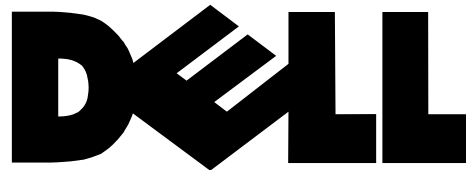




TPC BenchmarkTM H
Full Disclosure Report
DELL PowerEdge 6800/3.33GHz/8M-L3
Using
Red Hat Enterprise Linux AS Version 3.0
And
Oracle Database 10g Release 2 Enterprise Edition with
RAC



First Edition

Submitted for Review

July 08, 2005

Dell Computer Corp PowerEdge 6800 Server with Red Hat Enterprise Linux AS version 3.0 and Oracle Database 10g Release 2 Enterprise Edition with RAC

First Printing July 2005

All rights reserved. Permission is hereby granted to reproduce this document in whole or in part provided the copyright notice is included on the title page of each item reproduced.

Printed in U.S.A.

DELL believes that the technical, pricing and discounting information in this document is accurate as of its publication date. The performance information in this document is for guidance only. System performance is highly dependent on many factors including system hardware, system and user software, and user-application characteristics. Customer applications must be carefully evaluated before estimating performance. DELL does not warrant or represent that a user can or will achieve similar performance as expressed in this document.

THE TERMS AND CONDITIONS GOVERNING THE SALE OF DELL HARDWARE PRODUCTS AND THE LICENSING OF DELL SOFTWARE CONSIST SOLELY OF THOSE SET FORTH IN THE WRITTEN CONTRACTS BETWEEN DELL AND ITS CUSTOMERS. NO REPRESENTATION OR OTHER AFFIRMATION OF FACT CONTAINED IN THIS DOCUMENT INCLUDING BUT NOT LIMITED TO STATEMENTS REGARDING PRICE, CAPACITY, RESPONSE-TIME PERFORMANCE, SUITABILITY FOR USE, OR PERFORMANCE OF PRODUCTS DESCRIBED HEREIN SHALL BE DEEMED TO BE A WARRANTY BY DELL FOR ANY PURPOSE, OR GIVE RISES TO ANY LIABILITY OF DELL WHATSOEVER.

DELL assumes no responsibility for any errors that may appear in this document. DELL reserves the right to make changes in specifications and other information contained in this document without prior notice, and the reader should in all cases consult DELL to determine whether any such changes have been made.

PowerEdge is an U.S. registered trademark of the DELL.

Oracle Database 10g is a registered trademark of Oracle Corporation.
Red Hat Enterprise Linux is a registered trademark of Red Hat Inc.

Intel and Xeon MP are registered trademarks of Intel Corporation.

TPC Benchmark H is a trademark of the Transaction Processing Performance Council.

Abstract

This report document the methodology and results of the TPC Benchmark H test conducted on a cluster of 2 PowerEdge 6800 Servers using Oracle database 10g Release 2 in conformance with the requirements of the TPC-H Benchmark Specification. The operating system used for the benchmark was Red Hat Enterprise Linux Advanced Server 3.0.

Hardware	Software	Total System Cost
2xDell PowerEdge 6800 with Quad 3.33GHz Intel Xeon MP Dell PowerVault 22XS Storage Enclosures	Oracle Database 10g Release 2 Enterprise Edition Red Hat Enterprise Linux AS version 3.0	\$256,383

TPC-H Power@300GB	TPC-H Throughput@300GB	<u>OphH@300GB</u>	\$/QphH@300GB
15,188.9	9,077.9	11,742.8	\$22.00

The Transaction Processing Performance Council (TPC) developed the TPC-H Benchmark. The TPC was founded to define transactions processing benchmarks and to disseminate objective, verifiable performance data to the industry.

In order to verify compliance to the TPC-H benchmark specification, Lorna Livingtree, Performance Metrics, Inc., audited the benchmark configuration, environment and methodology used to produce and validate the test results, and the pricing model used to calculate the price/performance.



**DELL
PowerEdge 6800
with Oracle Database 10g R2**

**TPC-H Revision
2.1.0**

Report Date:
July 08, 2005

**Total System Cost
\$256,383**

**TPC-H Composite Query per Hour
11,742.8 QphH@300GB**

**Price/Performance
\$22/QphH@300GB**

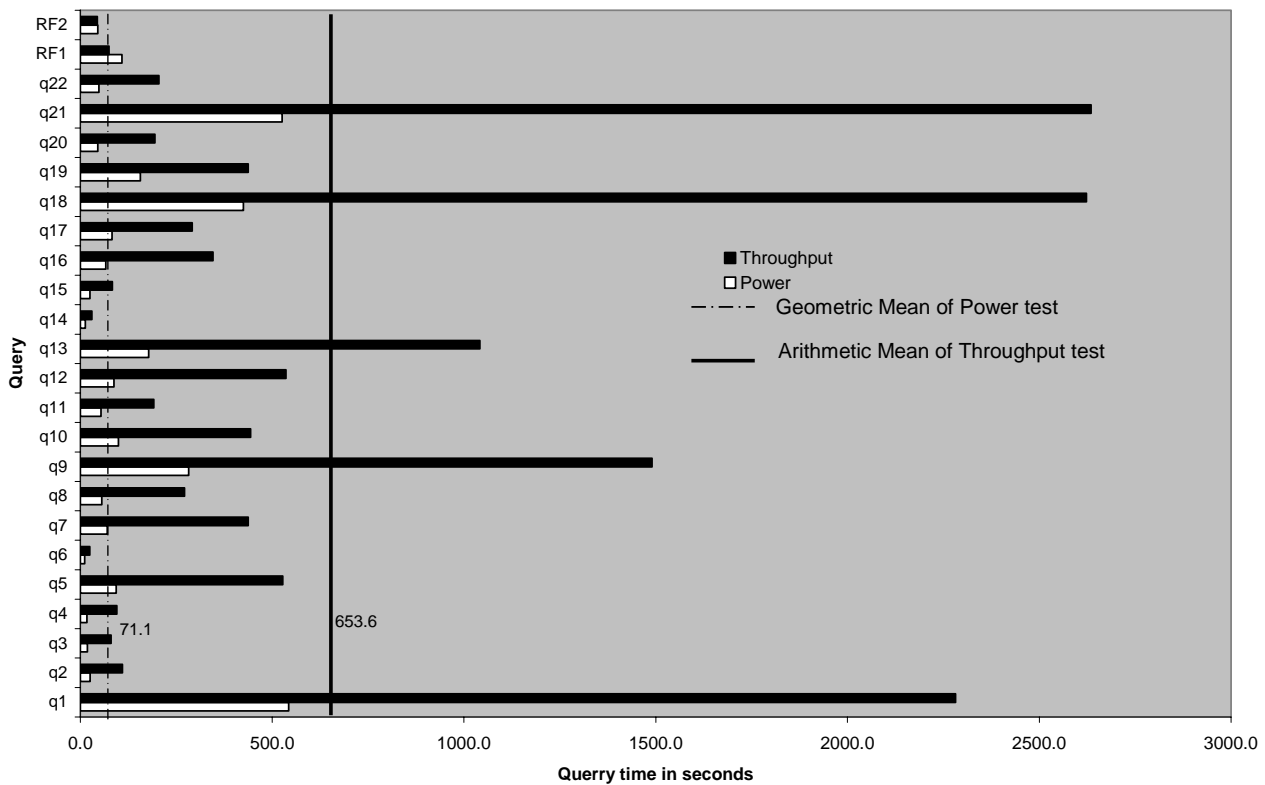
**Database Size
300GB**

Database Manager
Oracle Database 10g Release 2
Enterprise Edition with RAC and
Partitioning

Operating System
Red Hat Enterprise Linux
Advanced Server v3.0

Other Software

**Availability Date
January 08, 2006**



Database Load Time
4:07:19

Total Disk/Database Size
19.42

Load Includes Backup
N

RAID (Base tables only)
N

RAID (Base tables and auxiliary data structures)
N

RAID (Everything)
Y

System Configuration : 2 nodes
Processors (per node) : 4x 3.33GHz/1024KB L2/8MB L3 Intel Xeon MP
Memory: (per node) : 16GB RAM
Disk Drives: 172 – 36GB 15k HDD

“Database Size includes only raw data (e.g., no temp, index, redundant storage space, etc.)”

Dell		PowerEdge 6800		TPC-H rev 2.1.0 EXECUTIVE SUMMARY PAGE 2 OF 2			
		Client/Server		Report Date: 08-Jul-05			
Description	Part Number	Third Party	Unit Price	Qty	Extended Price	3 yr. Maint. Price	
Server Hardware							
		Brand	Pricing				
Dell PowerEdge 6800	221-8265	1	6,063	2	12,126	1,400	
Intel Xeon MP 3.33GHz / 8MB L3 - 4 processors	311-4931	1	14,997	2	29,994	-	
16 GB,DDR, 16 x 1024MB DIMMS	311-4651	1	7,498	2	14,996	-	
36 GB U320 SCSI 15K RPM Hard Drive	340-1897	1	299	4	1,196	-	
Crossover cable	43-425	3	2	3	5	-	
Dell 15" Monitor	320-2907	1	135	2	270	-	
PERC4-DC 128MB U320 SCSI RAID controller	341-1285	1	799	6	4,794	0	
PERC4e-DC 128MB U320 SCSI RAID controller	341-1913	1	799	8	6,392	0	
Subtotal					69,773	1,400	
PowerVault Disk Subsystem							
PV220S, U3 SCSI, PS, Rack mount	220-4476, etc.	1	1,668	14	23,352	6,986	
SCSI Cables	310-0679	1	99	14	1,386	-	
36 GB U320 SCSI 15K RPM Hard Drive *	340-9371	1	249	168	41,832	-	
42U Rack	220-4492	1	964	2	1,928	-	
Subtotal					68,498	6,986	
Server Software							
Oracle Database 10g Release 2, Enter. Ed., Named User Plus, 3yrs		2	10,000	8	80,000		
Partitioning, Named User Plus, 3yrs		2	2,500	8	20,000		
Real Application Clusters, Named User Plus, 3yrs		2	5,000	8	40,000		
Oracle Database server support package for 3yrs, per server		2	6,000			12,000	
Oracle Mandatory E-Business Discount		2			(\$22,800)	0	
Red Hat Enterprise Linux Advanced Server, 3yrs SUB NFI	420-4250	1	2,699	2	5,398	0	
Subtotal					122,598	12,000	
Other Discounts***					(24,872)	-	
Total					235,997	20,386	
Notes: * Maint. included in PowerVault 220S disk pod or PV650F/630F fibre channel disk pod					Three-Year Cost of Ownership: \$256,383		
** 10% or minimum 2 spares are added in place of onsite service (products have a three year return-to-vendor warranty) Pricing: 1 - Dell 2 - Oracle 3 - Fastcomputerparts					QphH Rating: 11742.80		
*** Discount based upon total system cost as purchased by a regular customer. Pricing may be verified by calling 1-800-BUY-DELL referencing quote numbers 230402082 as complex quotes.							
Audited by Lorna Livingtree, Performance Metrics Inc.					\$/ QphH: 22		
Prices used in TPC benchmarks reflect the actual prices a customer would pay for a one-time purchase of the stated components. Individually negotiated discounts are not permitted. Special prices based on assumptions about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed components. For complete details, see the pricing sections of the TPC benchmark specifications. If you find that the stated prices are not available according to these items, please inform the TPC at pricing@tpc.org.							



DELL
PowerEdge 6800
w/ Oracle 10g R2 / RHEL
v3.0

TPC-H Revision
2.1.0

Report Date:
July 08, 2005

Numerical Quantities Summary

Measurement Results:

Database Scale Factor	300
Total Data Storage/Database Size	19.42
Start of Database Load	06/15/2005 13:42:28
End of Database Load	06/15/2005 17:49:47
Database Load Time	4:07:19
Query Streams for Throughput Test	6
TPC-H Power	15,189.9
TPC-H Throughput	9,077.9
TPC-H Composite Query-per-Hour Metric (QpH@300GB)	11,742.8
Total System Price Over 3 Years	\$256,383
TPC-H Price Performance Metric (\$/QpH@300GB)	\$22

Measurement Interval:

Measurement Interval in Throughput Test (Ts) = 15,704.0 seconds

Duration of stream execution:

	Seed	Start Date/Time	End Date/Time	Duration
Stream00	615174947	6/15/2005 18:12:13	6/15/2005 19:00:56	00:48:43
Stream01	615174948	6/15/2005 19:01:43	6/15/2005 23:11:34	4:09:51
Stream02	615174949	6/15/2005 19:01:43	6/15/2005 23:09:55	4:08:12
Stream03	615174950	6/15/2005 19:01:43	6/15/2005 23:44:33	3:42:50
Stream04	615174951	6/15/2005 19:01:43	6/15/2005 23:55:12	3:53:29
Stream05	615174952	6/15/2005 19:01:43	6/15/2005 23:01:58	4:00:15
Stream06	615174953	6/15/2005 19:01:43	6/15/2005 23:05:00	4:03:17
Refresh00		6/15/2005 18:10:24	6/15/2005 18:12:12	0:01:48
		6/15/2005 19:00:56	6/15/2005 19:01:41	0:00:45
Refresh01		6/15/2005 23:11:34	6/15/2005 23:13:53	0:02:19
Refresh02		6/15/2005 23:13:53	6/15/2005 23:15:52	0:01:59
Refresh03		6/15/2005 23:15:52	6/15/2005 23:17:38	0:01:46
Refresh04		6/15/2005 23:17:38	6/15/2005 23:19:29	0:01:51
Refresh05		6/15/2005 23:19:29	6/15/2005 23:21:21	0:01:52
Refresh06		6/15/2005 23:21:21	6/15/2005 23:23:27	0:02:06



Numerical Quantities Summary

Timing Intervals in Seconds:

Query	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8
Stream00	543.2	25.7	18.0	17.0	93.1	11.6	69.9	55.8
Stream01	2551.2	115.7	92.1	19.6	585.7	17.6	412.9	274.1
Stream02	2193.5	117.9	90.7	129.2	535.3	45.1	460.0	318.4
Stream03	1645.3	103.6	81.7	93.7	529.5	20.5	393.8	285.5
Stream04	2547.7	135.0	61.4	119.7	327.7	18.4	240.2	248.0
Stream05	2244.7	97.7	77.2	87.3	610.1	20.2	355.6	265.6
Stream06	2509.4	89.8	79.3	121.8	578.3	25.3	763.4	239.3
Min Qi	1645.3	89.8	61.4	19.6	327.7	17.6	240.2	239.3
Max Qi	2551.2	135.0	92.1	129.2	610.1	45.1	763.4	318.4
Avg Qi	2282.0	110.0	80.4	95.2	527.8	24.5	437.7	271.8
Query	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16
Stream00	282.5	99.2	54.1	87.4	177.9	12.8	24.7	66.7
Stream01	1519.5	616.6	166.9	551.8	1022.3	29.3	94.3	305.2
Stream02	1834.4	390.9	237.0	515.6	1162.7	28.4	91.0	414.6
Stream03	1079.6	440.4	191.2	591.7	1138.4	23.1	61.6	358.9
Stream04	1656.7	605.0	177.4	369.2	754.4	32.2	66.7	424.5
Stream05	983.1	430.1	224.8	489.2	981.7	42.6	72.7	285.6
Stream06	1871.3	182.2	152.2	698.2	1191.1	24.4	115.5	285.4
Min Qi	983.1	182.2	152.2	369.2	754.4	23.1	61.6	285.4
Max Qi	1871.3	616.6	237.0	698.2	1191.1	42.6	115.5	424.5
Avg Qi	1490.7	444.2	191.6	535.9	1041.8	30.0	83.6	345.7
Query	Q17	Q18	Q19	Q20	Q21	Q22	RF1	RF2
Stream00	82.6	425.0	156.3	45.6	525.6	48.5	108.6	45.4
Stream01	188.2	2260.9	387.3	166.1	3274.0	340.6	84.1	54.4
Stream02	702.3	2534.8	291.6	302.2	2278.5	218.4	77.2	41.6
Stream03	204.5	2439.8	950.7	201.4	2390.5	144.7	68.5	37.4
Stream04	231.1	3155.9	311.9	125.5	2239.2	161.3	67.2	44.5
Stream05	241.9	2959.2	304.5	198.8	3243.9	199.2	70.9	40.4
Stream06	183.2	2387.7	380.2	171.9	2383.2	164.1	78.5	47.6
Min Qi	183.2	2260.9	291.6	125.5	2239.2	144.7	67.2	37.4
Max Qi	702.3	3155.9	950.7	302.2	3274.0	340.6	84.1	54.4
Avg Qi	291.9	2623.0	437.7	194.3	2634.9	204.7	74.4	44.3

Table of Contents

- General Items..... 1**
 - Test Sponsor..... 1
 - Parameter Settings..... 1
 - Configuration Items 1

- Clause 1: Logical Database Design 4**
 - Table Definitions..... 4
 - Physical Organization of Database 4
 - Horizontal Partitioning 4
 - Replication 4

- Clause 2: Queries and Update Functions..... 5**
 - Query Language 5
 - Random Number Generation 5
 - Substitution Parameters Generation 5
 - Query Text and Output Data from Database..... 5
 - Query Substitution Parameters and Seeds Used..... 5
 - Isolation Level 5
 - Refresh Function Source Code 5

- Clause 3: Database System Properties 7**
 - Atomicity..... 7
 - Completed Transaction..... 7
 - Aborted Transaction..... 7
 - Consistency..... 7
 - Consistency Test 7
 - Isolation..... 8
 - Read-Write Conflict with Commit 8
 - Read-Write Conflict with Rollback..... 8
 - Write-Write Conflict with Commit 8
 - Write-Write Conflict with Rollback 8
 - Concurrent Read and Write Transactions on Different Tables..... 9
 - Update Transactions During Continuous Read-Only Query Stream..... 9
 - Durability 9
 - Failure of a Durable Medium..... 9
 - System Crash 10
 - Memory Failure..... 10

- Clause 4: Scaling and Database Population 11**
 - Initial Cardinality of Tables 11
 - Distribution of Tables and Logs Across Media 11
 - Partitioning and Replication..... 19

RAID Implementations	19
DBGEN Version and Modifications.....	19
Database Load time	19
Data Storage Ratio.....	19
Database Load Mechanism Details and Illustration.....	19
Clause 5: Performance Metrics and Execution Rules	21
Steps in the Power Test	21
Timing Intervals.....	21
Number of Streams for The Throughput Test.....	21
Start/Finish Time of Each Query Stream.....	21
Total Elapsed Time	21
Start/Finish Time for Update Function.....	21
Timing Intervals for Each Query and Each Update	21
Performance Metrics	22
Reproducibility Method.....	22
System Activity Between Run1 and Run2	22
Clause 6: SUT and Driver Implementation	23
Driver.....	23
Implementation Specific Layer (ISL).....	23
Profile-Directed Optimization.....	18
Clause 7: Pricing	24
Hardware and Software Used in the Priced System.....	24
Total Three Year Price	24
Availability Date.....	24
Clause 8: Auditor-Related Items	25
Auditor's Report	25
Appendix A: Parameter settings	A
Appendix B: Database Build Scripts.....	B
Appendix C: ACID Scripts.....	C
Appendix D: Query text and Output.....	D
Appendix E: Seed and Input Parameters.....	E
Appendix F: Benchmark Scripts	F
Appendix G: Price Quotations.....	G

General Items

Test Sponsor

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

DELL is the sponsor of this TPC Benchmark™ H result.

Parameter Settings

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

- *Database Tuning Options*
- *Optimizer/Query execution options*
- *Query processing tool/language configuration parameters*
- *Recovery/commit options*
- *Consistency/locking options*
- *Operating system and configuration parameter*
- *Configuration parameters and options for any other software in the pricing structure*
- *Compiler optimization options*

Providing a full list of all parameters and options can satisfy this requirement, as long as all those that have been modified from their default values have been clearly identified and these parameters and options are only set once.

Comment 1: *In the event that some parameters and options are set multiple times, it must be easily discernible by an interested reader when the parameter or option was modified and what new value it received each time.*

Comment 2: *This requirement can be satisfied by providing a full list of all parameters and options, as long as all those that have been modified from their default values have been clearly identified and these parameters and options are only set once.*

Details of system and database configurations and parameters are provided in Appendix A.

Configuration Items

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

- *Number and type of processors*
- *Size of allocated memory, and any specific mapping/partitioning of memory in the test.*
- *Number and type of disk units (and controllers, if applicable).*
- *Number of channels or bus connections to disk units, including their protocol type.*
- *Number of LAN (e.g. Ethernet) Connections, including routers, workstations, terminals, etc., that were physically used in the test or are incorporated into the pricing structure.*

- *Type and the run-time execution location of software components (e.g., DBMS, query processing tools/languages, middle-ware components, software drivers, etc.).*

The System Under Test (SUT), 2x DELL PowerEdge 6800 Servers, depicted in the next diagram consists of:

- 8 - 3.33GHz Intel ® Xeon MP Processors, each with 8MB of L3 cache.
- 32GB RAM.
- 8 - DELL PERC4e-DC RAID Controllers, each with 128MB RAM of cache.
- 6 - DELL PERC4-DC RAID Controllers, each with 128MB RAM of cache.
- 14 - DELL PowerVault 22XS.
- 168 – 18GB,U320 15k SCSI
- 4 – 36GB 15k HDD.

Per *Clause 7.1.1:*

Specifically excluded from the priced system calculation are:

- *End-user communication devices and related cables, connectors, and concentrators;*
- *Equipment and tools used exclusively in the production of the full disclosure report;*
- *Equipment and tools used exclusively for the execution of the DBGEN or QGEN (see Clause 2.1.4 and Clause 4.2.1) programs.*

Measured and Priced Configuration

The measured and priced configurations are the same.

PE6800A

4xIntel Xeon MP 3.33GHz/8MB-L3
 16 GB DDR
 4xPERC4e-DC Controllers
 3xPERC4-DC Controllers
 2x36GB, U320, 15K SCSI

DBMS

Oracle 10g R2 w/ RAC

OS

Red Hat Linux AS 3.0 w/
 2.4.21-27.0.1.ELsmp kernel

PE6800B

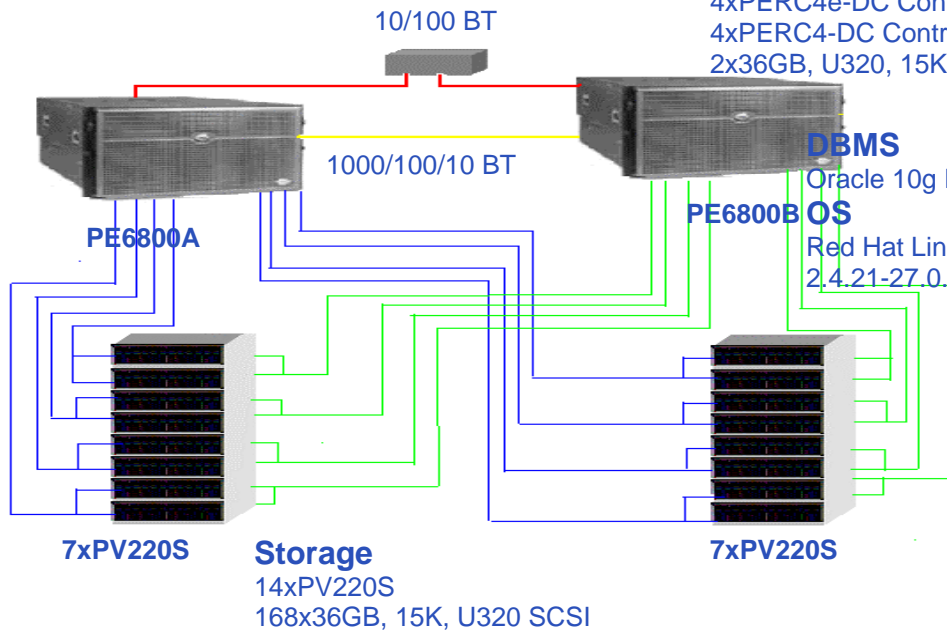
4xIntel Xeon MP 3.33GHz/8MB-L3
 16 GB DDR
 4xPERC4e-DC Controllers
 4xPERC4-DC Controllers
 2x36GB, U320, 15K SCSI

DBMS

Oracle 10g R2 w/ RAC

OS

Red Hat Linux AS 3.0 w/
 2.4.21-27.0.1.ELsmp kernel



Clause 1: Logical Database Design

Table Definitions

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

Appendix B contains the scripts that create and analyze the tables and indexes for the TPC-H database.

Physical Organization of Database

The physical organization of tables and indices, within the test and qualification databases, must be disclosed. If the column ordering of any table is different from that specified in Clause 1.4, it must be noted.

No record clustering or index clustering was used for this benchmark. Column ordering was reordered in tables. Refer to the table create statements in Appendix B for further details.

Horizontal Partitioning

Horizontal partitioning of tables and rows in the test and qualification databases (see Clause 1.5.4) must be disclosed.

Horizontal partitioning was used for all base and index tables except NATION and REGION. The details of this partitioning can be understood by examining the syntax of the table and index definition statements in Appendix B. Similar partitioning was used in the qualification database.

Replication

Any replication of physical objects must be disclosed and must conform to the requirements of Clause 1.5.6.

Replication was not used for this benchmark.

Clause 2: Queries and Update Functions

Query Language

The query language used to implement the queries must be identified.

SQL was the query language used to implement all queries.

Random Number Generation

The method of verification for the random number generation must be described unless the supplied DBGEN and QGEN were used.

Version 1.3.0 of DBGEN and version 1.3.0 of QGEN were used to generate the random numbers for this TPC-H benchmark.

Substitution Parameters Generation

The method used to generate values for substitution parameters must be disclosed. If QGEN is not used for this purpose, then the source code of any non-commercial tool used must be disclosed. If QGEN is used, the version number, release number, modification number and patch level of QGEN must be disclosed.

QGEN version 1.3.0 was used to generate the substitution parameters.

Query Text and Output Data from Database

The executable query text used for query validation must be disclosed along with the corresponding output data generated during the execution of the query text against the qualification database. If minor modifications (see Clause 2.2.3) have been applied to any functional query definitions or approved variants in order to obtain executable query text, these modifications must be disclosed and justified. The justification for a particular minor query modification can apply collectively to all queries for which it has been used. The output data for the power and throughput tests must be made available electronically upon request.

Appendix D contains the query text and query output. The minor query modifications used in this implementation include the following:

Query Substitution Parameters and Seeds Used

All the query substitution parameters used during the performance test must be disclosed in tabular format, along with the seeds used to generate these parameters.

Appendix E contains the seed and query substitution parameters.

Isolation Level

The isolation level used to run the queries must be disclosed. If the isolation level does not map closely to one of the isolation levels defined in Clause 3.4, additional descriptive detail must be provided.

The queries and transactions were run with isolation level 3 (repeatable read).

Refresh Function Source Code

The details of how the refresh functions were implemented must be disclosed (including source code of any non-commercial program used).

The source code for the refresh functions is included in Appendix F.

Clause 3: Database System Properties

Atomicity

The results of the ACID tests must be disclosed along with a description of how the ACID requirements were met. This includes disclosing the code written to implement the ACID Transaction and Query.

Atomicity of Completed Transaction

Perform the ACID Transaction for a randomly selected set of input data and verify that the appropriate rows have been changed in the ORDER, LINEITEM, and HISTORY tables.

1. The total prices from the ORDER table and the extended price from the LINEITEM table were retrieved for a randomly selected order key.
2. The ACID Transaction was performed using the order key from step 1.
3. The ACID Transaction was committed.
4. The total price from the ORDER table and the extended price from the LINEITEM table were retrieved for the same order key. It was verified that the appropriate rows had been changed.

Aborted Transaction

Perform the ACID transaction for a randomly selected set of input data, substituting a ROLLBACK of the transaction for the COMMIT of the transaction. Verify that the appropriate rows have not been changed in the ORDER, LINEITEM, and HISTORY tables.

1. The total price from the ORDER table and the extended price from the LINEITEM table were retrieved for a randomly selected order key.
2. The ACID Transaction was performed using the order key from step 1. The transaction was stopped prior to the commit.
3. The ACID Transaction was rolled back.
4. The total price from the ORDER table and the extended price from the LINEITEM table were retrieved for the same order key, and verified to not have changed.

Consistency

A consistent state for the TPC-H database is defined to exist when:

*$$O_TOTALPRICE = \text{SUM}(\text{trunc}(\text{trunc}(L_EXTENDEDPRICE * (1 - L_DISCOUNT), 2) * (1 + L_TAX), 2))$$
 for each ORDERS and LINEITEM defined by (O_ORDERKEY = L_ORDERKEY)*

Consistency Test

Verify that ORDER and LINEITEM tables are initially consistent, submit the prescribed number of ACID Transactions with randomly selected input parameters, and re-verify the consistency of the ORDER and LINEITEM tables.

1. The consistency of the ORDER and LINEITEM tables was verified based on a sample of O_ORDERKEYs.
2. 100 ACID transactions were submitted from each of 6 execution streams.
3. The consistency of the ORDER and LINEITEM tables was verified a second time with the same O_ORDERKEYs.

Isolation

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

Read-Write Conflict with Commit

Demonstrate isolation for the read-write conflict of a read-write transaction and a read-only transaction when the read-write transaction is committed.

1. An ACID transaction was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID transaction was suspended prior to commit.
2. An ACID query was started for the same O_KEY used in step 1. The ACID query completed and did not see the uncommitted changes made by the ACID transaction.
3. The ACID transaction was committed.

Read-Write Conflict with Rollback

Demonstrate isolation for the read-write conflict of a read-write transaction and a read-only transaction when the read-write transaction is rolled back.

1. An ACID transaction was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID transaction was suspended prior to rollback.
2. An ACID query was started for the same O_KEY used in step 1. The ACID query completed and did not see the uncommitted changes made by the ACID transaction.
3. The ACID transaction was rolled back.

Write-Write Conflict with Commit

Demonstrate isolation for the write-write conflict of two update transactions when the first transaction is committed.

1. An ACID transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID transaction was suspended prior to commit.
2. A second ACID transaction, T2, was started using the same O_KEY and L_KEY and a different randomly selected DELTA.
3. T2 waited.
4. T1 was allowed to commit and then T2 completed.
5. It was verified that T2.L_EXTENDPRICE was calculated correctly.

Write-Write Conflict with Rollback

Demonstrate isolation for the write-write conflict of two update transactions when the first transaction is rolled back.

1. An ACID transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID transaction was suspended prior to rollback.
2. A second ACID transaction, T2, was started using the same O_KEY and L_KEY and a different randomly selected DELTA.
3. T2 waited.
4. T1 was allowed to rollback and then T2 completed.
5. It was verified that T2.L_EXTENDPRICE was calculated correctly.

Concurrent Read and Write Transactions on Different Tables

Demonstrate the ability of read and write transactions affecting different database tables to make progress concurrently.

The following steps were performed:

1. An ACID Transaction T1 for a randomly selected O_KEY, L_KEY and DELTA. The ACID Transaction T1 was suspended prior to commit.
2. Another ACID Transaction T2 was started using random values for PS_PARTKEY and PS_SUPPKEY.
3. T2 completed.
4. T1 completed and the appropriate rows in the ORDER, LINEITEM and HISTORY tables were changed.

Update Transactions during Continuous Read-Only Query Stream

Demonstrate the continuous submission of arbitrary (read-only) queries against one or more tables of the database does not indefinitely delay update transactions affecting those tables from making progress.

The following steps were performed:

1. An ACID Transaction T1 was started, executing Q1 against the qualification database. The substitution parameter was chosen from the interval [0..2159] so that the query ran for a sufficient amount of time.
2. Before T1 completed, an ACID Transaction T2 was started using randomly selected values of O_KEY, L_KEY and DELTA.
3. T2 completed before T1 completed.
4. It was verified that the appropriate rows in the ORDER, LINEITEM and HISTORY tables were changed.

Durability

The tested system must guarantee durability: the ability to preserve the effects of committed transactions and insure database consistency after recovery from any one of the failures listed in Clause 3.5.2.

Failure of a Durable Medium

Guarantee the database and committed updates are preserved across a permanent irrecoverable failure of any single durable medium containing TPC-H database tables or recovery log tables.

The database logs were stored on a RAID-10.

The tables for the database were stored on RAID-10 stripes.

During the test, one disk from the DELL PV220 enclosure containing data and re-do log files was removed. As all data and re-do log files resided on RAID 10 volumes, the test continued uninterrupted.

System Crash

Guarantee the database and committed updates are preserved across an instantaneous interruption (system crash/system hang) in processing which requires the system to reboot to recover.

1. Six streams of ACID transactions were started.
2. While the streams of ACID transactions were running the system was powered off.
3. When power was restored the system rebooted and the database was restarted.
4. The database went through a recovery period.
5. The success file and the HISTORY table counts were compared, and they matched.

Memory Failure

Guarantee the database and committed updates are preserved across failure of all or part of memory (loss of contents).

The system crash test and the memory failure test were combined. See the previous section.

Appendix C contains source code for all transactions.

Clause 4: Scaling and Database Population

Initial Cardinality of Tables

The cardinality (e.g., the number of rows) of each table of the test database, as it existed at the completion of the database load (see clause 4.2.5) must be disclosed.

Initial number of rows

Table	Occurrences
Orders	450,000,000
Lineitem	1,799,989,091
Customer	45,000,000
Part	60,000,000
Supplier	3,000,000
Partsupp	240,000,000
Nation	25
Region	5

Distribution of Tables and Logs Across Media

The distribution of tables and logs across all media must be explicitly described.

The Oracle 10g Database was stored on a shared-storage system composed of the following:

- 14 x DELL PERC4-DC Disk Controllers
- 14 x PowerVault 22XS Disk Enclosures
- 168x 36GB,U320, 15k rpm, SCSI disk drives
- 4 x 36GB 15k rpm disk drives

14 logical volumes were configured each spanning 12 physical drives across both SCSI channels of the controller in a RAID10 formation. The PV220 disk enclosures and PERC4-DC controllers were configured for cluster-mode thereby enabling simultaneous storage access from both server nodes. Both nodes had 2 36GB, SCSI disks each for the operating system. Each logical volume had 2 Linux partitions except for volume1 and volume2, which had 3 partitions. Each logical volume had an ext2 Linux file system for flat file storage and a raw partition for the database files. Volume1 and Volume2 each had an extra raw partition for the Oracle Cluster Registry (OCR) and Voting disk respectively. The database raw partitions were configured into an Automatic Storage Management (ASM) disk group.

Oracle ASM transparently managed the placement of the database and log data onto the shared-storage system.

A detailed description of distribution of database file groups and log can be found below.

Distribution of Storage

Red Hat Linux AS v3.0 Disk Administration			DELL PERC4e-DC Configuration					
Disk 0 208080MB			Controller # 0					
Partition			Slot# 1		Channels			
1	2	3		SCSI ID	A	B		
/dev/sdc1 Raw7 DB data 100000MB Raw FS	/dev/sdc2 Raw15 OCR 200MB Raw FS	/dev/sdc3 ff7 Flat Files 100000MB Ext2 FS		0	A0-1	A1-1		
				1	A0-2	A1-2		
				2	A0-3	A1-3		
				3	A0-4	A1-4		
				4	A0-5	A1-5		
				5	A0-6	A1-6		
				8				
				9				
				10				
				11				
				12				
				13				
				14				
				15				

Red Hat Linux AS v3.0 Disk Administration			DELL PERC4e-DC Configuration					
Disk 1 208080MB			Controller # 0					
Partition			Slot# 1		Channels			
1	2	3		SCSI ID	A	B		
/dev/sdd1 raw8 DB data 100000MB Raw FS	/dev/sdd2 Raw16 Voting Disk 200MB Raw FS	/dev/sdd3 ff8 Flat Files 100000MB Ext2 FS		0				
				1				
				2				
				3				
				4				
				5				
				8				
				9	A0-6	A1-6		
				10	A0-7	A1-7		
				11	A0-8	A1-8		
				12	A0-9	A1-9		
				13	A0-10	A1-10		
				14	A0-11	A1-11		
				15				

Red Hat Linux AS v3.0 Disk Administration		DELL PERC4e-DC Configuration			
Disk 2 208080MB		Controller # 1			
Partition		Slot# 3	Channels		
1	2	SCSI ID	A	B	
/dev/sde1 raw9 DB data 100000MB Raw FS	/dev/sde2 ff9 Flat Files 100000MB Ext2 FS	0	A0-1	A1-1	
		1	A0-2	A1-2	
		2	A0-3	A1-3	
		3	A0-4	A1-4	
		4	A0-5	A1-5	
		5	A0-6	A1-6	
		8			
		9			
		10			
		11			
		12			
		13			
		14			
		15			

Red Hat Linux AS v3.0 Disk Administration		DELL PERC4e-DC Configuration			
Disk 3 208080MB		Controller # 1			
Partition		Slot# 3	Channels		
1	2	SCSI ID	A	B	
/dev/sdf1 raw10 DB data 100000MB Raw FS	/dev/sdf2 ff10 Flat Files 100000MB Ext2 FS	0			
		1			
		2			
		3			
		4			
		5			
		8			
		9	A0-6	A1-6	
		10	A0-7	A1-7	
		11	A0-8	A1-8	
		12	A0-9	A1-9	
		13	A0-10	A1-10	
		14	A0-11	A1-11	
		15			

Red Hat Linux AS v3.0 Disk Administration		DELL PERC4e-DC Configuration			
Disk 4 208080MB		Controller # 2			
Partition		Slot# 4	Channels		
1	2	SCSI ID	A	B	
/dev/sdg1 raw11 DB data 100000MB Raw FS	/dev/sdg2 ff11 Flat Files 100000MB Ext2 FS	0	A0-1	A1-1	
		1	A0-2	A1-2	
		2	A0-3	A1-3	
		3	A0-4	A1-4	
		4	A0-5	A1-5	
		5	A0-6	A1-6	
		8			
		9			
		10			
		11			
		12			
		13			
		14			
		15			

Red Hat Linux AS v3.0 Disk Administration		DELL PERC4e-DC Configuration			
Disk 5 208080MB		Controller # 2			
Partition		Slot# 4	Channels		
1	2	SCSI ID	A	B	
/dev/sdh1 raw12 DB data 100000MB Raw FS	/dev/sdh2 ff12 Flat Files 100000MB Ext2 FS	0			
		1			
		2			
		3			
		4			
		5			
		8			
		9	A0-6	A1-6	
		10	A0-7	A1-7	
		11	A0-8	A1-8	
		12	A0-9	A1-9	
		13	A0-10	A1-10	
		14	A0-11	A1-11	
		15			

Disk 6 208080MB		Controller # 3					
Partition		Slot# 5		Channels			
1	2		SCSI ID	A	B		
/dev/sdi1 raw13 DB data 100000MB Raw FS	/dev/sdi2 ff13 Flat Files 100000MB Ext2 FS		0	A0-1	A1-1		
			1	A0-2	A1-2		
			2	A0-3	A1-3		
			3	A0-4	A1-4		
			4	A0-5	A1-5		
			5	A0-6	A1-6		
			8				
			9				
			10				
			11				
			12				
			13				
			14				
			15				

Red Hat Linux AS v3.0 Disk Administration		DELL PERC4e-DC Configuration						
Disk 7 208080MB		Controller # 3						
Partition		Slot# 5		Channels				
1	2		SCSI ID	A	B			
/dev/sdj1 raw13 DB data 100000MB Raw FS	/dev/sdj2 ff13 Flat Files 100000MB Ext2 FS		0					
			1					
			2					
			3					
			4					
			5					
			8					
			9		A0-6	A1-6		
			10		A0-7	A1-7		
			11		A0-8	A1-8		
			12		A0-9	A1-9		
			13		A0-10	A1-10		
			14		A0-10	A1-11		
			15					

Red Hat Linux AS v3.0 Disk Administration	DELL PERC4-DC Configuration
---	-----------------------------

Disk 8 208080MB		Controller # 4			
Partition		Slot# 2	Channels		
1	2	SCSI ID	A	B	
/dev/sdk1 raw1 DB data 100000MB Raw FS	/dev/sdk2 ff1 Flat Files 100000MB Ext2 FS	0	A0-1	A1-1	
		1	A0-2	A1-2	
		2	A0-3	A1-3	
		3	A0-4	A1-4	
		4	A0-5	A1-5	
		5	A0-6	A1-6	
		8			
		9			
		10			
		11			
		12			
		13			
		14			
		15			
		Red Hat Linux AS v3.0 Disk Administration		DELL PERC4-DC Configuration	
Disk 9 208080MB		Controller # 4			
Partition		Slot# 2	Channels		
1	2	SCSI ID	A	B	
/dev/sdl1 raw2 DB data 100000MB Raw FS	/dev/sdl2 ff2 Flat Files 100000MB Ext2 FS	0			
		1			
		2			
		3			
		4			
		5			
		8			
		9	A0-6	A1-6	
		10	A0-7	A1-7	
		11	A0-8	A1-8	
		12	A0-9	A1-9	
		13	A0-10	A1-10	
		14	A0-11	A1-11	
		15			

Red Hat Linux AS v3.0 Disk Administration	DELL PERC4-DC Configuration
---	-----------------------------

Disk 10 208080MB		Controller # 5			
Partition		Slot# 6	Channels		
1	2	SCSI ID	A	B	
/dev/sdm1 raw3 DB data 100000MB Raw FS	/dev/sdm2 ff3 Flat Files 100000MB Ext2 FS	0	A0-1	A1-1	
		1	A0-2	A1-2	
		2	A0-3	A1-3	
		3	A0-4	A1-4	
		4	A0-5	A1-5	
		5	A0-6	A1-6	
		8			
		9			
		10			
		11			
		12			
		13			
		14			
		15			

Red Hat Linux AS v3.0 Disk Administration		DELL PERC4-DC Configuration			
Disk 11 208080MB		Controller # 5			
Partition		Slot# 6	Channels		
1	2	SCSI ID	A	B	
/dev/sdn1 raw4 DB data 100000MB Raw FS	/dev/sdn2 ff4 Flat Files 100000MB Ext2 FS	0			
		1			
		2			
		3			
		4			
		5			
		8			
		9	A0-6	A1-6	
		10	A0-7	A1-7	
		11	A0-8	A1-8	
		12	A0-9	A1-9	
		13	A0-10	A1-10	
		14	A0-11	A1-11	
		15			

Red Hat Linux AS v3.0 Disk Administration		DELL PERC4-DC Configuration			
Disk 12 208080MB		Controller # 6			

Partition		Slot# 7		Channels			
1	2		SCSI ID	A	B		
/dev/sdo1 raw5 DB data 100000MB Raw FS	/dev/sdo2 ff5 Flat Files 100000MB Ext2 FS		0	A0-1	A1-1		
			1	A0-2	A1-2		
			2	A0-3	A1-3		
			3	A0-4	A1-4		
			4	A0-5	A1-5		
			5	A0-6	A1-6		
			8				
			9				
			10				
			11				
			12				
			13				
			14				
			15				

Red Hat Linux AS v3.0 Disk Administration		DELL PERC4-DC Configuration						
Disk 13 208080MB		Controller # 6						
Partition		Slot# 7		Channels				
1	2		SCSI ID	A	B			
/dev/sdp1 raw6 DB data 100000MB Raw FS	/dev/sdp2 ff6 Flat Files 100000MB Ext2 FS		0					
			1					
			2					
			3					
			4					
			5					
			8					
			9		A0-6	A1-6		
			10		A0-7	A1-7		
			11		A0-8	A1-8		
			12		A0-9	A1-9		
			13		A0-10	A1-10		
			14		A0-11	A1-11		
			15					

Partitioning and Replication

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

The database was not replicated.

Horizontal partitioning was used for base tables except NATION and REGION.

RAID Implementations

Implementations may use some form of RAID to ensure high availability. If used for data, auxiliary storage (e.g. indexes) or temporary space, the level of RAID used must be disclosed for each device.

RAID 10 was used for all tables, indexes, temp files, logs and flat files.

DBGEN Version and Modifications

The version number, release number, modification number, and patch level of DBGEN must be disclosed. Any modifications to the DBGEN (see Clause 4.2.1) source code...must be disclosed. In the event that a program other than DBGEN was used to populate the database, it must be disclosed in its entirety.

The supplied DBGEN Version 1.3.0 was used for database population. No modifications were made to DBGEN.

Database Load time

The database load time for the test database (see clause 4.3) must be disclosed.

Database load time was 4hours 7 minutes 19 seconds.

Data Storage Ratio

The data storage ratio must be disclosed. It is computed by dividing the total data storage of the priced configuration (expressed in GB) by the size chosen for the test database as defined in 4.1.3.1. The ratio must be reported to the nearest 1/100, rounded up.

Data Storage Ratio

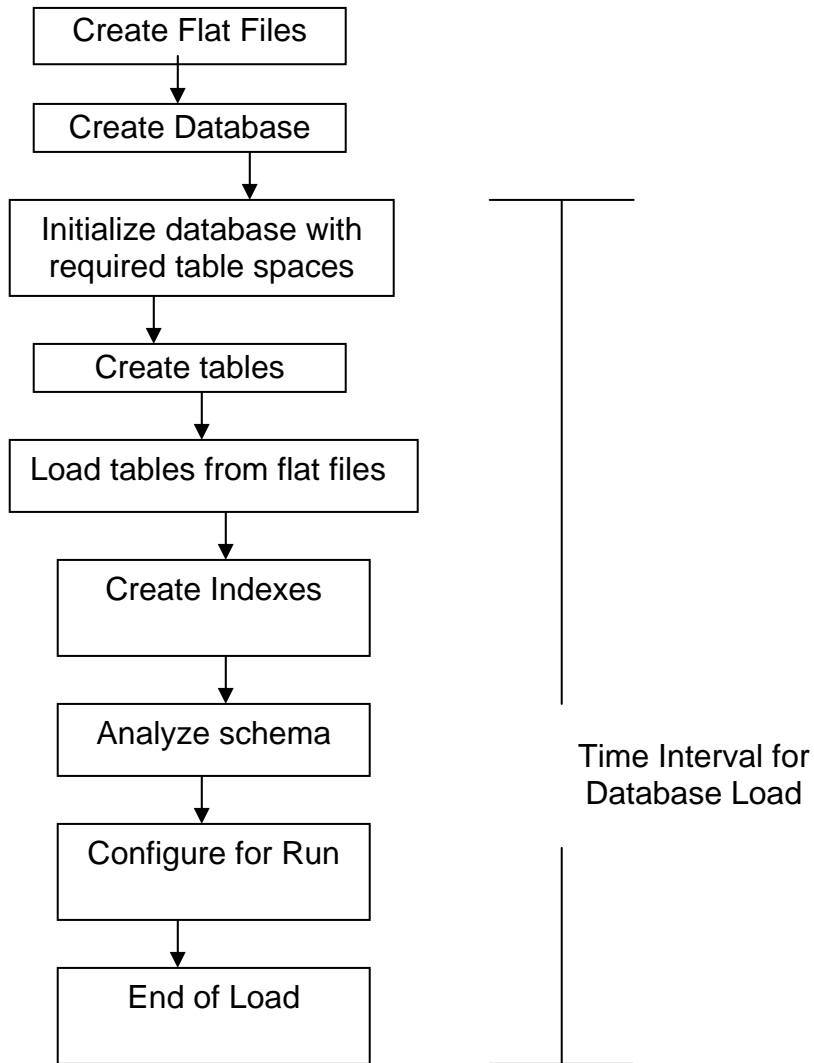
Disk Type	Number of Disks	Space per Disk	Subtotal Disk Space	Total Disk Storage	Data Storage Ratio
36GB 15k	168	33.87GB	5690.16GB		
36GB 15k	4	33.92GB	135.68GB		
				5825.84GB	19.42

Database Load Mechanism Details and Illustration

The details of the database load must be disclosed, including a block diagram illustrating the overall process. Disclosure of the load procedure includes all steps, scripts, input and configuration files required to completely reproduce the test and qualification databases.

DBGEN was used to create the flat files. The figure below describes the load procedure.

Database Load Process



Clause 5: Performance Metrics and Execution Rules

Steps in the Power Test

The details of the steps followed to implement the power test (e.g., system boot, database restart, etc.) must be disclosed.

The following steps were used to implement the power test:

1. Database restart.
2. RF1 update transaction.
3. Stream 00 execution.
4. RF2 update transaction.

Timing Intervals

The timing intervals (see Clause 5.3.6) for each query of the measured set and for both update functions must be reported for the power test.

The power test timing intervals are disclosed in the Numerical Quantities Summary at the beginning of this document.

Number of Streams for the Throughput Test

The number of execution streams used for the throughput test must be disclosed.

Six streams were used for the throughput test.

Start/Finish Time of Each Query Stream

The start time and finish time for each query execution stream must be reported for the throughput test.

The throughput test start time and finish time for each stream are disclosed in the Numerical Quantities Summary at the beginning of this document.

Total Elapsed Time

The total elapsed time of the measurement interval must be reported for the throughput test.

The total elapsed time of the throughput test was 15,704.0 seconds.

Start/Finish Time for Refresh Functions

Start and finish time for each refresh function in the update stream must be reported for the throughput test.

The start and finish time for each refresh function in the update stream are disclosed in the Numerical Quantities Summary at the beginning of this document.

Timing Intervals for Each Query and Each Update

The timing intervals (see Clause 5.3.6) for each query of each stream and for each update function must be reported for the throughput test.

The timing intervals for each query and each update function are contained in the Numerical Quantities Summary disclosed earlier in this document.

Performance Metrics

The computed performance metrics, related numerical quantities and the price performance metric must be reported.

The performance metrics, and the numbers, on which they are based, are contained in the Numerical Quantities section of the Executive Summary.

Reproducibility Method

A description of the method used to determine the reproducibility of the measurement results must be reported. This must include the performance metrics (QppH and QthH) from the reproducibility runs.

Performance results from the first two executions of the TPC-H benchmark indicated the following percent differences for the metrics:

Percentage Differences in Benchmark Executions

Run	QppH@300GB	QthH@300GB	QphH@300GB
1	15,188.9	9,077.9	11,742.8
2	15,674.9	9,164.9	11,985.5
Percent Difference	3.2%	1.0%	2.1%

System Activity Between Run1 and Run2

Any activity on the SUT that takes place between the conclusion of Run1 and the beginning of Run2 must be disclosed.

Database was not re-started between Run1 and Run2.

Clause 6: SUT and Driver Implementation

Driver

A detailed description of how the driver performs its functions must be supplied, including any related source code or scripts. This description should allow an independent reconstruction of the driver.

A single script performs all stream executions. QGEN is used to produce query text.

For each power-test run:

- RF1 refresh updates are submitted to the database
- QGEN-generated queries are submitted in the order defined by Clause 5.3.5.4
- RF2 refresh updates are submitted to the database

Implementation Specific Layer (ISL)

If an implementation-specific layer is used, then a detailed description of how it performs its functions must be supplied, including any related source code or scripts. This description should allow an independent reconstruction of the implementation-specific layer.

A command script was used to control and track the execution of queries. The scripts are contained in Appendix F. Qgen was used to generate the query streams, along with the appropriate substitution values.

Profile-Directed Optimization

If profile-directed optimization as described in Clause 5.2.9 is used, such use must be disclosed. In particular, the procedure and any scripts used to perform the optimization must be disclosed.

Profile-directed optimization was not used.

Clause 7: Pricing

Hardware and Software Used in the Priced System

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

A detailed list of the hardware and software used in the priced system is included in the Executive Summary at the beginning of this document. The price quotes are located in Appendix G.

Total Three Year Price

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.

A detailed price sheet of all the hardware and software used in this configuration, including the 3-year maintenance cost, and total price, is included in the executive summary at the beginning of this document. This purchase qualifies for a 16% discount from Dell Computer Corporation. The price quotes are located in Appendix G.

Availability Date

The committed delivery date for general availability of products used in the price calculations must be reported. When the priced system includes products with different availability dates, the availability date reported on the executive summary must be the date by which all components are committed to being available. The full disclosure report must report availability dates individually for at least each of the categories for which a pricing subtotal must be provided.

All Dell components are available at the time of publication.

Clause 8: Auditor-Related Items

Auditor's Report

The auditor's agency name, address, phone number, and Attestation letter with a brief audit summary report indicating compliance must be included in the full disclosure report. A statement should be included specifying who to contact in order to obtain further information regarding the audit process.

Lorna Livingtree of Performance Metrics audited this implementation of the TPC Benchmark H. Information regarding the audit process may be obtained from:

Performance Metrics, Inc.
P.O Box 984
Klamath, CA 95548
(Phone) (916) 985-1131
(Fax) (916) 985-1185
E-mail: lorna@perfmetrics.com

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

Transaction Processing Performance Council
Presidio of San Francisco
Building 572B (surface)
P.O Box 29920 (Mail)
San Francisco, CA 94129-0920 USA
Telephone: (415) 561-6272
Fax: (415) 561-6120
Info@tpc.org



June 23, 2005

Mr. Nicholas Wakou
Dell Computer Corporation
One Dell Way
Round Rock, TX 78682

I have verified the TPC Benchmark™ H for the following configuration:

Platform: DELL PowerEdge 6800 2 node cluster
Database Manager: Oracle Database 10g Enterprise Edition, Release 2
Operating System: Red Hat Enterprise Linux AS

CPU's	Memory	Total Disks	Qpph @300GB	QthH @300GB	QphH @300GB
8 Intel P4 Xeon MP @ 3.3 Ghz	32 GB	172 @ 36 GB	15,189.9	9,077.9	11,742.8

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

- The database tables were defined with the proper columns, layout and sizes.
- The tested database was correctly scaled and populated for 300 GB using DBGEN version 1.3.0.
- The qualification database layout was identical to the tested database except for the number and size of the files.
- The query text was verified to use only compliant variants and minor modifications.
- The executable query text was generated by QGEN and submitted through Oracle's standard interactive interface. The version of QGEN was 1.3.0.
- The validation of the query text against the qualification database produced compliant results.
- The refresh functions were properly implemented and executed the correct number of inserts and deletes.
- The load timing was properly measured and reported.
- The execution times were correctly measured and reported.

- The performance metrics were correctly computed and reported.
- The repeatability of the measurement was verified.
- The ACID properties were tested and verified.
- Sufficient mirrored log space was present on the tested system.
- The system pricing was checked for major components and maintenance.
- The executive summary pages of the FDR were verified for accuracy.

Auditor's Notes:

Special attention was given to the query text of Query 15. This query used an approved variant (A) and a minor query modification allowed for select list expression aliases. This query was compliant with both requirements.

Sincerely,

A handwritten signature in cursive script that reads "Lorna Livingtree".

Lorna Livingtree
Auditor

Appendix A: Parameter settings

Appendix A

init_build.ora

```
aq_tm_processes          = 0
audit_trail              = false
compatible               = 10.1.0.2
control_files            = (+data/control_001,
+data/control_002)
cpu_count                = 4
db_block_checksum        = false
db_block_size            = 16384
db_file_multiblock_read_count = 128
db_files                 = 500
db_name                  = 10i
db_writer_processes      = 4
dml_locks                = 5000
global_names             = false
instance_name            = pe68a
log_buffer               = 4194304
log_checkpoints_to_alert = true
max_dump_file_size       = unlimited
nls_date_format          = YYYY-MM-DD
open_cursors             = 600
optimizer_mode           = CHOOSE
optimizer_features_enable = 10.2.0.1
parallel_adaptive_multi_user = true
parallel_execution_message_size = 16384
parallel_max_servers     = 128
parallel_min_servers     = 64
pga_aggregate_target     = 12g
processes                = 1000
query_rewrite_integrity  = stale_tolerated
recovery_parallelism     = 8
replication_dependency_tracking = false
sga_target               = 2g
statistics_level         = typical
undo_management          = auto
undo_retention           = 400000
optimizer_index_cost_adj = 450
```

init_RACA.ora

```
instance_number         = 1
thread                  = 1
undo_management         = auto
undo_tablespace         = ts_undo1
cluster_database        = true
cluster_interconnects   = 192.1.2.170
ifile                   =
/home/oracle/10gR2/DB/dbs/init_build.ora
```

init_RACB.ora

```
instance_number         = 2
thread                  = 2
undo_management         = auto
undo_tablespace         = ts_undo2
```

```
cluster_database        = true
cluster_interconnects   = 192.1.2.175
ifile                   =
/home/oracle/10gR2/DB/dbs/init_build.ora
```

init+ASM1.ora

```
#####
#####
# Copyright (c) 1991, 2001, 2002 by Oracle Corporation
#####
#####

#####
# Cluster Database
#####
cluster_database=true

#####
# Diagnostics and Statistics
#####
background_dump_dest=/home/oracle/10gR2/admin/+ASM/b
dump
core_dump_dest=/home/oracle/10gR2/admin/+ASM/cdump
user_dump_dest=/home/oracle/10gR2/admin/+ASM/udump

#####
# Miscellaneous
#####
instance_type=asm

#####
# Pools
```

init+ASM2.ora

```
#####
#####
# Copyright (c) 1991, 2001, 2002 by Oracle Corporation
#####
#####

#####
# Cluster Database
#####
cluster_database=true

#####
# Diagnostics and Statistics
#####
background_dump_dest=/home/oracle/10gR2/admin/+ASM/b
dump
core_dump_dest=/home/oracle/10gR2/admin/+ASM/cdump
user_dump_dest=/home/oracle/10gR2/admin/+ASM/udump
```

alias ltt="ls -ltr |tail -10"

```
#####  
# Miscellaneous  
#####  
instance_type=asm  
  
#####  
# Pools
```

.bashrc

```
# .bashrc  
# User specific aliases and functions  
# Source global definitions  
if [ -f /etc/bashrc ]; then  
    . /etc/bashrc  
fi  
#export ORACLE_SID=pe68a  
export ORACLE_SID=1g_A  
export ASM_SID=+ASM1  
export ORACLE_BASE=/home/oracle/10gR2  
export  
PATH=.:$PATH:$HOME/bin:$O:$O/bin:$O/rdbms/log:$FR  
AME_PATH/bin  
export  
LD_LIBRARY_PATH=/usr/lib:$ORACLE_HOME/lib:$OR  
ACLE_HOME/rdbms/lib  
export ORACLE_HOME=$ORACLE_BASE/DB  
export O=$ORACLE_HOME  
export ORA_CRS_HOME=$ORACLE_BASE/CRS  
export FRAME_PATH=$HOME/frame  
export KIT_DIR=$HOME/kit  
export  
PATH=.:$PATH:$HOME/bin:$O:$O/bin:$O/rdbms/log:$FR  
AME_PATH/bin  
export  
LD_LIBRARY_PATH=/usr/lib:$ORACLE_HOME/lib:$OR  
ACLE_HOME/rdbms/lib  
export  
PATH=$PATH:/mnt/sdb2/usr/bin:/mnt/sdb2/home/oracle/kit/  
utils:/opt/intel/vtune/bin  
alias asmlog="tail -f  
$ORACLE_HOME/rdbms/log/alert_${ASM_SID}.log "  
alias cdb="cd $ORACLE_HOME/dbs"  
alias cdbin="cd $ORACLE_HOME/bin"  
alias cdbuild="cd $HOME/kit/schema/10.0/build"  
alias cdf="cd $FRAME_PATH"  
alias cdframe="cd $FRAME_PATH"  
alias cdfbin="cd $FRAME_PATH/bin"  
alias cdlog="cd $ORACLE_HOME/rdbms/log"  
alias cdkit="cd $KIT_DIR"  
alias cdwork="cd $KIT_DIR/saroj"  
alias cdfr="cd $FRAME_PATH"  
alias cdstats="cd $FRAME_PATH/stats"  
alias cdmq="cd $KIT_DIR/dell_qry"  
alias tlog="tail -f  
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log "  
alias alog="tail -f  
$ORACLE_HOME/rdbms/log/alert_${ASM_SID}.log "
```


Appendix A

Appendix B: Database Build Scripts

Appendix B

build_dg_10gR2.sh

```
export ORACLE_SID=+ASM1
sqlplus /NOLOG <<!
connect /as sysdba
drop diskgroup DATA including contents;
CREATE DISKGROUP DATA External REDUNDANCY
DISK
'/dev/raw/raw1' SIZE 95378M ,
'/dev/raw/raw2' SIZE 95378M ,
'/dev/raw/raw3' SIZE 95378M ,
'/dev/raw/raw4' SIZE 95378M ,
'/dev/raw/raw5' SIZE 95378M ,
'/dev/raw/raw6' SIZE 95378M ,
'/dev/raw/raw7' SIZE 95378M ,
'/dev/raw/raw8' SIZE 95378M ,
'/dev/raw/raw9' SIZE 95378M ,
'/dev/raw/raw10' SIZE 95378M ,
'/dev/raw/raw11' SIZE 95378M ,
'/dev/raw/raw12' SIZE 95378M ,
'/dev/raw/raw13' SIZE 95378M ,
'/dev/raw/raw14' SIZE 95378M;
!

dbcre_10gR2.sh
#!/bin/ksh
echo "database creation"
date;

sqlplus /NOLOG <<!
connect /as sysdba

startup pfile = /home/oracle/10gR2/DB/dbs/init_build.ora
nomount;
create database
controlfile reuse
logfile '+DATA/log_1' size 4096m reuse,
        '+DATA/log_2' size 4096m reuse
datafile '+DATA/sys_1' size 1024m reuse
sysaux datafile '+DATA/sysaux_1' size 1024m reuse
undo tablespace ts_undo1
        datafile '+DATA/undo_1' size 8192m reuse
default temporary tablespace ts_temp
        tempfile '+DATA/temp_1' size 14400m reuse
        extent management local uniform size 10m
maxdatafiles 4000
maxinstances 2;

create undo tablespace ts_undo2 datafile '+DATA/undo_2'
size 8192m reuse;
alter database add logfile thread 2 '+DATA/log_3' size 4096m
reuse,
        '+DATA/log_4' size 4096m reuse;
alter database enable public thread 2;
```

```
set termout off
set echo off
spool /tmp/cat
@?/rdbms/admin/catalog.sql;
@?/rdbms/admin/catproc.sql;
@?/rdbms/admin/catclust.sql;
connect system/manager
@?/sqlplus/admin/pupbld.sql;
spool off
!
echo "end of database creation"
date
```

dapop_10gR2.sh

```
#!/bin/bash
sqlplus /NOLOG <<EOF
connect / as sysdba
drop user tpch cascade;
grant DBA
to tpch identified by tpch;
connect tpch/tpch;
drop directory ff1;
drop directory ff2;
drop directory ff3;
drop directory ff4;
drop directory ff5;
drop directory ff6;
drop directory ff7;
drop directory ff8;
drop directory ff9;
drop directory ff10;
drop directory ff11;
drop directory ff12;
drop directory ff13;
drop directory ff14;
create directory ff1 as /ff1;
create directory ff2 as /ff2;
create directory ff3 as /ff3;
create directory ff4 as /ff4;
create directory ff5 as /ff5;
create directory ff6 as /ff6;
create directory ff7 as /ff7;
create directory ff8 as /ff8;
create directory ff9 as /ff9;
create directory ff10 as /ff10;
create directory ff11 as /ff11;
create directory ff12 as /ff12;
create directory ff13 as /ff13;
create directory ff14 as /ff14;

drop table l_et;
create table l_et(
        l_orderkey      number ,
        l_partkey       number ,
        l_suppkey        number ,
        l_linenumber     number ,
        l_quantity       number ,
        l_extendedprice  number ,
```

Appendix B

```

l_discount      number ,
l_tax           number ,
l_returnflag    char(1) ,
l_linestatus    char(1) ,
l_shipdate      date ,
l_commitdate    date ,
l_commitdate    date ,
l_receiptdate   date ,
l_shipinstruct  char(25) ,
l_shipmode      char(10) ,
l_comment       varchar(44)
)
organization external (
type ORACLE_LOADER
default directory ff1
access parameters
(
    records delimited by newline
    nobadfile
    nologfile
    fields terminated by '|'
    missing field values are null
)
    location (
ff1:'lineitem.tbl.1',
ff2:'lineitem.tbl.2',
ff3:'lineitem.tbl.3',
ff4:'lineitem.tbl.4',
ff5:'lineitem.tbl.5',
ff6:'lineitem.tbl.6',
ff7:'lineitem.tbl.7',
ff8:'lineitem.tbl.8',
ff9:'lineitem.tbl.9',
ff10:'lineitem.tbl.10',
ff11:'lineitem.tbl.11',
ff12:'lineitem.tbl.12',
ff13:'lineitem.tbl.13',
ff14:'lineitem.tbl.14'
))
reject limit unlimited;

drop table o_et;
create table o_et(
    o_orderkey      number ,
    o_custkey       number ,
    o_orderstatus   char(1) ,
    o_totalprice    number ,
    o_orderdate     date ,
    o_orderpriority char(15) ,
    o_clerk         char(15) ,
    o_shippriority  number ,
    o_comment       varchar(79)
)
organization external (
type ORACLE_LOADER
default directory ff1
access parameters
(
    records delimited by newline
    nobadfile
    nologfile
    fields terminated by '|'
    missing field values are null
)
    location (
ff1:'partsupp.tbl.1',
ff2:'partsupp.tbl.2',
ff3:'partsupp.tbl.3',
ff4:'partsupp.tbl.4',
ff5:'partsupp.tbl.5',
ff6:'partsupp.tbl.6',
ff7:'partsupp.tbl.7',
ff8:'partsupp.tbl.8',
ff9:'partsupp.tbl.9',
ff10:'partsupp.tbl.10',
ff11:'partsupp.tbl.11',
ff12:'partsupp.tbl.12',
ff13:'partsupp.tbl.13',
ff14:'partsupp.tbl.14'
))
reject limit unlimited;

nbadfile
nologfile
fields terminated by '|'
missing field values are null
)
    location (
ff1:'orders.tbl.1',
ff2:'orders.tbl.2',
ff3:'orders.tbl.3',
ff4:'orders.tbl.4',
ff5:'orders.tbl.5',
ff6:'orders.tbl.6',
ff7:'orders.tbl.7',
ff8:'orders.tbl.8',
ff9:'orders.tbl.9',
ff10:'orders.tbl.10',
ff11:'orders.tbl.11',
ff12:'orders.tbl.12',
ff13:'orders.tbl.13',
ff14:'orders.tbl.14'
))
reject limit unlimited;

drop table ps_et;
create table ps_et(
    ps_partkey      number ,
    ps_suppkey      number ,
    ps_availqty     number ,
    ps_supplycost   number ,
    ps_comment      varchar(199)
)
organization external (
type ORACLE_LOADER
default directory ff1
access parameters
(
    records delimited by newline
    nobadfile
    nologfile
    fields terminated by '|'
    missing field values are null
)
    location (
ff1:'partsupp.tbl.1',
ff2:'partsupp.tbl.2',
ff3:'partsupp.tbl.3',
ff4:'partsupp.tbl.4',
ff5:'partsupp.tbl.5',
ff6:'partsupp.tbl.6',
ff7:'partsupp.tbl.7',
ff8:'partsupp.tbl.8',
ff9:'partsupp.tbl.9',
ff10:'partsupp.tbl.10',
ff11:'partsupp.tbl.11',
ff12:'partsupp.tbl.12',
ff13:'partsupp.tbl.13',
ff14:'partsupp.tbl.14'
))
reject limit unlimited;

```

Appendix B

```
drop table p_et;
create table p_et(
  p_partkey      number ,
  p_name         varchar(55) ,
  p_mfgr        char(25) ,
  p_brand       char(10) ,
  p_type        varchar(25) ,
  p_size        number ,
  p_container   char(10) ,
  p_retailprice number ,
  p_comment     varchar(23)
)
organization external (
type ORACLE_LOADER
default directory ff1
access parameters
(
  records delimited by newline
  nobadfile
  nologfile
  fields terminated by '|'
  missing field values are null
)
  location (
ff1:'part.tbl.1',
ff2:'part.tbl.2',
ff3:'part.tbl.3',
ff4:'part.tbl.4',
ff5:'part.tbl.5',
ff6:'part.tbl.6',
ff7:'part.tbl.7',
ff8:'part.tbl.8',
ff9:'part.tbl.9',
ff10:'part.tbl.10',
ff11:'part.tbl.11',
ff12:'part.tbl.12',
ff13:'part.tbl.13',
ff14:'part.tbl.14'
))
reject limit unlimited;

drop table c_et;
create table c_et(
  c_custkey      number ,
  c_name         varchar(25) ,
  c_address     varchar(40) ,
  c_nationkey    number ,
  c_phone       char(15) ,
  c_acctbal     number ,
  c_mktsegment  char(10) ,
  c_comment     varchar(117)
)
organization external (
type ORACLE_LOADER
default directory ff1
access parameters
(
  records delimited by newline
  nobadfile
  nologfile
  fields terminated by '|'
  missing field values are null
)
  location (
ff1:'supplier.tbl.1',
ff2:'supplier.tbl.2',
ff3:'supplier.tbl.3',
ff4:'supplier.tbl.4',
ff5:'supplier.tbl.5',
ff6:'supplier.tbl.6',
ff7:'supplier.tbl.7',
ff8:'supplier.tbl.8',
ff9:'supplier.tbl.9',
ff10:'supplier.tbl.10',
ff11:'supplier.tbl.11',
ff12:'supplier.tbl.12',
ff13:'supplier.tbl.13',
ff14:'supplier.tbl.14'
))
reject limit unlimited;

drop table s_et;
create table s_et(
  s_supplekey   number ,
  s_name        char(25) ,
  s_address     varchar(40) ,
  s_nationkey   number ,
  s_phone       char(15) ,
  s_acctbal     number ,
  s_comment     varchar(101)
)
organization external (
type ORACLE_LOADER
default directory ff1
access parameters
(
  records delimited by newline
  nobadfile
  nologfile
  fields terminated by '|'
  missing field values are null
)
  location (
ff1:'customer.tbl.1',
ff2:'customer.tbl.2',
ff3:'customer.tbl.3',
ff4:'customer.tbl.4',
ff5:'customer.tbl.5',
ff6:'customer.tbl.6',
ff7:'customer.tbl.7',
ff8:'customer.tbl.8',
ff9:'customer.tbl.9',
ff10:'customer.tbl.10',
ff11:'customer.tbl.11',
ff12:'customer.tbl.12',
ff13:'customer.tbl.13',
ff14:'customer.tbl.14'
))
reject limit unlimited;
```

Appendix B

```
))
reject limit unlimited;

drop table n_et;
create table n_et(
  n_nationkey    number ,
  n_name         char(25) ,
  n_regionkey    number ,
  n_comment      varchar(152)
)
organization external (
type ORACLE_LOADER
default directory ff1
access parameters
(
  records delimited by newline
  nobadfile
  nologfile
  fields terminated by '|'
  missing field values are null
)
location (
ff14:'nation.tbl'
))
reject limit unlimited;

drop table r_et;
create table r_et(
  r_regionkey    number ,
  r_name         char(25) ,
  r_comment      varchar(152)
)
organization external (
type ORACLE_LOADER
default directory ff1
access parameters
(
  records delimited by newline
  nobadfile
  nologfile
  fields terminated by '|'
  missing field values are null
)
location (
ff14:'region.tbl'
))
reject limit unlimited;

alter table l_et parallel;
alter table o_et parallel;
alter table ps_et parallel;
alter table p_et parallel;
alter table c_et parallel;
alter table s_et parallel;

alter user tpch default tablespace ts_default;
alter user tpch temporary tablespace ts_temp;

@?/rdbms/admin/utlxplan.sql;

set timing on
set echo on
!date
rem drop table orders;
create table orders(
  o_orderdate    ,
  o_orderkey     NOT NULL,
  o_custkey      NOT NULL,
  o_orderpriority ,
  o_shippriority ,
  o_clerk        ,
  o_orderstatus  ,
  o_totalprice   ,
  o_comment
)
pctfree 1
pctused 99
intrans 10
tablespace ts_data
storage (initial 44m freelist groups 2 freelists 99)
compress
parallel
nologging
partition by range (o_orderdate)
subpartition by hash(o_custkey)
subpartitions 16
(
partition ord1 values less than (to_date('1992-01-01','YYYY-MM-DD')),
partition ord2 values less than (to_date('1992-02-01','YYYY-MM-DD')),
partition ord3 values less than (to_date('1992-03-01','YYYY-MM-DD')),
partition ord4 values less than (to_date('1992-04-01','YYYY-MM-DD')),
partition ord5 values less than (to_date('1992-05-01','YYYY-MM-DD')),
partition ord6 values less than (to_date('1992-06-01','YYYY-MM-DD')),
partition ord7 values less than (to_date('1992-07-01','YYYY-MM-DD')),
partition ord8 values less than (to_date('1992-08-01','YYYY-MM-DD')),
partition ord9 values less than (to_date('1992-09-01','YYYY-MM-DD')),
partition ord10 values less than (to_date('1992-10-01','YYYY-MM-DD')),
partition ord11 values less than (to_date('1992-11-01','YYYY-MM-DD')),
partition ord12 values less than (to_date('1992-12-01','YYYY-MM-DD')),
partition ord13 values less than (to_date('1993-01-01','YYYY-MM-DD')),
partition ord14 values less than (to_date('1993-02-01','YYYY-MM-DD')),
partition ord15 values less than (to_date('1993-03-01','YYYY-MM-DD')),
```


Appendix B

```

partition ord74 values less than (to_date('1998-02-01','YYYY-
MM-DD')),
partition ord75 values less than (to_date('1998-03-01','YYYY-
MM-DD')),
partition ord76 values less than (to_date('1998-04-01','YYYY-
MM-DD')),
partition ord77 values less than (to_date('1998-05-01','YYYY-
MM-DD')),
partition ord78 values less than (to_date('1998-06-01','YYYY-
MM-DD')),
partition ord79 values less than (to_date('1998-07-01','YYYY-
MM-DD')),
partition ord80 values less than (to_date('1998-08-01','YYYY-
MM-DD')),
partition ord81 values less than (to_date('1998-09-01','YYYY-
MM-DD')),
partition ord82 values less than (to_date('1998-10-01','YYYY-
MM-DD')),
partition ord83 values less than (to_date('1998-11-01','YYYY-
MM-DD')),
partition ord84 values less than (MAXVALUE))
as select
  o_orderdate      ,
  o_orderkey       ,
  o_custkey        ,
  o_orderpriority  ,
  o_shippriority   ,
  o_clerk          ,
  o_orderstatus    ,
  o_totalprice     ,
  o_comment
from o_et;
!date

rem drop table partsupp;
create table partsupp(
  ps_partkey      NOT NULL,
  ps_suppkey      NOT NULL,
  ps_supplycost   NOT NULL,
  ps_availqty     ,
  ps_comment
)
partition by hash(ps_partkey)
partitions 16
storage (initial 256m freelist groups 2 freelists 99)
parallel
nologging
tablespace ts_data
as select
  ps_partkey      ,
  ps_suppkey      ,
  ps_supplycost   ,
  ps_availqty     ,
  ps_comment
from ps_et;
!date

rem drop table customer;
create table customer(
  c_custkey      NOT NULL,
  c_mktsegment   ,
  c_nationkey    ,
  c_name         ,
  c_address      ,
  c_phone        ,
  c_acctbal      ,
  c_comment
)
pctfree 0
pctused 99
storage (initial 800m freelist groups 2 freelists 99)
compress
parallel
nologging
partition by hash (c_custkey)
partitions 16
tablespace ts_data
as select
  c_custkey      ,
  c_mktsegment   ,
  c_nationkey    ,
  c_name         ,
  c_address      ,
  c_phone        ,
  c_acctbal      ,
  c_comment
from c_et;
!date

rem drop table part;
create table part(
  p_partkey      NOT NULL,
  p_type         ,
  p_size         ,
  p_brand        ,
  p_name         ,
  p_container    ,
  p_mfgr         ,
  p_retailprice  ,
  p_comment
)
pctfree 0
pctused 99
storage (initial 800m freelist groups 2 freelists 99)
compress
parallel
nologging
partition by hash (p_partkey)
partitions 16
tablespace ts_data
as select
  p_partkey      ,
  p_type         ,
  p_size         ,
  p_brand        ,
  p_name         ,
  p_container    ,
  p_mfgr         ,

```

Appendix B

```
p_retailprice ,
p_comment
from p_et;
!date

rem drop table supplier;
create table supplier(
  s_suppkey NOT NULL,
  s_nationkey ,
  s_comment ,
  s_name ,
  s_address ,
  s_phone ,
  s_acctbal
)
```

```
pctfree 0
pctused 99
storage (initial 100m freelist groups 2 freelists 99)
compress
parallel
nologging
partition by hash (s_suppkey)
partitions 16
tablespace ts_data
as select
  s_suppkey ,
  s_nationkey ,
  s_comment ,
  s_name ,
  s_address ,
  s_phone ,
  s_acctbal
from s_et;
!date
```

```
rem drop table nation;
create table nation(
  n_nationkey NOT NULL,
  n_name ,
  n_regionkey ,
  n_comment )
as select * from n_et;
```

```
rem drop table region;
create table region(
  r_regionkey ,
  r_name ,
  r_comment )
as select * from r_et;
!date
```

```
rem drop table lineitem;
create table lineitem(
  l_shipdate ,
  l_orderkey NOT NULL,
  l_discount NOT NULL,
  l_extendedprice NOT NULL,
  l_suppkey NOT NULL,
  l_quantity NOT NULL,
```

```
  l_returnflag ,
  l_partkey NOT NULL,
  l_linestatus ,
  l_tax NOT NULL,
  l_commitdate ,
  l_receiptdate ,
  l_shipmode ,
  l_linenumber NOT NULL,
  l_shipinstruct ,
  l_comment
)
pctfree 1
pctused 99
initrans 10
tablespace ts_data
storage (initial 132m freelist groups 2 freelists 99)
compress
parallel
nologging
partition by range (l_shipdate)
subpartition by hash(l_partkey)
subpartitions 16
(
  partition item1 values less than (to_date('1992-01-01','YYYY-MM-DD')),
  partition item2 values less than (to_date('1992-02-01','YYYY-MM-DD')),
  partition item3 values less than (to_date('1992-03-01','YYYY-MM-DD')),
  partition item4 values less than (to_date('1992-04-01','YYYY-MM-DD')),
  partition item5 values less than (to_date('1992-05-01','YYYY-MM-DD')),
  partition item6 values less than (to_date('1992-06-01','YYYY-MM-DD')),
  partition item7 values less than (to_date('1992-07-01','YYYY-MM-DD')),
  partition item8 values less than (to_date('1992-08-01','YYYY-MM-DD')),
  partition item9 values less than (to_date('1992-09-01','YYYY-MM-DD')),
  partition item10 values less than (to_date('1992-10-01','YYYY-MM-DD')),
  partition item11 values less than (to_date('1992-11-01','YYYY-MM-DD')),
  partition item12 values less than (to_date('1992-12-01','YYYY-MM-DD')),
  partition item13 values less than (to_date('1993-01-01','YYYY-MM-DD')),
  partition item14 values less than (to_date('1993-02-01','YYYY-MM-DD')),
  partition item15 values less than (to_date('1993-03-01','YYYY-MM-DD')),
  partition item16 values less than (to_date('1993-04-01','YYYY-MM-DD')),
  partition item17 values less than (to_date('1993-05-01','YYYY-MM-DD')),
  partition item18 values less than (to_date('1993-06-01','YYYY-MM-DD')),
```


Appendix B

```
partition item77 values less than (to_date('1998-05-01','YYYY-MM-DD')) ,
partition item78 values less than (to_date('1998-06-01','YYYY-MM-DD')) ,
partition item79 values less than (to_date('1998-07-01','YYYY-MM-DD')) ,
partition item80 values less than (to_date('1998-08-01','YYYY-MM-DD')) ,
partition item81 values less than (to_date('1998-09-01','YYYY-MM-DD')) ,
partition item82 values less than (to_date('1998-10-01','YYYY-MM-DD')) ,
partition item83 values less than (to_date('1998-11-01','YYYY-MM-DD')) ,
partition item84 values less than (MAXVALUE))
as select
  l_shipdate      ,
  l_orderkey      ,
  l_discount      ,
  l_extendedprice ,
  l_suppkey       ,
  l_quantity      ,
  l_returnflag    ,
  l_partkey       ,
  l_linestatus    ,
  l_tax           ,
  l_commitdate   ,
  l_receiptdate  ,
  l_shipmode     ,
  l_linenumbr    ,
  l_shipinstruct ,
  l_comment
from l_et;
!date

drop table l_et;
drop table o_et;
drop table ps_et;
drop table p_et;
drop table c_et;
drop table s_et;
drop table n_et;
drop table r_et;

!date

rem drop index i_l_orderkey;
create index i_l_orderkey
on lineitem (l_orderkey) global partition by hash (l_orderkey)
partitions 16
pctfree 2
intrans 10
tablespace ts_data
storage (initial 132m freelist groups 2 freelists 99)
parallel
compute statistics
nologging;
alter index i_l_orderkey allocate extent (size 132m);
alter index i_l_orderkey allocate extent (size 132m);
```

```
alter index i_l_orderkey allocate extent (size 132m);
alter index i_l_orderkey allocate extent (size 132m);

!date

rem drop index i_o_orderkey;
create unique index i_o_orderkey
on orders (o_orderkey) global partition by hash (o_orderkey)
partitions 16
pctfree 2
intrans 10
tablespace ts_data
storage (initial 44m freelist groups 2 freelists 99)
parallel
compute statistics
nologging;
alter index i_o_orderkey allocate extent (size 44m);
alter index i_o_orderkey allocate extent (size 44m);
alter index i_o_orderkey allocate extent (size 44m);
alter index i_o_orderkey allocate extent (size 44m);

!date

rem drop index i_c_custkey;
create unique index i_c_custkey
on customer (c_custkey) global partition by hash (c_custkey)
partitions 16
pctfree 2
intrans 10
tablespace ts_data
storage (initial 10m freelist groups 2 freelists 99)
parallel
compute statistics
nologging;

create unique index i_ps_partkey_suppkey
on partsupp (ps_partkey,ps_suppkey) global partition by hash
(ps_partkey)
partitions 16
pctfree 2
intrans 10
tablespace ts_data
storage (freelist groups 2 freelists 99)
parallel
compute statistics
nologging;

!date
set timing on
execute dbms_stats.gather_schema_stats('TPCH' ,
estimate_percent => 1, degree =>
  16 , granularity => 'GLOBAL' );
connect / as sysdba
execute dbms_stats.gather_system_stats;
exec dbms_scheduler.disable('GATHER_STATS_JOB');
exec
dbms_scheduler.disable('AUTO_SPACE_ADVISOR_JOB');
exec
dbms_scheduler.disable('AUTO_TASKS_JOB_CLASS');
```

Appendix B

```
alter system switch logfile;
!date
EOF
```

addts.sh

```
sqlplus /NOLOG<<!
connect / as sysdba;
alter tablespace $1 add datafile '$2' size $3 reuse;
!
```

addtts.sh

```
sqlplus /NOLOG<<!
connect / as sysdba;
alter tablespace $1 add tempfile '$2' size $3 reuse;
!
```

tscre_10gR2.sh

```
#!/bin/ksh
echo "START: tablespace creation"
date;
sqlplus /NOLOG <<!
connect /as sysdba
create tablespace ts_default
datafile '+DATA/default_1' size 2048m reuse
extent management local autoallocate;
create tablespace ts_data
nologging
datafile '+DATA/data_1' size 30720m reuse
extent management local autoallocate;
!
i=1
while [ $i -lt 16 ]
do
    i=`expr $i + 1`
    addts.sh ts_data +DATA/data_`i` 30720m & done
wait;
i=1
while [ $i -lt 18 ]
do
    i=`expr $i + 1`
    addtts.sh ts_temp +DATA/temp_`i` 14400m &
done
wait;
echo "END: tablespace creation"
date;
```

Appendix C: ACID Scripts

Appendix C

a_query.sql

```
Rem
Rem $Header: a_query.sql 06-aug-99.10:51:10 mpoess Exp $
Rem
Rem a_query.sql
Rem
Rem Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
Rem
Rem NAME
Rem a_query.sql - <one-line expansion of the name>
Rem
rem DESCRIPTION
Rem Performs ACID Query for TPC-D benchmark.
Rem Asks user to input values for o_key
Rem The range of okey is 1 to 600000
Rem
=====
=====
Rem
Rem Usage: sqlplus tpcd/tpcd @a_query <o_key>
Rem
Rem
Rem MODIFIED (MM/DD/YY)
Rem mpoess 08/06/99 - Creation
Rem mpoess 08/06/99 - Created
Rem

set serverout on;

select
'BEFORE ACID QUERY' as STAGE,
substr(TO_CHAR(sysdate,'YYYY-MM-DD
HH:MI:SS'),1,20) as CURRENT_TIME
from dual;

select SUM(trunc(trunc(l_extendedprice * (1-l_discount),2) *
(1+l_tax),2)) AS RESULT
from lineitem
where l_orderkey = &&1;

select
'AFTER ACID QUERY' as STAGE,
substr(TO_CHAR(sysdate,'YYYY-MM-DD
HH:MI:SS'),1,20) as CURRENT_TIME
from dual;

exit;

a_query2.sql
Rem
Rem $Header: aquery2.sql 07-aug-99.23:54:47 mpoess Exp $
Rem
Rem aquery2.sql
Rem
```

```
Rem Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
Rem
Rem NAME
Rem aquery2.sql - <one-line expansion of the name>
Rem
Rem DESCRIPTION
Rem Performs query on PARTSUPP for TPC-D
benchmark
Rem Isolation Test 5.
Rem Asks user to input values for ps_partkey and
ps_suppkey
Rem The range for ps_partkey is 1 to 20000
Rem The range for ps_suppkey is 1 to 1000
Rem A valid combination is 46 and 47
Rem Usage: sqlplus tpcd/tpcd @a_query2 <ps_partkey>
<ps_suppkey>
Rem
Rem MODIFIED (MM/DD/YY)
Rem mpoess 08/07/99 - Creation
Rem mpoess 08/07/99 - Created
Rem
rem DESCRIPTION
rem Performs query on PARTSUPP for TPC-D
benchmark
rem Isolation Test 5.
rem Asks user to input values for ps_partkey and
ps_suppkey
rem The range for ps_partkey is 1 to 20000
rem The range for ps_suppkey is 1 to 1000
rem A valid combination is 46 and 47

set serverout on;

select
'BEFORE PARTSUPP QUERY' as STAGE,
substr(TO_CHAR(sysdate,'YYYY-MM-DD
HH:MI:SS'),1,20) as CURRENT_TIME
from dual;

select *
from partsupp
where ps_partkey = &&1
and ps_suppkey = &&2;

select
'AFTER PARTSUPP QUERY' as STAGE,
substr(TO_CHAR(sysdate,'YYYY-MM-DD
HH:MI:SS'),1,20) as CURRENT_TIME
from dual;

exit;

atom.sh
#!/bin/ksh
#
# $Header: atom.sh 08-aug-99.13:48:02 mpoess Exp $
#
# atom.sh
```

Appendix C

```
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
# atom.sh - <one-line expansion of the name>
#
# DESCRIPTION
# Performs atomicity tests.
# Usage: atom.sh [-n iter] [-p prog] [-u usr/pswd] -h
#
# Options: See usage below
#
# NOTES
# <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
# mpoess 08/08/99 - Creation
# mpoess 08/08/99 - Creation
#

.$KIT_DIR/env

OH=$ORACLE_HOME
# ACID_DIR=$TPCD_KIT_DIR/audit set in env
OUT_DIR=$ACID_OUT
DURA_DIR=$ACID_DIR/dura

usage() {
    echo ""
    echo "Usage: $0 [-n iter] [-p prog] [-u usr/pswd] -h"
    echo ""
    echo "-n iter : number of iterations, default is 100"
    echo "-p prog : program to run, default is atranspl.ott"
    echo "-u usr/pswd : user/password combo for database
access, default is tpcd/tpcd"
    echo "-h : print this usage summary"
    exit 1;
}

ITER=3
SF=1
PROG=$KIT_DIR/utills/atranspl
OUT=${OUT_DIR}/atom
USER=${DATABASE_USER}

set -- `getopt "n:p:u:h" "$@"` || usage

while :
do
    case "$1" in
    -n) shift; ITER=$1;;
    -p) shift; PROG=$1;;
    -u) shift; USER=$1;;
    -h) usage; exit 0;;
    --) break;;
    esac
    shift
done
```

```
done

echo "Starting Atomicity Test at `date`..."
echo ""
echo "Performing $ITER ACID transactions with COMMIT"
echo ""

$KIT_DIR/utills/randkey $ITER $SF u$USER | $PROG 1 1 1
0 u$USER > ${OUT}c 2>&1

echo "ACID transactions with COMMIT ended. Output in
${OUT}c"
echo ""
echo "Performing $ITER ACID transactions with
ROLLBACK"
echo ""

$KIT_DIR/utills/randkey $ITER $SF u$USER | $PROG 1 1 0
0 u$USER > ${OUT}r 2>&1

echo "ACID transactions with ROLLBACK ended. Output in
${OUT}r"
echo ""
echo "Ending Atomicity Test at `date`..."
```

atranspl.c

```
/* Copyright (c) 2001, 2002, Oracle Corporation. All rights
reserved. */

/*
NAME
    atranspl.c - <one-line expansion of the name>

DESCRIPTION
    TPC-HR benchmark ACID transaction driver, OCI version
8

NOTES
    <other useful comments, qualifications, etc.>

MODIFIED (MM/DD/YY)
mpoess 10/23/02 - mpoess_update_from_visa
mpoess 10/17/01 - add parameter in ACIDinit
mpoess 02/22/01 - enlarge timing array
mpoess 01/04/01 - Creation

*/

#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

#include "atranspl.h"

/* Declare error handling functions */
```

Appendix C

```
double gettime();
void sql_error();
void usage();
void ACIDinit();
void ACIDexit();
int atoi();
void srand48();
long lrand48();

/* declarations for ORDERS */

int o_key = 0;
double o_tprice = 0.0;
double o_newtprice = 0.0;

/* declarations for LINEITEM */

int l_key = 0;
int l_pkey = 0;
int l_skey = 0;

int l_quan = 0;
int l_newquan = 0;
double l_eprice = 0.0;
double l_neweprice = 0.0;
double l_disc = 0.0;
double l_tax = 0.0;

sb2 l_npricei;

/* other declarations */

int delta = 0;
double rprice;
double cost;

int proc_no = 1; /* process number, global */
int num_streams = 1; /* number of transaction streams */
/*
int trig = 0; /* Trigger Time */
int slp = 0; /* Sleep Time */

int logfile; /* fdes for logfile for durability (optional) */
/*
int outfile = 1; /* output file (optional) */
#ifdef LINUX
FILE *infile; /* input file (optional) */
#else
FILE *infile = stdin; /* input file (optional) */
/* in the format of <o_key> <delta> */
#endif
char lname[UNAME_LEN]; /* username/passwd combo */
/*
char *passwd; /* pointer to password */
/*
char buf[WRITE_BUF_LEN]; /* buffer to write */
/*
```

```
unsigned flag = (unsigned) 0; /* flag to store all sorts of
options */

#define INFILE 0x01u
#define OUTFILE 0x02u
#define LOGFILE 0x04u
#define COMMIT 0x08u
#define DELTA 0x10u

double tr_end = 0.0; /* transaction end time */
double tr_start = 0.0; /* transaction start time */

int num_iter = 0; /* number of iterations */

time_t curr_time; /* Current Time */

/* OCI handles */

OCIEnv *tpcenv = NULL;
OCIServer *tpcsrv = NULL;
OCIError *errhp = NULL;
OCISvcCtx *tpcsvc = NULL;
OCISession *tpcusr = NULL;
OCIStmt *curi = NULL;
OCIStmt *curr = NULL;
OCIStmt *cure1 = NULL;
OCIStmt *cure2 = NULL;

/* OCI bind handles */

#ifdef NOLKEY
OCIBind *l_keyi_bp = NULL;
OCIBind *o_keyi_bp = NULL;
#endif /* NOLKEY */

OCIBind *l_key_bp = NULL;
OCIBind *o_key_bp = NULL;
OCIBind *delta_bp = NULL;
OCIBind *l_pkey_bp = NULL;
OCIBind *l_skey_bp = NULL;
OCIBind *l_quan_bp = NULL;
OCIBind *l_newquan_bp = NULL;
OCIBind *l_tax_bp = NULL;
OCIBind *l_disc_bp = NULL;
OCIBind *l_eprice_bp = NULL;
OCIBind *l_neweprice_bp = NULL;
OCIBind *o_tprice_bp = NULL;
OCIBind *o_newtprice_bp = NULL;
OCIBind *rprice_bp = NULL;
OCIBind *cost_bp = NULL;

OCIBind *l_neweprice1_bp = NULL;
OCIBind *l_newquan1_bp = NULL;
OCIBind *o_key1_bp = NULL;
OCIBind *l_key1_bp = NULL;

OCIBind *o_newtprice2_bp = NULL;
OCIBind *o_key2_bp = NULL;
```

Appendix C

```

sword status = OCI_SUCCESS; /* OCI return value */

char sqlstmt[1024];

/* usage: prints the usage of the program */

void usage()
{
    fprintf(stderr, "\nUsage: atrans.o[st]t <proc_no>
<num_streams> <commit> <delta>[i<pathname for input>]
[o<pathname for output>] [d<pathname for durability file>]
[u<uid/passwd>] \n\n");

    fprintf(stderr, "  proc_no    :the process number within this
ACID\n");
    fprintf(stderr, "  num_streams :the total number of ACID
transaction streams\n");
    fprintf(stderr, "  commit     :1 to commit transaction, abort
otherwise\n");
    fprintf(stderr, "  delta       :1 to generate new random delta,
otherwise obtain delta from input\n");
    fprintf(stderr, "  OPTIONAL PARAMETERS:\n");
    fprintf(stderr, "  i<pathname for input>   :full path name
for input file - default is stdin\n");
    fprintf(stderr, "  o<pathname for output>   :full path name
for output file - default is stdout\n");
    fprintf(stderr, "  d<pathname for durability> :full path name
for durability success file - must specify for durability test\n");
    fprintf(stderr, "  u<uid/passwd>
:Username/Password string - default is tcpd/tcpd\n");
    fprintf(stderr, "  t<trigger>          :Trigger Time - sleep
<trigger> seconds before start\n");
    fprintf(stderr, "  s<sleep>          :Sleep Time - sleep
<sleep> seconds before commit or rollback\n");
    exit(-1);
}

void ACIDexit() {
    OCILogoff(tpcsvc, errhp);
    OCIhfree(tpcenv, OCI_HTYPE_STMT);
    OCIhfree(tpcsvc, OCI_HTYPE_SVCCTX);
    OCIhfree(tpcsrv, OCI_HTYPE_SERVER);
    OCIhfree(tpcusr, OCI_HTYPE_SESSION);
}

/* type: 0 if environment handle is passed, 1 if error handle is
passwd */

void sql_error(errhp, status, type)
    OCIError *errhp;
    sword status;
    sword type;
{
    char msg[2048];
    ub4 errcode;
    ub4 msglen;
    int i, j;

    switch(status) {
    case OCI_SUCCESS_WITH_INFO:
        fprintf(stderr, "Error: Statement returned with info.\n");
        if (type)
            (void) OCIErrorGet(errhp, 1, NULL, (sb4*) &errcode,
(text*) msg,
                2048, OCI_HTYPE_ERROR);
        else
            (void) OCIErrorGet(errhp, 1, NULL, (sb4*) &errcode,
(text*) msg,
                2048, OCI_HTYPE_ENV);
        fprintf(stderr, "%s\n", msg);
        break;
    case OCI_ERROR:
        fprintf(stderr, "Error: OCI call error.\n");
        if (type)
            (void) OCIErrorGet(errhp, 1, NULL, (sb4 *) &errcode,
(text*) msg,
                2048, OCI_HTYPE_ERROR);
        else
            (void) OCIErrorGet(errhp, 1, NULL, (sb4 *) &errcode,
(text*) msg,
                2048, OCI_HTYPE_ENV);
        fprintf(stderr, "%s\n", msg);
        break;
    case OCI_INVALID_HANDLE:
        fprintf(stderr, "Error: Invalid Handle.\n");
        if (type)
            (void) OCIErrorGet(errhp, 1, NULL, (sb4 *) &errcode,
(text*) msg,
                2048, OCI_HTYPE_ERROR);
        else
            (void) OCIErrorGet(errhp, 1, NULL, (sb4 *) &errcode,
(text*) msg,
                2048, OCI_HTYPE_ENV);
        fprintf(stderr, "%s\n", msg);
        break;
    }
    /* Rollback just in case */
    (void) OCITransRollback(tpcsvc, errhp, OCI_DEFAULT);

    fprintf(stderr, "Exiting Oracle...\n");
    fflush(stderr);

    ACIDexit();

    exit(1);
}

#ifdef LINUX
int main(argc, argv)
#else

```


Appendix C

```
void main(argc,argv)
#ifdef
    int argc;
    char *argv[];
{
    int i;
    char line[64];
    ub4 errcode;
    char msg[2048];
    int need_commit = 0;

    /* Initialize some variables */
#ifdef LINUX
    infile=fopen("/dev/stdin","r");
#endif
    strcpy((char *) lname, "tpcd/tpcd");

    if ((argc > 10) || (argc < 5)) {
        usage();
    }

    /* argv[1] -- Process Number */

    proc_no = atoi(argv[1]);

    /* argv[2] -- Number of Streams */

    num_streams = atoi(argv[2]);

    /* argv[3] -- Commit? */

    if (atoi(argv[3]) == 1)
        BIS(flag, COMMIT);

    /* argv[4] -- Delta? */

    if (atoi(argv[4]) == 1)
        BIS(flag, DELTA);

    /* Process optional parameters */

    argc -= 4;
    argv += 4;

    while(--argc) {
        ++argv;
        switch(argv[0][0]) {
            case 'u':
                strncpy((char *) lname, ++(argv[0]), UNAME_LEN);
                if (strchr((char *) lname, '/') == NULL) {
                    fprintf(stderr, "Login name must be in the format of
userid/passwd\n");
                    usage();
                    exit(-1);
                }
                break;
            case 'i':
                if ((infile = fopen(++(argv[0]), "r")) == NULL) {
                    fprintf(stderr, "Cannot open input file %s\n", argv[0]);
                    fprintf(stderr, "%s\n", strerror(errno));
                    exit(-1);
                }
                BIS(flag, INFILE);
                break;
            case 'o':
                if ((outfile = open(++(argv[0]), (O_RDWR | O_SYNC |
O_CREAT), S_IRWXU)) == -1) {
                    fprintf(stderr, "Cannot open output file %s\n", argv[0]);
                    fprintf(stderr, "%s\n", strerror(errno));
                    exit(-1);
                }
                BIS(flag, OUTFILE);
                break;
            case 'd':
                if ((logfile = open(++(argv[0]), (O_RDWR | O_SYNC |
O_CREAT), S_IRWXU)) == -1) {
                    fprintf(stderr, "Cannot open durability success file %s\n",
argv[0]);
                    fprintf(stderr, "%s\n", strerror(errno));
                    exit(-1);
                }
                BIS(flag, LOGFILE);
                break;
            case 'b':
                num_iter = atoi(++(argv[0]));
                break;
            case 't':
                trig = atoi(++(argv[0]));
                break;
            case 's':
                slp = atoi(++(argv[0]));
                break;
            default:
                fprintf(stderr, "Unknown argument %s\n", argv[0]);
                usage();
                break;
        }
    }

    FPRTF(outfile, "-----\n");

    /* Initialize the cursors etc. */

    (void) ACIDinit();

    /* sleep for some time (triggering) */

    sleep(trig);

    /* start doing the ACID transactions */

    tr_start = gettime();

    /* The number of iteration we will run depends on the
number of */
    /* input lines */
}
```

Appendix C

```

while (fgets(line, 64, infile) != NULL) {
#ifdef NOLKEY
    sscanf(line, "%d %d\n", &o_key, &delta);

    /* Obtain l_key from l_key query */

    OCIsexec(tpcsvc,curi,errhp,1);

    /* l_key is the highest l_linenumber available. We need to
pick */
    /* at random a number between 1..l_key. */

    l_key = (int) ((lrand48() % l_key) + 1);
#else
    sscanf(line, "%d %d %d\n", &o_key, &l_key, &delta);
#endif /* NOLKEY */

    /* Generate delta if necessary */

    if (BIT(flag, DELTA))
        delta = (int) (floor((drand48() * 100)) + 1);

    /* Now, we are ready to run the ACID transaction. */

    curr_time = time(NULL);

    FPRTF2(outfile, "Starting ACID transaction %d at %s...\n",
(++num_iter),
        ctime(&curr_time));

    FPRTF1(outfile, "o_key: %d\n", (int) o_key);
    FPRTF1(outfile, "l_key: %d\n", (int) l_key);
    FPRTF1(outfile, "delta: %d\n", (int) delta);

    OCIsexec(tpcsvc,curr,errhp,1);

    curr_time = time(NULL);

    if (!BIT(flag, LOGFILE)) {
        FPRTF1(outfile, "BEFORE COMMIT/ROLLBACK
TRANSACTION at %s\n", ctime(&curr_time));
        FPRTF1(outfile, "l_extendedprice: %.2f\n", l_епrice);
        FPRTF1(outfile, "l_quantity: %d\n", (int) l_quan);
        FPRTF1(outfile, "o_totalprice: %.2f\n\n", o_tprice);
    }

    FPRTF1(outfile, "Sleep %d seconds before
COMMIT/ROLLBACK...\n\n", slp);
    sleep(slp);

    /* Shall we commit? */

    if (BIT(flag, COMMIT)) {
        need_commit = 1;
        while (need_commit) {
            OCIrol(tpcsvc,errhp);
            OCIsexec(tpcsvc,curr,errhp,1);
        } else {
            need_commit = 0;
            curr_time = time(NULL);
            FPRTF2(outfile, "ACID Transaction iteration %d
COMMITTED at %s\n",
                num_iter, ctime(&curr_time));
        }
    } else {
        OCIrol(tpcsvc,errhp);
        curr_time = time(NULL);
        FPRTF2(outfile, "ACID Transaction iteration %d
ROLLBACK at %s\n",
            num_iter, ctime(&curr_time));
    }

    /* Report all results to outfile and if necessary, to success
file. */

    /* Report initial and new values for o_totalprice,
l_extendedprice, */
    /* l_quantity. */

    /*
curr_time = time(NULL);
FPRTF1(outfile, "Transaction Completed at %s\n",
ctime(&curr_time));
*/

    /* Get the values in LINEITEM and ORDERS after the
transaction */

    if (BIT(flag, LOGFILE)) {
        FPRTF1(logfile, "p_key: %d\n", (int) l_pkey);
        FPRTF1(logfile, "s_key: %d\n", (int) l_skey);
        FPRTF1(logfile, "o_key: %d\n", (int) o_key);
        FPRTF1(logfile, "l_key: %d\n", (int) l_key);
        FPRTF1(logfile, "delta: %d\n", (int) delta);
        FPRTF1(logfile, "Transaction Completed at %s\n",
ctime(&curr_time));
        FPRTF(logfile, "-----
\n");
    } else {
        OCIsexec(tpcsvc,cure1,errhp,1);
        OCIsexec(tpcsvc,cure2,errhp,1);

        FPRTF(outfile, "AFTER TRANSACTION:\n");
        FPRTF1(outfile, "l_extendedprice: %.2f\n", l_newepprice);
        FPRTF1(outfile, "l_quantity: %d\n", (int) l_newquan);
        FPRTF1(outfile, "o_totalprice: %.2f\n\n", o_newtprice);
        FPRTF1(outfile, "l_tax: %.2f\n", l_tax);
        FPRTF1(outfile, "l_discount: %.2f\n", l_disc);
        FPRTF1(outfile, "rprice: %.2f\n", rprice);
        FPRTF1(outfile, "cost: %.2f\n", cost);
        FPRTF(outfile, "-----
\n");
    }
}

```

Appendix C

```

    }
}

tr_end = gettimeofday();

if (!BIT(flag, LOGFILE)) {
    FPRTF1(outfile, "Start Time: %.2f\n", tr_start);
    FPRTF1(outfile, "End Time: %.2f\n", tr_end);
    FPRTF1(outfile, "Elapsed Time: %.2f\n", (tr_end -
tr_start));
    FPRTF1(outfile, "Transaction Count: %d\n", num_iter);
    FPRTF1(outfile, "Transaction Rate: %.2f\n",
num_iter/(tr_end - tr_start));
} else {
    FPRTF1(logfile, "Start Time: %.2f\n", tr_start);
    FPRTF1(logfile, "End Time: %.2f\n", tr_end);
    FPRTF1(logfile, "Elapsed Time: %.2f\n", (tr_end -
tr_start));
    FPRTF1(logfile, "Transaction Count: %d\n", num_iter);
}

/* Disconnect from ORACLE. */

if (BIT(flag, INFILE))
    fclose(infile);
if (BIT(flag, OUTFILE))
    close(outfile);
if (BIT(flag, LOGFILE))
    close(logfile);

ACIDexit();

exit(0);
}

void ACIDinit()
{

/* run random seed */

srand48(getpid());

/* Connect to ORACLE. Program will call sql_error()
if an error occurs in connecting to the default database. */

(void) OCIInitialize(OCI_DEFAULT, (dvoid *)0, 0, 0, 0);
if ((status = OCIEnvInit((OCIEnv
**)&tpcenv, OCI_DEFAULT, 0, (dvoid **)0)) !=
OCI_SUCCESS)
    sql_error(tpcenv, status, 0);

OCIhalloc(tpcenv, &errhp, OCI_HTYPE_ERROR);
OCIhalloc(tpcenv, &curi, OCI_HTYPE_STMT);
OCIhalloc(tpcenv, &curr, OCI_HTYPE_STMT);
OCIhalloc(tpcenv, &cure1, OCI_HTYPE_STMT);
OCIhalloc(tpcenv, &cure2, OCI_HTYPE_STMT);
OCIhalloc(tpcenv, &tpcsvc, OCI_HTYPE_SVCCTX);
OCIhalloc(tpcenv, &tpcsrv, OCI_HTYPE_SERVER);

OCIhalloc(tpcenv, &tpcusr, OCI_HTYPE_SESSION);

/* Disables auto commit */
/*
if (ocof(&tpclda)) {
    sql_error(&tpclda, &tpclda);
    ologof(&tpclda);
    exit(-1);
}
*/

/* get username and password */

passwd = strchr(lname, '/');
*passwd = '\0';
passwd++;

if ((status = OCIServerAttach(tpcsrv, errhp, (text
*)0, 0, OCI_DEFAULT)) != OCI_SUCCESS)
    sql_error(errhp, status, 1);

OCIaset(tpcsvc, OCI_HTYPE_SVCCTX, tpcsrv, 0, OCI_ATTR
_SERVER, errhp);

OCIaset(tpcusr, OCI_HTYPE_SESSION, lname, strlen(lname),
OCI_ATTR_USERNAME,
errhp);

OCIaset(tpcusr, OCI_HTYPE_SESSION, passwd, strlen(passw
d), OCI_ATTR_PASSWORD,
errhp);

if ((status = OCISessionBegin(tpcsvc, errhp, tpcusr,
OCI_CRED_RDBMS,
OCI_DEFAULT)) != OCI_SUCCESS)
    sql_error(errhp, status, 1);

OCIaset(tpcsvc, OCI_HTYPE_SVCCTX, tpcusr, 0, OCI_ATTR
_SESSION, errhp);

/* Enable session parallel dml */

sprintf((char *) sqlstmt, PDMLTXT);
OCISstmtPrepare(curi, errhp, (text *)sqlstmt, strlen((char
*)sqlstmt),
OCI_NTV_SYNTAX, OCI_DEFAULT);
OCIsexec(tpcsvc, curi, errhp, 1);

/* Enable session parallel ddl */

/*sprintf((char *) sqlstmt, PDDLTX);
OCISstmtPrepare(curi, errhp, (text *)sqlstmt, strlen((char
*)sqlstmt),
OCI_NTV_SYNTAX, OCI_DEFAULT);
OCIsexec(tpcsvc, curi, errhp, 1);*/

```

Appendix C

```
/* Make session serializable */

sprintf((char *) sqlstmt, ISOTXT);
OCIStmtPrepare(curi,errhp,(text *)sqlstmt,strlen((char
*)sqlstmt),
    OCI_NTV_SYNTAX,OCI_DEFAULT);
OCIExec(tcpsvc,curi,errhp,1);

/* Set optimizer_index_cost_adj = 25 */

sprintf((char *) sqlstmt, OICATXT);
OCIStmtPrepare(curi,errhp,(text *)sqlstmt,strlen((char
*)sqlstmt),
    OCI_NTV_SYNTAX,OCI_DEFAULT);
OCIExec(tcpsvc,curi,errhp,1);

curr_time = time(NULL);
printf("\nConnected to ORACLE as user: %s at %s\n\n",
lname, ctime(&curr_time));

#ifdef NOLKEY
/* Open and Parse cursor for query to choose determine
l_key. */
/* Binds l_key to :l_key. */

sprintf((char *) sqlstmt,SQLTXT1);
OCIStmtPrepare(curi,errhp,sqlstmt,strlen((char
*)sqlstmt),OCI_NTV_SYNTAX,OCI_DEFAULT);

OCIbname(curi,&l_keyi_bp,errhp,":l_key",ADR(l_key),SIZ(
l_key),SQLT_INT);

OCIbname(curi,&o_keyi_bp,errhp,":o_key",ADR(o_key),SI
Z(o_key),SQLT_INT);

#endif /* NOLKEY */

/* Open and Parse cursor for the ACID transaction. */

sprintf((char *) sqlstmt,SQLTXT2);
OCIStmtPrepare(curr,errhp,(text *)sqlstmt,strlen((char
*)sqlstmt),
    OCI_NTV_SYNTAX,OCI_DEFAULT);

/* bind variables */

OCIbname(curr,l_key_bp,errhp,":l_key",ADR(l_key),SIZ(l_
key),SQLT_INT);

OCIbname(curr,o_key_bp,errhp,":o_key",ADR(o_key),SIZ(
o_key),SQLT_INT);

OCIbname(curr,delta_bp,errhp,":delta",ADR(delta),SIZ(delt
a),SQLT_INT);

OCIbname(curr,l_pkey_bp,errhp,":l_pkey",ADR(l_pkey),SI
Z(l_pkey),SQLT_INT);

OCIbname(curr,l_skey_bp,errhp,":l_skey",ADR(l_skey),SIZ
(l_skey),SQLT_INT);

OCIbname(curr,l_quan_bp,errhp,":l_quan",ADR(l_quan),SI
Z(l_quan),SQLT_INT);

OCIbname(curr,l_newquan_bp,errhp,":l_newquan",ADR(l_n
ewquan),
    SIZ(l_newquan),SQLT_INT);

OCIbname(curr,l_tax_bp,errhp,":l_tax",ADR(l_tax),SIZ(l_t
ax),SQLT_FLT);

OCIbname(curr,l_disc_bp,errhp,":l_disc",ADR(l_disc),SIZ(
l_disc),SQLT_FLT);

OCIbname(curr,l_eprice_bp,errhp,":l_eprice",ADR(l_eprice
),SIZ(l_eprice),
    SQLT_FLT);

OCIbname(curr,l_neweprice_bp,errhp,":l_neweprice",ADR(
l_neweprice),
    SIZ(l_neweprice),SQLT_FLT);

OCIbname(curr,o_tprice_bp,errhp,":o_tprice",ADR(o_tprice
),SIZ(o_tprice),
    SQLT_FLT);

OCIbname(curr,o_newtprice_bp,errhp,":o_newtprice",ADR(
o_newtprice),
    SIZ(o_newtprice), SQLT_FLT);

OCIbname(curr,rprice_bp,errhp,":rprice",ADR(rprice),SIZ(r
price), SQLT_FLT);
OCIbname(curr,cost_bp,errhp,":cost",ADR(cost),SIZ(cost),
SQLT_FLT);

/* Open & Parse cursor for end values query */

sprintf((char *) sqlstmt,SQLTXT3);
OCIStmtPrepare(cure1,errhp,(text *)sqlstmt,strlen((char
*)sqlstmt),
    OCI_NTV_SYNTAX,OCI_DEFAULT);

sprintf((char *) sqlstmt,SQLTXT4);
OCIStmtPrepare(cure2,errhp,(text *)sqlstmt,strlen((char
*)sqlstmt),
    OCI_NTV_SYNTAX,OCI_DEFAULT);

/* bind variables */

OCIbname(cure1,l_neweprice1_bp,errhp,":l_neweprice",AD
R(l_neweprice),
```

Appendix C

```

        SIZ(l_neweprice),SQLT_FLT);

OCIbbname(cure1,l_newquan1_bp,errhp,":l_newquan",ADR(
l_newquan),
        SIZ(l_newquan),SQLT_INT);

OCIbbname(cure1,o_key1_bp,errhp,":o_key",ADR(o_key),SI
Z(o_key),SQLT_INT);

OCIbbname(cure1,l_key1_bp,errhp,":l_key",ADR(l_key),SIZ
(l_key),SQLT_INT);

OCIbbname(cure2,o_newtprice2_bp,errhp,":o_newtprice",AD
R(o_newtprice),
        SIZ(o_newtprice),SQLT_FLT);

OCIbbname(cure2,o_key2_bp,errhp,":o_key",ADR(o_key),SI
Z(o_key),SQLT_INT);

}

```

atranspl.h

/* Copyright (c) 2001, 2002, Oracle Corporation. All rights reserved. */

```

/*
NAME
    atranspl.h - <one-line expansion of the name>

DESCRIPTION

MODIFIED (MM/DD/YY)
mpoess    10/23/02 - mpoess_update_from_visa
mpoess    10/17/01 - add TXT parameter
mpoess    04/09/01 - add hint to find max linenumber
mpoess    01/04/01 - Creation
*/

```

```
#ifndef ATRANSPL_H
```

```
#define ATRANSPL_H
```

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/param.h>
#include <sys/types.h>
#include <time.h>
#include <errno.h>
#include <math.h>

```

```

#include <oratypes.h>
#ifndef OCIDFN
#include <ocidfn.h>
#endif /* OCIDFN */

```

```
#ifndef OCI_ORACLE
```

```

#include <oci.h>
#endif /* OCI_ORACLE */

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

```

```
extern int errno;
```

```

#ifndef NULL
#define NULL 0
#endif

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

```

```

#ifndef DISCARD
#define DISCARD (void)
#endif

```

```

#ifndef sword
#define sword int
#endif

```

```

#ifndef ub1
#define ub1 unsigned char
#endif

```

```

#define UNAME_LEN 64
#define WRITE_BUF_LEN 1024

```

```

#define NA        -1    /* ANSI SQL NULL */
#define VER7      2
#define NOT_SERIALIZABLE 8177 /* ORA-08177:
transaction not serializable */
#define WRITE_BUF_LEN 1024

```

```

#define ADR(object) ((ub1 *)&(object))
#define SIZ(object) ((sword)sizeof(object))
#define BIS(flag,mask) (unsigned) (flag |= (unsigned) mask)
#define BIT(flag,mask) (unsigned) ((unsigned) flag &
(unsigned) mask)

```

```

#define FPRTF(fd,s) \
{ sprintf(buf,s); write(fd, buf, strlen(s)); }
#define FPRTF1(fd,s,p) \
{ sprintf(buf,s,p); write(fd, buf, strlen(buf)); }
#define FPRTF2(fd,s,p1,p2) \
{ sprintf(buf,s,p1,p2); write(fd, buf, strlen(buf)); }

```

```

#define OCIhalloc(envh,hndl,htyp) \
if((status=OCIHandleAlloc((dvoid *)envh,(dvoid **)hndl,htyp,0,(dvoid **)0))!=OCI_SUCCESS) \
    sql_error(envh,status,0); \
else \

```

Appendix C

DISCARD 0

DISCARD 0

```
#define OCIfree(hndl,htyp) \
    if((status=OCIHandleFree((dvoid *)hndl,htyp)) ==
OCI_SUCCESS) \
    fprintf(stderr, "Error freeing handle of type %d\n", htyp)

#define OCIaget(hndl,htyp,attp,size,atyp,errh) \
    if((status=OCIAttrGet((dvoid *)hndl,htyp,(dvoid
*)attp,(dvoid *)size,atyp,errh)) != OCI_SUCCESS) \
    sql_error(errh,status,1); \
    else \
    DISCARD 0

#define OCIaset(hndl,htyp,attp,size,atyp,errh) \
    if((status=OCIAttrSet((dvoid *)hndl,htyp,(dvoid
*)attp,size,atyp,errh)) != OCI_SUCCESS) \
    sql_error(errh,status,1); \
    else \
    DISCARD 0

#define OCIsexec(svch,stmh,errh,iter) \

if((status=OCISmtExecute(svch,stmh,errh,iter,0,NULL,NUL
L,OCI_DEFAULT)) != OCI_SUCCESS) \
    sql_error(errh,status,1); \
    else \
    DISCARD 0

#define
OCIbname(stmh,bindp,errh,sqlvar,progv,progv1,ftype) \
    if((status=OCIBindByName(stmh,&bindp,errh,(text
*)sqlvar,strlen(sqlvar), \
        progvl,progv1,ftype,0,0,0,0,OCI_DEFAULT)) !=
OCI_SUCCESS) \
    sql_error(errh,status,1); \
    else \
    DISCARD 0

#define
OCIbnamei(stmh,bindp,errh,sqlvar,progv,progv1,ftype,indp)
\
    if((status=OCIHandleAlloc((dvoid *)stmh,(dvoid
**)&bindp,OCI_HTYPE_BIND, \
        0,(dvoid **)0))!=OCI_SUCCESS) \
    sql_error(stmh,status,0); \
    if((status=OCIBindByName(stmh,&bindp,errh,(text
*)sqlvar,strlen(sqlvar), \
        progvl,progv1,ftype,indp,0,0,0,0,OCI_DEFAULT))
!= OCI_SUCCESS) \
    sql_error(errh,status,1); \
    else \
    DISCARD 0

#define OCIcon(svcp,errh) \
    if((status=OCITransCommit(svcp,errh,OCI_DEFAULT))
!= OCI_SUCCESS) \
    sql_error(errh,status,1); \
    else \
```

```
#define OCIfrol(svcp,errh) \
    if((status=OCITransRollback(svcp,errh,OCI_DEFAULT))
!= OCI_SUCCESS) \
    sql_error(errh,status,1); \
    else \
    DISCARD 0

#define ISOTXT "alter session set isolation_level =
serializable"
#define PDMLTXT "alter session force parallel dml parallel
(degree 2)"
#define PDDLTX "alter session force parallel ddl parallel
(degree 2)"
#define OICATXT "alter session set
optimizer_index_cost_adj=25"

#define SQLTXT1 "BEGIN SELECT /*+
index(lineitem,i_l_orderkey) */ MAX(l_linenumber) INTO
:l_key FROM lineitem \
WHERE l_orderkey = :o_key; END;"

#define SQLTXT2 "BEGIN d_atrans.doatrans(:l_key, :o_key,
:delta, :l_pkey, \
:l_skey, :l_quan, :l_newquan, :l_tax, :l_disc, :l_eprice,
:l_neweprice, \
:o_tprice, :o_newtprice, :rprice, :cost); END;"

#define SQLTXT3 "BEGIN SELECT l_extendedprice,
l_quantity \
INTO :l_neweprice, :l_newquan \
FROM lineitem \
WHERE l_orderkey = :o_key \
AND l_linenumber = :l_key; END;"

#define SQLTXT4 "BEGIN SELECT o_totalprice INTO
:o_newtprice \
FROM orders \
WHERE o_orderkey = :o_key; END;"

#define SQLTXT5 "BEGIN SELECT l_extendedprice,
l_quantity \
INTO :l_eprice, :l_quan \
FROM lineitem \
WHERE l_orderkey = :o_key \
AND l_linenumber = :l_key; END;"

#define SQLTXT6 "BEGIN SELECT o_totalprice INTO
:o_tprice \
FROM orders \
WHERE o_orderkey = :o_key; END;"

#endif /* ATRANSPL_H */

ckpt.sh
#!/bin/ksh
#
# $Header: ckpt.sh 08-aug-99.17:32:22 mpoess Exp $
```

Appendix C

```
#
# ckpt.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
# ckpt.sh - <one-line expansion of the name>
#
# DESCRIPTION
# <short description of component this file
declares/defines>
#
# NOTES
# <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
# mpoess 08/08/99 - Creation
# mpoess 08/08/99 - Creation
#
. $KIT_DIR/env
sqlplus -s /NOLOG << !

    connect / as sysdba;
    alter system switch logfile;
    alter system switch logfile;
    exit;
!

consist.sh
#!/bin/ksh
#
# $Header: consist.sh 08-aug-99.14:20:51 mpoess Exp $
#
# consist.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
# consist.sh - <one-line expansion of the name>
#
# DESCRIPTION
# Performs consistency tests.
# Usage: consist.sh [-n iter] [-s number of stream] [-p prog]
# [-u usr/pswd] -h
#
# Options: See usage below
#
# NOTES
# <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
# mpoess 08/08/99 - Creation
# mpoess 08/08/99 - Creation
#
. $KIT_DIR/env
```

```
OH=$ORACLE_HOME
# ACID_DIR=$TPCD_KIT_DIR/audit set in env
OUT_DIR=$ACID_OUT

KEY=$OUT_DIR/key$$_
OUTFILE=${OUT_DIR}/consrte
CON1=${OUT_DIR}/conb
CON2=${OUT_DIR}/cona
CHK=${OUT_DIR}/conskcpt

/bin/rm -rf ${KEY}* $CON1 $CON2 $OUTFILE $CHK

trap "/bin/rm -rf ${KEY}*; exit 1" 1 2 3 15

STREAM=${NUM_STREAMS}
let STREAM="$STREAM + 1" # add one for the update
stream
ITER=100
PROG=atranspl
USER=${DATABASE_USER}
CK=10

usage() {
    echo ""
    echo "Usage: $0 [-n iter] [-s number of stream] [-p prog] [-u
usr/pswd] -h"
    echo ""
    echo "-n iter          : number of iterations, default is 100"
    echo "-s number of stream : number of streams, default is 2"
    echo "-p prog           : program to run, default is
atranspl.ott"
    echo "-u usr/pswd      : user/password for database access,
default is tpcd/tpcd"
    echo "-t chkpt        : time after the start of ACID
transaction to perform the checkpoint"
    echo "                default is 10 seconds"
    echo "-h              : print this usage summary"
    exit 1;
}

set -- `getopt "n:p:u:s:h" "$@"` || usage

while :
do
    case "$1" in
-s) shift; STREAM=$1;;
-n) shift; ITER=$1;;
-p) shift; PROG=$1;;
-u) shift; USER=$1;;
-t) shift; CK=$1;;
-h) usage; exit 0;;
--) break;;
esac
    shift
done

if [ $ITER -lt 100 ]
```

Appendix C

```
then
echo "Error: Must at least run 100 iterations!"
echo "Exiting..."
exit 1
fi

if [ $STREAM -lt 2 ]
then
echo "Error: Must at least run 2 streams!"
echo "Exiting..."
exit 1
fi

echo "Starting Consistency Test at `date`..."
echo ""
echo "Generate some keys first"
echo ""

i=0

while [ $i -lt $STREAM ]
do
echo randkey $ITER 1 u$USER
randkey $ITER 1 u$USER > ${KEY}$i
i=`expr $i + 1`
done

echo "Check consistency before Submitting Transactions
`date`"
echo "Check consistency before Submitting Transactions
`date`" >> $CON1

echo "Obtain 10 keys from the each key file to check
consistency"

i=0
while [ $i -lt $STREAM ]
do
KEYS=`head -10 ${KEY}$i | awk '{printf "%d ", $1}`
echo "The 10 Keys for file $i are: $KEYS"
#for j in `head -10 ${KEY}$i | awk '{printf "%d ", $1}`
for j in $KEYS
do
sqlplus $USER @consist $j >> $CON1
echo "-----" >> $CON1
done
i=`expr $i + 1`
done

echo ""
echo "Starting ACID transactions at `date`"
echo ""

i=0

while [ $i -lt $STREAM ]
do
$PROG $i $STREAM 1 0 u${USER} i${KEY}$i
o${OUTFILE}$i s1 &
i=`expr $i + 1`
done

done

echo "Schedule a Checkpoint"
echo "Checkpoint scheduled at $CK seconds after `date`"

(sleep $CK; $ACID_DIR/ckpt.sh) &

wait

echo ""
echo "Ending ACID transactions at `date`"
echo ""

echo "Completed $STREAM transaction streams with $ITER
iterations each"
echo ""

echo "Check consistency after Submitting Transactions
`date`"
echo "Check consistency after Submitting Transactions
`date`" >> $CON2

cat
${ORACLE_HOME}/rdbms/log/alert_${ORACLE_SID}.log
>> $CHK

i=0
while [ $i -lt $STREAM ]
do
KEYS=`head -10 ${KEY}$i | awk '{printf "%d ", $1}`
#for j in `head -10 ${KEY}$i | awk '{printf "%d ", $1}`
echo "The keys to check for consistency after the test from file
$i are:"
echo "$KEYS"
for j in $KEYS
do
sqlplus $USER @consist $j >> $CON2
echo "-----" >> $CON2
done
i=`expr $i + 1`
done

consist.sql
Rem
Rem $Header: consist.sql 08-aug-99.16:59:17 mpoess Exp $
Rem
Rem consist.sql
Rem
Rem Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
Rem
Rem NAME
Rem consist.sql - <one-line expansion of the name>
Rem
Rem DESCRIPTION
Rem Verifies the consistency of TPC-D database using the
Rem consistency condition.
Rem
```


Appendix C

```
Rem Usage: sqlplus tpcd/tpcd @consist
Rem
Rem NOTE
Rem REQUIRES PACKAGES prvtotpt and dbmsotpt
rem
Rem MODIFIED (MM/DD/YY)
Rem mpoess 08/08/99 - Creation
Rem mpoess 08/08/99 - Created
Rem

set verify off
rem set termout on
rem set echo on

REM
REM Get today's date.
REM

select
substr(TO_CHAR(sysdate,'YYYY-MM-DD
HH:MI:SS'),1,20) as CURRENT_TIME
from dual;

set serverout on;

DECLARE
    o_okey    number;
    o_tprice number;
    l_tprice number;
    diff      number;
BEGIN
    select o_totalprice
    into o_tprice
    from orders
    where o_orderkey = &&1;

    select sum(trunc((trunc((l_extendedprice * (1-l_discount)),
2)
    * (1+l_tax)), 2))
    into l_tprice
    from lineitem
    where l_orderkey = &&1;

    diff := l_tprice - o_tprice;

    dbms_output.put_line('O_TOTALPRICE: ' ||
TO_CHAR(trunc(o_tprice,2)));
    dbms_output.put_line('L_TOTALPRICE: ' ||
TO_CHAR(trunc(l_tprice,2)));
    dbms_output.put_line('Difference: ' ||
TO_CHAR(trunc(diff,2)));

END;
.
```

```
spool off
exit

dura.sh
#!/bin/ksh
#
# $Header: dura.sh 08-aug-99.15:21:38 mpoess Exp $
#
# dura.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
#   dura.sh - <one-line expansion of the name>
#
# DESCRIPTION
#   <short description of component this file
declares/defines>
#
# NOTES
#   <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
#   mpoess 08/08/99 - Creation
#   mpoess 08/08/99 - Creation
#

.$KIT_DIR/env

# Create history table

# Count number of entries in the history table

SERVER="ultraperf2"

echo "-----"
echo "Capturing Process information before durability tests
`date`"
rsh $SERVER -n -l spyda ps -ef; date
echo "-----"

echo "-----"
echo "Starting the durability tests `date`"
run_acid.sh &
echo "-----"

sleep 1200

echo "-----"
echo "Collecting user information. `date`"
./cnt_user.sh pswong spyda ultraperf2 > dura/duraucnt 2>&1
echo "-----"

echo "-----"
echo "Capturing Process information while running
Transactions `date`"
rsh $SERVER -n -l spyda ps -ef; date
echo "-----"
```

Appendix C

```

echo "-----"
echo "Capturing disk information on Server: Ultraperf2
`date`"
rsh $SERVER -n -l spyda vxprint -ht ; date
echo "-----"

echo "-----"
echo "Detaching mirror on data disk. `date`"
rsh $SERVER -n -l root "vxplex -v ordr23 det ordr23-01"
echo "-----"

echo "-----"
echo "Capturing Disk information information on Server:
Ultraperf2 `date`"
rsh $SERVER -n -l spyda vxprint -ht ; date
echo "-----"

sleep 120

echo "-----"
echo "Capturing Process information after breaking data
mirror. `date`"
rsh $SERVER -n -l spyda ps -ef; date
echo "-----"

echo "-----"
echo "Detaching mirror on log2 disk. `date`"
rsh $SERVER -n -l root "vxplex -v log2 det log2-01"
echo "-----"

echo "-----"
echo "Capturing Disk information information on Server:
Ultraperf2 `date`"
rsh $SERVER -n -l spyda vxprint -ht ; date
echo "-----"

sleep 120

echo "-----"
echo "Capturing Process information after detaching log
mirror. `date`"
rsh $SERVER -n -l spyda ps -ef; date
echo "-----"

# Power Off

end_acid.sh

#!/bin/ksh
#
# $Header: end_acid.sh 08-aug-99.17:06:20 mpoess Exp $
#
# end_acid.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#

```

```

# NAME
#   end_acid.sh - <one-line expansion of the name>
#
# DESCRIPTION
#   end_cons.sh <pid of the durability run>
#   Options: See usage below
#
# NOTES
#   <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
#   mpoess   08/08/99 - Creation
#   mpoess   08/08/99 - Creation
#

. $KIT_DIR/env

OH=$ORACLE_HOME
# ACID_DIR=$OH/tpcd/audit set in env
OUT_DIR=$ACID_OUT/
DURA_DIR=$ACID_OUT/dura
RUN_ID_FILE=$ACID_DIR/run_id

SHELL_PID=`cat ${DURA_DIR}/shellpid`
ITER=100
STEM=${NUM_STREAMS}
let STEM="$STEM + 1" # add one for the update stream
PROG=${ACID_DIR}/atranspl.ott
IN=${ACID_DIR}/acid_in
DURA=${DURA_DIR}/drate
OUT=${DURA_DIR}/drate
DSMPL=${DURA_DIR}/durasmpl
KEY=${DURA_DIR}/key${SHELL_PID}_
USER=tpch/tpch
TRIG=1
HCNT=duracnta

# get history count

sqlplus $USER @cnt_hist > $DURA_DIR/$HCNT 2>&1

# perform the consistency

i=0
while [ $i -lt $STEM ]
do
  for j in `head -10 ${KEY}${i} | awk '{printf "%d ",$1}`
  do
    sqlplus tpch/tpch @consist $j >> $DURA_DIR/duraconsa
  done
  i=`expr $i + 1`
done

i=0
while [ $i -lt $STEM ]
do
  sample.sh $DURAS${i} > ${DSMPL}${i} 2>&1
  i=`expr $i + 1`
done

```

Appendix C

gettime.c

```
#ifndef RCSID
static char *RCSid =
"$Header: gettime.c 15-jul-99.14:27:44 mpoess Exp $ ";
#endif /* RCSID */

/* Copyright (c) Oracle Corporation 1999. All Rights
Reserved. */

/*

NAME
    gettime.c

DESCRIPTION
    get wall clock time.
    get cpu time.

FUNCTIONS
    get wall clock time.
    get cpu time.

NOTES
    Both routines return time in seconds as a double.
MODIFIED (MM/DD/YY)
    mpoess 07/15/99 - Creation
    mpoess 07/15/99 - Creation

*/

/*
** Options:
**  TIME_W_TIMES:    implement gettime() with times().
**  TIME_W_GETTIME: implement gettime() with
gettimeofday().
**  CPU_W_TIMES:    implement getcpu() with times().
**  CPU_W_GETTRU:   implement getcpu() with
getrusage().
**  GETRU_STATS:    collect getrusage statistics
**  GET_P_STATS:    collect get_process_stats statistics
*/

#define SUN_OS5

#if defined(SUN_OS5)
#define TIME_W_GETTIME
#define CPU_W_TIMES
#undef GETRU_STATS
#undef CPU_W_GETTRU
#endif /* SUN_OS5 */

#if defined(sequent) || defined(SEQ_PSX)
#define GET_P_STATS
#endif /* sequent */

#if defined(aix) || defined(AIXRIOS)
```

```
# define TIME_W_GETTIME
# define CPU_W_TIMES
# define GETRU_STATS
#endif /* AIXRIOS */

#if defined(a_osf) || defined(A_OSF)
# define TIME_W_GETTIME
# define CPU_W_GETTRU
# define GETRU_STATS
#endif /* AIXRIOS */

#if defined(HPUX) || defined(XENIX_386) ||
defined(SYSV_386) || defined(ATT_3B)
# define TIME_W_TIMES
# define CPU_W_TIMES
#endif /* HPUX || XENIX_386 || SYSV_386 */

#if !defined(TIME_W_GETTIME) &&
!defined(TIME_W_TIMES)
# define TIME_W_TIMES
#endif

#if !defined(CPU_W_GETTRU) &&
!defined(CPU_W_TIMES)
# define CPU_W_TIMES
#endif

#ifdef GET_P_STATS
# ifdef GETRU_STATS
# undef GETRU_STATS
# endif
#endif

#if defined(TIME_W_GETTIME) ||
defined(CPU_W_GETTRU) || defined(GETRU_STATS)
# include <sys/time.h>
#endif /* TIME_W_GETTIME || CPU_W_GETTRU ||
GETRU_STATS */

#if defined(CPU_W_GETTRU) || defined(GETRU_STATS)
# include <sys/resource.h>
#endif /* CPU_W_GETTRU || GETRU_STATS */

#if defined(TIME_W_TIMES) || defined(CPU_W_TIMES)
# include <sys/types.h>
# include <sys/times.h>
# include <sys/param.h> /* most systems define HZ here */
#endif /* TIME_W_TIMES or CPU_W_TIMES */

#ifdef GET_P_STATS
# include <sys/types.h>
# include <sys/procstats.h>
#endif /* GET_P_STATS */

# include <stdio.h>

#ifdef GETRU_STATS
struct rusage selfru;
struct rusage kidsru;
```

Appendix C

```
#endif /* GETRU_STATS */

#ifdef GET_P_STATS
struct process_stats selfru;
struct process_stats kidsru;
#endif /* GET_P_STATS */

double gettime ()

{

#ifdef TIME_W_GETTIME
    struct timeval tv;

    (void) gettimeofday (&tv, (struct timezone *) 0);
    return ((double) tv.tv_sec + (1.0e-6 * (double) tv.tv_usec));
#endif /* TIME_W_GETTIME */

#ifdef TIME_W_TIMES
    struct tms buf;

    return ((double) times (&buf) / HZ);
#endif /* TIME_W_TIMES */

}

double getcpu ()

{

#ifdef CPU_W_TIMES
    struct tms buf;

    (void) times (&buf);
    return (((double) buf.tms_utime + (double) buf.tms_stime) /
    HZ);
#endif /* CPU_W_TIMES */

#ifdef CPU_W_GETRU
    struct rusage ru;
    double usecs;

    (void) getrusage (0, &ru);
    usecs = 1.0e-6 * (double) (ru.ru_utime.tv_usec +
    ru.ru_stime.tv_usec);
    return ((double) (ru.ru_utime.tv_sec + ru.ru_stime.tv_sec) +
    usecs);
#endif /* CPU_W_GETRU */

}

getru (fp, kids, config, runname, proc_no)

FILE *fp;
int kids;
char *config;
char *runname;
int proc_no;

{

#ifdef GETRU_STATS
    struct rusage ru;

    fprintf (fp, "%-10.10s %-10.10s %10d %10d ",
    config,runname, proc_no, kids);
    getrusage (kids ? RUSAGE_CHILDREN :
    RUSAGE_SELF, &ru);
    print_ru (fp, &ru);
    fprintf (fp, "\n");
#endif /* GETRU_STATS */

#ifdef GET_P_STATS
    timeval_t tv;
    struct process_stats ru;

    fprintf (fp, "%-10.10s %-10.10s %10d %10d ",
    config,runname, proc_no, kids);
    if (kids)
        get_process_stats (&tv, PS_SELF, (struct process_stats *)
    0, &ru);
    else
        get_process_stats (&tv, PS_SELF, &ru, (struct
    process_stats *) 0);
    print_ru (fp, &ru);
    fprintf (fp, "\n");
#endif /* GET_P_STATS */

}

getru1 (kids)

int kids;

{

#ifdef GETRU_STATS
    if (kids) {
        memset (&kidsru, 0, sizeof (kidsru));
        getrusage (RUSAGE_CHILDREN, &kidsru);
    }
    else {
        memset (&selfru, 0, sizeof (selfru));
        getrusage (RUSAGE_SELF, &selfru);
    }
#endif /* GETRU_STATS */

#ifdef GET_P_STATS
    timeval_t tv;

```

Appendix C

```

if (kids) {
    memset (&kidsru, 0, sizeof (kidsru));
    get_process_stats (&tv, PS_SELF, (struct process_stats *)
0, &kidsru);
}
else {
    memset (&selfru, 0, sizeof (selfru));
    get_process_stats (&tv, PS_SELF, &selfru, (struct
process_stats *) 0);
}
#endif /* GET_P_STATS */
}

getru2 (fp, kids, config, runname, proc_no)

FILE *fp;
int kids;
char *config;
char *runname;
int proc_no;

{
#ifdef GETRU_STATS
    struct rusage ru;

    fprintf (fp, "%-10.10s %-10.10s %10d %10d ", config,
runname, proc_no, kids);
    getrusage (kids ? RUSAGE_CHILDREN :
RUSAGE_SELF, &ru);
    if (kids)
        diffru (&ru, &kidsru);
    else
        diffru (&ru, &selfru);
    print_ru (fp, &ru);
    fprintf (fp, "\n");
#endif /* GETRU_STATS */

#ifdef GET_P_STATS
    timeval_t tv;
    struct process_stats ru;

    fprintf (fp, "%-10.10s %-10.10s %10d %10d ", config,
runname, proc_no, kids);
    if (kids)
        get_process_stats (&tv, PS_SELF, (struct process_stats *)
0, &ru);
    else
        get_process_stats (&tv, PS_SELF, &ru, (struct
process_stats *) 0);
    if (kids)
        diffru (&ru, &kidsru);
    else
        diffru (&ru, &selfru);
    print_ru (fp, &ru);
    fprintf (fp, "\n");
#endif /* GET_P_STATS */
}
#endif /* GET_P_STATS */
}

fprintf (fp, "%10ld ", ru->ru_utime.tv_sec * 1000 +
(ru->ru_utime.tv_usec/1000));
fprintf (fp, "%10ld ", ru->ru_stime.tv_sec * 1000 +
(ru->ru_stime.tv_usec/1000));
fprintf (fp, "%10ld ", ru->ru_maxrss);
fprintf (fp, "%10ld ", ru->ru_majflt);
fprintf (fp, "%10ld ", ru->ru_minflt);
fprintf (fp, "%10ld ", 0);
fprintf (fp, "%10ld ", 0);
fprintf (fp, "%10ld ", 0);
fprintf (fp, "%10ld ", ru->ru_nswap);
fprintf (fp, "%10ld ", 0);
fprintf (fp, "%10ld ", ru->ru_nvcsw);
fprintf (fp, "%10ld ", ru->ru_nivcsw);
fprintf (fp, "%10ld ", ru->ru_nsignals);
fprintf (fp, "%10ld ", 0);
fprintf (fp, "%10ld ", 0);
fprintf (fp, "%10ld ", ru->ru_inblock);
fprintf (fp, "%10ld ", ru->ru_oublock);
fprintf (fp, "%10ld ", 0);
fprintf (fp, "%10ld", 0);
}

diffru (ru2, ru)

struct rusage *ru2;
struct rusage *ru;

{
    ru2->ru_utime.tv_sec -= ru->ru_utime.tv_sec;
    ru2->ru_utime.tv_usec -= ru->ru_utime.tv_usec;
    ru2->ru_stime.tv_sec -= ru->ru_stime.tv_sec;
    ru2->ru_stime.tv_usec -= ru->ru_stime.tv_usec;
    ru2->ru_maxrss -= ru->ru_maxrss;
    ru2->ru_ixrss -= ru->ru_ixrss;
    ru2->ru_idrss -= ru->ru_idrss;
    ru2->ru_minflt -= ru->ru_minflt;
    ru2->ru_majflt -= ru->ru_majflt;
    ru2->ru_nswap -= ru->ru_nswap;
    ru2->ru_inblock -= ru->ru_inblock;
}

```

Appendix C

```
ru2->ru_oublock -= ru->ru_oublock;
ru2->ru_msgsnd -= ru->ru_msgsnd;
ru2->ru_msgrcv -= ru->ru_msgrcv;
ru2->ru_nsignals -= ru->ru_nsignals;
ru2->ru_nvcsw -= ru->ru_nvcsw;
ru2->ru_nivcsw -= ru->ru_nivcsw;
```

```
}
```

```
#endif /* GETRU_STATS */
```

```
#ifdef GET_P_STATS
```

```
print_ru (fp, ps)
```

```
FILE *fp;
```

```
struct process_stats *ps;
```

```
{
```

```
fprintf (fp, "%lu ", ps->ps_untime.tv_sec * 1000 +
(ps->ps_untime.tv_usec/1000));
```

```
fprintf (fp, "%lu ", ps->ps_stime.tv_sec * 1000 +
(ps->ps_stime.tv_usec/1000));
```

```
fprintf (fp, "%lu ", ps->ps_maxrss);
```

```
fprintf (fp, "%lu ", ps->ps_pagein);
```

```
fprintf (fp, "%lu ", ps->ps_reclaim);
```

```
fprintf (fp, "%lu ", ps->ps_zerofill);
```

```
fprintf (fp, "%lu ", ps->ps_pffincr);
```

```
fprintf (fp, "%lu ", ps->ps_pffdecr);
```

```
fprintf (fp, "%lu ", ps->ps_swap);
```

```
fprintf (fp, "%lu ", ps->ps_syscall);
```

```
fprintf (fp, "%lu ", ps->ps_volcsw);
```

```
fprintf (fp, "%lu ", ps->ps_involcsw);
```

```
fprintf (fp, "%lu ", ps->ps_signal);
```

```
fprintf (fp, "%lu ", ps->ps_lread);
```

```
fprintf (fp, "%lu ", ps->ps_lwrite);
```

```
fprintf (fp, "%lu ", ps->ps_bread);
```

```
fprintf (fp, "%lu ", ps->ps_bwrite);
```

```
fprintf (fp, "%lu ", ps->ps_phread);
```

```
fprintf (fp, "%lu", ps->ps_phwrite);
```

```
}
```

```
diffru (ru2, ru)
```

```
struct process_stats *ru2;
```

```
struct process_stats *ru;
```

```
{
```

```
ru2->ps_untime.tv_sec -= ru->ps_untime.tv_sec;
```

```
ru2->ps_untime.tv_usec -= ru->ps_untime.tv_usec;
```

```
ru2->ps_stime.tv_sec -= ru->ps_stime.tv_sec;
```

```
ru2->ps_stime.tv_usec -= ru->ps_stime.tv_usec;
```

```
ru2->ps_maxrss -= ru->ps_maxrss;
```

```
ru2->ps_pagein -= ru->ps_pagein;
```

```
ru2->ps_reclaim -= ru->ps_reclaim;
```

```
ru2->ps_zerofill -= ru->ps_zerofill;
```

```
ru2->ps_pffincr -= ru->ps_pffincr;
```

```
ru2->ps_pffdecr -= ru->ps_pffdecr;
```

```
ru2->ps_swap -= ru->ps_swap;
```

```
ru2->ps_syscall -= ru->ps_syscall;
```

```
ru2->ps_volcsw -= ru->ps_volcsw;
```

```
ru2->ps_involcsw -= ru->ps_involcsw;
```

```
ru2->ps_signal -= ru->ps_signal;
```

```
ru2->ps_lread -= ru->ps_lread;
```

```
ru2->ps_lwrite -= ru->ps_lwrite;
```

```
ru2->ps_bread -= ru->ps_bread;
```

```
ru2->ps_bwrite -= ru->ps_bwrite;
```

```
ru2->ps_phread -= ru->ps_phread;
```

```
ru2->ps_phwrite -= ru->ps_phwrite;
```

```
}
```

```
#endif /* GET_P_STATS */
```

gtime.c

```
/* Copyright (c) 2001, 2002, Oracle Corporation. All rights
reserved. */
```

```
/*
```

NAME

gtime.c - <one-line expansion of the name>

DESCRIPTION

<short description of facility this file declares/defines>

EXPORT FUNCTION(S)

<external functions defined for use outside package - one-line descriptions>

INTERNAL FUNCTION(S)

<other external functions defined - one-line descriptions>

STATIC FUNCTION(S)

<static functions defined - one-line descriptions>

NOTES

<other useful comments, qualifications, etc.>

MODIFIED (MM/DD/YY)

mposs 10/23/02 - mposs_update_from_visa

mposs 08/29/01 - Creation

```
*/
```

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

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

```
# include <sys/time.h>
```

```
main ()
```

```
{
```

Appendix C

```
struct timeval tv;

(void) gettimeofday (&tv, (struct timezone *) 0);

printf ("%0.2f\n", ((double) tv.tv_sec + (1.0e-6 * (double)
tv.tv_usec)) );

}

/* end of file gtime.c */
```

iso1.sh

```
#!/bin/ksh
#
# $Header: iso1.sh 29-jul-98.17:00:11 akarasik Exp $
#
# iso1.sh
#
# Copyright (c) Oracle Corporation 1998. All Rights
Reserved.
#
# NAME
#   iso1.sh
#
# DESCRIPTION
#   Usage: iso1.sh [-u user/password] [-n remote_node] -h
#   Options: See usage below
# NOTES
#   For a cross node isolation test, assume the local node
is
#   one of the participating nodes. The other node can
be
#   specified by the -n option.
#   You need to set the environment variable
TPCD_KIT_DIR
#
# MODIFIED (MM/DD/YY)
# mpoess 12/16/98 - update to version 8.1.6
# mpoess 09/25/98 - update audit
# akarasik 07/29/98 -
# akarasik 07/29/98 - Creation
#

.$KIT_DIR/env

# May need to change the following:
RSH=ssh

OH=$ORACLE_HOME
#ACID_DIR=$KIT_DIR/acid is set in env
OUT_DIR=$ACID_OUT

TXN1FILE=$OUT_DIR/txn1$$$.out
TXN2FILE=$OUT_DIR/txn2$$$.out
KEYFILE=$OUT_DIR/key$$$.out
ISOFILE=$OUT_DIR/iso1
```

```
USER=$DATABASE_USER
PROG=atranspl
```

```
/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE
```

```
trap "/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE; exit
1" 1 2 3 15
```

```
usage() {
```

```
    echo ""
    echo "Usage: $0 [-u user/passwd] [-n remote_node] -h"
    echo ""
    exit 1;
}
```

```
set -- `getopt "u:n:h" "$@"` || usage
```

```
while :
```

```
do
    case "$1" in
        -u) shift; USER=$1;;
        -n) shift; HOST="$1";;
        -h) usage; exit 0;;
        --) break;;
    esac
    shift;
done
```

```
de=`direxists.sh $ACID_OUT c` # I am not using $de
afterward, but I want to avoid the output of direxists
```

```
# generate key files
```

```
randkey 1 0.1 u"$USER" > $KEYFILE
```

```
OKEY=`cat $KEYFILE | awk '{print $1}'`
echo "o_key is "$OKEY
```

```
# before the ACID transaction, let's run a ACID query to
record the
# initial state of lineitem
```

```
echo "Running ACID query BEFORE the start of Isolation
Test 1" >> $TXN2FILE
echo "`date`" >> $TXN2FILE
echo "" >> $TXN2FILE
sqlplus $USER @$ACID_DIR/isolation/a_query $OKEY >>
$TXN2FILE
echo "" >> $TXN2FILE
echo "-----" >>
$TXN2FILE
```

```
sleep 1
```

```
# start ACID transaction, Sleep for 60 second before
COMMIT
```

Appendix C

```
$PROG 1 1 1 0 i$KEYFILE u$USER s60 b0 >> $TXN1FILE
&
```

```
# let's sleep 10 seconds before starting ACID query
```

```
sleep 15
```

```
# start ACID query with the same OKEY
```

```
echo "Running ACID query 15 seconds AFTER the start of
ACID Transaction" \
>> $TXN2FILE
echo "`date`" >> $TXN2FILE
if [ "$HOST" != "" ]
then
echo "Starting ACID query on node $HOST" >> $TXN2FILE
${RSH} -n ${HOST} sqlplus $USER
@$ACID_DIR/isolation/a_query $OKEY >> $TXN2FILE
else
sqlplus $USER @$ACID_DIR/isolation/a_query $OKEY >>
$TXN2FILE
fi
```

```
echo "-----" >>
$TXN2FILE
wait
echo "-----" >>
$TXN1FILE
```

```
cat $TXN1FILE $TXN2FILE >> $ISOFILE
```

```
/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE
```

iso2.sh

```
#!/bin/ksh
#
# $Header: iso2.sh 04-aug-99.09:19:54 mpoess Exp $
#
# iso2.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
# iso2.sh - <one-line expansion of the name>
#
# DESCRIPTION
# Usage: iso2.sh [-u user/password] [-n remote_node] -h
# Options: See usage below
# NOTES
# For a cross node isolation test, assume the local node is
# one of the participating nodes. The other node can be
# specified by the -n option.
# You need to set the environment variable
TPCD_KIT_DIR
#
# MODIFIED (MM/DD/YY)
# mpoess 08/04/99 - Creation
```

```
# mpoess 08/04/99 - Creation
#
#
#
```

```
=====
=====+
# May need to change the following:
```

```
. $KIT_DIR/env
```

```
RSH=ssh
```

```
OH=$ORACLE_HOME
# ACID_DIR=$TPCD_KIT_DIR/audit is set in env
OUT_DIR=$ACID_OUT
```

```
DURA_DIR=$ACID_DIR/dura
```

```
TXN1FILE=$OUT_DIR/txn1$.out
TXN2FILE=$OUT_DIR/txn2$.out
KEYFILE=$OUT_DIR/key$.out
ISOFILE=$OUT_DIR/iso2
```

```
USER=$DATABASE_USER
PROG=atranspl
```

```
/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE
```

```
trap "/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE; exit
1" 1 2 3 15
```

```
usage() {
```

```
echo ""
echo "Usage: $0 [-u user/passwd] [-n remote_node] -h"
echo ""
exit 1;
}
```

```
set -- `getopt "u:n:h" "$@"` || usage
```

```
while :
```

```
do
case "$1" in
-u) shift; USER=$1;;
-n) shift; HOST="$1";;
-h) usage; exit 0;;
--) break;;
esac
shift;
done
```

```
# generate key files
```

```
randkey 1 0.1 u"$USER" > $KEYFILE
```

```
OKEY=`cat $KEYFILE | awk '{print $1}'`
echo "o_key is "$OKEY
```


Appendix C

```
# before the ACID transaction, let's run a ACID query to
record the
# initial state of lineitem

echo "Running ACID query BEFORE the start of Isolation
Test 1" >> $TXN2FILE
echo "`date`" >> $TXN2FILE
echo "" >> $TXN2FILE
sqlplus "$USER" @$ACID_DIR/isolation/a_query $OKEY
>> $TXN2FILE
echo "" >> $TXN2FILE
echo "-----" >>
$TXN2FILE

sleep 1

# start ACID transaction, Sleep for 30 second before
ROLLBACK

$PROG 1 1 0 0 i$KEYFILE u$USER s30 >> $TXN1FILE &

# let's sleep 15 seconds before starting ACID query

sleep 15

# start ACID query with the same OKEY

echo "Running ACID query 15 seconds AFTER the start of
ACID transaction" \
>> $TXN2FILE
echo "`date`" >> $TXN2FILE
if [ "$HOST" != "" ]
then
echo "Starting ACID query on node $HOST" >> $TXN2FILE
${RSH} -n ${HOST} sqlplus "$USER"
@$ACID_DIR/isolation/a_query $OKEY >> $TXN2FILE
else
sqlplus $USER @$ACID_DIR/isolation/a_query $OKEY >>
$TXN2FILE
fi

echo "-----" >>
$TXN2FILE
wait
echo "-----" >>
$TXN1FILE

cat $TXN1FILE $TXN2FILE >> $ISOFILE

/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE

iso3.sh
#!/bin/ksh
#
# $Header: iso3.sh 04-aug-99.09:20:35 mpoess Exp $
#
# iso3.sh
#
```

```
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
# iso3.sh - <one-line expansion of the name>
#
# DESCRIPTION
# Usage: iso3.sh [-u user/password] [-n remote_node] -h
# Options: See usage below
# NOTES
# For a cross node isolation test, assume the local node is
# one of the participating nodes. The other node can be
# specified by the -n option.
# We need to make sure the remote node has access to
the
# file system on the local node. Otherwise, we need to
rcp
# the keyfile to the remote system.
# You need to set the environment variable
TPCD_KIT_DIR
#
# MODIFIED (MM/DD/YY)
# mpoess 08/04/99 - Creation
# mpoess 08/04/99 - Creation
#

.$KIT_DIR/env

# May need to change the following:
RSH=ssh

OH=$ORACLE_HOME
#ACID_DIR=$TPCD_KIT_DIR/audit is set in env
OUT_DIR=$ACID_OUT

DURA_DIR=$ACID_DIR/dura

TXN1FILE=$OUT_DIR/txn1$.out
TXN2FILE=$OUT_DIR/txn2$.out
KEYFILE=$OUT_DIR/key$.out
ISOFILE=$OUT_DIR/iso3

USER=$DATABASE_USER
PROG=atranspl

/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE

trap "/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE; exit
1" 1 2 3 15

usage() {

echo ""
echo "Usage: $0 [-u user/passwd] [-n remote_node] -h"
echo ""
exit 1;
}

set -- `getopt "u:n:h" "$@"` || usage
```

Appendix C

```
while :
do
  case "$1" in
    -u) shift; USER=$1;;
    -n) shift; HOST="$1";;
    -h) usage; exit 0;;
    --) break;;
    esac
  shift
done

# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE
scp $KEYFILE ${HOST}:$KEYFILE

sleep 1

# start ACID transaction, Sleep for 30 second before
COMMIT

$PROG 1 2 1 0 i$KEYFILE u$USER s30 b0 >> $TXN1FILE
&

# let's sleep 15 seconds before starting second ACID
transaction

sleep 15

# start another ACID transaction with the same LKEY and
OKEY
# but different DELTA

# Do not sleep before COMMIT so that we can see TXN2 has
waited.

if [ "$HOST" != "" ]
then
  echo "Starting TXN2 on node $HOST" >> $TXN2FILE
  ${RSH} -n ${HOST} $PROG 2 2 1 1 i$KEYFILE u$USER
  s1 b1 >> $TXN2FILE &
else
  $PROG 2 2 1 1 i$KEYFILE u$USER s1 b1 >> $TXN2FILE
  &
fi

wait
echo "-----" >>
$TXN2FILE
echo "-----" >>
$TXN1FILE

cat $TXN1FILE $TXN2FILE >> $ISOFILE

/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE
```

iso4.sh

64

```
#!/bin/ksh
#
# $Header: iso4.sh 04-aug-99.09:21:12 mpoess Exp $
#
# iso4.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
#   iso4.sh - <one-line expansion of the name>
#
# DESCRIPTION
#   Usage: iso4.sh [-u user/password] [-n remote_node] -h
#   Options: See usage below
# NOTES
#   For a cross node isolation test, assume the local node is
#   one of the participating nodes. The other node can be
#   specified by the -n option.
#   We need to make sure the remote node has access to the
#   file system on the local node. Otherwise, we need to rcp
#   the keyfile to the remote system.
#   You need to set the environment variable
TPCD_KIT_DIR
#
# MODIFIED (MM/DD/YY)
# mpoess 08/04/99 - Creation
# mpoess 08/04/99 - Creation
#

. $KIT_DIR/env

# May need to change the following:
RSH=ssh

OH=$ORACLE_HOME
# ACID_DIR=$TPCD_KIT_DIR/audit is set in env
OUT_DIR=$ACID_OUT

DURA_DIR=$ACID_DIR/dura

TXN1FILE=$OUT_DIR/txn1$.out
TXN2FILE=$OUT_DIR/txn2$.out
KEYFILE=$OUT_DIR/key$.out
ISOFILE=$OUT_DIR/iso4

USER=$DATABASE_USER
PROG=atranspl

/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE

trap "/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE; exit
1" 1 2 3 15

usage() {

  echo ""
  echo "Usage: $0 [-u user/passwd] [-n remote_node] -h"
  echo ""
```

Appendix C

```
exit 1;
}

set -- `getopt "u:n:h" "$@"` || usage

while :
do
  case "$1" in
    -u) shift; USER=$1;;
    -n) shift; HOST="$1";;
    -h) usage; exit 0;;
    --) break;;
    esac
  shift
done

# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE
scp $KEYFILE ${HOST}:$KEYFILE

sleep 1

# start ACID transaction, Sleep for 30 second before
ROLLBACK

$PROG 1 2 0 0 i$KEYFILE u$USER s30 b0 >> $TXN1FILE
&

# let's sleep 15 seconds before starting second ACID
transaction

sleep 15

# start another ACID transaction with the same LKEY and
OKEY
# but different DELTA

# Do not sleep before COMMIT so that we can see TXN2 has
waited.

if [ "$HOST" != "" ]
then
echo "Starting TXN2 on node $HOST" >> $TXN2FILE
${RSH} -n ${HOST} $PROG 2 2 1 1 i$KEYFILE u$USER
s1 b1 >> $TXN2FILE &
else
$PROG 2 2 1 1 i$KEYFILE u$USER s1 b1 >> $TXN2FILE
&
fi

wait
echo "-----" >>
$TXN2FILE
echo "-----" >>
$TXN1FILE

cat $TXN1FILE $TXN2FILE >> $ISOFILE
```

```
/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE
```

iso5.sh

```
#!/bin/ksh
#
# $Header: iso5.sh 04-aug-99.09:21:45 mpoess Exp $
#
# iso5.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
# iso5.sh - <one-line expansion of the name>
#
# DESCRIPTION
# Usage: iso5.sh [-u user/password] [-n remote_node] -h
# Options: See usage below
# NOTES
# For a cross node isolation test, assume the local node is
# one of the participating nodes. The other node can be
# specified by the -n option.
# You need to set the environment variable
TPCD_KIT_DIR
#
# MODIFIED (MM/DD/YY)
# mpoess 08/04/99 - Creation
# mpoess 08/04/99 - Creation
#

.$KIT_DIR/env

# May need to change the following:
RSH=ssh

OH=$ORACLE_HOME
# ACID_DIR=$TPCD_KIT_DIR/audit is set in env
OUT_DIR=$ACID_OUT
DURA_DIR=$ACID_DIR/dura

TXN1FILE=$OUT_DIR/txn1$.out
TXN2FILE=$OUT_DIR/txn2$.out
KEYFILE=$OUT_DIR/key$.out
ISOFILE=$OUT_DIR/iso5

USER=$DATABASE_USER
PROG=atranspl

/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE

trap "/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE; exit
1" 1 2 3 15

usage() {
  echo ""
  echo "Usage: $0 [-u user/passwd] [-n remote_node] -h"
  echo ""
```

Appendix C

```
exit 1;
}

set -- `getopt "u:n:h" "$@"` || usage

while :
do
  case "$1" in
    -u) shift; USER=$1;;
    -n) shift; HOST="$1";;
    -h) usage; exit 0;;
    --) break;;
    esac
  shift;
done

# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE
scp $KEYFILE ${HOST}:$KEYFILE

OKEY=`cat $KEYFILE | awk '{print $1}'`
echo "o_key is "$OKEY

# before the ACID transaction, let's run a ACID query to
record the
# initial state of lineitem

echo "Running ACID query BEFORE the start of Isolation
Test 5" >> $TXN1FILE
echo "`date`" >> $TXN1FILE
echo "" >> $TXN1FILE
sqlplus $USER @$ACID_DIR/isolation/a_query $OKEY >>
$TXN1FILE
echo "" >> $TXN1FILE
echo "-----" >>
$TXN1FILE

sleep 1

# start ACID transaction, Sleep for 60 second before
COMMIT

$PROG 1 1 1 0 i$KEYFILE u$USER s60 >> $TXN1FILE &

# let's sleep 5 seconds before starting PARTSUPP query

sleep 5

# First generate PS_PARTKEY and PS_SUPPKEY

PSKEY=`randpsup 1`

echo "Running PARTSUPP query 5 seconds AFTER the start
of ACID Transaction" \
>> $TXN2FILE
echo "`date`" >> $TXN2FILE
```

```
echo "PS_PARTKEY and PS_SUPPKEY are: $PSKEY" >>
$TXN2FILE

if [ "$HOST" != "" ]
then
echo "Starting PARTSUPP query on node $HOST" >>
$TXN2FILE
${RSH} -n ${HOST} sqlplus $USER
@$ACID_DIR/isolation/a_query2 ${PSKEY} >>
$TXN2FILE &
else
sqlplus $USER @$ACID_DIR/isolation/a_query2 ${PSKEY}
>> $TXN2FILE &
fi

wait

echo "-----" >>
$TXN2FILE
echo "-----" >>
$TXN1FILE

cat $TXN1FILE $TXN2FILE >> $ISOFILE

/bin/rm -rf $TXN1FILE $TXN2FILE $KEYFILE

iso6.sh
#!/bin/ksh
#
# $Header: iso6.sh 04-aug-99.09:22:12 mpoess Exp $
#
# iso6.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
# iso6.sh - <one-line expansion of the name>
#
# DESCRIPTION
# Usage: iso6.sh [-u user/password] [-n remote_node] -h
# Options: See usage below
# NOTES
# For a cross node isolation test, assume the local node is
# one of the participating nodes. The other node can be
# specified by the -n option.
# We need to make sure the remote node has access to the
# file system on the local node. Otherwise, we need to rcp
# the keyfile to the remote system.
# You need to set the environment variable
TPCD_KIT_DIR
#
# MODIFIED (MM/DD/YY)
# mpoess 08/04/99 - Creation
# mpoess 08/04/99 - Creation
#
. $KIT_DIR/env
```

Appendix C

```
# May need to change the following:
RSH=ssh

OH=/private/tpcd
# ACID_DIR=$TPCD_KIT_DIR/audit is set in env
OUT_DIR=$ACID_OUT

DURA_DIR=$ACID_DIR/dura

TXN1FILE=$OUT_DIR/txn1$.out
TXN2FILE=$OUT_DIR/txn2$.out
TXN3FILE=$OUT_DIR/txn3$.out
KEYFILE=$OUT_DIR/key$.out
ISOFILE=$OUT_DIR/iso6

USER=$DATABASE_USER
PROG=atranspl

/bin/rm -rf $TXN1FILE $TXN2FILE $TXN3FILE
$KEYFILE

trap "/bin/rm -rf $TXN1FILE $TXN2FILE $TXN3FILE
$KEYFILE; exit 1" 1 2 3 15

usage() {
    echo ""
    echo "Usage: $0 [-u user/passwd] [-n remote_node] -h"
    echo ""
    exit 1;
}

set -- `getopt "u:n:h" "$@"` || usage

while :
do
    case "$1" in
    -u) shift; USER=$1;;
    -n) shift; HOST="$1";;
    -h) usage; exit 0;;
    --) break;;
    esac
    shift;
done

# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE
scp $KEYFILE ${HOST}:$KEYFILE

OKEY=`cat $KEYFILE | awk '{print $1}'`
echo "o_key is "$OKEY

# before the any transaction, let's run a ACID query to record
the
# initial state of lineitem

echo "Running ACID query BEFORE the start of Isolation
Test 6" >> $TXN2FILE
echo "`date`" >> $TXN2FILE
echo "" >> $TXN2FILE
sqlplus $USER @$ACID_DIR/isolation/a_query $OKEY >>
$TXN2FILE
echo "" >> $TXN2FILE
echo "-----" >>
$TXN2FILE

sleep 1

# start Query 1, use 0 as the delta

echo "Running Query 17b at `date`" >> $TXN1FILE
sqlplus $USER @q1 >> $TXN1FILE &

# sleep 2 seconds before starting ACID transaction

sleep 2

# start ACID transaction, COMMIT after one second

echo "Starting AICD transaction at `date`" >> $TXN2FILE

if [ "$HOST" != "" ]
then
echo "Starting ACID transaction on node $HOST" >>
$TXN2FILE
${RSH} -n ${HOST} $PROG 1 1 1 0 i$KEYFILE u$USER
s1 >> $TXN2FILE &
else
$PROG 1 1 1 0 i$KEYFILE u$USER s1 >> $TXN2FILE &
fi

# start Query 17

sleep 2

echo "Running 2nd Query 17b at `date`" >> $TXN3FILE
sqlplus $USER @q1 >> $TXN3FILE &
# wait for everyone to finish

wait

echo "-----" >>
$TXN3FILE
echo "-----" >>
$TXN2FILE
echo "-----" >>
$TXN1FILE

cat $TXN1FILE $TXN2FILE $TXN3FILE >> $ISOFILE

/bin/rm -rf $TXN1FILE $TXN2FILE $TXN3FILE
$KEYFILE
```

randkey.c

Appendix C

```
/* Copyright (c) 2001, 2002, Oracle Corporation. All rights
reserved. */
```

```
/*
```

```
NAME
```

```
randkey.c - <one-line expansion of the name>
```

```
DESCRIPTION
```

```
Generate random keys for ACID transactions:
```

```
O_ORDERKEY unique random (1..SF*150000*4) and
only
```

```
first 8 keys out of every 32 are populated.
```

```
and
```

```
L_ORDERKEY based on Clause 3.1.6.2
```

```
DELTA random (1..100)
```

```
*/
```

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

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

```
#include <math.h>
```

```
#include "atranspl.h"
```

```
#define ORDERCNT 150000.0
```

```
/* MK_SPARSE adopted from dss.h */
```

```
#define MK_SPARSE(key, seq) \
((((key>>3)<<2)|(seq & 0x0003))<<3)|(key & 0x0007))
```

```
void sql_error();
```

```
void usage();
```

```
void ACIDinit();
```

```
long atol();
```

```
void srand48();
```

```
long lrand48();
```

```
/* Not really used here, but retained it for future purposes. */
```

```
typedef struct aciddef {
```

```
long okey;
```

```
long lkey;
```

```
int delta;
```

```
} adef;
```

```
long l_key = 0;
```

```
long o_key = 0;
```

```
char lname[UNAME_LEN];
```

```
char *passwd;
```

```
/* OCI handles */
```

```
OCIEnv *tpcenv;
```

```
OCIServer *tpcsrv;
```

```
OCIError *errhp;
```

```
OCISvcCtx *tpcsvc;
```

```
OCISession *tpcusr;
```

```
OCISmt *curi;
```

```
OCIBind *l_key_bp;
```

```
OCIBind *o_key_bp;
```

```
sword status = OCI_SUCCESS; /* OCI return value */
```

```
char sqlstmt[1024];
```

```
void ACIDexit() {
```

```
OCILogoff(tpcsvc,errhp);
```

```
OCIhfree(tpcenv,OCI_HTYPE_STMT);
```

```
OCIhfree(tpcsvc,OCI_HTYPE_SVCCTX);
```

```
OCIhfree(tpcsrv,OCI_HTYPE_SERVER);
```

```
OCIhfree(tpcusr,OCI_HTYPE_SESSION);
```

```
}
```

```
/* type: 0 if environment handle is passed, 1 if error handle is
passwd */
```

```
void sql_error(errhp,status,type)
```

```
OCIError *errhp;
```

```
sword status;
```

```
sword type;
```

```
{
```

```
char msg[2048];
```

```
sb4 errcode;
```

```
ub4 msglen;
```

```
int i,j;
```

```
switch(status) {
```

```
case OCI_SUCCESS_WITH_INFO:
```

```
fprintf(stderr, "Error: Statement returned with info.\n");
```

```
if (type)
```

```
(void) OCIErrorGet(errhp,1,NULL,(sb4 *) &errcode,(text
*)msg,
```

```
2048,OCI_HTYPE_ERROR);
```

```
else
```

```
(void) OCIErrorGet(errhp,1,NULL,(sb4 *) &errcode,(text
*)msg,
```

```
2048,OCI_HTYPE_ENV);
```

```
fprintf(stderr,"%s\n",msg);
```

```
break;
```

```
case OCI_ERROR:
```

```
fprintf(stderr, "Error: OCI call error.\n");
```

```
if (type)
```

```
(void) OCIErrorGet(errhp,1,NULL,(sb4 *) &errcode,(text
*)msg,
```

```
2048,OCI_HTYPE_ERROR);
```

```
else
```

```
(void) OCIErrorGet(errhp,1,NULL,(sb4 *) &errcode,(text
*)msg,
```

```
2048,OCI_HTYPE_ENV);
```

```
fprintf(stderr,"%s\n",msg);
```

```
break;
```

```
case OCI_INVALID_HANDLE:
```

```
fprintf(stderr, "Error: Invalid Handle.\n");
```

```
if (type)
```

Appendix C

```

(void) OCIErrorGet(errhp,1,NULL,(sb4 *) &errcode,(text
*)msg,
        2048,OCI_HTYPE_ERROR);
else
(void) OCIErrorGet(errhp,1,NULL,(sb4 *) &errcode,(text
*)msg,
        2048,OCI_HTYPE_ENV);
fprintf(stderr,"%s\n",msg);
break;
}
/* Rollback just in case */

(void) OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);

fprintf(stderr, "Exiting Oracle...\n");
fflush(stderr);

ACIDexit();

exit(1);
}

main(argc, argv)
int argc;
char **argv;
{
long count;
long i;
double sf; /* need to accomodate sf 0.1 */
double random;
double ordcnt;
adef *res;

if ((argc < 3) || (argc > 4)) {
usage();
exit(-1);
}

strcpy((char *) lname, "tpcd/tpcd");

count = atol(argv[1]);
sf = atof(argv[2]);

argc -= 2;
argv += 2;

while (--argc) {
++argv;
switch(argv[0][0]) {
case 'u':
strncpy((char *) lname, ++(argv[0]), UNAME_LEN);
if (strchr((char *) lname, '/') == NULL) {
usage();
exit(-1);
}
break;
default:
fprintf(stderr, "Unknown argument %s\n", argv[0]);
usage();
break;
}
}

ACIDinit();

/* initialize array for random numbers */

res = (adef *) malloc(count*sizeof(adef));
ordcnt = (double) ORDERCNT * (double) sf;

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

/* The algorithm: */
/* Assumes drand's output is 'unique', first get a number
within */
/* the range of [0..sf*ORDERCNT) and then maps the
different */
/* ranges to generate the real output. */

random = floor(drand48() * (double) ordcnt) + 1;
res[i].okey = o_key = (long) MK_SPARSE((long) random,
0);
res[i].delta = (long) floor(drand48() * 100) + 1;

/* Obtain l_key from l_key query */

OCIsexec(tpcsvc,curi,errhp,1);

/* l_key is the highest l_linenummer available. We need to
pick */
/* at random a number between 1..l_key. */

res[i].lkey = (lrand48() % l_key) + 1;

printf("%ld %ld %d\n", res[i].okey, res[i].lkey, res[i].delta);
}

ACIDexit();
free(res);
}

void usage() {
fprintf(stderr, "Usage: randkey <number of random keys to
generate> <SF> u<user/password>\n");
fprintf(stderr, "\n");
}

void ACIDinit()
{
/* run random seed */

srand48(getpid());

```

Appendix C

```

/* Connect to ORACLE. Program will call sql_error()
   if an error occurs in connecting to the default database. */

(void) OCIInitialize(OCI_DEFAULT,(dvoid *)0,0,0,0);
if((status=OCIEnvInit((OCIEnv
***)&tpcenv,OCI_DEFAULT,0,(dvoid **)0)) !=
OCI_SUCCESS)
    sql_error(tpcenv, status, 0);

OCIhalloc(tpcenv,&errhp,OCI_HTYPE_ERROR);
OCIhalloc(tpcenv,&curi,OCI_HTYPE_STMT);
OCIhalloc(tpcenv,&tpcsvc,OCI_HTYPE_SVCCTX);
OCIhalloc(tpcenv,&tpcsrv,OCI_HTYPE_SERVER);
OCIhalloc(tpcenv,&tpcusr,OCI_HTYPE_SESSION);

/* get username and password */

passwd = strchr(lname, '/');
*passwd = '\0';
passwd++;

if ((status=OCIServerAttach(tpcsrv,errhp,(text
*)0,0,OCI_DEFAULT))!=OCI_SUCCESS)
    sql_error(errhp,status,1);

OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcsrv,0,OCI_ATTR
_SERVER,errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,lname,strlen(lname),
OCI_ATTR_USERNAME,
errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(passw
d),OCI_ATTR_PASSWORD,
errhp);

if ((status = OCISessionBegin(tpcsvc, errhp, tpcusr,
OCI_CRED_RDBMS,
OCI_DEFAULT)) != OCI_SUCCESS)
    sql_error(errhp,status,1);

OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcusr,0,OCI_ATTR
_SESSION,errhp);

/* Open and Parse cursor for query to choose determine
l_key. */
/* Binds l_key to :l_key. */

sprintf((char *) sqlstmt,SQLTXT1);
OCIStmtPrepare(cur_i,errhp,(text *)sqlstmt,strlen((char
*)sqlstmt),
OCI_NTV_SYNTAX,OCI_DEFAULT);

OCIbname(cur_i,l_key_bp,errhp,":l_key",ADR(l_key),SIZ(l_
key),SQLT_INT);

```

```

OCIbname(cur_i,o_key_bp,errhp,":o_key",ADR(o_key),SIZ(
o_key),SQLT_INT);
}

```

randpsup.c

```

/* Copyright (c) 2001, 2002, Oracle Corporation. All rights
reserved. */

```

```

/*

```

NAME

randpsup.c - <one-line expansion of the name>

DESCRIPTION

Generate random keys for ACID PARTSUPP transactions:
(Clause 4.2.3)

PS_PARTKEY random within [SF*200000]

and

PS_SUPPKEY = (PS_PARTKEY + (i * ((S/4) +
(int)(PS_PARTKEY - 1)
/S))) % S + 1

where i random within [0..3] and S = SF * 10000

MODIFIED

mposs 10/23/02 - mposs_update_from_visa

mposs 01/04/01 - Creation

```

*/

```

```

#include <stdio.h>

```

```

#include <stdlib.h>

```

```

#include <math.h>

```

```

#define PS_PER_SF 200000.0

```

```

#define S_PER_SF 10000.0

```

```

#define SUPP_PER_PART 4

```

```

/* borrowed from build.c in the dbgen distribution */

```

```

#define PART_SUPP_BRIDGE(tgt, p, s) \

```

```

{ \

```

```

    long tot_scnt = (long) (S_PER_SF * sf); \

```

```

    tgt = (p + s * (tot_scnt / SUPP_PER_PART + \

```

```

        (long) ((p - 1) / tot_scnt))) % tot_scnt + 1; \

```

```

}

```

```

void usage();

```

```

double atof();

```

```

void srand48();

```

```

long lrand48();

```

```

main(argc, argv)

```

```

    int argc;

```

```

    char **argv;

```

```

{

```

```

    double sf = 0.1; /* scale factor */

```


Appendix C

```
long supp;          /* the i-th supplier */
long pkey;          /* partkey */
long maxpkey;      /* highest partkey */
long ps_skey;      /* ps_suppkey */

if (argc < 2) {
    usage();
    exit(-1);
}

/* seed the random number generator */

srand48(getpid());

sf = atof(argv[1]);
maxpkey = (long) (sf * PS_PER_SF);
supp = lrand48() % 4;
pkey = lrand48() % maxpkey + 1;

PART_SUPP_BRIDGE(ps_skey, pkey, supp);

fprintf(stdout, "%ld %ld", pkey, ps_skey);

exit(0);
}

void usage()
{
    fprintf(stderr, "Usage: randpsup <SF>\n\n");
}
```

run_acid.sh

```
#!/bin/ksh
#
# $Header: run_acid.sh 08-aug-99.15:30:10 mpoess Exp $
#
# run_acid.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
#   run_acid.sh - <one-line expansion of the name>
#
# DESCRIPTION
#   Usage: run_acid.sh [-n iter] [-s stream] [-p prog] [-i
infile]
#           [-o outfile] [-d durafile] [-u usr/pswd]
#           [-t trigger] [-f scale factor] -h
#
#   Options: See usage below
#
#   MODIFIED (MM/DD/YY)
#   mpoess   08/08/99 - Creation
#   mpoess   08/08/99 - Creation
#
```

```
. $KIT_DIR/env

OH=$ORACLE_HOME
ACID_DIR=$ACID_DIR
OUT_DIR=$ACID_OUT

usage() {
    echo ""
    echo "Usage: $0 [-n iter] [-s stream] [-p prog] [-i infile] [-o
outfile]"
    echo "       [-d durafile] [-u usr/pswd] -h"
    echo ""
    echo "-n iter   : number of iterations, default is 100"
    echo "-s stream  : number of streams, default is 2"
    echo "-p prog    : program to run, default is atranspl.ott"
    echo "-i infile  : input file prefix, suffix by process number
within a"
    echo "           stream and run ID, default is ./acid_in"
    echo "-o outfile : output file prefix, similar to input file"
    echo "           default is ./out/acid_out"
    echo "-d durafile : durability file prefix, used for durability
tests"
    echo "           default is ./dura/acid_dura"
    echo "-u usr/pswd : user/password combo for database
access, default is tpch/tpch"
    echo "-t trigger : trigger time between process starts, default
is 1 second"
    echo "-h        : print this usage summary"
    exit 1;
}

ITER=600
STEM=${NUM_STREAMS}
let STEM="$STEM + 1" # add one for the update stream
SF=1
PROG=atranspl
IN=${ACID_DIR}/acid_in
DURA_DIR=$ACID_OUT/dura
OUT=$DURA_DIR/drate
DURA=$DURA_DIR/dura
KEY=${DURA_DIR}/key$$_
echo "$$" > ${DURA_DIR}/shellpid
USER=tpch/tpch
TRIG=1
HCNT=duracontb

set -- `getopt "n:s:p:i:o:d:u:ht:f:" "$@"` || usage

# get all the options

while :
do
    case "$1" in
        -n) shift; ITER=$1;;
        -s) shift; STEM=$1;;
        -p) shift; PROG=$1;;
        -i) shift; IN=$1;;
        -o) shift; OUT=$1;;
```

Appendix C

```
-d) shift; DURA=$1;;
-u) shift; USER=$1;;
-h) usage; exit 0;;
-t) shift; TRIG=$1;;
-f) shift; SF=$1;;
--) break;;
esac
shift;
done

echo "Starting ACID run..."

i=0
T=`expr $STEM \* $TRIG + 6`

# Get history count before the run

sqlplus $USER @cnt_hist > $DURA_DIR/$HCNT 2>&1

while [ $i -lt $STEM ]
do
    randkey $ITER ${SF} u${USER} > ${KEY}${i} &
    i=`expr $i + 1`
done

wait
# perform the consistency

i=0
while [ $i -lt $STEM ]
do
    for j in `head -10 ${KEY}${i} | awk '{printf "%d ",$1}'`
    do
        sqlplus tpch/tpch @consist $j >> $DURA_DIR/duraconsb
        done
        i=`expr $i + 1`
    done

echo "Starting Transaction Counting Program"
count_tx.sh $STEM 100 $DURA_DIR &

i=0
while [ $i -lt $STEM ]
do

    $PROG $i $STEM 1 0 i${KEY}${i} o${OUT}${i}
d${DURA}${i} u$USER s1 &
    T=`expr $T - $TRIG`
    i=`expr $i + 1`

done

wait

echo "ACID run completed"

sample.sh
#!/bin/ksh

#
# $Header: sample.sh 08-aug-99.17:10:00 mpoess Exp $
#
# sample.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
#   sample.sh - <one-line expansion of the name>
#
# DESCRIPTION
#   <short description of component this file
declares/defines>
#
# NOTES
#   <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
#   mpoess    08/08/99 - Creation
#   mpoess    08/08/99 - Creation
#

# $1 durability output file

. $KIT_DIR/env

cat $1 | grep o_key | awk '{printf "%d \n", $2}' | head -106 >
/tmp/okey$$
cat $1 | grep l_key | awk '{printf "%d \n", $2}' | head -106 >
/tmp/lkey$$

paste /tmp/okey$$ /tmp/lkey$$ > /tmp/keys$$
tail -6 /tmp/keys$$ > /tmp/6keys$$

echo "Keys chosen are:"
cat /tmp/6keys$$

i=1
while [ $i -le 6 ]
do
    j=`cat /tmp/6keys$$ | tail -${i} | head -1`
    sqlplus tpch/tpch @sample $j
    i=`expr $i + 1`
done
#/bin/rm -f /tmp/*key*

sample.sql
Rem
Rem $Header: sample.sql 08-aug-99.17:10:34 mpoess Exp $
Rem
Rem sample.sql
Rem
Rem Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
Rem
Rem NAME
```

Appendix C

```
Rem    sample.sql - <one-line expansion of the name>
Rem
Rem    DESCRIPTION
Rem    <short description of component this file
declares/defines>
Rem
Rem    NOTES
Rem    <other useful comments, qualifications, etc.>
Rem
Rem    MODIFIED (MM/DD/YY)
Rem    mpoess    08/08/99 - Creation
Rem    mpoess    08/08/99 - Created
Rem
alter session set nls_date_format = 'YYYY-MM-DD
HH:MI:SS';
select * from history where h_o_key = &&1 and
h_l_key = &&2;
exit;
```

Appendix D: Query text and Output

Appendix D

Stream Started at 1119990971.09
Stream Ended at 1119990976.68
Stream Processed in 5.59 seconds

1.log

Begin Execution at Tue Jun 28 15:36:11 2005

-- using default substitutions

```
select
l_returnflag,
l_linestatus,
sum(l_quantity) as sum_qty,
sum(l_extendedprice) as sum_base_price,
sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as
sum_charge,
avg(l_quantity) as avg_qty,
avg(l_extendedprice) as avg_price,
avg(l_discount) as avg_disc,
count(*) as count_order
from
lineitem
where
l_shipdate <= to_date ('1998-12-01','YYYY-MM-DD') - 90
group by
l_returnflag,
l_linestatus
order by
l_returnflag,
l_linestatus
```

L_RETURNFLAG	L_LINESTATUS	SUM_QTY	SUM_BASE_PRICE	SUM_DISC_PRICE	SUM_CHARGE	AVG_QTY	AVG_PRICE	AVG_DISC	COUNT_ORDER
A	F	37734107.00	56586554400.73	53758257134.87	55909065222.83	25.52	38273.13	0.05	1478493.00
N	F	991417.00	1487504710.38	1413082168.05	1469649223.19	25.52	38284.47	0.05	38854.00
N	O	74476040.00	111701729697.74	106118230307.61	110367043872.50	25.50	38249.12	0.05	2920374.00
R	F	37719753.00	56568041380.90	53741292684.60	55889619119.83	25.51	38250.85	0.05	1478870.00

4 rows processed.

Query Processed in 5.59 seconds.

Ended Executing this Stream at Tue Jun 28 15:36:16 2005

2.log

Begin Execution at Tue Jun 28 15:36:16 2005

-- using default substitutions

```
select * from (
select
s_acctbal,
s_name,
n_name,
p_partkey,
p_mfgr,
s_address,
s_phone,
s_comment
from
part,
supplier,
partsupp,
nation,
region
```

Appendix D

<p>where</p> <p>p_partkey = ps_partkey</p> <p>and s_suppkey = ps_suppkey</p> <p>and p_size = 15</p> <p>and p_type like '%BRASS'</p> <p>and s_nationkey = n_nationkey</p> <p>and n_regionkey = r_regionkey</p> <p>and r_name = 'EUROPE'</p> <p>and ps_supplycost = (</p> <p>select</p> <p>min(ps_supplycost)</p> <p>from</p> <p>partsupp,</p> <p>supplier,</p> <p>nation,</p> <p>region</p> <p>where</p> <p>p_partkey = ps_partkey</p> <p>and s_suppkey = ps_suppkey</p> <p>and s_nationkey = n_nationkey</p> <p>and n_regionkey = r_regionkey</p> <p>and r_name = 'EUROPE'</p> <p>)</p> <p>order by</p> <p>s_acctbal desc,</p> <p>n_name,</p>	<p>s_name,</p> <p>p_partkey</p> <p>)</p> <p>where rownum <= 100</p> <table border="0" style="width: 100%;"> <tr> <td style="width: 33%;">S_ACCTBAL</td> <td style="width: 33%;">S_NAME</td> <td style="width: 33%;">N_NAME</td> </tr> <tr> <td>P_PARTKEY</td> <td>P_MFGR</td> <td></td> </tr> <tr> <td>S_ADDRESS</td> <td colspan="2">S_PHONE</td> </tr> <tr> <td colspan="3">S_COMMENT</td> </tr> </table> <p>9938.53 Supplier#000005359 UNITED KINGDOM 1 185358.00 Manufacturer#4 QKuHYh,vZGiwu2FWEJoLDx04 33-429-790- 6131 blithely silent pinto beans are furiously. slyly final deposits acros</p> <p style="text-align: right;">6,0-1 7%</p> <p>9937.84 Supplier#000005969 ROMANIA 1 108438.00 Manufacturer#1 ANDENSOSmk,miq23Xfb5RWt6dvUcvt6Qa 29-520- 692-3537 carefully slow deposits use furiously. slyly ironic platelets above the ironic</p> <p style="text-align: right;">11,1 14%</p> <p>9936.22 Supplier#000005250 UNITED KINGDOM 249.00 Manufacturer#4 B3rqp0xbSEim4Mpy2RH J 33-320-228-2957 blithely special packages are. stealthily express deposits across the closely fi</p> <p style="text-align: right;">16,1 22%</p> <p>9923.77 Supplier#000002324 GERMANY 29821.00 Manufacturer#4 y3OD9UywSTOk 17-779-299-1839 quickly express packages breach quiet pinto beans. requ</p> <p style="text-align: right;">21,1 28%</p> <p>9871.22 Supplier#000006373 GERMANY 43868.00 Manufacturer#5 J8fcXWsTqM 17-813-485-8637 never silent deposits integrate furiously blit</p> <p style="text-align: right;">26,1 35%</p> <p>9870.78 Supplier#000001286 GERMANY 81285.00 Manufacturer#2 YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH 17-516- 924-4574 final theodolites cajole slyly special,</p> <p style="text-align: right;">31,1 42%</p> <p>..... delete lines</p>	S_ACCTBAL	S_NAME	N_NAME	P_PARTKEY	P_MFGR		S_ADDRESS	S_PHONE		S_COMMENT		
S_ACCTBAL	S_NAME	N_NAME											
P_PARTKEY	P_MFGR												
S_ADDRESS	S_PHONE												
S_COMMENT													

Appendix D

7894.56 Supplier#000007981 GERMANY
85472.00 Manufacturer#4
NSJ96vMROAbeXP 17-963-404-3760
regular, even theodolites integrate carefully. bold, special
theodolites are sly
ly fluffily iron 37,1 50%
7887.08 Supplier#000009792 GERMANY
164759.00 Manufacturer#3
Y28ITVeYriT3kIGdV2K8fSZ V2UqT5H1Otz 17-988-
938-4296
pending, ironic packages sleep among the carefully ironic
accounts. quickly fina
l accounts 42,1 56%
7871.50 Supplier#000007206 RUSSIA
104695.00 Manufacturer#1
3w fNCnrVmvJjE95sgWZzvW 32-432-452-7731
furiously dogged pinto beans cajole. bold, express notornis
until the slyly pend
ing 47,1 62%
7852.45 Supplier#000005864 RUSSIA
8363.00 Manufacturer#4
WCNfBPZeSXh3h,c 32-454-883-3821
blithely regular deposits
7850.66 Supplier#000001518 UNITED
KINGDOM
86501.00 Manufacturer#1
ONda3YJiHKJOC 33-730-383-3892
furiously final accounts wake carefully idle requests. even
dolphins wake acc
7843.52 Supplier#000006683 FRANCE
11680.00 Manufacturer#4
2Z0JGkiv01Y00oCFwUGfviIbhZCdy 16-464-517-
8943
carefully bold accounts doub
"2.log" 122L, 4719C 61,1
78%

100 rows processed.
Query Processed in 0.72 seconds.

Ended Executing this Stream at Tue Jun 28 15:36:17 2005

Stream Started at 1119990976.76
Stream Ended at 1119990977.48
Stream Processed in 0.72 seconds

SQL statements processed: 1

3.log

Begin Execution at Tue Jun 28 15:36:17 2005

-- using default substitutions

select * from (

select

l_orderkey,

sum(l_extendedprice * (1 - l_discount)) as revenue,

o_orderdate,

o_shippriority

from

customer,

orders,

lineitem

where

c_mktsegment = 'BUILDING'

and c_custkey = o_custkey

and l_orderkey = o_orderkey

and o_orderdate < to_date('1995-03-15', 'YYYY-MM-DD')

and l_shipdate > to_date('1995-03-15', 'YYYY-MM-DD')

group by

l_orderkey,

o_orderdate,

o_shippriority

order by

Appendix D

revenue desc,

o_orderdate)

where rownum <= 10

10 rows processed.

Query Processed in 1.83 seconds.

Ended Executing this Stream at Tue Jun 28 15:36:19 2005

```
L_ORDERKEY      REVENUE
O_ORDERDATE O_SHIPRIORITY
```

Stream Started at 1119990977.56
Stream Ended at 1119990979.39
Stream Processed in 1.83 seconds

2456423.00 406181.01 1995-03-05 0.00

SQL statements processed: 1

3459808.00 405838.70 1995-03-04 0.00

4.log

Begin Execution at Thu May 5 17:20:17 2005

492164.00 390324.06 1995-02-19 0.00

-- using default substitutions

1188320.00 384537.94 1995-03-09 0.00

```
select
o_orderpriority,
count(*) as order_count
from
orders
where
o_orderdate >= to_date('1993-07-01', 'YYYY-MM-DD')
and o_orderdate < add_months(to_date('1993-07-01',
'YYYY-MM-DD'),3)
and exists (
select
*
from
lineitem
where
l_orderkey = o_orderkey
and l_commitdate < l_receiptdate
)
group by
o_orderpriority
order by
o_orderpriority
```

2435712.00 378673.06 1995-02-26 0.00

4878020.00 378376.80 1995-03-12 0.00

5521732.00 375153.92 1995-03-13 0.00

2628192.00 373133.31 1995-02-22 0.00

993600.00 371407.46 1995-03-05 0.00

```
O_ORDERPRIORITY ORDER_COUNT
1-URGENT          10594.00
2-HIGH            10476.00
3-MEDIUM         10410.00
4-NOT SPECIFIED 10556.00
5-LOW            10487.00
```

2300070.00 367371.15 1995-03-13 0.00

Appendix D

Query Processed in 2.93 seconds.

5 rows processed.
Query Processed in 2.14 seconds.

Ended Executing this Stream at Thu May 5 17:20:23 2005

Ended Executing this Stream at Thu May 5 17:20:20 2005

Stream Started at 1115331620.20
Stream Ended at 1115331623.13
Stream Processed in 2.93 seconds

Stream Started at 1115331617.97
Stream Ended at 1115331620.11
Stream Processed in 2.14 seconds

SQL statements processed: 1

SQL statements processed: 1

6.log

5.log

Begin Execution at Thu May 5 17:20:23 2005

Begin Execution at Thu May 5 17:20:20 2005

-- using default substitutions

-- using default substitutions

```
select
sum(l_extendedprice * l_discount) as revenue
from
lineitem
where
l_shipdate >= to_date('1994-01-01', 'YYYY-MM-DD')
and l_shipdate < add_months(to_date('1994-01-01', 'YYYY-
MM-DD'), 12)
and l_discount between .06 - 0.01 and .06 + 0.01
and l_quantity < 24
```

```
select
n_name,
sum(l_extendedprice * (1 - l_discount)) as revenue
from
customer,
orders,
lineitem,
supplier,
nation,
region
where
c_custkey = o_custkey
and l_orderkey = o_orderkey
and l_suppkey = s_suppkey
and c_nationkey = s_nationkey
and s_nationkey = n_nationkey
and n_regionkey = r_regionkey
and r_name = 'ASIA'
and o_orderdate >= to_date('1994-01-01', 'YYYY-MM-DD')
and o_orderdate < add_months(to_date('1994-01-01',
'YYYY-MM-DD'), 12)
group by
n_name
order by
revenue desc
```

REVENUE
123141078.23

1 row processed.
Query Processed in 0.62 seconds.

Ended Executing this Stream at Thu May 5 17:20:23 2005

Stream Started at 1115331623.22
Stream Ended at 1115331623.84
Stream Processed in 0.62 seconds

SQL statements processed: 1

N_NAME	REVENUE
INDONESIA	55502041.17
VIETNAM	55295087.00
CHINA	53724494.26
INDIA	52035512.00
JAPAN	45410175.70

7.log

Begin Execution at Thu May 5 17:20:23 2005

5 rows processed.

-- using default substitutions

Appendix D

Ended Executing this Stream at Thu May 5 17:20:25 2005

```
select
supp_nation,
cust_nation,
l_year,
sum(volume) as revenue
from
(
select
n1.n_name as supp_nation,
n2.n_name as cust_nation,
to_number(to_char(l_shipdate,'yyyy')) as
l_year,
l_extendedprice * (1 - l_discount) as volume
from
supplier,
lineitem,
orders,
customer,
nation n1,
nation n2
where
s_suppkey = l_suppkey
and o_orderkey = l_orderkey
and c_custkey = o_custkey
and s_nationkey = n1.n_nationkey
and c_nationkey = n2.n_nationkey
and (
(n1.n_name = 'FRANCE' and n2.n_name = 'GERMANY')
or (n1.n_name = 'GERMANY' and n2.n_name = 'FRANCE')
)
and l_shipdate between to_date('1995-01-01', 'YYYY-MM-
DD') and to_date('1996-1
2-31', 'YYYY-MM-DD')
) shipping
group by
supp_nation,
cust_nation,
l_year
order by
supp_nation,
cust_nation,
l_year
```

SUPP_NATION	CUST_NATION	L_YEAR	REVENUE
FRANCE	GERMANY	1995.00	54639732.73
FRANCE	GERMANY	1996.00	54633083.31
GERMANY	FRANCE	1995.00	52531746.67
GERMANY	FRANCE	1996.00	52520549.02

4 rows processed.
Query Processed in 2.06 seconds.

Stream Started at 1115331623.94
Stream Ended at 1115331625.99
Stream Processed in 2.06 seconds

SQL statements processed: 1

8.log

Begin Execution at Thu May 5 17:20:26 2005

-- using default substitutions

```
select
o_year,
sum(case when nation='BRAZIL' then volume else 0 end)/
sum(volume)
as mkt_share
from
(
select
to_number(to_char(o_orderdate, 'yyyy')) as o_year,
l_extendedprice * (1 - l_discount) as volume,
n2.n_name as nation
from
part,
supplier,
lineitem,
orders,
customer,
nation n1,
nation n2,
region
where
p_partkey = l_partkey
and s_suppkey = l_suppkey
and l_orderkey = o_orderkey
and o_custkey = c_custkey
and c_nationkey = n1.n_nationkey
and n1.n_regionkey = r_regionkey
and r_name = 'AMERICA'
and s_nationkey = n2.n_nationkey
and o_orderdate between to_date('1995-01-01', 'YYYY-MM-
DD') and to_date('1996-
12-31', 'YYYY-MM-DD')
and p_type = 'ECONOMY ANODIZED STEEL'
) all_nations
group by
o_year
order by
o_year
```

O_YEAR MKT_SHARE

Appendix D

1995.00 0.03
1996.00 0.04

l_extendedprice * (1 - l_discount) - ps_supplycost *
l_quantity as amount

2 rows processed.
Query Processed in 2.68 seconds.

from

Ended Executing this Stream at Thu May 5 17:20:28 2005

part,

supplier,

Stream Started at 1115331626.08
Stream Ended at 1115331628.76
Stream Processed in 2.68 seconds

lineitem,

partsupp,

orders,

SQL statements processed: 1

nation

where

9.log

s_suppkey = l_suppkey

Begin Execution at Tue Jun 28 15:37:05 2005

and ps_suppkey = l_suppkey

and ps_partkey = l_partkey

and p_partkey = l_partkey

and o_orderkey = l_orderkey

-- using default substitutions

and s_nationkey = n_nationkey

and p_name like '%green%'

) profit

select

group by

nation,

nation,

o_year,

o_year

sum(amount) as sum_profit

order by

from

nation,

(

o_year desc

select

n_name as nation,

NATION

O_YEAR

SUM_PROFIT

to_number (to_char (o_orderdate, 'yyyy')) as o_year,

ALGERIA

1998.00

31342867.23

Appendix D

ALGERIA	1997.00	57138193.02		Stream Ended at 1119991029.83 Stream Processed in 4.04 seconds
ALGERIA	1996.00	56140140.13		
ALGERIA	1995.00	53051469.65		SQL statements processed: 1
ALGERIA	1994.00	53867582.13		10.log Begin Execution at Tue Jun 28 15:37:09 2005
ALGERIA	1993.00	54942718.13		
ALGERIA	1992.00	54628034.71		-- using default substitutions
ARGENTINA	1998.00	30211185.71		select * from (
ARGENTINA	1997.00	50805741.75		select
ARGENTINA	1996.00	51923746.58		c_custkey,
ARGENTINA	1995.00	49298625.77		c_name,
ARGENTINA	1994.00	50835610.11		sum(l_extendedprice * (1 - l_discount)) as revenue,
ARGENTINA	1993.00	51646079.18		c_acctbal,
				n_name,
				c_address,
				c_phone,
				c_comment
				from
				customer,
				orders,
				lineitem,
				nation
				where
				c_custkey = o_custkey
				and l_orderkey = o_orderkey
				and o_orderdate >= to_date ('1993-10-01', 'YYYY-MM-DD')
				and o_orderdate < add_months(to_date('1993-10-01',
				'YYYY-MM-DD'), 3)
				and l_returnflag = 'R'
				and c_nationkey = n_nationkey
				group by
				c_custkey,
				c_name,
				c_acctbal,
				c_phone,
				n_name,
				c_address,
				c_comment
				order by
				revenue desc)
				where rownum <= 20
..... delete lines				
UNITED STATES	1997.00	50077306.42		
UNITED STATES	1996.00	48048649.47		
UNITED STATES	1995.00	48809032.42		
UNITED STATES	1995.00	48809032.42		
UNITED STATES	1994.00	49296747.18		
UNITED STATES	1993.00	48029946.80		
1				
UNITED STATES	1992.00	48671944.50		
1				
VIETNAM	1998.00	30442736.06	1	
VIETNAM	1997.00	50309179.79		
VIETNAM	1996.00	50488161.41	1	
VIETNAM	1995.00	49658284.61	1	
VIETNAM	1994.00	50596057.26		
VIETNAM	1993.00	50953919.15		
VIETNAM	1992.00	49613838.32		

175 rows processed.
Query Processed in 4.04 seconds.

C_CUSTKEY	C_NAME	REVENUE
C_ACCTBAL	N_NAME	
C_ADDRESS	C_PHONE	
C_COMMENT		

Ended Executing this Stream at Tue Jun 28 15:37:09 2005

57040.00	Customer#000057040	734235.25
632.87	JAPAN	
Eioyzzf4pp	22-895-641-3466	
requests sleep	blithely about the	furiiously i

Stream Started at 1119991025.79

Appendix D

Begin Execution at Tue Jun 28 15:37:12 2005

143347.00 Customer#000143347 721002.69
2557.47 EGYPT
1aReFYv,Kw4 14-742-935-3718
fluffily bold excuses haggle finally after the u

-- using default substitutions

60838.00 Customer#000060838 679127.31
2454.77 BRAZIL
64EaJ5vMAHWJIBOxJklpNc2RJiWE 12-913-494-
9813
furiously even pinto beans integrate under the ruthless foxes;
ironic, even dolp
hins across the slyl 3,0-1 5%

```
select
ps_partkey,
sum(ps_supplycost * ps_availqty) as value
from
partsupp,
supplier,
nation
where
```

..... delete lines

```
ps_suppkey = s_suppkey
and s_nationkey = n_nationkey
and n_name = 'GERMANY'
group by
ps_partkey having
sum(ps_supplycost * ps_availqty) > (
select
sum(ps_supplycost * ps_availqty) * 0.0001000000
```

142549.0091L, 3307C Customer#000142549
563537.24
5085.99 INDONESIA
ChqEoK43OysjdHbtKcP6dKqjNyvvi9 19-955-562-
2398
regular, unusual dependencies boost slyly; ironic attainments
nag fluffily into
the unusual packages?

```
from
partsupp,
supplier,
nation
where
ps_suppkey = s_suppkey
and s_nationkey = n_nationkey
and n_name = 'GERMANY'
)
order by
value desc
```

146149.00 Customer#000146149 557254.99
1791.55 ROMANIA
s87fvzFQpU 29-744-164-6487
silent, unusual requests detect quickly slyly regul

52528.00 Customer#000052528 556397.35
551.79 ARGENTINA
NFztyTOR10UOJ 11-208-192-3205
unusual requests detect. slyly dogged theodolites use slyly.
deposit

PS_PARTKEY	VALUE
129760.00	17538456.86
166726.00	16503353.92
191287.00	16474801.97
161758.00	16101755.54
34452.00	15983844.72
139035.00	15907078.34
9403.00	15451755.62
154358.00	15212937.88
38823.00	15064802.86
85606.00	15053957.15
33354.00	14408297.40

23431.00 Customer#000023431 554269.54
3381.86 ROMANIA
HgiV0phqhaIa9aydNoIlb 29-915-458-2654
instructions nag quickly. furiously bold accounts cajol

20 rows processed.
Query Processed in 2.39 seconds.

Ended Executing this Stream at Tue Jun 28 15:37:12 2005

..... delete lines

Stream Started at 1119991029.91
Stream Ended at 1119991032.30
Stream Processed in 2.39 seconds

10157.00	7897046.25
171154.00	7896814.50
79373.00	7896186.00
113808.00	7893353.88
27901.00	7892952.00
128820.00	7892882.72
25891.00	7890511.20
122819.00	7888881.02
154731.00	7888301.33

SQL statements processed: 1

11.log

Appendix D

101674.00 7879324.60
51968.00 7879102.21
72073.00 7877736.11
5182.00 7874521.73

1048 rows processed.
Query Processed in 1.03 seconds.

Ended Executing this Stream at Tue Jun 28 15:37:13 2005

Stream Started at 1119991032.38
Stream Ended at 1119991033.41
Stream Processed in 1.03 seconds

SQL statements processed: 1

12.log

Begin Execution at Thu May 5 17:20:37 2005

-- using default substitutions

```
select
  l_shipmode,
  sum(case
    when o_orderpriority = '1-URGENT'
    or o_orderpriority = '2-HIGH'
    then 1
    else 0
  end) as high_line_count,
  sum(case
    when o_orderpriority <> '1-URGENT'
    and o_orderpriority <> '2-HIGH'
    then 1
    else 0
  end) as low_line_count
from
  orders,
  lineitem
where
  o_orderkey = l_orderkey
  and l_shipmode in ('MAIL', 'SHIP')
  and l_commitdate < l_receiptdate
  and l_shipdate < l_commitdate
  and l_receiptdate >= to_date('1994-01-01', 'YYYY-MM-DD')
  and l_receiptdate < add_months(to_date('1994-01-01',
'YYYY-MM-DD'), 12)
group by
  l_shipmode
order by
```

l_shipmode

	L_SHIPMODE	HIGH_LINE_COUNT	LOW_LINE_COUNT
MAIL	6202.00	9324.00	
SHIP	6200.00	9262.00	

2 rows processed.
Query Processed in 1.88 seconds.

Ended Executing this Stream at Thu May 5 17:20:39 2005

Stream Started at 1115331637.64
Stream Ended at 1115331639.53
Stream Processed in 1.88 seconds

SQL statements processed: 1

13.log

Begin Execution at Thu May 5 17:20:39 2005

-- using default substitutions

```
select
  c_count,
  count(*) as custdist
from
  (
  select
    c_custkey,
    count(o_orderkey) as c_count
  from
    customer, orders where
    c_custkey = o_custkey(+)
    and o_comment(+) not like '%special%requests%'
  group by
    c_custkey
  ) c_orders
group by
  c_count
order by
  custdist desc,
  c_count desc
```

C_COUNT	CUSTDIST
0.00	50004.00
9.00	6641.00
10.00	6566.00
11.00	6058.00
8.00	5949.00
12.00	5553.00

Appendix D

13.00	4989.00
19.00	4748.00
7.00	4707.00
18.00	4625.00
15.00	4552.00
17.00	4530.00
14.00	4484.00
20.00	4461.00
16.00	4323.00
21.00	4217.00
22.00	3730.00
6.00	3334.00
23.00	3129.00
24.00	2622.00
25.00	2079.00
5.00	1972.00
26.00	1593.00
27.00	1185.00
4.00	1033.00
28.00	869.00
29.00	559.00
3.00	398.00
30.00	373.00
31.00	235.00
2.00	144.00
32.00	128.00
33.00	71.00
34.00	48.00
35.00	33.00
1.00	23.00
36.00	17.00
37.00	7.00
40.00	4.00
38.00	4.00
39.00	2.00
41.00	1.00

42 rows processed.
Query Processed in 2.32 seconds.

Ended Executing this Stream at Thu May 5 17:20:41 2005

Stream Started at 1115331639.62
Stream Ended at 1115331641.93
Stream Processed in 2.32 seconds

SQL statements processed: 1

14.log

Begin Execution at Thu May 5 17:20:42 2005

-- using default substitutions

```
select
  100.00 * sum(case
    when p_type like 'PROMO%'
      then l_extendedprice * (1 - l_discount)
    else 0
  end) / sum(l_extendedprice * (1 - l_discount)) as
  promo_revenue
from
  lineitem,
  part
where
  l_partkey = p_partkey
  and l_shipdate >= date '1995-09-01'
  and l_shipdate < date '1995-09-01' + interval '1' month
```

PROMO_REVENUE
16.38

1 row processed.
Query Processed in 0.51 seconds.

Ended Executing this Stream at Thu May 5 17:20:42 2005

Stream Started at 1115331642.02
Stream Ended at 1115331642.53
Stream Processed in 0.51 seconds

SQL statements processed: 1

15.log

Begin Execution at Thu May 5 17:20:42 2005

-- using default substitutions

```
with revenue
as (select
  l_suppkey supplier_no,
  sum(l_extendedprice * (1 - l_discount)) total_revenue
from
  lineitem
where
  l_shipdate >= date '1996-01-01'
  and l_shipdate < date '1996-01-01' + interval '3'
  month
group by
  l_suppkey)
select
  s_suppkey,
```

Appendix D

```
s_name,  
s_address,  
s_phone,  
total_revenue  
from  
supplier,  
revenue  
where  
s_suppkey = supplier_no  
and total_revenue = (  
select  
max(total_revenue)  
from  
revenue )  
order by  
s_suppkey
```

```
S_SUPPKEY      S_NAME  
S_ADDRESS      S_PHONE  
TOTAL_REVENUE  
8449.00        Supplier#000008449  
Wp34zim9qYFbVctdW      20-469-856-8873  
1772627.21
```

1 row processed.
Query Processed in 3.08 seconds.

Ended Executing this Stream at Thu May 5 17:20:45 2005

Stream Started at 1115331642.62
Stream Ended at 1115331645.69
Stream Processed in 3.08 seconds

SQL statements processed: 1

16.log

Begin Execution at Tue Jun 28 15:37:20 2005

-- using default substitutions

```
select  
p_brand,  
p_type,  
p_size,  
count(distinct ps_suppkey) as supplier_cnt  
from  
partsupp,  
part  
where  
p_partkey = ps_partkey  
and p_brand <> 'Brand#45'  
and p_type not like 'MEDIUM POLISHED%'
```

```
and p_size in (49, 14, 23, 45, 19, 3, 36, 9)  
and ps_suppkey not in (  
select  
s_suppkey  
from  
supplier  
where  
s_comment like '%Customer%Complaints%'  
)  
group by  
p_brand,  
p_type,  
p_size  
order by  
supplier_cnt desc,  
p_brand,  
p_type,  
p_size
```

```
P_BRAND  P_TYPE      P_SIZE  
SUPPLIER_CNT  
  
Brand#41  MEDIUM BRUSHED TIN      3.00  
28.00  
  
Brand#54  STANDARD BRUSHED COPPER  14.00  
27.00  
  
Brand#11  STANDARD BRUSHED TIN     23.00  
24.00  
  
Brand#11  STANDARD BURNISHED BRASS 36.00  
24.00  
  
Brand#15  MEDIUM ANODIZED NICKEL  3.00  
24.00  
  
Brand#15  SMALL ANODIZED BRASS    45.00  
24.00  
  
Brand#15  SMALL BURNISHED NICKEL  19.00  
24.00  
  
Brand#21  MEDIUM ANODIZED COPPER  3.00  
24.00  
  
Brand#22  SMALL BRUSHED NICKEL    3.00  
24.00  
Brand#25  MEDIUM BURNISHED COPPER 36.00  
24.00  
  
Brand#31  PROMO POLISHED COPPER   36.00  
24.00  
  
Brand#33  LARGE POLISHED TIN      23.00  
24.00  
  
Brand#33  PROMO POLISHED STEEL    14.00  
24.00
```


Appendix D

Brand#35 PROMO BRUSHED NICKEL 14.00
24.00

Brand#41 ECONOMY BRUSHED STEEL 9.00
24.00

Brand#41 ECONOMY POLISHED TIN 19.00
24.00

Brand#41 LARGE PLATED COPPER 36.00
24.00

Brand#42 ECONOMY PLATED BRASS 3.00
24.00

Brand#42 STANDARD POLISHED TIN 49.00
24.00

Brand#43 PROMO BRUSHED TIN 3.00
24.00

Brand#43 SMALL ANODIZED COPPER 36.00
24.00

Brand#44 STANDARD POLISHED NICKEL 3.00
24.00

..... delete lines

Brand#55 STANDARD POLISHED TIN 36.00
4.00

Brand#11 SMALL BRUSHED TIN 19.00
3.00

Brand#15 LARGE PLATED NICKEL 45.00
3.00

Brand#15 LARGE POLISHED NICKEL 9.00
3.00

Brand#21 PROMO BURNISHED STEEL 45.00
3.00

Brand#22 STANDARD PLATED STEEL 23.00
3.00

Brand#25 LARGE PLATED STEEL 19.00
3.00

Brand#32 STANDARD ANODIZED COPPER 23.00
3.00

Brand#33 SMALL ANODIZED BRASS 9.00
3.00

Brand#35 MEDIUM ANODIZED TIN 19.00
3.00

Brand#51 SMALL PLATED BRASS 23.00
3.00

Brand#52 MEDIUM BRUSHED BRASS 45.00
3.00

Brand#53 MEDIUM BRUSHED TIN 45.00
3.00

Brand#54 ECONOMY POLISHED BRASS 9.00
3.00

Brand#55 PROMO PLATED BRASS 19.00
3.00

Brand#55 STANDARD PLATED TIN 49.00
3.00

18314 rows processed.
Query Processed in 0.73 seconds.

Ended Executing this Stream at Tue Jun 28 15:37:21 2005

Stream Started at 1119991040.62
Stream Ended at 1119991041.35
Stream Processed in 0.73 seconds

SQL statements processed: 1

17.log
Begin Execution at Thu May 5 17:20:48 2005

-- using default substitutions

```
select
sum(l_extendedprice) / 7.0 as avg_yearly
from
lineitem ,
part
where
p_partkey = l_partkey
and p_brand = 'Brand#23'
and p_container = 'MED BOX'
and l_quantity < (
select
0.2 * avg(l_quantity)
from
lineitem
where
l_partkey = p_partkey
)
```

Appendix D

AVG_YEARLY
348406.05

1 row processed.
Query Processed in 2.32 seconds.

Ended Executing this Stream at Thu May 5 17:20:50 2005

Stream Started at 1115331648.46
Stream Ended at 1115331650.78
Stream Processed in 2.32 seconds

SQL statements processed: 1

18.log
Begin Execution at Tue Jun 28 15:37:24 2005

-- using default substitutions

```
select * from (
select
c_name,
c_custkey,
o_orderkey,
o_orderdate,
o_totalprice,
sum(l_quantity)
from
customer,
orders,
lineitem
where
o_orderkey in (
select
l_orderkey
from
lineitem
group by
l_orderkey having
sum(l_quantity) > 300
)
and c_custkey = o_custkey
and o_orderkey = l_orderkey
group by
c_name,
c_custkey,
o_orderkey,
o_orderdate,
o_totalprice
order by
o_totalprice desc,
o_orderdate
```

88

)
where rownum <= 100

C_NAME	C_CUSTKEY	O_ORDERKEY
O_ORDERD		
ATE		
O_TOTALPRICE	SUM(L_QUANTITY)	
Customer#000128120	128120.00	4722021.00
1994-04-07		
544089.09	323.00	
Customer#000144617	144617.00	3043270.00
1997-02-12		
530604.44	317.00	
Customer#000013940	13940.00	2232932.00
1997-04-13		
522720.61	304.00	
Customer#000066790	66790.00	2199712.00
1996-09-30		
515531.82	327.00	

19.log
Begin Execution at Thu May 5 17:20:55 2005

-- using default substitutions

```
select
sum(l_extendedprice* (1 - l_discount)) as revenue
from
lineitem,
part
where
(
p_partkey = l_partkey
and p_brand = 'Brand#12'
and p_container in ('SM CASE', 'SM BOX', 'SM PACK', 'SM
PKG')
and l_quantity >= 1 and l_quantity <= 1 + 10
and p_size between 1 and 5
and l_shipmode in ('AIR', 'AIR REG')
and l_shipinstruct = 'DELIVER IN PERSON'
)
or
(
p_partkey = l_partkey
and p_brand = 'Brand#23'
and p_container in ('MED BAG', 'MED BOX', 'MED PKG',
'MED PACK')
and l_quantity >= 10 and l_quantity <= 10 + 10
and p_size between 1 and 10
and l_shipmode in ('AIR', 'AIR REG')
and l_shipinstruct = 'DELIVER IN PERSON'
)
```

Appendix D

```
or
(
p_partkey = l_partkey
and p_brand = 'Brand#34'
and p_container in ('LG CASE', 'LG BOX', 'LG PACK', 'LG
PKG')
and l_quantity >= 20 and l_quantity <= 20 + 10
and p_size between 1 and 15
and l_shipmode in ('AIR', 'AIR REG')
and l_shipinstruct = 'DELIVER IN PERSON'
)
```

REVENUE
3083843.06

1 row processed.
Query Processed in 2.69 seconds.

Ended Executing this Stream at Thu May 5 17:20:58 2005

Stream Started at 1115331655.75
Stream Ended at 1115331658.44
Stream Processed in 2.69 seconds
SQL statements processed: 1

20.log

Begin Execution at Tue Jun 28 15:37:30 2005

-- using default substitutions

```
select
s_name,
s_address
from
supplier,
nation
where
s_suppkey in (
select
ps_suppkey
from
partsupp
where
ps_partkey in (
select
p_partkey
from
part
where
p_name like 'forest%'
)
and ps_availqty > (
```

```
0.5 * sum(l_quantity)
from
lineitem
where
l_partkey = ps_partkey
and l_suppkey = ps_suppkey
and l_shipdate >= to_date ('1994-01-01', 'YYYY-MM-DD')
and l_shipdate < add_months( to_date ('1994-01-01', 'YYYY-
MM-DD'), 12)
)
)
and s_nationkey = n_nationkey
and n_name = 'CANADA'
order by
s_name
```

```
S_NAME          S_ADDRESS
Supplier#000000020  iybAE,RmTymrZVYyAfZva2SH,j
Supplier#000000091  YV45D7TkfdQanOOZ7q9QxkyGUapU1oOWU6q3
Supplier#000000197  YC2Acon6kjY3zj3Fbxs2k4Vdf7X0cd2F
Supplier#000000226  83qOdU2EYRdPQAQhEtn GRZEd
Supplier#000000285  Br7e1nnt1yxrw6ImgpJ7YdhFDjuBf
Supplier#000000378  FfbhyCxWvcPrO8ltp9
Supplier#000000402  i9Sw4DoyMhzhKXCH9By,AYSgmD
Supplier#000000530  0qwCMwobKY
OcmLyfRXlagA8ukENJv,
Supplier#000000688  D
fw5ocppmZpYBBIP1718hCihLDZ5KhKX
Supplier#000000710  f19YPvOyb
QoYwjKC,oPycpGfieBAcwKJo
Supplier#000000736  l6i2nMwVuovfKnuVgaSGK2rDy65D1AFLegiL7
Supplier#000000761  zLSLe1QUj2XrvTTFnv7WAcYZGvvMTx882d4
Supplier#000000884  bmhEShejaS
..... delete lines .....
Supplier#000009601  51m637bO,Rw5DnHWFUvLacRx9
Supplier#000009709  rRnCbHYgDgl9PZYnyWKVYSUW0vKg
Supplier#000009753  wLhVEcRmd7PkJF4FBnGK7Z
Supplier#000009796  z,y4Idmr15DOvPUqYG
Supplier#000009799  4wNjXGa4OKW1
Supplier#000009811  E3iuyq7UnZxU7oPZIE2Gu6
Supplier#000009812  APFRMy3lCbGfGa53n5t9DxzFPQPgnjrGt32
Supplier#000009862  rJzweWeN58
Supplier#000009868  ROjGgx5gvtkmnUUoeyy7v
Supplier#000009869  ucLqxzrpBTRMewGSM29t0rNTM30g1Tu3Xgg3mKag
Supplier#000009899  7XdpAHzr1t,UQFZE
Supplier#000009974  7wJ,J5DKcxSU4Kp1cQLpbcAvB5AsvKT
```

204 rows processed.
Query Processed in 1.02 seconds.

Appendix D

s_name)
where rownum <= 100

Ended Executing this Stream at Tue Jun 28 15:37:31 2005

Stream Started at 1119991050.59
Stream Ended at 1119991051.61
Stream Processed in 1.02 seconds

SQL statements processed: 1

21.log

Begin Execution at Thu May 5 17:20:59 2005

-- using default substitutions

```
select * from (
select
s_name,
count(*) numwait
from
supplier,
lineitem l1,
orders,
nation
where
s_suppkey = l1.l_suppkey
and o_orderkey = l1.l_orderkey
and o_orderstatus = 'F'
and l1.l_receiptdate > l1.l_commitdate
and exists (
select
*
from
lineitem l2
where
l2.l_orderkey = l1.l_orderkey
and l2.l_suppkey <> l1.l_suppkey
)
and not exists (
select
*
from
lineitem l3
where
l3.l_orderkey = l1.l_orderkey
and l3.l_suppkey <> l1.l_suppkey
and l3.l_receiptdate > l3.l_commitdate
)
and s_nationkey = n_nationkey
and n_name = 'SAUDI ARABIA'
group by
s_name
order by
numwait desc,
```

S_NAME	NUMWAIT
Supplier#000002829	20.00
Supplier#000005808	18.00
Supplier#000000262	17.00
Supplier#000000496	17.00
Supplier#000002160	17.00
Supplier#000002301	17.00
Supplier#000002540	17.00
Supplier#000003063	17.00
Supplier#000005178	17.00
Supplier#000008331	17.00
Supplier#000002005	16.00
Supplier#000002095	16.00
Supplier#000005799	16.00
Supplier#000005842	16.00
Supplier#000006450	16.00
Supplier#000006939	16.00
Supplier#000009200	16.00
Supplier#000009727	16.00
Supplier#000000486	15.00
Supplier#000000565	15.00
Supplier#000001046	15.00
Supplier#000001047	15.00
Supplier#000001161	15.00
Supplier#000001336	15.00
Supplier#000001435	15.00
Supplier#000003075	15.00
Supplier#000003335	15.00
Supplier#000005649	15.00
Supplier#000006027	15.00
Supplier#000006795	15.00
Supplier#000006800	15.00
Supplier#000006824	15.00
Supplier#000007131	15.00
Supplier#000007382	15.00
Supplier#000008913	15.00
Supplier#000009787	15.00
Supplier#000000633	14.00
Supplier#000001960	14.00
Supplier#000002323	14.00
Supplier#000002490	14.00
Supplier#000002993	14.00
Supplier#000003101	14.00
Supplier#000004489	14.00
Supplier#000005435	14.00
Supplier#000005583	14.00
Supplier#000005774	14.00
Supplier#000007579	14.00
Supplier#000008180	14.00
Supplier#000008695	14.00
Supplier#000009224	14.00
Supplier#000000357	13.00
Supplier#000000436	13.00
Supplier#000000610	13.00
Supplier#000000788	13.00
Supplier#000000889	13.00

Appendix D

Supplier#000001062 13.00
 Supplier#000001498 13.00
 Supplier#000002056 13.00
 Supplier#000002312 13.00
 Supplier#000002344 13.00
 Supplier#000002596 13.00
 Supplier#000002615 13.00
 Supplier#000002978 13.00
 Supplier#000003048 13.00
 Supplier#000003234 13.00
 Supplier#000003727 13.00
 Supplier#000003806 13.00
 Supplier#000004472 13.00
 Supplier#000005236 13.00
 Supplier#000005906 13.00
 Supplier#000006241 13.00
 Supplier#000006326 13.00
 Supplier#000006384 13.00
 Supplier#000006394 13.00
 Supplier#000006624 13.00
 Supplier#000006629 13.00
 Supplier#000006682 13.00
 Supplier#000006737 13.00
 Supplier#000006825 13.00
 Supplier#000007021 13.00
 Supplier#000007417 13.00
 Supplier#000007497 13.00
 Supplier#000007602 13.00
 Supplier#000008134 13.00
 Supplier#000008234 13.00
 Supplier#000009435 13.00
 Supplier#000009436 13.00
 Supplier#000009564 13.00
 Supplier#000009896 13.00
 Supplier#000000379 12.00
 Supplier#000000673 12.00
 Supplier#000000762 12.00
 Supplier#000000811 12.00
 Supplier#000000821 12.00
 Supplier#000001337 12.00
 Supplier#000001916 12.00
 Supplier#000001925 12.00
 Supplier#000002039 12.00
 Supplier#000002357 12.00
 Supplier#000002483 12.00

SQL statements processed: 1

22.log

Begin Execution at Thu May 5 17:21:06 2005

-- using default substitutions

```
select
  cntrycode,
  count(*) as numcust,
  sum(c_acctbal) as totacctbal
from
  (
  select
    substr(c_phone, 1, 2) as cntrycode,
    c_acctbal
  from
    customer
  where
    substr(c_phone,1, 2) in
    ('13', '31', '23', '29', '30', '18', '17')
  and c_acctbal > (
  select
    avg(c_acctbal)
  from
    customer
  where
    c_acctbal > 0.00
  and substr(c_phone, 1, 2) in
    ('13', '31', '23', '29', '30', '18', '17')
  )
  and not exists (
  select
    *
  from
    orders
  where
    o_custkey = c_custkey
  )
  ) custsale
group by
  cntrycode
order by
  cntrycode
```

100 rows processed.
 Query Processed in 6.78 seconds.

Ended Executing this Stream at Thu May 5 17:21:06 2005

Stream Started at 1115331659.71
 Stream Ended at 1115331666.49
 Stream Processed in 6.78 seconds

CNTRYCODE	NUMCUST	TOTACCTBAL
13	888.00	6737713.99
17	861.00	6460573.72
18	964.00	7236687.40
23	892.00	6701457.95
29	948.00	7158866.63
30	909.00	6808436.13
31	922.00	6806670.18

Appendix D

7 rows processed.
Query Processed in 1.80 seconds.

Ended Executing this Stream at Thu May 5 17:21:08 2005

Stream Started at 1115331666.58
Stream Ended at 1115331668.38
Stream Processed in 1.80 seconds

SQL statements processed: 1

Appendix E

Appendix E: Seed and Input Parameters

Appendix E

Seed

0615174947

Stream 0

14 1995-08-01
2 2 COPPER AFRICA
9 powder
20 cream 1997-01-01 ARGENTINA
6 1994-01-01 0.03 24
17 Brand#24 WRAP PACK
18 314
8 IRAQ MIDDLE EAST ECONOMY BURNISHED
STEEL
21 UNITED KINGDOM
13 pending accounts
3 FURNITURE 1995-03-02
22 12 18 11 23 33 15 16
16 Brand#42 LARGE POLISHED 29 25 4
1 17
11 48 40
4 1993-04-01
11 CANADA 0.0000003333
15 1994-10-01
1 111
10 1993-06-01
19 Brand#52 Brand#51 Brand#21 9 18
30
5 MIDDLE EAST 1994-01-01
7 ETHIOPIA IRAQ
12 FOB AIR 1995-01-01

Stream 1

21 MOROCCO
3 AUTOMOBILE 1995-03-18
18 312
5 AFRICA 1994-01-01
11 MOZAMBIQUE 0.0000003333
7 RUSSIA CANADA
6 1994-01-01 0.09 24
20 olive 1996-01-01 MOZAMBIQUE
17 Brand#21 WRAP DRUM
12 MAIL RAIL 1995-01-01
16 Brand#32 STANDARD ANODIZED 50 42
29 16
14 20 30 45
15 1997-05-01
13 pending accounts
10 1994-03-01

94

2 40 STEEL ASIA
8 CANADA AMERICA LARGE BRUSHED STEEL
14 1995-11-01
19 Brand#54 Brand#44 Brand#25 4 19
26
9 pale
22 26 22 10 18 32 11 33
1 119
4 1995-11-01

Stream 2

6 1994-01-01 0.06 25
17 Brand#22 SM BAG
14 1996-02-01
16 Brand#12 MEDIUM BURNISHED 3 1
22 12
10 2 49 50
19 Brand#11 Brand#22 Brand#14 9 20
23
10 1995-01-01
9 moccasin
2 28 BRASS AFRICA
15 1995-02-01
8 SAUDI ARABIA MIDDLE EAST LARGE
PLATED STEEL
5 AMERICA 1994-01-01
22 15 14 26 12 22 18 31
12 RAIL MAIL 1995-01-01
7 KENYA SAUDI ARABIA
13 pending accounts
18 313
1 66
4 1993-08-01
20 beige 1994-01-01 ETHIOPIA
3 FURNITURE 1995-03-04
11 EGYPT 0.0000003333
21 GERMANY

Stream 3

8 JAPAN ASIA LARGE ANODIZED STEEL
5 ASIA 1995-01-01
4 1996-02-01
6 1995-01-01 0.04 24
17 Brand#24 SM PACK
7 FRANCE JAPAN
1 74
18 315
22 30 14 25 34 29 33 11
14 1996-05-01
9 maroon
10 1993-10-01
15 1997-09-01
11 PERU 0.0000003333
20 lawn 1997-01-01 SAUDI ARABIA
2 16 NICKEL EUROPE
21 UNITED STATES

Appendix E

19 Brand#13 Brand#55 Brand#13 5 10
30
13 pending accounts
16 Brand#42 ECONOMY POLISHED 7 24
50 49
3 5 16 42
12 AIR RAIL 1996-01-01
3 MACHINERY 1995-03-20

Stream 4

5 EUROPE 1995-01-01
21 MOZAMBIQUE
14 1996-09-01
19 Brand#15 Brand#43 Brand#13 10 11
26
15 1995-05-01
17 Brand#21 SM DRUM
12 REG AIR RAIL 1996-01-01
6 1995-01-01 0.09 24
4 1993-11-01
9 lawn
8 EGYPT MIDDLE EAST MEDIUM POLISHED
COPPER
16 Brand#32 STANDARD BRUSHED 35 50
37 21
39 4 33 9
11 ETHIOPIA 0.0000003333
2 3 COPPER AFRICA
10 1994-07-01
18 312
1 82
13 pending accounts
7 UNITED KINGDOM EGYPT
22 17 11 26 33 29 24 34
3 FURNITURE 1995-03-06
20 smoke 1996-01-01 IRAN

Stream 5

21 INDONESIA
15 1993-02-01
4 1996-06-01
6 1995-01-01 0.06 25
7 MOROCCO VIETNAM
16 Brand#12 LARGE BURNISHED 16 37 19
3 4
1 23 7
19 Brand#22 Brand#25 Brand#52 5 12
22
18 314
14 1996-12-01
22 22 26 16 20 12 27 19
11 CHINA 0.0000003333
13 pending deposits
3 MACHINERY 1995-03-22
1 90
2 41 STEEL EUROPE
5 MIDDLE EAST 1995-01-01
8 VIETNAM ASIA MEDIUM BURNISHED COPPER

20 floral 1994-01-01 UNITED STATES
12 SHIP TRUCK 1996-01-01
17 Brand#23 LG BAG
10 1993-04-01
9 hot

Stream 6

10 1994-01-01
3 BUILDING 1995-03-08
15 1995-09-01
13 pending deposits
6 1995-01-01 0.04 25
8 JORDAN MIDDLE EAST SMALL BRUSHED
COPPER
9 gainsboro
7 GERMANY JORDAN
4 1994-03-01
11 FRANCE 0.0000003333
22 21 26 22 16 18 11 10
18 315
12 FOB TRUCK 1997-01-01
1 98
5 AFRICA 1995-01-01
16 Brand#52 PROMO PLATED 17 28 14
43 33
12 50 23
2 29 BRASS AMERICA
14 1997-03-01
19 Brand#24 Brand#13 Brand#51 10 13
29
20 plum 1993-01-01 KENYA
17 Brand#25 LG PACK
21 ARGENTINA

Appendix F

Appendix F: Benchmark Scripts

Appendix F

```
FROM PARTSUPP WHERE PS_PARTKEY =19763);
```

dbtables.sql

```
set echo on
set numwidth 25
spool rdbtablest
SELECT COUNT(*) FROM LINEITEM;

SELECT * FROM LINEITEM
WHERE L_ORDERKEY IN
( 4, 26598, 148577, 387431, 56704, 517442, 600000)
AND L_LINENUMBER = 1
ORDER BY L_ORDERKEY;

SELECT * FROM REGION;

SELECT COUNT(*) FROM NATION;

SELECT * FROM NATION
WHERE N_NATIONKEY IN (3,10,14,20)
ORDER BY N_NATIONKEY;
SELECT COUNT(*) FROM ORDERS;
SELECT * FROM ORDERS
WHERE O_ORDERKEY IN ( 7, 44065, 287590, 411111,
483876, 599942 )
ORDER BY O_ORDERKEY;

SELECT COUNT(*) FROM PART;

SELECT * FROM PART
WHERE P_PARTKEY IN
(1,984,8743,9028,13876,17899,20000)
ORDER BY P_PARTKEY;

SELECT COUNT(*) FROM PARTSUPP;

SELECT* FROM PARTSUPP
WHERE PS_PARTKEY = 3398
AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
FROM PARTSUPP WHERE PS_PARTKEY = 3398);

SELECT* FROM PARTSUPP
WHERE PS_PARTKEY =15873
AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
FROM PARTSUPP WHERE PS_PARTKEY = 15873);

SELECT* FROM PARTSUPP
WHERE PS_PARTKEY = 11394
AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
FROM PARTSUPP WHERE PS_PARTKEY = 11394);

SELECT* FROM PARTSUPP
WHERE PS_PARTKEY = 6743
AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
FROM PARTSUPP WHERE PS_PARTKEY = 6743);

SELECT* FROM PARTSUPP
WHERE PS_PARTKEY = 19763
AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
```

```
SELECT COUNT(*) FROM SUPPLIER;

SELECT * FROM SUPPLIER
WHERE S_SUPPKEY IN (83,265,492,784,901,1000)
ORDER BY S_SUPPKEY;

SELECT COUNT(*) FROM CUSTOMER;

DROP TABLE MINMAX;

CREATE TABLE MINMAX
(TNAME CHAR(15),
KEYMIN INTEGER,
KEYMAX INTEGER);

INSERT INTO MINMAX
SELECT
'LINEITEM_ORD',MIN(L_ORDERKEY),MAX(L_ORDER
KEY)
FROM LINEITEM ;

INSERT INTO MINMAX
SELECT
'LINEITEM_NBR',MIN(L_LINENUMBER),MAX(L_LINE
NUMBER)
FROM LINEITEM;

INSERT INTO MINMAX
SELECT
'ORDERTBL',MIN(O_ORDERKEY),MAX(O_ORDERKEY)
FROM ORDERS;

INSERT INTO MINMAX
SELECT
'CUSTOMER',MIN(C_CUSTKEY),MAX(C_CUSTKEY)
FROM CUSTOMER;

INSERT INTO MINMAX
SELECT 'PART',MIN(P_PARTKEY),MAX(P_PARTKEY)
FROM PART;

INSERT INTO MINMAX
SELECT
'SUPPLIER',MIN(S_SUPPKEY),MAX(S_SUPPKEY)
FROM SUPPLIER;

INSERT INTO MINMAX
SELECT
'PARTSUPP_PART',MIN(PS_PARTKEY),MAX(PS_PART
KEY)
FROM PARTSUPP;

INSERT INTO MINMAX
```

Appendix F

```
SELECT
'PARTSUPP_SUPP',MIN(PS_SUPPKEY),MAX(PS_SUPPK
EY)
FROM PARTSUPP ;
```

```
INSERT INTO MINMAX
SELECT
'NATION',MIN(N_NATIONKEY),MAX(N_NATIONKEY)
FROM NATION;
```

```
INSERT INTO MINMAX
SELECT
'REGION',MIN(R_REGIONKEY),MAX(R_REGIONKEY)
FROM REGION;
```

```
SELECT * FROM MINMAX;
spool off
exit;
```

dbinsert.sql

```
rem
rem
=====
=====+
rem FILENAME
rem   inserts.sql
rem DESCRIPTION
rem   Inserts duplicate rows with new key numbers and
rem   inserts rows with values beyond the TPC-D values.
rem
rem
=====
=====
rem
rem Usage:  sqlplus tpcc/tpcc @insert
rem
```

```
set pagesize 100
set termout on
set echo on
set timing on
spool rdbinsert
```

```
rem
=====
=====
rem Duplicates
rem
=====
=====
```

```
REM get timestamp
```

```
select to_char (sysdate, 'HH24:MI:SS') timestamp from dual;
```

```
drop table temp_part;
create table temp_part as
  select * from part
    where p_partkey = 1;
update temp_part
  set p_partkey = 2147483647;
insert into part
  (select * from temp_part);
select * from part
  where p_partkey = 2147483647
 or p_partkey = 1;
delete from part
  where p_partkey = 2147483647;
drop table temp_part;
commit;
```

```
drop table temp_supplier;
create table temp_supplier as
  select * from supplier
    where s_suppkey = 1;
update temp_supplier
  set s_suppkey = 2147483647;
insert into supplier
  (select * from temp_supplier);
select * from supplier
  where s_suppkey = 2147483647
 or s_suppkey = 1;
delete from supplier
  where s_suppkey = 2147483647;
drop table temp_supplier;
commit;
```

```
drop table temp_partsupp;
create table temp_partsupp as
  select * from partsupp
    where ps_partkey = 1
      and ps_suppkey = 2;
update temp_partsupp
  set ps_partkey = 2147483647,
    ps_suppkey = 2147483647;
insert into partsupp
  (select * from temp_partsupp);
select * from partsupp
  where (ps_partkey = 2147483647
 and ps_suppkey = 2147483647)
 or (ps_partkey = 1
 and ps_suppkey = 2);
delete from partsupp
  where ps_partkey = 2147483647
 and ps_suppkey = 2147483647;
drop table temp_partsupp;
commit;
```

```
drop table temp_customer;
create table temp_customer as
```

Appendix F

```
select * from customer
  where c_custkey = 1;
update temp_customer
  set c_custkey = 2147483647;
insert into customer
  (select * from temp_customer);
select * from customer
  where c_custkey = 2147483647
 or c_custkey = 1;
delete from customer
  where c_custkey = 2147483647;
drop table temp_customer;
commit;

drop table temp_orders;
create table temp_orders as
  select * from orders
  where o_orderkey = (select min(o_orderkey) from
orders);
update temp_orders
  set o_orderkey = 2147483647;
insert into orders
  (select * from temp_orders);
select * from orders
  where o_orderkey = 2147483647
 or o_orderkey = (select min(o_orderkey) from
orders);
delete from orders
  where o_orderkey = 2147483647;
drop table temp_orders;
commit;

drop table temp_lineitem;
create table temp_lineitem as
  select * from lineitem
  where l_orderkey = (select min(o_orderkey) from
orders)
 and l_linenumber = 1;
update temp_lineitem
  set l_orderkey = 2147483647,
  l_partkey = 2147483647,
  l_suppkey = 2147483647,
  l_linenumber = -2147483646;
insert into lineitem
  (select * from temp_lineitem);
select * from lineitem
  where (l_orderkey = 2147483647
 and l_partkey = 2147483647
 and l_suppkey = 2147483647
 and l_linenumber = -2147483646)
 or (l_orderkey = (select min(o_orderkey) from
orders)
 and l_linenumber = 1);
delete from lineitem
  where l_orderkey = 2147483647
 and l_partkey = 2147483647
 and l_suppkey = 2147483647
 and l_linenumber = -2147483646;
drop table temp_lineitem;
```

```
commit;

drop table temp_nation;
create table temp_nation as
  select * from nation
  where n_nationkey = 1;
update temp_nation
  set n_nationkey = 2147483647;
insert into nation
  (select * from temp_nation);
select * from nation
  where n_nationkey = 2147483647
 or n_nationkey = 1;
delete from nation
  where n_nationkey = 2147483647;
drop table temp_nation;
commit;

drop table temp_region;
create table temp_region as
  select * from region
  where r_regionkey = 1;
update temp_region
  set r_regionkey = 2147483647;
insert into region
  (select * from temp_region);
select * from region
  where r_regionkey = 2147483647
 or r_regionkey = 1;
delete from region
  where r_regionkey = 2147483647;
drop table temp_region;
commit;

rem
=====
=====
rem Duplicates finished starting inserts for domain range
rem
=====
=====

REM get timestamp
select to_char (sysdate, 'HH24:MI:SS') timestamp from dual;

insert into supplier
  (s_suppkey, s_name, s_address, s_nationkey,
s_phone,
s_acctbal, s_comment)
values
(2147483647, 'NAME text .....25E',
'Address varchar .....30.....40E',
2147483647, 'This is phone E', 123456789012,
'Supplier comment field is 101 long no E');
select * from supplier
  where s_suppkey = 2147483647;
delete from supplier D
```

Appendix F

```

where s_suppkey = 2147483647;
rem
=====
=====
insert into part
  (p_partkey, p_name, p_mfgr, p_brand, p_type,
   p_size, p_container, p_retailprice, p_comment)
values
(2147483647, 'Pname text .....2.....3.....4....5E',
 'Pmfgr text.....2....5E', 'Pbrand 10E',
 'Ptype varchar.....2....5E', 2147483646,
 'PcontainrE', 123456789012,
 'Part comment field 23E');
select * from part
  where p_partkey = 2147483647;
delete from part
  where p_partkey = 2147483647;
rem
=====
=====
insert into partsupp
  (ps_partkey, ps_suppkey, ps_availqty, ps_supplycost,
   ps_comment)
values
(2147483647, 2147483647, -2147483646,
123456789012,
 'PS comment field is 199 long no E');
select * from partsupp
  where ps_partkey = 2147483647
    and ps_suppkey = 2147483647;
delete from partsupp
  where ps_partkey = 2147483647
    and ps_suppkey = 2147483647;
rem
=====
=====
insert into customer
  (c_custkey, c_name, c_address, c_nationkey,
   c_phone, c_acctbal, c_mktsegment, c_comment)
values
(2147483647, 'Customer Name goes to 25E',
 'Customer Address goes here..3.....4E',
 2147483647, 'This is phone E', 123456789012,
 'ZMark segE', 'Customer comments fiels is 117 long
no E');
select * from customer
  where c_custkey = 2147483647;
delete from customer
  where c_custkey = 2147483647;
rem
=====
=====
insert into orders
  (o_orderkey, o_custkey, o_orderstatus, o_totalprice,
   o_orderdate, o_orderpriority, o_clerk, o_shippriority,
   o_comment)
values
(2147483647, 2147483647, 'X', 123456789012,
TO_DATE('2005-12-30','YYYY-MM-DD'),

```

```

'Order Priority5E', 'Fixed text 15E', -2147483646,
'Order comments field is 79 no E');
select * from orders
  where o_orderkey = 2147483647
    and o_custkey = 2147483647;
delete from orders
  where o_orderkey = 2147483647
    and o_custkey = 2147483647;
rem
=====
=====
insert into lineitem
  (l_orderkey, l_partkey, l_suppkey, l_linenummer,
   l_quantity, l_extendedprice, l_discount, l_tax,
   l_returnflag, l_linestatus, l_shipdate, l_commitdate,
   l_receiptdate, l_shipinstruct, l_shipmode,
   l_comment)
values
(2147483647,
 2147483647,
 2147483647,
-2147483646,
-123456789012,
-123456789012,
-123456789012,
-123456789012,
'Q',
'R',
TO_DATE('2005-12-30','YYYY-MM-DD'),
TO_DATE('2005-12-30','YYYY-MM-DD'),
TO_DATE('2005-12-30','YYYY-MM-DD'),
'Ship by camel .....5E',
'Ship ASAPE',
'Is this really what you wanted? 44 long...E');
select * from lineitem
  where l_orderkey = 2147483647
    and l_partkey = 2147483647
    and l_suppkey = 2147483647
    and l_linenummer = -2147483646;
delete from lineitem
  where l_orderkey = 2147483647
    and l_partkey = 2147483647
    and l_suppkey = 2147483647
    and l_linenummer = -2147483646;
rem
=====
=====
insert into nation
  (n_nationkey, n_name, n_regionkey, n_comment)
values
(2147483647,
'Ze Republic d MakebelievE',
2147483647,
'A nation comment for field size 152 no E');
select * from nation
  where n_nationkey = 2147483647
    and n_regionkey = 2147483647;
delete from nation
  where n_nationkey = 2147483647

```

Appendix F

```
and n_regionkey = 2147483647;
rem
=====
=====
insert into region
  (r_regionkey, r_name, r_comment)
  values
  (2147483647,
   'Ze ends of the earth....E',
   'A reasonable comment would go herE');
select * from region
  where r_regionkey = 2147483647;
delete from region
  where r_regionkey = 2147483647;
rem
=====
=====
REM get timestamp
select to_char (sysdate, 'HH24:MI:SS') timestamp from dual;
rem
=====
=====
rem Done
rem
=====
=====
spool off;
exit;
```

gen_seed.sh

```
#!/bin/ksh

SEED_FILE=$1

#Generate the seed

echo "Setting the random number seed"

PSEED=`date +%m:%d:%H:%M:%S | sed -e 's://g'`

echo "Using ${PSEED} as seed0"

echo ${PSEED} > $SEED_FILE

echo "Done setting the random number seed"
```

runTPCHall.beforeload

```
#!/bin/ksh

. $KIT_DIR/env

ECHO=echo

sqlplus=$ORACLE_HOME/bin/sqlplus

GTIME=${KIT_DIR}/utils/gtime

RUN_ID_FILE=${KIT_DIR}/audit/r_id

if [ ! -f $RUN_ID_FILE ]

then

  echo "0" > $RUN_ID_FILE

fi

RUN_ID=`cat $RUN_ID_FILE`

RUN_ID=`expr $RUN_ID + 1`

echo $RUN_ID > $RUN_ID_FILE

OUT_DIR=${KIT_DIR}/audit/tests/${RUN_ID}

if [ ! -d $OUT_DIR ]

then

  mkdir $OUT_DIR

fi

SCRIPT_LOG_FILE=${OUT_DIR}/main.out
RDB_TABLES=${OUT_DIR}/rdbtablest
```

Appendix F

```
FIRST_TEN=${OUT_DIR}/firstten
LD1DBCRE=${OUT_DIR}/Ld1dbcre
LD2SCTSO=${OUT_DIR}/Ld2sctso
LD3DAPOP=${OUT_DIR}/Ld3dapop

2shut abort
2asmshut

echo Start TPC-H Benchmark SEQUENCE NUMBER:
$RUN_ID > $SCRIPT_LOG_FILE
echo >> $SCRIPT_LOG_FILE
echo "Starting a new Oracle log file:
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}
.log" >> $SCRIPT_LOG_FILE
echo >> $SCRIPT_LOG_FILE

mv
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log
$ORACLE_HOME/rdbms/log/alert_${
ORACLE_SID}.log.preAudit.$RUN_ID
touch
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log

echo "Start: load database `date`" >> $SCRIPT_LOG_FILE
asmstart
build_dg_AUDIT2.sh >> $LD1DBCRE
dbcre_AUDIT2.sh >> $LD1DBCRE
tscre_AUDIT2.sh >> $LD2SCTSO
tshut
asmshut
2asmstart
2start
STIME=`$GTIME`
echo "Start: timed load portion `date`" >>
$SCRIPT_LOG_FILE
dapop_AUDIT2.sh >> $LD3DAPOP
$KIT_DIR/audit/gen_seed.sh $KIT_DIR/audit/seed
echo Generated seed: `cat $KIT_DIR/audit/seed` >>
$SCRIPT_LOG_FILE
2shut
2asmshut
2asmstart
2start
2shut
2asmshut
echo "End: timed load portion `date`" >>
$SCRIPT_LOG_FILE

exit

2asmstart >> $SCRIPT_LOG_FILE
2start >> $SCRIPT_LOG_FILE
ckpnt.sh
echo "Start: dbtables.sql and count.sql" >>
$SCRIPT_LOG_FILE
$sqlplus ${DATABASE_USER} @$KIT_DIR/audit/dbtables
> ${RDB_TABLES} 2>&1
$sqlplus ${DATABASE_USER} @$KIT_DIR/audit/firstten
> ${FIRST_TEN} 2>&1
```

```
echo "End: dbtables.sql and count.sql `date`" >>
$SCRIPT_LOG_FILE

runTPCHpt ${SCALE_FACTOR} 1 ${RUN_ID}

#2shut > $SCRIPT_LOG_FILE
#2start >> $SCRIPT_LOG_FILE
#ckpnt.sh
runTPCHpt ${SCALE_FACTOR} 2 ${RUN_ID}

sleep 600
2shut >> $SCRIPT_LOG_FILE

cp $ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log
$OUT_DIR

echo "End TPC-H Benchmark SEQUENCE NUMBER:
$RUN_ID `date`" >> $SCRIPT_LOG_FILE

runTPCHall.afterload
#!/bin/ksh
. $KIT_DIR/env

ECHO=echo

sqlplus=$ORACLE_HOME/bin/sqlplus
GTIME=${KIT_DIR}/utils/gtime

RUN_ID_FILE=${KIT_DIR}/audit/r_id

if [ ! -f $RUN_ID_FILE ]
then
echo "0" > $RUN_ID_FILE
fi

RUN_ID=`cat $RUN_ID_FILE`
#RUN_ID=`expr $RUN_ID + 1`
echo $RUN_ID > $RUN_ID_FILE

OUT_DIR=${KIT_DIR}/audit/tests/${RUN_ID}
if [ ! -d $OUT_DIR ]
then
mkdir $OUT_DIR
fi
SCRIPT_LOG_FILE=${OUT_DIR}/main.out
RDB_TABLES=${OUT_DIR}/rdbtablest
FIRST_TEN=${OUT_DIR}/firstten
LD1DBCRE=${OUT_DIR}/Ld1dbcre
LD2SCTSO=${OUT_DIR}/Ld2sctso
LD3DAPOP=${OUT_DIR}/Ld3dapop

#echo Start TPC-H Benchmark SEQUENCE NUMBER:
$RUN_ID > $SCRIPT_LOG_FILE
```


Appendix F

```
#echo >> $SCRIPT_LOG_FILE
#echo "Starting a new Oracle log file:
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}
}.log" >> $SCRIPT_LOG_FILE
#echo >> $SCRIPT_LOG_FILE
#
#mv
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log
$ORACLE_HOME/rdbms/log/alert_
${ORACLE_SID}.log.preAudit.$RUN_ID
#touch
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log

#echo "Start: load database `date`" >> $SCRIPT_LOG_FILE
#asmstart
#build_dg_AUDIT2.sh >> $LD1DBCRE
#dbcre_AUDIT2.sh >> $LD1DBCRE
#tscre_AUDIT2.sh >> $LD2SCTSO
#tshut
#asmshut
#2asmstart
#2start
#STIME=`$GTIME`
#echo "Start: timed load portion `date`" >>
$SCRIPT_LOG_FILE
#dapop_AUDIT2.sh >> $LD3DAPOP
#KIT_DIR/audit/gen_seed.sh $KIT_DIR/audit/seed
#echo Generated seed: `cat $KIT_DIR/audit/seed` >>
$SCRIPT_LOG_FILE
#2shut
#2asmshut
#echo "End: timed load portion `date`" >>
$SCRIPT_LOG_FILE

#2asmstart >> $SCRIPT_LOG_FILE
#2start >> $SCRIPT_LOG_FILE
#ckpnt.sh

#echo "Start: dbtables.sql and count.sql" >>
$SCRIPT_LOG_FILE
#sqlplus ${DATABASE_USER}
@KIT_DIR/audit/dbtables > ${RDB_TABLES} 2>&1
#sqlplus ${DATABASE_USER} @KIT_DIR/audit/firstten
> ${FIRST_TEN} 2>&1
#echo "End: dbtables.sql and count.sql `date`" >>
$SCRIPT_LOG_FILE

#runTPCHpt ${SCALE_FACTOR} 1 ${RUN_ID}
#2shut >> $SCRIPT_LOG_FILE
#2asmshut >> ${SCRIPT_LOG_FILE}
#exit
2asmstart >> ${SCRIPT_LOG_FILE}
2start >> ${SCRIPT_LOG_FILE}
ckpnt.sh
runTPCHpt ${SCALE_FACTOR} 2 ${RUN_ID}

sleep 600
2shut >> ${SCRIPT_LOG_FILE}
```

```
2asmshut >> ${SCRIPT_LOG_FILE}

cp $ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log
$OUT_DIR

echo "End TPC-H Benchmark SEQUENCE NUMBER:
$RUN_ID `date`" >> $SCRIPT_LOG_FILE

runuf1.sh
#!/bin/ksh
#
# $Header: runuf1.sh 25-oct-2001.15:56:04 mpoess Exp $
#
# runuf1.sh
#
# Copyright (c) 1999, 2001, Oracle Corporation. All rights
reserved.
#
# NAME
# runuf1.sh - <one-line expansion of the name>
#
# DESCRIPTION
# runuf1.sh -l [<path name for reports>] -u [<uid/passwd>]
# -p [<program>] <run_id> <scale factor> <pair
number>
# <parallelism>
# USAGE
# To execute UF1.
#
# NOTES
# <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
# mpoess 10/25/01 - change default directory for update
sets
# mpoess 10/17/01 - add support for external tables
# mpoess 08/15/99 - Creation
# mpoess 08/15/99 - Creation
#
#
. $KIT_DIR/env
O=${ORACLE_HOME}
UPDATE_DIR=${KIT_DIR}/update
SCRIPT_DIR=${UPDATE_DIR}/scripts
UTILS_DIR=${KIT_DIR}/utils
LOG_DIR=${UPDATE_DIR}/log
GTIME=${UTILS_DIR}/gtime
SF=${SCALE_FACTOR}
PAR_HINT=16

LOGPATH=.
PASSWORD=${DATABASE_USER}

if [ $# -lt 1 ];
then
echo runuf1.sh setnum
exit 1
fi
SETNUM=$1
```

Appendix F

```
i=1
PID=""

# perform the update function 1

START=`$GTIME`

# first create the temp tables

sqlplus /NOLOG << !

connect $PASSWD;
set timing on
set serveroutput on
set echo on

drop directory data_dir;
create directory data_dir as '/ff1/';

drop table temp_l_et;
create table temp_l_et(
  l_orderkey      number ,
  l_partkey       number ,
  l_suppkey       number ,
  l_linenumbr     number ,
  l_quantity      number ,
  l_extendedprice number ,
  l_discount      number ,
  l_tax           number ,
  l_returnflag    char(1) ,
  l_linestatus    char(1) ,
  l_shipdate      date ,
  l_commitdate    date ,
  l_receiptdate   date ,
  l_shipinstruct  char(25) ,
  l_shipmode      char(10) ,
  l_comment       varchar(44)
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
  records delimited by newline
  nobadfile
  nologfile
  fields terminated by '|'
  missing field values are null
)
location (
'orders.tbl.u${SETNUM}'
))
reject limit unlimited;
alter table temp_l_et parallel ${PAR_HINT};
alter table temp_o_et parallel ${PAR_HINT};

alter session force parallel dml parallel (degree
${PAR_HINT});
alter session set isolation_level = serializable;
alter session set optimizer_index_cost_adj = 10;

insert into orders (
select
  o_orderdate      ,
  o_orderkey       ,
  o_custkey        ,
  o_orderpriority  ,
  o_shippriority   ,
  o_clerk          ,
  o_orderstatus    ,
  o_totalprice     ,
  o_comment
from temp_o_et);

insert into lineitem (
select
  l_shipdate      ,
  l_orderkey       ,
  l_discount       ,
  l_extendedprice ,
  l_suppkey        ,
  l_quantity       ,
  l_returnflag     ,
  l_partkey        ,
  l_linestatus     ,
  l_tax            ,
  l_commitdate     ,
  l_receiptdate    ,
  l_shipmode       ,
```

Appendix F

```
l_linenumber      ,
l_shipinstruct    ,
l_comment
from temp_l_et);
commit;
drop table temp_l_et;
drop table temp_o_et;

exit;
!

END=`$GTIME`

# Done

echo ""
echo "Update Function 1 Set $SETNUM done!"
echo "Elapsed Time is `echo $END - $START | bc`"
echo ""

runuf2.sh
#!/bin/ksh
#
# $Header: runuf2.sh 25-oct-2001.15:56:05 mpoess Exp $
#
# runuf2.sh
#
# Copyright (c) 1999, 2001, Oracle Corporation. All rights
reserved.
#
# NAME
#   runuf2.sh - <one-line expansion of the name>
#
# DESCRIPTION
#   runuf2.sh [-u <uid/passwd to login>] [-p <program>]
<run_id>
#       <scale factor> <pair number> <parallelism>
# USAGE
#   To execute UF2.
#
# NOTES
#   <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
#   mpoess   10/25/01 - change default directory for update
sets
#   mpoess   10/17/01 - add support for external tables
#   mpoess   08/15/99 - Creation
#   mpoess   08/15/99 - Creation
#
. $KIT_DIR/env
UPDATE_DIR=${KIT_DIR}/update
SCRIPT_DIR=${UPDATE_DIR}/scripts
UTILS_DIR=${KIT_DIR}/utils
GTIME=${UTILS_DIR}/gtime
LOG_DIR=${UPDATE_DIR}/log
PAR_HINT=16
SF=${SCALE_FACTOR}
```

```
PASSWD=${DATABASE_USER}

if [ $# -lt 1 ]
then
    usage
    exit 1
fi

SETNUM=$1

i=1
PID=""

START=`$GTIME`
# first create the temp tables

sqlplus /NOLOG << !

connect $PASSWD;
set timing on
set serveroutput on
set echo on

drop directory data_dir;
create directory data_dir as '/ff1/';

drop table temp_okey_et;
drop table temp_okey;

create table temp_okey_et(
    t_orderkey      number
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
    records delimited by newline
    nobadfile
    nologfile
    fields terminated by '|'
    missing field values are null
)
location (
'delete.${SETNUM}')
reject limit unlimited;

alter table temp_okey_et parallel ${PAR_HINT};

create table temp_okey parallel ${PAR_HINT} nologging as
select * from temp_okey
_et;

create unique index i_temp_okey on temp_okey (t_orderkey)
parallel ${PAR_HINT}
nologging compute statistics;

analyze table temp_okey estimate statistics sample 2 percent;
```

Appendix F

```
alter session force parallel dml parallel ${PAR_HINT};
alter session set isolation_level=serializable;
alter session set optimizer_index_cost_adj = 10;
delete from (select /*+ use_nl(o) */ o.rowid from orders o,
temp_okey t where o.
o_orderkey = t.t_orderkey order by 1);

delete from (select /*+ use_nl(l) */ l.rowid from lineitem
l,temp_okey t where l
.l_orderkey = t.t_orderkey order by 1);

commit;
drop table temp_okey;
drop table temp_okey_et;
exit;
!
```

END=`\$GTIME`

Done

```
echo ""
echo "Update Function 2 Set $SETNUM done!"
echo "Elapsed Time is `echo $END - $START | bc`"
echo ""
```

qexecpl.c

```
#ifdef RCSID
static char *RCSid =
"$Header: qexecpl.c 17-oct-2001.09:29:47 mpoess Exp $ ";
#endif /* RCSID */

/* Copyright (c) 1999, 2001, Oracle Corporation. All rights
reserved. */

/*

NAME
    qexecpl.c - <one-line expansion of the name>

DESCRIPTION
    SQL Execution Engine, Oracle v8, OCI version

PRIVATE FUNCTION(S)
    <list of static functions defined in .c file - with one-line
descriptions>

MODIFIED (MM/DD/YY)
    mpoess 10/17/01 - add serialization level in SQLinit
    mpoess 02/22/01 - add linux changes
    mpoess 08/05/99 - make compile
    mpoess 11/13/98 - fix pdll statement
    pswong 02/19/97 - migrating to version 8
    pswong 04/02/96 - more polishing
    pswong 03/25/96 - polish up
    pswong 03/06/96 - created

*/
```

```
#include <stdio.h>
#include <string.h>
#include <setjmp.h>
#include <sys/param.h>
#include <errno.h>
#include <math.h>
#include <string.h>
#include <sys/types.h>
#include <time.h>
#include <stdlib.h>

#include "qexecpl.h"

/* Function Prototypes */

extern double gettimeofday();

/* function prototypes from gen.c */

int get_statement();

/* Declare error handling functions */

void sql_error();

/* Other prototypes */

int define_output_variables();
void process_select_list();
void usage();
void SQLinit();
void SQLexec();
void SQLexit();
void *memalloc();
void print_header();
void print_rows();
int OFEN();
void remove_newline();

char logname[UNAME_LEN]; /* username/passwd combo */
char *passwd;

double tr_start = 0.0; /* query start time */
double tr_end = 0.0; /* query end time */

double s_tr_start = 0.0; /* statement start time */
double s_tr_end = 0.0; /* statement end time */

/* For our purpose of timing, we will treat comments as
delimiters */
/* for queries. Thus, we will collect query timings whenever
we */
/* encounter a comment (of course not for the first comment
in a */
/* file). */

int end_flag = 0; /* flag to indicate that we have reached
*/
```

Appendix F

```

/* the end of a query */

int stmt_cnt = 0; /* Number of statements processed.
*/
int qry_cnt = 0; /* Number of query processed. */

double product = 1.0; /* cumulative product of query times
*/
int rows_ret = 0; /* the number of rows fetched */
int num_sel_list = 0; /* the number of select list item */

long num_to_fetch = -1; /* Number of rows to fetch. -1
means fetch all */

slist slist[MAX_SEL_LIST]; /* Array for describing Select
List */
dlist *dlist[MAX_SEL_LIST]; /* Array of ptrs for Defining
Select List */

char stmt[SQL_LEN]; /* The SQL statement or comment
line. */
char qn[4]; /* Number of the query being executed
*/
char qnp[4]; /* Number of the previous query
executed */
char cmnt[5000]; /* Buffer to save the comment.
*/
#ifdef LINUX
FILE *qtemp; /* fd for query template */
FILE *logfile; /* log and report files */
FILE *rep;
#else
FILE *qtemp = stdin; /* fd for query template */
FILE *logfile = stdout; /* log and report files */
FILE *rep = stdout;
#endif
void *defbuf; /* Buffer pointer for ODEFIN */
int deflen = 0; /* Size of data type for ODEFIN */
int deftype = 1; /* Oracle type number for ODEFIN
*/

int pfmem = PFMEMSIZE; /* Memory to prefetch rows
*/

time_t tim; /* To get wall clock time */

/* OCI handles */

OCIEnv *tpcenv = NULL;
OCIServer *tpcsrv = NULL;
OCIError *errhp = NULL;
OCISvcCtx *tpcsvc = NULL;
OCISession *tpcusr = NULL;
OCISmt *curq = NULL;
OCISmt *cur_dml = NULL;
OCISmt *cur_ddl = NULL;
OCIParm *tpcpar = NULL;

sword status = OCI_SUCCESS; /* OCI return value */

```

```

/* usage: prints the usage of the program */

void usage() {

    fprintf(stderr, "\nUsage: qexec username/password [q<path
name for query template file>]\n");
    fprintf(stderr, "          [l<path name for log>] [r<path
name for reports>]\n\n");
    fprintf(stderr, "Options:\n");
    fprintf(stderr, "q<path for query>      : full path name for the
query template file.\n");
    fprintf(stderr, "          (default is stdin)\n");
    fprintf(stderr, "l<path name for log>      : full path name for
log files\n");
    fprintf(stderr, "          (default is stdout)\n");
    fprintf(stderr, "r<path name for reports> : full path name for
reports\n");
    fprintf(stderr, "          (default is stdout)\n");
    exit(-1);
}

/* type: 0 if environment handle is passed, 1 if error handle is
passwd */

void sql_error(errhp, status, type)
    OCIError *errhp;
    sword status;
    sword type;
{
    char msg[2048];
    ub4 errcode;
    ub4 msglen;
    int i, j;

    switch(status) {
    case OCI_SUCCESS_WITH_INFO:
        fprintf(stderr, "Error: Statement returned with info.\n");
        if (type)
            (void)
OCIErrorGet(errhp, 1, NULL, (sb4*)&errcode, (text*)msg,
                2048, OCI_HTYPE_ERROR);
        else
            (void)
OCIErrorGet(errhp, 1, NULL, (sb4*)&errcode, (text*)msg,
                2048, OCI_HTYPE_ENV);
        fprintf(stderr, "%s\n", msg);
        break;
    case OCI_ERROR:
        fprintf(stderr, "Error: OCI call error.\n");
        if (type)
            (void)
OCIErrorGet(errhp, 1, NULL, (sb4*)&errcode, (text*)msg,
                2048, OCI_HTYPE_ERROR);
        else
            (void)
OCIErrorGet(errhp, 1, NULL, (sb4*)&errcode, (text*)msg,

```

Appendix F

```

        2048,OCI_HTYPE_ENV);
    fprintf(stderr,"%s\n",msg);
    break;
case OCI_INVALID_HANDLE:
    fprintf(stderr, "Error: Invalid Handle.\n");
    if (type)
        (void)
OCIErrorGet(errhp,1,NULL,(sb4*)&errcode,(text*)msg,
            2048,OCI_HTYPE_ERROR);
    else
        (void)
OCIErrorGet(errhp,1,NULL,(sb4*)&errcode,(text*)msg,
            2048,OCI_HTYPE_ENV);
    fprintf(stderr,"%s\n",msg);
    break;
}

/* Rollback just in case */

(void) OCITransRollback(tpcsvc,errhp,OCI_DEFAULT);

fprintf(stderr, "Exiting Oracle...\n");
fflush(stderr);

SQLexit();

exit(1);
}

#ifdef LINUX
int main(argc,argv)
#else
void main(argc,argv)
#endif
    int argc;
    char *argv[];
{
    int i,pos,pos2;
    int retcode; /* Return code for get_statement */
#ifdef LINUX
    logfile=fopen("/dev/stdout","w");
    qtemp=fopen("/dev/stdin","rw");
    rep=fopen("/dev/stdout","w");
#endif
    /* Initialize some variables */

    if ((argc > 5) || (argc < 2)) {
        usage();
    }

    /* argv[1] -- User and Password for Database */

    strcpy(logname, argv[1]);

    /* Process optional parameters */

    argc -= 1;
    argv += 1;

while(--argc) {
    ++argv;
    switch(argv[0][0]) {
    case 'q':
        if ((qtemp = fopen(++(argv[0]),"r")) == NULL) {
            fprintf(stderr,"Unable to open file '%s'\n", argv[0]);
            fprintf(stderr,"%s: %s\n", argv[0], strerror(errno));
            exit(-1);
        }
        break;
    case 'r':
        if ((rep = fopen(++(argv[0]),"a")) == NULL) {
            fprintf(stderr,"Unable to open file '%s'\n", argv[0]);
            fprintf(stderr,"%s: %s\n", argv[0], strerror(errno));
            exit(-1);
        }
        break;
    case 'l':
        if ((logfile = fopen(++(argv[0]),"a")) == NULL) {
            fprintf(stderr,"Unable to open file '%s'\n", argv[0]);
            fprintf(stderr,"%s: %s\n", argv[0], strerror(errno));
            exit(-1);
        }
        break;
    default:
        fprintf(stderr,"Invalid Option: %c\n", argv[0][0]);
        usage();
        break;
    }
}

/* Do some initialization and establish connection with the
database */

SQLinit();

/* May want to add some triggering mechanism here */

time(&tim);
fprintf(logfile, "Begin Execution at %s\n\n", ctime(&tim));
fprintf(rep, "Begin Executing this Stream at %s\n\n",
ctime(&tim));
/* Get the next statement and start processing it */

while ((retcode = get_statement()) > 0) {

    switch (retcode) {

        /* If this is a comment, skips it */
    case COMMENT:
        /*if (end_flag) {
            end_flag = 0; /* /* reset query end flag */
            /* save the comment so that we can print it out later
on */
            /* strcpy(cmnt, stmt);
            break;
        } */
        if (stmt[3]=='@') {

```

Appendix F

```

pos=4;
strcpy(qnp,qn);
while (stmt[pos] != ')') {
    pos++;
}
pos2=0;
pos++;
while (stmt[pos] != '.') {
    /*printf ("qn %d %c \n",pos2,stmt[pos]);*/
    qn[pos2]=stmt[pos];
    pos2++;
    pos++;
}
qn[pos2] = 0;
/* printf("found a new query: %s\n",qn); */
}
/* save the comment so that we can print it out later on */
strcat(cmnt, stmt);
break;

/* if this is a set_row_fetch command */
case SET_FETCHROW:
    fprintf(logfile,"Setting the number of rows to fetch to:
%ld\n\n",
            num_to_fetch);
    break;

/* if this is a SQL statement */
case SQL_STMT:

    /* Executes the query */
    SQLexec();

    stmt_cnt++;
    qry_cnt++;
    fflush(rep);
    fflush(logfile);
    /*
    fprintf(logfile,"\nStatement Started at %.2f\n", s_tr_start);
    fprintf(logfile,"Statement Ended at %.2f\n", s_tr_end);

    fprintf(logfile,"Statement Processed in %.2f seconds.\n",
            (s_tr_end - s_tr_start));
    fprintf(rep, "Query %s: Execution Time: %.2f started %.2f
ended %.2f\n",
            qn,(s_tr_end - s_tr_start)s_tr_start,s_tr_end);
    fflush(rep);
    fflush(logfile);*/
    break;

/* Should never reach here */
default:
    fprintf(stderr, "Invalid statement type!!\n");
    SQLexit();
    break;
}
}

/* Get Timing for the last query */
tr_end = gettimeofday();

fprintf(logfile,"Query Processed in %.2f
seconds.\n\n",(tr_end - s_tr_start));

/* print comments for this query that we have saved */
/* fprintf(logfile, "%s\n", cmnt); */

/* fprintf(rep, "Query %s : Execution time %.2f\n",
qn,(tr_end - s_tr_start));*/
fprintf(rep, "Query %s: Execution Time: %.2f started %.2f
ended %.2f\n",
        qn,(tr_end - s_tr_start),s_tr_start,tr_end);

time(&tim);
fprintf(logfile,"\nEnded Executing this Stream at %s\n",
ctime(&tim));
fprintf(logfile,"\nStream Started at %.2f\n", tr_start);
fprintf(logfile,"Stream Ended at %.2f\n", tr_end);
fprintf(logfile,"Stream Processed in %.2f
seconds\n\n",(tr_end - tr_start));

fprintf(rep,"\nEnded Executing this Stream at %s\n",
ctime(&tim));
fprintf(rep,"\nStream Started at %.2f\n", tr_start);
fprintf(rep,"Stream Ended at %.2f\n", tr_end);
fprintf(rep,"Stream Processed in %.2f seconds\n\n",
        (tr_end - tr_start));

fprintf(logfile, "\nSQL statements processed: %d\n",
stmt_cnt);
/*fprintf(logfile, "Queries processed: %d\n", qry_cnt);*/

fflush(rep);
fflush(logfile);

/* Close the query template file */
fclose(qtemp);

/* Disconnect from ORACLE. */
SQLexit();
exit(0);
}

/* SQLinit(): Perform initialization tasks. */
/* Logs on to Oracle, opens some files and open a
cursor for */
/* later use. */

void SQLinit() {

    int i;

```

Appendix F

```

/* preallocate MAX_PREALLOC members of the dlist array
*/
/* initializes others to NULL so that we can determine who
to free later */

for (i=0; i<MAX_SEL_LIST; i++) {
    if (i < MAX_PREALLOC) {
        dlist[i] = (dlttype *) memalloc (sizeof(dlttype));
        dlist[i]->defhdl = NULL;
    }
    /* OCIhalloccurq,&(dlist[i]-
>defhdl),OCI_HTYPE_DEFINE); */
    else
        dlist[i] = NULL;
}

/* Connect to ORACLE. Program will call sql_error()
*/
/* if an error occurs in connecting to the default database. */

(void) OCIInitialize(OCI_DEFAULT,(dvoid *)0,0,0,0);

if((status=OCIEnvInit((OCIEnv
**) &tpcenv,OCI_DEFAULT,0,(dvoid **)0)) !=
OCI_SUCCESS)
    sql_error(tpcenv, status, 0);

OCIhallocc(tpcenv,&errhp,OCI_HTYPE_ERROR);
OCIhallocc(tpcenv,&curq,OCI_HTYPE_STMT);
OCIhallocc(tpcenv,&cur_dml,OCI_HTYPE_STMT);
OCIhallocc(tpcenv,&cur_ddl,OCI_HTYPE_STMT);
OCIhallocc(tpcenv,&tpcsvc,OCI_HTYPE_SVCCTX);
OCIhallocc(tpcenv,&tpcsrv,OCI_HTYPE_SERVER);
OCIhallocc(tpcenv,&tpcusr,OCI_HTYPE_SESSION);

/* get username and password */

passwd = strchr(logname, '/');
*passwd = '\0';
passwd++;

if ((status = OCIServerAttach(tpcsrv,errhp,(text
*)0,0,OCI_DEFAULT)) != OCI_SUCCESS)
    sql_error(errhp,status,1);

OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcsrv,0,OCI_ATTR
_SERVER,errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,logname,strlen(logna
me),OCI_ATTR_USERNAME,
errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(passw
d),OCI_ATTR_PASSWORD,
errhp);

if ((status = OCISessionBegin(tpcsvc, errhp, tpcusr,
OCI_CRED_RDBMS,
OCI_DEFAULT)) !=
OCI_SUCCESS)
    sql_error(errhp,status,1);

/* SQLexec() Executes the SQL statement.
*/
/* Parse the SQL statement. */
/* If DDL or DML statements, execute right away. */
/* Else describe and define select list outputs, */
/* execute and fetch results. */

void SQLexec()
{
    int i;
    ub2 stmttyp = OCI_STMT_SELECT; /* default is a
SELECT statement */

    /* Clause 5.3.6.2: QI(i,s) is the time between the first
character */
    /* of this query text is submitted and the first */
    /* character of the next query text is submitted. */

    if (qry_cnt) {
        time(&tim);
        s_tr_end = gettime();
        fprintf(logfile,"Query Processed in %.2f seconds.\n\n",
(s_tr_end - s_tr_start));

        /* print comments for this query that we have saved */
        /* fprintf(logfile, "%s\n", cmnt); */

        /*fprintf(rep, "Query %s : Execution time %.2f\n",
qnp,(s_tr_end - s_tr_start));*/
        fprintf(rep, "Query %s: Execution Time: %.2f started %.2f
ended %.2f\n",
qnp,(s_tr_end - s_tr_start),s_tr_start,s_tr_end);
    }
}

```


Appendix F

```

/* Let's fflush stuff so that we can see what's going on */
/* Fix for Q15 */
fflush(logfile);
fflush(rep);

strcpy(qnp,qn);

fflush(logfile);
fflush(rep);
}
else
    tr_start = gettime();

s_tr_start = gettime();

/* prepare the statement */

if ((status = OCIStmtPrepare(curq, errhp, (text*) stmt, (ub4)
strlen(stmt),
                                OCI_NTV_SYNTAX,
OCI_DEFAULT)) != OCI_SUCCESS)
    sql_error(errhp,status,1);

/* Prints the query text and comment to the logfile */

fprintf(logfile, "\n%s\n", cmnt);
cmnt[0]=0;
fprintf(logfile, "\n%s\n", stmt);

/* if this is a DDL or DML statement, execute it right away
*/
/* only worries about SELECT statements right now, cannot
*/
/* execute a stored PL/SQL procedure in this version */

OCIaget(curq,OCI_HTYPE_STMT,&stmttp, NULL,OCI_AT
TR_STMT_TYPE,errhp);

if (stmttp != OCI_STMT_SELECT) {
    OCIsexec(tpcsvc,curq,errhp,1);
    return;
}

/* otherwise, this is a select statement */
/* Describe and define output variables */

/* first let's execute it to get the select-list definition */

OCIaset(curq, OCI_HTYPE_STMT, &pfmem, 0,
OCI_ATTR_PREFETCH_MEMORY, errhp);

OCIsexec(tpcsvc,curq,errhp,0);

num_sel_list = define_output_variables();

/* Executes the query and fetches the rows */
(void) process_select_list(num_sel_list);

/* Need to get the number of rows fetched first */
/* since the following statments will screw it up */

OCIaget(curq,OCI_HTYPE_STMT,&rows_ret,NULL,OCI_A
TTR_ROW_COUNT,errhp);

/* To control memory usage, let's free up the extra dlist
entries */
/* that we have allocated. */

i=MAX_PREALLOC;
while(dlist[i] != NULL) {
    free(dlist[i]);
    dlist[i++] = NULL;
}

/* reset set_fetchrows */

num_to_fetch = -1;

}

void SQLexit() {

    int i;

    OCILogoff(tpcsvc,errhp);
    OCIhfree(tpcenv,OCI_HTYPE_STMT);
    OCIhfree(tpcsvc,OCI_HTYPE_SVCCTX);
    OCIhfree(tpcsrv,OCI_HTYPE_SERVER);
    OCIhfree(tpcusr,OCI_HTYPE_SESSION);

    /* free all memory */

    for (i=0; i<MAX_SEL_LIST; i++) {
        if (dlist[i] != NULL) {
            free(dlist[i]);
        }
    }

    /* Flush all output */

    fflush(rep);
    fflush(logfile);

}

/* define_output_variables(): Describe and define select-list
items for */
/* a query statement. */
/* Returns the number of select-list items */
/* for this query. */

int define_output_variables()

```

Appendix F

```

{
    int i;
    int retflag = 0;

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

        slist[i].buflen = MAX_COLNAME_SIZE;

        if (OCIParamGet(curq, OCI_HTYPE_STMT, errhp, (dvoid
***) &tpcpar,
                POS(i)) != OCI_SUCCESS)
            break;

        /* dsize and nullok fields of dlist not used */

        OCIaget(tpcpar, OCI_DTYPE_PARAM, &(slist[i].dbsize),
                NULL, OCI_ATTR_DATA_SIZE, errhp);
        OCIaget(tpcpar, OCI_DTYPE_PARAM, &(slist[i].dbtype),
                NULL, OCI_ATTR_DATA_TYPE, errhp);
        OCIaget(tpcpar, OCI_DTYPE_PARAM, &(slist[i].buf),
                &(slist[i].buflen), OCI_ATTR_NAME, errhp);
        OCIaget(tpcpar, OCI_DTYPE_PARAM,
&(slist[i].precision),
                NULL, OCI_ATTR_PRECISION, errhp);
        OCIaget(tpcpar, OCI_DTYPE_PARAM, &(slist[i].scale),
                NULL, OCI_ATTR_SCALE, errhp);

        /* For formatting purpose, remove trailing blanks in select-
list name. */

        /*
        if (slist[i].buflen < MAX_COLNAME_SIZE)
            (slist[i].buf)[slist[i].buflen] = '\0';
        */

        /* Well, we need to allocate for entries for dlist */

        if (i >= MAX_PREALLOC) {
            dlist[i] = (dtype *) memalloc(sizeof(dtype));
            dlist[i]->defhdl = NULL;
        }

        /* Let's check the sizes and types for this select list item */

        switch (slist[i].dbtype) {

        case OCI_TYPECODE_NUMBER:

            /* The odescr will not give a good estimate to the scale if
            */
            /* no scale was given in the Oracle table definition. */

#ifdef HAVE_SCALE
            if (slist[i].scale != 0) {
                defbuf = (double *) dlist[i]->fbuf;
                deflen = FLT;
                deftype = OCI_TYPECODE_DOUBLE;
                slist[i].dbtype = OCI_TYPECODE_DOUBLE;
            } else {
                defbuf = (int *) dlist[i]->ibuf;
                deflen = INT;
                deftype = OCI_TYPECODE_INTEGER;
                slist[i].dbtype = OCI_TYPECODE_INTEGER;
            }
#else
            defbuf = (double *) dlist[i]->fbuf;
            deflen = FLT;
            deftype = OCI_TYPECODE_FLOAT;
            slist[i].dbtype = OCI_TYPECODE_FLOAT;
#endif /* HAVE_SCALE */

            break;

        default:

            /* default is character string */

            defbuf = (char **) dlist[i]->sbuf;
            deflen = MAX_STR_LEN;
            deftype = SQLT_STR;
            /* deftype = OCI_TYPECODE_CHAR; */
            break;
        }

        /* Define the column */

        if ((status=OCIDefineByPos(curq,&(dlist[i]-
>defhdl),errhp,POS(i),
                defbuf,deflen,deftype,NULL,
                dlist[i]-
>rlen,NULL,OCI_DEFAULT))!=OCI_SUCCESS)
            sql_error(errhp,status,1);
        }
        return i;
    }

    /* process_select_list(): Fetch rows from a query. */

    void process_select_list(num)
        int num; /* number of select list items */
    {
        int i,j;
        int ntf;
        int num_so_far;
        sword stats = OCI_SUCCESS;

        /* Print the headers for the query execution result */

        print_header(num);

        /* See if we need to limit the rows to fetch */

        ntf = (num_to_fetch >= 0) ? num_to_fetch : MAX_ARRAY;

        /* Fetch the rows and print them out */
    }
}

```

Appendix F

```

if ((ntf > MAX_ARRAY) || (num_to_fetch == -1)) {

    stats = OCISstmtFetch(curq, errhp, MAX_ARRAY,
OCI_FETCH_NEXT, OCI_DEFAULT);

OCIaget(curq,OCI_HTYPE_STMT,&rows_ret,NULL,OCI_A
TTR_ROW_COUNT,errhp);

    print_rows(num,rows_ret);

    /* To avoid 1022 from OFEN */
    /* More rows to fetch... */

    if (stats != OCI_NO_DATA) {
        if (num_to_fetch == -1) {
            while ((stats =
OCIISstmtFetch(curq,errhp,MAX_ARRAY,OCI_FETCH_NEX
T,
OCI_DEFAULT)) ==
OCI_SUCCESS) {
OCIaget(curq,OCI_HTYPE_STMT,&num_so_far,NULL,
OCI_ATTR_ROW_COUNT,errhp);
    print_rows(num,(num_so_far-rows_ret));
    rows_ret = num_so_far;
    }
    /* Print the final rows */
    OCIaget(curq,OCI_HTYPE_STMT,&num_so_far,N
ULL,
OCI_ATTR_ROW_COUNT,errhp);
    print_rows(num,(num_so_far-rows_ret));
    rows_ret = num_so_far;
    } else {
        ntf -= MAX_ARRAY;

        while ((stats = OCISstmtFetch(curq,errhp,
((ntf>MAX_ARRAY)
? MAX_ARRAY:ntf),
OCI_FETCH_NEXT,
OCI_DEFAULT)) ==
OCI_SUCCESS) {
            ntf -= MAX_ARRAY;

OCIaget(curq,OCI_HTYPE_STMT,&num_so_far,NULL,
OCI_ATTR_ROW_COUNT,errhp);
    print_rows(num,(num_so_far-rows_ret));
    rows_ret = num_so_far;
    if (ntf <= 0) break;
    }
    OCIaget(curq,OCI_HTYPE_STMT,&num_so_far,N
ULL,
OCI_ATTR_ROW_COUNT,errhp);
    print_rows(num,(num_so_far-rows_ret));
    rows_ret = num_so_far;
    }
    } else {

```

```

OCIISstmtFetch(curq, errhp, ntf, OCI_FETCH_NEXT,
OCI_DEFAULT);

OCIaget(curq,OCI_HTYPE_STMT,&rows_ret,NULL,OCI_A
TTR_ROW_COUNT,errhp);
    print_rows(num,rows_ret);
    }

    fprintf(logfile, "\n\n%d %s processed.\n", rows_ret,
rows_ret == 1 ? "row" : "rows");
}

int get_statement()
{
    char line[128];
    char *pos, *str;

    /* Reset statement buffer */

    stmt[0] = '\0';

    while (fgets(line, 127, qtemp) != NULL) {

        /* skip blank lines */
        if (line[0] == '\n')
            continue;

        /* remove blanks */

        str = line;

        while (*str == ' ') str++;

        /* Let's get the line together first */

        strcat(stmt, str);

        /* if this is a comment line */
        if ((str[0] == '-') && (str[1] == '-'))
            return COMMENT;

        /* see if this is a set_fetchrows line */
        if (strncmp(str, "set_fetchrows", 13) == 0) {
            pos = strchr(str, ';');
            *pos = '\0';
            pos = strchr(str, '=');
            num_to_fetch = atol(++pos);
            return SET_FETCHROW;
        }

        /* if this is the end of the current statement */
        if ((pos = strchr(stmt, ';')) != NULL) {
            *pos = '\0';
            return SQL_STMT;
        }
    }
}

```

Appendix F

```

return END_OF_FILE;
}

/* memalloc(): Allocates memory, exit program if we have a
problem. */

void *memalloc(size)
    int size;
{
    void *tmp;

    if ((tmp = (void *) malloc(size)) == NULL) {
        fprintf(stderr, "Error in malloc\n");
        SQLexit();
        return NULL;    /* should never reach here */
    } else {
        return tmp;
    }
}

void print_header(nsel)
    int nsel;    /* Number of select list items */
{
    int i, diff;
    char colname[MAX_COLNAME_SIZE];
    int len = 0;    /* Running column length */
    int cwid = 0;

    fprintf(logfile, "\n");

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

        /* extract the column name */

        strncpy((char *)colname, (char *)slist[i].buf, slist[i].buflen);
        colname[slist[i].buflen] = '\0';

        /* format the output a little */

        cwid = MAX(slist[i].dbsize, slist[i].buflen);

        /* do a little bit of formatting */

        if (cwid > 80) {
            fprintf(logfile, "\n");
            len = 0;
        } else if ((len += cwid) > 80) {
            fprintf(logfile, "\n");
            len = cwid;
        }
#ifdef FORMAT1
        if ((slist[i].dbtype == INT_TYPE) || (slist[i].dbtype ==
FLT_TYPE))
            fprintf(logfile, "%*s ", cwid, slist[i].buf);
            else /* string type */
                fprintf(logfile, "%*s ", -cwid, slist[i].buf);
        #else
            fprintf(logfile, "%*s ", -cwid, colname);
        #endif /* FORMAT1 */
    }

    fprintf(logfile, "\n");
}

void print_rows(ncol, nrow)
    int ncol;
    int nrow;
{
    int i, j;
    int len;
    int diff;
    int cwid;

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

        len = 0;

        for (j=0; j<ncol; j++) {

            cwid = MAX(slist[j].dbsize, slist[j].buflen);

            /* do a little bit of formatting */

            if (cwid > 80) {
                fprintf(logfile, "\n");
                len = 0;
            } else if ((len += cwid) > 80) {
                fprintf(logfile, "\n");
                len = cwid;
            }

            switch(slist[j].dbtype) {
                case INT_TYPE:
#ifdef HAVE_SCALE
                    fprintf(logfile, "%*ld", cwid, (dlist[j]-
>ibuf)[i]);
                    break;
#endif /* HAVE_SCALE */
                case FLT_TYPE:
#ifdef FORMAT1
                    fprintf(logfile, "%*.2f ", cwid, (dlist[j]->fbuf)[i]);
                #else
                    fprintf(logfile, "%*.2f ", -cwid, (dlist[j]->fbuf)[i]);
                #endif /* FORMAT1 */
                default:
                    fprintf(logfile, "%*s ", -(cwid), (dlist[j]->sbuf)[i]);
                    break;
            }
        }

        fprintf(logfile, "\n");
    }
}

```

Appendix F

```

}
}

/* remove_newline(): Remove newline character from str. */

void remove_newline(str)
    char *str;
{
    char *p;

    while ((p = strchr(str, '\n')) != NULL)
        *p = ' ';
}

qexecpl.h
/*
 * $Header: qexecpl.h 13-nov-2001.17:52:35 mpoess Exp $
 */

/* Copyright (c) 1999, 2001, Oracle Corporation. All rights
reserved. */

/* NOTE: See 'header_template.doc' in the 'doc' dve under the
'forms'
    directory for the header file template that includes
instructions.
*/

/*
NAME
    qexecpl.h

DESCRIPTION
    SQL statement execution front-end header file.

PUBLIC FUNCTION(S)
    <list of external functions declared/defined - with one-line
descriptions>

PRIVATE FUNCTION(S)
    <list of static functions defined in .c file - with one-line
descriptions>

EXAMPLES
NOTES
    <other useful comments, qualifications, etc.>

MODIFIED (MM/DD/YY)
mpoess    11/13/01 - change DOP to 84 for DML and DDL
mpoess    02/22/01 - add linux changes
mpoess    08/05/99 - make compile
mpoess    07/15/99 - Creation
mpoess    07/15/99 - Creation

*/

/*
    #ifndef S_ORACLE
    #include <s.h>
    #endif
    */
    #ifndef QSTREAMPL_H
    #define QSTREAMPL_H

    #include <stdio.h>
    #include <string.h>
    #include <sys/param.h>
    #include <sys/types.h>
    #include <time.h>
    #include <errno.h>
    #include <math.h>

    #include <oratypes.h>

    #include <oratypes.h>

    #ifndef OCIDFN
    #include <ocidfn.h>
    #endif /* OCIDFN */

    #ifndef OCI_ORACLE
    #include <oci.h>
    #endif /* OCI_ORACLE */
    /*
    #ifdef __STDC__
    #include <ociapr.h>
    #else
    #include <ocikpr.h>
    #endif /* __STDC__ */

    /* some basic definitions */
    #define UNAME_LEN 64
    #define MAX_FILE_PATH_LEN 128

    #ifndef TRUE
    #define TRUE 1
    #endif /* TRUE */

    #ifndef FALSE
    #define FALSE 1
    #endif /* FALSE */
    #ifndef LINUX
    #define MAX(x,y) ((x >= y) ? x : y)
    #define MIN(x,y) ((x <= y) ? x : y)
    #endif
    /* defines and typedefs for parsing */

    #define CRT_TBL 1
    #define INS_STMT 3
    #define SEL_STMT 4
    #define UPD_STMT 5
    #define DRP_VIEW 7
    #define DRP_TBL 8
    #define DEL_STMT 9
    #define CRT_VIEW 10

```

Appendix F

```
/* defines and typedefs for query description */
```

```
#define MAX_COLNAME_SIZE 32 /* Maximum length of  
Column name */
```

```
#define MAX_SEL_LIST 16 /* Maximum items on a  
select list */
```

```
#define END_OF_LIST 1007 /* Error code when we reach  
the end of the */  
/* select list. */
```

```
/* types for describe */
```

```
#define CHAR_TYPE 1  
#define NUM_TYPE 2  
#define INT_TYPE 3  
#define FLT_TYPE 4  
#define STR_TYPE 5  
#define DATE_TYPE 12
```

```
#define NUMWIDTH 16 /* Width of the numeric fields  
*/
```

```
#define POS(i) (i+1) /* The position is 1...n instead */  
#define IND(i) (i-1) /* of 0..n-1 as in an array. */
```

```
typedef struct des
```

```
{  
    ub2 dbsize;  
    ub4 buflen;  
    /* sb2 dsize; */  
    sb4 scale;  
    /* sb2 nullok; */  
    OCITextCode dbtype;  
    /* text buf[MAX_COLNAME_SIZE]; */  
    text *buf;  
    ub1 precision;  
} sltype;
```

```
/* defines and typedefs for query select list definition */
```

```
#define MAX_ARRAY 50 /* Maximum array size for  
array fetch */
```

```
#define PFMEMSIZE 65536 /* Memory size of prefetch  
buffer */
```

```
#define MAX_STR_LEN 256 /* Maximum size for string  
variables */
```

```
#define MAX_PREALLOC 8 /* Maximum number of  
preallocated select list */  
/* definitions. */
```

```
#define INT sizeof(long)  
#define STR sizeof(char)  
#define FLT sizeof(double)  
#define FLTP (double *)  
#define INTP (long *)  
#define STRP (char **)
```

```
typedef struct def
```

```
{  
    long ibuf[MAX_ARRAY];  
    double fbuf[MAX_ARRAY];  
    char sbuf[MAX_ARRAY][MAX_STR_LEN];  
    ub2 rlen[MAX_ARRAY]; /* return length */  
    OCIDefine *defhdl;  
} dltype;
```

```
extern int errno;
```

```
#define SQL_LEN 2048
```

```
#ifndef NULL  
#define NULL 0  
#endif
```

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

```
#ifndef sword  
#define sword int  
#endif
```

```
#ifndef ub1  
#define ub1 unsigned char  
#endif
```

```
#define NA -1 /* ANSI SQL NULL */  
#define VER7 2  
#define NOT_SERIALIZABLE 8177 /* ORA-08177:  
transaction not serializable */
```

```
#define ADR(object) ((ub1 *)&(object))  
#define SIZ(object) ((sword)sizeof(object))  
#define SID(sid) ((sid == -1) ? 0 : sid)
```

```
/* For get_statement */
```

```
#define END_OF_FILE -1  
#define COMMENT 1  
#define SQL_STMT 2  
#define SET_FETCHROW 3
```

```
#define OCIhalloc(envh,hndl,htyp) \  
    if((status=OCIHandleAlloc((dvoid *)envh,(dvoid  
***)hndl,htyp,0,(dvoid ***)0))!  
=OCI_SUCCESS) \  
    sql_error(envh,status,0); \  
    else \  
    DISCARD 0
```

```
#define OCIhfree(hndl,htyp) \  
    OCIHandleFree(hndl,htyp);
```

Appendix F

```

if((status=OCIHandleFree((dvoid *)hndl,htyp)) ==
OCI_SUCCESS) \
    fprintf(stderr, "Error freeing handle of type %d\n", htyp)

```

```

#define OCIaget(hndl,htyp,attp,size,atyp,errh) \
    if((status=OCIAttrGet((dvoid *)hndl,htyp,(dvoid *)attp,(dvoid *)size,atyp,errh)) != OCI_SUCCESS) \
        sql_error(errh,status,1); \
    else \
        DISCARD 0

```

```

#define OCIaset(hndl,htyp,attp,size,atyp,errh) \
    if((status=OCIAttrSet((dvoid *)hndl,htyp,(dvoid *)attp,size,atyp,errh)) != OCI_SUCCESS) \
        sql_error(errh,status,1); \
    else \
        DISCARD 0

```

```

#define OCIsexec(svch,stmh,errh,iter) \

if((status=OCISmtExecute(svch,stmh,errh,iter,0,NULL,NUL
L,OCI_DEFAULT)) != OCI_SUCCESS) \
    sql_error(errh,status,1); \
else \
    DISCARD 0

```

```

#define ISOTXT "alter session set isolation_level =
serializable"
#define PDMLTXT "alter session force parallel dml parallel
(degree 84)"
#define PDDLTX "alter session force parallel ddl parallel
(degree 84)"

```

```

#endif /* QSTREAMPL_H */

```

2asmshut

```

#!/bin/ksh
. $FRAME_PATH/env
export ORACLE_SID=$ASM_SID
if [ "$1" = "abort" ]; then
for i in $SECONDARY_NODES
do
ssh $i -n $KIT_DIR/rasmshuta
done
sqlplus << !
connect / as sysdba
shutdown abort
exit
!
else
for i in $SECONDARY_NODES
do
ssh $i -n $KIT_DIR/rasmshut
done
sqlplus << !

```

```

connect / as sysdba
shutdown immediate
exit

```

```
!
```

```
fi
```

2asmstart

```

#!/bin/ksh

. $FRAME_PATH/env
export ORACLE_SID=$ASM_SID
sqlplus /NOLOG << EOF
!date
set timing on
connect / as sysdba
startup pfile=$ORACLE_HOME/dbs/init${ASM_SID}.ora
!date
exit
EOF

```

```

for i in $SECONDARY_NODES
do
ssh $i -n $KIT_DIR/rasmstart
done

```

2shut

```

#!/bin/ksh
. $FRAME_PATH/env

if [ "$1" = "abort" ]; then
for i in $SECONDARY_NODES
do
ssh $i -n /mnt/sdb2/home/oracle/frame/bin/tshut
done
sqlplus << !
connect / as sysdba
shutdown abort
exit
!
else
for i in $SECONDARY_NODES
do
ssh $i -n /mnt/sdb2/home/oracle/frame/bin/tshut abort
done
sqlplus << !
connect / as sysdba
shutdown immediate
exit
!
fi

```

2start

```

#!/bin/ksh

. $FRAME_PATH/env

tstart

```

Appendix F

```
for i in $SECONDARY_NODES
do
ssh $i -n /mnt/sdb2/home/oracle/frame/bin/tstart
done

tshut
#!/bin/ksh
#
# $Header: tshut.sh 08-aug-99.18:06:22 mpoess Exp $
#
# tshut.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
#   tshut.sh
#
# DESCRIPTION
#   shuts down a database
#
# NOTES
#   <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
#   mpoess   08/08/99 - Creation
#   mpoess   08/08/99 - Creation
#
#!/bin/ksh
```

```
if [ "$1" = "abort" ]; then
sqlplus /NOLOG<< !
connect / as sysdba
shutdown abort
exit
!
else
sqlplus /NOLOG<< !
connect / as sysdba
shutdown
exit
!
fi
```

```
tstart
#!/bin/ksh
#
# $Header: tstart.sh 08-aug-99.18:05:50 mpoess Exp $
#
# tstart.sh
#
# Copyright (c) Oracle Corporation 1999. All Rights
Reserved.
#
# NAME
#   tstart.sh
#
# DESCRIPTION
```

```
#   starts a database with a specific init.ora or uses the
default.
#
# NOTES
#   <other useful comments, qualifications, etc.>
#
# MODIFIED (MM/DD/YY)
#   mpoess   08/08/99 - Creation
#   mpoess   08/08/99 - Creation
#
#!/bin/ksh

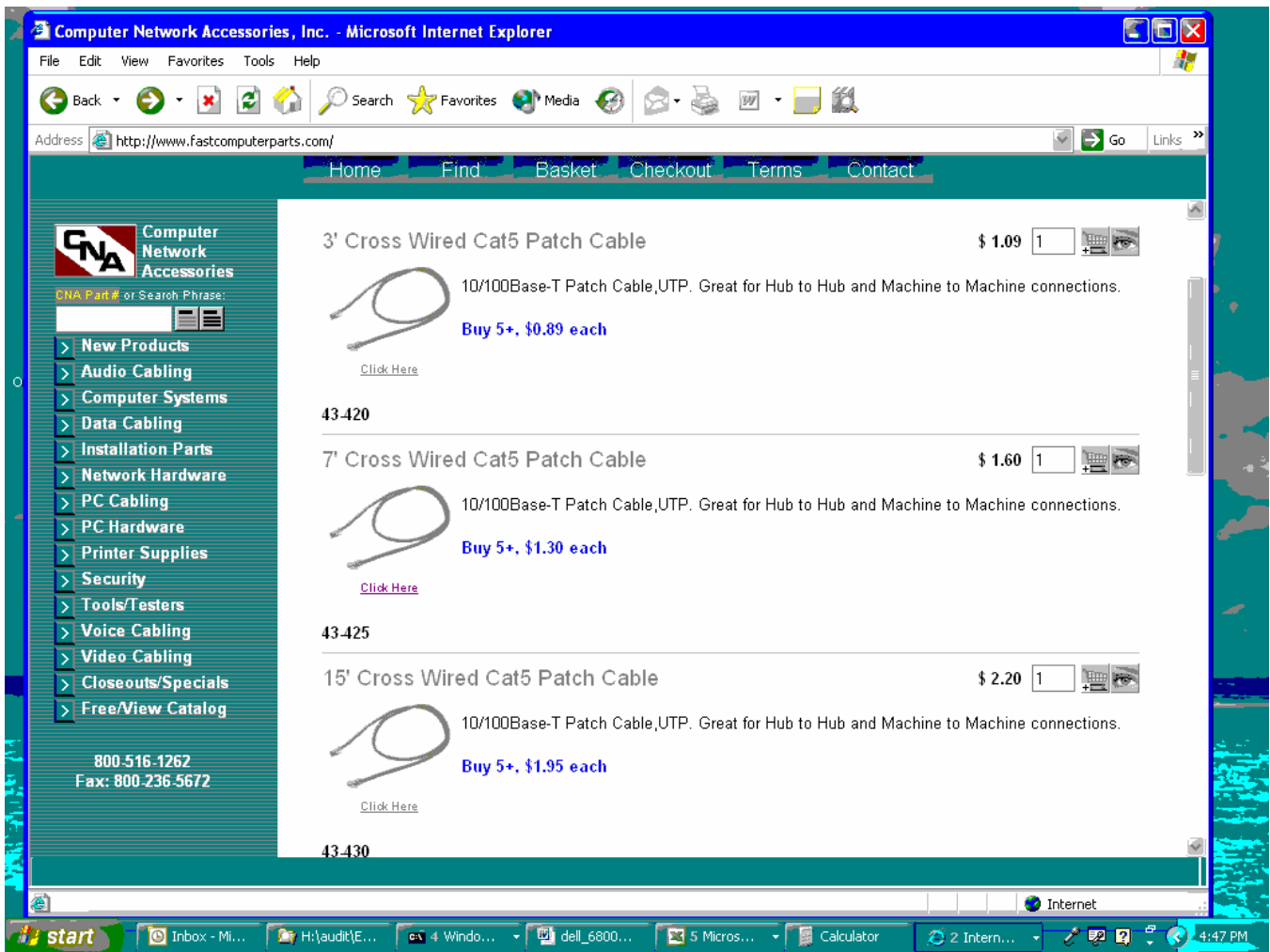
DIR=`pwd`
cd $ORACLE_HOME/dbs

if [ "$1" != "" ]; then
  PFILE="pfile=$ORACLE_HOME/dbs/$1.ora"
else
  PFILE="pfile=$ORACLE_HOME/dbs/init_${ORACLE_SID}.o
ra"
fi
```


Appendix F

Appendix G: Price Quotations

Appendix G



Appendix G

From: MaryBeth Pierantoni [mary.beth.pierantoni@oracle.com]

Sent: Thursday, June 23, 2005 6:19 PM

To: Wakou, Nicholas

Subject: Oracle Pricing

To follow is the Oracle pricing for Dell 2x4 cluster. Thank you.

Product	Quantity	Price	Extended Price
Oracle Database 10g Release 2 Enterprise Edition, Named User Plus for 3 years	8	\$10,000	\$80,000
Real Application Clusters, Named User Plus for 3 years	8	\$5,000	\$40,000
Partitioning, Named User Plus for 3 years	8	\$2,500	\$20,000
Database Server Support Package for 3 years	3 yrs	\$4,000	\$12,000
Oracle Mandatory E-Business Discount			<\$22,800>
Total			\$129,200

Oracle pricing contact: MaryBeth Pierantoni, mary.beth.pierantoni@oracle.com, 916-315-5081