
    OCI_NTV_SYNTAX, OCI_DEFAULT);

OCIError *errhp,
    OCIAttrSet(octx->uro4, OCI_HTYPE_STMT, (dvoid **)&octx->norow, 0,
    OCI_ATTR_PREFETCH_ROWS, errhp);
#endif

for (i = 0; i < NITEMS; i++) {
    octx->o_l_supply_w_id_ind[i] = TRUE;
    octx->o_l_i_id_ind[i] = TRUE;
    octx->o_l_quantity_ind[i] = TRUE;
    octx->o_l_amount_ind[i] = TRUE;
    octx->o_l_delivery_d_ind[i] = TRUE;

    octx->o_l_supply_w_id_len[i] = sizeof(int);
    octx->o_l_i_id_len[i] = sizeof(int);
    octx->o_l_quantity_len[i] = sizeof(int);
    octx->o_l_amount_len[i] = sizeof(int);
    octx->o_l_delivery_d_len[i] = sizeof(ordP->o_l_d_base[0]);
}

octx->o_l_supply_w_id_csiz = NITEMS;
octx->o_l_i_id_csiz = NITEMS;
octx->o_l_quantity_csiz = NITEMS;
octx->o_l_amount_csiz = NITEMS;
octx->o_l_delivery_d_csiz = NITEMS;
octx->o_l_w_id_csiz = NITEMS;
octx->o_l_o_id_csiz = NITEMS;
octx->o_l_d_id_csiz = NITEMS;
octx->o_l_w_id_ind = TRUE;
octx->o_l_d_id_ind = TRUE;
octx->o_l_o_id_ind = TRUE;
octx->o_l_w_id_len = sizeof(int);
octx->o_l_d_id_len = sizeof(int);
octx->o_l_o_id_len = sizeof(int);

/* bind variables */
#ifdef PLSQLORD
    OCIBND(octx->uro0, octx->w_id_bp0, errhp, "w_id", ADR(ordP->w_id),
    SIZ(int), SQLT_INT);
    OCIBND(octx->uro0, octx->d_id_bp0, errhp, "d_id", ADR(ordP->d_id),
    SIZ(int), SQLT_INT);
    OCIBND(octx->uro0, octx->c_id_bp, errhp, "c_id", ADR(ordP->c_id),
    SIZ(c_id), SQLT_INT);
    OCIBND(octx->uro0, octx->byln_bp, errhp, "byln", ADR(ordP->bylastname),
    SIZ(int), SQLT_INT);
    OCIBND(octx->uro0, octx->c_last_bp, errhp, "c_last", ordP->c_last,
    SIZ(ordP->c_last), SQLT_STR);
    OCIBND(octx->uro0, octx->c_first_bp, errhp, "c_first", ordP->c_first,
    SIZ(ordP->c_first), SQLT_STR);
    OCIBND(octx->uro0, octx->c_middle_bp, errhp, "c_middle", ordP->c_middle,
    SIZ(ordP->c_middle), SQLT_STR);
    OCIBND(octx->uro0, octx->c_balance_bp, errhp, "c_balance",
    ADR(ordP->c_balance), SIZ(float), SQLT_FLT);
    OCIBND(octx->uro0, octx->c_id_bp, errhp, "o_id", ADR(ordP->o_id),
    SIZ(int), SQLT_INT);
    OCIBND(octx->uro0, octx->o_entry_d_bp, errhp, "o_entry_d", ordP->o_entry_d,
    SIZ(ordP->o_entry_d), SQLT_STR);
    OCIBND(octx->uro0, octx->o_cr_id_bp, errhp, "o_cr_id", ADR(ordP->o_carrier_id),
    SIZ(int), SQLT_INT);
    OCIBND(octx->uro0, octx->o_o_l_cnt_bp, errhp, "o_o_l_cnt", ADR(ordP->o_o_l_cnt),
    SIZ(int), SQLT_INT);

    OCIBNDRAA(octx->uro0, octx->o_l_i_id_bp, errhp, "o_l_i_id",
    ordP->o_l_i_id, SIZ(int), SQLT_INT,
    octx->o_l_i_id_ind, octx->o_l_i_id_len,
    octx->o_l_i_id_rcode, NITEMS, &octx->o_l_i_id_cnt);
    OCIBNDRAA(octx->uro0, octx->o_l_supply_w_id_bp, errhp, "o_l_s_w_id",

```

```

ordP->ol_supply_w_id,SIZ(int),SQLT_INT,
octx->ol_supply_w_id_ind,octx->ol_supply_w_id_len,
octx->ol_supply_w_id_rcode,NITEMS,&octx->ol_sup_cnt);
OCIBNDRAA(octx->куро0, octx->ol_quantity_bp,errhp,"ol_quantity",
ordP->ol_quantity,SIZ(int),SQLT_INT,
octx->ol_quantity_ind,octx->ol_quantity_len,
octx->ol_quantity_rcode,NITEMS,&octx->ol_qty_cnt);
OCIBNDRAA(octx->куро0,octx->ol_amount_bp,errhp,"ol_amount",ordP->ol_amount,
SIZ(float),SQLT_FLT,octx->ol_amount_ind,
octx->ol_amount_len, octx->ol_amount_rcode,NITEMS,
&octx->ol_amt_cnt);
OCIBNDRAA(octx->куро0,octx->ol_d_base_bp,errhp,"ol_d_d",ordP->ol_d_base,
SIZ(OCIDate),SQLT_ODT,octx->ol_delivery_d_ind,
octx->ol_delivery_d_len, octx->ol_delivery_d_rcode,NITEMS,
&octx->ol_del_d_cnt);
#else
/* c_id (customer id) is not known */
OCIBND(octx->куро0,octx->w_id_bp0,errhp,"w_id",ADR(ordP->w_id),SIZ(int),SQLT_INT);
OCIBND(octx->куро0,octx->d_id_bp0,errhp,"d_id",ADR(ordP->d_id),SIZ(int),SQLT_INT);
OCIBND(octx->куро0,octx->c_last_bp,errhp,"c_last",ordP->c_last,SIZ(ordP->c_last),
SQLT_STR);
ctxp_octx->ctxp = cbctx;
ctxp_octx->octx = octx;
OCIDFNDRY(octx->куро0,octx->c_rowid_dp,errhp,1,octx->c_rowid_ptr,
SIZ(OCIRowid*),SQLT_RDD,octx->c_rowid_ind,(dvoid *)ctxp_octx, rid_data);
OCIBND(octx->куро1,octx->c_rowid_bp,errhp,"cust_rowid",
&octx->c_rowid_ptr[octx->cust_idx],
sizeof(octx->c_rowid_ptr[0]),SQLT_RDD);
OCIDEF(octx->куро1,octx->c_id_dp,errhp,1,ADR(ordP->c_id),SIZ(int),SQLT_INT);
OCIDEF(octx->куро1,octx->c_balance_dp1,errhp,2,ADR(ordP->c_balance),
SIZ(double),SQLT_FLT);
OCIDEF(octx->куро1,octx->c_first_dp1,errhp,3,ordP->c_first,SIZ(ordP->c_first)-1,
SQLT_CHR);
OCIDEF(octx->куро1,octx->c_middle_dp1,errhp,4,ordP->c_middle,
SIZ(ordP->c_middle)-1,SQLT_AFC);
OCIDEF(octx->куро1,octx->c_last_dp1,errhp,5,ordP->c_last,SIZ(ordP->c_last)-1,
SQLT_CHR);
OCIDEF(octx->куро1,octx->o_id_dp1,errhp,6,ADR(ordP->o_id),SIZ(int),SQLT_INT);
OCIDEF(octx->куро1,octx->o_entry_d_dp1,errhp,7,
&ordP->o_entry_d_base,SIZ(OCIDate),SQLT_ODT);
OCIDEF(octx->куро1,octx->o_cr_id_dp1,errhp,8,ADR(ordP->o_carrier_id),
SIZ(int),SQLT_INT);
OCIDEF(octx->куро1,octx->o_ol_cnt_dp1,errhp,9,ADR(ordP->o_ol_cnt),
SIZ(int),SQLT_INT);
/* Bind for third cursor , no-zero customer id */
OCIBND(octx->куро2,octx->w_id_bp2,errhp,"w_id",ADR(ordP->w_id),SIZ(int),SQLT_INT);
OCIBND(octx->куро2,octx->d_id_bp2,errhp,"d_id",ADR(ordP->d_id),SIZ(int),SQLT_INT);
OCIBND(octx->куро2,octx->c_id_bp,errhp,"c_id",ADR(ordP->c_id),SIZ(int),SQLT_INT);
OCIDEF(octx->куро2,octx->c_balance_dp2,errhp,1,ADR(ordP->c_balance),
SIZ(double),SQLT_FLT);
OCIDEF(octx->куро2,octx->c_first_dp2,errhp,2,ordP->c_first,SIZ(ordP->c_first)-1,
SQLT_CHR);
OCIDEF(octx->куро2,octx->c_middle_dp2,errhp,3,ordP->c_middle,
SIZ(ordP->c_middle)-1,SQLT_AFC);
OCIDEF(octx->куро2,octx->c_last_dp2,errhp,4,ordP->c_last,SIZ(ordP->c_last)-1,SQLT_CHR);
OCIDEF(octx->куро2,octx->o_id_dp2,errhp,5,ADR(ordP->o_id),SIZ(int),SQLT_INT);
OCIDEF(octx->куро2,octx->o_entry_d_dp2,errhp,6, &ordP->o_entry_d_base,
SIZ(OCIDate),SQLT_ODT);
OCIDEF(octx->куро2, octx->o_cr_id_dp2,errhp,7,ADR(ordP->o_carrier_id),
SIZ(int),SQLT_INT);
OCIDEF(octx->куро2,octx->o_ol_cnt_dp2,errhp,8,ADR(ordP->o_ol_cnt),
SIZ(int),SQLT_INT);
/* Bind for last cursor */
OCIBND(octx->куро3,octx->w_id_bp3,errhp,"w_id",ADR(ordP->w_id),SIZ(int),SQLT_INT);
OCIBND(octx->куро3,octx->d_id_bp3,errhp,"d_id",ADR(ordP->d_id),SIZ(int),SQLT_INT);
OCIBND(octx->куро3,octx->o_id_bp,errhp,"o_id",ADR(ordP->o_id),SIZ(int),SQLT_INT);
OCIDFNRA(octx->куро3, octx->ol_i_id_dp, errhp, 1, ordP->ol_i_id,SIZ(int),SQLT_INT,
octx->ol_i_id_ind,octx->ol_i_id_len, octx->ol_i_id_rcode);
OCIDFNRA(octx->куро3,octx->ol_supply_w_id_dp,errhp,2, ordP->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->куро3, octx->ol_quantity_dp,errhp,3, ordP->ol_quantity,SIZ(int),
SQLT_INT, octx->ol_quantity_ind,octx->ol_quantity_len,
octx->ol_quantity_rcode);
OCIDFNRA(octx->куро3,octx->ol_amount_dp,errhp,4,ordP->ol_amount, SIZ(int),
SQLT_INT,octx->ol_amount_ind, octx->ol_amount_len,
octx->ol_amount_rcode);
OCIDFNRA(octx->куро3,octx->ol_d_base_dp,errhp,5,ordP->ol_d_base,SIZ(OCIDate),
SQLT_ODT, octx->ol_delivery_d_ind,octx->ol_delivery_d_len,
octx->ol_delivery_d_rcode);
OCIBND(octx->куро4,octx->w_id_bp4,errhp,"w_id",ADR(ordP->w_id),SIZ(int),SQLT_INT);
OCIBND(octx->куро4,octx->d_id_bp4,errhp,"d_id",ADR(ordP->d_id),SIZ(int),SQLT_INT);
OCIBND(octx->куро4,octx->c_last_bp4,errhp,"c_last",ordP->c_last,SIZ(ordP->c_last),
SQLT_STR);
OCIDEF(octx->куро4,octx->c_count_dp,errhp,1,ADR(octx->rcount),SIZ(int),
SQLT_INT);
#endif
return (0);
}
tkvco (ora_cn_data_t *ora_SlotDataP)
{
int i;
int rcount;
/* */
ordctx *octx = (ordctx *)ora_SlotDataP->octx;
defctx *cbctx = (defctx *)ora_SlotDataP->cbctx;
global_order_t *ordP = ora_SlotDataP->ordP;
OCISrv *tpcenv = ora_SlotDataP->tpcenv;
OCISrv *tpcsrv = ora_SlotDataP->tpcsrv;
OCISrv *errhp = ora_SlotDataP->errhp;
OCISvcCtx *tpcsvc = ora_SlotDataP->tpcsvc;
OCISession *tpcusr = ora_SlotDataP->tpcusr;
OCISmt *curi = ora_SlotDataP->curi;
for (i = 0; i < NITEMS; i++) {
octx->ol_supply_w_id_ind[i] = TRUE;
octx->ol_i_id_ind[i] = TRUE;
octx->ol_quantity_ind[i] = TRUE;
octx->ol_amount_ind[i] = TRUE;
octx->ol_delivery_d_ind[i] = TRUE;
octx->ol_supply_w_id_len[i] = sizeof(int);
octx->ol_i_id_len[i] = sizeof(int);
octx->ol_quantity_len[i] = sizeof(int);
octx->ol_amount_len[i] = sizeof(int);
octx->ol_delivery_d_len[i] = sizeof(OCIDate);
}
octx->ol_supply_w_id_csize = NITEMS;
octx->ol_i_id_csize = NITEMS;
octx->ol_quantity_csize = NITEMS;
octx->ol_amount_csize = NITEMS;
octx->ol_delivery_d_csize = NITEMS;
#ifdef PLSQLORD
octx->ol_i_id_cnt = 0;
octx->ol_sup_cnt = 0;
octx->ol_qty_cnt = 0;
octx->ol_amt_cnt = 0;
octx->ol_del_d_cnt = 0;
OCIERROR(errhp,
OCIStmtExecute(tpcsvc,octx->куро0,errhp,1,0,0,OCI_DEFAULT));
#else
retry:
if (ordP->bylastname)
{
cbctx->reexec = FALSE;
ordP->execstatus=OCIStmtExecute(tpcsvc,octx->куро0,errhp,100,0,0,OCI_DEFAULT);
/* will get OCI_NO_DATA if <100 found */
if ((ordP->execstatus != OCI_NO_DATA) && (ordP->execstatus != OCI_SUCCESS))
{
ordP->errcode=OCIERROR(errhp, ordP->execstatus);
if ((ordP->errcode == NOT_SERIALIZABLE)|| (ordP->errcode == RECOVER))
{
OCITransCommit(tpcsvc,errhp,OCI_DEFAULT);
ordP->retries++;
goto retry;
} else {
return -1;
}
}
}
if (ordP->execstatus == OCI_NO_DATA) /* there are no more rows */
{
/* get rowcount, find middle one */
OCIAttrGet(octx->куро0,OCI_HTYPE_STMT,&rcount,NULL,OCI_ATTR_ROW_COUNT,errhp);
if (rcount < 1)
{
userlog("ORDERSTATUS rcount=%d\n",rcount);
return (-1);
}
octx->cust_idx=(rcount+1)/2 ;
}
else
{
/* count the number of rows */
ordP->execstatus=OCIStmtExecute(tpcsvc,octx->куро4,errhp,1,0,0,OCI_DEFAULT);
if ((ordP->execstatus != OCI_NO_DATA) && (ordP->execstatus != OCI_SUCCESS))
{
if ((ordP->errcode == NOT_SERIALIZABLE)|| (ordP->errcode == RECOVER))
{
OCITransCommit(tpcsvc,errhp,OCI_DEFAULT);
ordP->retries++;
goto retry;
} else {
return -1;
}
}
}
if (octx->rcount+1 < 2*10)
octx->cust_idx=(octx->rcount+1)/2 ;
else
/* */
{
cbctx->reexec = TRUE;
cbctx->count = (octx->rcount+1)/2 ;
ordP->execstatus=OCIStmtExecute(tpcsvc,octx->куро0,errhp,cbctx->count,

```

```

0,0,0,OCI_DEFAULT);
/* will get OCI_NO_DATA if <100 found */
if (cbctx->count > 0)
{
    userlog ("did not get all rows ");
    return (-1);
}

if ((ordP->execstatus != OCI_NO_DATA) && (ordP->execstatus != OCI_SUCCESS))
{
    ordP->errcode=OCIERROR(errhp, ordP->execstatus);
    if ((ordP->errcode == NOT_SERIALIZABLE) || (ordP->errcode == RECOVER))
    {
        OCITransCommit(tpcsvc, errhp, OCI_DEFAULT);
        ordP->retries++;
        goto retry;
    } else {
        return -1;
    }
}
octx->cust_idx=0;
}

ordP->execstatus=OCIStmtExecute(tpcsvc, octx->куро1, errhp, 1, 0, 0, OCI_DEFAULT);
if (ordP->execstatus != OCI_SUCCESS)
{
    ordP->errcode=OCIERROR(errhp, ordP->execstatus);
    OCITransCommit(tpcsvc, errhp, OCI_DEFAULT);
    if ((ordP->errcode == NOT_SERIALIZABLE) || (ordP->errcode == RECOVER))
    {
        ordP->retries++;
        goto retry;
    } else {
        return -1;
    }
}
else
{
    ordP->execstatus=OCIStmtExecute(tpcsvc, octx->куро2, errhp, 1, 0, 0, OCI_DEFAULT);
    if (ordP->execstatus != OCI_SUCCESS)
    {
        ordP->errcode=OCIERROR(errhp, ordP->execstatus);
        OCITransCommit(tpcsvc, errhp, OCI_DEFAULT);
        if ((ordP->errcode == NOT_SERIALIZABLE) || (ordP->errcode == RECOVER))
        {
            ordP->retries++;
            goto retry;
        } else
        {
            return -1;
        }
    }
}
octx->o_l_w_id_ind = TRUE;
octx->o_l_d_id_ind = TRUE;
octx->o_l_o_id_ind = TRUE;
octx->o_l_w_id_len = sizeof(int);
octx->o_l_d_id_len = sizeof(int);
octx->o_l_o_id_len = sizeof(int);

ordP->execstatus = OCIStmtExecute(tpcsvc, octx->куро3, errhp, ordP->o_o_l_cnt, 0, 0, 0,
OCI_DEFAULT | OCI_COMMIT_ON_SUCCESS);
if (ordP->execstatus != OCI_SUCCESS)
{
    ordP->errcode=OCIERROR(errhp, ordP->execstatus);
    OCITransCommit(tpcsvc, errhp, OCI_DEFAULT);
    if ((ordP->errcode == NOT_SERIALIZABLE) || (ordP->errcode == RECOVER))
    {
        ordP->retries++;
        goto retry;
    } else
    {
        return -1;
    }
}
}
#endif
/* clean up and convert the delivery dates */
for (i = 0; i < ordP->o_o_l_cnt; i++)
{
    if (octx->o_l_delivery_d_ind[i] == -1) /* null date in field */
        strncpy((char*)ordP->o_l_delivery_d[i], "01-01-1811", 10);
    else
    {
        ordP->o_l_del_len[i]=sizeof(ordP->o_l_delivery_d[i]);
        OCIERROR(errhp, OCIDateToText(errhp, &ordP->o_l_base[i],
(text*)SHORTDATE, strlen(SHORTDATE), (text*)0, 0, &ordP->o_l_del_len[i], ordP->o_l_delivery_d[i]));
    }
}
/*
    cvtdmy(o_l_base[i], o_l_delivery_d[i]);
*/
}
return (0);

```

```

}

void tkvcodone (ora_cn_data_t *ora_SlotDataP)
{
    /* TODO: Should we free the cursor handles?? */

    if (ora_SlotDataP->octx) {
        free(ora_SlotDataP->octx);
        ora_SlotDataP->octx = NULL;
    }
    if (ora_SlotDataP->ordP) {
        free(ora_SlotDataP->ordP);
        ora_SlotDataP->ordP = NULL;
    }
}

```

plpay.c

```

#ifdef RCSID
static char *RCSid =
"$Header: /afs/transarc.com/project/encina/rcs/test/src/benchmarks/tpcc/server/ora8.1_mt/RCS/plpay.c,v
1.3 1999/05/26 16:29:58 wenjian Exp $ Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*****
| Copyright (c) 1995 OracleCorp, 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 <oci.h>
#include "tpcc.h"
#include "plora.h" /* */
#include "tpccflags.h"

#define SQLTXT_INIT "BEGIN initpay.pay_init; END;"
#define SQLTXT_STP "begin payment.dopayment(:w_id,:d_id,:c_w_id,:c_d_id, \
:c_id,:by_lname,:h_amount,:c_last,:w_street_1,:w_street_2, \
:w_city,:w_state,:w_zip,:d_street_1,:d_street_2,:d_city, \
:d_state,:d_zip,:c_first,:c_middle,:c_street_1, \
:c_street_2,:c_city,:c_state,:c_zip,:c_phone,:c_since, \
:c_credit,:c_credit_lim,:c_discount,:c_balance,:c_data, \
:cr_date,:retry); end;"

struct payctx {
    OCIStmt *curpi;
    OCIStmt *curp0;
    OCIStmt *curp1;
    OCIBind *w_id_bp;
    OCIBind *w_id_bp1;
    sb2 w_id_ind;
    ub2 w_id_len;
    ub2 w_id_rc;

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

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

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

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

    OCIBind *by_lname_bp;

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

    OCIBind *c_last_bp;

```

<pre> 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; </pre>	<pre> ub2 c_state_rc; OCIBind *c_zip_bp; OCIBind *c_zip_bp1; sb2 c_zip_ind; ub2 c_zip_len; ub2 c_zip_rc; OCIBind *c_phone_bp; OCIBind *c_phone_bp1; sb2 c_phone_ind; ub2 c_phone_len; ub2 c_phone_rc; OCIBind *c_since_bp; OCIBind *c_since_bp1; sb2 c_since_ind; ub2 c_since_len; ub2 c_since_rc; OCIBind *c_credit_bp; OCIBind *c_credit_bp1; sb2 c_credit_ind; ub2 c_credit_len; ub2 c_credit_rc; OCIBind *c_credit_lim_bp; OCIBind *c_credit_lim_bp1; sb2 c_credit_lim_ind; ub2 c_credit_lim_len; ub2 c_credit_lim_rc; OCIBind *c_discount_bp; OCIBind *c_discount_bp1; sb2 c_discount_ind; ub2 c_discount_len; ub2 c_discount_rc; OCIBind *c_balance_bp; OCIBind *c_balance_bp1; sb2 c_balance_ind; ub2 c_balance_len; ub2 c_balance_rc; OCIBind *c_data_bp; OCIBind *c_data_bp1; sb2 c_data_ind; ub2 c_data_len; ub2 c_data_rc; OCIBind *h_date_bp; OCIBind *h_date_bp1; sb2 h_date_ind; ub2 h_date_len; ub2 h_date_rc; OCIBind *retries_bp; OCIBind *retries_bp1; sb2 retries_ind; ub2 retries_len; ub2 retries_rc; OCIBind *cr_date_bp; OCIBind *cr_date_bp1; sb2 cr_date_ind; ub2 cr_date_len; ub2 cr_date_rc; OCIBind *byln_bp; sb2 byln_ind; ub2 byln_len; ub2 byln_rc; }; typedef struct payctx payctx; /* payctx *pctx; */ tkvcpinit (ora_cn_data_t *ora_SlotDataP) { /* */ char *ora_home = getenv("ORACLE_HOME"); char sql_file_name[256]; payctx *pctx; global_payment_t *payP; OCIEnv *tpcenv = ora_SlotDataP->tpcenv; OCIServer *tpcsrv = ora_SlotDataP->tpcsrv; OCLError *errhp = ora_SlotDataP->errhp; OCISvcCtx *tpcsvc = ora_SlotDataP->tpcsvc; OCISession *tpcusr = ora_SlotDataP->tpcusr; OCISmtt *curi = ora_SlotDataP->curi; text stmbuf[SQL_BUF_SIZE]; if (!ora_home) { </pre>
--	--

<pre> err_print("Cannot find env variable ORACLE_HOME\n"); exit(13); } pctx = (payctx *)malloc(sizeof(payctx)); memset(pctx,(char)0,sizeof(payctx)); ora_SlotDataP->pctx = (void *)pctx; ora_SlotDataP->payP = (global_payment_t *)malloc(sizeof(global_payment_t)); memset(ora_SlotDataP->payP,(char)0,sizeof(global_payment_t)); payP = ora_SlotDataP->payP; /* cursor for init */ OCIERROR(errhp,OCIHandleAlloc(tpcenv, (dvoid **)&(pctx->curp1)), 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->curp1, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT)); OCIERROR(errhp, OCIStmtExecute(tpcenv,pctx->curp1,errhp,1,0,0,OCI_DEFAULT)); #endif PLSQLPAY /* prepare the stub for calling psql stored procedure */ sprintf((char*)stmbuf, SQLTXT_STP); OCIERROR(errhp,OCIStmtPrepare(pctx->curp0, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT)); #else /* customer id != 0, go by last name */ sqlfile("paynz.sql",stmbuf); OCIERROR(errhp,OCIStmtPrepare(pctx->curp0, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT)); /* customer id == 0, go by last name */ sqlfile("payz.sql",stmbuf);/* sqlfile opens \$O/bench/.../blocks/... */ OCIERROR(errhp,OCIStmtPrepare(pctx->curp1, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT)); #endif pctx->w_id_ind = TRUE; pctx->w_id_len = SIZ(payP->w_id); pctx->d_id_ind = TRUE; pctx->d_id_len = SIZ(payP->d_id); pctx->c_w_id_ind = TRUE; pctx->c_w_id_len = SIZ(payP->c_w_id); pctx->c_d_id_ind = TRUE; pctx->c_d_id_len = SIZ(payP->c_d_id); pctx->c_id_ind = TRUE; pctx->c_id_len = 0; pctx->h_amount_len = SIZ(payP->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; </pre>	<pre> 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(payP->w_id),SIZ(int), SQLT_INT,&pctx->w_id_ind, NULL, NULL); OCIBNDR(pctx->curp0, pctx->d_id_bp, errhp, "d_id",ADR(payP->d_id),SIZ(int), SQLT_INT,&pctx->d_id_ind, NULL, NULL); OCIBNDR(pctx->curp0, pctx->c_w_id_bp, errhp, "c_w_id",ADR(payP->c_w_id),SIZ(int), SQLT_INT); OCIBNDR(pctx->curp0, pctx->c_d_id_bp, errhp, "c_d_id",ADR(payP->c_d_id),SIZ(int), SQLT_INT); OCIBNDR(pctx->curp0, pctx->c_id_bp, errhp, "c_id",ADR(payP->c_id),SIZ(int), SQLT_INT); #endif PLSQLPAY OCIBNDR(pctx->curp0, pctx->by_lname_bp, errhp, "by_lname",ADR(payP->bylastname), SIZ(int),SQLT_INT); #endif OCIBNDR(pctx->curp0, pctx->h_amount_bp, errhp, "h_amount",ADR(payP->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",payP->c_last,SIZ(payP->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",payP->w_street_1, SIZ(payP->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",payP->w_street_2, SIZ(payP->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",payP->w_city,SIZ(payP->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",payP->w_state,SIZ(payP->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",payP->w_zip,SIZ(payP->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",payP->d_street_1, SIZ(payP->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",payP->d_street_2, SIZ(payP->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",payP->d_city,SIZ(payP->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",payP->d_state,SIZ(payP->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",payP->d_zip,SIZ(payP->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",payP->c_first,SIZ(payP->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",payP->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",payP->c_street_1, SIZ(payP->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",payP->c_street_2, SIZ(payP->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",payP->c_city,SIZ(payP->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",payP->c_state,SIZ(payP->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",payP->c_zip,SIZ(payP->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",payP->c_phone,SIZ(payP->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", &payP->c_since, SIZ(OCIDate),SQLT_ODT, &pctx->c_since_ind, &pctx->c_since_len, &pctx->c_since_rc); OCIBNDR(pctx->curp0, pctx->c_credit_bp, errhp, "c_credit",payP->c_credit, SIZ(payP->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(payP->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(payP->c_discount),SIZ(float),SQLT_FLT, &pctx->c_discount_ind, &pctx->c_discount_len, &pctx->c_discount_rc); OCIBNDR(pctx->curp0, pctx->c_balance_bp, errhp, "c_balance",ADR(payP->c_balance), SIZ(double),SQLT_FLT,&pctx->c_balance_ind, &pctx->c_balance_len, </pre>
---	---

<pre> &pctx->c_balance_rc); OCIBNDR(pctx->curp0, pctx->c_data_bp, errhp, "c_data", payP->c_data, SIZ(payP->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", payP->h_date, SIZ(payP->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(payP->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(payP->cr_date), SIZ(OCIDate), SQLT_ODT, &pctx->cr_date_ind, &pctx->cr_date_len, &pctx->cr_date_rc); #endifdef PLSQLPAY /* ---- Binds for the second cursor */ OCIBNDR(pctx->curp1, pctx->w_id_bp1, errhp, "w_id", ADR(payP->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(payP->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(payP->c_w_id), SIZ(int), SQLT_INT); OCIBND(pctx->curp1, pctx->c_d_id_bp1, errhp, "c_d_id", ADR(payP->c_d_id), SIZ(int), SQLT_INT); OCIBNDR(pctx->curp1, pctx->c_id_bp1, errhp, "c_id", ADR(payP->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(payP->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", payP->c_last, SIZ(payP->c_last), SQLT_STR); OCIBNDR(pctx->curp1, pctx->w_street_1_bp1, errhp, "w_street_1", payP->w_street_1, SIZ(payP->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", payP->w_street_2, SIZ(payP->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", payP->w_city, SIZ(payP->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", payP->w_state, SIZ(payP->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", payP->w_zip, SIZ(payP->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", payP->d_street_1, SIZ(payP->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", payP->d_street_2, SIZ(payP->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", payP->d_city, SIZ(payP->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", payP->d_state, SIZ(payP->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", payP->d_zip, SIZ(payP->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", payP->c_first, SIZ(payP->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", payP->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", payP->c_street_1, SIZ(payP->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", payP->c_street_2, SIZ(payP->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", payP->c_city, SIZ(payP->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", payP->c_state, SIZ(payP->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", payP->c_zip, SIZ(payP->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", payP->c_phone, SIZ(payP->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", &payP->c_since, SIZ(OCIDate), SQLT_ODT, &pctx->c_since_ind, &pctx->c_since_len, &pctx->c_since_rc); OCIBNDR(pctx->curp1, pctx->c_credit_bp1, errhp, "c_credit", payP->c_credit, SIZ(payP->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(payP->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(payP->c_discount), SIZ(int), SQLT_FLT, &pctx->c_discount_ind, &pctx->c_discount_len, &pctx->c_discount_rc); OCIBNDR(pctx->curp1, pctx->c_balance_bp1, errhp, "c_balance", ADR(payP->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", payP->c_data, SIZ(payP->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", payP->h_date, SIZ(payP->h_date), SQLT_STR, &pctx->h_date_ind, &pctx->h_date_len, &pctx->h_date_rc); */ </pre>	<pre> OCIBNDR(pctx->curp1, pctx->retries_bp1, errhp, "retry", ADR(payP->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(payP->cr_date), SIZ(OCIDate), SQLT_ODT, &pctx->cr_date_ind, &pctx->cr_date_len, &pctx->cr_date_rc); #endif return (0); } tkvcv (ora_cn_data_t *ora_SlotDataP) { /* */ payctx *pctx = ora_SlotDataP->pctx; global_payment_t *payP = ora_SlotDataP->payP; OCIEnv *tpcenv = ora_SlotDataP->tpcenv; OCIServer *tpcsrv = ora_SlotDataP->tpcsrv; OCISvcCtx *tpscvc = ora_SlotDataP->tpscvc; OCISession *tpcusr = ora_SlotDataP->tpcusr; OCISmt *curi = ora_SlotDataP->curi; retry: pctx->w_id_ind = TRUE; pctx->w_id_len = SIZ(payP->w_id); pctx->d_id_ind = TRUE; pctx->d_id_len = SIZ(payP->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(payP->h_amount); pctx->h_amount_ind = TRUE; pctx->c_last_ind = TRUE; pctx->c_last_len = SIZ(payP->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->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; #ifndef PLSQLPAY </pre>
--	---

```

payP->execstatus=OCISmtExecute(tpcsvc,ptcx->curp0,errhp,1,0,0,0,OCI_DEFAULT|OCI_COMMIT_ON
_SUCCESS);
#else
if(payP->bylastname) {
payP->execstatus=OCISmtExecute(tpcsvc,ptcx->curp1,errhp,1,0,0,0,OCI_DEFAULT|OCI_COMMIT_ON
_SUCCESS);
} else {
payP->execstatus=OCISmtExecute(tpcsvc,ptcx->curp0,errhp,1,0,0,0,OCI_DEFAULT|OCI_COMMIT_ON
_SUCCESS);
}
#endif

if(payP->execstatus != OCI_SUCCESS) {
OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);
payP->errcode = OCIERROR(errhp,payP->execstatus);
if(payP->errcode == NOT_SERIALIZABLE) {
payP->retries++;
goto retry;
} else if(payP->errcode == RECOVER) {
payP->retries++;
goto retry;
} else {
return -1;
}
}
return 0;
}

void tkvcpdone (ora_cn_data_t *ora_SlotDataP)
{
/* TODO: Should we free the cursor handles?? */
if(ora_SlotDataP->ptcx) {
free(ora_SlotDataP->ptcx);
ora_SlotDataP->ptcx = NULL;
}
if (ora_SlotDataP->payP) {
free(ora_SlotDataP->payP);
ora_SlotDataP->payP = NULL;
}
}

plsto.c

#ifndef RCSID
static char *RCSid =
"$Header: /afs/transarc.com/project/encina/rcs/test/src/benchmarks/tpcc/server/ora8.1_mt/RCS/plsto.c,v
1.2 1999/04/15 12:16:52 oZExp S Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=====+
| Copyright (c) 1994 OracleCorp, 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 "plora.h" /* */
#include "tpccflags.h"

#ifndef PLSQLSTO
#define SQLTXT "BEGIN stocklevel.getstocklevel(:w_id,:d_id,:threshold,\
:low_stock);END;"
#else
#define SQLTXT "SELECT /*+ nocache(stock) hash(stock) no_index(stock) */ \
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)"
/* query using functional index */
/*
#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_o_id BETWEEN (d_next_o_id - 20) AND (d_next_o_id - 1) AND \
decode(SIGN(s_quantity - 21), -1, s_w_id*100000 + s_i_id, NULL) \
=ol_w_id*100000 + ol_i_id AND \
s_quantity < :threshold;"
*/
#endif

struct stoctx {

```

```

OCISmt *curs;
OCIBind *w_id_bp;
OCIBind *d_id_bp;
OCIBind *threshold_bp;
#ifdef PLSQLSTO
OCIBind *low_stock_bp;
#else
OCIDefine *low_stock_bp;
#endif
int norow;
};

typedef struct stoctx stoctx;

/* stoctx *sctx; */

tkvcscinit (ora_cn_data_t *ora_SlotDataP)
{
/* */
stoctx *sctx;
global_stock_t *stoP;
OCIEnv *tpcenv = ora_SlotDataP->tpcenv;
OCIServer *tpcsrv = ora_SlotDataP->tpcsrv;
OCIError *errhp = ora_SlotDataP->errhp;
OCISvcCtx *tpcsvc = ora_SlotDataP->tpcsvc;
OCISession *tpcusr = ora_SlotDataP->tpcusr;
OCISmt *curi = ora_SlotDataP->curi;

text stmbuf[SQL_BUF_SIZE];

sctx = (stoctx *)malloc(sizeof(stoctx));
memset(sctx,(char)0,sizeof(stoctx));
ora_SlotDataP->sctx = (void *)sctx;

ora_SlotDataP->stoP = (global_stock_t *)malloc(sizeof(global_stock_t));
memset(ora_SlotDataP->stoP,(char)0,sizeof(global_stock_t));
stoP = ora_SlotDataP->stoP;

sctx->norow=0;

OCIERROR(errhp,
OCIHandleAlloc(tpcenv,(dvoid*)&sctx->curs,OCI_HTYPE_STMT,0,(dvoid**)0));
sprintf ((char *) stmbuf, SQLTXT);
OCIERROR(errhp,OCISmtPrepare(sctx->curs,errhp,stmbuf,strlen((char *)stmbuf),
OCI_NTV_SYNTAX,OCI_DEFAULT));

#ifdef PLSQLSTO
OCIERROR(errhp,
OCIAttrSet(sctx->curs,OCI_HTYPE_STMT,(dvoid*)&sctx->norow,0,
OCI_ATTR_PREFETCH_ROWS,errhp));
#endif

/* bind variables */

OCIBND(sctx->curs,sctx->w_id_bp,errhp, ":w_id", ADR(stoP->w_id),sizeof(int),
SQLT_INT);
OCIBND(sctx->curs,sctx->d_id_bp,errhp, ":d_id", ADR(stoP->d_id),sizeof(int),
SQLT_INT);
OCIBND(sctx->curs,sctx->threshold_bp,errhp, ":threshold", ADR(stoP->threshold),
sizeof(int),SQLT_INT);
#ifdef PLSQLSTO
OCIBND(sctx->curs,sctx->low_stock_bp,errhp,":low_stock", ADR(stoP->low_stock),
sizeof(int), SQLT_INT);
#else
OCIDefine(sctx->curs,sctx->low_stock_bp,errhp, 1, ADR(stoP->low_stock),
sizeof(int), SQLT_INT);
#endif
}

return (0);
}

tkvcs (ora_cn_data_t *ora_SlotDataP)
{
stoctx *sctx = (stoctx *)ora_SlotDataP->sctx;
global_stock_t *stoP = ora_SlotDataP->stoP;
OCIEnv *tpcenv = ora_SlotDataP->tpcenv;
OCIServer *tpcsrv = ora_SlotDataP->tpcsrv;
OCIError *errhp = ora_SlotDataP->errhp;
OCISvcCtx *tpcsvc = ora_SlotDataP->tpcsvc;
OCISession *tpcusr = ora_SlotDataP->tpcusr;
OCISmt *curi = ora_SlotDataP->curi;

retry:
stoP->execstatus= OCISmtExecute(tpcsvc,sctx->curs,errhp,1,0,0,0,
OCI_COMMIT_ON_SUCCESS | OCI_DEFAULT);
if (stoP->execstatus != OCI_SUCCESS)
{
stoP->errcode=OCIERROR(errhp,stoP->execstatus);
OCITransCommit(tpcsvc,errhp,OCI_DEFAULT);
if((stoP->errcode == NOT_SERIALIZABLE)|| (stoP->errcode == RECOVER))
{
stoP->retries++;
goto retry;
}
}
}

```

```

} else {
    return -1;
}
}

return (0);
}

void tkvcsdone (ora_cn_data_t *ora_SlotDataP)
{
    /* */
    stoctx *sctx = (stoctx *)ora_SlotDataP->sctx;
    if (sctx) {
        free(sctx);
        ora_SlotDataP->sctx = NULL;
    }
    if (ora_SlotDataP->stoP) {
        free(ora_SlotDataP->stoP);
        ora_SlotDataP->stoP = NULL;
    }
}

-- The initnew package for storing variables used in the
-- New Order anonymous block

CREATE OR REPLACE PACKAGE initnew
AS
TYPE intarray IS TABLE OF INTEGER index by binary_integer;
TYPE distarray IS TABLE OF VARCHAR(24) index by binary_integer;
nulldate DATE;
s_dist distarray;
idx1arr intarray;
s_remote intarray;
PROCEDURE new_init(idxarr intarray);
END initnew;
/
show errors;

CREATE OR REPLACE PACKAGE BODY initnew AS
PROCEDURE new_init (idxarr intarray)
IS
BEGIN
    -- initialize null date
    nulldate := TO_DATE('01-01-1811','MM-DD-YYYY');
    idx1arr := idxarr;
    END new_init;
END initnew;
/
show errors
exit

tkvcpnew.sql

-- New Order Anonymous block

DECLARE
    idx BINARY_INTEGER;
    dummy_local BINARY_INTEGER;
    not_serializable EXCEPTION;
    PRAGMA EXCEPTION_INIT(not_serializable,-8177);
    deadlock EXCEPTION;
    PRAGMA EXCEPTION_INIT(deadlock,-60);
    snapshot_too_old EXCEPTION;
    PRAGMA EXCEPTION_INIT(snapshot_too_old,-1555);
PROCEDURE u1 IS
BEGIN
    FORALL idx IN 1 ..:o_ol_cnt
        UPDATEstock_item
        SETs_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),
        s_quantity = s_quantity - :ol_quantity(idx) +
            DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
        WHEREi_id = :ol_i_id(idx)
        ANDs_w_id = :ol_supply_w_id(idx)
        RETURNINGi_price, i_name, s_quantity, s_dist_01,
            DECODE (instr(i_data,'original'),0, 'G',
            DECODE(instr(s_data,'original'),0, 'G', 'B'))
            BULK COLLECT INTO:i_price, :i_name, :s_quantity, initnew.s_dist,
:brand_generic;
END u1;

PROCEDURE u2 IS
BEGIN
    FORALL idx IN 1 ..:o_ol_cnt
        UPDATEstock_item
        SETs_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),

```

tkvcin.sql

```

s_quantity = s_quantity - :ol_quantity(idx) +
DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
WHEREi_id = :ol_i_id(idx)
ANDs_w_id = :ol_supply_w_id(idx)
RETURNINGi_price, i_name, s_quantity, s_dist_02,
    DECODE (instr(i_data,'original'),0, 'G',
    DECODE(instr(s_data,'original'),0, 'G', 'B'))
    BULK COLLECT INTO:i_price, :i_name, :s_quantity, initnew.s_dist,
:brand_generic;
END u2;

PROCEDURE u3 IS
BEGIN
    FORALL idx IN 1 ..:o_ol_cnt
        UPDATEstock_item
        SETs_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),
        s_quantity = s_quantity - :ol_quantity(idx) +
            DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
        WHEREi_id = :ol_i_id(idx)
        ANDs_w_id = :ol_supply_w_id(idx)
        RETURNINGi_price, i_name, s_quantity, s_dist_03,
            DECODE (instr(i_data,'original'),0, 'G',
            DECODE(instr(s_data,'original'),0, 'G', 'B'))
            BULK COLLECT INTO:i_price, :i_name, :s_quantity, initnew.s_dist,
:brand_generic;
END u3;

PROCEDURE u4 IS
BEGIN
    FORALL idx IN 1 ..:o_ol_cnt
        UPDATEstock_item
        SETs_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),
        s_quantity = s_quantity - :ol_quantity(idx) +
            DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
        WHEREi_id = :ol_i_id(idx)
        ANDs_w_id = :ol_supply_w_id(idx)
        RETURNINGi_price, i_name, s_quantity, s_dist_04,
            DECODE (instr(i_data,'original'),0, 'G',
            DECODE(instr(s_data,'original'),0, 'G', 'B'))
            BULK COLLECT INTO:i_price, :i_name, :s_quantity, initnew.s_dist,
:brand_generic;
END u4;

PROCEDURE u5 IS
BEGIN
    FORALL idx IN 1 ..:o_ol_cnt
        UPDATEstock_item
        SETs_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),
        s_quantity = s_quantity - :ol_quantity(idx) +
            DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
        WHEREi_id = :ol_i_id(idx)
        ANDs_w_id = :ol_supply_w_id(idx)
        RETURNINGi_price, i_name, s_quantity, s_dist_05,
            DECODE (instr(i_data,'original'),0, 'G',
            DECODE(instr(s_data,'original'),0, 'G', 'B'))
            BULK COLLECT INTO:i_price, :i_name, :s_quantity, initnew.s_dist,
:brand_generic;
END u5;

PROCEDURE u6 IS
BEGIN
    FORALL idx IN 1 ..:o_ol_cnt
        UPDATEstock_item
        SETs_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),
        s_quantity = s_quantity - :ol_quantity(idx) +
            DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
        WHEREi_id = :ol_i_id(idx)
        ANDs_w_id = :ol_supply_w_id(idx)
        RETURNINGi_price, i_name, s_quantity, s_dist_06,
            DECODE (instr(i_data,'original'),0, 'G',
            DECODE(instr(s_data,'original'),0, 'G', 'B'))
            BULK COLLECT INTO:i_price, :i_name, :s_quantity, initnew.s_dist,
:brand_generic;
END u6;

PROCEDURE u7 IS
BEGIN
    FORALL idx IN 1 ..:o_ol_cnt
        UPDATEstock_item
        SETs_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),
        s_quantity = s_quantity - :ol_quantity(idx) +
            DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
        WHEREi_id = :ol_i_id(idx)

```

```

AND s_w_id = :ol_supply_w_id(idx)
RETURNING i_price, i_name, s_quantity, s_dist_07,
        DECODE(instr(i_data,'original'),0,'G',
        DECODE(instr(s_data,'original'),0,'G','B'))
        BULK COLLECT INTO :i_price, :i_name, :s_quantity, initnew.s_dist,
        :brand_generic;
END u7;

PROCEDURE u8 IS
BEGIN
FORALL idx IN 1 .. :o_ol_cnt
    UPDATE stock_item
    SET s_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),
        s_quantity = s_quantity - :ol_quantity(idx) +
            DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
    WHERE i_id = :ol_i_id(idx)
    AND s_w_id = :ol_supply_w_id(idx)
    RETURNING i_price, i_name, s_quantity, s_dist_08,
            DECODE(instr(i_data,'original'),0,'G',
            DECODE(instr(s_data,'original'),0,'G','B'))
            BULK COLLECT INTO :i_price, :i_name, :s_quantity, initnew.s_dist,
            :brand_generic;
END u8;

PROCEDURE u9 IS
BEGIN
FORALL idx IN 1 .. :o_ol_cnt
    UPDATE stock_item
    SET s_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),
        s_quantity = s_quantity - :ol_quantity(idx) +
            DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
    WHERE i_id = :ol_i_id(idx)
    AND s_w_id = :ol_supply_w_id(idx)
    RETURNING i_price, i_name, s_quantity, s_dist_09,
            DECODE(instr(i_data,'original'),0,'G',
            DECODE(instr(s_data,'original'),0,'G','B'))
            BULK COLLECT INTO :i_price, :i_name, :s_quantity, initnew.s_dist,
            :brand_generic;
END u9;

PROCEDURE u10 IS
BEGIN
FORALL idx IN 1 .. :o_ol_cnt
    UPDATE stock_item
    SET s_order_cnt = s_order_cnt + 1,
        s_ytd = s_ytd + :ol_quantity(idx),
        s_remote_cnt = s_remote_cnt + :s_remote(idx),
        s_quantity = s_quantity - :ol_quantity(idx) +
            DECODE(sign(s_quantity - :ol_quantity(idx) -
10),-1,91,0)
    WHERE i_id = :ol_i_id(idx)
    AND s_w_id = :ol_supply_w_id(idx)
    RETURNING i_price, i_name, s_quantity, s_dist_10,
            DECODE(instr(i_data,'original'),0,'G',
            DECODE(instr(s_data,'original'),0,'G','B'))
            BULK COLLECT INTO :i_price, :i_name, :s_quantity, initnew.s_dist,
            :brand_generic;
END u10;

PROCEDURE fix_items IS
rows_lost      BINARY_INTEGER;
max_index      BINARY_INTEGER;
temp_index     BINARY_INTEGER;
BEGIN
-- gotta shift price, name, s_quantity, brand_generic, s_dist, ol_amount
idx := 1;
-- found 0 bad rows
rows_lost := 0;
-- so many rows in out array to begin with
max_index := sql%rowcount;

WHILE (max_index != :o_ol_cnt) LOOP

-- find item where item ids dont match
WHILE (idx <= sql%rowcount AND
sql%bulk_rowcount(idx + rows_lost) = 1)
    LOOP
        idx := idx + 1;
    END LOOP;

-- shift the items please
temp_index := max_index;
WHILE (temp_index >= idx + rows_lost) LOOP
    :i_price(temp_index + 1) := :i_price(temp_index);
    :i_name(temp_index + 1) := :i_name(temp_index);
    :s_quantity(temp_index + 1) := :s_quantity(temp_index);
    initnew.s_dist(temp_index + 1) := initnew.s_dist(temp_index);
    :brand_generic(temp_index + 1) := :brand_generic(temp_index);
    temp_index := temp_index - 1;
END LOOP;

-- values for the non-existent items if not at end
IF (idx + rows_lost <= :o_ol_cnt) THEN
        :i_price(idx + rows_lost) := 0;
        :i_name(idx + rows_lost) := NULL;
        :s_quantity(idx + rows_lost) := 0;
        initnew.s_dist(idx + rows_lost) := NULL;
        :brand_generic(idx + rows_lost) := NULL;

-- one more bad row
rows_lost := rows_lost + 1;
max_index := max_index + 1;
END IF;

END LOOP;
END fix_items;

BEGIN
LOOP BEGIN
UPDATE district SET d_next_o_id = d_next_o_id + 1
WHERE d_id = :d_id AND d_w_id = :w_id
RETURNING d_tax, d_next_o_id-1
INTO :d_tax, :o_id;

SELECT c_discount, c_last, c_credit, w_tax
INTO :c_discount, :c_last, :c_credit, :w_tax
FROM customer, warehouse
WHERE c_id = :c_id AND c_d_id = :d_id AND c_w_id = :w_id
AND w_id = :w_id;

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_d_id, o_w_id, o_c_id, o_entry_d,
o_carrier_id, o_ol_cnt, o_all_local)
VALUES (:o_id, :d_id, :w_id, :c_id,
:cr_date, 11, :o_ol_cnt, :o_all_local);

-- copying :d_id in local variable .
dummy_local := :d_id;

IF (dummy_local = 1) THEN u1; END IF;
IF (dummy_local = 2) THEN u2; END IF;
IF (dummy_local = 3) THEN u3; END IF;
IF (dummy_local = 4) THEN u4; END IF;
IF (dummy_local = 5) THEN u5; END IF;
IF (dummy_local = 6) THEN u6; END IF;
IF (dummy_local = 7) THEN u7; END IF;
IF (dummy_local = 8) THEN u8; END IF;
IF (dummy_local = 9) THEN u9; END IF;
IF (dummy_local = 10) THEN u10; END IF;

-- cache the no of rows processed
dummy_local := sql%rowcount;

-- fix the rows if necessary
IF (dummy_local != :o_ol_cnt) THEN fix_items; END IF;

-- calculate ol_amount

FOR idx IN 1 .. :o_ol_cnt LOOP
:ol_amount(idx) := :ol_quantity(idx) * :i_price(idx);
END LOOP;

FORALL idx IN 1 .. :o_ol_cnt
-- doesnt hurt if we insert entries for invalid item too
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 (:o_id, :d_id, :w_id, initnew.idx1arr(idx), initnew.nulldate,
:ol_i_id(idx), :ol_supply_w_id(idx),
:ol_quantity(idx), :ol_amount(idx), initnew.s_dist(idx));

--If there are no errors, then just return without COMMITTING
--The COMMIT is done on the driver side by OCI
-- If there are errors, then rollback and set o_ol_cnt to the processed value
IF (dummy_local != :o_ol_cnt) THEN
:o_ol_cnt := dummy_local;
ROLLBACK;
END IF;

EXIT;

EXCEPTION
WHEN not_serializable OR deadlock OR snapshot_too_old THEN
ROLLBACK;
:retry := :retry + 1;
END;
END LOOP;
END;

```

<pre> /* * \$Header: /afs/transarc.com/project/encina/rcs/test/src/benchmarks/tpcc/server/ora8.1_mt/RCS/tpcc.h,v 1.1 1999/04/14 19:03:06wenjian Exp S Copyr (c) 1993 Oracle */ =====+ Copyright (c) 1995 OracleCorp, Redwood Shores, CA OPEN SYSTEMS PERFORMANCE GROUP All Rights Reserved =====+ FILENAME tpcc.h DESCRIPTION Include file for TPC-C benchmark programs. =====+ */ #ifndef TPCC_H #define TPCC_H #ifndef FALSE # define FALSE 0 #endif #ifndef TRUE # define TRUE 1 #endif #include <stdio.h> #include <stdlib.h> #include <ctype.h> #include <string.h> #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 #define FULLDATE "dd-mon-yy.hh:mi:ss" #define SHORTDATE "dd-mm-yyyy" #define DELRT 80.0 extern int tkvcninit (); extern int tkvcnpnit (); extern int tkvcoint (); extern int tkvcdinit (); extern int tkvcsinit (); extern int tkvcn (); extern int tkvcp (); extern int tkvco (); extern int tkvcd (); extern int tkvcs (); extern void tkvcndone (); extern void tkvcpdone (); extern void tkvcodone (); extern void tkvcddone (); </pre>	<pre> extern void tkvcsdone (); extern int tkvcss (); /* for alter session to get memory size and trace */ extern boolean multitrans; extern int ord_init; extern errprt (); extern int ocierror(char *fname, int lineno, OCIError *errhp, sword status); extern int sqlfile(char *fname, text *linebuf); extern FILE *fip; extern FILE *fopen (); extern int proc_no; extern int doid[]; #if 0 extern int execstatus; extern int errcode; extern OCIEnv *tpcenv; extern OCIServer *tpcsrv; extern OCIError *errhp; extern OCISvcCtx *tpcsvc; extern OCISession *tpcsur; extern OCISStmt *curmtest; /* The bind and define handles for each transaction are included in their respective header files.*/ /* for stock-level transaction */ extern int w_id; extern int d_id; extern int c_id; extern int threshold; extern int low_stock; /* for delivery transaction */ extern int del_o_id[10]; extern int carrier_id; extern int retries; /* for order-status transaction */ extern int bylastname; extern char c_last[17]; extern char c_first[17]; extern char c_middle[3]; extern double c_balance; extern int o_id; extern text o_entry_d[20]; extern int o_carrier_id; extern int o_ol_cnt; extern int ol_supply_w_id[15]; extern int ol_i_id[15]; extern int ol_quantity[15]; extern int ol_amount[15]; extern ub4 ol_del_len[15]; extern text ol_delivery_d[15][11]; /* for payment transaction */ extern int c_w_id; extern int c_d_id; extern int h_amount; extern char w_street_1[21]; extern char w_street_2[21]; extern char w_city[21]; extern char w_state[3]; extern char w_zip[10]; extern char d_street_1[21]; extern char d_street_2[21]; extern char d_city[21]; extern char d_state[3]; extern char d_zip[10]; extern char c_street_1[21]; extern char c_street_2[21]; extern char c_city[21]; extern char c_state[3]; extern char c_zip[10]; extern char c_phone[17]; extern text c_since_d[11]; extern char c_credit[3]; extern int c_credit_lim; extern float c_discount; extern char c_data[201]; extern text h_date[20]; /* for new order transaction */ extern int nol_i_id[15]; extern int nol_supply_w_id[15]; extern int nol_quantity[15]; extern int nol_quant10[15]; extern int nol_quant19[15]; extern int nol_ytdqty[15]; </pre>
--	--

<pre> extern int no_amount[15]; extern int o_all_local; extern float w_tax; extern float d_tax; extern float total_amount; extern char i_name[15][25]; extern int i_name_strlen[15]; extern ub2 i_name_strlen_len[15]; extern ub2 i_name_strlen_rcode[15]; extern ub4 i_name_strlen_csize; extern int s_quantity[15]; extern char brand_gen[15]; extern ub2 brand_gen_len[15]; extern ub2 brand_gen_rcode[15]; extern ub4 brand_gen_csize; extern int i_price[15]; extern char brand_generic[15][1]; extern int status; extern int tracelevel; /* Miscellaneous */ extern OCIDate cr_date; extern OCIDate c_since; extern OCIDate o_entry_d_base; extern OCIDate ol_d_base[15]; #endif #ifndef DISCARD #define DISCARD (void) #endif #ifndef sword #define sword int #endif #define VER7 2 #define NA -1 /* ANSI SQL NULL */ #define NLT 1 /* length for string null terminator */ #define DEADLOCK 60 /* ORA-00060: deadlock */ #define NO_DATA_FOUND 1403 /* ORA-01403: no data found */ #define NOT_SERIALIZABLE 8177 /* ORA-08177: transaction not serializable */ #define SNAPSHOT_TOO_OLD 1555 /* ORA-01555: snapshot too old */ #ifndef NULLP #define NULLP (void *)NULL #endif /* NULLP */ #define ADR(object) ((ub1 *) &(object)) #define SIZ(object) ((sword) sizeof(object)) typedef char date[24+NLT]; typedef char varchar2; #define min(x,y) (((x) < (y)) ? (x) : (y)) #define OCIERROR(errp,function) \ ocierror(__FILE__,__LINE__,(errp),(function)); #define OCIBND(stmp, bndp, errp, sqlvar, 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)), \ (progvl), (progvl), (ftype), 0, 0, 0, 0, OCI_DEFAULT)); #define OCIBNDRA(stmp,bndp,errp,sqlvar,progvl,ftype,indp,alen,arcodes) \ 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)), \ (progvl),(progvl),(ftype),(indp),(alen),(arcodes),0,0,OCI_DEFAULT)); #define OCIBNDRAD(stmp,bndp,errp,sqlvar,progvl,ftype,indp,ctxp,cbf_nodata,cbf_data) \ ocierror(__FILE__,__LINE__,(errp), \ OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_BIND,0,(dvoid**0)); \ ocierror(__FILE__,__LINE__,(errp), \ OCIBindByName((stmp),&(bndp),(errp),(text *) (sqlvar), \ strlen((sqlvar)),0,(progvl),(ftype), \ indp,0,0,0,0,OCI_DATA_AT_EXEC)); \ ocierror(__FILE__,__LINE__,(errp), \ OCIBindDynamic((bndp),(errp),(ctxp),(cbf_nodata),(ctxp),(cbf_data)); #define OCIBNDR(stmp,bndp,errp,sqlvar,progvl,ftype,indp,alen,arcodes) \ 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)), \ (progvl),(progvl),(ftype),(indp),(alen),(arcodes),0,0,OCI_DEFAULT)); #define OCIBNDRAA(stmp,bndp,errp,sqlvar,progvl,ftype,indp,alen,arcodes,ms,cu) \ ocierror(__FILE__,__LINE__,(errp), \ OCIHandleAlloc((stmp),(dvoid**)&(bndp),OCI_HTYPE_BIND,0,(dvoid**0)); \ </pre>	<pre> ocierror(__FILE__,__LINE__,(errp), \ OCIBindByName((stmp),&(bndp),(errp),(text *) (sqlvar),strlen((sqlvar)), \ (progvl),(progvl),(ftype),(indp),(alen),(arcodes),(ms),(cu),OCI_DEFAULT)); #define OCIDEFINE(stmp,dfnp,errp,pos,progvl,ftype) \ OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progvl),(progvl),(ftype), \ 0,0,0,OCI_DEFAULT); #define OCIDDEF(stmp,dfnp,errp,pos,progvl,ftype) \ OCIHandleAlloc((stmp),(dvoid**)&(dfnp),OCI_HTYPE_DEFINE,0, \ (dvoid**0)); \ OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progvl),(progvl), \ (ftype),NULL,NULL,NULL,OCI_DEFAULT)); #define OCIDFNRA(stmp,dfnp,errp,pos,progvl,ftype,indp,alen,arcodes) \ OCIHandleAlloc((stmp),(dvoid**)&(dfnp),OCI_HTYPE_DEFINE,0, \ (dvoid**0)); \ OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progvl), \ (progvl),(ftype),(indp),(alen), \ (arcodes),OCI_DEFAULT); #define OCIDFNDR(stmp,dfnp,errp,pos,progvl,ftype,indp,ctxp,cbf_data) \ ocierror(__FILE__,__LINE__,(errp), \ OCIHandleAlloc((stmp),(dvoid**)&(dfnp),OCI_HTYPE_DEFINE,0, \ (dvoid**0)); \ ocierror(__FILE__,__LINE__,(errp), \ OCIDefineByPos((stmp),&(dfnp),(errp),(pos),(progvl), \ (progvl),(ftype), \ (indp),NULL,NULL,OCI_DYNAMIC_FETCH)); \ ocierror(__FILE__,__LINE__,(errp), \ OCIDefineDynamic((dfnp),(errp),(ctxp),(cbf_data)); #endif /* New order */ struct newinstruct { int w_id; int d_id; int c_id; int ol_i_id[15]; int ol_supply_w_id[15]; int ol_quantity[15]; }; struct newoutstruct { int terror; int o_id; int o_ol_cnt; char c_last[17]; char c_credit[3]; float c_discount; float w_tax; float d_tax; char o_entry_d[20]; float total_amount; char i_name[15][25]; int s_quantity[15]; char brand_generic[15]; float i_price[15]; float ol_amount[15]; char status[26]; int retry; }; struct newstruct { struct newinstruct newin; struct newoutstruct newout; }; /* Payment */ struct payinstruct { int w_id; int d_id; int c_w_id; int c_d_id; int c_id; int bylastname; int h_amount; char c_last[17]; }; struct payoutstruct { int terror; char w_street_1[21]; char w_street_2[21]; char w_city[21]; char w_state[3]; char w_zip[10]; char d_street_1[21]; char d_street_2[21]; char d_city[21]; char d_state[3]; char d_zip[10]; int c_id; </pre>
--	--

```

char c_first[17];
char c_middle[3];
char c_last[17];
char c_street_1[21];
char c_street_2[21];
char c_city[21];
char c_state[3];
char c_zip[10];
char c_phone[17];
char c_since[11];
char c_credit[3];
double c_credit_lim;
float c_discount;
double c_balance;
char c_data[201];
char h_date[20];
int retry;
};

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

/* Order status */

struct ordinstruc {
    int w_id;
    int d_id;
    int c_id;
    int bylastname;
    char c_last[17];
};

struct ordoutstruct {
    int terror;
    int c_id;
    char c_last[17];
    char c_first[17];
    char c_middle[3];
    double c_balance;
    int o_id;
    char o_entry_d[20];
    int o_carrier_id;
    int o_ol_cnt;
    int ol_supply_w_id[15];
    int ol_i_id[15];
    int ol_quantity[15];
    float ol_amount[15];
    char ol_delivery_d[15][11];
    int retry;
};

struct ordstruct {
    struct ordinstruc ordin;
    struct ordoutstruct ordout;
};

/* Delivery */

struct delinstruc {
    int w_id;
    int o_carrier_id;
    double qtime;
    int in_timing_int;
};

struct deloutstruct {
    int terror;
    int retry;
};

struct delstruct {
    struct delinstruc delin;
    struct deloutstruct delout;
};

/* Stock level */

struct stoinstruct {
    int w_id;
    int d_id;
    int threshold;
};

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

struct stostruct {
    struct stoinstruct stoin;
    struct stoostruct stooout;
};

#endif

#endif

                                tpcc_info.h

/*
 * $Header:
 */
/afs/transarc.com/project/encina/rcs/test/src/benchmarks/tpcc/server/ora8.1_mt/RCS/tpcc_info.h,v 1.1
1999/04/14 19:03:06wenjian Exp $ Copyr (c) 1993 Oracle
*/
/*=====+
|      Copyright (c) 1995 OracleCorp, Redwood Shores, CA      |
|      OPEN SYSTEMS PERFORMANCE GROUP                          |
|      All Rights Reserved                                      |
+=====+
| FILENAME
| tpcc.h
| DESCRIPTION
| Include file for TPC-C benchmark programs.
+=====+*/

#ifdef TPCC_INFO_H
#define TPCC_INFO_H

/* New order */

struct newinstruct {
    int w_id;
    int d_id;
    int c_id;
    int ol_i_id[15];
    int ol_supply_w_id[15];
    int ol_quantity[15];
};

struct newoutstruct {
    int terror;
    int o_id;
    int o_ol_cnt;
    char c_last[17];
    char c_credit[3];
    float c_discount;
    float w_tax;
    float d_tax;
    char o_entry_d[20];
    float total_amount;
    char i_name[15][25];
    int s_quantity[15];
    char brand_generic[15];
    float i_price[15];
    float ol_amount[15];
    char status[26];
    int retry;
};

struct newstruct {
    struct newinstruct newin;
    struct newoutstruct newout;
};

/* Payment */

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

struct payoutstruct {
    int terror;
    char w_street_1[21];
    char w_street_2[21];
    char w_city[21];
    char w_state[3];
    char w_zip[10];
    char d_street_1[21];
    char d_street_2[21];
    char d_city[21];
    char d_state[3];
    char d_zip[10];
    int c_id;
    char c_first[17];
    char c_middle[3];
    char c_last[17];
    char c_street_1[21];
    char c_street_2[21];
    char c_city[21];
    char c_state[3];
    char c_zip[10];
    char c_phone[17];
};

```



```

char c_since[11];
char c_credit[3];
double c_credit_lim;
float c_discount;
double c_balance;
char c_data[201];
char h_date[20];
int retry;
};

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

/* Order status */

struct ordinstruct {
    int w_id;
    int d_id;
    int c_id;
    int bylastname;
    char c_last[17];
};

struct ordoutstruct {
    int terror;
    int c_id;
    char c_last[17];
    char c_first[17];
    char c_middle[3];
    double c_balance;
    int o_id;
    char o_entry_d[20];
    int o_carrier_id;
    int o_ol_cnt;
    int ol_supply_w_id[15];
    int ol_i_id[15];
    int ol_quantity[15];
    float ol_amount[15];
    char ol_delivery_d[15][11];
    int retry;
};

struct ordstruct {
    struct ordinstruct ordin;
    struct ordoutstruct ordout;
};

/* Delivery */

struct delinstruct {
    int w_id;
    int o_carrier_id;
    double qtime;
    int in_timing_int;
};

struct deloutstruct {
    int terror;
    int retry;
};

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

/* Stock level */

struct stoinstruct {
    int w_id;
    int d_id;
    int threshold;
};

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

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

#endif

tpccflags.h

/* #define DMLRETNO */
#define PLSOLNO
#define DMLRETDEL
/* #define PLSQLORD */

```

tpccpl.c

```

#ifdef RCSID
static char *RCSid =
    "SHheader: /afs/transarc.com/project/encina/rcs/test/src/benchmarks/tpcc/server/ora8.1_mt/RCS/tpccpl.c,v
    1.7 1999/05/26 16:29:59wenjian Exp S Copyr (c) 1994 Oracle";
#endif /* RCSID */

/*=====+
|   Copyright (c) 1994 OracleCorp, 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 "tpcc.h"
#ifdef TUX
#include <userlog.h>
#else
#include <stdarg.h>
#endif
#ifdef MULTI_THREADED
#include <dce/pthread.h>
#endif
#include "plora.h"

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

int proc_no;
static char *db_uid;
static char *db_pwd;

/* Delivery fileinfomation: Global.
 * One output file for deliveries for the server
 */
static char delivery_file_name[100];

#ifdef MULTI_THREADED
pthread_mutex_t dvry_log_lock;
#define DVRY_LOCK pthread_mutex_lock(&dvry_log_lock);
#define DVRY_UNLOCK pthread_mutex_unlock(&dvry_log_lock);
#define DVRY_LOCK_INIT pthread_mutex_init(&dvry_log_lock, pthread_mutexattr_default);
#else
#define DVRY_LOCK
#define DVRY_UNLOCK
#define DVRY_LOCK_INIT
#endif
FILE *fip;
FILE *fopen ();

#ifdef ORA_NT
extern double dpbtimef();
#define gettime dpbtimef
#else
double gettime ();
#endif

/* Initialization of one connection */
static void initOCHandles(ora_cn_data_t *cn_dataP, char* uid, char *pwd);

static int init_cn_data(ora_cn_data_t *dataP);

extern char oracle_home[256];

/* NewOrder Binding stuff */

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

/* vmm313 void ocierror(fname, lineno, errhp, status) */
int ocierror(fname, lineno, errhp, status)
char *fname;
int lineno;
OCIError *errhp;
sword status;
{
    text errbuf[512];
    ub4 buflen;
    sb4 errcode;
    sb4 lstat;

```

```

ub4 recno=2;

switch (status) {
case OCI_SUCCESS:
    break;
case OCI_SUCCESS_WITH_INFO:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_SUCCESS_WITH_INFO\n");
    lstat = OCIErrorGet (errhp, recno++, (text *) NULL, &errcode, errbuf,
        (ub4) sizeof(errbuf), OCI_HTYPE_ERROR);
    fprintf(stderr,"Error - %s\n", errbuf);
    break;
case OCI_NEED_DATA:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_NEED_DATA\n");
    return (IRRECERR);
case OCI_NO_DATA:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_NO_DATA\n");
    return (IRRECERR);
case OCI_ERROR:
    lstat = OCIErrorGet (errhp, (ub4) 1,
        (text *) NULL, &errcode, errbuf,
        (ub4) sizeof(errbuf), OCI_HTYPE_ERROR);
    if (errcode == NOT_SERIALIZABLE) return (errcode);
    while (lstat != OCI_NO_DATA)
    {
        fprintf(stderr,"Module %s Line %d\n", fname, lineno);
        fprintf(stderr,"Error - %s\n", errbuf);
        lstat = OCIErrorGet (errhp, recno++, (text *) NULL, &errcode, errbuf,
            (ub4) sizeof(errbuf), OCI_HTYPE_ERROR);
    }
    return (errcode);
/* vmm313 TPCexit(1); */
/* vmm313 exit(1); */
case OCI_INVALID_HANDLE:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_INVALID_HANDLE\n");
    TPCexit(1);
    exit(-1);
case OCI_STILL_EXECUTING:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_STILL_EXECUTE\n");
    return (IRRECERR);
case OCI_CONTINUE:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Error - OCI_CONTINUE\n");
    return (IRRECERR);
default:
    fprintf(stderr,"Module %s Line %d\n", fname, lineno);
    fprintf(stderr,"Status - %s\n", status);
    return (IRRECERR);
}
return (RECOVER);
}

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

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

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

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

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

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

#ifdef NOT
void vgetdate (unsigned char *oradt)
{

```

```

struct tm *loctime;
time_t int_time;

struct ORADATE {
    unsigned 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);

    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",
        day,month,year,hour,min,sec);

return;
}
#endif

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

/* clean_connection
* Called to clean a connection.
* When using pthread this is registered duringpthread_create
* and called automatically by pthread when the thread exits.
*/
void clean_connection(void *ptr)
{
/* free trans specific cursor handles first and later theora handles */
ora_cn_data_t *cn_dataP = (ora_cn_data_t *)ptr;

if (cn_dataP != NULL) {
OCIServer *tpcsrv;
OCISession *tpcsrv;
OCIEnv *tpcenv;
OCIError *errhp;
OCISvcCtx *tpscvc;

fprintf(stderr,"clean_connection, Freeing OCI handles\n");
tkvcndone(cn_dataP);
tkvcpdone(cn_dataP);
tkvcodone(cn_dataP);
tkvcddone(cn_dataP);
tkvcsdone(cn_dataP);

/* free OCI handles */
if (tpcsrv = cn_dataP->tpcsrv) {
fprintf(stderr, "free_handles> OCIServer\n");
OCIHandleFree((dvoid *)tpcsrv, OCI_HTYPE_SESSION);
}
if (tpscvc = cn_dataP->tpscvc) {
fprintf(stderr, "free_handles> OCISvcCtx\n");
OCIHandleFree((dvoid *)tpscvc, OCI_HTYPE_SVCCTX);
}
if (errhp = cn_dataP->errhp) {
fprintf(stderr, "free_handles> OCIError\n");
OCIHandleFree((dvoid *)errhp, OCI_HTYPE_ERROR);
}
if (tpcsrv = cn_dataP->tpcsrv) {
fprintf(stderr, "free_handles> OCIServer\n");
OCIHandleFree((dvoid *)tpcsrv, OCI_HTYPE_SERVER);
}
if (tpcenv = cn_dataP->tpcenv) {
fprintf(stderr, "free_handles> OCIEnv\n");
OCIHandleFree((dvoid *)tpcenv, OCI_HTYPE_ENV);
}

fprintf(stderr, "free_handles> free cn_dataP\n");
}
}

TPCinit (id, uid, pwd, dvryFileName)
int id;
char *uid;
char *pwd;
char *dvryFileName;
{
int i;
text stmbuf[100];

fprintf(stderr, "TPCinit id %d, uid %s pwd %s\n", id, uid, pwd);

DVRY_LOCK_INIT;

proc_no = id;
db_uid = (char *)calloc(strlen(uid) + 1, sizeof(char));
strcpy(db_uid, uid);
db_pwd = (char *)calloc(strlen(pwd) + 1, sizeof(char));
strcpy(db_pwd, pwd);

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

OCIInitialize(OCI_DEFAULT|OCI_OBJECT,(dvoid*)0,0,0,0);/* check tpccpl.c */

return (0);
}

static void initOCIhandles(ora_cn_data_t *cn_dataP, char *uid, char *pwd)
{
int tracelevel= 0; /* new define */
OCIDate cr_date;
text stmbuf[100];
OCIEnv *tpcenv;
OCIServer *tpcsrv;
OCIError *errhp;
OCISvcCtx *tpscvc;
OCISession *tpcsrv;
OCIStmnt *curi;

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 **)&tpscvc, OCI_HTYPE_SVCCTX, 0, (dvoid **)0);
OCIServerAttach(tpcsrv, errhp, (text *)0, OCI_DEFAULT);
OCIAttrSet((dvoid *)tpscvc, OCI_HTYPE_SVCCTX, (dvoid *)tpcsrv, (ub4)0, OCI_ATTR_SERVER,
errhp);
OCIHandleAlloc((dvoid *)tpcenv, (dvoid **)&tpcsrv, OCI_HTYPE_SESSION, 0, (dvoid **)0);
OCIAttrSet((dvoid *)tpcsrv, OCI_HTYPE_SESSION, (dvoid *)uid,
(ub4)strlen(uid), OCI_ATTR_USERNAME, errhp);
OCIAttrSet((dvoid *)tpcsrv, OCI_HTYPE_SESSION, (dvoid *)pwd, (ub4)strlen(pwd),
OCI_ATTR_PASSWORD, errhp);
OCIERROR(errhp, OCI_SessionBegin(tpscvc, errhp, tpcsrv, OCI_CRED_RDBMS, OCI_DEFAULT));

OCIAttrSet(tpscvc, OCI_HTYPE_SVCCTX, tpcsrv, 0, OCI_ATTR_SESSION, errhp);

/* run all transaction in serializable mode */

OCIHandleAlloc(tpcenv, (dvoid **)&curi, OCI_HTYPE_STMT, 0, (dvoid **)0);
sprintf ((char *) stmbuf, SQLTXTR);
OCIStmntPrepare(curi, errhp, stmbuf, strlen((char *)stmbuf), OCI_NTV_SYNTAX, OCI_DEFAULT);
OCIERROR(errhp, OCIStmntExecute(tpscvc, curi, errhp, 1, 0, 0, OCI_DEFAULT));
OCIHandleFree(curi, OCI_HTYPE_STMT);

/*
This is done in cvdrv.c
if (tracelevel== 2) {
OCIHandleAlloc(tpcenv, (dvoid **)&curi, OCI_HTYPE_STMT, 0, (dvoid **)0);
memset(stmbuf, 0, 100);
sprintf ((char *) stmbuf, SQLTXTR);
OCIStmntPrepare(curi, errhp, stmbuf, strlen((char *)stmbuf),
OCI_NTV_SYNTAX, OCI_DEFAULT);
OCIERROR(errhp, OCIStmntExecute(tpscvc, curi, errhp, 1, 0, 0, OCI_DEFAULT));
OCIHandleFree((dvoid *)curi, OCI_HTYPE_STMT);
}
*/
if (tracelevel== 3) {
OCIHandleAlloc(tpcenv, (dvoid **)&curi, OCI_HTYPE_STMT, 0, (dvoid **)0);
memset(stmbuf, 0, 100);
sprintf ((char *) stmbuf, SQLTXTR);
OCIStmntPrepare(curi, errhp, stmbuf, strlen((char *)stmbuf),
OCI_NTV_SYNTAX, OCI_DEFAULT);
OCIERROR(errhp, OCIStmntExecute(tpscvc, curi, errhp, 1, 0, 0, OCI_DEFAULT));
OCIHandleFree((dvoid *)curi, OCI_HTYPE_STMT);
}

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

/* Store the handles just initialized in the thread slot
*/
cn_dataP->tpcenv = tpcenv;
cn_dataP->tpcsrv = tpcsrv;
cn_dataP->errhp = errhp;
cn_dataP->tpscvc = tpscvc;
cn_dataP->tpcsrv = tpcsrv;
cn_dataP->curi = curi;
}

/*
* init_cn_data
* Initializes all the transactions for a single connection
*/
static int init_cn_data(ora_cn_data_t *cnP)
{
int status;

initOCIhandles(cnP, db_uid, db_pwd);

if (status = tkvcninit(cnP)) { /* new order */

```

<pre> fprintf(stderr, "tkvcninit failed: %d\n", status); TPCexit (); return (status); } if (status = tkvcpin (cnP)) { /* payment */ fprintf(stderr, "tkvcpin failed: %d\n", status); TPCexit (); return (status); } if (status = tkvcoint (cnP)) { /* order status */ fprintf(stderr, "tkvcoint failed: %d\n", status); TPCexit (); return (status); } if (status = tkvcdinit (cnP)) { /* delivery */ fprintf(stderr, "tkvcdinit failed: %d\n", status); TPCexit (); return (status); } if (status = tkvcsinit (cnP)) { /* stock level */ fprintf(stderr, "tkvcsinit failed: %d\n", status); TPCexit (); return (status); } return 0; } void *create_ora_connection() { ora_cn_data_t *cnP = (ora_cn_data_t *)malloc(sizeof(ora_cn_data_t)); init_cn_data(cnP); return (void *)cnP; } ***** The Transaction Code *****/ TPCnew (cnP, str) void *cnP; struct newstruct *str; { int i; ora_cn_data_t *cn_dataP = (ora_cn_data_t *)cnP; global_newOrder_t *newP = cn_dataP->globals; OCIError *errhp = cn_dataP->errhp; newP->w_id = str->newin.w_id; newP->d_id = str->newin.d_id; newP->c_id = str->newin.c_id; for (i = 0; i < 15; i++) { newP->no_l_id[i] = str->newin.ol_i_id[i]; newP->no_l_supply_w_id[i] = str->newin.ol_supply_w_id[i]; newP->no_l_quantity[i] = str->newin.ol_quantity[i]; } newP->retries = 0; } vgetdate(newP->cr_date); /* OCIERROR(errhp,OCIDateSysDate(errhp,&newP->cr_date)); if (str->newout.terror = tkvcn (cn_dataP)) { if (str->newout.terror != RECOVERR) str->newout.terror = IRRECERR; str->newout.retry = newP->retries; return (-1); } /* fill in date foro_entry_d from time in beginning of txn*/ cvtdmyhms(newP->cr_date,newP->o_entry_d); newP->datelen = sizeof(newP->o_entry_d); OCIERROR(errhp, OCIDateToText(errhp,&newP->cr_date,(text*)FULLDATE,SIZ(FULLDATE),(text*)0,0, &newP->datelen,newP->o_entry_d)); str->newout.terror = NOERR; str->newout.o_id = newP->o_id; str->newout.o_ol_cnt = newP->o_ol_cnt; strncpy (str->newout.c_last, newP->c_last, 17); strncpy (str->newout.c_credit, newP->c_credit, 3); str->newout.c_discount = newP->c_discount; str->newout.w_tax = newP->w_tax; str->newout.d_tax = newP->d_tax; strncpy (str->newout.o_entry_d, (char*)newP->o_entry_d, 20); str->newout.total_amount = newP->total_amount; for (i = 0; i < newP->o_ol_cnt; i++) { strncpy (str->newout.i_name[i], newP->i_name[i], 25); str->newout.s_quantity[i] = newP->s_quantity[i]; str->newout.brand_generic[i] = newP->brand_gen[i]; str->newout.i_price[i] = (float)(newP->i_price[i])/100; str->newout.ol_amount[i] = (float)(newP->no_l_amount[i])/100; } if (newP->status) strcpy (str->newout.status, "Item number is not valid"); else str->newout.status[0] = '0'; </pre>	<pre> str->newout.retry = newP->retries; return (0); } TPCpay (cnP, str) void *cnP; struct paystruct *str; { ora_cn_data_t *cn_dataP = (ora_cn_data_t *)cnP; global_payment_t *payP = cn_dataP->payP; OCIError *errhp = cn_dataP->errhp; payP->w_id = str->payin.w_id; payP->d_id = str->payin.d_id; payP->c_w_id = str->payin.c_w_id; payP->c_d_id = str->payin.c_d_id; payP->h_amount = str->payin.h_amount; payP->bylastname = str->payin.bylastname; /* vgetdate(payP->cr_date); /* OCIERROR(errhp,OCIDateSysDate(errhp,&payP->cr_date)); if (payP->bylastname) { payP->c_id = 0; strncpy (payP->c_last, str->payin.c_last, 17); } else { payP->c_id = str->payin.c_id; strcpy (payP->c_last, " "); } payP->retries = 0; if (str->payout.terror = tkvcp (cn_dataP)) { if (str->payout.terror != RECOVERR) str->payout.terror = IRRECERR; return (-1); } /* cvtdmyhms(cr_date,h_date); payP->hlen=SIZ(payP->h_date); OCIERROR(errhp,OCIDateToText(errhp,&payP->cr_date, (text*)FULLDATE,strlen(FULLDATE),(text*)0,0,&payP->hlen,payP->h_date)); cvtdmy(c_since,c_since_d); payP->syncelen=SIZ(payP->c_since_d); OCIERROR(errhp,OCIDateToText(errhp,&payP->c_since, (text*)SHORTDATE,strlen(SHORTDATE),(text*)0,0,&payP->syncelen,payP->c_since_d)); str->payout.terror = NOERR; strncpy (str->payout.w_street_1, payP->w_street_1, 21); strncpy (str->payout.w_street_2, payP->w_street_2, 21); strncpy (str->payout.w_city, payP->w_city, 21); strncpy (str->payout.w_state, payP->w_state, 3); strncpy (str->payout.w_zip, payP->w_zip, 10); strncpy (str->payout.d_street_1, payP->d_street_1, 21); strncpy (str->payout.d_street_2, payP->d_street_2, 21); strncpy (str->payout.d_city, payP->d_city, 21); strncpy (str->payout.d_state, payP->d_state, 3); strncpy (str->payout.d_zip, payP->d_zip, 10); str->payout.c_id = payP->c_id; strncpy (str->payout.c_first, payP->c_first, 17); strncpy (str->payout.c_middle, payP->c_middle, 3); strncpy (str->payout.c_last, payP->c_last, 17); strncpy (str->payout.c_street_1, payP->c_street_1, 21); strncpy (str->payout.c_street_2, payP->c_street_2, 21); strncpy (str->payout.c_city, payP->c_city, 21); strncpy (str->payout.c_state, payP->c_state, 3); strncpy (str->payout.c_zip, payP->c_zip, 10); strncpy (str->payout.c_phone, payP->c_phone, 17); strncpy (str->payout.c_since, (char*)payP->c_since_d, 11); strncpy (str->payout.c_credit, payP->c_credit, 3); str->payout.c_credit_lim = (float)(payP->c_credit_lim)/100; str->payout.c_discount = payP->c_discount; str->payout.c_balance = (float)(payP->c_balance)/100; strncpy (str->payout.c_data, payP->c_data, 201); strncpy (str->payout.h_date, (char*)payP->h_date, 20); str->payout.retry = payP->retries; return (0); } TPCord (cnP, str) void *cnP; struct ordstruct *str; </pre>
--	---

```

{
ora_cn_data_t *cn_dataP = (ora_cn_data_t *)cnP;
global_order_t *ordP = cn_dataP->ordP;
int i;

ordP->w_id = str->ordin.w_id;
ordP->d_id = str->ordin.d_id;
ordP->bylastname = str->ordin.bylastname;
if (ordP->bylastname) {
ordP->c_id = 0;
strncpy (ordP->c_last, str->ordin.c_last, 17);
}
else {
ordP->c_id = str->ordin.c_id;
strncpy (ordP->c_last, " ");
}
ordP->retries = 0;

if (str->ordout.terror = tkvco (cn_dataP)) {
if (str->ordout.terror != RECOVERR)
str->ordout.terror = IRRECERR;
return (-1);
}

str->ordout.terror = NOERR;
str->ordout.c_id = ordP->c_id;
strncpy (str->ordout.c_last, ordP->c_last, 17);
strncpy (str->ordout.c_first, ordP->c_first, 17);
strncpy (str->ordout.c_middle, ordP->c_middle, 3);
str->ordout.c_balance = ordP->c_balance/100;
str->ordout.o_id = ordP->o_id;
strncpy (str->ordout.o_entry_d, (char*)ordP->o_entry_d, 20);
if (ordP->o_carrier_id == 11 )
str->ordout.o_carrier_id = 0;
else
str->ordout.o_carrier_id = ordP->o_carrier_id;
str->ordout.o_ol_cnt = ordP->o_ol_cnt;
for (i = 0; i < ordP->o_ol_cnt; i++) {
ordP->o_l_delivery_d[i][10] = '\0';
if (!strcmp((char*)ordP->o_l_delivery_d[i], "01-01-1811"))
strncpy((char*)ordP->o_l_delivery_d[i], "NOTDELIVR", 10);
str->ordout.o_l_supply_w_id[i] = ordP->o_l_supply_w_id[i];
str->ordout.o_l_i_id[i] = ordP->o_l_i_id[i];
str->ordout.o_l_quantity[i] = ordP->o_l_quantity[i];
str->ordout.o_l_amount[i] = (float)(ordP->o_l_amount[i])/100;
strncpy (str->ordout.o_l_delivery_d[i], (char*)ordP->o_l_delivery_d[i], 11);
}
str->ordout.retry = ordP->retries;

return (0);
}

TPCdel (cnP, str)
void *cnP;
struct delstruct *str;

{
ora_cn_data_t *cn_dataP = (ora_cn_data_t *)cnP;
global_delivery_t *delP = cn_dataP->delP;
OCIERror *errhp = cn_dataP->errhp;
double tr_end, tr_begin;
int i, skipped;
struct timeval cur_time;
static int tran_cntr=0;
int pos, len;
int queue_time, start_time, end_time;
char stdout_buf[1024];

/* Open the delivery log file if needed */
if (lfp == NULL) {
DVRVY_LOCK;
if (lfp == NULL) {
err_printf("TPCdel: delivery_file_name is %s\n", delivery_file_name);
if ((lfp = fopen (delivery_file_name, "w")) == NULL) {
fprintf (stderr, "Error in TPC-C server: Failed to open %s\n",
delivery_file_name);
DVRVY_UNLOCK;
return(-1);
}
err_printf("Opened delivery file%s\n", delivery_file_name);
}
DVRVY_UNLOCK;
}

#ifdef USE_ORACLE_DVRVY_FORMAT
gettimeofday(&cur_time, NULL);
tr_begin = (double)cur_time.tv_sec + 1.0e-6 *(double)cur_time.tv_usec;
start_time = cur_time.tv_sec;
#endif

delP->w_id = str->delin.w_id;
delP->o_carrier_id = str->delin.o_carrier_id;
delP->retries = 0;

/*
vgetdate(cr_date); */
OCIERror(errhp, OCIDateSysDate(errhp, &delP->cr_date));

```

```

if (str->delout.terror = tkvco (cn_dataP)) {
if (str->delout.terror == DEL_ERROR)
return DEL_ERROR;
if (str->delout.terror != RECOVERR)
str->delout.terror = IRRECERR;
return (-1);
}

#ifdef USE_ORACLE_DVRVY_FORMAT
tr_end = gettimeofday ();
DVRVY_LOCK;
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, delP->w_id, delP->o_carrier_id);
for (i = 0; i < 10; i++) {
fprintf (lfp, " %d %d", i + 1, delP->del_o_id[i]);
if (delP->del_o_id[i] <= 0) {
#ifdef TUX
userlog ("DELIVERY: no new order for w_id: %d, d_id %d\n",
delP->w_id, i + 1);
#else
fprintf (stderr, "DELIVERY: no new order for w_id: %d, d_id %d\n",
delP->w_id, i + 1);
#endif
}
}
fprintf (lfp, " %d\n", delP->retries);

/* not USE_ORACLE_DVRVY_FORMAT */
gettimeofday(&cur_time, NULL);
tr_end = (double)cur_time.tv_sec + 1.0e-6 *(double)cur_time.tv_usec;
end_time = cur_time.tv_sec;

queue_time = str->delin.qtime;
pos = 0;

DVRVY_LOCK;
++tran_cntr;

pos += sprintf(&stdout_buf[pos], "--Tran %d Queue %.3f Start %.3f\n",
tran_cntr, str->delin.qtime, tr_begin);
pos += sprintf(&stdout_buf[pos], "W_ID: %d, CARRIER_ID: %d",
str->delin.w_id, str->delin.o_carrier_id);

if (str->delout.terror == DEL_ERROR) {
pos += sprintf(&stdout_buf[pos],
"\nDelivery transaction failed (DEL_ERROR)\n");
} else if (str->delout.terror != 0) {
pos += sprintf(&stdout_buf[pos], "Delivery transaction failed (%d)",
str->delout.terror);
} else {
int skipped[10];
int num_skipped = 0;
for (i = 0; i < 10; i++) {
if (delP->del_o_id[i] <= 0) {
skipped[i] = 1;
num_skipped ++;
} else {
skipped[i] = 0;
}
pos += sprintf(&stdout_buf[pos], " %d", delP->del_o_id[i]);
}
pos += sprintf(&stdout_buf[pos], "\n");
if (num_skipped > 0) {
for (i=0; i<10; i++) {
if (skipped[i] == 1) {
pos += sprintf(&stdout_buf[pos],
"D_ID %d has no new orders.\n", i+1);
}
}
}
}

fprintf(lfp, "%send-time: %.3f\n", stdout_buf, tr_end);
fflush (lfp);
#ifdef USE_ORACLE_DVRVY_FORMAT */
DVRVY_UNLOCK;
str->delout.terror = NOERR;
str->delout.retry = delP->retries;

return (0);
}

TPCsto (cnP, str)
void *cnP;
struct stostruct *str;

{
ora_cn_data_t *cn_dataP = (ora_cn_data_t *)cnP;
global_stock_t *stoP = cn_dataP->stoP;

stoP->w_id = str->stoin.w_id;
stoP->d_id = str->stoin.d_id;
stoP->threshold = str->stoin.threshold;

```

```

stoP->retries = 0;

if (str->stoout.terror = tkvcs (cn_dataP)) {
  if (str->stoout.terror != RECOVERR)
    str->stoout.terror = IRRECERR;
  return (-1);
}

str->stoout.terror = NOERR;
str->stoout.low_stock = stoP->low_stock;
str->stoout.retry = stoP->retries;

return (0);
}

```

views.sql

```

create or replace view wh_cust
(w_id, w_tax, c_id, c_d_id, c_w_id, c_discount, c_last, c_credit)
as select w.w_id, w.w_tax,
       c.c_id, c.c_d_id, c.c_w_id, c.c_discount, c.c_last, c.c_credit
   from customer c, warehouse w
  where w.w_id = c.c_w_id
/

create or replace view wh_dist
(w_id, d_id, d_tax, d_next_o_id, w_tax )
as select w.w_id, d.d_id, d.d_tax, d.d_next_o_id, w.w_tax
   from 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
/
exit

```

APPENDIX B: Tunable Parameters

B.1 Database Parameters

```
# SHeader: p_run.ora 7030100.1 95/07/14 18:49:15plai Generic<base> $ Copyr (c) 1993 Oracle
#
#====#+
# Copyright (c) 1995 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#====#+
# FILENAME
# p_run.ora
# DESCRIPTION
# Oracle parameter file for runningTPC-C.
#====#+
control_files = /dev/rtpc_lvcnt1, /dev/rtpc_lvcnt2, /dev/rtpc_lvcnt3

statistics_level = basic
log_parallelism = 4
timed_statistics = FALSE
remote_login_passwordfile = shared
java_pool_size = 1M
disk_asynch_io = TRUE
db_writer_processes = 8
recovery_parallelism = 32
parallel_max_servers = 200
#%
_igwr_async_io = FALSE
_db_writer_chunk_writes = 1000
_db_writer_max_writes = 1000
_db_aging_hot_criteria = 2
db_block_checksum = FALSE
trace_enabled = FALSE
enqueue_resources = 60000
compatible = 9.2.0.0.0
db_name = tpcc
db_files = 2000
db_block_size = 4096
dml_locks = 500

hash_join_enabled = FALSE

log_archive_start = FALSE
log_checkpoint_timeout = 0
log_checkpoint_interval = 1000000000
log_checkpoints_to_alert = TRUE
max_rollback_segments = 1250
max_dump_file_size = 3000
open_cursors = 2000
processes = 2000
sessions = 2250
transactions = 6000

transactions_per_rollback_segment = 1

cursor_space_for_time = TRUE
replication_dependency_tracking = FALSE

shared_pool_size = 500000000

db_cache_size = 57100M
db_16k_cache_size = 51000M
db_keep_cache_size = 250000M
db_recycle_cache_size = 100000M
_db_block_max_dirty_target = 57354485
log_buffer = 33554432 # 1.0M xcpu

_disable_incremental_checkpoints = TRUE

lock_sga = TRUE

# DB Writer important parameters
_db_aging_stay_count = 1
####

transaction_auditing = false
db_file_multiblock_read_count = 1

_log_simultaneous_copies = 32

rollback_segments = (t1,t2,t3,t4,t5,t6,t7,t8,t9,t10),
t11,t12,t13,t14,t15,t16,t17,t18,t19,t20,
t21,t22,t23,t24,t25,t26,t27,t28,t29,t30,
t31,t32,t33,t34,t35,t36,t37,t38,t39,t40,
t41,t42,t43,t44,t45,t46,t47,t48,t49,t50,
t51,t52,t53,t54,t55,t56,t57,t58,t59,t60,
t61,t62,t63,t64,t65,t66,t67,t68,t69,t70,
t71,t72,t73,t74,t75,t76,t77,t78,t79,t80,
t81,t82,t83,t84,t85,t86,t87,t88,t89,t90,
t91,t92,t93,t94,t95,t96,t97,t98,t99,t100,
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,t210,
t211,t212,t213,t214,t215,t216,t217,t218,t219,t220,
t221,t222,t223,t224,t225,t226,t227,t228,t229,t230,
t231,t232,t233,t234,t235,t236,t237,t238,t239,t240,
t241,t242,t243,t244,t245,t246,t247,t248,t249,t250,
t251,t252,t253,t254,t255,t256,t257,t258,t259,t260,
t261,t262,t263,t264,t265,t266,t267,t268,t269,t270,
t271,t272,t273,t274,t275,t276,t277,t278,t279,t280,
t281,t282,t283,t284,t285,t286,t287,t288,t289,t290,
t291,t292,t293,t294,t295,t296,t297,t298,t299,t300,
t301,t302,t303,t304,t305,t306,t307,t308,t309,t310,
t311,t312,t313,t314,t315,t316,t317,t318,t319,t320,
t321,t322,t323,t324,t325,t326,t327,t328,t329,t330,
t331,t332,t333,t334,t335,t336,t337,t338,t339,t340,
t341,t342,t343,t344,t345,t346,t347,t348,t349,t350,
t351,t352,t353,t354,t355,t356,t357,t358,t359,t360,
t361,t362,t363,t364,t365,t366,t367,t368,t369,t370,
t371,t372,t373,t374,t375,t376,t377,t378,t379,t380,
t381,t382,t383,t384,t385,t386,t387,t388,t389,t390,
t391,t392,t393,t394,t395,t396,t397,t398,t399,t400,
t401,t402,t403,t404,t405,t406,t407,t408,t409,t410,
t411,t412,t413,t414,t415,t416,t417,t418,t419,t420,
t421,t422,t423,t424,t425,t426,t427,t428,t429,t430,
t431,t432,t433,t434,t435,t436,t437,t438,t439,t440,
t441,t442,t443,t444,t445,t446,t447,t448,t449,t450,
t451,t452,t453,t454,t455,t456,t457,t458,t459,t460,
t461,t462,t463,t464,t465,t466,t467,t468,t469,t470,
t471,t472,t473,t474,t475,t476,t477,t478,t479,t480,
t481,t482,t483,t484,t485,t486,t487,t488,t489,t490,
t491,t492,t493,t494,t495,t496,t497,t498,t499,t500,
t501,t502,t503,t504,t505,t506,t507,t508,t509,t510,
t511,t512,t513,t514,t515,t516,t517,t518,t519,t520,
t521,t522,t523,t524,t525,t526,t527,t528,t529,t530,
t531,t532,t533,t534,t535,t536,t537,t538,t539,t540,
t541,t542,t543,t544,t545,t546,t547,t548,t549,t550,
t551,t552,t553,t554,t555,t556,t557,t558,t559,t560,
t561,t562,t563,t564,t565,t566,t567,t568,t569,t570,
t571,t572,t573,t574,t575,t576,t577,t578,t579,t580,
t581,t582,t583,t584,t585,t586,t587,t588,t589,t590,
t591,t592,t593,t594,t595,t596,t597,t598,t599,t600,
t601,t602,t603,t604,t605,t606,t607,t608,t609,t610,
t611,t612,t613,t614,t615,t616,t617,t618,t619,t620,
t621,t622,t623,t624,t625,t626,t627,t628,t629,t630,
t631,t632,t633,t634,t635,t636,t637,t638,t639,t640,
t641,t642,t643,t644,t645,t646,t647,t648,t649,t650,
t651,t652,t653,t654,t655,t656,t657,t658,t659,t660,
t661,t662,t663,t664,t665,t666,t667,t668,t669,t670,
t671,t672,t673,t674,t675,t676,t677,t678,t679,t680,
t681,t682,t683,t684,t685,t686,t687,t688,t689,t690,
t691,t692,t693,t694,t695,t696,t697,t698,t699,t700,
t701,t702,t703,t704,t705,t706,t707,t708,t709,t710,
t711,t712,t713,t714,t715,t716,t717,t718,t719,t720,
t721,t722,t723,t724,t725,t726,t727,t728,t729,t730,
t731,t732,t733,t734,t735,t736,t737,t738,t739,t740,
t741,t742,t743,t744,t745,t746,t747,t748,t749,t750,
t751,t752,t753,t754,t755,t756,t757,t758,t759,t760,
t761,t762,t763,t764,t765,t766,t767,t768,t769,t770,
t771,t772,t773,t774,t775,t776,t777,t778,t779,t780,
t781,t782,t783,t784,t785,t786,t787,t788,t789,t790,
t791,t792,t793,t794,t795,t796,t797,t798,t799,t800,
t801,t802,t803,t804,t805,t806,t807,t808,t809,t810,
t811,t812,t813,t814,t815,t816,t817,t818,t819,t820,
t821,t822,t823,t824,t825,t826,t827,t828,t829,t830,
t831,t832,t833,t834,t835,t836,t837,t838,t839,t840,
t841,t842,t843,t844,t845,t846,t847,t848,t849,t850,
t851,t852,t853,t854,t855,t856,t857,t858,t859,t860,
t861,t862,t863,t864,t865,t866,t867,t868,t869,t870,
t871,t872,t873,t874,t875,t876,t877,t879,t880,
t881,t882,t883,t884,t885,t886,t887,t888,t889,t890,
t891,t892,t893,t894,t895,t896,t897,t898,t899,t900,
t901,t902,t903,t904,t905,t906,t907,t908,t909,t910,
t911,t912,t913,t914,t915,t916,t917,t918,t919,t920,
t921,t922,t923,t924,t925,t926,t927,t929,t930,
t931,t932,t933,t934,t935,t936,t937,t938,t939,t940,
t941,t942,t943,t944,t945,t946,t947,t948,t949,t950,
t951,t952,t953,t954,t955,t956,t957,t958,t959,t960,
t961,t962,t963,t964,t965,t966,t967,t968,t969,t970,
t971,t972,t973,t974,t975,t976,t977,t978,t979,t980,
t981,t982,t983,t984,t985,t986,t987,t988,t989,t990,
t991,t992,t993,t994,t995,t996,t997,t998,t999,t1000)

B.2 Transaction Monitor Parameters

tpccrc

CellLogVolume ecmllog
CellDataVolume ecmdat
NodeLogVolume enmllog

TpccApplicationDirectory/home/encina
TpccDbServer oratpcc.world
StatsFrequency 10
Version 1.0
Servers:delivery PAS 2 Threads 3 Name del IFS ---D-Dvry 2
Servers:lonline PAS 50 Threads 1 Name on1 IFS NPO-S Dvry 0

B.3 AIX Parameters

IBM eServer pSeries 690

OS PARAMETERS

keylock	normal	State of system keylock at boot time	False
maxbuf	20	Maximum number of pages in block I/O BUFFER CACHE	True
maxmbuf	0	Maximum Kbytes of real memory allowed for MBUFS	True
maxuproc	40000	Maximum number of PROCESSES allowed per user	True
autorestart	true	Automatically REBOOT system after a crash	True
iostat	false	Continuously maintain DISK I/O history	True
realmem	536870912	Amount of usable physical memory in Kbytes	False
conslogin	enable	System Console Login	False
fwversion	IBM, RG020712	Firmware version and revision levels	False
maxpout	0	HIGH water mark for pending write I/Os per file	True
minpout	0	LOW water mark for pending write I/Os per file	True
fullcore	false	Enable full CORE dump	True
pre430core	false	Use pre-430 style CORE dump	True
ncargs	20	ARG/ENV list size in 4K byte blocks	True
pre520tune	disable	Pre-520 tuning compatibility mode	True
rtasversion	1	Open Firmware RTAS version	False
modelname	IBM, 7040-681	Machine name	False
systemid	IBM, 010YHRH0G	Hardware system identifier	False
boottype	disk	N/A	False
SW_dist_intr	false	Enable SW distribution of interrupts	True
cpuguard	enable	CPU Guard	True
frequency	0	System Bus Frequency	False

vmo -o v_pinshm=1

vmo -r -o lgpg_size=16777216-o lgpg_regions=24000

chuser capabilities=CAP_BYPASS_RAC_VMM,CAP_PROPOGATE oracle

MEMORY_AFFINITY=MCM

REPLICATE_TEXT=/home/oracle/bin/oracle

DATA_SEG_SPECIAL=Y

Appendix C: Database Setup Code

C.1 Database Creation Scripts

addfile.sh

```
##
## addfile.sh 7030100.1 96/05/02 10:30:04plai Generic<base> $ Copyr (c) 1995 Oracle
##
#####
# Copyright (c) 1996 OracleCorp, 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>
#####*/
.setenv
FILE='basename $2'

if [ -d ./outdir ]
then
echo `date` > ./outdir/${FILE}.addf
fi

sqlsys <<!
set echo on
alter tablespace $1 add datafile '$2' size $3 reuse;
exit;
!

if [ -d ./outdir ]
then
echo `date` >> ./outdir/${FILE}.addf
fi
```

addfs.sh

```
#!/bin/ksh
#####
# Copyright (c) 1996 OracleCorp, Redwood Shores, CA |
# OPEN SYSTEMS PERFORMANCE GROUP |
# All Rights Reserved |
#####
# FILENAME
# addfs.sh
# DESCRIPTION
# Add tablespace to database.
# USAGE
# addfs.sh <tablespace> <data file> <size> <# of files>
#####*/

.setenv

tablespace=$1
datafile=$2
size=$3
nfiles=$4
let total=$nfiles+1

i=2
while [ $i -le $total ]
do
j=1
while [ $j -le 70 ]
do
if [ $i -le $total ]
then
echo "addfile.sh $tablespace $ {datafile} $i$size"
addfile.sh $tablespace $ {datafile} $i$size &
let i=$i+1
let j=$j+1
else
j=21
fi
done
wait
done
```

addroll.sh

```
#!/bin/ksh
#####
# Copyright (c) 1996 OracleCorp, Redwood Shores, CA |
# OPEN SYSTEMS PERFORMANCE GROUP |
# All Rights Reserved |
#####
# FILENAME
# addroll.sh
# DESCRIPTION
# Add tablespace to database.
# USAGE
# addroll.sh <data file> <size>
#####*/

.setenv
echo 'ORACLE_HOME=' $ORACLE_HOME
echo 'ORACLE_SID=' $ORACLE_SID

FILE='basename $1'

if [ -d ./outdir ]
then
echo `date` > ./outdir/${FILE}.addts
fi

# create tablespace roll datafile '$1' size $2 reuse extent management local uniform size 40Knologging ;

sqlsys <<!
create tablespace roll datafile '$1' size $2 reuseblocksize $3 nologging;

exit;
!

if [ -d ./outdir ]
then
echo `date` >> ./outdir/${FILE}.addts
fi
```

addts.sh

```
#!/bin/ksh
#####
# Copyright (c) 1996 OracleCorp, Redwood Shores, CA |
# OPEN SYSTEMS PERFORMANCE GROUP |
# All Rights Reserved |
#####
# FILENAME
# addts.sh
# DESCRIPTION
# Add tablespace to database.
# USAGE
# addts.sh <tablespace> <data file> <size>
#####*/

.setenv
FILE='basename $2'

if [ -d ./outdir ]
then
echo `date` > ./outdir/${FILE}.addts
fi

sqlsys <<!
create tablespace $1 datafile '$2' size $3 reuse extent management local uniform size $4nologging ;

exit;
!

if [ -d ./outdir ]
then
echo `date` >> ./outdir/${FILE}.addts
fi
```

alter temp.sh

```
#!/bin/ksh

# alter_temp 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle
#
#####
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA |
# OPEN SYSTEMS PERFORMANCE GROUP |
# All Rights Reserved |
#####
# NAME
# alter_temp.sh
# DESCRIPTION
# Usage: alter_temp.sh [options]
#####
```

```

#
.setenv

sqlplus system/manager <<!
alter user tpcc temporary tablespace temp;
quit;
!

sqlplus <<!
connect /as sysdba
alter tablespace temp
default storage (initial 20M next 20Mptcincrease 0);
# default storage (initial 5M next 5Mptcincrease 0);
exit;
!

benchdb.sh

#!/bin/ksh
#
# benchdb.sh 8030100 98/7/7 15:45vmakhija
# Copyr (c) 1998 Oracle
#
#=====+
# Copyright (c) 1997 OracleCorp, Redwood Shores, CA |
# OPEN SYSTEMS PERFORMANCE GROUP |
# All Rights Reserved |
#=====+
# FILENAME
# benchdb.sh
# DESCRIPTION
# Usage:benchdb.sh [options]
# -n do not create newtpcc database
# -c do not run catalog scripts
#=====+
#
.setenv

while [ "$#" != "0" ]
do
case $1 in
-n) shift
NO_CREATE="y"
;;
-c) shift
NO_CAT="y"
;;
*) echo "Bad arg: $1"
exit 1;
;;
esac
done

#
# Create database if NO_CREATE unset
#
if [ "$NO_CREATE" = "" ]
then
sqlsys <<!
set echo on
startup pfile=$TPCC_ADMIN/p_create.ora nomount
create database tpcc controlfile reuse maxdatafiles 1800
datafile '/dev/rtpc_lvsystem1' size 4159M reuse
DEFAULT TEMPORARYTABLESPACE temp
TEMPFILE'/dev/rtpc_lvtempl1' size 6271M reuse
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 16M
logfile '/dev/rtpc_redolog1' size 4000M reuse,
'/dev/rtpc_redolog2' size 4000M reuse;
exit
!
#
#
# Create more rollback segments
# Not in 9i
sqlsys <<!
create rollback segment s1 storage (initial 200kminextents 2 next 200k);
create rollback segment s2 storage (initial 200kminextents 2 next 200k);
create rollback segment s3 storage (initial 200kminextents 2 next 200k);
create rollback segment s4 storage (initial 200kminextents 2 next 200k);
create rollback segment s5 storage (initial 200kminextents 2 next 200k);
create rollback segment s6 storage (initial 200kminextents 2 next 200k);
create rollback segment s7 storage (initial 200kminextents 2 next 200k);
create rollback segment s8 storage (initial 200kminextents 2 next 200k);
create rollback segment s9 storage (initial 200kminextents 2 next 200k);
create rollback segment s10 storage (initial 200kminextents 2 next 200k);
create rollback segment s11 storage (initial 200kminextents 2 next 200k);
create rollback segment s12 storage (initial 200kminextents 2 next 200k);
create rollback segment s13 storage (initial 200kminextents 2 next 200k);
create rollback segment s14 storage (initial 200kminextents 2 next 200k);
create rollback segment s15 storage (initial 200kminextents 2 next 200k);
create rollback segment s16 storage (initial 200kminextents 2 next 200k);
create rollback segment s17 storage (initial 200kminextents 2 next 200k);
create rollback segment s18 storage (initial 200kminextents 2 next 200k);
create rollback segment s19 storage (initial 200kminextents 2 next 200k);

create rollback segment s20 storage (initial 200kminextents 2 next 200k);
create rollback segment s21 storage (initial 200kminextents 2 next 200k);
create rollback segment s22 storage (initial 200kminextents 2 next 200k);
create rollback segment s23 storage (initial 200kminextents 2 next 200k);
create rollback segment s24 storage (initial 200kminextents 2 next 200k);
create rollback segment s25 storage (initial 200kminextents 2 next 200k);
create rollback segment s26 storage (initial 200kminextents 2 next 200k);
create rollback segment s27 storage (initial 200kminextents 2 next 200k);
create rollback segment s28 storage (initial 200kminextents 2 next 200k);
create rollback segment s29 storage (initial 200kminextents 2 next 200k);
create rollback segment s30 storage (initial 200kminextents 2 next 200k);
create rollback segment s31 storage (initial 200kminextents 2 next 200k);
create rollback segment s32 storage (initial 200kminextents 2 next 200k);
create rollback segment s33 storage (initial 200kminextents 2 next 200k);
create rollback segment s34 storage (initial 200kminextents 2 next 200k);
create rollback segment s35 storage (initial 200kminextents 2 next 200k);
create rollback segment s36 storage (initial 200kminextents 2 next 200k);
create rollback segment s37 storage (initial 200kminextents 2 next 200k);
create rollback segment s38 storage (initial 200kminextents 2 next 200k);
create rollback segment s39 storage (initial 200kminextents 2 next 200k);
create rollback segment s40 storage (initial 200kminextents 2 next 200k);
create rollback segment s41 storage (initial 200kminextents 2 next 200k);
create rollback segment s42 storage (initial 200kminextents 2 next 200k);
create rollback segment s43 storage (initial 200kminextents 2 next 200k);
create rollback segment s44 storage (initial 200kminextents 2 next 200k);
create rollback segment s45 storage (initial 200kminextents 2 next 200k);
create rollback segment s46 storage (initial 200kminextents 2 next 200k);
create rollback segment s47 storage (initial 200kminextents 2 next 200k);
create rollback segment s48 storage (initial 200kminextents 2 next 200k);
create rollback segment s49 storage (initial 200kminextents 2 next 200k);
create rollback segment s50 storage (initial 200kminextents 2 next 200k);
create rollback segment s51 storage (initial 200kminextents 2 next 200k);
create rollback segment s52 storage (initial 200kminextents 2 next 200k);
create rollback segment s54 storage (initial 200kminextents 2 next 200k);
create rollback segment s55 storage (initial 200kminextents 2 next 200k);
create rollback segment s56 storage (initial 200kminextents 2 next 200k);
create rollback segment s57 storage (initial 200kminextents 2 next 200k);
create rollback segment s58 storage (initial 200kminextents 2 next 200k);
create rollback segment s59 storage (initial 200kminextents 2 next 200k);
create rollback segment s60 storage (initial 200kminextents 2 next 200k);
shutdown;
exit;
!

#
# Startup database with params file that includes new rollback segments
#

sqlsys <<!
startup pfile=$TPCC_ADMIN/p_build.ora;
exit;
!

#
#
#
# Add tablespaces in parallel
#
addrll.sh /dev/rtpc_lvroll1 3199M 16K&
addts.sh hist /dev/rtpc_lvhist1 5855M 1171M & # F=5 MAXE=100
addts.sh ware /dev/rtpc_lvware1 319M 3M & # F=533 MAXE=533
addts.sh cust /dev/rtpc_lvcust1 1983M 660M & # F=3 MAXE=1590
addts.sh items /dev/rtpc_lvitems1 31M 5M & # F=6 MAXE=6
addts.sh ord /dev/rtpc_lvord1 6271M 1254M & # F=5 MAXE=75
addts.sh nord /dev/rtpc_lvnord1 2111M 2110M & # F=1 MAXE=5
addts.sh ordl /dev/rtpc_lvordl1 6175M 1234M 16K & # F=5 MAXE=1400
addts.sh stocks /dev/rtpc_lvstk1 2623M 1311M & # F=2 MAXE=980
addts.sh icust1 /dev/rtpc_lvi1cust1 3327M 166M & # F=20MAXE=200
addts.sh icust2 /dev/rtpc_lvi2cust1 2527M 126M & # F=20MAXE=400
addts.sh istk /dev/rtpc_lvistk1 8863M 443M & # F=20 MAXE=200
addts.sh iord1 /dev/rtpc_lvi1ord1 2719M 135M & # F=20MAXE=400
addts.sh iord2 /dev/rtpc_lvi2ord1 1839M 91M & # F=20 MAXE=880
wait

#
#
# Add datafiles to tablespaces in parallel
#
addfs.sh roll /dev/rtpc_lvroll1 3199M 16 &
fi
addfs.sh hist /dev/rtpc_lvhist1 5855M 20 &

addfs.sh ware /dev/rtpc_lvware1 319M 6 &

addfs.sh cust /dev/rtpc_lvcust1 1983M 530 &
addfs.sh ord /dev/rtpc_lvord1 6271M 15 &

addfs.sh nord /dev/rtpc_lvnord1 2111M 5 &

addfs.sh ordl /dev/rtpc_lvordl1 6175M 280 &

addfs.sh stocks /dev/rtpc_lvstk1 2623M 490 &
wait

addfs.sh icust1 /dev/rtpc_lvi1cust1 3327M 10 &
addfs.sh icust2 /dev/rtpc_lvi2cust1 2527M 20 &

```



```

${LOAD_SCRIPTS}/create_ware.sh > ${OUTDIR}/create_ware.out 2>&1
fi

if runnable
then
  echo "Creating district ..."
${LOAD_SCRIPTS}/create_dist.sh > ${OUTDIR}/create_dist.out 2>&1
fi

if runnable
then
  echo "Creating history ..."
${LOAD_SCRIPTS}/create_hist.sh > ${OUTDIR}/create_hist.out 2>&1
fi

if runnable
then
  echo "Creating orders ..."
${LOAD_SCRIPTS}/create_ordr.sh > ${OUTDIR}/create_ordr.out 2>&1
fi

if runnable
then
  echo "Creating new-order ..."
${LOAD_SCRIPTS}/create_nord.sh > ${OUTDIR}/create_nord.out 2>&1
fi

if runnable
then
  echo "Creating order-line ..."
${LOAD_SCRIPTS}/create_ordl.sh > ${OUTDIR}/create_ordl.out 2>&1
fi

if runnable
then
  echo "Creating item ..."
${LOAD_SCRIPTS}/create_item.sh > ${OUTDIR}/create_item.out 2>&1
fi

if runnable
then
  echo "Creating customer ..."
${LOAD_SCRIPTS}/create_cust.sh > ${OUTDIR}/create_cust.out 2>&1 &
fi

if runnable
then
  echo "Creating stock ..."
${LOAD_SCRIPTS}/create_stok.sh > ${OUTDIR}/create_stok.out 2>&1 &
fi

wait

if runnable
then
  echo "Creating rollback segment ..."
${LOAD_SCRIPTS}/tpcc_rol.sh > ${OUTDIR}/tpcc_rol.out 2>&1 &
fi

echo "Switching Logs ..."
${TPCC_UTILS}/switchlog.sh >> ${OUTDIR}/switchlog.out 2>&1

if runnable
then
  echo "Loading new-order ..."
${LOAD_SCRIPTS}/load_nord.sh > ${OUTDIR}/load_nord.out 2>&1
fi

if runnable
then
  echo "Loading history ..."
${LOAD_SCRIPTS}/load_hist.sh > ${OUTDIR}/load_hist.out 2>&1
fi

if runnable
then
  echo "Loading orders and order-line ..."
${LOAD_SCRIPTS}/load_ordr.sh > ${OUTDIR}/load_ordr.out 2>&1
fi

echo "Switching Logs ..."
${TPCC_UTILS}/switchlog.sh >> ${OUTDIR}/switchlog.out 2>&1

if runnable
then
  echo "Loading warehouse ..."
${LOAD_SCRIPTS}/load_ware.sh > ${OUTDIR}/load_ware.out 2>&1
fi

if runnable
then
  echo "Loading district ..."
${LOAD_SCRIPTS}/load_dist.sh > ${OUTDIR}/load_dist.out 2>&1
fi

if runnable
then
  echo "Loading item ..."
${LOAD_SCRIPTS}/load_item.sh > ${OUTDIR}/load_item.out 2>&1

```

```

fi

if runnable
then
  echo "Loading customer ..."
${LOAD_SCRIPTS}/load_cust.sh > ${OUTDIR}/load_cust.out 2>&1
fi

if runnable
then
  echo "Loading stock ..."
${LOAD_SCRIPTS}/load_stok.sh > ${OUTDIR}/load_stok.out 2>&1
fi

wait

echo "Switching Logs ..."
${TPCC_UTILS}/switchlog.sh >> ${OUTDIR}/switchlog.out 2>&1

if runnable
then
  echo "Alter temp ..."
${LOAD_SCRIPTS}/alter_temp.sh > ${OUTDIR}/alter_temp.out 2>&1
fi

if runnable
then
  echo "Creating warehouse index ..."
${LOAD_SCRIPTS}/create_iware.sh > ${OUTDIR}/create_iware.out 2>&1
fi

if runnable
then
  echo "Creating district index ..."
${LOAD_SCRIPTS}/create_idist.sh > ${OUTDIR}/create_idist.out 2>&1
fi

if runnable
then
  echo "Creating item index ..."
${LOAD_SCRIPTS}/create_iitem.sh > ${OUTDIR}/create_iitem.out 2>&1
fi

if runnable
then
  echo "Creating customer index ..."
${LOAD_SCRIPTS}/create_icust.sh > ${OUTDIR}/create_icust.out 2>&1
fi

if runnable
then
  echo "Creating customer2 index ..."
${LOAD_SCRIPTS}/create_icust2.sh > ${OUTDIR}/create_icust2.out 2>&1
fi

if runnable
then
  echo "Creating stock index ..."
${LOAD_SCRIPTS}/create_istok.sh > ${OUTDIR}/create_istok.out 2>&1
fi

if runnable
then
  echo "Creating orders index ..."
${LOAD_SCRIPTS}/create_iordr.sh > ${OUTDIR}/create_iordr.out 2>&1
fi

if runnable
then
  echo "Creating orders2 index ..."
${LOAD_SCRIPTS}/create_iordr2.sh > ${OUTDIR}/create_iordr2.out 2>&1
fi

if runnable
then
  echo "No need to create inord"
# ${LOAD_SCRIPTS}/create_inord.sh > ${OUTDIR}/create_inord.out 2>&1
fi

if runnable
then
  echo "No need to create iordl"
# ${LOAD_SCRIPTS}/create_iordl.sh > ${OUTDIR}/create_iordl.out 2>&1
fi

if runnable
then
  echo "Re-alter temp ..."
${LOAD_SCRIPTS}/realter_temp.sh > ${OUTDIR}/realter_temp.out 2>&1
fi

if runnable
then
  echo "Analyze ..."
sqlplus tpcc/tpcc @$TPCC_SQL/tpcc_ana > ${OUTDIR}/tpcc_ana.out 2>&1
fi

```

```

if runnable
then
echo "Creating report tables ..."
${LOAD_SCRIPTS}/tpcc_reports.sh > ${OUTDIR}/tpcc_reports.out 2>&1
fi

if runnable
then
echo "Creating stored proc ..."
${LOAD_SCRIPTS}/tpcc_stored_proc.sh > ${OUTDIR}/tpcc_stored_prod.out 2>&1
${TPCC_UTILS}/create_cache_views.sh > ${OUTDIR}/create_cache_views.out 2>&1
fi

if runnable
then
echo "Space rpts / etc. ..."
${LOAD_SCRIPTS}/tpcc_misc.sh > ${OUTDIR}/tpcc_misc.out 2>&1
fi

if runnable
then
echo "Alter extents and lock ..."
# ${LOAD_SCRIPTS}/alter.sh > ${OUTDIR}/alter.out 2>&1
${LOAD_SCRIPTS}/altundef.sh > ${OUTDIR}/altundef.out 2>&1
${TPCC_UTILS}/dml.sh > ${OUTDIR}/dml.out 2>&1
fi

if runnable
then
echo "Running catalog scripts ..."
${LOAD_SCRIPTS}/cat.sh > ${OUTDIR}/cat.out 2>&1
fi

if runnable
then
sqlsys <<!
alter system switch logfile;
alter system switch logfile;
shutdown;
exit;
!
fi

date

```

cat.sh

```

#!/bin/ksh

# benchsetup 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle
#
#=====  

# Copyright (c) 1998 OracleCorp, Redwood Shores, CA |  

# OPEN SYSTEMS PERFORMANCE GROUP |  

# All Rights Reserved |  

#=====  

# NAME  

# benchsetup  

# DESCRIPTION  

# Usage: benchsetup.sh [options]  

#=====  

#  

. setenv

sqlsys <<!
set echo off;
@ ?/rdbms/admin/catparr;
exit;
!

```

create cust.sh

```

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#=====  

# Copyright (c) 1998 OracleCorp, Redwood Shores, CA |  

# OPEN SYSTEMS PERFORMANCE GROUP |  

# All Rights Reserved |  

#=====  

# FILENAME  

# create_cust.sh  

# DESCRIPTION  

# Usage: create_cust.sh  

#=====  

#  

. setenv

sqlplus tpcc/tpcc @cust

```

create dist.sh

```

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#=====  

# Copyright (c) 1998 OracleCorp, Redwood Shores, CA |  

# OPEN SYSTEMS PERFORMANCE GROUP |  

# All Rights Reserved |  

#=====  

# FILENAME  

# create_dist.sh  

# DESCRIPTION  

# Usage: create_dist.sh  

#=====  

#  

. setenv

sqlplus tpcc/tpcc @dist

```

create hist.sh

```

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#=====  

# Copyright (c) 1998 OracleCorp, Redwood Shores, CA |  

# OPEN SYSTEMS PERFORMANCE GROUP |  

# All Rights Reserved |  

#=====  

# FILENAME  

# create_hist.sh  

# DESCRIPTION  

# Usage: create_hist.sh  

#=====  

#  

. setenv

sqlplus tpcc/tpcc @hist

```

create icust.sh

```

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#=====  

# Copyright (c) 1998 OracleCorp, Redwood Shores, CA |  

# OPEN SYSTEMS PERFORMANCE GROUP |  

# All Rights Reserved |  

#=====  

# FILENAME  

# create_icust.sh  

# DESCRIPTION  

# Usage: create_icust.sh  

#=====  

#  

. setenv

sqlplus tpcc/tpcc @icust

```

create icust2.sh

```

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#=====  

# Copyright (c) 1998 OracleCorp, Redwood Shores, CA |  

# OPEN SYSTEMS PERFORMANCE GROUP |  

# All Rights Reserved |  

#=====  

# FILENAME  

# create_icust2.sh  

# DESCRIPTION  

# Usage: create_icust2.sh

```

```

#
=====
#
# .setenv
#
sqlplus tpcc/tpcc @icust2

                                create idist.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#-----+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
# OPEN SYSTEMS PERFORMANCE GROUP                          |
# All Rights Reserved                                     |
#-----+
# FILENAME
# create_idist.sh
# DESCRIPTION
# Usage: create_idist.sh
#
#-----+
#
# .setenv
#
sqlplus tpcc/tpcc @idist

                                create iitem.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#-----+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
# OPEN SYSTEMS PERFORMANCE GROUP                          |
# All Rights Reserved                                     |
#-----+
# FILENAME
# create_iitem.sh
# DESCRIPTION
# Usage: create_iitem.sh
#
#-----+
#
# .setenv
#
sqlplus tpcc/tpcc @iitem

                                create iordr.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#-----+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
# OPEN SYSTEMS PERFORMANCE GROUP                          |
# All Rights Reserved                                     |
#-----+
# FILENAME
# create_iordr.sh
# DESCRIPTION
# Usage: create_iordr.sh
#
#-----+
#
# .setenv
#
sqlplus tpcc/tpcc @iordr

                                create iordr2.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#-----+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
# OPEN SYSTEMS PERFORMANCE GROUP                          |
# All Rights Reserved                                     |
#-----+
# FILENAME

```

```

# create_iordr2.sh
# DESCRIPTION
# Usage: create_iordr2.sh
#
#-----+
#
# .setenv
#
sqlplus tpcc/tpcc @iordr2

                                create istok.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#-----+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
# OPEN SYSTEMS PERFORMANCE GROUP                          |
# All Rights Reserved                                     |
#-----+
# FILENAME
# create_istok.sh
# DESCRIPTION
# Usage: create_istok.sh
#
#-----+
#
# .setenv
#
sqlplus tpcc/tpcc @istok

                                create.item.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#-----+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
# OPEN SYSTEMS PERFORMANCE GROUP                          |
# All Rights Reserved                                     |
#-----+
# FILENAME
# create_item.sh
# DESCRIPTION
# Usage: create_item.sh
#
#-----+
#
# .setenv
#
sqlplus tpcc/tpcc @item

                                create iware.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#-----+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
# OPEN SYSTEMS PERFORMANCE GROUP                          |
# All Rights Reserved                                     |
#-----+
# FILENAME
# create_iware.sh
# DESCRIPTION
# Usage: create_iware.sh
#
#-----+
#
# .setenv
#
sqlplus tpcc/tpcc @iware

                                create nord.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#-----+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
# OPEN SYSTEMS PERFORMANCE GROUP                          |
# All Rights Reserved                                     |
#-----+
# FILENAME

```

```

# FILENAME
# create_nord.sh
# DESCRIPTION
# Usage: create_nord.sh
#
=====
#
.setenv
sqlplus tpcc/tpcc @nord

                                create ordl.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
=====
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#
=====
# FILENAME
# create_ordl.sh
# DESCRIPTION
# Usage: create_ordl.sh
#
=====
#
.setenv
sqlplus tpcc/tpcc @ordl

                                create ordr.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
=====
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#
=====
# FILENAME
# create_ordr.sh
# DESCRIPTION
# Usage: create_ordr.sh
#
=====
#
.setenv
sqlplus tpcc/tpcc @ordr

                                create stok.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
=====
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#
=====
# FILENAME
# create_stok.sh
# DESCRIPTION
# Usage: create_stok.sh
#
=====
#
.setenv
sqlplus tpcc/tpcc @stok

                                create user.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
=====
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#
=====

```

```

# All Rights Reserved
#
=====
# FILENAME
# create_user.sh
# DESCRIPTION
# Usage: create_user.sh
# createtpcc user on DB
#
=====
#
.setenv
#
# Copyright (c) 1997 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#
=====
# FILENAME
# tpcc_user.sql
# DESCRIPTION
# Create user for TPC-C database.
#
=====
#
# Create TPCC userid and connect to it.
#
sqlsys <<!
grant connect,resource,unlimited tablespace to tpcc identified by tpcc;
alter user tpcc temporary tablespace temp;
connect tpcc/tpcc;
exit;
!

                                create ware.sh

#!/bin/ksh
#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
=====
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#
=====
# FILENAME
# create_ware.sh
# DESCRIPTION
# Usage: create_ware.sh
#
=====
#
.setenv
sqlplus tpcc/tpcc @ware

                                dml.sh

#
# $Header: dml.sh 7030100.1 96/05/02 10:22:53lai Generic<base> $ Copyr (c) 1995 Oracle
#
#
=====
# Copyright (c) 1996 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#
=====
# FILENAME
# dml.sh
# DESCRIPTION
# Disable table locks for TPC-C tables.
# USAGE
# dml.sh
#
=====*/

sqlplus tpcc/tpcc <<!
alter table warehouse disable table lock;
alter table district disable table lock;
alter table customer disable table lock;
alter table history disable table lock;
alter table item disable table lock;
alter table stock disable table lock;
alter table orders disable table lock;
alter table new_order disable table lock;
alter table order_line disable table lock;
quit;
!

                                load cust.sh

#!/bin/ksh
# 80301 98/7/7 15:45vmakhija

```



```

# Copyright (c) 1998 Oracle
#
#====+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#====+
# NAME
# load_cust.sh
# DESCRIPTION
# Usage:load_cust.sh [options]
#====+
#
.setenv

#
# Load customer table (in parallel with loading stock table)
#
I=31
SW=6001
INC=200
let EW=$SW+$INC-1
J=4
X=$J
while [ $J -le 17 ]
do
let X=$J*10
while [ $I -le $X ]
do
echo "j = $J, x = $X, sw = $SW, ew = $EW"
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
wait
J=`expr $J + 1`
done
wait

```

load_dist.sh

```

#!/bin/ksh

# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle
#
#====+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#====+
# NAME
# load_dist.sh
# DESCRIPTION
# Usage:load_dist.sh [options]
#====+
#
.setenv

tpccload -M $MULT -d

```

load_hist.sh

```

#!/bin/ksh

# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
#====+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#====+
# FILENAME
# load_hist.sh
# DESCRIPTION
# Usage:load_hist.sh [options]
# -mu <multiplier> (# of warehouses)
#====+
#
.setenv

if echo "c" | grep c >/dev/null 2>&1; then
N='-n'
else
C='\c'
fi
export N C

while [ "$#" != "0" ]
do

```

```

case $I 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 "Bad 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 $BUILD_HOME ]
then
mkdir $BUILD_HOME
fi

if [ ! -d $LOAD_SCRIPTS ]
then
mkdir $LOAD_SCRIPTS
fi

if [ ! -d $LDIR ]
then
mkdir $LDIR
fi

if [ ! -d $OUTDIR ]
then
mkdir $OUTDIR
fi
#
# Load history table
#
I=1
SW=1
INC=200
let EW=$SW+$INC-1
J=1
X=$J
while [ $J -le 17 ]
do
let X=$J*10
while [ $I -le $X ]
do
echo "j = $J, x = $X, sw = $SW, ew = $EW"
tpccload -M $MULT -h -b $SW -e $EW > ${LDIR}/hist${I}.dat 2> ${OUTDIR}/hist${I}.out &
I=`expr $I + 1`
SW=`expr $SW + $INC`
EW=`expr $EW + $INC`
done
wait
J=`expr $J + 1`
done
wait

```

load_item.sh

```

#!/bin/ksh

# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle
#
#====+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
#====+
# NAME
# load_item.sh
# DESCRIPTION
# Usage:load_item.sh [options]

```

```

=====
#
# .setenv
tpccload -M $MULT -i

                                load_nord.sh

#!/bin/ksh

#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
=====+
#      Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
#      OPEN SYSTEMS PERFORMANCE GROUP                        |
#      All Rights Reserved                                     |
=====+
# FILENAME
# load_nord.sh
# DESCRIPTION
# Usage:load_nord.sh [options]
#      -mu <multiplier>  (# of warehouses)
#
=====
#
# .setenv

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 "Bad 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 $BUILD_HOME ]
then
  mkdir $BUILD_HOME
fi

if [ ! -d $LOAD_SCRIPTS ]
then
  mkdir $LOAD_SCRIPTS
fi

if [ ! -d $LDIR ]
then
  mkdir $LDIR
fi

if [ ! -d $OUTDIR ]
then
  mkdir $OUTDIR
fi

#
# Load new-order table
#

I=1
SW=1
INC=100

```

```

let EW=$SW+$INC-1
J=1
X=$J
while [ $J -le 17 ]
do
  let X=$J*20
  while [ $I -le $X ]
  do
    echo "j = $J, x = $X, sw = $SW, ew = $EW"
    tpccload -M $MULT -n -b $SW -e $EW > ${LDIR}/neword${I}.dat 2> ${OUTDIR}/neword${I}.out
    &
    I=`expr $I + 1`
    SW=`expr $SW + $INC`
    EW=`expr $EW + $INC`
  done
  wait
  J=`expr $J + 1`
done
wait

                                load_ordr.sh

#!/bin/ksh

#
# 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle Corp.
#
=====+
#      Copyright (c) 1998 OracleCorp, Redwood Shores, CA      |
#      OPEN SYSTEMS PERFORMANCE GROUP                        |
#      All Rights Reserved                                     |
=====+
# FILENAME
# load_ordr.sh
# DESCRIPTION
# Usage:load_ordr.sh [options]
#      -mu <multiplier>  (# of warehouses)
#
=====
#
# .setenv

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 "Bad 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 $BUILD_HOME ]
then
  mkdir $BUILD_HOME
fi

if [ ! -d $LOAD_SCRIPTS ]
then
  mkdir $LOAD_SCRIPTS
fi

if [ ! -d $LDIR ]
then
  mkdir $LDIR
fi

if [ ! -d $OUTDIR ]
then
  mkdir $OUTDIR
fi

#
# Load new-order table
#

I=1
SW=1
INC=100

```



```

STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t654
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t655
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t656
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t657
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t658
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t659
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t660
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t661
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t662
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t663
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t664
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t665
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t666
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t667
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t668
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t669
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t670
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t671
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t672
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t673
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t674
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t675
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t676
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t677
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t678
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t679
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t680
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t681
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t682
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t683
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t684
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t685
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t686
TABLESPACE roll

```

```

STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t687
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t688
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t689
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t690
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t691
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t692
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t693
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t694
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t695
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t696
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t697
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t698
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t699
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);
CREATE ROLLBACK SEGMENT t700
TABLESPACE roll
STORAGE (initial 100K next 100Kminextents 2);

```

exit;

tpcc stored_proc.sh

```

#!/bin/ksh

# tpcc_stored_proc 80301 98/7/7 15:45vmakhija
# Copyright (c) 1998 Oracle
#
# =====+
# Copyright (c) 1998 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
# =====+
# NAME
# tpcc_stored_proc.sh
# DESCRIPTION
# Usage:tpcc_stored_proc.sh [options]
# =====+
#
.setenv

sqlplus tpcc/tpcc @$TPCC_BLOCKS/views

# Oracle said to use payz.sql, it is better than pay.sql
#sqlplus tpcc/tpcc @$TPCC_BLOCKS/pay
#sqlplus tpcc/tpcc @$TPCC_BLOCKS/payz
# Sun is using initpay
sqlplus tpcc/tpcc @$TPCC_BLOCKS/initpay
# this is needed to do runs
sqlplus tpcc/tpcc @$TPCC_BLOCKS/tkvcin

```

undml.sh

```

#
# $Header: undml.sh 7030100.2 96/05/02 10:29:30plai Generic<base> $ Copyr (c) 1995 Oracle
#
# =====+
# Copyright (c) 1996 OracleCorp, Redwood Shores, CA
# OPEN SYSTEMS PERFORMANCE GROUP
# All Rights Reserved
# =====+
# FILENAME
# undml.sh
# DESCRIPTION
# Enable table locks forTPC-C tables.
# USAGE
# undml.sh

```

```
#=====*/
```

```
sqlplus tpcc/tpcc <<!  
alter table warehouse enable table lock;  
alter table district enable table lock;  
alter table customer enable table lock;  
alter table history enable table lock;  
alter table item enable table lock;  
alter table stock enable table lock;  
alter table orders enable table lock;  
alter table new_order enable table lock;  
alter table order_line enable table lock;  
quit;
```

C.2 SQL Scripts

cust.sql

```
rem
rem =====+
rem Copyright (c) 1996 OracleCorp, Redwood Shores, CA |
rem OPEN SYSTEMS PERFORMANCE GROUP |
rem All Rights Reserved |
rem =====+
rem FILENAME
rem cust.sql
rem DESCRIPTION
rem Create customer table for TPC-C database.
rem =====+
rem
rem
rem DROP all first
rem
rem drop cluster ccluster including tables;
rem 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(5,0)
)
single table
hashkeys 102000000
hash is ((c_id * 340000) + (c_w_id * 10) + c_d_id)
size 850
initrans 3
pctfree 0
tablespace cust
storage (buffer_pool recycle);

create table customer (
  c_id number(5,0),
  c_d_id number(2,0),
  c_w_id number(5,0),
  c_discount number,
  c_credit char(2),
  c_last varchar2(16),
  c_first varchar2(16),
  c_credit_lim number,
  c_balance number,
  c_ytd_payment number,
  c_payment_cnt number,
  c_delivery_cnt number,
  c_street_1 varchar2(20),
  c_street_2 varchar2(20),
  c_city varchar2(20),
  c_state char(2),
  c_zip char(9),
  c_phone char(16),
  c_since date,
  c_middle char(2),
  c_data varchar2(500)
)
cluster ccluster (c_id, c_d_id, c_w_id);

rem
rem done
rem
rem
rem exit;
```

dist.sql

```
rem
rem =====+
rem Copyright (c) 1996 OracleCorp, Redwood Shores, CA |
rem OPEN SYSTEMS PERFORMANCE GROUP |
rem All Rights Reserved |
rem =====+
rem FILENAME
rem dist.sql
rem DESCRIPTION
rem Create district table for TPC-C database.
rem =====+
rem
rem
rem DROP all first
rem
rem drop table district;
```

drop cluster dcluster including tables;

set timing on

```
rem
rem DISTRICT table
rem
create cluster dcluster (
  d_w_id number(5,0),
  d_id number(2,0)
)
single table
hashkeys 340000
hash is (d_w_id) * 10 + d_id
size 1536
initrans 3
pctfree 0
tablespace ware;

create table district (
  d_id number(2,0),
  d_w_id number(5,0),
  d_ytd number,
  d_tax number,
  d_next_o_id number,
  d_name varchar2(10),
  d_street_1 varchar2(20),
  d_street_2 varchar2(20),
  d_city varchar2(20),
  d_state char(2),
  d_zip char(9)
)
cluster dcluster (d_w_id, d_id);
```

```
rem
rem done
rem
```

exit;

hist.sql

```
rem
rem =====+
rem Copyright (c) 1996 OracleCorp, Redwood Shores, CA |
rem OPEN SYSTEMS PERFORMANCE GROUP |
rem All Rights Reserved |
rem =====+
rem FILENAME
rem hist.sql
rem DESCRIPTION
rem Create history table for TPC-C database.
rem =====+
rem
rem
rem DROP all first
rem
rem drop table history;

set timing on

rem
rem HISTORY table
rem
create table history (
  h_c_id number,
  h_c_d_id number,
  h_c_w_id number,
  h_d_id number,
  h_w_id number,
  h_date date,
  h_amount number,
  h_data varchar2(24)
)
tablespace hist
initrans 4
pctfree 5
storage (freelist groups 43 freelists 19 minextents 68
buffer_pool recycle );

rem
rem done
rem
rem
rem exit;
```

icust.sql

```
rem
```



```

alter table orders disable table lock;
rem
rem done
rem
    exit;

iordr2.sql

rem
rem =====+
rem      Copyright (c) 1996 OracleCorp, Redwood Shores, CA      |
rem      OPEN SYSTEMS PERFORMANCE GROUP                        |
rem      All Rights Reserved                                    |
rem =====+
rem FILENAME
rem iordr2.sql
rem DESCRIPTION
rem Create orders index 2 for TPC-C database.
rem =====+
rem
rem
rem DROP all first
rem
    alter table orders enable table lock;
    drop index iorders2;

set timing on

rem
rem ORDERS index 2
rem
rem storage (initial 1015M next 1015M pctincrease 0
rem          maxextents unlimited freelist groups 43 freelists 19)
rem create unique index iorders2 on orders(o_w_id, o_d_id, o_c_id, o_id)

create unique index iorders2 on orders(o_c_id, o_d_id, o_w_id, o_id)
tablespace iord2
initrans 4
        parallel 64
pctfree 25
storage (freelist groups 86 freelists 19);

    alter table orders disable table lock;
rem
rem done
rem
    exit;

istok.sql

rem
rem =====+
rem      Copyright (c) 1996 OracleCorp, Redwood Shores, CA      |
rem      OPEN SYSTEMS PERFORMANCE GROUP                        |
rem      All Rights Reserved                                    |
rem =====+
rem FILENAME
rem istok.sql
rem DESCRIPTION
rem Create stock index for TPC-C database.
rem =====+
rem
rem
rem DROP all first
rem
    drop index istock;

set timing on

rem
rem STOCK index
rem
rem create unique index istock on stock(s_i_id, s_w_id)
rem tablespace istk
rem initrans 3
rem          parallel 22
rem pctfree 1
rem storage (freelist groups 43 freelists 19);

rem
rem done
rem
    exit;

item.sql

```

```

rem
rem =====+
rem      Copyright (c) 1996 OracleCorp, Redwood Shores, CA      |
rem      OPEN SYSTEMS PERFORMANCE GROUP                        |
rem      All Rights Reserved                                    |
rem =====+
rem FILENAME
rem item.sql
rem DESCRIPTION
rem Create ITEM table for TPC-C database.
rem =====+
rem
rem
rem DROP item cluster and table
rem
    drop cluster icluster including tables;
    drop table item;

    set timing on;

rem
rem ITEM table
rem
rem create cluster icluster (
rem   i_id      number(6,0)
rem )
rem   single          table
rem hashkeys 100000
rem hash is   i_id
rem size     120
rem initrans 3
rem pctfree  0
rem tablespace items
rem          storage (buffer_pool keep);

rem create table item (
rem   i_id      number(6,0),
rem   i_name    varchar2(24),
rem   i_price   number,
rem   i_data    varchar2(50),
rem   i_im_id   number
rem )
rem cluster icluster(i_id);

rem
rem done
rem
    exit;

iware.sql

rem
rem =====+
rem      Copyright (c) 1996 OracleCorp, Redwood Shores, CA      |
rem      OPEN SYSTEMS PERFORMANCE GROUP                        |
rem      All Rights Reserved                                    |
rem =====+
rem FILENAME
rem iware.sql
rem DESCRIPTION
rem Create warehouse index for TPC-C database.
rem =====+
rem
rem
rem DROP all first
rem
rem drop index iwarehouse;

set timing on

rem
rem WAREHOUSE index
rem
rem create unique index iwarehouse on warehouse (w_id)
rem tablespace ware
rem initrans 3
rem          pctfree 1;

rem
rem done
rem
    exit;

nord.sql

rem
rem =====+
rem      Copyright (c) 1996 OracleCorp, Redwood Shores, CA      |

```

```

rem      OPEN SYSTEMS PERFORMANCE GROUP
rem      All Rights Reserved
=====+
rem FILENAME
rem      nord.sql
rem DESCRIPTION
rem      Create NEW_ORDER table for TPC-C database.
=====+
rem
rem
rem DROP all first
rem
rem      alter table new_order enable table lock;
rem      drop table new_order;

set timing on

rem
rem NEW_ORDER table
rem
create table new_order (
  no_w_id number,
  no_d_id number,
  no_o_id number,
  constraint inord primary key (no_w_id, no_d_id, no_o_id)
)
  organization index tablespace nord
initrans 4
pctfree 5
  storage ( freelist groups 43 freelists 19 minextents 5 );

rem      alter table new_order disable table lock;

rem
rem done
rem
exit;

```

ordl.sql

```

rem
=====+
rem      Copyright (c) 1996 OracleCorp, Redwood Shores, CA
rem      OPEN SYSTEMS PERFORMANCE GROUP
rem      All Rights Reserved
=====+
rem FILENAME
rem      ordl.sql
rem DESCRIPTION
rem      Create ORDER_LINE table for TPC-C database.
=====+
rem
rem
rem DROP all first
rem
rem      drop table order_line;

set timing on

rem
rem ORDER_LINE table
rem
create table order_line (
  ol_w_id number,
  ol_d_id number,
  ol_o_id number,
  ol_number number,
  ol_i_id number,
  ol_delivery_d date,
  ol_amount number,
  ol_supply_w_id number,
  ol_quantity number,
  ol_dist_info char(24),
  constraint iordl primary key (ol_w_id, ol_d_id, ol_o_id, ol_number)
)
  organization index tablespace ordl
initrans 4
pctfree 5
  storage ( freelist groups 43 freelists 19 minextents 1380 );

rem
rem done
rem
exit;

```

ordr.sql

```

rem
=====+
rem      Copyright (c) 1996 OracleCorp, Redwood Shores, CA
rem      OPEN SYSTEMS PERFORMANCE GROUP
rem      All Rights Reserved
=====+
rem FILENAME
rem      ordr.sql
rem DESCRIPTION
rem      Create orders table for TPC-C database.
=====+
rem
rem
rem DROP all first
rem
rem      drop table orders;

set timing on

rem
rem ORDERS table
rem
create table orders (
  o_id number,
  o_w_id number,
  o_d_id number,
  o_c_id number,
  o_carrier_id number,
  o_ol_cnt number,
  o_all_local number,
  o_entry_d date
)
  tablespace ord
initrans 4
pctfree 5
  storage ( freelist groups 43 freelists 19 minextents 65 );

rem
rem done
rem
exit;

```

stok.sql

```

rem
=====+
rem      Copyright (c) 1996 OracleCorp, Redwood Shores, CA
rem      OPEN SYSTEMS PERFORMANCE GROUP
rem      All Rights Reserved
=====+
rem FILENAME
rem      stok.sql
rem DESCRIPTION
rem      Create stock table for TPC-C database.
=====+
rem
rem
rem DROP all first
rem
rem      drop cluster scluster including tables;
rem      drop table stock;

set timing on

rem
rem STOCK table
rem
create cluster scluster (
  s_i_id number(6,0),
  s_w_id number(5,0)
)
  single table
hashkeys 3400000000
hash is (s_i_id * 34000 + s_w_id)
size 350
initrans 3
pctfree 0
tablespace stocks
  storage (freelist groups 43 freelists 19 buffer_pool keep);

create table stock (
  s_i_id number(6,0),
  s_w_id number(5,0),
  s_quantity number,
  s_ytd number,
  s_order_cnt number,
  s_remote_cnt number,
  s_data varchar2(50),
  s_dist_01 char(24),
  s_dist_02 char(24),
  s_dist_03 char(24),
  s_dist_04 char(24),
  s_dist_05 char(24),

```

```

s_dist_06 char(24),
s_dist_07 char(24),
s_dist_08 char(24),
s_dist_09 char(24),
s_dist_10 char(24)
)
cluster scluster (s_i_id, s_w_id);

rem
rem done
rem

exit;

```

tpcc ana.sql

```

rem
rem =====
rem Copyright (c) 1995 OracleCorp, 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 item 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;

```

ware.sql

```

rem
rem =====
rem Copyright (c) 1996 OracleCorp, Redwood Shores, CA
rem OPEN SYSTEMS PERFORMANCE GROUP
rem All Rights Reserved
rem =====
rem FILENAME
rem ware.sql
rem DESCRIPTION
rem Create warehouse table for TPC-C database.
rem =====
rem

```

```

rem
rem DROP all first
rem
drop table warehouse;
drop cluster wcluster including tables;

```

set timing on

```

rem
rem WAREHOUSE table
rem
create cluster wcluster (
    w_id number (5,0)
)
single table
hashkeys 34000
hash is w_id
size 1536
initrans 3
pctfree 0
tablespace ware;

create table warehouse (
    w_id number(5,0),

```

```

w_ytd number,
w_tax number,
w_name varchar2(10),
w_street_1 varchar2(20),
w_street_2 varchar2(20),
w_city varchar2(20),
w_state char(2),
w_zip char(9)
)
cluster wcluster (w_id);

```

```

rem
rem done
rem

```

exit;

C.3 Data Generation Code

tpccload.c

```

#ifndef RCSID
static char *RCSid =
    "$Header: tpccload.c 7030100.1 96/05/13 16:20:36plai Generic<base> $ Copyr (c) 1993 Oracle";
#endif /* RCSID */

```

```

/*
Copyright (c) 1994 OracleCorp, Redwood Shores, CA
OPEN SYSTEMS PERFORMANCE GROUP
All Rights Reserved
*/

```

```

FILENAME
tpccload.c
DESCRIPTION
Load or generate TPC-C database tables.
Usage: tpccload -M <# of warehouses> [options]
options: -A load all tables
        -w load warehouse table
        -d load district table
        -c load customer table
        -i load item table
        -s load stock table (cluster arounds_w_id)
        -S load stock table (cluster arounds_i_id)
        -h load history table
        -n load new-order table
        -o <oline file> load order and order-line table
        -b <ware#> beginning warehouse number
        -e <ware#> ending warehouse number
        -j <item#> beginning item number (with -S)
        -k <item#> ending item number (with -S)
        -g generate rows to standard output
*/

```

```

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

```

```

#define DISTARR 10 /* district insert array size */
#define CUSTARR 100 /* customer insert array size */
#define STOCARR 100 /* stock insert array size */
#define ITEMARR 100 /* item insert array size */
#define HISTARR 100 /* history insert array size */
#define ORDEARR 100 /* order insert array size */
#define NEWOARR 100 /* new order insert array size */

```

```

#define DISTFAC 10 /* max. district id */
#define CUSTFAC 3000 /* max. customer id */
#define STOCFAC 100000 /* max. stock id */
#define ITEMFAC 100000 /* max. item id */
#define HISTFAC 30000 /* history / warehouse */
#define ORDEFAC 3000 /* order / district */
#define NEWOFAC 900 /* new order / district */

```

```

#define C 0 /* constant in non-uniform dist. eqt. */
#define CNUM1 1 /* first constant in non-uniform dist. eqt. */
#define CNUM2 2 /* second constant in non-uniform dist. eqt. */
#define CNUM3 3 /* third constant in non-uniform dist. eqt. */

```

```

#define SEED 2 /* seed for random functions */

```

```

#define SQLTXTW "INSERT INTO warehouse (w_id, w_ytd, w_tax, w_name, w_street_1, w_street_2,
w_city, w_state, w_zip) VALUES (w_id, 30000000, w_tax, w_name, w_street_1, \
:w_street_2, w_city, w_state, w_zip)"

```

```

#define SQLTXTD "INSERT INTO district(d_id, d_w_id, d_ytd, d_tax, d_next_o_id, d_name, d_street_1,
d_street_2, d_city, d_state, d_zip) VALUES (:d_id, :d_w_id, 30000000, :d_tax, \
3001, :d_name, :d_street_1, :d_street_2, :d_city, :d_state, :d_zip)"

```

```

#define SQLTXTC "INSERT INTO customer(C_ID, C_D_ID, C_W_ID, C_FIRST, C_MIDDLE,
C_LAST, C_STREET_1, C_STREET_2, C_CITY, C_STATE, C_ZIP, C_PHONE, C_SINCE,
C_CREDIT, C_CREDIT_LIM, C_DISCOUNT, C_BALANCE, C_YTD_PAYMENT,

```


<pre> char d_street_1[10][21]; char d_street_2[10][21]; char d_city[10][21]; char d_state[10][2]; char d_zip[10][9]; float d_tax[10]; int c_id[100]; int c_d_id[100]; int c_w_id[100]; char c_first[100][17]; char c_last[100][17]; char c_street_1[100][21]; char c_street_2[100][21]; char c_city[100][21]; char c_state[100][2]; char c_zip[100][9]; char c_phone[100][16]; char c_credit[100][2]; float c_discount[100]; char c_data[100][501]; int i_id[100]; int i_im_id[100]; int i_price[100]; char i_name[100][25]; char i_data[100][51]; int s_i_id[100]; int s_w_id[100]; int s_quantity[100]; char s_dist_01[100][24]; char s_dist_02[100][24]; char s_dist_03[100][24]; char s_dist_04[100][24]; char s_dist_05[100][24]; char s_dist_06[100][24]; char s_dist_07[100][24]; char s_dist_08[100][24]; char s_dist_09[100][24]; char s_dist_10[100][24]; char s_data[100][51]; int h_w_id[100]; int h_d_id[100]; int h_c_id[100]; char h_data[100][25]; int o_id[100]; int o_d_id[100]; int o_w_id[100]; int o_c_id[100]; int o_carrier_id[100]; int o_ol_cnt[100]; int ol_o_id[15]; int ol_d_id[15]; int ol_w_id[15]; int ol_number[15]; int ol_i_id[15]; int ol_supply_w_id[15]; int ol_amount[15]; char ol_dist_info[15][24]; int no_o_id[100]; int no_d_id[100]; int no_w_id[100]; char sdate[30]; double begin_time, end_time; double begin_cpu, end_cpu; double gettime(), getcpu(); extern int getopt(); extern char *optarg; extern int optind, opterr; char *argstr="M:AwdcisShno:b:e:j:k:g"; int opt; int do_A=0; int do_w=0; int do_d=0; int do_i=0; int do_c=0; int do_s=0; int do_S=0; int do_h=0; int do_o=0; int do_n=0; int gen=0; int bware=1; int eware=0; int bitem=1; int eitem=0; FILE *olfp=NULL; char olfname[100]; #define FIRSTNAME_WITH_CLAST </pre>	<pre> #ifdef FIRSTNAME_WITH_CLAST char firstname_with_clast[100]; sprintf(firstname_with_clast, "C_LAST=%d", CNUM1); #endif /* FIRSTNAME_WITH_CLAST */ /*-----+ Parse command line -- look for scale factor. +-----*/ if (argc == 1) { myusage (); } while ((opt = getopt (argc, argv, argstr)) != -1) { switch (opt) { case '?':myusage (); break; case 'M': scale =atoi (optarg); break; case 'A':do_A = 1; break; case 'w':do_w = 1; break; case 'd':do_d = 1; break; case 'c':do_c = 1; break; case 'i':do_i = 1; break; case 's':do_s = 1; break; case 'S':do_S = 1; break; case 'h':do_h = 1; break; case 'n':do_n = 1; break; case 'o':do_o = 1; strcpy (olfname, optarg); break; case 'b':bware = atoi (optarg); break; case 'e':eware = atoi (optarg); break; case 'j':bitem = atoi (optarg); break; case 'k':eitem = atoi (optarg); break; case 'g':gen = 1; break; default:fprintf (stderr, "THIS SHOULD NEVER HAPPEN!!!\n"); fprintf (stderr, "(reached default case in getopt ())\n"); myusage (); } } /*-----* Rudimentary error checking +-----*/ if (scale < 1) { fprintf (stderr, "Invalid scale factor:%d\n", scale); myusage (); } if (!(do_A do_w do_d do_c do_i do_s do_S do_h do_o do_n)) { fprintf (stderr, "What should I load???\n"); myusage (); } if (gen && (do_A (do_w + do_d + do_c + do_i + do_s + do_S + do_h + do_o + do_n > 1))) { fprintf (stderr, "Can only generate table one at a time\n"); myusage (); } if (do_S && (do_A do_s)) { fprintf (stderr, "Cluster stock table around s_w_id or s_i_id?\n"); myusage (); } if (eware <= 0) eware = scale; if (eitem <= 0) eitem = STOCFAC; if (do_S) { if ((bitem < 1) (bitem > STOCFAC)) { fprintf (stderr, "Invalid beginning item number:%d\n", bitem); myusage (); } if ((eitem < bitem) (eitem > STOCFAC)) { fprintf (stderr, "Invalid ending item number:%d\n", eitem); myusage (); } } </pre>
---	--


```

errprt (&tpclda, &curv);
quit ();
exit (1);
}

sprintf ((char *) sqlbuf, SQLXTXD);
if (oparse (&curi, sqlbuf, -1, 0, 1)) {
errprt (&tpclda, &curi);
quit ();
exit (1);
}

sprintf ((char *) sqlbuf, SQLXTXO1);
if (oparse (&curol, sqlbuf, -1, 0, 1)) {
errprt (&tpclda, &curol);
quit ();
exit (1);
}

sprintf ((char *) sqlbuf, SQLXTXO2);
if (oparse (&curol, sqlbuf, -1, 0, 1)) {
errprt (&tpclda, &curol);
quit ();
exit (1);
}

sprintf ((char *) sqlbuf, SQLXTXO1);
if (oparse (&curol1, sqlbuf, -1, 0, 1)) {
errprt (&tpclda, &curol1);
quit ();
exit (1);
}

sprintf ((char *) sqlbuf, SQLXTXO2);
if (oparse (&curol2, sqlbuf, -1, 0, 1)) {
errprt (&tpclda, &curol2);
quit ();
exit (1);
}

sprintf ((char *) sqlbuf, SQLXTXNO);
if (oparse (&curno, sqlbuf, -1, 0, 1)) {
errprt (&tpclda, &curno);
quit ();
exit (1);
}

/* bind variables */

/* warehouse */

if (obndrv (&curw, (text *) "w_id", -1, (ub1 *) &w_id, sizeof (w_id),
SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curw);
quit ();
exit (1);
}

if (obndrv (&curw, (text *) "w_name", -1, (ub1 *) w_name, 11,
SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curw);
quit ();
exit (1);
}

if (obndrv (&curw, (text *) "w_street_1", -1, (ub1 *) w_street_1, 21,
SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curw);
quit ();
exit (1);
}

if (obndrv (&curw, (text *) "w_street_2", -1, (ub1 *) w_street_2, 21,
SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curw);
quit ();
exit (1);
}

if (obndrv (&curw, (text *) "w_city", -1, (ub1 *) w_city, 21,
SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curw);
quit ();
exit (1);
}

if (obndrv (&curw, (text *) "w_state", -1, (ub1 *) w_state, 2,
SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curw);
quit ();
exit (1);
}

if (obndrv (&curw, (text *) "w_zip", -1, (ub1 *) w_zip, 9,
SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curw);
quit ();
exit (1);
}

```

```

if (obndrv (&curw, (text *) "w_tax", -1, (ub1 *) &w_tax, sizeof (w_tax),
SQLT_FLT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curw);
quit ();
exit (1);
}

/* district */

if (obndrv (&curd, (text *) "d_id", -1, (ub1 *) d_id, sizeof (int),
SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curd);
quit ();
exit (1);
}

if (obndrv (&curd, (text *) "d_w_id", -1, (ub1 *) d_w_id, sizeof (int),
SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curd);
quit ();
exit (1);
}

if (obndrv (&curd, (text *) "d_name", -1, (ub1 *) d_name, 11,
SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curd);
quit ();
exit (1);
}

if (obndrv (&curd, (text *) "d_street_1", -1, (ub1 *) d_street_1, 21,
SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curd);
quit ();
exit (1);
}

if (obndrv (&curd, (text *) "d_street_2", -1, (ub1 *) d_street_2, 21,
SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curd);
quit ();
exit (1);
}

if (obndrv (&curd, (text *) "d_city", -1, (ub1 *) d_city, 21,
SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curd);
quit ();
exit (1);
}

if (obndrv (&curd, (text *) "d_state", -1, (ub1 *) d_state, 2,
SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curd);
quit ();
exit (1);
}

if (obndrv (&curd, (text *) "d_zip", -1, (ub1 *) d_zip, 9,
SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curd);
quit ();
exit (1);
}

if (obndrv (&curd, (text *) "d_tax", -1, (ub1 *) d_tax, sizeof (int),
SQLT_FLT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curd);
quit ();
exit (1);
}

/* customer */

if (obndrv (&curc, (text *) "c_id", -1, (ub1 *) c_id, sizeof (int),
SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curc);
quit ();
exit (1);
}

if (obndrv (&curc, (text *) "c_d_id", -1, (ub1 *) c_d_id, sizeof (int),
SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curc);
quit ();
exit (1);
}

if (obndrv (&curc, (text *) "c_w_id", -1, (ub1 *) c_w_id, sizeof (int),
SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curc);
quit ();
exit (1);
}

if (obndrv (&curc, (text *) "c_first", -1, (ub1 *) c_first, 17,
SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
errprt (&tpclda, &curc);

```


<pre> quit (); exit (1); } if (obndrv (&curc, (text *) ":c_last", -1, (ub1 *) c_last, 17, SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":c_street_1", -1, (ub1 *) c_street_1, 21, SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":c_street_2", -1, (ub1 *) c_street_2, 21, SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":c_city", -1, (ub1 *) c_city, 21, SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":c_state", -1, (ub1 *) c_state, 2, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":c_zip", -1, (ub1 *) c_zip, 9, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":c_phone", -1, (ub1 *) c_phone, 16, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":c_credit", -1, (ub1 *) c_credit, 2, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":c_discount", -1, (ub1 *) c_discount, sizeof (int), SQLT_FLT, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":c_data", -1, (ub1 *) c_data, 501, SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } /* item */ if (obndrv (&curi, (text *) ":i_id", -1, (ub1 *) i_id, sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curi); quit (); exit (1); } if (obndrv (&curi, (text *) ":i_im_id", -1, (ub1 *) i_im_id, sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curi); quit (); exit (1); } if (obndrv (&curi, (text *) ":i_name", -1, (ub1 *) i_name, 25, SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curi); quit (); exit (1); } if (obndrv (&curi, (text *) ":i_price", -1, (ub1 *) i_price, </pre>	<pre> sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curi); quit (); exit (1); } if (obndrv (&curi, (text *) ":i_data", -1, (ub1 *) i_data, 51, SQLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curi); quit (); exit (1); } /* stock */ if (obndrv (&curc, (text *) ":s_i_id", -1, (ub1 *) s_i_id, sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_w_id", -1, (ub1 *) s_w_id, sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_quantity", -1, (ub1 *) s_quantity, sizeof (int), SQLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_dist_01", -1, (ub1 *) s_dist_01, 24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_dist_02", -1, (ub1 *) s_dist_02, 24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_dist_03", -1, (ub1 *) s_dist_03, 24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_dist_04", -1, (ub1 *) s_dist_04, 24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_dist_05", -1, (ub1 *) s_dist_05, 24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_dist_06", -1, (ub1 *) s_dist_06, 24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_dist_07", -1, (ub1 *) s_dist_07, 24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_dist_08", -1, (ub1 *) s_dist_08, 24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } if (obndrv (&curc, (text *) ":s_dist_09", -1, (ub1 *) s_dist_09, 24, SQLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) { errprt (&tpclda, &curc); quit (); exit (1); } </pre>
---	---

```

if(obndrv (&curs, (text *) "s_dist_10", -1, (ub1 *) s_dist_10, 24,
    SFLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curs);
    quit ();
    exit (1);
}

if(obndrv (&curs, (text *) "s_data", -1, (ub1 *) s_data, 51,
    SFLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curs);
    quit ();
    exit (1);
}

/* history */

if(obndrv (&curh, (text *) "h_c_id", -1, (ub1 *) h_c_id, sizeof (int),
    SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curh);
    quit ();
    exit (1);
}

if(obndrv (&curh, (text *) "h_c_d_id", -1, (ub1 *) h_d_id, sizeof (int),
    SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curh);
    quit ();
    exit (1);
}

if(obndrv (&curh, (text *) "h_c_w_id", -1, (ub1 *) h_w_id, sizeof (int),
    SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curh);
    quit ();
    exit (1);
}

if(obndrv (&curh, (text *) "h_d_id", -1, (ub1 *) h_d_id, sizeof (int),
    SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curh);
    quit ();
    exit (1);
}

if(obndrv (&curh, (text *) "h_w_id", -1, (ub1 *) h_w_id, sizeof (int),
    SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curh);
    quit ();
    exit (1);
}

if(obndrv (&curh, (text *) "h_data", -1, (ub1 *) h_data, 25,
    SFLT_STR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curh);
    quit ();
    exit (1);
}

/* order_line (delivered) */

if(obndrv (&curol1, (text *) "ol_o_id", -1, (ub1 *) ol_o_id,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

if(obndrv (&curol1, (text *) "ol_d_id", -1, (ub1 *) ol_d_id,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

if(obndrv (&curol1, (text *) "ol_w_id", -1, (ub1 *) ol_w_id,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

if(obndrv (&curol1, (text *) "ol_number", -1, (ub1 *) ol_number,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

if(obndrv (&curol1, (text *) "ol_i_id", -1, (ub1 *) ol_i_id,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

if(obndrv (&curol1, (text *) "ol_supply_w_id", -1,
    (ub1 *) ol_supply_w_id, sizeof (int), SFLT_INT, -1,
    (sb2 *) 0, (text *) 0, -1, -1)) {
}

```

```

errprt (&tpclda, &curol1);
quit ();
exit (1);
}

if(obndrv (&curol1, (text *) "ol_dist_info", -1, (ub1 *) ol_dist_info,
    24, SFLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

/* order_line (not delivered) */

if(obndrv (&curol2, (text *) "ol_o_id", -1, (ub1 *) ol_o_id,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol2);
    quit ();
    exit (1);
}

if(obndrv (&curol2, (text *) "ol_d_id", -1, (ub1 *) ol_d_id,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol2);
    quit ();
    exit (1);
}

if(obndrv (&curol2, (text *) "ol_w_id", -1, (ub1 *) ol_w_id,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol2);
    quit ();
    exit (1);
}

if(obndrv (&curol2, (text *) "ol_number", -1, (ub1 *) ol_number,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol2);
    quit ();
    exit (1);
}

if(obndrv (&curol2, (text *) "ol_i_id", -1, (ub1 *) ol_i_id,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol2);
    quit ();
    exit (1);
}

if(obndrv (&curol2, (text *) "ol_supply_w_id", -1,
    (ub1 *) ol_supply_w_id, sizeof (int), SFLT_INT, -1,
    (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol2);
    quit ();
    exit (1);
}

if(obndrv (&curol2, (text *) "ol_amount", -1, (ub1 *) ol_amount,
    sizeof (int), SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol2);
    quit ();
    exit (1);
}

if(obndrv (&curol2, (text *) "ol_dist_info", -1, (ub1 *) ol_dist_info,
    24, SFLT_CHR, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol2);
    quit ();
    exit (1);
}

/* orders (delivered) */

if(obndrv (&curol1, (text *) "o_id", -1, (ub1 *) o_id, sizeof (int),
    SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

if(obndrv (&curol1, (text *) "o_d_id", -1, (ub1 *) o_d_id, sizeof (int),
    SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

if(obndrv (&curol1, (text *) "o_w_id", -1, (ub1 *) o_w_id, sizeof (int),
    SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

if(obndrv (&curol1, (text *) "o_c_id", -1, (ub1 *) o_c_id, sizeof (int),
    SFLT_INT, -1, (sb2 *) 0, (text *) 0, -1, -1)) {
    errprt (&tpclda, &curol1);
    quit ();
    exit (1);
}

```



```

cidid++; /* shift district cycle*/
if (cidid > DISTFAC) {
    cidid = 1;
    cwid++; /* shift warehouse cycle*/
}
}
h_c_id[i] = cid;
h_d_id[i] = cidid;
h_w_id[i] = cwid;
randstr(h_data[i], 12, 24);
if(gen) {
    printf("%d %d %d %d %d %s 1000 %s\n", cid, cidid, cwid, cidid,
        cwid, sdate, h_data[i]);
}
}

if(gen) {
    fflush(stdout);
}
else {
    if(oexn (&curh, HISTARR, 0)) {
        errprt (&tpclda, &curh);
        orol (&tpclda);
        fprintf(stderr, "Aborted at w_id %d, d_id %d, c_id %d\n",
            h_w_id[0], h_d_id[0], h_c_id[0]);
        quit ();
        exit (1);
    }
    else if(ocom (&tpclda)) {
        errprt (&tpclda, &tpclda);
        orol (&tpclda);
        fprintf(stderr, "Aborted at w_id %d, d_id %d, c_id %d\n",
            h_w_id[0], h_d_id[0], h_c_id[0]);
        quit ();
        exit (1);
    }
}

if(++loopcount % 50)
    fprintf(stderr, ".");
else
    fprintf(stderr, "%d rows committed\n", row);
}

end_time = gettimeofday ();
end_cpu = getcpu ();
fprintf(stderr, "Done. %d rows loaded/generated in %10.2f sec. (%10.2fcpu)\n\n",
    nrows, end_time - begin_time, end_cpu - begin_cpu);
}

/*-----+
| Load the ORDERS and ORDER-LINE table. |
+-----*/

if(do_A || do_o) {
    nrows = (eware - bware + 1) * ORDEFAC * DISTFAC;

    fprintf(stderr, "Loading/generating orders and order-line: w%d - w%d (%d ord, ~%d ordl)\n",
        bware, eware, nrows, nrows * 10);

    begin_time = gettimeofday ();
    begin_cpu = getcpu ();

    cid = 0;
    cidid = 1;
    cwid = bware;
    loopcount = 0;

    for(row = 0; row < nrows; ) {
        for(i = 0; i < ORDEARR; i++, row++) {
            cid++;
            if (cid > ORDEFAC) { /* cycle cust id*/
                cid = 1; /* cheap mod*/
                cidid++; /* shift district cycle*/
                if (cidid > DISTFAC) {
                    cidid = 1;
                    cwid++; /* shift warehouse cycle*/
                }
            }
            o_carrier_id[i] = rand () % 10 + 1;
            o_ol_cnt[i] = olcnt = rand () % 11 + 5;

            if(gen) {
                if (cid < 2101) {
                    printf("%d %d %d %d %s %d %d 1\n", cid, cidid, cwid,
                        randperm3000[cid - 1], sdate, o_carrier_id[i],
                        o_ol_cnt[i]);
                }
                else {
                    /* set carrierid to 11 instead of null*/
                    printf("%d %d %d %d %s 11 %d 1\n", cid, cidid, cwid,
                        randperm3000[cid - 1], sdate, o_ol_cnt[i]);
                }
            }
            else {
                o_id[i] = cid;
                o_d_id[i] = cidid;
                o_w_id[i] = cwid;
                o_c_id[i] = randperm3000[cid - 1];
            }
        }
    }
}

```

```

}

for (j = 0; j < o_ol_cnt[i]; j++) {
    ol_i_id[j] = sid = lrand48 () % 100000 + 1;
    if (cid < 2101)
        ol_amount[j] = 0;
    else
        ol_amount[j] = (lrand48 () % 999999 + 1);
    randstr (str24[j], 24, 24);

    if(gen) {
        if (cid < 2101) {
            fprintf(olfp, "%d %d %d %d %s %d %d 5 %ld %s\n",
                cidid, cwid, j + 1, sdate, ol_i_id[j], cwid,
                ol_amount[j], str24[j]);
        }
        else {
            /* Insert a default date instead of null date*/
            fprintf(olfp, "%d %d %d %d 01-Jan-1811 %d %d 5 %ld %s\n",
                cidid, cwid, j + 1, ol_i_id[j], cwid,
                ol_amount[j], str24[j]);
        }
    }
    else {
        ol_o_id[j] = cid;
        ol_d_id[j] = cidid;
        ol_w_id[j] = cwid;
        ol_number[j] = j + 1;
        ol_supply_w_id[j] = cwid;
        strncpy (ol_dist_info[j], str24[j], 24);
    }
}

if(gen) {
    fflush(olfp);
}
else {
    if (cid < 2101) {
        if(oexn (&curol1, olcnt, 0)) {
            errprt (&tpclda, &curol1);
            orol (&tpclda);
            fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                cwid, cidid, cid);
            quit ();
            exit (1);
        }
        else if(ocom (&tpclda)) {
            errprt (&tpclda, &tpclda);
            orol (&tpclda);
            fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                cwid, cidid, cid);
            quit ();
            exit (1);
        }
    }
    else {
        if(oexn (&curol2, olcnt, 0)) {
            errprt (&tpclda, &curol2);
            orol (&tpclda);
            fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                cwid, cidid, cid);
            quit ();
            exit (1);
        }
        else if(ocom (&tpclda)) {
            errprt (&tpclda, &tpclda);
            orol (&tpclda);
            fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                cwid, cidid, cid);
            quit ();
            exit (1);
        }
    }
}

if(gen) {
    fflush(stdout);
}
else {
    if (cid < 2101) {
        if(oexn (&curol, ORDEARR, 0)) {
            errprt (&tpclda, &curol);
            orol (&tpclda);
            fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                cwid, cidid, cid);
            quit ();
            exit (1);
        }
        else if(ocom (&tpclda)) {
            errprt (&tpclda, &tpclda);
            orol (&tpclda);
            fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                cwid, cidid, cid);
            quit ();
            exit (1);
        }
    }
    else {
        if(oexn (&curol, ORDEARR, 0)) {
            errprt (&tpclda, &curol);
            orol (&tpclda);
            fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                cwid, cidid, cid);
            quit ();
            exit (1);
        }
        else if(ocom (&tpclda)) {
            errprt (&tpclda, &tpclda);
            orol (&tpclda);
            fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n",
                cwid, cidid, cid);
            quit ();
            exit (1);
        }
    }
}
}
else {
}
}

```

<pre> if(oexn (&curo2, ORDEARR, 0)) { errprt (&tpclda, &curo2); orol (&tpclda); fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n ", cwid, cdid, cid); quit (); exit (1); } else if(ocom (&tpclda)) { errprt (&tpclda, &tpclda); orol (&tpclda); fprintf(stderr, "Aborted at w_id %d, d_id %d, o_id %d\n ", cwid, cdid, cid); quit (); exit (1); } } if(++loopcount % 50) fprintf(stderr, "."); else fprintf(stderr, "%d orders committed\n ", row); } end_time = gettime (); end_cpu = getcpu (); fprintf(stderr, "Done. %d orders loaded/generated in %10.2f sec. (%10.2fcpu)\n\n", nrows, end_time - begin_time, end_cpu - begin_cpu); } /*-----+ Load the NEW-ORDER table. +-----*/ if(do_A do_n) { nrows = (eware - bware + 1) * NEWOFAC * DISTFAC; fprintf(stderr, "Loading/generating new-order: w%d - w%d (%d rows)\n ", bware, aware, nrows); begin_time = gettime (); begin_cpu = getcpu (); cid = 0; cdid = 1; cwid = bware; loopcount = 0; for (row = 0; row < nrows;) { for (i = 0; i < NEWOARR; i++, row++) { cid++; if (cid > NEWOFAC) { cid = 1; cdid++; if (cdid > DISTFAC) { cdid = 1; cwid++; } } if (gen) { printf ("%d %d %d\n", cid + 2100, cdid, cwid); } else { no_o_id[i] = cid + 2100; no_d_id[i] = cdid; no_w_id[i] = cwid; } } } if (gen) { fflush (stdout); } else { if (oexn (&curno, NEWOARR, 0)) { errprt (&tpclda, &curno); orol (&tpclda); fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n ", cwid, cdid, cid + 2100); quit (); exit (1); } else if (ocom (&tpclda)) { errprt (&tpclda, &tpclda); orol (&tpclda); fprintf (stderr, "Aborted at w_id %d, d_id %d, o_id %d\n ", cwid, cdid, cid + 2100); quit (); exit (1); } } if (++loopcount % 45) fprintf (stderr, "."); else fprintf (stderr, "%d rows committed\n ", row); } </pre>	<pre> end_time = gettime (); end_cpu = getcpu (); fprintf (stderr, "Done. %d rows loaded/generated in %10.2f sec. (%10.2fcpu)\n\n", nrows, end_time - begin_time, end_cpu - begin_cpu); } /*-----+ clean up and exit. +-----*/ if (olfp) fclose (olfp); if (!gen) quit (); exit (0); } initperm () { int i; int pos; int temp; /* init randperm3000 */ for (i = 0; i < 3000; i++) randperm3000[i] = i + 1; for (i = 3000; i > 0; i--) { pos = rand () % i; temp = randperm3000[i - 1]; randperm3000[i - 1] = randperm3000[pos]; randperm3000[pos] = temp; } } randstr (str, x, y) char *str; int x; int y; { int i, j; int len; len = (rand () % (y - x + 1)) + x; for (i = 0; i < len; i++) { j = rand () % 62; if (j < 26) str[i] = (char) (j + 'a'); else if (j < 52) str[i] = (char) (j - 26 + 'A'); else str[i] = (char) (j - 52 + '0'); } str[len] = '\0'; } randdatastr (str, x, y) char *str; int x; int y; { int i, j; int len; int pos; len = (rand () % (y - x + 1)) + x; for (i = 0; i < len; i++) { j = rand () % 62; if (j < 26) str[i] = (char) (j + 'a'); else if (j < 52) str[i] = (char) (j - 26 + 'A'); else str[i] = (char) (j - 52 + '0'); } str[len] = '\0'; if ((rand () % 10) == 0) { pos = (rand () % (len - 8)); str[pos] = 'O'; str[pos + 1] = 'R'; str[pos + 2] = 'T'; str[pos + 3] = 'G'; } } </pre>
---	--

```
    str[pos + 4] = 'T';
    str[pos + 5] = 'N';
    str[pos + 6] = 'A';
    str[pos + 7] = 'L';
}

}

randnum (str, len)

char *str;
int len;

{

    int i;

    for (i = 0; i < len; i++)
        str[i] = (char) (rand () % 10 + '0');
    str[len] = '\0';

}

randlastname (str, id)

char *str;
int id;

{

    id = id % 1000;
    strcpy (str, lastname[id / 100]);
    strcat (str, lastname[(id / 10) % 10]);
    strcat (str, lastname[id % 10]);

}

NURand (A, x, y, cnum)

int A, x, y, cnum;

{

    int a, b;

    a = lrand48 () % (A + 1);
    b = (lrand48 () % (y - x + 1)) + x;
    return (((a | b) + cnum) % (y - x + 1)) + x;

}

sysdate (sdate)

char *sdate;

{

    time_t tp;
    struct tm *tmptr;

    time (&tp);
    tmptr = localtime (&tp);
    strftime (sdate, 29, "%d-%b-%Y", tmptr);

}

}
```


Appendix D: RTE Scripts

D.1 RTE Parameters

```
/* For Oracle in the tpcload program C_LAST=1. */
/* C-Delta be the difference between C-LOAD and C-Run. */
/* C-Delta must be a value between 65..119 including the */
/* values of 65 and 119 and excluding the value of 96 and 112*/

#define MASTER_NUM1 1
#define MASTER_NUM2 0
#define MASTER_NUM3 0
#define MASTER_NUM4 0
#define MASTER_NUM5 0
#define MASTER_NUM6 0
#define MASTER_NUM7 0

#if MASTER_NUM1
MASTER "master1"
#endif
#if MASTER_NUM2
MASTER "master2"
#endif
#if MASTER_NUM3
MASTER "master3"
#endif
#if MASTER_NUM4
MASTER "master4"
#endif
#if MASTER_NUM5
MASTER "master5"
#endif
#if MASTER_NUM6
MASTER "master6"
#endif
#if MASTER_NUM7
MASTER "master7"
#endif

/*---- SUT -----*/
SUT="tpcc3"
/*-----*/

LASTC=86
MEASUREMENT="1"
WAREHOUSES=34000
/*---- SLAVES -----*/

#if MASTER_NUM1
SLAVES driver1a, driver1b, driver1c, driver1d, driver2a, driver2b, driver2c, driver2d, driver3a, driver3b,
driver3c, driver3d, driver4a, driver4b, driver4c, driver4d
#endif
#if MASTER_NUM2
SLAVES driver5a, driver5b, driver5c, driver5d, driver6a, driver6b, driver6c, driver6d, driver7a, driver7b,
driver7c, driver7d, driver8a, driver8b, driver8c, driver8d
#endif
#if MASTER_NUM3
SLAVES driver9a, driver9b, driver9c, driver9d, driver10a, driver10b, driver10c, driver10d, driver11a,
driver11b, driver11c, driver11d, driver12a, driver12b, driver12c, driver12d
#endif
#if MASTER_NUM4
SLAVES driver13a, driver13b, driver13c, driver13d, driver14a, driver14b, driver14c, driver14d, driver15a,
driver15b, driver15c, driver15d, driver16a, driver16b, driver16c, driver16d
#endif
#if MASTER_NUM5
SLAVES driver17a, driver17b, driver17c, driver17d, driver18a, driver18b, driver18c, driver18d, driver19a,
driver19b, driver19c, driver19d, driver20a, driver20b, driver20c, driver20d
#endif
#if MASTER_NUM6
SLAVES driver21a, driver21b, driver21c, driver21d, driver22a, driver22b, driver22c, driver22d, driver23a,
driver23b, driver23c, driver23d, driver24a, driver24b, driver24c, driver24d
#endif
#if MASTER_NUM7
SLAVES driver25a, driver25b, driver25c, driver25d, driver26a, driver26b, driver26c, driver26d, driver27a,
driver27b, driver27c, driver27d, driver28a, driver28b, driver28c, driver28d
#endif

/*---- CLIENTS -----*/
#if MASTER_NUM1
MAIN_CLIENT = client1
CLIENT_REAL = "client1 client2"
#endif
#if MASTER_NUM2
MAIN_CLIENT = client3
CLIENT_REAL = "client3 client4"
#endif
#if MASTER_NUM3
MAIN_CLIENT = client5
CLIENT_REAL = "client5 client6"
#endif
#if MASTER_NUM4
MAIN_CLIENT = client7
CLIENT_REAL = "client7 client8"
#endif
#if MASTER_NUM5
MAIN_CLIENT = client9
CLIENT_REAL = "client9 client10"
#endif
#if MASTER_NUM6
MAIN_CLIENT = client11
CLIENT_REAL = "client11 client12"
#endif
#if MASTER_NUM7
MAIN_CLIENT = client13
CLIENT_REAL = "client13 client14"
#endif

/*---- more client ctuff -----*/
#if MASTER_NUM1
CLIENT client11a oracle orif1db
```

```
CLIENT client12a oracle orif1db
CLIENT client11b oracle orif1db
CLIENT client12b oracle orif1db
CLIENT client11c oracle orif1db
CLIENT client12c oracle orif1db
CLIENT client11d oracle orif1db
CLIENT client12d oracle orif1db
CLIENT client11e oracle orif1db
CLIENT client12e oracle orif1db
CLIENT client11f oracle orif1db
CLIENT client12f oracle orif1db
CLIENT client11g oracle orif1db
CLIENT client12g oracle orif1db
CLIENT client11h oracle orif1db
CLIENT client12h oracle orif1db
CLIENT client11i oracle orif1db
CLIENT client12i oracle orif1db
CLIENT client11j oracle orif1db
CLIENT client12j oracle orif1db
CLIENT client11k oracle orif1db
CLIENT client12k oracle orif1db
CLIENT client11l oracle orif1db
CLIENT client12l oracle orif1db
CLIENT client11m oracle orif1db
CLIENT client12m oracle orif1db
CLIENT client11n oracle orif1db
CLIENT client12n oracle orif1db
```

```
CLIENT client21a oracle orif1db
CLIENT client22a oracle orif1db
CLIENT client21b oracle orif1db
CLIENT client22b oracle orif1db
CLIENT client21c oracle orif1db
CLIENT client22c oracle orif1db
CLIENT client21d oracle orif1db
CLIENT client22d oracle orif1db
CLIENT client21e oracle orif1db
CLIENT client22e oracle orif1db
CLIENT client21f oracle orif1db
CLIENT client22f oracle orif1db
CLIENT client21g oracle orif1db
CLIENT client22g oracle orif1db
CLIENT client21h oracle orif1db
CLIENT client22h oracle orif1db
CLIENT client21i oracle orif1db
CLIENT client22i oracle orif1db
CLIENT client21j oracle orif1db
CLIENT client22j oracle orif1db
CLIENT client21k oracle orif1db
CLIENT client22k oracle orif1db
CLIENT client21l oracle orif1db
CLIENT client22l oracle orif1db
CLIENT client21m oracle orif1db
CLIENT client22m oracle orif1db
CLIENT client21n oracle orif1db
CLIENT client22n oracle orif1db
```

```
#elif MASTER_NUM2
CLIENT client31a oracle orif1db
CLIENT client32a oracle orif1db
CLIENT client31b oracle orif1db
CLIENT client32b oracle orif1db
CLIENT client31c oracle orif1db
CLIENT client32c oracle orif1db
CLIENT client31d oracle orif1db
CLIENT client32d oracle orif1db
CLIENT client31e oracle orif1db
CLIENT client32e oracle orif1db
CLIENT client31f oracle orif1db
CLIENT client32f oracle orif1db
CLIENT client31g oracle orif1db
CLIENT client32g oracle orif1db
CLIENT client31h oracle orif1db
CLIENT client32h oracle orif1db
CLIENT client31i oracle orif1db
CLIENT client32i oracle orif1db
CLIENT client31j oracle orif1db
CLIENT client32j oracle orif1db
CLIENT client31k oracle orif1db
CLIENT client32k oracle orif1db
CLIENT client31l oracle orif1db
CLIENT client32l oracle orif1db
CLIENT client31m oracle orif1db
CLIENT client32m oracle orif1db
CLIENT client31n oracle orif1db
CLIENT client32n oracle orif1db
```

```
CLIENT client41a oracle orif1db
CLIENT client42a oracle orif1db
CLIENT client41b oracle orif1db
CLIENT client42b oracle orif1db
CLIENT client41c oracle orif1db
CLIENT client42c oracle orif1db
CLIENT client41d oracle orif1db
CLIENT client42d oracle orif1db
CLIENT client41e oracle orif1db
CLIENT client42e oracle orif1db
CLIENT client41f oracle orif1db
CLIENT client42f oracle orif1db
```


<pre> CLIENT client112b oracle orif1db CLIENT client111c oracle orif1db CLIENT client112c oracle orif1db CLIENT client111d oracle orif1db CLIENT client112d oracle orif1db CLIENT client111e oracle orif1db CLIENT client112e oracle orif1db CLIENT client111f oracle orif1db CLIENT client112f oracle orif1db CLIENT client111g oracle orif1db CLIENT client112g oracle orif1db CLIENT client111h oracle orif1db CLIENT client112h oracle orif1db CLIENT client111i oracle orif1db CLIENT client112i oracle orif1db CLIENT client111j oracle orif1db CLIENT client112j oracle orif1db CLIENT client111k oracle orif1db CLIENT client112k oracle orif1db CLIENT client111l oracle orif1db CLIENT client112l oracle orif1db CLIENT client111m oracle orif1db CLIENT client112m oracle orif1db CLIENT client111n oracle orif1db CLIENT client112n oracle orif1db CLIENT client121a oracle orif1db CLIENT client122a oracle orif1db CLIENT client121b oracle orif1db CLIENT client122b oracle orif1db CLIENT client121c oracle orif1db CLIENT client122c oracle orif1db CLIENT client121d oracle orif1db CLIENT client122d oracle orif1db CLIENT client121e oracle orif1db CLIENT client122e oracle orif1db CLIENT client121f oracle orif1db CLIENT client122f oracle orif1db CLIENT client121g oracle orif1db CLIENT client122g oracle orif1db CLIENT client121h oracle orif1db CLIENT client122h oracle orif1db CLIENT client121i oracle orif1db CLIENT client122i oracle orif1db CLIENT client121j oracle orif1db CLIENT client122j oracle orif1db CLIENT client121k oracle orif1db CLIENT client122k oracle orif1db CLIENT client121l oracle orif1db CLIENT client122l oracle orif1db CLIENT client121m oracle orif1db CLIENT client122m oracle orif1db CLIENT client121n oracle orif1db CLIENT client122n oracle orif1db #elif MASTER_NUM7 CLIENT client131a oracle orif1db CLIENT client132a oracle orif1db CLIENT client131b oracle orif1db CLIENT client132b oracle orif1db CLIENT client131c oracle orif1db CLIENT client132c oracle orif1db CLIENT client131d oracle orif1db CLIENT client132d oracle orif1db CLIENT client131e oracle orif1db CLIENT client132e oracle orif1db CLIENT client131f oracle orif1db CLIENT client132f oracle orif1db CLIENT client131g oracle orif1db CLIENT client132g oracle orif1db CLIENT client131h oracle orif1db CLIENT client132h oracle orif1db CLIENT client131i oracle orif1db CLIENT client132i oracle orif1db CLIENT client131j oracle orif1db CLIENT client132j oracle orif1db CLIENT client131k oracle orif1db CLIENT client132k oracle orif1db CLIENT client131l oracle orif1db CLIENT client132l oracle orif1db CLIENT client131m oracle orif1db CLIENT client132m oracle orif1db CLIENT client131n oracle orif1db CLIENT client132n oracle orif1db CLIENT client141a oracle orif1db CLIENT client142a oracle orif1db CLIENT client141b oracle orif1db CLIENT client142b oracle orif1db CLIENT client141c oracle orif1db CLIENT client142c oracle orif1db CLIENT client141d oracle orif1db CLIENT client142d oracle orif1db CLIENT client141e oracle orif1db CLIENT client142e oracle orif1db CLIENT client141f oracle orif1db CLIENT client142f oracle orif1db CLIENT client141g oracle orif1db CLIENT client142g oracle orif1db </pre>	<pre> CLIENT client141h oracle orif1db CLIENT client142h oracle orif1db CLIENT client141i oracle orif1db CLIENT client142i oracle orif1db CLIENT client141j oracle orif1db CLIENT client142j oracle orif1db CLIENT client141k oracle orif1db CLIENT client142k oracle orif1db CLIENT client141l oracle orif1db CLIENT client142l oracle orif1db CLIENT client141m oracle orif1db CLIENT client142m oracle orif1db CLIENT client141n oracle orif1db CLIENT client142n oracle orif1db #endif /*-----*/ TELNET telnet 23 SOCKET socket 199703 /* --- Sockets -----*/ #if MASTER_NUM1 SOCKET_NETWORK socket1 6700 driver1a SOCKET_NETWORK socket2 6701 driver1b SOCKET_NETWORK socket3 6702 driver1c SOCKET_NETWORK socket4 6703 driver1d SOCKET_NETWORK socket5 6704 driver1a SOCKET_NETWORK socket6 6705 driver1b SOCKET_NETWORK socket7 6706 driver1c SOCKET_NETWORK socket8 6707 driver1d SOCKET_NETWORK socket9 6708 driver1a SOCKET_NETWORK socket10 6709 driver1b SOCKET_NETWORK socket11 6710 driver1c SOCKET_NETWORK socket12 6711 driver1d SOCKET_NETWORK socket13 6712 driver1a SOCKET_NETWORK socket14 6713 driver1b SOCKET_NETWORK socket15 6714 driver1c SOCKET_NETWORK socket16 6715 driver1d SOCKET_NETWORK socket17 6716 driver1a SOCKET_NETWORK socket18 6717 driver1b SOCKET_NETWORK socket19 6718 driver1c SOCKET_NETWORK socket20 6719 driver1d SOCKET_NETWORK socket21 6720 driver1a SOCKET_NETWORK socket22 6721 driver1b SOCKET_NETWORK socket23 6722 driver1c SOCKET_NETWORK socket24 6723 driver1d SOCKET_NETWORK socket25 6724 driver1a SOCKET_NETWORK socket26 6725 driver1b SOCKET_NETWORK socket27 6726 driver1c SOCKET_NETWORK socket28 6727 driver1d SOCKET_NETWORK socket29 6728 driver1a SOCKET_NETWORK socket30 6729 driver1b SOCKET_NETWORK socket31 6730 driver1c SOCKET_NETWORK socket32 6731 driver1d SOCKET_NETWORK socket33 6732 driver1a SOCKET_NETWORK socket34 6733 driver1b SOCKET_NETWORK socket35 6734 driver1c SOCKET_NETWORK socket36 6735 driver1d SOCKET_NETWORK socket37 6736 driver1a SOCKET_NETWORK socket38 6737 driver1b SOCKET_NETWORK socket39 6738 driver1c SOCKET_NETWORK socket40 6739 driver1d SOCKET_NETWORK socket41 6740 driver1a SOCKET_NETWORK socket42 6741 driver1b SOCKET_NETWORK socket43 6742 driver1c SOCKET_NETWORK socket44 6743 driver1d SOCKET_NETWORK socket45 6744 driver1a SOCKET_NETWORK socket46 6745 driver1b SOCKET_NETWORK socket47 6746 driver1c SOCKET_NETWORK socket48 6747 driver1d SOCKET_NETWORK socket49 6748 driver1a SOCKET_NETWORK socket50 6749 driver1b SOCKET_NETWORK socket51 6750 driver1c SOCKET_NETWORK socket52 6751 driver1d SOCKET_NETWORK socket53 6752 driver1a SOCKET_NETWORK socket54 6753 driver1b SOCKET_NETWORK socket55 6754 driver1c SOCKET_NETWORK socket56 6755 driver1d SOCKET_NETWORK socket57 6756 driver2a SOCKET_NETWORK socket58 6757 driver2b SOCKET_NETWORK socket59 6758 driver2c SOCKET_NETWORK socket60 6759 driver2d SOCKET_NETWORK socket61 6760 driver2a SOCKET_NETWORK socket62 6761 driver2b SOCKET_NETWORK socket63 6762 driver2c SOCKET_NETWORK socket64 6763 driver2d SOCKET_NETWORK socket65 6764 driver2a SOCKET_NETWORK socket66 6765 driver2b SOCKET_NETWORK socket67 6766 driver2c SOCKET_NETWORK socket68 6767 driver2d SOCKET_NETWORK socket69 6768 driver2a SOCKET_NETWORK socket70 6769 driver2b SOCKET_NETWORK socket71 6770 driver2c SOCKET_NETWORK socket72 6771 driver2d SOCKET_NETWORK socket73 6772 driver2a SOCKET_NETWORK socket74 6773 driver2b SOCKET_NETWORK socket75 6774 driver2c SOCKET_NETWORK socket76 6775 driver2d SOCKET_NETWORK socket77 6776 driver2a SOCKET_NETWORK socket78 6777 driver2b </pre>
--	--

SOCKET_NETWORK socket79	6778 driver2c	SOCKET_NETWORK socket178	6765 driver4b
SOCKET_NETWORK socket80	6779 driver2d	SOCKET_NETWORK socket179	6766 driver4c
SOCKET_NETWORK socket81	6780 driver2a	SOCKET_NETWORK socket180	6767 driver4d
SOCKET_NETWORK socket82	6781 driver2b	SOCKET_NETWORK socket181	6768 driver4a
SOCKET_NETWORK socket83	6782 driver2c	SOCKET_NETWORK socket182	6769 driver4b
SOCKET_NETWORK socket84	6783 driver2d	SOCKET_NETWORK socket183	6770 driver4c
SOCKET_NETWORK socket85	6784 driver2a	SOCKET_NETWORK socket184	6771 driver4d
SOCKET_NETWORK socket86	6785 driver2b	SOCKET_NETWORK socket185	6772 driver4a
SOCKET_NETWORK socket87	6786 driver2c	SOCKET_NETWORK socket186	6773 driver4b
SOCKET_NETWORK socket88	6787 driver2d	SOCKET_NETWORK socket187	6774 driver4c
SOCKET_NETWORK socket89	6788 driver2a	SOCKET_NETWORK socket188	6775 driver4d
SOCKET_NETWORK socket90	6789 driver2b	SOCKET_NETWORK socket189	6776 driver4a
SOCKET_NETWORK socket91	6790 driver2c	SOCKET_NETWORK socket190	6777 driver4b
SOCKET_NETWORK socket92	6791 driver2d	SOCKET_NETWORK socket191	6778 driver4c
SOCKET_NETWORK socket93	6792 driver2a	SOCKET_NETWORK socket192	6779 driver4d
SOCKET_NETWORK socket94	6793 driver2b	SOCKET_NETWORK socket193	6780 driver4a
SOCKET_NETWORK socket95	6794 driver2c	SOCKET_NETWORK socket194	6781 driver4b
SOCKET_NETWORK socket96	6795 driver2d	SOCKET_NETWORK socket195	6782 driver4c
SOCKET_NETWORK socket97	6796 driver2a	SOCKET_NETWORK socket196	6783 driver4d
SOCKET_NETWORK socket98	6797 driver2b	SOCKET_NETWORK socket197	6784 driver4a
SOCKET_NETWORK socket99	6798 driver2c	SOCKET_NETWORK socket198	6785 driver4b
SOCKET_NETWORK socket100	6799 driver2d	SOCKET_NETWORK socket199	6786 driver4c
SOCKET_NETWORK socket101	6800 driver2a	SOCKET_NETWORK socket200	6787 driver4d
SOCKET_NETWORK socket102	6801 driver2b	SOCKET_NETWORK socket201	6788 driver4a
SOCKET_NETWORK socket103	6802 driver2c	SOCKET_NETWORK socket202	6789 driver4b
SOCKET_NETWORK socket104	6803 driver2d	SOCKET_NETWORK socket203	6790 driver4c
SOCKET_NETWORK socket105	6804 driver2a	SOCKET_NETWORK socket204	6791 driver4d
SOCKET_NETWORK socket106	6805 driver2b	SOCKET_NETWORK socket205	6792 driver4a
SOCKET_NETWORK socket107	6806 driver2c	SOCKET_NETWORK socket206	6793 driver4b
SOCKET_NETWORK socket108	6807 driver2d	SOCKET_NETWORK socket207	6794 driver4c
SOCKET_NETWORK socket109	6808 driver2a	SOCKET_NETWORK socket208	6795 driver4d
SOCKET_NETWORK socket110	6809 driver2b	SOCKET_NETWORK socket209	6796 driver4a
SOCKET_NETWORK socket111	6810 driver2c	SOCKET_NETWORK socket210	6797 driver4b
SOCKET_NETWORK socket112	6811 driver2d	SOCKET_NETWORK socket211	6798 driver4c
SOCKET_NETWORK socket113	6700 driver3a	SOCKET_NETWORK socket212	6799 driver4d
SOCKET_NETWORK socket114	6701 driver3b	SOCKET_NETWORK socket213	6800 driver4a
SOCKET_NETWORK socket115	6702 driver3c	SOCKET_NETWORK socket214	6801 driver4b
SOCKET_NETWORK socket116	6703 driver3d	SOCKET_NETWORK socket215	6802 driver4c
SOCKET_NETWORK socket117	6704 driver3a	SOCKET_NETWORK socket216	6803 driver4d
SOCKET_NETWORK socket118	6705 driver3b	SOCKET_NETWORK socket217	6804 driver4a
SOCKET_NETWORK socket119	6706 driver3c	SOCKET_NETWORK socket218	6805 driver4b
SOCKET_NETWORK socket120	6707 driver3d	SOCKET_NETWORK socket219	6806 driver4c
SOCKET_NETWORK socket121	6708 driver3a	SOCKET_NETWORK socket220	6807 driver4d
SOCKET_NETWORK socket122	6709 driver3b	SOCKET_NETWORK socket221	6808 driver4a
SOCKET_NETWORK socket123	6710 driver3c	SOCKET_NETWORK socket222	6809 driver4b
SOCKET_NETWORK socket124	6711 driver3d	SOCKET_NETWORK socket223	6810 driver4c
SOCKET_NETWORK socket125	6712 driver3a	SOCKET_NETWORK socket224	6811 driver4d
SOCKET_NETWORK socket126	6713 driver3b	#elif MASTER_NUM2	
SOCKET_NETWORK socket127	6714 driver3c	SOCKET_NETWORK socket225	6700 driver5a
SOCKET_NETWORK socket128	6715 driver3d	SOCKET_NETWORK socket226	6701 driver5b
SOCKET_NETWORK socket129	6716 driver3a	SOCKET_NETWORK socket227	6702 driver5c
SOCKET_NETWORK socket130	6717 driver3b	SOCKET_NETWORK socket228	6703 driver5d
SOCKET_NETWORK socket131	6718 driver3c	SOCKET_NETWORK socket229	6704 driver5a
SOCKET_NETWORK socket132	6719 driver3d	SOCKET_NETWORK socket230	6705 driver5b
SOCKET_NETWORK socket133	6720 driver3a	SOCKET_NETWORK socket231	6706 driver5c
SOCKET_NETWORK socket134	6721 driver3b	SOCKET_NETWORK socket232	6707 driver5d
SOCKET_NETWORK socket135	6722 driver3c	SOCKET_NETWORK socket233	6708 driver5a
SOCKET_NETWORK socket136	6723 driver3d	SOCKET_NETWORK socket234	6709 driver5b
SOCKET_NETWORK socket137	6724 driver3a	SOCKET_NETWORK socket235	6710 driver5c
SOCKET_NETWORK socket138	6725 driver3b	SOCKET_NETWORK socket236	6711 driver5d
SOCKET_NETWORK socket139	6726 driver3c	SOCKET_NETWORK socket237	6712 driver5a
SOCKET_NETWORK socket140	6727 driver3d	SOCKET_NETWORK socket238	6713 driver5b
SOCKET_NETWORK socket141	6728 driver3a	SOCKET_NETWORK socket239	6714 driver5c
SOCKET_NETWORK socket142	6729 driver3b	SOCKET_NETWORK socket240	6715 driver5d
SOCKET_NETWORK socket143	6730 driver3c	SOCKET_NETWORK socket241	6716 driver5a
SOCKET_NETWORK socket144	6731 driver3d	SOCKET_NETWORK socket242	6717 driver5b
SOCKET_NETWORK socket145	6732 driver3a	SOCKET_NETWORK socket243	6718 driver5c
SOCKET_NETWORK socket146	6733 driver3b	SOCKET_NETWORK socket244	6719 driver5d
SOCKET_NETWORK socket147	6734 driver3c	SOCKET_NETWORK socket245	6720 driver5a
SOCKET_NETWORK socket148	6735 driver3d	SOCKET_NETWORK socket246	6721 driver5b
SOCKET_NETWORK socket149	6736 driver3a	SOCKET_NETWORK socket247	6722 driver5c
SOCKET_NETWORK socket150	6737 driver3b	SOCKET_NETWORK socket248	6723 driver5d
SOCKET_NETWORK socket151	6738 driver3c	SOCKET_NETWORK socket249	6724 driver5a
SOCKET_NETWORK socket152	6739 driver3d	SOCKET_NETWORK socket250	6725 driver5b
SOCKET_NETWORK socket153	6740 driver3a	SOCKET_NETWORK socket251	6726 driver5c
SOCKET_NETWORK socket154	6741 driver3b	SOCKET_NETWORK socket252	6727 driver5d
SOCKET_NETWORK socket155	6742 driver3c	SOCKET_NETWORK socket253	6728 driver5a
SOCKET_NETWORK socket156	6743 driver3d	SOCKET_NETWORK socket254	6729 driver5b
SOCKET_NETWORK socket157	6744 driver3a	SOCKET_NETWORK socket255	6730 driver5c
SOCKET_NETWORK socket158	6745 driver3b	SOCKET_NETWORK socket256	6731 driver5d
SOCKET_NETWORK socket159	6746 driver3c	SOCKET_NETWORK socket257	6732 driver5a
SOCKET_NETWORK socket160	6747 driver3d	SOCKET_NETWORK socket258	6733 driver5b
SOCKET_NETWORK socket161	6748 driver3a	SOCKET_NETWORK socket259	6734 driver5c
SOCKET_NETWORK socket162	6749 driver3b	SOCKET_NETWORK socket260	6735 driver5d
SOCKET_NETWORK socket163	6750 driver3c	SOCKET_NETWORK socket261	6736 driver5a
SOCKET_NETWORK socket164	6751 driver3d	SOCKET_NETWORK socket262	6737 driver5b
SOCKET_NETWORK socket165	6752 driver3a	SOCKET_NETWORK socket263	6738 driver5c
SOCKET_NETWORK socket166	6753 driver3b	SOCKET_NETWORK socket264	6739 driver5d
SOCKET_NETWORK socket167	6754 driver3c	SOCKET_NETWORK socket265	6740 driver5a
SOCKET_NETWORK socket168	6755 driver3d	SOCKET_NETWORK socket266	6741 driver5b
SOCKET_NETWORK socket169	6756 driver4a	SOCKET_NETWORK socket267	6742 driver5c
SOCKET_NETWORK socket170	6757 driver4b	SOCKET_NETWORK socket268	6743 driver5d
SOCKET_NETWORK socket171	6758 driver4c	SOCKET_NETWORK socket269	6744 driver5a
SOCKET_NETWORK socket172	6759 driver4d	SOCKET_NETWORK socket270	6745 driver5b
SOCKET_NETWORK socket173	6760 driver4a	SOCKET_NETWORK socket271	6746 driver5c
SOCKET_NETWORK socket174	6761 driver4b	SOCKET_NETWORK socket272	6747 driver5d
SOCKET_NETWORK socket175	6762 driver4c	SOCKET_NETWORK socket273	6748 driver5a
SOCKET_NETWORK socket176	6763 driver4d	SOCKET_NETWORK socket274	6749 driver5b
SOCKET_NETWORK socket177	6764 driver4a	SOCKET_NETWORK socket275	6750 driver5c

SOCKET_NETWORK socket868	6783 driver16d	SOCKET_NETWORK socket966	6769 driver18b
SOCKET_NETWORK socket869	6784 driver16a	SOCKET_NETWORK socket967	6770 driver18c
SOCKET_NETWORK socket870	6785 driver16b	SOCKET_NETWORK socket968	6771 driver18d
SOCKET_NETWORK socket871	6786 driver16c	SOCKET_NETWORK socket969	6772 driver18a
SOCKET_NETWORK socket872	6787 driver16d	SOCKET_NETWORK socket970	6773 driver18b
SOCKET_NETWORK socket873	6788 driver16a	SOCKET_NETWORK socket971	6774 driver18c
SOCKET_NETWORK socket874	6789 driver16b	SOCKET_NETWORK socket972	6775 driver18d
SOCKET_NETWORK socket875	6790 driver16c	SOCKET_NETWORK socket973	6776 driver18a
SOCKET_NETWORK socket876	6791 driver16d	SOCKET_NETWORK socket974	6777 driver18b
SOCKET_NETWORK socket877	6792 driver16a	SOCKET_NETWORK socket975	6778 driver18c
SOCKET_NETWORK socket878	6793 driver16b	SOCKET_NETWORK socket976	6779 driver18d
SOCKET_NETWORK socket879	6794 driver16c	SOCKET_NETWORK socket977	6780 driver18a
SOCKET_NETWORK socket880	6795 driver16d	SOCKET_NETWORK socket978	6781 driver18b
SOCKET_NETWORK socket881	6796 driver16a	SOCKET_NETWORK socket979	6782 driver18c
SOCKET_NETWORK socket882	6797 driver16b	SOCKET_NETWORK socket980	6783 driver18d
SOCKET_NETWORK socket883	6798 driver16c	SOCKET_NETWORK socket981	6784 driver18a
SOCKET_NETWORK socket884	6799 driver16d	SOCKET_NETWORK socket982	6785 driver18b
SOCKET_NETWORK socket885	6800 driver16a	SOCKET_NETWORK socket983	6786 driver18c
SOCKET_NETWORK socket886	6801 driver16b	SOCKET_NETWORK socket984	6787 driver18d
SOCKET_NETWORK socket887	6802 driver16c	SOCKET_NETWORK socket985	6788 driver18a
SOCKET_NETWORK socket888	6803 driver16d	SOCKET_NETWORK socket986	6789 driver18b
SOCKET_NETWORK socket889	6804 driver16a	SOCKET_NETWORK socket987	6790 driver18c
SOCKET_NETWORK socket890	6805 driver16b	SOCKET_NETWORK socket988	6791 driver18d
SOCKET_NETWORK socket891	6806 driver16c	SOCKET_NETWORK socket989	6792 driver18a
SOCKET_NETWORK socket892	6807 driver16d	SOCKET_NETWORK socket990	6793 driver18b
SOCKET_NETWORK socket893	6808 driver16a	SOCKET_NETWORK socket991	6794 driver18c
SOCKET_NETWORK socket894	6809 driver16b	SOCKET_NETWORK socket992	6795 driver18d
SOCKET_NETWORK socket895	6810 driver16c	SOCKET_NETWORK socket993	6796 driver18a
SOCKET_NETWORK socket896	6811 driver16d	SOCKET_NETWORK socket994	6797 driver18b
#elif MASTER_NUM5			
SOCKET_NETWORK socket897	6700 driver17a	SOCKET_NETWORK socket995	6798 driver18c
SOCKET_NETWORK socket898	6701 driver17b	SOCKET_NETWORK socket996	6799 driver18d
SOCKET_NETWORK socket899	6702 driver17c	SOCKET_NETWORK socket997	6800 driver18a
SOCKET_NETWORK socket900	6703 driver17d	SOCKET_NETWORK socket998	6801 driver18b
SOCKET_NETWORK socket901	6704 driver17a	SOCKET_NETWORK socket999	6802 driver18c
SOCKET_NETWORK socket902	6705 driver17b	SOCKET_NETWORK socket1000	6803 driver18d
SOCKET_NETWORK socket903	6706 driver17c	SOCKET_NETWORK socket1001	6804 driver18a
SOCKET_NETWORK socket904	6707 driver17d	SOCKET_NETWORK socket1002	6805 driver18b
SOCKET_NETWORK socket905	6708 driver17a	SOCKET_NETWORK socket1003	6806 driver18c
SOCKET_NETWORK socket906	6709 driver17b	SOCKET_NETWORK socket1004	6807 driver18d
SOCKET_NETWORK socket907	6710 driver17c	SOCKET_NETWORK socket1005	6808 driver18a
SOCKET_NETWORK socket908	6711 driver17d	SOCKET_NETWORK socket1006	6809 driver18b
SOCKET_NETWORK socket909	6712 driver17a	SOCKET_NETWORK socket1007	6810 driver18c
SOCKET_NETWORK socket910	6713 driver17b	SOCKET_NETWORK socket1008	6811 driver18d
SOCKET_NETWORK socket911	6714 driver17c	SOCKET_NETWORK socket1009	6700 driver19a
SOCKET_NETWORK socket912	6715 driver17d	SOCKET_NETWORK socket1010	6701 driver19b
SOCKET_NETWORK socket913	6716 driver17a	SOCKET_NETWORK socket1011	6702 driver19c
SOCKET_NETWORK socket914	6717 driver17b	SOCKET_NETWORK socket1012	6703 driver19d
SOCKET_NETWORK socket915	6718 driver17c	SOCKET_NETWORK socket1013	6704 driver19a
SOCKET_NETWORK socket916	6719 driver17d	SOCKET_NETWORK socket1014	6705 driver19b
SOCKET_NETWORK socket917	6720 driver17a	SOCKET_NETWORK socket1015	6706 driver19c
SOCKET_NETWORK socket918	6721 driver17b	SOCKET_NETWORK socket1016	6707 driver19d
SOCKET_NETWORK socket919	6722 driver17c	SOCKET_NETWORK socket1017	6708 driver19a
SOCKET_NETWORK socket920	6723 driver17d	SOCKET_NETWORK socket1018	6709 driver19b
SOCKET_NETWORK socket921	6724 driver17a	SOCKET_NETWORK socket1019	6710 driver19c
SOCKET_NETWORK socket922	6725 driver17b	SOCKET_NETWORK socket1020	6711 driver19d
SOCKET_NETWORK socket923	6726 driver17c	SOCKET_NETWORK socket1021	6712 driver19a
SOCKET_NETWORK socket924	6727 driver17d	SOCKET_NETWORK socket1022	6713 driver19b
SOCKET_NETWORK socket925	6728 driver17a	SOCKET_NETWORK socket1023	6714 driver19c
SOCKET_NETWORK socket926	6729 driver17b	SOCKET_NETWORK socket1024	6715 driver19d
SOCKET_NETWORK socket927	6730 driver17c	SOCKET_NETWORK socket1025	6716 driver19a
SOCKET_NETWORK socket928	6731 driver17d	SOCKET_NETWORK socket1026	6717 driver19b
SOCKET_NETWORK socket929	6732 driver17a	SOCKET_NETWORK socket1027	6718 driver19c
SOCKET_NETWORK socket930	6733 driver17b	SOCKET_NETWORK socket1028	6719 driver19d
SOCKET_NETWORK socket931	6734 driver17c	SOCKET_NETWORK socket1029	6720 driver19a
SOCKET_NETWORK socket932	6735 driver17d	SOCKET_NETWORK socket1030	6721 driver19b
SOCKET_NETWORK socket933	6736 driver17a	SOCKET_NETWORK socket1031	6722 driver19c
SOCKET_NETWORK socket934	6737 driver17b	SOCKET_NETWORK socket1032	6723 driver19d
SOCKET_NETWORK socket935	6738 driver17c	SOCKET_NETWORK socket1033	6724 driver19a
SOCKET_NETWORK socket936	6739 driver17d	SOCKET_NETWORK socket1034	6725 driver19b
SOCKET_NETWORK socket937	6740 driver17a	SOCKET_NETWORK socket1035	6726 driver19c
SOCKET_NETWORK socket938	6741 driver17b	SOCKET_NETWORK socket1036	6727 driver19d
SOCKET_NETWORK socket939	6742 driver17c	SOCKET_NETWORK socket1037	6728 driver19a
SOCKET_NETWORK socket940	6743 driver17d	SOCKET_NETWORK socket1038	6729 driver19b
SOCKET_NETWORK socket941	6744 driver17a	SOCKET_NETWORK socket1039	6730 driver19c
SOCKET_NETWORK socket942	6745 driver17b	SOCKET_NETWORK socket1040	6731 driver19d
SOCKET_NETWORK socket943	6746 driver17c	SOCKET_NETWORK socket1041	6732 driver19a
SOCKET_NETWORK socket944	6747 driver17d	SOCKET_NETWORK socket1042	6733 driver19b
SOCKET_NETWORK socket945	6748 driver17a	SOCKET_NETWORK socket1043	6734 driver19c
SOCKET_NETWORK socket946	6749 driver17b	SOCKET_NETWORK socket1044	6735 driver19d
SOCKET_NETWORK socket947	6750 driver17c	SOCKET_NETWORK socket1045	6736 driver19a
SOCKET_NETWORK socket948	6751 driver17d	SOCKET_NETWORK socket1046	6737 driver19b
SOCKET_NETWORK socket949	6752 driver17a	SOCKET_NETWORK socket1047	6738 driver19c
SOCKET_NETWORK socket950	6753 driver17b	SOCKET_NETWORK socket1048	6739 driver19d
SOCKET_NETWORK socket951	6754 driver17c	SOCKET_NETWORK socket1049	6740 driver19a
SOCKET_NETWORK socket952	6755 driver17d	SOCKET_NETWORK socket1050	6741 driver19b
SOCKET_NETWORK socket953	6756 driver18a	SOCKET_NETWORK socket1051	6742 driver19c
SOCKET_NETWORK socket954	6757 driver18b	SOCKET_NETWORK socket1052	6743 driver19d
SOCKET_NETWORK socket955	6758 driver18c	SOCKET_NETWORK socket1053	6744 driver19a
SOCKET_NETWORK socket956	6759 driver18d	SOCKET_NETWORK socket1054	6745 driver19b
SOCKET_NETWORK socket957	6760 driver18a	SOCKET_NETWORK socket1055	6746 driver19c
SOCKET_NETWORK socket958	6761 driver18b	SOCKET_NETWORK socket1056	6747 driver19d
SOCKET_NETWORK socket959	6762 driver18c	SOCKET_NETWORK socket1057	6748 driver19a
SOCKET_NETWORK socket960	6763 driver18d	SOCKET_NETWORK socket1058	6749 driver19b
SOCKET_NETWORK socket961	6764 driver18a	SOCKET_NETWORK socket1059	6750 driver19c
SOCKET_NETWORK socket962	6765 driver18b	SOCKET_NETWORK socket1060	6751 driver19d
SOCKET_NETWORK socket963	6766 driver18c	SOCKET_NETWORK socket1061	6752 driver19a
SOCKET_NETWORK socket964	6767 driver18d	SOCKET_NETWORK socket1062	6753 driver19b
SOCKET_NETWORK socket965	6768 driver18a	SOCKET_NETWORK socket1063	6754 driver19c
		SOCKET_NETWORK socket1064	6755 driver19d

SOCKET_NETWORK socket1459	6702 driver27c	SOCKET_NETWORK socket1558	6801 driver28b
SOCKET_NETWORK socket1460	6703 driver27d	SOCKET_NETWORK socket1559	6802 driver28c
SOCKET_NETWORK socket1461	6704 driver27a	SOCKET_NETWORK socket1560	6803 driver28d
SOCKET_NETWORK socket1462	6705 driver27b	SOCKET_NETWORK socket1561	6804 driver28a
SOCKET_NETWORK socket1463	6706 driver27c	SOCKET_NETWORK socket1562	6805 driver28b
SOCKET_NETWORK socket1464	6707 driver27d	SOCKET_NETWORK socket1563	6806 driver28c
SOCKET_NETWORK socket1465	6708 driver27a	SOCKET_NETWORK socket1564	6807 driver28d
SOCKET_NETWORK socket1466	6709 driver27b	SOCKET_NETWORK socket1565	6808 driver28a
SOCKET_NETWORK socket1467	6710 driver27c	SOCKET_NETWORK socket1566	6809 driver28b
SOCKET_NETWORK socket1468	6711 driver27d	SOCKET_NETWORK socket1567	6810 driver28c
SOCKET_NETWORK socket1469	6712 driver27a	SOCKET_NETWORK socket1568	6811 driver28d
SOCKET_NETWORK socket1470	6713 driver27b	#endif	
SOCKET_NETWORK socket1471	6714 driver27c	/*-----*/	
SOCKET_NETWORK socket1472	6715 driver27d	OUTPUTNAME="regattaH"	
SOCKET_NETWORK socket1473	6716 driver27a	CPU=48	
SOCKET_NETWORK socket1474	6717 driver27b	#if 0	
SOCKET_NETWORK socket1475	6718 driver27c	BEGIN_WAIT=10:00	
SOCKET_NETWORK socket1476	6719 driver27d	RAMPUP=12:00	
SOCKET_NETWORK socket1477	6720 driver27a	RUNTIME=30:00	
SOCKET_NETWORK socket1478	6721 driver27b	RAMPDOWN_WAIT=5:00	
SOCKET_NETWORK socket1479	6722 driver27c	RAMPDOWN=17:00	
SOCKET_NETWORK socket1480	6723 driver27d	#else	
SOCKET_NETWORK socket1481	6724 driver27a	BEGIN_WAIT=10:00	
SOCKET_NETWORK socket1482	6725 driver27b	RAMPUP=78:00	
SOCKET_NETWORK socket1483	6726 driver27c	RUNTIME=120:00	
SOCKET_NETWORK socket1484	6727 driver27d	RAMPDOWN_WAIT=5:00	
SOCKET_NETWORK socket1485	6728 driver27a	RAMPDOWN=20:00	
SOCKET_NETWORK socket1486	6729 driver27b	#endif	
SOCKET_NETWORK socket1487	6730 driver27c	INTERVAL=1:00/* Interval to calculate mix from*/	
SOCKET_NETWORK socket1488	6731 driver27d	LOGIN_MAX_LOAD = 4	
SOCKET_NETWORK socket1489	6732 driver27a	LOGIN_BEGIN = 0 /* skip login state if set to 1 */	
SOCKET_NETWORK socket1490	6733 driver27b	NOBEGIN = 1	
SOCKET_NETWORK socket1491	6734 driver27c	KEYSTROKE_PACKET_SIZE= 0	
SOCKET_NETWORK socket1492	6735 driver27d	MAX_CONCURRENT_SPAWN = 10	
SOCKET_NETWORK socket1493	6736 driver27a	SPAWN_COUNT = 4	
SOCKET_NETWORK socket1494	6737 driver27b	MIN_PORT = 8088	
SOCKET_NETWORK socket1495	6738 driver27c	MAX_PORT = 8089	
SOCKET_NETWORK socket1496	6739 driver27d	/* User variables. Think,Emulex Delay, %desired, %min, %max */	
SOCKET_NETWORK socket1497	6740 driver27a	#if 1 /* Testing */	
SOCKET_NETWORK socket1498	6741 driver27b	NEWORDER = "12.02, 0, 0"	
SOCKET_NETWORK socket1499	6742 driver27c	PAYMENT = "12.02, 0, 0, 43.01, 43.01, 43.01 "	
SOCKET_NETWORK socket1500	6743 driver27d	ORDSTAT = "10.02, 0, 0, 4.02, 4.02, 4.02 "	
SOCKET_NETWORK socket1501	6744 driver27a	DELIVERY = "05.02, 0, 0, 4.02, 4.02, 4.02 "	
SOCKET_NETWORK socket1502	6745 driver27b	STOCKLEV = "05.02, 0, 0, 4.02, 4.02, 4.02 "	
SOCKET_NETWORK socket1503	6746 driver27c	#elif 0 /* From rtparams.null */	
SOCKET_NETWORK socket1504	6747 driver27d	NEWORDER = "12.25, 0.42, 0.38"	
SOCKET_NETWORK socket1505	6748 driver27a	PAYMENT = "12.25, 0.19, 0.23, 43.2, 41.1, 45.3 "	
SOCKET_NETWORK socket1506	6749 driver27b	ORDSTAT = "10.50, 0.39, 0.21, 4.1, 3.9, 4.3 "	
SOCKET_NETWORK socket1507	6750 driver27c	DELIVERY = "05.5, 0.19, 0.15, 4.1, 3.9, 4.3 "	
SOCKET_NETWORK socket1508	6751 driver27d	STOCKLEV = "05.5, 0.25, 0.18, 4.1, 3.9, 4.3 "	
SOCKET_NETWORK socket1509	6752 driver27a	#elif 0 /* From Pookeepsie */	
SOCKET_NETWORK socket1510	6753 driver27b	NEWORDER = "16.25, 0.42, 0.38"	
SOCKET_NETWORK socket1511	6754 driver27c	PAYMENT = "16.25, 0.19, 0.23, 43.15, 43.15, 43.15 "	
SOCKET_NETWORK socket1512	6755 driver27d	ORDSTAT = "14.50, 0.39, 0.21, 4.03, 4.03, 4.03 "	
SOCKET_NETWORK socket1513	6756 driver28a	DELIVERY = "09.50, 0.19, 0.15, 4.03, 4.03, 4.03 "	
SOCKET_NETWORK socket1514	6757 driver28b	STOCKLEV = "09.50, 0.25, 0.18, 4.03, 4.03, 4.03 "	
SOCKET_NETWORK socket1515	6758 driver28c	#endif	
SOCKET_NETWORK socket1516	6759 driver28d	/*---- Starting users on sockets -----*/	
SOCKET_NETWORK socket1517	6760 driver28a	#if MASTER_NUM1	
SOCKET_NETWORK socket1518	6761 driver28b	START_RANGE client1a socket1 210 0-21	
SOCKET_NETWORK socket1519	6762 driver28c	START_RANGE client1b socket2 220 21-43	
SOCKET_NETWORK socket1520	6763 driver28d	START_RANGE client1c socket3 220 43-65	
SOCKET_NETWORK socket1521	6764 driver28a	START_RANGE client1d socket4 210 65-86	
SOCKET_NETWORK socket1522	6765 driver28b	START_RANGE client1e socket5 220 86-108	
SOCKET_NETWORK socket1523	6766 driver28c	START_RANGE client1f socket6 220 108-130	
SOCKET_NETWORK socket1524	6767 driver28d	START_RANGE client1g socket7 210 130-151	
SOCKET_NETWORK socket1525	6768 driver28a	START_RANGE client1h socket8 220 151-173	
SOCKET_NETWORK socket1526	6769 driver28b	START_RANGE client1i socket9 220 173-195	
SOCKET_NETWORK socket1527	6770 driver28c	START_RANGE client1j socket10 210 195-216	
SOCKET_NETWORK socket1528	6771 driver28d	START_RANGE client1k socket11 220 216-238	
SOCKET_NETWORK socket1529	6772 driver28a	START_RANGE client1l socket12 220 238-260	
SOCKET_NETWORK socket1530	6773 driver28b	START_RANGE client1m socket13 210 260-281	
SOCKET_NETWORK socket1531	6774 driver28c	START_RANGE client1n socket14 220 281-303	
SOCKET_NETWORK socket1532	6775 driver28d	START_RANGE client1a socket15 220 303-325	
SOCKET_NETWORK socket1533	6776 driver28a	START_RANGE client1b socket16 210 325-346	
SOCKET_NETWORK socket1534	6777 driver28b	START_RANGE client1c socket17 220 346-368	
SOCKET_NETWORK socket1535	6778 driver28c	START_RANGE client1d socket18 220 368-390	
SOCKET_NETWORK socket1536	6779 driver28d	START_RANGE client1e socket19 210 390-411	
SOCKET_NETWORK socket1537	6780 driver28a	START_RANGE client1f socket20 220 411-433	
SOCKET_NETWORK socket1538	6781 driver28b	START_RANGE client1g socket21 220 433-455	
SOCKET_NETWORK socket1539	6782 driver28c	START_RANGE client1h socket22 220 455-477	
SOCKET_NETWORK socket1540	6783 driver28d	START_RANGE client1i socket23 210 477-498	
SOCKET_NETWORK socket1541	6784 driver28a	START_RANGE client1j socket24 220 498-520	
SOCKET_NETWORK socket1542	6785 driver28b	START_RANGE client1k socket25 220 520-542	
SOCKET_NETWORK socket1543	6786 driver28c	START_RANGE client1l socket26 210 542-563	
SOCKET_NETWORK socket1544	6787 driver28d	START_RANGE client1m socket27 220 563-585	
SOCKET_NETWORK socket1545	6788 driver28a	START_RANGE client1n socket28 220 585-607	
SOCKET_NETWORK socket1546	6789 driver28b	START_RANGE client1a socket29 210 607-628	
SOCKET_NETWORK socket1547	6790 driver28c	START_RANGE client1b socket30 220 628-650	
SOCKET_NETWORK socket1548	6791 driver28d	START_RANGE client1c socket31 220 650-672	
SOCKET_NETWORK socket1549	6792 driver28a	START_RANGE client1d socket32 210 672-693	
SOCKET_NETWORK socket1550	6793 driver28b	START_RANGE client1e socket33 220 693-715	
SOCKET_NETWORK socket1551	6794 driver28c	START_RANGE client1f socket34 220 715-737	
SOCKET_NETWORK socket1552	6795 driver28d	START_RANGE client1g socket35 210 737-758	
SOCKET_NETWORK socket1553	6796 driver28a	START_RANGE client1h socket36 220 758-780	
SOCKET_NETWORK socket1554	6797 driver28b	START_RANGE client1i socket37 220 780-802	
SOCKET_NETWORK socket1555	6798 driver28c	START_RANGE client1j socket38 210 802-823	
SOCKET_NETWORK socket1556	6799 driver28d	START_RANGE client1k socket39 220 823-845	
SOCKET_NETWORK socket1557	6800 driver28a	START_RANGE client1l socket40 220 845-867	

<p>START_RANGE client11m socket41 220 867-889 START_RANGE client11n socket42 210 889-910 START_RANGE client11a socket43 220 910-932 START_RANGE client11b socket44 220 932-954 START_RANGE client11c socket45 210 954-975 START_RANGE client11d socket46 220 975-997 START_RANGE client11e socket47 220 997-1019 START_RANGE client11f socket48 210 1019-1040 START_RANGE client11g socket49 220 1040-1062 START_RANGE client11h socket50 220 1062-1084 START_RANGE client11i socket51 210 1084-1105 START_RANGE client11j socket52 220 1105-1127 START_RANGE client11k socket53 220 1127-1149 START_RANGE client11l socket54 210 1149-1170 START_RANGE client11m socket55 220 1170-1192 START_RANGE client11n socket56 220 1192-1214</p> <p>START_RANGE client12a socket57 210 1214-1235 START_RANGE client12b socket58 220 1235-1257 START_RANGE client12c socket59 220 1257-1279 START_RANGE client12d socket60 220 1279-1301 START_RANGE client12e socket61 210 1301-1322 START_RANGE client12f socket62 220 1322-1344 START_RANGE client12g socket63 220 1344-1366 START_RANGE client12h socket64 210 1366-1387 START_RANGE client12i socket65 220 1387-1409 START_RANGE client12j socket66 220 1409-1431 START_RANGE client12k socket67 210 1431-1452 START_RANGE client12l socket68 220 1452-1474 START_RANGE client12m socket69 220 1474-1496 START_RANGE client12n socket70 210 1496-1517 START_RANGE client12a socket71 220 1517-1539 START_RANGE client12b socket72 220 1539-1561 START_RANGE client12c socket73 210 1561-1582 START_RANGE client12d socket74 220 1582-1604 START_RANGE client12e socket75 220 1604-1626 START_RANGE client12f socket76 210 1626-1647 START_RANGE client12g socket77 220 1647-1669 START_RANGE client12h socket78 220 1669-1691 START_RANGE client12i socket79 220 1691-1713 START_RANGE client12j socket80 210 1713-1734 START_RANGE client12k socket81 220 1734-1756 START_RANGE client12l socket82 220 1756-1778 START_RANGE client12m socket83 210 1778-1799 START_RANGE client12n socket84 220 1799-1821 START_RANGE client12a socket85 220 1821-1843 START_RANGE client12b socket86 210 1843-1864 START_RANGE client12c socket87 220 1864-1886 START_RANGE client12d socket88 220 1886-1908 START_RANGE client12e socket89 210 1908-1929 START_RANGE client12f socket90 220 1929-1951 START_RANGE client12g socket91 220 1951-1973 START_RANGE client12h socket92 210 1973-1994 START_RANGE client12i socket93 220 1994-2016 START_RANGE client12j socket94 220 2016-2038 START_RANGE client12k socket95 210 2038-2059 START_RANGE client12l socket96 220 2059-2081 START_RANGE client12m socket97 220 2081-2103 START_RANGE client12n socket98 210 2103-2124 START_RANGE client12a socket99 220 2124-2146 START_RANGE client12b socket100 220 2146-2168 START_RANGE client12c socket101 220 2168-2190 START_RANGE client12d socket102 210 2190-2211 START_RANGE client12e socket103 220 2211-2233 START_RANGE client12f socket104 220 2233-2255 START_RANGE client12g socket105 210 2255-2276 START_RANGE client12h socket106 220 2276-2298 START_RANGE client12i socket107 220 2298-2320 START_RANGE client12j socket108 210 2320-2341 START_RANGE client12k socket109 220 2341-2363 START_RANGE client12l socket110 220 2363-2385 START_RANGE client12m socket111 210 2385-2406 START_RANGE client12n socket112 220 2406-2428</p> <p>START_RANGE client21a socket113 220 2428-2450 START_RANGE client21b socket114 210 2450-2471 START_RANGE client21c socket115 220 2471-2493 START_RANGE client21d socket116 220 2493-2515 START_RANGE client21e socket117 210 2515-2536 START_RANGE client21f socket118 220 2536-2558 START_RANGE client21g socket119 220 2558-2580 START_RANGE client21h socket120 220 2580-2602 START_RANGE client21i socket121 210 2602-2623 START_RANGE client21j socket122 220 2623-2645 START_RANGE client21k socket123 220 2645-2667 START_RANGE client21l socket124 210 2667-2688 START_RANGE client21m socket125 220 2688-2710 START_RANGE client21n socket126 220 2710-2732 START_RANGE client21a socket127 210 2732-2753 START_RANGE client21b socket128 220 2753-2775 START_RANGE client21c socket129 220 2775-2797 START_RANGE client21d socket130 210 2797-2818 START_RANGE client21e socket131 220 2818-2840 START_RANGE client21f socket132 220 2840-2862 START_RANGE client21g socket133 210 2862-2883 START_RANGE client21h socket134 220 2883-2905 START_RANGE client21i socket135 220 2905-2927 START_RANGE client21j socket136 210 2927-2948 START_RANGE client21k socket137 220 2948-2970</p>	<p>START_RANGE client21l socket138 220 2970-2992 START_RANGE client21m socket139 220 2992-3014 START_RANGE client21n socket140 210 3014-3035 START_RANGE client21a socket141 220 3035-3057 START_RANGE client21b socket142 220 3057-3079 START_RANGE client21c socket143 210 3079-3100 START_RANGE client21d socket144 220 3100-3122 START_RANGE client21e socket145 220 3122-3144 START_RANGE client21f socket146 210 3144-3165 START_RANGE client21g socket147 220 3165-3187 START_RANGE client21h socket148 220 3187-3209 START_RANGE client21i socket149 210 3209-3230 START_RANGE client21j socket150 220 3230-3252 START_RANGE client21k socket151 220 3252-3274 START_RANGE client21l socket152 210 3274-3295 START_RANGE client21m socket153 220 3295-3317 START_RANGE client21n socket154 220 3317-3339 START_RANGE client21a socket155 210 3339-3360 START_RANGE client21b socket156 220 3360-3382 START_RANGE client21c socket157 220 3382-3404 START_RANGE client21d socket158 220 3404-3426 START_RANGE client21e socket159 210 3426-3447 START_RANGE client21f socket160 220 3447-3469 START_RANGE client21g socket161 220 3469-3491 START_RANGE client21h socket162 210 3491-3512 START_RANGE client21i socket163 220 3512-3534 START_RANGE client21j socket164 220 3534-3556 START_RANGE client21k socket165 210 3556-3577 START_RANGE client21l socket166 220 3577-3599 START_RANGE client21m socket167 220 3599-3621 START_RANGE client21n socket168 210 3621-3642</p> <p>START_RANGE client22a socket169 220 3642-3664 START_RANGE client22b socket170 220 3664-3686 START_RANGE client22c socket171 210 3686-3707 START_RANGE client22d socket172 220 3707-3729 START_RANGE client22e socket173 220 3729-3751 START_RANGE client22f socket174 210 3751-3772 START_RANGE client22g socket175 220 3772-3794 START_RANGE client22h socket176 220 3794-3816 START_RANGE client22i socket177 220 3816-3838 START_RANGE client22j socket178 210 3838-3859 START_RANGE client22k socket179 220 3859-3881 START_RANGE client22l socket180 220 3881-3903 START_RANGE client22m socket181 210 3903-3924 START_RANGE client22n socket182 220 3924-3946 START_RANGE client22a socket183 220 3946-3968 START_RANGE client22b socket184 210 3968-3989 START_RANGE client22c socket185 220 3989-4011 START_RANGE client22d socket186 220 4011-4033 START_RANGE client22e socket187 210 4033-4054 START_RANGE client22f socket188 220 4054-4076 START_RANGE client22g socket189 220 4076-4098 START_RANGE client22h socket190 210 4098-4119 START_RANGE client22i socket191 220 4119-4141 START_RANGE client22j socket192 220 4141-4163 START_RANGE client22k socket193 210 4163-4184 START_RANGE client22l socket194 220 4184-4206 START_RANGE client22m socket195 220 4206-4228 START_RANGE client22n socket196 210 4228-4249 START_RANGE client22a socket197 220 4249-4271 START_RANGE client22b socket198 220 4271-4293 START_RANGE client22c socket199 220 4293-4315 START_RANGE client22d socket200 210 4315-4336 START_RANGE client22e socket201 220 4336-4358 START_RANGE client22f socket202 220 4358-4380 START_RANGE client22g socket203 210 4380-4401 START_RANGE client22h socket204 220 4401-4423 START_RANGE client22i socket205 220 4423-4445 START_RANGE client22j socket206 210 4445-4466 START_RANGE client22k socket207 220 4466-4488 START_RANGE client22l socket208 220 4488-4510 START_RANGE client22m socket209 210 4510-4531 START_RANGE client22n socket210 220 4531-4553 START_RANGE client22a socket211 220 4553-4575 START_RANGE client22b socket212 210 4575-4596 START_RANGE client22c socket213 220 4596-4618 START_RANGE client22d socket214 220 4618-4640 START_RANGE client22e socket215 210 4640-4661 START_RANGE client22f socket216 220 4661-4683 START_RANGE client22g socket217 220 4683-4705 START_RANGE client22h socket218 220 4705-4727 START_RANGE client22i socket219 210 4727-4748 START_RANGE client22j socket220 220 4748-4770 START_RANGE client22k socket221 220 4770-4792 START_RANGE client22l socket222 210 4792-4813 START_RANGE client22m socket223 220 4813-4835 START_RANGE client22n socket224 220 4835-4857</p> <p>#elif MASTER_NUM2 START_RANGE client31a socket225 210 4857-4878 START_RANGE client31b socket226 220 4878-4900 START_RANGE client31c socket227 220 4900-4922 START_RANGE client31d socket228 210 4922-4943 START_RANGE client31e socket229 220 4943-4965 START_RANGE client31f socket230 220 4965-4987 START_RANGE client31g socket231 210 4987-5008 START_RANGE client31h socket232 220 5008-5030 START_RANGE client31i socket233 220 5030-5052</p>
--	--

START_RANGE client31j socket234 210 5052-5073
START_RANGE client31k socket235 220 5073-5095
START_RANGE client31l socket236 220 5095-5117
START_RANGE client31m socket237 220 5117-5139
START_RANGE client31n socket238 210 5139-5160
START_RANGE client31a socket239 220 5160-5182
START_RANGE client31b socket240 220 5182-5204
START_RANGE client31c socket241 210 5204-5225
START_RANGE client31d socket242 220 5225-5247
START_RANGE client31e socket243 220 5247-5269
START_RANGE client31f socket244 210 5269-5290
START_RANGE client31g socket245 220 5290-5312
START_RANGE client31h socket246 220 5312-5334
START_RANGE client31i socket247 210 5334-5355
START_RANGE client31j socket248 220 5355-5377
START_RANGE client31k socket249 220 5377-5399
START_RANGE client31l socket250 210 5399-5420
START_RANGE client31m socket251 220 5420-5442
START_RANGE client31n socket252 220 5442-5464
START_RANGE client31a socket253 210 5464-5485
START_RANGE client31b socket254 220 5485-5507
START_RANGE client31c socket255 220 5507-5529
START_RANGE client31d socket256 220 5529-5551
START_RANGE client31e socket257 210 5551-5572
START_RANGE client31f socket258 220 5572-5594
START_RANGE client31g socket259 220 5594-5616
START_RANGE client31h socket260 210 5616-5637
START_RANGE client31i socket261 220 5637-5659
START_RANGE client31j socket262 220 5659-5681
START_RANGE client31k socket263 210 5681-5702
START_RANGE client31l socket264 220 5702-5724
START_RANGE client31m socket265 220 5724-5746
START_RANGE client31n socket266 210 5746-5767
START_RANGE client31a socket267 220 5767-5789
START_RANGE client31b socket268 220 5789-5811
START_RANGE client31c socket269 210 5811-5832
START_RANGE client31d socket270 220 5832-5854
START_RANGE client31e socket271 220 5854-5876
START_RANGE client31f socket272 210 5876-5897
START_RANGE client31g socket273 220 5897-5919
START_RANGE client31h socket274 220 5919-5941
START_RANGE client31i socket275 220 5941-5963
START_RANGE client31j socket276 210 5963-5984
START_RANGE client31k socket277 220 5984-6006
START_RANGE client31l socket278 220 6006-6028
START_RANGE client31m socket279 210 6028-6049
START_RANGE client31n socket280 220 6049-6071

START_RANGE client32a socket281 220 6071-6093
START_RANGE client32b socket282 210 6093-6114
START_RANGE client32c socket283 220 6114-6136
START_RANGE client32d socket284 220 6136-6158
START_RANGE client32e socket285 210 6158-6179
START_RANGE client32f socket286 220 6179-6201
START_RANGE client32g socket287 220 6201-6223
START_RANGE client32h socket288 210 6223-6244
START_RANGE client32i socket289 220 6244-6266
START_RANGE client32j socket290 220 6266-6288
START_RANGE client32k socket291 210 6288-6309
START_RANGE client32l socket292 220 6309-6331
START_RANGE client32m socket293 220 6331-6353
START_RANGE client32n socket294 210 6353-6374
START_RANGE client32a socket295 220 6374-6396
START_RANGE client32b socket296 220 6396-6418
START_RANGE client32c socket297 220 6418-6440
START_RANGE client32d socket298 210 6440-6461
START_RANGE client32e socket299 220 6461-6483
START_RANGE client32f socket300 220 6483-6505
START_RANGE client32g socket301 210 6505-6526
START_RANGE client32h socket302 220 6526-6548
START_RANGE client32i socket303 220 6548-6570
START_RANGE client32j socket304 210 6570-6591
START_RANGE client32k socket305 220 6591-6613
START_RANGE client32l socket306 220 6613-6635
START_RANGE client32m socket307 210 6635-6656
START_RANGE client32n socket308 220 6656-6678
START_RANGE client32a socket309 220 6678-6700
START_RANGE client32b socket310 210 6700-6721
START_RANGE client32c socket311 220 6721-6743
START_RANGE client32d socket312 220 6743-6765
START_RANGE client32e socket313 210 6765-6786
START_RANGE client32f socket314 220 6786-6808
START_RANGE client32g socket315 220 6808-6830
START_RANGE client32h socket316 220 6830-6852
START_RANGE client32i socket317 210 6852-6873
START_RANGE client32j socket318 220 6873-6895
START_RANGE client32k socket319 220 6895-6917
START_RANGE client32l socket320 210 6917-6938
START_RANGE client32m socket321 220 6938-6960
START_RANGE client32n socket322 220 6960-6982
START_RANGE client32a socket323 210 6982-7003
START_RANGE client32b socket324 220 7003-7025
START_RANGE client32c socket325 220 7025-7047
START_RANGE client32d socket326 210 7047-7068
START_RANGE client32e socket327 220 7068-7090
START_RANGE client32f socket328 220 7090-7112
START_RANGE client32g socket329 210 7112-7133
START_RANGE client32h socket330 220 7133-7155
START_RANGE client32i socket331 220 7155-7177

START_RANGE client32j socket332 210 7177-7198
START_RANGE client32k socket333 220 7198-7220
START_RANGE client32l socket334 220 7220-7242
START_RANGE client32m socket335 220 7242-7264
START_RANGE client32n socket336 210 7264-7285

START_RANGE client41a socket337 220 7285-7307
START_RANGE client41b socket338 220 7307-7329
START_RANGE client41c socket339 210 7329-7350
START_RANGE client41d socket340 220 7350-7372
START_RANGE client41e socket341 220 7372-7394
START_RANGE client41f socket342 210 7394-7415
START_RANGE client41g socket343 220 7415-7437
START_RANGE client41h socket344 220 7437-7459
START_RANGE client41i socket345 210 7459-7480
START_RANGE client41j socket346 220 7480-7502
START_RANGE client41k socket347 220 7502-7524
START_RANGE client41l socket348 210 7524-7545
START_RANGE client41m socket349 220 7545-7567
START_RANGE client41n socket350 220 7567-7589
START_RANGE client41a socket351 210 7589-7610
START_RANGE client41b socket352 220 7610-7632
START_RANGE client41c socket353 220 7632-7654
START_RANGE client41d socket354 220 7654-7676
START_RANGE client41e socket355 210 7676-7697
START_RANGE client41f socket356 220 7697-7719
START_RANGE client41g socket357 220 7719-7741
START_RANGE client41h socket358 210 7741-7762
START_RANGE client41i socket359 220 7762-7784
START_RANGE client41j socket360 220 7784-7806
START_RANGE client41k socket361 210 7806-7827
START_RANGE client41l socket362 220 7827-7849
START_RANGE client41m socket363 220 7849-7871
START_RANGE client41n socket364 210 7871-7892
START_RANGE client41a socket365 220 7892-7914
START_RANGE client41b socket366 220 7914-7936
START_RANGE client41c socket367 210 7936-7957
START_RANGE client41d socket368 220 7957-7979
START_RANGE client41e socket369 220 7979-8001
START_RANGE client41f socket370 210 8001-8022
START_RANGE client41g socket371 220 8022-8044
START_RANGE client41h socket372 220 8044-8066
START_RANGE client41i socket373 220 8066-8088
START_RANGE client41j socket374 210 8088-8109
START_RANGE client41k socket375 220 8109-8131
START_RANGE client41l socket376 220 8131-8153
START_RANGE client41m socket377 210 8153-8174
START_RANGE client41n socket378 220 8174-8196
START_RANGE client41a socket379 220 8196-8218
START_RANGE client41b socket380 210 8218-8239
START_RANGE client41c socket381 220 8239-8261
START_RANGE client41d socket382 220 8261-8283
START_RANGE client41e socket383 210 8283-8304
START_RANGE client41f socket384 220 8304-8326
START_RANGE client41g socket385 220 8326-8348
START_RANGE client41h socket386 210 8348-8369
START_RANGE client41i socket387 220 8369-8391
START_RANGE client41j socket388 220 8391-8413
START_RANGE client41k socket389 210 8413-8434
START_RANGE client41l socket390 220 8434-8456
START_RANGE client41m socket391 220 8456-8478
START_RANGE client41n socket392 210 8478-8499

START_RANGE client42a socket393 220 8499-8521
START_RANGE client42b socket394 220 8521-8543
START_RANGE client42c socket395 220 8543-8565
START_RANGE client42d socket396 210 8565-8586
START_RANGE client42e socket397 220 8586-8608
START_RANGE client42f socket398 220 8608-8630
START_RANGE client42g socket399 210 8630-8651
START_RANGE client42h socket400 220 8651-8673
START_RANGE client42i socket401 220 8673-8695
START_RANGE client42j socket402 210 8695-8716
START_RANGE client42k socket403 220 8716-8738
START_RANGE client42l socket404 220 8738-8760
START_RANGE client42m socket405 210 8760-8781
START_RANGE client42n socket406 220 8781-8803
START_RANGE client42a socket407 220 8803-8825
START_RANGE client42b socket408 210 8825-8846
START_RANGE client42c socket409 220 8846-8868
START_RANGE client42d socket410 220 8868-8890
START_RANGE client42e socket411 210 8890-8911
START_RANGE client42f socket412 220 8911-8933
START_RANGE client42g socket413 220 8933-8955
START_RANGE client42h socket414 220 8955-8977
START_RANGE client42i socket415 210 8977-8998
START_RANGE client42j socket416 220 8998-9020
START_RANGE client42k socket417 220 9020-9042
START_RANGE client42l socket418 210 9042-9063
START_RANGE client42m socket419 220 9063-9085
START_RANGE client42n socket420 220 9085-9107
START_RANGE client42a socket421 210 9107-9128
START_RANGE client42b socket422 220 9128-9150
START_RANGE client42c socket423 220 9150-9172
START_RANGE client42d socket424 210 9172-9193
START_RANGE client42e socket425 220 9193-9215
START_RANGE client42f socket426 220 9215-9237
START_RANGE client42g socket427 210 9237-9258
START_RANGE client42h socket428 220 9258-9280

```

START_RANGE client42i socket429 220 9280-9302
START_RANGE client42j socket430 210 9302-9323
START_RANGE client42k socket431 220 9323-9345
START_RANGE client42l socket432 220 9345-9367
START_RANGE client42m socket433 220 9367-9389
START_RANGE client42n socket434 210 9389-9410
START_RANGE client42a socket435 220 9410-9432
START_RANGE client42b socket436 220 9432-9454
START_RANGE client42c socket437 210 9454-9475
START_RANGE client42d socket438 220 9475-9497
START_RANGE client42e socket439 220 9497-9519
START_RANGE client42f socket440 210 9519-9540
START_RANGE client42g socket441 220 9540-9562
START_RANGE client42h socket442 220 9562-9584
START_RANGE client42i socket443 210 9584-9605
START_RANGE client42j socket444 220 9605-9627
START_RANGE client42k socket445 220 9627-9649
START_RANGE client42l socket446 210 9649-9670
START_RANGE client42m socket447 220 9670-9692
START_RANGE client42n socket448 220 9692-9714

#elif MASTER_NUM3
START_RANGE client51a socket449 210 9714-9735
START_RANGE client51b socket450 220 9735-9757
START_RANGE client51c socket451 220 9757-9779
START_RANGE client51d socket452 220 9779-9801
START_RANGE client51e socket453 210 9801-9822
START_RANGE client51f socket454 220 9822-9844
START_RANGE client51g socket455 220 9844-9866
START_RANGE client51h socket456 210 9866-9887
START_RANGE client51i socket457 220 9887-9909
START_RANGE client51j socket458 220 9909-9931
START_RANGE client51k socket459 210 9931-9952
START_RANGE client51l socket460 220 9952-9974
START_RANGE client51m socket461 220 9974-9996
START_RANGE client51n socket462 210 9996-10017
START_RANGE client51a socket463 220 10017-10039
START_RANGE client51b socket464 220 10039-10061
START_RANGE client51c socket465 210 10061-10082
START_RANGE client51d socket466 220 10082-10104
START_RANGE client51e socket467 220 10104-10126
START_RANGE client51f socket468 210 10126-10147
START_RANGE client51g socket469 220 10147-10169
START_RANGE client51h socket470 220 10169-10191
START_RANGE client51i socket471 220 10191-10213
START_RANGE client51j socket472 210 10213-10234
START_RANGE client51k socket473 220 10234-10256
START_RANGE client51l socket474 220 10256-10278
START_RANGE client51m socket475 210 10278-10299
START_RANGE client51n socket476 220 10299-10321
START_RANGE client51a socket477 220 10321-10343
START_RANGE client51b socket478 210 10343-10364
START_RANGE client51c socket479 220 10364-10386
START_RANGE client51d socket480 220 10386-10408
START_RANGE client51e socket481 210 10408-10429
START_RANGE client51f socket482 220 10429-10451
START_RANGE client51g socket483 220 10451-10473
START_RANGE client51h socket484 210 10473-10494
START_RANGE client51i socket485 220 10494-10516
START_RANGE client51j socket486 220 10516-10538
START_RANGE client51k socket487 210 10538-10559
START_RANGE client51l socket488 220 10559-10581
START_RANGE client51m socket489 220 10581-10603
START_RANGE client51n socket490 210 10603-10624
START_RANGE client51a socket491 220 10624-10646
START_RANGE client51b socket492 220 10646-10668
START_RANGE client51c socket493 220 10668-10690
START_RANGE client51d socket494 210 10690-10711
START_RANGE client51e socket495 220 10711-10733
START_RANGE client51f socket496 220 10733-10755
START_RANGE client51g socket497 210 10755-10776
START_RANGE client51h socket498 220 10776-10798
START_RANGE client51i socket499 220 10798-10820
START_RANGE client51j socket500 210 10820-10841
START_RANGE client51k socket501 220 10841-10863
START_RANGE client51l socket502 220 10863-10885
START_RANGE client51m socket503 210 10885-10906
START_RANGE client51n socket504 220 10906-10928

START_RANGE client52a socket505 220 10928-10950
START_RANGE client52b socket506 210 10950-10971
START_RANGE client52c socket507 220 10971-10993
START_RANGE client52d socket508 220 10993-11015
START_RANGE client52e socket509 210 11015-11036
START_RANGE client52f socket510 220 11036-11058
START_RANGE client52g socket511 220 11058-11080
START_RANGE client52h socket512 220 11080-11102
START_RANGE client52i socket513 210 11102-11123
START_RANGE client52j socket514 220 11123-11145
START_RANGE client52k socket515 220 11145-11167
START_RANGE client52l socket516 210 11167-11188
START_RANGE client52m socket517 220 11188-11210
START_RANGE client52n socket518 220 11210-11232
START_RANGE client52a socket519 210 11232-11253
START_RANGE client52b socket520 220 11253-11275
START_RANGE client52c socket521 220 11275-11297
START_RANGE client52d socket522 210 11297-11318
START_RANGE client52e socket523 220 11318-11340
START_RANGE client52f socket524 220 11340-11362

START_RANGE client52g socket525 210 11362-11383
START_RANGE client52h socket526 220 11383-11405
START_RANGE client52i socket527 220 11405-11427
START_RANGE client52j socket528 210 11427-11448
START_RANGE client52k socket529 220 11448-11470
START_RANGE client52l socket530 220 11470-11492
START_RANGE client52m socket531 220 11492-11514
START_RANGE client52n socket532 210 11514-11535
START_RANGE client52a socket533 220 11535-11557
START_RANGE client52b socket534 220 11557-11579
START_RANGE client52c socket535 210 11579-11600
START_RANGE client52d socket536 220 11600-11622
START_RANGE client52e socket537 220 11622-11644
START_RANGE client52f socket538 210 11644-11665
START_RANGE client52g socket539 220 11665-11687
START_RANGE client52h socket540 220 11687-11709
START_RANGE client52i socket541 210 11709-11730
START_RANGE client52j socket542 220 11730-11752
START_RANGE client52k socket543 220 11752-11774
START_RANGE client52l socket544 210 11774-11795
START_RANGE client52m socket545 220 11795-11817
START_RANGE client52n socket546 220 11817-11839
START_RANGE client52a socket547 210 11839-11860
START_RANGE client52b socket548 220 11860-11882
START_RANGE client52c socket549 220 11882-11904
START_RANGE client52d socket550 220 11904-11926
START_RANGE client52e socket551 210 11926-11947
START_RANGE client52f socket552 220 11947-11969
START_RANGE client52g socket553 220 11969-11991
START_RANGE client52h socket554 210 11991-12012
START_RANGE client52i socket555 220 12012-12034
START_RANGE client52j socket556 220 12034-12056
START_RANGE client52k socket557 210 12056-12077
START_RANGE client52l socket558 220 12077-12099
START_RANGE client52m socket559 220 12099-12121
START_RANGE client52n socket560 210 12121-12142

START_RANGE client61a socket561 220 12142-12164
START_RANGE client61b socket562 220 12164-12186
START_RANGE client61c socket563 210 12186-12207
START_RANGE client61d socket564 220 12207-12229
START_RANGE client61e socket565 220 12229-12251
START_RANGE client61f socket566 210 12251-12272
START_RANGE client61g socket567 220 12272-12294
START_RANGE client61h socket568 220 12294-12316
START_RANGE client61i socket569 220 12316-12338
START_RANGE client61j socket570 210 12338-12359
START_RANGE client61k socket571 220 12359-12381
START_RANGE client61l socket572 220 12381-12403
START_RANGE client61m socket573 210 12403-12424
START_RANGE client61n socket574 220 12424-12446
START_RANGE client61a socket575 220 12446-12468
START_RANGE client61b socket576 210 12468-12489
START_RANGE client61c socket577 220 12489-12511
START_RANGE client61d socket578 220 12511-12533
START_RANGE client61e socket579 210 12533-12554
START_RANGE client61f socket580 220 12554-12576
START_RANGE client61g socket581 220 12576-12598
START_RANGE client61h socket582 210 12598-12619
START_RANGE client61i socket583 220 12619-12641
START_RANGE client61j socket584 220 12641-12663
START_RANGE client61k socket585 210 12663-12684
START_RANGE client61l socket586 220 12684-12706
START_RANGE client61m socket587 220 12706-12728
START_RANGE client61n socket588 210 12728-12749
START_RANGE client61a socket589 220 12749-12771
START_RANGE client61b socket590 220 12771-12793
START_RANGE client61c socket591 220 12793-12815
START_RANGE client61d socket592 210 12815-12836
START_RANGE client61e socket593 220 12836-12858
START_RANGE client61f socket594 220 12858-12880
START_RANGE client61g socket595 210 12880-12901
START_RANGE client61h socket596 220 12901-12923
START_RANGE client61i socket597 220 12923-12945
START_RANGE client61j socket598 210 12945-12966
START_RANGE client61k socket599 220 12966-12988
START_RANGE client61l socket600 220 12988-13010
START_RANGE client61m socket601 210 13010-13031
START_RANGE client61n socket602 220 13031-13053
START_RANGE client61a socket603 220 13053-13075
START_RANGE client61b socket604 210 13075-13096
START_RANGE client61c socket605 220 13096-13118
START_RANGE client61d socket606 220 13118-13140
START_RANGE client61e socket607 210 13140-13161
START_RANGE client61f socket608 220 13161-13183
START_RANGE client61g socket609 220 13183-13205
START_RANGE client61h socket610 220 13205-13227
START_RANGE client61i socket611 210 13227-13248
START_RANGE client61j socket612 220 13248-13270
START_RANGE client61k socket613 220 13270-13292
START_RANGE client61l socket614 210 13292-13313
START_RANGE client61m socket615 220 13313-13335
START_RANGE client61n socket616 220 13335-13357

START_RANGE client62a socket617 210 13357-13378
START_RANGE client62b socket618 220 13378-13400
START_RANGE client62c socket619 220 13400-13422
START_RANGE client62d socket620 210 13422-13443
START_RANGE client62e socket621 220 13443-13465

```

```

START_RANGE client62fsocket622 220 13465-13487
START_RANGE client62gsocket623 210 13487-13508
START_RANGE client62h socket624 220 13508-13530
START_RANGE client62i socket625 220 13530-13552
START_RANGE client62j socket626 210 13552-13573
START_RANGE client62k socket627 220 13573-13595
START_RANGE client62l socket628 220 13595-13617
START_RANGE client62m socket629 220 13617-13639
START_RANGE client62n socket630 210 13639-13660
START_RANGE client62a socket631 220 13660-13682
START_RANGE client62b socket632 220 13682-13704
START_RANGE client62c socket633 210 13704-13725
START_RANGE client62d socket634 220 13725-13747
START_RANGE client62e socket635 220 13747-13769
START_RANGE client62f socket636 210 13769-13790
START_RANGE client62g socket637 220 13790-13812
START_RANGE client62h socket638 220 13812-13834
START_RANGE client62i socket639 210 13834-13855
START_RANGE client62j socket640 220 13855-13877
START_RANGE client62k socket641 220 13877-13899
START_RANGE client62l socket642 210 13899-13920
START_RANGE client62m socket643 220 13920-13942
START_RANGE client62n socket644 220 13942-13964
START_RANGE client62a socket645 210 13964-13985
START_RANGE client62b socket646 220 13985-14007
START_RANGE client62c socket647 220 14007-14029
START_RANGE client62d socket648 220 14029-14051
START_RANGE client62e socket649 210 14051-14072
START_RANGE client62f socket650 220 14072-14094
START_RANGE client62g socket651 220 14094-14116
START_RANGE client62h socket652 210 14116-14137
START_RANGE client62i socket653 220 14137-14159
START_RANGE client62j socket654 220 14159-14181
START_RANGE client62k socket655 210 14181-14202
START_RANGE client62l socket656 220 14202-14224
START_RANGE client62m socket657 220 14224-14246
START_RANGE client62n socket658 210 14246-14267
START_RANGE client62a socket659 220 14267-14289
START_RANGE client62b socket660 220 14289-14311
START_RANGE client62c socket661 210 14311-14332
START_RANGE client62d socket662 220 14332-14354
START_RANGE client62e socket663 220 14354-14376
START_RANGE client62f socket664 210 14376-14397
START_RANGE client62g socket665 220 14397-14419
START_RANGE client62h socket666 220 14419-14441
START_RANGE client62i socket667 220 14441-14463
START_RANGE client62j socket668 210 14463-14484
START_RANGE client62k socket669 220 14484-14506
START_RANGE client62l socket670 220 14506-14528
START_RANGE client62m socket671 210 14528-14549
START_RANGE client62n socket672 220 14549-14571

#elif MASTER_NUM4
START_RANGE client71a socket673 220 14571-14593
START_RANGE client71b socket674 210 14593-14614
START_RANGE client71c socket675 220 14614-14636
START_RANGE client71d socket676 220 14636-14658
START_RANGE client71e socket677 210 14658-14679
START_RANGE client71f socket678 220 14679-14701
START_RANGE client71g socket679 220 14701-14723
START_RANGE client71h socket680 210 14723-14744
START_RANGE client71i socket681 220 14744-14766
START_RANGE client71j socket682 220 14766-14788
START_RANGE client71k socket683 210 14788-14809
START_RANGE client71l socket684 220 14809-14831
START_RANGE client71m socket685 220 14831-14853
START_RANGE client71n socket686 210 14853-14874
START_RANGE client71a socket687 220 14874-14896
START_RANGE client71b socket688 220 14896-14918
START_RANGE client71c socket689 220 14918-14940
START_RANGE client71d socket690 210 14940-14961
START_RANGE client71e socket691 220 14961-14983
START_RANGE client71f socket692 220 14983-15005
START_RANGE client71g socket693 210 15005-15026
START_RANGE client71h socket694 220 15026-15048
START_RANGE client71i socket695 220 15048-15070
START_RANGE client71j socket696 210 15070-15091
START_RANGE client71k socket697 220 15091-15113
START_RANGE client71l socket698 220 15113-15135
START_RANGE client71m socket699 210 15135-15156
START_RANGE client71n socket700 220 15156-15178
START_RANGE client71a socket701 220 15178-15200
START_RANGE client71b socket702 210 15200-15221
START_RANGE client71c socket703 220 15221-15243
START_RANGE client71d socket704 220 15243-15265
START_RANGE client71e socket705 210 15265-15286
START_RANGE client71f socket706 220 15286-15308
START_RANGE client71g socket707 220 15308-15330
START_RANGE client71h socket708 220 15330-15352
START_RANGE client71i socket709 210 15352-15373
START_RANGE client71j socket710 220 15373-15395
START_RANGE client71k socket711 220 15395-15417
START_RANGE client71l socket712 210 15417-15438
START_RANGE client71m socket713 220 15438-15460
START_RANGE client71n socket714 220 15460-15482
START_RANGE client71a socket715 210 15482-15503
START_RANGE client71b socket716 220 15503-15525
START_RANGE client71c socket717 220 15525-15547
START_RANGE client71d socket718 210 15547-15568

START_RANGE client71e socket719 220 15568-15590
START_RANGE client71f socket720 220 15590-15612
START_RANGE client71g socket721 210 15612-15633
START_RANGE client71h socket722 220 15633-15655
START_RANGE client71i socket723 220 15655-15677
START_RANGE client71j socket724 210 15677-15698
START_RANGE client71k socket725 220 15698-15720
START_RANGE client71l socket726 220 15720-15742
START_RANGE client71m socket727 220 15742-15764
START_RANGE client71n socket728 210 15764-15785

START_RANGE client72a socket729 220 15785-15807
START_RANGE client72b socket730 220 15807-15829
START_RANGE client72c socket731 210 15829-15850
START_RANGE client72d socket732 220 15850-15872
START_RANGE client72e socket733 220 15872-15894
START_RANGE client72f socket734 210 15894-15915
START_RANGE client72g socket735 220 15915-15937
START_RANGE client72h socket736 220 15937-15959
START_RANGE client72i socket737 210 15959-15980
START_RANGE client72j socket738 220 15980-16002
START_RANGE client72k socket739 220 16002-16024
START_RANGE client72l socket740 210 16024-16045
START_RANGE client72m socket741 220 16045-16067
START_RANGE client72n socket742 220 16067-16089
START_RANGE client72a socket743 210 16089-16110
START_RANGE client72b socket744 220 16110-16132
START_RANGE client72c socket745 220 16132-16154
START_RANGE client72d socket746 220 16154-16176
START_RANGE client72e socket747 210 16176-16197
START_RANGE client72f socket748 220 16197-16219
START_RANGE client72g socket749 220 16219-16241
START_RANGE client72h socket750 210 16241-16262
START_RANGE client72i socket751 220 16262-16284
START_RANGE client72j socket752 220 16284-16306
START_RANGE client72k socket753 210 16306-16327
START_RANGE client72l socket754 220 16327-16349
START_RANGE client72m socket755 220 16349-16371
START_RANGE client72n socket756 210 16371-16392
START_RANGE client72a socket757 220 16392-16414
START_RANGE client72b socket758 220 16414-16436
START_RANGE client72c socket759 210 16436-16457
START_RANGE client72d socket760 220 16457-16479
START_RANGE client72e socket761 220 16479-16501
START_RANGE client72f socket762 210 16501-16522
START_RANGE client72g socket763 220 16522-16544
START_RANGE client72h socket764 220 16544-16566
START_RANGE client72i socket765 220 16566-16588
START_RANGE client72j socket766 210 16588-16609
START_RANGE client72k socket767 220 16609-16631
START_RANGE client72l socket768 220 16631-16653
START_RANGE client72m socket769 210 16653-16674
START_RANGE client72n socket770 220 16674-16696
START_RANGE client72a socket771 220 16696-16718
START_RANGE client72b socket772 210 16718-16739
START_RANGE client72c socket773 220 16739-16761
START_RANGE client72d socket774 220 16761-16783
START_RANGE client72e socket775 210 16783-16804
START_RANGE client72f socket776 220 16804-16826
START_RANGE client72g socket777 220 16826-16848
START_RANGE client72h socket778 210 16848-16869
START_RANGE client72i socket779 220 16869-16891
START_RANGE client72j socket780 220 16891-16913
START_RANGE client72k socket781 210 16913-16934
START_RANGE client72l socket782 220 16934-16956
START_RANGE client72m socket783 220 16956-16978
START_RANGE client72n socket784 210 16978-16999

START_RANGE client81a socket785 220 16999-17021
START_RANGE client81b socket786 220 17021-17043
START_RANGE client81c socket787 220 17043-17065
START_RANGE client81d socket788 210 17065-17086
START_RANGE client81e socket789 220 17086-17108
START_RANGE client81f socket790 220 17108-17130
START_RANGE client81g socket791 210 17130-17151
START_RANGE client81h socket792 220 17151-17173
START_RANGE client81i socket793 220 17173-17195
START_RANGE client81j socket794 210 17195-17216
START_RANGE client81k socket795 220 17216-17238
START_RANGE client81l socket796 220 17238-17260
START_RANGE client81m socket797 210 17260-17281
START_RANGE client81n socket798 220 17281-17303
START_RANGE client81a socket799 220 17303-17325
START_RANGE client81b socket800 210 17325-17346
START_RANGE client81c socket801 220 17346-17368
START_RANGE client81d socket802 220 17368-17390
START_RANGE client81e socket803 210 17390-17411
START_RANGE client81f socket804 220 17411-17433
START_RANGE client81g socket805 220 17433-17455
START_RANGE client81h socket806 220 17455-17477
START_RANGE client81i socket807 210 17477-17498
START_RANGE client81j socket808 220 17498-17520
START_RANGE client81k socket809 220 17520-17542
START_RANGE client81l socket810 210 17542-17563
START_RANGE client81m socket811 220 17563-17585
START_RANGE client81n socket812 220 17585-17607
START_RANGE client81a socket813 210 17607-17628
START_RANGE client81b socket814 220 17628-17650
START_RANGE client81c socket815 220 17650-17672

```

```

START_RANGE client81d socket816 210 17672-17693
START_RANGE client81e socket817 220 17693-17715
START_RANGE client81f socket818 220 17715-17737
START_RANGE client81g socket819 210 17737-17758
START_RANGE client81h socket820 220 17758-17780
START_RANGE client81i socket821 220 17780-17802
START_RANGE client81j socket822 210 17802-17823
START_RANGE client81k socket823 220 17823-17845
START_RANGE client81l socket824 220 17845-17867
START_RANGE client81m socket825 220 17867-17889
START_RANGE client81n socket826 210 17889-17910
START_RANGE client81a socket827 220 17910-17932
START_RANGE client81b socket828 220 17932-17954
START_RANGE client81c socket829 210 17954-17975
START_RANGE client81d socket830 220 17975-17997
START_RANGE client81e socket831 220 17997-18019
START_RANGE client81f socket832 210 18019-18040
START_RANGE client81g socket833 220 18040-18062
START_RANGE client81h socket834 220 18062-18084
START_RANGE client81i socket835 210 18084-18105
START_RANGE client81j socket836 220 18105-18127
START_RANGE client81k socket837 220 18127-18149
START_RANGE client81l socket838 210 18149-18170
START_RANGE client81m socket839 220 18170-18192
START_RANGE client81n socket840 220 18192-18214

START_RANGE client82a socket841 210 18214-18235
START_RANGE client82b socket842 220 18235-18257
START_RANGE client82c socket843 220 18257-18279
START_RANGE client82d socket844 220 18279-18301
START_RANGE client82e socket845 210 18301-18322
START_RANGE client82f socket846 220 18322-18344
START_RANGE client82g socket847 220 18344-18366
START_RANGE client82h socket848 210 18366-18387
START_RANGE client82i socket849 220 18387-18409
START_RANGE client82j socket850 220 18409-18431
START_RANGE client82k socket851 210 18431-18452
START_RANGE client82l socket852 220 18452-18474
START_RANGE client82m socket853 220 18474-18496
START_RANGE client82n socket854 210 18496-18517
START_RANGE client82a socket855 220 18517-18539
START_RANGE client82b socket856 220 18539-18561
START_RANGE client82c socket857 210 18561-18582
START_RANGE client82d socket858 220 18582-18604
START_RANGE client82e socket859 220 18604-18626
START_RANGE client82f socket860 210 18626-18647
START_RANGE client82g socket861 220 18647-18669
START_RANGE client82h socket862 220 18669-18691
START_RANGE client82i socket863 220 18691-18713
START_RANGE client82j socket864 210 18713-18734
START_RANGE client82k socket865 220 18734-18756
START_RANGE client82l socket866 220 18756-18778
START_RANGE client82m socket867 210 18778-18799
START_RANGE client82n socket868 220 18799-18821
START_RANGE client82a socket869 220 18821-18843
START_RANGE client82b socket870 210 18843-18864
START_RANGE client82c socket871 220 18864-18886
START_RANGE client82d socket872 220 18886-18908
START_RANGE client82e socket873 210 18908-18929
START_RANGE client82f socket874 220 18929-18951
START_RANGE client82g socket875 220 18951-18973
START_RANGE client82h socket876 210 18973-18994
START_RANGE client82i socket877 220 18994-19016
START_RANGE client82j socket878 220 19016-19038
START_RANGE client82k socket879 210 19038-19059
START_RANGE client82l socket880 220 19059-19081
START_RANGE client82m socket881 220 19081-19103
START_RANGE client82n socket882 210 19103-19124
START_RANGE client82a socket883 220 19124-19146
START_RANGE client82b socket884 220 19146-19168
START_RANGE client82c socket885 220 19168-19190
START_RANGE client82d socket886 210 19190-19211
START_RANGE client82e socket887 220 19211-19233
START_RANGE client82f socket888 220 19233-19255
START_RANGE client82g socket889 210 19255-19276
START_RANGE client82h socket890 220 19276-19298
START_RANGE client82i socket891 220 19298-19320
START_RANGE client82j socket892 210 19320-19341
START_RANGE client82k socket893 220 19341-19363
START_RANGE client82l socket894 220 19363-19385
START_RANGE client82m socket895 210 19385-19406
START_RANGE client82n socket896 220 19406-19428

#elif MASTER_NUM5
START_RANGE client91a socket897 220 19428-19450
START_RANGE client91b socket898 210 19450-19471
START_RANGE client91c socket899 220 19471-19493
START_RANGE client91d socket900 220 19493-19515
START_RANGE client91e socket901 210 19515-19536
START_RANGE client91f socket902 220 19536-19558
START_RANGE client91g socket903 220 19558-19580
START_RANGE client91h socket904 220 19580-19602
START_RANGE client91i socket905 210 19602-19623
START_RANGE client91j socket906 220 19623-19645
START_RANGE client91k socket907 220 19645-19667
START_RANGE client91l socket908 210 19667-19688
START_RANGE client91m socket909 220 19688-19710
START_RANGE client91n socket910 220 19710-19732
START_RANGE client91a socket911 210 19732-19753

START_RANGE client91b socket912 220 19753-19775
START_RANGE client91c socket913 220 19775-19797
START_RANGE client91d socket914 210 19797-19818
START_RANGE client91e socket915 220 19818-19840
START_RANGE client91f socket916 220 19840-19862
START_RANGE client91g socket917 210 19862-19883
START_RANGE client91h socket918 220 19883-19905
START_RANGE client91i socket919 220 19905-19927
START_RANGE client91j socket920 210 19927-19948
START_RANGE client91k socket921 220 19948-19970
START_RANGE client91l socket922 220 19970-19992
START_RANGE client91m socket923 220 19992-20014
START_RANGE client91n socket924 210 20014-20035
START_RANGE client91a socket925 220 20035-20057
START_RANGE client91b socket926 220 20057-20079
START_RANGE client91c socket927 210 20079-20100
START_RANGE client91d socket928 220 20100-20122
START_RANGE client91e socket929 220 20122-20144
START_RANGE client91f socket930 210 20144-20165
START_RANGE client91g socket931 220 20165-20187
START_RANGE client91h socket932 220 20187-20209
START_RANGE client91i socket933 210 20209-20230
START_RANGE client91j socket934 220 20230-20252
START_RANGE client91k socket935 220 20252-20274
START_RANGE client91l socket936 210 20274-20295
START_RANGE client91m socket937 220 20295-20317
START_RANGE client91n socket938 220 20317-20339
START_RANGE client91a socket939 210 20339-20360
START_RANGE client91b socket940 220 20360-20382
START_RANGE client91c socket941 220 20382-20404
START_RANGE client91d socket942 220 20404-20426
START_RANGE client91e socket943 210 20426-20447
START_RANGE client91f socket944 220 20447-20469
START_RANGE client91g socket945 220 20469-20491
START_RANGE client91h socket946 210 20491-20513
START_RANGE client91i socket947 220 20513-20534
START_RANGE client91j socket948 220 20534-20556
START_RANGE client91k socket949 210 20556-20577
START_RANGE client91l socket950 220 20577-20599
START_RANGE client91m socket951 220 20599-20621
START_RANGE client91n socket952 210 20621-20642

START_RANGE client92a socket953 220 20642-20664
START_RANGE client92b socket954 220 20664-20686
START_RANGE client92c socket955 210 20686-20707
START_RANGE client92d socket956 220 20707-20729
START_RANGE client92e socket957 220 20729-20751
START_RANGE client92f socket958 210 20751-20772
START_RANGE client92g socket959 220 20772-20794
START_RANGE client92h socket960 220 20794-20816
START_RANGE client92i socket961 220 20816-20838
START_RANGE client92j socket962 210 20838-20859
START_RANGE client92k socket963 220 20859-20881
START_RANGE client92l socket964 220 20881-20903
START_RANGE client92m socket965 210 20903-20924
START_RANGE client92n socket966 220 20924-20946
START_RANGE client92a socket967 220 20946-20968
START_RANGE client92b socket968 210 20968-20989
START_RANGE client92c socket969 220 20989-21011
START_RANGE client92d socket970 220 21011-21033
START_RANGE client92e socket971 210 21033-21054
START_RANGE client92f socket972 220 21054-21076
START_RANGE client92g socket973 220 21076-21098
START_RANGE client92h socket974 210 21098-21119
START_RANGE client92i socket975 220 21119-21141
START_RANGE client92j socket976 220 21141-21163
START_RANGE client92k socket977 210 21163-21184
START_RANGE client92l socket978 220 21184-21206
START_RANGE client92m socket979 220 21206-21228
START_RANGE client92n socket980 210 21228-21249
START_RANGE client92a socket981 220 21249-21271
START_RANGE client92b socket982 220 21271-21293
START_RANGE client92c socket983 220 21293-21315
START_RANGE client92d socket984 210 21315-21336
START_RANGE client92e socket985 220 21336-21358
START_RANGE client92f socket986 220 21358-21380
START_RANGE client92g socket987 210 21380-21401
START_RANGE client92h socket988 220 21401-21423
START_RANGE client92i socket989 220 21423-21445
START_RANGE client92j socket990 210 21445-21466
START_RANGE client92k socket991 220 21466-21488
START_RANGE client92l socket992 220 21488-21510
START_RANGE client92m socket993 210 21510-21531
START_RANGE client92n socket994 220 21531-21553
START_RANGE client92a socket995 220 21553-21575
START_RANGE client92b socket996 210 21575-21596
START_RANGE client92c socket997 220 21596-21618
START_RANGE client92d socket998 220 21618-21640
START_RANGE client92e socket999 210 21640-21661
START_RANGE client92f socket1000 220 21661-21683
START_RANGE client92g socket1001 220 21683-21705
START_RANGE client92h socket1002 220 21705-21727
START_RANGE client92i socket1003 210 21727-21748
START_RANGE client92j socket1004 220 21748-21770
START_RANGE client92k socket1005 220 21770-21792
START_RANGE client92l socket1006 210 21792-21813
START_RANGE client92m socket1007 220 21813-21835
START_RANGE client92n socket1008 220 21835-21857

```


START_RANGE client101a socket1009 210 21857-21878	START_RANGE client102a socket1107 210 23982-24003
START_RANGE client101b socket1010 220 21878-21900	START_RANGE client102b socket1108 220 24003-24025
START_RANGE client101c socket1011 220 21900-21922	START_RANGE client102c socket1109 220 24025-24047
START_RANGE client101d socket1012 210 21922-21943	START_RANGE client102d socket1110 210 24047-24068
START_RANGE client101e socket1013 220 21943-21965	START_RANGE client102e socket1111 220 24068-24090
START_RANGE client101f socket1014 220 21965-21987	START_RANGE client102f socket1112 220 24090-24112
START_RANGE client101g socket1015 210 21987-22008	START_RANGE client102g socket1113 210 24112-24133
START_RANGE client101h socket1016 220 22008-22030	START_RANGE client102h socket1114 220 24133-24155
START_RANGE client101i socket1017 220 22030-22052	START_RANGE client102i socket1115 220 24155-24177
START_RANGE client101j socket1018 210 22052-22073	START_RANGE client102j socket1116 210 24177-24198
START_RANGE client101k socket1019 220 22073-22095	START_RANGE client102k socket1117 220 24198-24220
START_RANGE client101l socket1020 220 22095-22117	START_RANGE client102l socket1118 220 24220-24242
START_RANGE client101m socket1021 220 22117-22139	START_RANGE client102m socket1119 220 24242-24264
START_RANGE client101n socket1022 210 22139-22160	START_RANGE client102n socket1120 210 24264-24285
START_RANGE client101a socket1023 220 22160-22182	
START_RANGE client101b socket1024 220 22182-22204	#elif MASTER_NUM6
START_RANGE client101c socket1025 210 22204-22225	START_RANGE client111a socket1121 220 24285-24307
START_RANGE client101d socket1026 220 22225-22247	START_RANGE client111b socket1122 220 24307-24329
START_RANGE client101e socket1027 220 22247-22269	START_RANGE client111c socket1123 210 24329-24350
START_RANGE client101f socket1028 210 22269-22290	START_RANGE client111d socket1124 220 24350-24372
START_RANGE client101g socket1029 220 22290-22312	START_RANGE client111e socket1125 220 24372-24394
START_RANGE client101h socket1030 220 22312-22334	START_RANGE client111f socket1126 210 24394-24415
START_RANGE client101i socket1031 210 22334-22355	START_RANGE client111g socket1127 220 24415-24437
START_RANGE client101j socket1032 220 22355-22377	START_RANGE client111h socket1128 220 24437-24459
START_RANGE client101k socket1033 220 22377-22399	START_RANGE client111i socket1129 210 24459-24480
START_RANGE client101l socket1034 210 22399-22420	START_RANGE client111j socket1130 220 24480-24502
START_RANGE client101m socket1035 220 22420-22442	START_RANGE client111k socket1131 220 24502-24524
START_RANGE client101n socket1036 220 22442-22464	START_RANGE client111l socket1132 210 24524-24545
START_RANGE client101a socket1037 210 22464-22485	START_RANGE client111m socket1133 220 24545-24567
START_RANGE client101b socket1038 220 22485-22507	START_RANGE client111n socket1134 220 24567-24589
START_RANGE client101c socket1039 220 22507-22529	START_RANGE client111a socket1135 210 24589-24610
START_RANGE client101d socket1040 220 22529-22551	START_RANGE client111b socket1136 220 24610-24632
START_RANGE client101e socket1041 210 22551-22572	START_RANGE client111c socket1137 220 24632-24654
START_RANGE client101f socket1042 220 22572-22594	START_RANGE client111d socket1138 220 24654-24676
START_RANGE client101g socket1043 220 22594-22616	START_RANGE client111e socket1139 210 24676-24697
START_RANGE client101h socket1044 210 22616-22637	START_RANGE client111f socket1140 220 24697-24719
START_RANGE client101i socket1045 220 22637-22659	START_RANGE client111g socket1141 220 24719-24741
START_RANGE client101j socket1046 220 22659-22681	START_RANGE client111h socket1142 210 24741-24762
START_RANGE client101k socket1047 210 22681-22702	START_RANGE client111i socket1143 220 24762-24784
START_RANGE client101l socket1048 220 22702-22724	START_RANGE client111j socket1144 220 24784-24806
START_RANGE client101m socket1049 220 22724-22746	START_RANGE client111k socket1145 210 24806-24827
START_RANGE client101n socket1050 210 22746-22767	START_RANGE client111l socket1146 220 24827-24849
START_RANGE client101a socket1051 220 22767-22789	START_RANGE client111m socket1147 220 24849-24871
START_RANGE client101b socket1052 220 22789-22811	START_RANGE client111n socket1148 210 24871-24892
START_RANGE client101c socket1053 210 22811-22832	START_RANGE client111a socket1149 220 24892-24914
START_RANGE client101d socket1054 220 22832-22854	START_RANGE client111b socket1150 220 24914-24936
START_RANGE client101e socket1055 220 22854-22876	START_RANGE client111c socket1151 210 24936-24957
START_RANGE client101f socket1056 210 22876-22897	START_RANGE client111d socket1152 220 24957-24979
START_RANGE client101g socket1057 220 22897-22919	START_RANGE client111e socket1153 220 24979-25001
START_RANGE client101h socket1058 220 22919-22941	START_RANGE client111f socket1154 210 25001-25022
START_RANGE client101i socket1059 220 22941-22963	START_RANGE client111g socket1155 220 25022-25044
START_RANGE client101j socket1060 210 22963-22984	START_RANGE client111h socket1156 220 25044-25066
START_RANGE client101k socket1061 220 22984-23006	START_RANGE client111i socket1157 220 25066-25088
START_RANGE client101l socket1062 220 23006-23028	START_RANGE client111j socket1158 210 25088-25109
START_RANGE client101m socket1063 210 23028-23049	START_RANGE client111k socket1159 220 25109-25131
START_RANGE client101n socket1064 220 23049-23071	START_RANGE client111l socket1160 220 25131-25153
	START_RANGE client111m socket1161 210 25153-25174
START_RANGE client102a socket1065 220 23071-23093	START_RANGE client111n socket1162 220 25174-25196
START_RANGE client102b socket1066 210 23093-23114	START_RANGE client111a socket1163 220 25196-25218
START_RANGE client102c socket1067 220 23114-23136	START_RANGE client111b socket1164 210 25218-25239
START_RANGE client102d socket1068 220 23136-23158	START_RANGE client111c socket1165 220 25239-25261
START_RANGE client102e socket1069 210 23158-23179	START_RANGE client111d socket1166 220 25261-25283
START_RANGE client102f socket1070 220 23179-23201	START_RANGE client111e socket1167 210 25283-25304
START_RANGE client102g socket1071 220 23201-23223	START_RANGE client111f socket1168 220 25304-25326
START_RANGE client102h socket1072 210 23223-23244	START_RANGE client111g socket1169 220 25326-25348
START_RANGE client102i socket1073 220 23244-23266	START_RANGE client111h socket1170 210 25348-25369
START_RANGE client102j socket1074 220 23266-23288	START_RANGE client111i socket1171 220 25369-25391
START_RANGE client102k socket1075 210 23288-23309	START_RANGE client111j socket1172 220 25391-25413
START_RANGE client102l socket1076 220 23309-23331	START_RANGE client111k socket1173 210 25413-25434
START_RANGE client102m socket1077 220 23331-23353	START_RANGE client111l socket1174 220 25434-25456
START_RANGE client102n socket1078 210 23353-23374	START_RANGE client111m socket1175 220 25456-25478
START_RANGE client102a socket1079 220 23374-23396	START_RANGE client111n socket1176 210 25478-25499
START_RANGE client102b socket1080 220 23396-23418	
START_RANGE client102c socket1081 220 23418-23440	START_RANGE client112a socket1177 220 25499-25521
START_RANGE client102d socket1082 210 23440-23461	START_RANGE client112b socket1178 220 25521-25543
START_RANGE client102e socket1083 220 23461-23483	START_RANGE client112c socket1179 220 25543-25565
START_RANGE client102f socket1084 220 23483-23505	START_RANGE client112d socket1180 210 25565-25587
START_RANGE client102g socket1085 210 23505-23526	START_RANGE client112e socket1181 220 25587-25608
START_RANGE client102h socket1086 220 23526-23548	START_RANGE client112f socket1182 220 25608-25630
START_RANGE client102i socket1087 220 23548-23570	START_RANGE client112g socket1183 210 25630-25651
START_RANGE client102j socket1088 210 23570-23591	START_RANGE client112h socket1184 220 25651-25673
START_RANGE client102k socket1089 220 23591-23613	START_RANGE client112i socket1185 220 25673-25695
START_RANGE client102l socket1090 220 23613-23635	START_RANGE client112j socket1186 210 25695-25717
START_RANGE client102m socket1091 210 23635-23656	START_RANGE client112k socket1187 220 25717-25738
START_RANGE client102n socket1092 220 23656-23678	START_RANGE client112l socket1188 220 25738-25760
START_RANGE client102a socket1093 220 23678-23700	START_RANGE client112m socket1189 210 25760-25781
START_RANGE client102b socket1094 210 23700-23721	START_RANGE client112n socket1190 220 25781-25803
START_RANGE client102c socket1095 220 23721-23743	START_RANGE client112a socket1191 220 25803-25825
START_RANGE client102d socket1096 220 23743-23765	START_RANGE client112b socket1192 210 25825-25846
START_RANGE client102e socket1097 210 23765-23786	START_RANGE client112c socket1193 220 25846-25868
START_RANGE client102f socket1098 220 23786-23808	START_RANGE client112d socket1194 220 25868-25890
START_RANGE client102g socket1099 220 23808-23830	START_RANGE client112e socket1195 210 25890-25911
START_RANGE client102h socket1100 220 23830-23852	START_RANGE client112f socket1196 220 25911-25933
START_RANGE client102i socket1101 210 23852-23873	START_RANGE client112g socket1197 220 25933-25955
START_RANGE client102j socket1102 220 23873-23895	START_RANGE client112h socket1198 220 25955-25977
START_RANGE client102k socket1103 220 23895-23917	START_RANGE client112i socket1199 210 25977-25998
START_RANGE client102l socket1104 210 23917-23938	START_RANGE client112j socket1200 220 25998-26020
START_RANGE client102m socket1105 220 23938-23960	START_RANGE client112k socket1201 220 26020-26042
START_RANGE client102n socket1106 220 23960-23982	START_RANGE client112l socket1202 210 26042-26063

START_RANGE client112m socket1203 220 26063-26085	START_RANGE client122l socket1300 210 28167-28188
START_RANGE client112n socket1204 220 26085-26107	START_RANGE client122m socket1301 220 28188-28210
START_RANGE client112a socket1205 210 26107-26128	START_RANGE client122n socket1302 220 28210-28232
START_RANGE client112b socket1206 220 26128-26150	START_RANGE client122a socket1303 210 28232-28253
START_RANGE client112c socket1207 220 26150-26172	START_RANGE client122b socket1304 220 28253-28275
START_RANGE client112d socket1208 210 26172-26193	START_RANGE client122c socket1305 220 28275-28297
START_RANGE client112e socket1209 220 26193-26215	START_RANGE client122d socket1306 210 28297-28318
START_RANGE client112f socket1210 220 26215-26237	START_RANGE client122e socket1307 220 28318-28340
START_RANGE client112g socket1211 210 26237-26258	START_RANGE client122f socket1308 220 28340-28362
START_RANGE client112h socket1212 220 26258-26280	START_RANGE client122g socket1309 210 28362-28383
START_RANGE client112i socket1213 220 26280-26302	START_RANGE client122h socket1310 220 28383-28405
START_RANGE client112j socket1214 210 26302-26323	START_RANGE client122i socket1311 220 28405-28427
START_RANGE client112k socket1215 220 26323-26345	START_RANGE client122j socket1312 210 28427-28448
START_RANGE client112l socket1216 220 26345-26367	START_RANGE client122k socket1313 220 28448-28470
START_RANGE client112m socket1217 220 26367-26389	START_RANGE client122l socket1314 220 28470-28492
START_RANGE client112n socket1218 210 26389-26410	START_RANGE client122m socket1315 220 28492-28514
START_RANGE client112a socket1219 220 26410-26432	START_RANGE client122n socket1316 210 28514-28535
START_RANGE client112b socket1220 220 26432-26454	START_RANGE client122a socket1317 220 28535-28557
START_RANGE client112c socket1221 210 26454-26475	START_RANGE client122b socket1318 220 28557-28579
START_RANGE client112d socket1222 220 26475-26497	START_RANGE client122c socket1319 210 28579-28600
START_RANGE client112e socket1223 220 26497-26519	START_RANGE client122d socket1320 220 28600-28622
START_RANGE client112f socket1224 210 26519-26540	START_RANGE client122e socket1321 220 28622-28644
START_RANGE client112g socket1225 220 26540-26562	START_RANGE client122f socket1322 210 28644-28665
START_RANGE client112h socket1226 220 26562-26584	START_RANGE client122g socket1323 220 28665-28687
START_RANGE client112i socket1227 210 26584-26605	START_RANGE client122h socket1324 220 28687-28709
START_RANGE client112j socket1228 220 26605-26627	START_RANGE client122i socket1325 210 28709-28730
START_RANGE client112k socket1229 220 26627-26649	START_RANGE client122j socket1326 220 28730-28752
START_RANGE client112l socket1230 210 26649-26670	START_RANGE client122k socket1327 220 28752-28774
START_RANGE client112m socket1231 220 26670-26692	START_RANGE client122l socket1328 210 28774-28795
START_RANGE client112n socket1232 220 26692-26714	START_RANGE client122m socket1329 220 28795-28817
START_RANGE client121a socket1233 210 26714-26735	START_RANGE client122n socket1330 220 28817-28839
START_RANGE client121b socket1234 220 26735-26757	START_RANGE client122a socket1331 210 28839-28860
START_RANGE client121c socket1235 220 26757-26779	START_RANGE client122b socket1332 220 28860-28882
START_RANGE client121d socket1236 220 26779-26801	START_RANGE client122c socket1333 220 28882-28904
START_RANGE client121e socket1237 210 26801-26822	START_RANGE client122d socket1334 220 28904-28926
START_RANGE client121f socket1238 220 26822-26844	START_RANGE client122e socket1335 210 28926-28947
START_RANGE client121g socket1239 220 26844-26866	START_RANGE client122f socket1336 220 28947-28969
START_RANGE client121h socket1240 210 26866-26887	START_RANGE client122g socket1337 220 28969-28991
START_RANGE client121i socket1241 220 26887-26909	START_RANGE client122h socket1338 210 28991-29012
START_RANGE client121j socket1242 220 26909-26931	START_RANGE client122i socket1339 220 29012-29034
START_RANGE client121k socket1243 210 26931-26952	START_RANGE client122j socket1340 220 29034-29056
START_RANGE client121l socket1244 220 26952-26974	START_RANGE client122k socket1341 210 29056-29077
START_RANGE client121m socket1245 220 26974-26996	START_RANGE client122l socket1342 220 29077-29099
START_RANGE client121n socket1246 210 26996-27017	START_RANGE client122m socket1343 220 29099-29121
START_RANGE client121a socket1247 220 27017-27039	START_RANGE client122n socket1344 210 29121-29142
START_RANGE client121b socket1248 220 27039-27061	
START_RANGE client121c socket1249 210 27061-27082	#elif MASTER_NUM7
START_RANGE client121d socket1250 220 27082-27104	START_RANGE client131a socket1345 220 29142-29164
START_RANGE client121e socket1251 220 27104-27126	START_RANGE client131b socket1346 220 29164-29186
START_RANGE client121f socket1252 210 27126-27147	START_RANGE client131c socket1347 210 29186-29207
START_RANGE client121g socket1253 220 27147-27169	START_RANGE client131d socket1348 220 29207-29229
START_RANGE client121h socket1254 220 27169-27191	START_RANGE client131e socket1349 220 29229-29251
START_RANGE client121i socket1255 220 27191-27213	START_RANGE client131f socket1350 210 29251-29272
START_RANGE client121j socket1256 210 27213-27234	START_RANGE client131g socket1351 220 29272-29294
START_RANGE client121k socket1257 220 27234-27256	START_RANGE client131h socket1352 220 29294-29316
START_RANGE client121l socket1258 220 27256-27278	START_RANGE client131i socket1353 220 29316-29338
START_RANGE client121m socket1259 210 27278-27299	START_RANGE client131j socket1354 210 29338-29359
START_RANGE client121n socket1260 220 27299-27321	START_RANGE client131k socket1355 220 29359-29381
START_RANGE client121a socket1261 220 27321-27343	START_RANGE client131l socket1356 220 29381-29403
START_RANGE client121b socket1262 210 27343-27364	START_RANGE client131m socket1357 210 29403-29424
START_RANGE client121c socket1263 220 27364-27386	START_RANGE client131n socket1358 220 29424-29446
START_RANGE client121d socket1264 220 27386-27408	START_RANGE client131a socket1359 220 29446-29468
START_RANGE client121e socket1265 210 27408-27429	START_RANGE client131b socket1360 210 29468-29489
START_RANGE client121f socket1266 220 27429-27451	START_RANGE client131c socket1361 220 29489-29511
START_RANGE client121g socket1267 220 27451-27473	START_RANGE client131d socket1362 220 29511-29533
START_RANGE client121h socket1268 210 27473-27494	START_RANGE client131e socket1363 210 29533-29554
START_RANGE client121i socket1269 220 27494-27516	START_RANGE client131f socket1364 220 29554-29576
START_RANGE client121j socket1270 220 27516-27538	START_RANGE client131g socket1365 220 29576-29598
START_RANGE client121k socket1271 210 27538-27559	START_RANGE client131h socket1366 210 29598-29619
START_RANGE client121l socket1272 220 27559-27581	START_RANGE client131i socket1367 220 29619-29641
START_RANGE client121m socket1273 220 27581-27603	START_RANGE client131j socket1368 220 29641-29663
START_RANGE client121n socket1274 210 27603-27624	START_RANGE client131k socket1369 210 29663-29684
START_RANGE client121a socket1275 220 27624-27646	START_RANGE client131l socket1370 220 29684-29706
START_RANGE client121b socket1276 220 27646-27668	START_RANGE client131m socket1371 220 29706-29728
START_RANGE client121c socket1277 220 27668-27690	START_RANGE client131n socket1372 210 29728-29749
START_RANGE client121d socket1278 210 27690-27711	START_RANGE client131a socket1373 220 29749-29771
START_RANGE client121e socket1279 220 27711-27733	START_RANGE client131b socket1374 220 29771-29793
START_RANGE client121f socket1280 220 27733-27755	START_RANGE client131c socket1375 220 29793-29815
START_RANGE client121g socket1281 210 27755-27776	START_RANGE client131d socket1376 210 29815-29836
START_RANGE client121h socket1282 220 27776-27798	START_RANGE client131e socket1377 220 29836-29858
START_RANGE client121i socket1283 220 27798-27820	START_RANGE client131f socket1378 220 29858-29880
START_RANGE client121j socket1284 210 27820-27841	START_RANGE client131g socket1379 210 29880-29901
START_RANGE client121k socket1285 220 27841-27863	START_RANGE client131h socket1380 220 29901-29923
START_RANGE client121l socket1286 220 27863-27885	START_RANGE client131i socket1381 220 29923-29944
START_RANGE client121m socket1287 210 27885-27906	START_RANGE client131j socket1382 210 29944-29966
START_RANGE client121n socket1288 220 27906-27928	START_RANGE client131k socket1383 220 29966-29988
	START_RANGE client131l socket1384 220 29988-30010
START_RANGE client122a socket1289 220 27928-27950	START_RANGE client131m socket1385 210 30010-30031
START_RANGE client122b socket1290 210 27950-27971	START_RANGE client131n socket1386 220 30031-30053
START_RANGE client122c socket1291 220 27971-27993	START_RANGE client131a socket1387 220 30053-30075
START_RANGE client122d socket1292 220 27993-28015	START_RANGE client131b socket1388 210 30075-30096
START_RANGE client122e socket1293 210 28015-28036	START_RANGE client131c socket1389 220 30096-30118
START_RANGE client122f socket1294 220 28036-28058	START_RANGE client131d socket1390 220 30118-30140
START_RANGE client122g socket1295 220 28058-28080	START_RANGE client131e socket1391 210 30140-30161
START_RANGE client122h socket1296 220 28080-28102	START_RANGE client131f socket1392 220 30161-30183
START_RANGE client122i socket1297 210 28102-28123	START_RANGE client131g socket1393 220 30183-30205
START_RANGE client122j socket1298 220 28123-28145	START_RANGE client131h socket1394 220 30205-30227
START_RANGE client122k socket1299 220 28145-28167	START_RANGE client131i socket1395 210 30227-30248
	START_RANGE client131j socket1396 220 30248-30270

```
START_RANGE client131ksocket1397 220 30270-30292
START_RANGE client131lsocket1398 210 30292-30313
START_RANGE client131msocket1399 220 30313-30335
START_RANGE client131nsocket1400 220 30335-30357

START_RANGE client132asocket1401 210 30357-30378
START_RANGE client132bsocket1402 220 30378-30400
START_RANGE client132csocket1403 220 30400-30422
START_RANGE client132dsocket1404 210 30422-30443
START_RANGE client132esocket1405 220 30443-30465
START_RANGE client132fssocket1406 220 30465-30487
START_RANGE client132gsocket1407 210 30487-30508
START_RANGE client132hsocket1408 220 30508-30530
START_RANGE client132isocket1409 220 30530-30552
START_RANGE client132jsocket1410 210 30552-30573
START_RANGE client132ksocket1411 220 30573-30595
START_RANGE client132lsocket1412 220 30595-30617
START_RANGE client132msocket1413 220 30617-30639
START_RANGE client132nsocket1414 210 30639-30660
START_RANGE client132asocket1415 220 30660-30682
START_RANGE client132bsocket1416 220 30682-30704
START_RANGE client132csocket1417 210 30704-30725
START_RANGE client132dsocket1418 220 30725-30747
START_RANGE client132esocket1419 220 30747-30769
START_RANGE client132fsocket1420 210 30769-30790
START_RANGE client132gsocket1421 220 30790-30812
START_RANGE client132hsocket1422 220 30812-30834
START_RANGE client132isocket1423 210 30834-30855
START_RANGE client132jsocket1424 220 30855-30877
START_RANGE client132ksocket1425 220 30877-30899
START_RANGE client132lsocket1426 210 30899-30920
START_RANGE client132msocket1427 220 30920-30942
START_RANGE client132nsocket1428 220 30942-30964
START_RANGE client132asocket1429 210 30964-30985
START_RANGE client132bsocket1430 220 30985-31007
START_RANGE client132csocket1431 220 31007-31029
START_RANGE client132dsocket1432 220 31029-31051
START_RANGE client132esocket1433 210 31051-31072
START_RANGE client132fsocket1434 220 31072-31094
START_RANGE client132gsocket1435 220 31094-31116
START_RANGE client132hsocket1436 210 31116-31137
START_RANGE client132isocket1437 220 31137-31159
START_RANGE client132jsocket1438 220 31159-31181
START_RANGE client132ksocket1439 210 31181-31202
START_RANGE client132lsocket1440 220 31202-31224
START_RANGE client132msocket1441 220 31224-31246
START_RANGE client132nsocket1442 210 31246-31267
START_RANGE client132asocket1443 220 31267-31289
START_RANGE client132bsocket1444 220 31289-31311
START_RANGE client132csocket1445 210 31311-31332
START_RANGE client132dsocket1446 220 31332-31354
START_RANGE client132esocket1447 220 31354-31376
START_RANGE client132fsocket1448 210 31376-31397
START_RANGE client132gsocket1449 220 31397-31419
START_RANGE client132hsocket1450 220 31419-31441
START_RANGE client132isocket1451 220 31441-31463
START_RANGE client132jsocket1452 210 31463-31484
START_RANGE client132ksocket1453 220 31484-31506
START_RANGE client132lsocket1454 220 31506-31528
START_RANGE client132msocket1455 210 31528-31549
START_RANGE client132nsocket1456 220 31549-31571

START_RANGE client141asocket1457 220 31571-31593
START_RANGE client141bsocket1458 210 31593-31614
START_RANGE client141csocket1459 220 31614-31636
START_RANGE client141dsocket1460 220 31636-31658
START_RANGE client141esocket1461 210 31658-31679
START_RANGE client141fsocket1462 220 31679-31701
START_RANGE client141gsocket1463 220 31701-31723
START_RANGE client141hsocket1464 210 31723-31744
START_RANGE client141isocket1465 220 31744-31766
START_RANGE client141jsocket1466 220 31766-31788
START_RANGE client141ksocket1467 210 31788-31809
START_RANGE client141lsocket1468 220 31809-31831
START_RANGE client141msocket1469 220 31831-31853
START_RANGE client141nsocket1470 210 31853-31874
START_RANGE client141asocket1471 220 31874-31896
START_RANGE client141bsocket1472 220 31896-31918
START_RANGE client141csocket1473 220 31918-31940
START_RANGE client141dsocket1474 210 31940-31961
START_RANGE client141esocket1475 220 31961-31983
START_RANGE client141fsocket1476 220 31983-32005
START_RANGE client141gsocket1477 210 32005-32026
START_RANGE client141hsocket1478 220 32026-32048
START_RANGE client141isocket1479 220 32048-32070
START_RANGE client141jsocket1480 210 32070-32091
START_RANGE client141ksocket1481 220 32091-32113
START_RANGE client141lsocket1482 220 32113-32135
START_RANGE client141msocket1483 210 32135-32156
START_RANGE client141nsocket1484 220 32156-32178
START_RANGE client141asocket1485 220 32178-32200
START_RANGE client141bsocket1486 210 32200-32221
START_RANGE client141csocket1487 220 32221-32243
START_RANGE client141dsocket1488 220 32243-32265
START_RANGE client141esocket1489 210 32265-32286
START_RANGE client141fsocket1490 220 32286-32308
START_RANGE client141gsocket1491 220 32308-32330
START_RANGE client141hsocket1492 220 32330-32352
START_RANGE client141isocket1493 210 32352-32373
```

```
START_RANGE client141jsocket1494 220 32373-32395
START_RANGE client141ksocket1495 220 32395-32417
START_RANGE client141lsocket1496 210 32417-32438
START_RANGE client141msocket1497 220 32438-32460
START_RANGE client141nsocket1498 220 32460-32482
START_RANGE client141asocket1499 210 32482-32503
START_RANGE client141bsocket1500 220 32503-32525
START_RANGE client141csocket1501 220 32525-32547
START_RANGE client141dsocket1502 210 32547-32568
START_RANGE client141esocket1503 220 32568-32590
START_RANGE client141fsocket1504 220 32590-32612
START_RANGE client141gsocket1505 210 32612-32633
START_RANGE client141hsocket1506 220 32633-32655
START_RANGE client141isocket1507 220 32655-32677
START_RANGE client141jsocket1508 210 32677-32698
START_RANGE client141ksocket1509 220 32698-32720
START_RANGE client141lsocket1510 220 32720-32742
START_RANGE client141msocket1511 220 32742-32764
START_RANGE client141nsocket1512 210 32764-32785
```

```
START_RANGE client142asocket1513 220 32785-32807
START_RANGE client142bsocket1514 220 32807-32829
START_RANGE client142csocket1515 210 32829-32850
START_RANGE client142dsocket1516 220 32850-32872
START_RANGE client142esocket1517 220 32872-32894
START_RANGE client142fsocket1518 210 32894-32915
START_RANGE client142gsocket1519 220 32915-32937
START_RANGE client142hsocket1520 220 32937-32959
START_RANGE client142isocket1521 210 32959-32980
START_RANGE client142jsocket1522 220 32980-33002
START_RANGE client142ksocket1523 220 33002-33024
START_RANGE client142lsocket1524 210 33024-33045
START_RANGE client142msocket1525 220 33045-33067
START_RANGE client142nsocket1526 220 33067-33089
START_RANGE client142asocket1527 210 33089-33110
START_RANGE client142bsocket1528 220 33110-33132
START_RANGE client142csocket1529 220 33132-33154
START_RANGE client142dsocket1530 220 33154-33176
START_RANGE client142esocket1531 210 33176-33197
START_RANGE client142fsocket1532 220 33197-33219
START_RANGE client142gsocket1533 220 33219-33241
START_RANGE client142hsocket1534 210 33241-33262
START_RANGE client142isocket1535 220 33262-33284
START_RANGE client142jsocket1536 220 33284-33306
START_RANGE client142ksocket1537 210 33306-33327
START_RANGE client142lsocket1538 220 33327-33349
START_RANGE client142msocket1539 220 33349-33371
START_RANGE client142nsocket1540 210 33371-33392
START_RANGE client142asocket1541 220 33392-33414
START_RANGE client142bsocket1542 220 33414-33436
START_RANGE client142csocket1543 210 33436-33457
START_RANGE client142dsocket1544 220 33457-33479
START_RANGE client142esocket1545 220 33479-33501
START_RANGE client142fsocket1546 210 33501-33522
START_RANGE client142gsocket1547 220 33522-33544
START_RANGE client142hsocket1548 220 33544-33566
START_RANGE client142isocket1549 220 33566-33588
START_RANGE client142jsocket1550 210 33588-33609
START_RANGE client142ksocket1551 220 33609-33631
START_RANGE client142lsocket1552 220 33631-33653
START_RANGE client142msocket1553 210 33653-33674
START_RANGE client142nsocket1554 220 33674-33696
START_RANGE client142asocket1555 220 33696-33718
START_RANGE client142bsocket1556 210 33718-33739
START_RANGE client142csocket1557 220 33739-33761
START_RANGE client142dsocket1558 220 33761-33783
START_RANGE client142esocket1559 210 33783-33804
START_RANGE client142fsocket1560 220 33804-33826
START_RANGE client142gsocket1561 220 33826-33848
START_RANGE client142hsocket1562 210 33848-33869
START_RANGE client142isocket1563 220 33869-33891
START_RANGE client142jsocket1564 220 33891-33913
START_RANGE client142ksocket1565 210 33913-33934
START_RANGE client142lsocket1566 220 33934-33956
START_RANGE client142msocket1567 220 33956-33978
START_RANGE client142nsocket1568 220 33978-34000
```

```
#elif MASTER_NUM8
#endif
/*-----*/
#define TES_FLAG_TRACE 0x00000010
#define TES_FLAG_KEYSTROKE_TIME 0x00000200
#define TES_FLAG_LOCAL_LOG 0x00000400
#define TES_FLAG_LOCAL_TRACE 0x00000800
#define TES_FLAG_LOCAL_IPRINT 0x00004000
#if 0
/* SETFLAG ALL TES_FLAG_TRACE */
SETFLAG ALL TES_FLAG_LOCAL_TRACE
SETFLAG ALL TES_FLAG_LOCAL_IPRINT
#endif
#if 0
SETFLAG client31 telnet 1 TES_FLAG_KEYSTROKE_TIME
#endif
```

D.2 user_master.C

```
/* user_master.C Audit: 05/30/96/ */
static char *rcsid="$Id: user_master.C,v 1.1 1999/02/22 06:31:05:channui Exp $";

#include <iostream.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#define _H_CUR01
#include <cur00.h>
#undef _H_CUR01
extern "C" {
#include "data/cur01.h"
int wrefresh (WINDOW *);
int wclrtoeol(WINDOW *);
int setupterm(char*,FILE*,int*);
int nodelay(int);
int keypad(int);
int wgetch(WINDOW *);
}
#include "data/rte.h"
#include "data/Stats.h"
#include "data/misc.h"
#include "user_tpc.h"

struct header_s {
int slave;
int num;
int type;
int num_timestamps;
int user_data_length;
int data_type;
};

char *get_variable(char *name);
int get_variable(char *name, int *number);
int send_global_data(void);
int make_ratios (double *buffer);
extern int ramp_up_complete;
extern int interval_start_time, interval_stop_time;
extern "C" int strcasecmp(char *s1, char *s2);
extern "C" int strcmpsecmp(char *s1, char *s2, int n);

struct UserSpawnData {
int Warehouse;
int District;
};

/* user_master.C */
int user_statistics_print(void);
// int user_spawn(int *length, char *buffer);
int user_spawn(int min, int max, int number, int *length, char *buffer);
int user_finished(int length, char *buffer);

extern SlaveStatus slave_status[MAX_SLAVES];

extern Stats status[MAX_TRAN_TYPE][MAX_TIMES];
extern WINDOW *statistics_win;
extern UserGlobal *shmglobal;

/* Transaction mix parameters */
double ratio_desired[6], ratio_min[6], ratio_max[6], ratio_range[6];
char *ratio_names[] = { "RTE", "NEWORDER", "PAYMENT", "ORDSTAT", "DELIVERY",
"STOCKLEV", NULL };
char *Status_Names[] = { "Menu", "Keying", "Response", "Think";

char *transaction_names[] = { "RTE", "New Order", "Payment", "OrderStat",
"Delivery", "Stock Level", NULL };

static int current_status = 2, status_needs_refresh = 1;

int user_statistics_print(void) {
int i;
static int count = 0;
double ratios[6];
if (status_needs_refresh) {
count = 0;
status_needs_refresh = 0;
wmove (statistics_win, 0, 0);
wprintw (statistics_win, "%11s %8s %8s %8s %8s %6s %6s %6s",
Status_Names[current_status], "90%", "Avg", "Min", "Max",
"Samples", "Ratio", "Mix", "Think");
}
make_ratios(ratios);

for (i = 1; i <= 5; i++) {
/* The reason we do this is because calculating the percentiles
is expensive */
```

```
if (count % 10 == 0) {
wmove (statistics_win, i, 0);
wprintw (statistics_win, "%11s %8.2f",
transaction_names[i], status[i][current_status].ninety()/1000.0);
count = 0;
}
wmove (statistics_win, i, 21);
wprintw (statistics_win, "%8.2f %8.2f %8.2f %8d %6.2f %6.2f %6.2f",
status[i][current_status].average()/1000.0,
status[i][current_status].min()/1000.0,
status[i][current_status].max()/1000.0,
status[i][current_status].samples(),
ratios[i].shmglobal->chances[i],
status[i][3].average()/1000.0);
}
wmove (statistics_win, 7, 0);

extern int runtime_counts[MAX_TRAN_TYPE];
extern int begin_time, ramp_up, run_time;
int start = interval_start_time;
int stop = interval_stop_time;
double interval = ((double)(stop-start) / (1000*60));
double samples = status[1][2].samples();
if (interval <= 0 || samples <= 0) {
wprintw (statistics_win, "TPM-C: %7s / ", "-----");
} else {
wprintw (statistics_win, "TPM-C: %7.2f / ", samples/interval);
}
samples = runtime_counts[1];
if (samples > 0) {
start = begin_time + ((ramp_up >= 0) ? ramp_up : 0);
if (run_time > 0 && stop > begin_time + ramp_up + run_time) {
stop = begin_time + ramp_up + run_time;
}
interval = ((double)(stop - start) / (1000.0 * 60.0));
wprintw (statistics_win, "%7.2f", samples/interval);
} else {
wprintw (statistics_win, "-----");
}

count++;
return RTE_OK;
}

extern int login_begin;
int login_max_load;

#ifdef WHSEARRAYDBG
int outofboundwarn;
#endif
extern int min_warehouse;
extern int max_warehouse;

const int MAX_WAREHOUSES=100000;
/* All of this 10 stuff is district size. Should be a constant.
Maybe fix that later */
int num_warehouses = -1;
int warehouses[MAX_WAREHOUSES*10];
int user_spawn(int min, int max, int number, int *length, char *buffer) {
//int user_spawn(int number, int *length, char *buffer) {
int i, min_index;
int adj_wh = num_warehouses; // adjusted warehouse number
UserSpawnData *ptr = (UserSpawnData *)buffer;
*length = sizeof(*ptr);

// min_index = 0;
// for (i = 1; i < (num_warehouses)*10 && i < MAX_WAREHOUSES*10; i++) {
//
// if both min and max are zero, running START, otherwise running
// START_RANGE. Must also determine what the ending warehouse number
// will be for said range
//
if (min == 0 && max == 0) {
min++;
min_index = 0;
} else {
adj_wh = max; // inclusive range of wh-s
min = min * 10;
min_index = min;
}
for (i = min; i < (adj_wh)*10 && i < ((MAX_WAREHOUSES+min_warehouse)*10); i++) {
if (warehouses[i - (min_warehouse*10)] < warehouses[min_index - (min_warehouse*10)]) {
min_index = i;
}
}
ptr->Warehouse = min_index / 10 + 1;
ptr->District = min_index % 10 + 1;
#ifdef WHSEARRAYDBG
if ((min_index - (min_warehouse*10) < 0) || (min_index - (min_warehouse*10) >=
(MAX_WAREHOUSES*10))) {
if (outofboundwarn) {
iprint (IPRINT_INFO, "(spawn) Out of range warehouse number %d, (%d-%d (start) = %d
(rel.num))n",
min_index, min_index, min_warehouse, min_index - (min_warehouse*10));
outofboundwarn=0;
}
}
}
}
```

```

#endif
    warehouses[min_index - (min_warehouse*10)]++;
    /* iprint (IPRINT_INFO, "Driver for Warehouse %d, District %d started. warehouses[%d]++ = %d\n",
    ptr->Warehouse, ptr->District, min_index, warehouses[min_index - (min_warehouse*10)]); */
    return RTE_OK;
}

int user_finished(int length, char *buffer) {
    UserSpawnData *ptr = (UserSpawnData *)buffer;
    int temp = (ptr->Warehouse-1)*10+ptr->District-1;

#ifdef WHSEARRAYDBG
    if ((temp - min_warehouse*10 < 0) || (temp - min_warehouse*10 >= MAX_WAREHOUSES*10)) {
        if (outofboundwarn) {
            iprint (IPRINT_INFO, "(finish) Out of range warehouse number %d, (%d-%d (start) = %d
            (rel. num)n",
                    min_index, min_index, min_warehouse, min_index - (min_warehouse*10));
            outofboundwarn=0;
        }
    }
#endif

    warehouses[temp - (min_warehouse*10)]--;
    /* iprint (IPRINT_INFO, "Driver for Warehouse %d, District %d died. warehouses[%d]-- = %d\n",
    ptr->Warehouse, ptr->District, temp, warehouses[temp - (min_warehouse*10)]); */
    return RTE_OK;
}

double limit(double min, double max, double val) {
    if (val < min)
        return min;
    if (val > max)
        return max;
    return val;
}

int make_ratios (double *buffer) {
    int neword = status[NEWORDER][0].samples();
    int payment = status[PAYMENT][0].samples();
    int ordstat = status[ORDSTAT][0].samples();
    int delivery = status[DELIVERY][0].samples();
    int stocklev = status[STOCKLEV][0].samples();
    int total = neword + payment + ordstat + delivery + stocklev;
    int i;

    if (total == 0) {
        buffer[NEWORDER] = 100.0;
        for (i = 2; i < 6; i++) {
            buffer[i] = ratio_desired[i];
            buffer[NEWORDER] -= buffer[i];
        }
        return 0;
    }

    buffer[PAYMENT] = (double)payment / (double)total * 100.0;
    buffer[ORDSTAT] = (double)ordstat / (double)total * 100.0;
    buffer[DELIVERY] = (double)delivery / (double)total * 100.0;
    buffer[STOCKLEV] = (double)stocklev / (double)total * 100.0;
    buffer[NEWORDER] = 100.0 - buffer[PAYMENT] - buffer[ORDSTAT] -
        buffer[DELIVERY] - buffer[STOCKLEV];

    return total;
}

int user_global_update(int *length, char *buffer) {
    UserGlobal *shmglobal = (UserGlobal *)buffer;
    static double last[6];
    static last_test_state = 0;
    static int users_last=-1;
    double ratios[6];
    double current[6];
    int i, different = 0;
    int desired = 0;
    int host_busy, all_zero;

    *length = sizeof(*shmglobal);

    make_ratios(ratios);

    /* Calculate ratios we want for next time */
    /* Note: we just keep on with the desired values until ramp-up is complete
    this at least starts us out without any humps or spikes in the
    graph */
    if (ramp_up_complete) {
        current[NEWORDER] = 100.0;
        for (i = 2; i < 6; i++) {
            if (ratio_desired[i] > ratios[i]) {
                current[i] = ratio_max[i];
            } else {
                current[i] = 2*ratio_desired[i] - ratios[i];
                if (current[i] < ratio_min[i])
                    current[i] = ratio_min[i];
            }
        }
        current[NEWORDER] -= current[i];
    } else {
        for (i = 1; i < 6; i++) {
            current[i] = ratio_desired[i];
        }
    }

    if (shmglobal->host_busy && all_zero) {
        host_busy = 1;
    }

    if (host_busy != shmglobal->host_busy) {
        shmglobal->host_busy = host_busy;
        different = 1;
    }

    for (i = 2; i < 6; i++) {
        if (current[i] != last[i])
            different = 1;
    }

    if (last_test_state != shmglobal->test_state) {
        different = 1;
        last_test_state = shmglobal->test_state;
    }

    // Don't send if it's the same as last time
    if ( !different && shmglobal->total_users == users_last ) {
        return RTE_ERROR;
    }

    users_last = shmglobal->total_users;
    for (i = 1; i < 6; i++) {
        shmglobal->chances[i] = last[i] = current[i];
    }

    return RTE_OK;
}

int user_isbusy() {
    return shmglobal->host_busy;
}

int parse_array(char *string, int max, int *buffer) {
    int i, rc;
    char *ptr;
    char *temp = strdup(string);
    ptr = strtok(temp, ",");
    for (i = 0; ptr && i < max; i++) {
        rc = sscanf(ptr, "%d", &buffer[i]);
        if (rc < 1) {
            free(temp);
            return i;
        }
        ptr = strtok(NULL, ",");
    }
    free(temp);
    return i;
}

int parse_array(char *string, int max, double *buffer) {
    int i, rc;
    char *ptr;
    char *temp = strdup(string);
    ptr = strtok(temp, ",");
    for (i = 0; ptr && i < max; i++) {
        rc = sscanf(ptr, "%lf", &buffer[i]);
        if (rc < 1) {
            free(temp);
            return i;
        }
        ptr = strtok(NULL, ",");
    }
    free(temp);
    return i;
}

```



```

/* A: 255 for C_LAST, 1023 for C_ID, 8191 for OL_I_ID */
/* x: 0 for C_LAST, 1 for C_ID and OL_I_ID */
/* y: 999 for C_LAST, 3000 for C_ID, 10000 for OL_I_ID */
*/
long
NURand(int A, int x, int y, long cval)
{
    return (((long)uniform((long) 0, (long) A) | (long)uniform((long) x, (long) y)) + cval) % (y - x + 1) + x;
}

/*-----*/
/* getname */
/*-----*/
/* generates a random number from 0 to 999 inclusive */
/* a random name is generated by associating a random */
/* string with each digit of the generated number */
/* three strings are concatenated to generate lastname */
/*-----*/
char *
getname()
{
    char *last_name_parts[] =
    {
        "BAR",
        "OUGHT",
        "ABLE",
        "PRI",
        "PRES",
        "ESE",
        "ANTI",
        "CALLY",
        "ATION",
        "EING"
    };
    static char lastname[128];
    int random_num;

#if 1
    random_num = NURand(255, 0, 999, shmglbal->lastc);
#else
    random_num = NURand(255, 0, 999, LASTC);
#endif
    strcpy(lastname, last_name_parts[random_num / 100]);
    random_num %= 100;
    strcat(lastname, last_name_parts[random_num / 10]);
    random_num %= 10;
    strcat(lastname, last_name_parts[random_num]);
    return (lastname);
}

typedef struct gen_tran_s {
    int invalid;
    void *data;
    long len;
    long keywait;
    long type;
    char *menu;
    char *request;
} gen_tran_t;

int generic_transaction( gen_tran_t *data ) {
    char buffer[2048];
    static int consecutive_errs = 0;
    int rc;
    set_typing_delay(0);
    fprintf(IPRINT_TRACE, "> generic_transaction sleep (%d) type(%d) *data (%d)\n",
data->type,data->menu,data);
#ifdef NOSLEEP
    if (shmglbal->test_state == 0)
        transaction_sleep_do();
#endif

#ifdef EXPECT_TIMEOUT
    int timeout = EXPECT_TIMEOUT;
#else
    int timeout = 0;
#endif

    // Start the transaction (MENU)
    fprintf(IPRINT_TRACE, "> generic_transaction start (%d)\n", data->type);
    transaction_start(data->type, data->len, data->data);

    fprintf(IPRINT_TRACE, "> transmit data->menu: (%s)\n", data->menu);
    transmit(data->menu);
    echo_trace ("Waiting for Menu");
    if (expect(TRIGGER, timeout) == ERROR) {
        fprintf(IPRINT_ERROR, "Slave %d: Failed to receive %screen\n",
shmentry->num, data->menu);
        return (ERROR);
    }
#ifdef NOSLEEP
    usleep(shmglbal->emulex_menu[data->type]*1000000.0+0.9);
#endif

    // Send our request (KEYING)
    transaction_mark(WHERE_NOW);
    echo_trace ("Keying");
#ifdef NOSLEEP
    usleep(data->keywait*1000000+KEYWAIT_FUDGE); // Keying delay
#endif
}

// Wait for response (RESPONSE)
transaction_mark(WHERE_NOW);

fprintf(IPRINT_TRACE, "> transmit data->request\n");
transmit(data->request);

echo_trace ("Wait for Response");
if (expect(TRIGGER, timeout) == ERROR) {
    fprintf(IPRINT_ERROR, "Slave %d: Failed to receive %sresponse\n",
shmentry->num, data->menu);
    return (ERROR);
}
#ifdef NOSLEEP
    usleep(shmglbal->emulex_response[data->type]*1000000.0+0.9);
#endif

// Look for errors and set our think time (THINK)
transaction_mark(WHERE_NOW);
if (expect_after_match ("ERROR: ") {
    data->invalid = 1;
    fprintf(IPRINT_ERROR, "Slave %d: %s found %s\n",
shmentry->num, data->menu, "ERROR:");
    // Very dangerous, keep going rather than exiting...
    return RTE_ERROR;
    // Check for consecutive errors and if there are more than
    // 4 of them exit - allow for transient errors to make
    // tuning and testing easier -oz
    // In either case the transaction is marked as invalid and
    // will be reported as an error by the analyze program.
    // if (consecutive_errs++ > 4)
    //     return RTE_ERROR;
} else {
    consecutive_errs = 0;
}
echo_trace ("Thinking");
transaction_sleep_set(neg_exp_4(shmglbal->think[data->type])*1000.0);
fprintf(IPRINT_TRACE, "< generic_transaction finish\n");
return (RTE_OK);
}

/*-----*/
/* Delivery Transaction */
/*-----*/
int
Delivery()
{
    static struct delivery_struct delivery, delivery_new;
    int rc;
    char *ptr;
    char buffer[256];
    gen_tran_t tran;

    tran.invalid = 0;
    tran.data = &delivery;
    tran.len = sizeof(delivery);
    tran.keywait = 2;
    tran.type = DELIVERY;
    tran.menu = "4";
    tran.request = buffer;

    // Set up all data for new transactions
    delivery_new.carrier = uniform(1, 10); // carrier # 1 to 10

    // Now create the actual request
    ptr = buffer;
    ptr += sprintf(ptr, "%d\n", delivery_new.carrier);

    // Go do the transaction
    rc = generic_transaction(&tran);
    delivery = delivery_new;
    delivery.invalid = tran.invalid;

    return (rc);
}

/*-----*/
/* New Order Transaction */
/*-----*/
int
NewOrder() {
    static struct neword_struct neword, neword_new;
    int i, rc, whses, low_whse=1;
    char buffer[2048];
    char *ptr;
    const char *ptr2;
    gen_tran_t tran;

    tran.invalid = 0;
    tran.data = &neword;
    tran.len = sizeof(neword);
    tran.keywait = 18;
    tran.type = NEWORDER;
    tran.menu = "1";
    tran.request = buffer;

    neword_new.rollback=0;

    /* SECTION TO DETERMINE ROLLBACK TRANSACTION FOR 1% OF NEW ORDERS*/
}

```

```

neword_new.did = uniform(1, 10); // district number
neword_new.cid = NURand(1023, 1, 3000,CUSTC); // customer # 1 to 3000
neword_new.nloop = uniform(5, 15); // number of items to order (5-15)
neword_new.olremote=0; // find total number of remote order-lines

whses = shmglobal->max_warehouses;

for (i = 0; i < neword_new.nloop; i++) {
    // Warehouse Number
    neword_new.item[i].olswid = WHSEID;
    if (whses > 1 && (uniform(0.0, 100.0) < 1.0)) {
        /* for 1% of items (if *uniform()=0)*/
        /* Generate a uniform whose number that's different from WHSEID */
        neword_new.item[i].olswid =
            (long) uniform((long) low_whse, (long) whses-1);
        if (neword_new.item[i].olswid >= WHSEID)
            neword_new.item[i].olswid++;
        neword_new.olremote++; // find total number of remote order-lines
    }
    // Item number 1-100000
    neword_new.item[i].oliid = NURand(8191, 1, 100000,ITEMC);
    // Quantity 1-10
    neword_new.item[i].olquantity = uniform(1, 10);
} // end of for n_loop */
// We occasionally force a transaction to have invalid data to force a
// rollback
if (uniform(1, 5000) <= 50)
    neword_new.item[neword_new.nloop-1].oliid = 999999;

neword_new.oremove = (neword_new.olremote > 0);

// Now create the actual request
ptr = buffer;
ptr += sprintf(ptr, "%d\t%d", neword_new.did, neword_new.cid);
for (i = 0; i < neword_new.nloop; i++) {
    ptr += sprintf(ptr, "%d\t%d\t%d",
        neword_new.item[i].olswid,
        neword_new.item[i].oliid,
        neword_new.item[i].olquantity);
}
ptr += sprintf(ptr, "\n");

// Go do the transaction
rc = generic_transaction(&tran);
neword = neword_new;
neword.invalid = tran.invalid;

// Check for a rollback
if (expect_after_match (ROLLBACK_MESSAGE)) {
    neword.rollback=1;
    echo_trace ("Found rollback!\n");
}

// Grab the orderID from the
if (! (ptr2 = expect_after_match ("033[6:15H]"))) {
    echo_trace ("Didn't find order-id for neworder");
    iprint (IPRINT_ERROR, "Neworder didn't have Order-ID");
    neword.oid = -1;
} else {
    neword.oid = atoi(ptr2+8);
}

// This is really not useful since we aren't going to be sending individual
// keystrokes anymore
if (shmentry->flags & TES_FLAG_KEYSTROKE_TIME) {
    log_data(RTE_ITEM_KEYSTROKE_TIMES,
keystroke_length*sizeof(int),keystroke_times);
}

return (rc);
}

/*****
*** Order Status Transaction ***
*****/
int OrderStatus() {
    static struct ordstat_struct ordstat, ordstat_new;
    char buffer[2048];
    int rc;
    char *ptr;
    gen_tran_t tran;

    tran.invalid = 0;
    tran.data = &ordstat;
    tran.len = sizeof(ordstat);
    tran.keywait = 2;
    tran.type = ORDSTAT;
    tran.menu = "3";
    tran.request = buffer;

    // Set up all data for new transactions
    ordstat_new.did = uniform(1, 10); // district number 1 to 10 */
    if (uniform(1, 100) <= 60) /* for 60% of transactions */
        char *tmp = getname();
        strcpy(ordstat_new.clast, tmp); // by customer last name */
        if (ordstat_new.clast[0] < 'A' || ordstat_new.clast[0] > 'Z') {
            iprint (IPRINT_ERROR,
                "ASSERTION:OrderStatus getname() returns invalid name! '%s'\n",

```

```

ordstat_new.clast);
return RTE_ERROR;
}
ordstat_new.byname = 1;
ordstat_new.cid = 0;
} else {
    ordstat_new.cid = NURand(1023, 1, 3000,CUSTC); // cust. # 1 to 3000*/
    ordstat_new.byname = 0;
    ordstat_new.clast[0] = (char) NULL;
}

// Now create the actual request
ptr = buffer;
ptr += sprintf(ptr, "%d\t", ordstat_new.did);
if (ordstat_new.byname) {
    ptr += sprintf(ptr, "%s\n", ordstat_new.clast);
} else {
    ptr += sprintf(ptr, "%d\n", ordstat_new.cid);
}

// Go do the transaction
rc = generic_transaction(&tran);
ordstat = ordstat_new;
ordstat.invalid = tran.invalid;

return (rc);
}

/*****
*** Payment Transaction ***
*****/
int
Payment()
{
    static struct payment_struct payment, payment_new;
    int dollars, cents,rc, whses, low_whse = 1;
    char buffer[2048];
    char *ptr;
    gen_tran_t tran;

    tran.invalid = 0;
    tran.data = &payment;
    tran.len = sizeof(payment);
    tran.keywait = 3;
    tran.type = PAYMENT;
    tran.menu = "2";
    tran.request = buffer;

    payment_new.did = uniform(1, 10); // district number 1 to 10 */
    if (uniform(1, 100) <= 60) /* for 60% of transactions */
        strncpy(payment_new.clast, getname(), 17); // by customer last name
        if (payment_new.clast[0] < 'A' || payment_new.clast[0] > 'Z') {
            iprint (IPRINT_ERROR,
                "ASSERTION:payment_new getname() returns invalid name! '%s'\n",
                payment_new.clast);
            return RTE_ERROR;
        }
        payment_new.byname = 1;
        payment_new.cid = 0;
    } else {
        payment_new.cid = NURand(1023, 1, 3000,CUSTC); // cust. # 1 to 3000*/
        payment_new.byname = 0;
        payment_new.clast[0] = (char) NULL;
    }

    whses = shmglobal->max_warehouses;

    if (whses < 2 || uniform(1, 100) <= 85) /* for 85 % of transactions */
        payment_new.cwid = WHSEID;
        payment_new.cdoid = payment_new.did;
        payment_new.remote = 0;
    } else {
        // for 15 % of transactions */
        payment_new.cwid = (long) uniform((long)low_whse, (long) whses-1);
        if (payment_new.cwid >= WHSEID)
            payment_new.cwid++;

        payment_new.remote = 1;
        payment_new.cdoid = uniform(1, 10); // district 1 to 10*/
    }

    dollars = uniform(1, 5000); /* dollar amt = 1 to 5000 */
    if (dollars == 5000)
        cents = 0;
    else
        cents = uniform(0, 99);

    payment_new.amount = ((double) dollars) + ((double) cents) / 100.0;

    // Now create the actual request
    ptr = buffer;
    ptr += sprintf(ptr, "%d\t", payment_new.did);
    if (payment_new.byname) {
        ptr += sprintf(ptr, "%s\t", payment_new.clast);
    } else {
        ptr += sprintf(ptr, "%d\t", payment_new.cid);
    }
    ptr += sprintf(ptr, "%d\t%d\t", payment_new.cwid, payment_new.cdoid);
    ptr += sprintf(ptr, "%d.%02.2d\n", dollars, cents);

```



```

// Go do the transaction

rc = generic_transaction(&tran);
payment = payment_new;
payment.invalid = tran.invalid;

return (rc);
}

/*****
*** Stock Level Transaction *****/
/*****
int
StockLevel()
{
static struct stocklev_struct stocklevel, stocklevel_new;
char buffer[2048];
int rc;
char *ptr;
gen_tran_t tran;

tran.invalid = 0;
tran.data = &stocklevel;
tran.len = sizeof(stocklevel);
tran.keywait = 2;
tran.type = STOCKLEV;
tran.menu = "5";
tran.request = buffer;

stocklevel_new.invalid = 0;
stocklevel_new.threshold = uniform(10, 20); /* uniform no. between 10 and * 20 */

// Now create the actual request
ptr = buffer;
ptr += sprintf(ptr, "%d\n", stocklevel_new.threshold);

// Go do the transaction
rc = generic_transaction(&tran);
stocklevel = stocklevel_new;
stocklevel.invalid = tran.invalid;

return (rc);
}

/*****
*** MAIN() *****/
/*****
int
user_transaction()
{
char logout[32];
double ntask;
int resp;
static int task = 0;

if(shmentry->flags & TES_FLAG_KEYSTROKE_TIME){
int rc;
/* Wait for specified period of time */
sleep (shmglobal->keystroke_sleep);
/* Quit after one transaction */
shm->lock(shmentry->pid);
shmentry->flags |= TES_FLAG_DIE;
shm->unlock(shmentry->pid);
rc = NewOrder();
iprint (IPRINT_INFO, "Slave %d: Keystroke timing setting dieflag\n", shmentry->num);
return rc;
}

#if 1
switch (shmglobal->test_state) {
case 0: // Normal
break;
case 1: // pause
sleep (1);
return RTE_OK;
case 2: // warmup
switch(task++) {
case 0: return Delivery();
case 1: return OrderStatus();
case 2: return Payment();
case 3: return StockLevel();
case 4: task = 0; return NewOrder();
}
}
/*****
*** CHOOSE ONE OF THE TRANSACTIONS *****/
/*****
ntask = (double) uniform(0.0, 100.0);
if (ntask <= shmglobal->chances[DELIVERY]) {
return Delivery();
}
ntask -= shmglobal->chances[DELIVERY];
if (ntask <= shmglobal->chances[ORDSTAT]) {
return OrderStatus();
}
ntask -= shmglobal->chances[ORDSTAT];
if (ntask <= shmglobal->chances[PAYMENT]) {
return Payment();
}
}
ntask -= shmglobal->chances[PAYMENT];
if (ntask <= shmglobal->chances[STOCKLEV]) {
return StockLevel();
}
}
return NewOrder();
#else
// this code should be shared between all of the users on a slave
// int the best case it should be shared between all of the slaves,
// but that would be too costly.
// for now it is done on a per user basis. If this thing is ever
// modified to be threaded then it will probably go to the per-process
// basis. Although with shared memory, it would be possible to go to
// per-slave. Actually, before this code is put into use it must be
// fixed up to share across processes. Right now it will take, on average,
// 22 minutes for one user to just key in the 100 entries.

// use a card deck with no replacement to fulfill the requirements
{
int deck[100], count=-1, i, size=1, tmp;
// lock deck
if (count < 0) {
// deck is empty fill it up
count = 0;
for (i = 0; i < 43 * size; i++) {
deck[count++] = Payment;
}
for (i = 0; i < 4 * size; i++) {
deck[count++] = StockLevel;
}
for (i = 0; i < 4 * size; i++) {
deck[count++] = OrderStatus;
}
for (i = 0; i < 4 * size; i++) {
deck[count++] = Delivery;
}
for (; count < 100 * size; i++) {
deck[count++] = NewOrder;
}
// randomize the deck
for (i = 0; i < 100 * size; i++) {
int tmp;
int pick = uniform(i+1, 100);
tmp = deck[i];
deck[i] = deck[pick];
deck[pick] = tmp;
}
}
tmp = deck[count--];
// unlock deck
switch(tmp) {
case Delivery: return Delivery();
case OrderStatus: return OrderStatus();
case Payment: return Payment();
case StockLevel: return StockLevel();
case NewOrder: return NewOrder();
}
}
#endif

#if 0
if (resp != RTE_OK) { /* logoff if response is not correct */
strcpy(logout, "9\n"); /* menu option 9 */
transmit(logout);
resp = expect("tpcex_cstux_inf");
return (ERROR);
} else
return (RTE_OK);
#endif

/* end of main */
}

int user_parameter_change(void) {
#if 0
int i;
iprint(IPRINT_TRACE, "Slave %d: total_users = %d\n", shmentry->num);
iprint(IPRINT_TRACE, "Slave %d: chances = ", shmentry->num);
for (i = 0; i < MAX_TRAN_TYPE; i++)
iprint(IPRINT_TRACE, "%6.2f ", shmglobal->chances[i]);
iprint(IPRINT_TRACE, "\nSlave %d: think = ", shmentry->num);
for (i = 0; i < MAX_TRAN_TYPE; i++)
iprint(IPRINT_TRACE, "%6.2f ", shmglobal->think[i]);
iprint(IPRINT_TRACE, "\n");
#endif
return RTE_OK;
}

int user_login(char *user, char *password, void *data) {
UserLocal *localdata = (UserLocal *)data;
int rc;
int timeout_value = shmglobal->login_timeout;
char buffer[32];
set_typing_delay(0);

rc = expect (TRIGGER, timeout_value);
if (rc == RTE_ERROR) {
iprint (IPRINT_ERROR, "Slave %d: didn't find Warehouse prompt\n", shmentry->num);
}
sprintf(buffer, "%d(%d)\n", localdata->Warehouse, localdata->District);
transmit(buffer);
}

```

```

iprint (IPRINT_TRACE, "Slave %d: Warehouse=%d, District=%d, pid=%d\n", shmentry->num,
localdata->Warehouse, localdata->District, getpid());

rc = expect (TRIGGER, timeout_value);
if (rc != RTE_OK) {
    iprint (IPRINT_ERROR, "Slave %d: Failed loggingin\n", shmentry->num);
    return RTE_ERROR;
}
return RTE_OK;
}

int user_init () {
    extern int expect_save_active;
    WHSEID = shmlocal->Warehouse;

    status->max_transmit = shmglobal->keystroke_packet_size;
    expect_save_active = 1;
    return RTE_OK;
}

int user_logout () {
    transmit("9");
    iprint (IPRINT_TRACE, "Slave %d: Warehouse=%d, District=%d\n", shmentry->num,
shmlocal->Warehouse, shmlocal->District);
    return RTE_OK;
}

int user_cleanup () {
    transaction_sleep_do();
    transaction_start(0, 0, NULL); // Just something to clear out the buffer...
    return RTE_OK;
}

int user_spawn_ok() {
    int rc, hb;
    hb = ((UserGlobal *) (shm->global_data))->host_busy;
    rc = hb?RTE_ERROR:RTE_OK;
    return rc;
}

```

D.4 user tpcc.h

```

/*****
*/
/* user_tpcc.h Audit: 05/30/96/
*/
/*****

/* $Id: user_tpcc.h,v 1.1 1999/02/22 06:31:06channui Exp $ */

#ifndef USER_TPCC_H
#define USER_TPCC_H

#include "data/rte_common.h"

/*****
*/
/**
run-time constant for customer last name from 0 to 255,
*/
/**
run-time constant for customer id from 0 to 1023,
*/
/**
run-time constant for item id from 0 to 8191.
*/
/*****
*/
#define LASTC 117 /*
Change for 3.1 */
#define LASTC 193
#define CUSTC 319
#define ITEM 3849

/*****
*/
/**
response type
*/
/*****
*/
#define OK 1 /*
#define ERROR -1 /*

/*****
*/
/**
transaction type
*/
/*****
*/
#define NEWORDER 1
#define PAYMENT 2
#define ORDSTAT 3
#define DELIVERY 4
#define STOCKLEV 5

/*****
*/
/**
transaction structures
*/
/*****
*/
struct neword_struct {
    char invalid; /* transaction completed successfully */
    long did;
    long cid;
    long oid; /* Order-ID returned from client */
    long nloop; /* number of order line, avg = 15 */
    char oremote; /* 1 for remote order, 10% */
    long olremote; /* number of remote order line, 1% */
    char rollback; /* actually saw rollback text on screen */
    struct items_struct {

```

```

        long olswid;
        long oliid;
        long olquantity;
    } item[15];
};

struct payment_struct {
    char invalid; /* transaction completed successfully */
    long did;
    long cid;
    long cwid;
    long cdid;
    char clast[17];
    double amount;
    char byname; /* 1 for by last name, 0 for by id */
    char remote; /* 1 for remote warehouse, 0 otherwise */
};

struct ordstat_struct {
    char invalid; /* transaction completed successfully */
    long did;
    long cid;
    char clast[17];
    char byname; /* 1 for by last name, 0 for by id */
};

struct delivery_struct {
    char invalid; /* transaction completed successfully */
    char carrier;
};

struct stocklev_struct {
    char invalid; /* transaction completed successfully */
    long threshold;
};

struct generic_struct {
    char invalid; /* transaction completed successfully */
};

typedef union transaction_info {
    char invalid;
    struct generic_struct generic;
    struct neword_struct neword;
    struct payment_struct payment;
    struct ordstat_struct ordstat;
    struct delivery_struct delivery;
    struct stocklev_struct stocklev;
} transaction_info;

struct UserGlobal {
    int total_users;
    int max_warehouses;
    int keystroke_sleep;
    int login_timeout;
    int keystroke_packet_size;
    int lastc;
    int test_state;
    int host_busy;
    double chances[MAX_TRAN_TYPE];
    double think[MAX_TRAN_TYPE];
    double emulex_response[MAX_TRAN_TYPE];
    double emulex_menu [MAX_TRAN_TYPE];
};

struct UserLocal {
    int Warehouse;
    int District;
};

/*
*/
struct user_data_header {
};
*/

extern struct UserGlobal *shmglobal;
extern struct UserLocal *shmlocal;

#endif

```

APPENDIX E: Third Party Quotes

<http://www.buynetgear.com/product.asp?sku=1039336>

FS108NA 10/100 8PORT DUAL SPEED SWITCH RJ45 W/ UPLINKBUTTON

Part Number : FS108NA
In Stock : **NO**
Platform : PC Hardware
Media : Peripherals

Price: \$84.99

[BuyNow](#)



Description:

Eight port Workgroup Fast Ethernet Switch which improves Network traffic by Segmentation. The FS108 Fast Ethernet Switch brings the 100 Mbps switching technology in a compact form factor to the small office marketplace at an aggressive price. This switch segments the network into smaller, connected subnets for improved performance, enabling the most demanding multimedia and imaging applications. Since each port is auto-speed-sensing, individual subnets or directly attached servers can easily be upgraded from 10 to 100 Mbps.

Features:

- Enhance productivity and network reliability by segmenting the workgroup into smaller units. Eight auto-sensing UTP ports One speed-sensing UTP port selectable between Normal or Uplink connection. Stores up to 4,096 MAC addresses per system. Filtering and Forwarding Rates: 14,800 packets per second for 10 Mbps ports 148,000 packets per second for 100 Mbps ports Half/full duplex switch.

[Terms and Conditions](#) [Privacy](#) - Copyright © 2002 NETGEAR™

APPENDIX G: Pricing Information

The price of the IBM-supplied configuration will be available for at least sixty days. Prices are in US Dollars and are for sale within the United States of America. Prices in other countries may vary. Discounts are determined based on the overall value of the specific components in this single quotation, without requirement for any past, future, or additional purchase. Discounts for similarly sized configurations will be similar to those quoted here, but may vary based on the components in the configuration.

For assistance regarding these prices or their applicability to any customer's requirements, please contact one of the following individuals:

Bill Casey,
EServer pSeries Offering Manager
wrcasey@us.ibm.com
(512) 838-1422

Herve Lejeune,
herve.lejeune@oracle.com
(650) 506-1894

Thank you,

Robert Schramer,
Mgr, Product Pricing & Business Evaluation -eServer pSeries
schramer@us.ibm.com
(512) 838-5297