



**TPC Benchmark™ H**  
**Full Disclosure Report**

---

---

**Sun Microsystems Sun Fire™ E25K**  
**Using Oracle Database 10g Enterprise Edition Release 2**  
**with Partitioning and Oracle Automatic Storage**  
**Management**

---

---

**Submitted for Review**  
**Report Date: November 29, 2005**

TPC Benchmark H Full Disclosure Report

First Printing

© 2005 Sun Microsystems, Inc.

901 San Antonio Road, Palo Alto, California 94303 U.S.A.

All rights reserved. This product and related documentation are protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or related documentation may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the United States Government is subject to the restrictions set forth in DFARS 252.227-7013 (c)(1)(ii) and FAR 52.227-19, Rights in Technical Data and Computer Software (October 1988).

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

#### TRADEMARKS

Sun, Sun Microsystems, the Sun logo, Sun Fire™ E25K Server, SMCC, the SMCC logo, SunSoft, the SunSoft logo, Solaris, SunOS, OpenWindows, DeskSet, ONC, and NFS are trademarks or registered trademarks of Sun Microsystems, Inc. All other product names mentioned herein are the trademarks of their respective owners.

All SPARC trademarks, including the SCD Compliant Logo, are trademarks or registered trademarks of SPARC International, Inc. SPARCstation, SPARCserver, SPARCengine, SPARCworks, and SPARCcompiler are licensed exclusively to Sun Microsystems, Inc. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK™ and Sun™ Graphical User Interfaces were developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

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

Oracle Database 10g Release 2, SQL\*DBA, SQL\*Loader, SQL\*Net and SQL\*Plus are registered trademarks of Oracle Corporation.

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

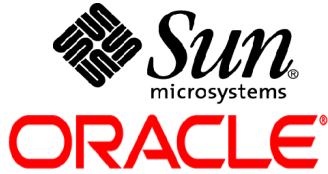
THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THE PUBLICATION. SUN MICROSYSTEMS, INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS PUBLICATION AT ANY TIME.

Sun Microsystems, Inc., believes that the information in this document is accurate as of its publication date. The information in this document is subject to change without notice. Sun Microsystems, Inc., assumes no responsibility for any errors that may appear in this document.

The pricing information in this document is believed to accurately reflect prices in effect on November 29, 2005. However, Sun Microsystems and Oracle Corporation provide no warranty on the pricing information in this document.

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. Sun Microsystems, Inc., does not warrant or represent that a user can or will achieve a similar performance. No warranty on system performance or price/performance is expressed or implied in this document.

---



**Sun Fire™ E25K Server  
with Oracle Database 10g  
Release 2**

TPC-H Rev. 2.3

November 29, 2005

Total System Cost

Composite Query per Hour Metric

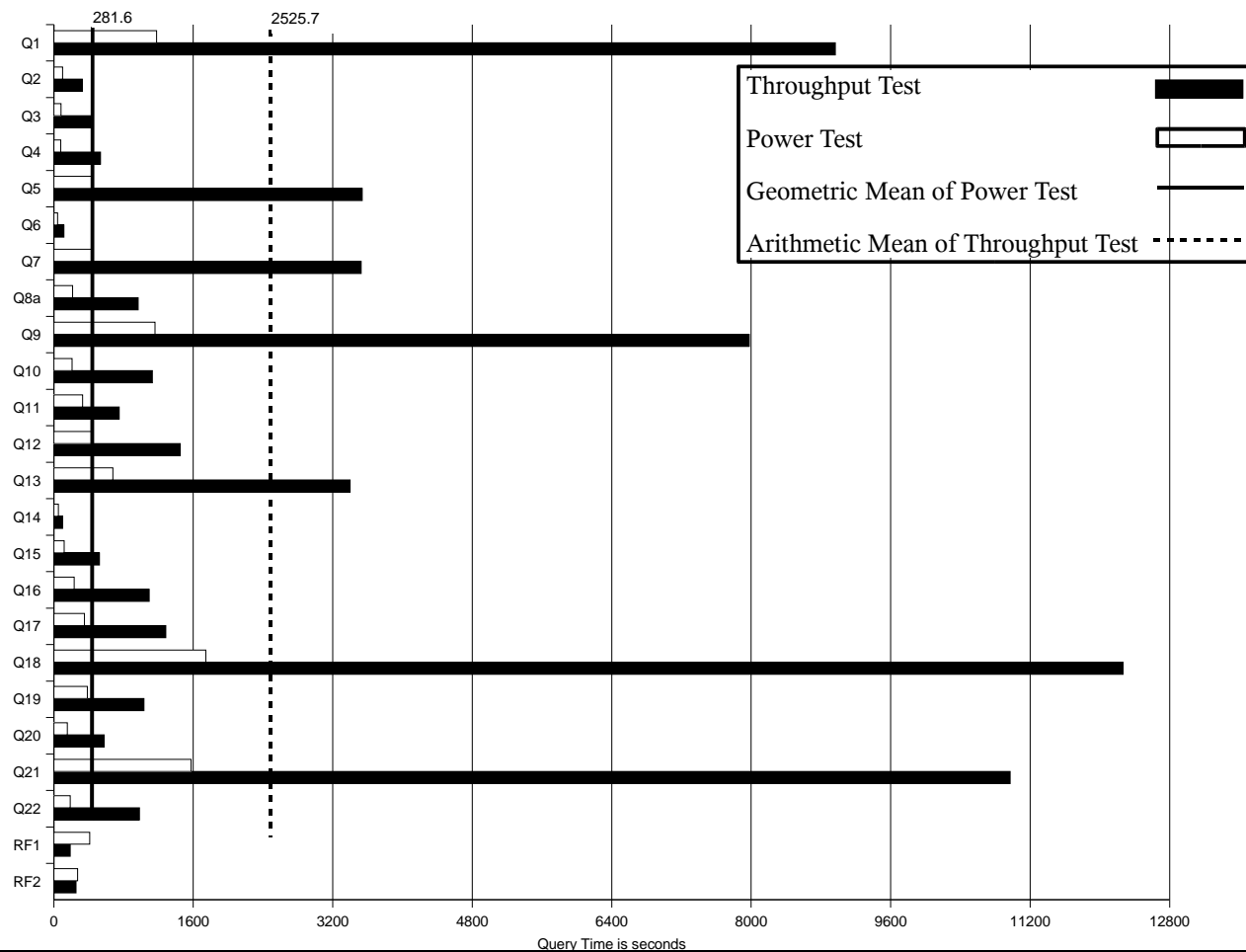
Price/Performance

\$5,815,147

**108,099.7**  
QphH@10000GB

**\$53.80**  
\$/QphH@10000GB

Database Size	Database Manager	Operating System	Other Software	Availability Date
10000GB	Oracle Database 10g Enterprise Edition Release 2 with Partitioning and Oracle Automatic Storage Management	Solaris 10		January 23, 2006



Database Load Time = 18:13

Load Includes Backup: N

Total Data Storage/Database Size=9.7

RAID (Base tables): N

RAID (Base tables and auxiliary data structures): N

RAID (All): Y

System Configuration: Sun Fire™ E25K Server  
 Processors: 72 UltraSPARC™ IV+ 1500 MHz processors on 72 chips, 144 cores, 144 threads  
 Memory: 288GB memory  
 Disks: 108 StorEdge 3510 FC Arrays (12x73.4GB), 1 SE6120 (14x73.4GB), 4 S1 (3x73.4GB)  
 Total Storage: 97,034.8 (in this calculation one GB is defined as 1024\*1024\*1024 bytes)



## Sun Fire™ E25K Server with Oracle Database 10g Release 2

TPC-H Rev. 2.3

November 29, 2005

Description	Part Number	Source	Unit Price	Qty	Ext. Price	3 Yr. Maint.	Unit 3 Yr Maint.
<b>Server Hardware</b>							
Sun Fire E25K base	E25K-BASE		1 315,000	1	315,000	197,244	197,244
UNIBRD 4 US-IV+ @1.5GHz w/16GB	US4BRD-482-1500		1 170,000	18	3,060,000	252,720	14,040
Opt QFE PCI card w/SW	RFBX1034A		1 915	1	915		
PCI Dual GigE + Dual SCSI	RFBX4422A		1 995	1	995		
Cable,SCSI,SCSI-3/SCSI-3, 2.0m	X1139A		1 95	1	95		
hsPCI+ IO Assembly	X4576A		1 15,000	18	270,000		
Opt Pwr Cord For Enterpr.(US)	X3800A		1 0	12	0		
2GB PCI Dual FC Network Adapter	RFBX6768A		1 3,875	54	209,250		
15M Fibre Channel Cable	X9724A		1 190	108	20,520		
<i>Server Hardware Subtotal</i>					3,876,775	449,964	
<b>Storage</b>							
876 GB StorEdge 3510 2 RAID Controllers	TA3510M01A2R876		1 36,495	108	3,941,460	797,040	7,380
StorEdge S1 3x73GB	NS-XDSKS1-373GAC		1 4,695	4	18,780	7,632	1,908
1022 GB StoreEdge 6120	XTA6120R11A1T1022		1 36,995	1	36,995	6,732	6,732
72" StorEdge Expansion Rack	RSG-SG-XARY030A		1 6,500	7	45,500		
Exp Cab 2U Universal Rack Mount Kit	TA-3000-2URK-19U		1 350	96	33,600		
Power Cord for StorEdge	X3858A		1 0	14	0		
<i>Storage Subtotal</i>					4,076,335	811,404	
<b>Server Software</b>							
Solaris 10			3 0	1	0		
Sun One Studio 7	FDEIS-T999-3ST		1 2,376	1	2,376	792	792
Sun StorEdge Comp Mgr	SCMMS-210-R99R		1 0	1	0		
Oracle Database 10g Enterprise Edition, Named User Plus for 3 years			2 10,000	108‡	1,080,000		
Partitioning, Named User Plus for 3 years			2 2,500	108‡	270,000		
Oracle Server Support Package for 3 years			2 6,000	1		6,000	
<i>Server Software Subtotal</i>					1,352,376	6,792	
Sun Volume Discounts (50%) and Support Prepayment (35%)			1		-3,977,743	-441,756	
Oracle Mandatory E-Business Discount (25%)			2		-339,000		
<b>Total</b>					4,988,743	826,404	
Service for all Sun products is from Sun Microsystems, Inc. and is based on SunSpectrum Instant Upgrade Gold 7x24					<b>3 Yr. Cost</b>	5,815,147	
Service for Oracle products is from Oracle Corp.					<b>QphH@10000GB</b>	108,099.70	
					<b>\$/QphH@10000GB</b>	\$53.80	

**Notes (Source):**

1. Sun Microsystems, Inc. (see Appendix G) ‡108 = 0.75 \* 144. Explanation: For the purposes of counting the number of processors which
2. Oracle Corp. (see Appendix G) require licensing, a multicore chip with "n" cores shall be determined by multiplying "n" cores
3. Download from SunSolve by a factor of .75

Audited by: François Raab, InfoSizing, Inc. (www.sizing.com)

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



**Sun Fire™ 15K Server  
with Oracle Database 10g  
Release 2**

TPC-H Rev. 2.3

November 29, 2005

**Numerical Quantities**

**Measurement Results:**

Database Scale Factor	= 10000GB
Total Data Storage / Database Size	= 9.7
Start of database load time	= 11-09-2005 22:43:31
End of database load time	= 11-10-2005 16:56:48
Database Load Time	= 18:13
Query Streams for Throughput Test	= 9
TPC-H Power	= 127,836.1
TPC-H Throughput	= 91,410.4
TPC-H Composite Query-per-Hour Rating (QphH@10000GB)	= 108,099.7
Total System Price Over 3 Years	= \$5,815,147
TPC-H Price/Performance Metric (\$/QphH@10000GB)	= \$53.80

**Measurement Intervals:**

Measurement Interval in Throughput Test (Ts)	= 77,978 seconds
--	------------------

**Duration of Stream Execution:**

Stream ID	Seed	Start Date	Start Time	End Date	End Time	Duration
Stream 00	1110165648	11/10/05	17:09:57	11/10/05	20:11:39	03:01:42
Stream 01	1110165649	11/10/05	20:12:22	11/11/05	13:52:38	17:40:16
Stream 02	1110165650	11/10/05	20:12:22	11/11/05	15:05:36	18:53:14
Stream 03	1110165651	11/10/05	20:12:22	11/11/05	15:59:24	19:47:02
Stream 04	1110165652	11/10/05	20:12:23	11/11/05	15:13:35	19:01:12
Stream 05	1110165653	11/10/05	20:12:23	11/11/05	16:45:37	20:33:14
Stream 06	1110165654	11/10/05	20:12:23	11/11/05	15:35:32	19:23:09
Stream 07	1110165655	11/10/05	20:12:23	11/11/05	05:44:53	09:32:30
Stream 08	1110165656	11/10/05	20:12:23	11/11/05	10:16:43	14:04:20
Stream 09	1110165657	11/10/05	20:12:23	11/11/05	12:10:26	15:58:03
Refresh		11/11/05	16:45:37	11/11/05	17:52:00	01:06:23



**Sun Fire™ 15K Server  
with Oracle Database 10g  
Release 2**

TPC-H Rev. 2.3

November 29, 2005

**TPC-H Timing Intervals (in seconds)**

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
<b>Stream 00</b>	1178.3	101.0	82.5	80.5	435.9	43.3	448.3	215.6	1160.9	209.9	330.0	446.1
<b>Stream 01</b>	2794.3	183.4	612.1	175.6	3771.6	55.3	849.6	412.9	2667.7	464.2	497.8	686.8
<b>Stream 02</b>	2954.8	463.2	332.8	465.6	5460.9	87.1	3954.8	1659.4	13753.1	2404.8	621.9	3597.0
<b>Stream 03</b>	11978.5	159.9	120.8	1094.7	5567.0	101.4	5454.7	1230.2	12467.4	1449.2	913.8	774.3
<b>Stream 04</b>	9008.9	516.1	159.2	191.3	1984.8	75.3	5550.3	391.1	3116.3	2345.8	549.1	728.6
<b>Stream 05</b>	8594.0	251.0	447.7	820.6	748.2	91.9	5127.9	210.5	1274.7	216.8	780.8	439.7
<b>Stream 06</b>	7558.7	334.7	1703.1	741.0	3835.0	344.2	5051.8	1862.1	15509.3	507.9	1101.3	2217.4
<b>Stream 07</b>	12796.5	190.8	156.2	184.0	910.3	73.5	909.2	314.9	2967.4	436.6	567.8	451.8
<b>Stream 08</b>	13865.6	145.1	148.6	164.8	5700.5	71.6	872.5	1391.4	11679.6	461.6	521.4	2317.9
<b>Stream 09</b>	11127.4	701.9	252.1	976.2	3852.7	139.9	3968.0	1229.3	8371.6	1895.2	1184.9	1839.8
<b>Minimum</b>	2794.3	145.1	120.8	164.8	748.2	55.3	849.6	210.5	1274.7	216.8	497.8	439.7
<b>Average</b>	8964.3	327.3	436.9	534.9	3536.8	115.6	3526.5	966.9	7978.6	1131.3	748.8	1450.4
<b>Maximum</b>	13865.6	701.9	1703.1	1094.7	5700.5	344.2	5550.3	1862.1	15509.3	2404.8	1184.9	3597.0

	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	RF1	RF2
<b>Stream 00</b>	677.3	51.9	118.8	233.4	350.9	1745.1	386.2	156.5	1576.2	186.9	413.1	273.0
<b>Stream 01</b>	1664.7	83.7	110.0	496.0	488.6	19412.3	910.0	236.3	26688.7	353.8	212.2	273.3
<b>Stream 02</b>	5004.8	109.8	664.4	1984.9	4162.0	13428.7	1133.5	671.8	3754.4	1324.2	189.6	250.6
<b>Stream 03</b>	669.4	113.4	349.0	277.1	1012.9	21757.9	487.7	664.2	3439.6	1138.4	182.3	251.0
<b>Stream 04</b>	4382.8	104.4	642.6	568.6	652.9	19759.9	975.9	296.9	15194.4	1277.1	189.0	252.3
<b>Stream 05</b>	5175.7	109.7	731.7	2879.2	358.7	17348.8	1148.8	153.9	25839.4	1244.7	185.4	250.7
<b>Stream 06</b>	5612.1	108.8	745.5	1867.0	657.4	3989.8	986.6	593.4	13234.2	1228.0	189.1	250.9
<b>Stream 07</b>	1336.9	99.1	128.4	527.8	801.4	5939.8	810.8	323.6	3992.5	431.0	182.5	252.1
<b>Stream 08</b>	1309.3	97.5	711.7	515.5	814.1	3878.4	1763.9	767.9	3072.3	388.9	183.2	251.3
<b>Stream 09</b>	5438.7	90.9	626.6	729.3	2620.0	4885.7	1080.2	1489.9	3514.0	1468.7	185.3	250.8
<b>Minimum</b>	669.4	83.7	110.0	277.1	358.7	3878.4	487.7	153.9	3072.3	353.8	182.3	250.6
<b>Average</b>	3399.4	101.9	523.3	1093.9	1285.3	12266.8	1033.0	577.5	10969.9	983.9	188.7	253.7
<b>Maximum</b>	5612.1	113.4	745.5	2879.2	4162.0	21757.9	1763.9	1489.9	26688.7	1468.7	212.2	273.3

Benchmark Sponsors: Ray Glasstone  
 Manger, DSS  
 Performance.  
 Oracle Corporation  
 100 Oracle Parkway  
 Redwood Shores, CA  
 94065

Brad Carlile  
 Director, Enterprise Benchmarking  
 Sun Microsystems, Inc.  
 3295 N.W. 211th Terrace  
 Hillsboro OR, 97124

November 21, 2005

I verified the TPC Benchmark™ H performance of the following configuration:

Platform: **Sun Fire E25K Server**  
 Database Manager: **Oracle Database 10g Enterprise Edition Release 2**  
 Operating System: **Solaris 10**

The results were:

CPU (Speed)	Memory	Disks	QphH@10,000GB
<b>Sun Fire E25K Server</b>			
72 UltraSPARC™ IV+ (1500 MHz)	32 MB L3 cache/processor 288 GB Main	1296 x 73.4 GB (SE3510) 14 x 73.4 GB (SE6120) 12 x 73.4 GB (S1)	<b>108,099.7</b>

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

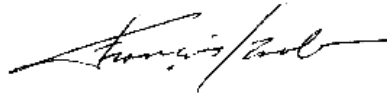
- The database records were defined with the proper layout and size
- The database population was generated using DBGEN
- The database was properly scaled to 10,000GB and populated accordingly
- The compliance of the database auxiliary data structures was verified
- The database load time was correctly measured and reported

- The required ACID properties were verified and met
- The query input variables were generated by QGEN
- The query text was produced using minor modifications and one query variant
- The execution of the queries against the SF1 database produced compliant answers
- A compliant implementation specific layer was used to drive the tests
- The throughput tests involved 9 query streams
- The ratio of longest to shortest query was such that no query timing was adjusted
- The execution times for queries and refresh functions were correctly measured and reported
- The repeatability of the measured results was verified
- The required amount of database log was configured
- The system pricing was verified for major components and maintenance
- The major pages from the FDR were verified for accuracy

Additional Audit Notes:

In the measured configuration 98 of the 108 StorEdge 3510 FC disk arrays were populated with (12) 146GB disk drives each. In the priced configuration, all 3510 disk arrays were populated with (12) 73.4GB disk drives each. Based on the specifications of these disks and on I/O data collected during testing, it is my opinion that this substitution has no significant effect on performance.

Respectfully Yours,



François Raab  
President



# Table of Contents

1. General Items .....	12
1.1 Benchmark Sponsor .....	12
1.2 Parameter Settings .....	12
1.3 Configuration Diagram .....	13
2. Clause 1 Logical Database Design .....	15
2.1 Database Definition Statements .....	15
2.2 Physical Organization .....	15
2.3 Horizontal Partitioning .....	15
2.4 Replication .....	15
3. Clause 2 Queries and Refresh Functions .....	16
3.1 Query Language .....	16
3.2 Verifying Method for Random Number Generation .....	16
3.3 Generating Values for Substitution Parameters .....	16
3.4 Query Text and Output Data from Qualification Database .....	16
3.5 Query Substitution Parameters and Seeds Used .....	16
3.6 Query Isolation Level .....	16
3.7 Source Code of Refresh Functions .....	17
4. Clause 3 Database System Properties .....	18
4.1 ACID Properties .....	18
4.2 Atomicity .....	18
4.2.1 Completed Transaction.....	18
4.2.2 Aborted Transaction.....	18
4.3 Consistency .....	18
4.3.1 Consistency Test.....	19
4.4 Isolation .....	19
4.4.1 Read-Write Conflict with Commit.....	19
4.4.2 Read-Write Conflict with Rollback.....	19
4.4.3 Write-Write Conflict with Commit.....	19
4.4.4 Write-Write Conflict with Rollback.....	20
4.4.5 Concurrent Progress of Read and Write Transactions.....	20
4.4.6 Read-Only Query Conflict with Update Transaction.....	20
4.5 Durability .....	21
4.5.1 Failure of a Durable Medium.....	21
4.5.2 System Crash.....	21
4.5.3 Memory Failure.....	21
5. Clause 4 Scaling and Database Population .....	22
5.1 Ending Cardinality of Tables .....	22
5.2 Distribution of Tables and Logs Across Media .....	22
5.3 Database partition/replication mapping .....	22
5.4 RAID Feature .....	23
5.5 Modifications to the DBGGEN .....	23
5.6 Database Load Time .....	23
5.7 Data Storage Ratio .....	23
5.8 Database Load Mechanism Details and Illustration .....	24
5.9 Qualification Database Configuration .....	24
6. Clause 5 Performance Metrics and Execution Rules .....	25

6.1 System Activity Between Load and Performance Tests .....	25
6.2 Steps in the Power Test .....	25
6.3 Timing Intervals for Each Query and Refresh Functions .....	25
6.4 Number of Streams for the Throughput Test .....	25
6.5 Start and End Date/Times for Each Query Stream .....	25
6.6 Total Elapsed Time of the Measurement Interval .....	26
6.7 Refresh Function Start Date/Time and Finish Date/Time .....	26
6.8 Timing Intervals for Each Query and Each Refresh Function for Each Stream .....	26
6.9 Performance Metrics .....	26
6.10 The Performance Metric and Numerical Quantities from Both Runs .....	26
6.11 System Activity Between Performance Tests .....	26
7. Clause 6 SUT and Driver Implementation .....	27
7.1 Driver .....	27
7.2 Implementation-Specific Layer .....	27
7.3 Profile-Directed Optimization .....	27
8. Clause 7 Pricing .....	28
8.1 Hardware and Software Used .....	28
8.2 Total Three Year Price .....	28
8.3 Availability Date .....	28
9. Auditor's Information and Attestation Letter .....	29
Appendix A. Solaris 10 and Oracle10g Parameters .....	30
Appendix B. Programs and Scripts .....	31
Appendix C. Query Text and Query Output .....	61
Appendix D. Seed and Query Substitution Parameters .....	78
Appendix E. Implementation-Specific Layer/Driver Code .....	80
Appendix F. Misc database scripts .....	93
Appendix G. Pricing information .....	95

## TPC Benchmark H Overview

The TPC Benchmark™ H (TPC-H) is a Decision Support benchmark. It is a suite of business-oriented ad-hoc queries and concurrent modifications. The queries and the data populating the database have been chosen to have broad industry-wide relevance while maintaining a sufficient degree of ease of implementation. This benchmark illustrates Decision Support systems that:

- Examine large volumes of data
- Execute queries with a high degree of complexity
- Give answers to critical business questions

TPC-H evaluates the performance of various Decision Support systems by the execution of sets of queries against a standard database under controlled conditions. The TPC-H queries:

- Give answers to real-world business questions
- Simulate generated ad-hoc queries
- Are far more complex than most OLTP transactions
- Include a rich breadth of operators and selectivity constraints
- Generate intensive activity on the part of the database server component of the system under test
- Are executed against a database complying to specific population and scaling requirements
- Are implemented with constraints derived from staying closely synchronized with an on-line production database

---

## 1. General Items

---

### 1.1 Benchmark Sponsor

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

Sun Microsystems, Inc. and Oracle Corp. are the sponsors of this TPC-H benchmark.

### 1.2 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 parameters*
- *Configuration parameters and options for any other software component incorporated into the pricing structure*
- *Compiler optimization options*

Appendix A contains the Solaris and Oracle parameters used in this benchmark.

---

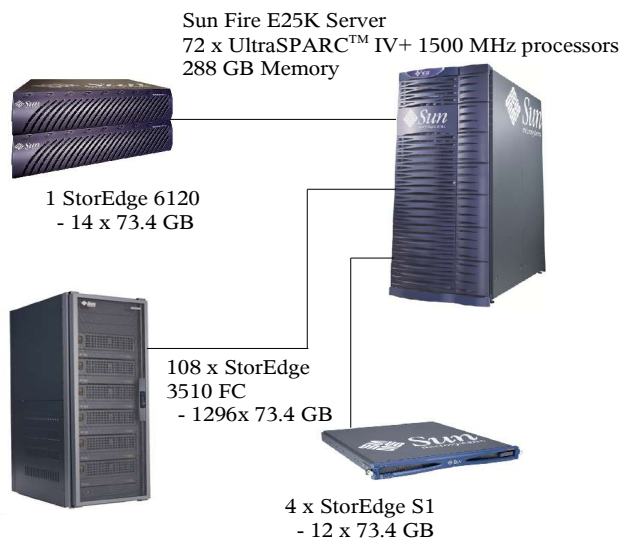
### 1.3 Configuration Diagram

*Provide diagrams of both the measured and priced configurations, accompanied by a description of the differences.*

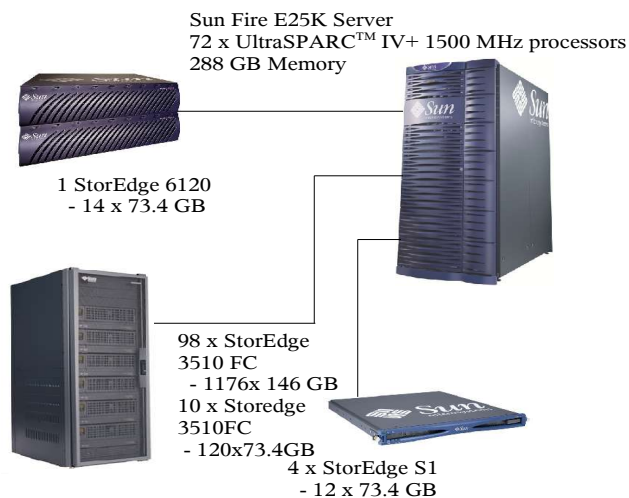
The measured and priced configurations differ regarding the size disk drives used in the StorEdge 3510 FC Arrays.

Sun Fire™ E25K Server, was configured with:

- 72 UltraSPARC IV+ 1500 MHz processors
- 288 GB memory
- 1 Ethernet controller
- 4 S1 disk arrays, each containing 3 x 73.4GB disk drives
- 1 StorEdge 6120 disk array, containing 14 x 73.4GB disk drives
- priced configuration:
  - 108 StorEdge 3510 FC disk arrays, each containing 12 x 73.4GB disk drives @10,000 rpm
- measured configuration:
  - 98 StorEdge 3510 FC disk arrays, each containing 12 x 146GB disk drives @10,000 rpm
  - 10 StorEdge 3510 FC disk arrays, each containing 12 x 73.4GB disk drives @10,000 rpm



Priced configuration



Measured configuration

---

## 2. Clause 1 Logical Database Design

---

### 2.1 Database Definition Statements

*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 programs and scripts that create and analyze the tables and indexes for the TPC-H database.

### 2.2 Physical Organization

*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. Column ordering was changed for some tables. Refer to the table create statements in Appendix B for further details.

### 2.3 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 tables except NATION and REGION. Refer to the table/index create statements in Appendix B for more details.

### 2.4 Replication

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

No replication was used.

---

## 3. Clause 2 Queries and Refresh Functions

---

### 3.1 Query Language

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

SQL was the query language used to implement all queries.

### 3.2 Verifying Method for Random Number Generation

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

TPC supplied versions 2.3.0 of DBGEN and QGEN were used for this TPC-H benchmark.

### 3.3 Generating Values for Substitution Parameters

*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.*

The supplied QGEN version 2.3.0 was used to generate the substitution parameters.

### 3.4 Query Text and Output Data from Qualification 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 C contains the qualification query text and query output.

### 3.5 Query Substitution Parameters and Seeds Used

*The query substitution parameters used for all performance tests must be disclosed in tabular format, along with the seeds used to generate these parameters.*

Appendix D contains the seed and query substitution parameters.

### 3.6 Query Isolation Level

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

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



---

### 3.7 Source Code of Refresh Functions

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

The refresh function is part of the driver code included in Appendix E.

---

## 4. Clause 3 Database System Properties

---

### 4.1 ACID Properties

*The ACID (Atomicity, Consistency, Isolation and Durability) properties of transaction processing systems must be supported by the system under test during the timed portion of this benchmark. Since TPC-H is not a transaction processing benchmark, the ACID properties must be evaluated outside the timed portion of the test.*

Source code for the ACID test is included in Appendix B.

### 4.2 Atomicity

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

#### 4.2.1 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 `ORDERS`, `LINEITEM`, and `HISTORY` tables*

1. The total price from the `ORDERS` 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 committed.
4. The total price from the `ORDERS` 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.

#### 4.2.2 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 `ORDERS`, `LINEITEM`, and `HISTORY` tables.*

1. The total price from the `ORDERS` 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 `ORDERS` table and the extended price from the `LINEITEM` table were retrieved for the same order key. It was verified that the appropriate rows had not been changed.

### 4.3 Consistency

*Consistency is the property of the application that requires any execution of transactions to take the database from one consistent state to another.*

---

### 4.3.1 Consistency Test

*Verify that ORDERS 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 ORDERS and LINEITEM.*

1. The consistency of the ORDERS and LINEITEM tables was verified based on a sample of order keys.
2. 100 ACID Transactions were submitted by each of ten execution streams.
3. The consistency of the ORDERS and LINEITEM tables was re-verified.

## 4.4 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 the proper order.*

### 4.4.1 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 blocked and did not see the uncommitted changes made by the ACID Transaction.
3. The ACID Transaction was resumed and COMMITTED.
4. The ACID Query completed. It returned the data as committed by the ACID Transaction.

### 4.4.2 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 did not see the uncommitted changes made by the ACID Transaction.
3. The ACID Transaction was ROLLED BACK.
4. The ACID Query completed.

### 4.4.3 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. T1 was suspended prior to COMMIT.
2. Another ACID Transaction, T2, was started using the same O\_KEY and L\_KEY and a randomly selected DELTA.
3. T2 waited.

- 
4. T1 was allowed to COMMIT and T2 completed.
  5. It was verified that  $T2.L\_EXTENDEDPRICE = T1.L\_EXTENDEDPRICE + (DELTA1 * (T1.L\_EXTENDEDPRICE / T1.L\_QUANTITY))$

#### 4.4.4 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. T1 was suspended prior to ROLLBACK.
2. Another ACID Transaction, T2, was started using the same O\_KEY and L\_KEY and a randomly selected DELTA.
3. T2 waited.
4. T1 was allowed to ROLLBACK and T2 completed.
5. It was verified that  $T2.L\_EXTENDEDPRICE = T1.L\_EXTENDEDPRICE$ .

#### 4.4.5 Concurrent Progress of Read and Write Transactions

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

1. An ACID Transaction, T1, was started for a randomly selected O\_KEY, L\_KEY, and DELTA. T1 was suspended prior to ROLLBACK.
2. Another Transaction, T2, was started which did the following:  
  
For random values of PS\_PARTKEY and PS\_SUPPKEY, all columns of the PARTSUPP table for which PS\_PARTKEY and PS\_SUPPKEY are equal, are returned.
3. T2 completed.
4. T1 was allowed to COMMIT.
5. It was verified that appropriate rows in ORDERS, LINEITEM and HISTORY tables were changed.

#### 4.4.6 Read-Only Query Conflict with Update Transaction

*Demonstrate that 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.*

1. A Transaction, T1, executing Q1 against the qualification database, was started using a randomly selected DELTA.
2. An ACID Transaction T2, was started for a randomly selected O\_KEY, L\_KEY and DELTA.
3. T2 completed and appropriate rows in the ORDERS, LINEITEM and HISTORY tables had been changed.
4. Transaction T1 completed executing Q1.

---

## 4.5 Durability

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

### 4.5.1 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 disks containing TPC-H tables and, log files were mirrored. During the durability test the disk containing one side of a data file mirror was removed from its cabinet. Similarly the disk containing one side of a log file mirror was removed from its cabinet. The test continued uninterrupted, using the remaining side of the mirror.

### 4.5.2 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.*

The system crash and memory failure tests were combined. Power to the server was turned off by flipping breakers at the main electrical panel during the durability test. When power was restored, the system rebooted and the database was restarted. The durability success file and the HISTORY table were compared successfully.

### 4.5.3 Memory Failure

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

See section 4.5.2.

---

## 5. Clause 4 Scaling and Database Population

---

### 5.1 Ending Cardinality of Tables

*The cardinality (i.e., 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.*

Table	Rows
Orders	15,000,000,000
Lineitem	59,999,994,267
Customer	1,500,000,000
Part	2,000,000,000
Supplier	100,000,000
Partsupp	8,000,000,000
Nation	25
Region	5

### 5.2 Distribution of Tables and Logs Across Media

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

- All tables, indexes, redo logs and temporary tablespace were mirrored and striped across all 1296 disks in the StorEdge 3510 FC disk arrays. The control file resided on a filesystem that was mirrored and striped on a single StorEdge 3510 FC disk array.
- For more details refer to disk configuration section in Appendix B.

### 5.3 Database partition/replication mapping

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

The database was not replicated.

Horizontal partitioning was used for base tables LINEITEM, ORDERS, PARTSUPP, PART, SUPPLIER and CUSTOMER. The details for this partitioning can be understood by examining the syntax of the table and index definition statements in Appendix B.

---

## 5.4 RAID Feature

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 must be disclosed for each device.

Table/Index	RAID type
tables	RAID 1+0
indexes	RAID 1+0
temp tablespace	RAID 1+0
log	RAID 1+0
System tablespace	RAID 1+0

## 5.5 Modifications to the DBGEN

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 2.3.0 was used to generate the database population for this benchmark.

## 5.6 Database Load Time

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

The database load time was 18 hours 13 minutes.

## 5.7 Data Storage Ratio

The data storage ratio must be disclosed. It is computed as the ratio between the total amount of priced disk space, and the chosen test database size as defined in Clause 4.1.3.

The data storage ratio is computed from the following information:

\* Disk manufacturer definition of one GB is  $10^9$  bytes

\*\*In this calculation one GB is defined as  $2^{30}$  bytes

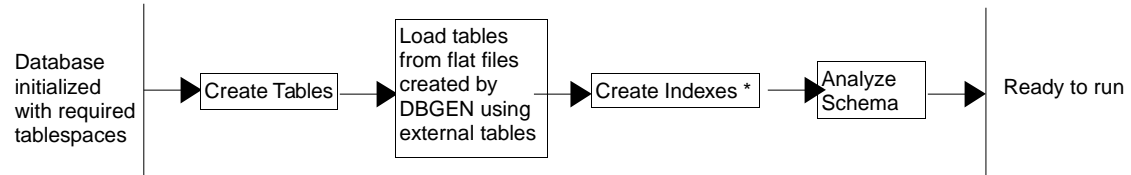
Disk Type	# Of Disks	Space Per Disk*	Sub-Total Disk Space**
3510 FC	1296	73.4GB	95,126.4 GB
6120	14	73.4GB	1027.6 GB
S1	12	73.4GB	880.8 GB
		<b>Total Space</b>	<b>97,034.8</b>
		<b>Data Storage Ratio</b>	<b>9.7</b>

---

## 5.8 Database Load Mechanism Details and Illustration

*The details of the database load must be described, including a block diagram illustrating the overall process.*

The database was loaded using data generation stored on flat files all on the tested and priced configurations. Oracle created external tables using the files that were created by the DBGEN program.



\* Analyze index performed during index creation

## 5.9 Qualification Database Configuration

*Any differences between the configuration of the qualification database and the test database must be disclosed.*

The qualification database used identical scripts to create and load the data with adjustments for the size difference.



---

## 6. Clause 5 Performance Metrics and Execution Rules

---

### 6.1 System Activity Between Load and Performance Tests

*Any system activity on the SUT that takes place between the conclusion of the load test and the beginning of the performance test must be fully disclosed.*

1. Auditor requested queries were run against the database to verify the correctness of the load

All scripts and queries used are included in Appendix F

### 6.2 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. RF1 Refresh Transaction
2. Stream 00 Execution
3. RF2 Refresh Transaction

### 6.3 Timing Intervals for Each Query and Refresh Functions

*The timing intervals for each query and for both refresh functions must be reported for the power test.*

The power test timing intervals are:

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8a	Q9	Q10	Q11	Q12
Stream 00	1178.3	101.0	82.5	80.5	435.9	43.3	448.3	215.6	1160.9	209.9	330.0	446.1
	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	RF1	RF2
Stream 00	677.3	51.9	118.9	233.4	350.9	1745.1	386.2	156.5	1576.2	186.9	413.1	273.0

### 6.4 Number of Streams for the Throughput Test

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

Nine streams were used for the throughput test.

### 6.5 Start and End Date/Times for Each Query Stream

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

The throughput test start time and finish time for each stream are contained in the Numerical Quantity Summary earlier in this document.

---

## 6.6 Total Elapsed Time of the Measurement Interval

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

The total elapsed time of the throughput test is contained in the Numerical Quantity Summary earlier in this document.

## 6.7 Refresh Function Start Date/Time and Finish Date/Time

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

The start and finish times for each refresh function in the refresh stream are contained in the Numerical Quantity Summary earlier in this document.

## 6.8 Timing Intervals for Each Query and Each Refresh Function for Each Stream

*The timing intervals for each query of each stream and each refresh function must be reported for the throughput test.*

The timing intervals for each query and each refresh function for the throughput test are contained in the Numerical Quantity Summary earlier in this document.

## 6.9 Performance Metrics

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

The performance metrics, and the numbers on which they are based, are contained in the Numerical Quantity Summary earlier in this document.

## 6.10 The Performance Metric and Numerical Quantities from Both Runs

*The performance metric and numerical quantities from both runs must be disclosed.*

Performance results from the first two executions of the TPC-H benchmark indicated the following percent difference for the metric points:

Run ID	QppH@10000GB	QthH@10000GB	QphH@10000GB
Run 1	127,836.1	91,410.4	108,099.7
Run 2	136,796.2	90,466.0	111,244.8

## 6.11 System Activity Between Performance Tests

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

There was no activity on the SUT between run1 and run2.

---

## 7. Clause 6 SUT and Driver Implementation

---

### 7.1 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.*

The Power Test and Throughput Test are performed by a shell script called `runTPCpt`. QGEN is first called with a stream id of 0 to generate the QET for the Power Test. UF1 is then started by executing the `runuf1.sh` script. Query submission follows, with the `qexecpl.c` ISL program. The execution of the UF2 script `runuf2.sh` rounds out the Power Test execution. Both wall-clock and high-resolution times are collected for all measurement intervals.

Following the Power Test, QGEN is again called with the subsequent 9 stream ids to generate new QET for each Throughput Test. `qexecpl.c` is called simultaneously for all 9 streams to execute the queries as above. Then the `update_stream.sh` script is called to run all 9 update pairs to finish the throughput run.

### 7.2 Implementation-Specific Layer

*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.*

Query execution text generated by QGEN is picked up by the ISL program which submits the query to the SUT.

The ISL program (`qexecpl.c`) utilizes the Oracle Call Interface (OCI) to communicate with the Oracle database on the SUT. EQTs directly generated by QGEN are read and submitted to the SUT via the ISL program (`qexecpl.c`) as dynamic SQL statements. The ISL program then fetches the query execution output and reports it to the user. Timings are taken at intervals specified in Section 5.3.7 of the TPC-H benchmark specification.

The Update Functions use external tables to load data from flat files. Oracle9i's parallel insert and delete functionality was used to perform the Update Functions, selecting data from the temporary tables.

### 7.3 Profile-Directed Optimization

*If profile-directed optimization as described in Clause 5.2.9 is used, such use must be disclosed.*

Profile-directed optimization was not used.

---

## 8. Clause 7 Pricing

---

### 8.1 Hardware and Software Used

*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.*

Refer to the Executive Summary.

### 8.2 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.*

The total 3-year price of the configuration is \$5,815,147. For details of pricing, see the second page of the Executive Summary.

The following generally available discounts to any buyer with like conditions were applied to the priced configuration:

- a 35% Sun support volume and yearly pre-payment discount
- a 50% discount from list for Sun supplied system components

### 8.3 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 reported availability date for the priced system must be the date at which all components are committed to be available.*

All Hardware and Software components, except Solaris10 Update1, are available immediately. Solaris10 Update1 will be available January 23, 2006.

---

---

## 9. Auditor's Information and Attestation Letter

---

*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.*

The auditor's attestation letter is included at the front of this report.

---

## Appendix A. Solaris 10 and Oracle10g Parameters

---

This Appendix contains Solaris kernel parameters and environment variables and Oracle initialization parameters.

### Oracle 10g Parameters

(altered from default)

#### inittpch.ora

```
=====  
aq_tm_processes           = 0  
audit_trail               = FALSE  
compatible                = 10.2.0.1  
control_files             = /ff1/control.dbf  
db_block_checksum        = FALSE  
db_block_size             = 32768  
db_create_file_dest       = +dg_tpch  
db_file_multiblock_read_count = 32  
db_files                  = 1023  
db_name                   = tpch  
db_writer_processes       = 20  
dml_locks                 = 80000  
global_names              = FALSE  
java_pool_size            = 0  
log_checkpoints_to_alert  = TRUE  
max_dump_file_size        = unlimited  
nls_date_format           = YYYY-MM-DD  
open_cursors              = 1024  
optimizer_features_enable = 10.2.0.1.1  
optimizer_mode            = CHOOSE  
parallel_execution_message_size = 32768  
parallel_max_servers      = 2000  
parallel_min_servers      = 1700  
parallel_threads_per_cpu  = 1  
pga_aggregate_target      = 100g  
processes                 = 4000  
recovery_parallelism      = 64  
replication_dependency_tracking = FALSE  
sessions                  = 2048  
sga_target                = 90g  
undo_management           = AUTO  
undo_tablespace           = ts_undo  
=====
```

### Solaris Parameters

(altered from default)

```
=====  
set shmsys:shminfo_shmmax=0xffffffffffffffff  
set shmsys:shminfo_shmseg=200  
set msgsys:msginfo_msgmax=1048576  
set msgsys:msginfo_msgmnb=4194304  
set msgsys:msginfo_msgmni=4400  
set msgsys:msginfo_msgtql=32768  
set msgsys:msginfo_msgseg=32767  
set msgsys:msginfo_msgssz=128  
set msgsys:msginfo_msgmap=3002  
set semsys:seminfo_semmmap=100  
set semsys:seminfo_semmni=8000  
set semsys:seminfo_semmns=8000  
set semsys:seminfo_semmnu=8000  
set semsys:seminfo_semmsl=512  
set semsys:seminfo_semume=100  
set maxpgio=131072  
set maxphys=4194304  
set autoup=900  
set bufhwm=8000  
set segspt_minfree=16000  
set kernel_cage_enable=0  
set segmap_percent=1  
set lgrp_mem_pset_aware=0x1  
=====
```

---

## Appendix B. Programs and Scripts

---

### Database Load Scripts

#### dbcre.sh

```
#!/bin/ksh
#####
# create tpch database
#####
echo Start Database Creation at `date`

sqlplus / as sysdba<<EOF
set echo on
set timing on

shutdown abort;

startup pfile=?/dbs/inittpch.ora nomount;
create database
  controlfile reuse
  set default bigfile tablespace
  logfile '+dg_tpch' size 50000m reuse,
         '+dg_tpch' size 50000m reuse
  datafile '+dg_tpch'
           size 5000m reuse
  sysaux datafile '+dg_tpch'
                size 5000m reuse
  smallfile undo tablespace ts_undo
  datafile '+dg_tpch'
           size 5000m reuse
  default temporary tablespace ts_temp
  tempfile '+dg_tpch'
           size 9200000m reuse
           extent management local uniform size 10m
  maxdatafiles 2000
  maxinstances 1
;

set termout off
set echo off
spool /tmp/cat
@?/rdbms/admin/catalog.sql;
@?/rdbms/admin/catparrr.sql;
@?/rdbms/admin/catproc.sql;
@?/rdbms/admin/utlxplan.sql;

connect system/manager
@?/sqlplus/admin/pupbld.sql;
exit;
EOF
```

```
echo End Database Creation at `date`
```

#### sctso.sh

```
#!/bin/ksh
#####
#####
# Schema Creation Phase - datafiles only (no tables or
users)
# creating data tablespaces, datafiles
# creating tpch's ts_default tablespace
#####
#####
echo START Tablespace Creation at `date`

# create ts_default tablespace
sqlplus / as sysdba<<! &
set echo on;
set timing on;

drop tablespace ts_default;
create tablespace ts_default
datafile '+dg_tpch' size 20000m reuse
extent management local autoallocate ;
exit;
!
```

```
# create ts_s tablespace
sqlplus / as sysdba<<! &
set echo on;
set timing on;
drop tablespace ts_s;
create tablespace ts_s
datafile '+dg_tpch' size 20000m reuse
extent management local autoallocate nologging ;
exit;
!
```

```
# create ts_c tablespace
sqlplus / as sysdba<<! &
set echo on;
set timing on;
drop tablespace ts_c;
create tablespace ts_c
datafile '+dg_tpch' size 280000m reuse
extent management local autoallocate nologging ;
exit;
!

# create ts_o tablespaces
let i=1
while ((i<=21))
do
sqlplus / as sysdba<<! &
set echo on;
set timing on;
drop tablespace ts_o${i};
create tablespace ts_o${i}
datafile '+dg_tpch' size 120000m reuse
extent management local uniform size 10m nologging ;
exit;
!
let i+=1
done
# create ts_ps tablespace
sqlplus / as sysdba<<! &
set echo on;
set timing on;
drop tablespace ts_ps;
create tablespace ts_ps
datafile '+dg_tpch' size 1600000m reuse
extent management local autoallocate nologging ;
exit;
!
```

```
# create ts_p tablespace
sqlplus / as sysdba<<! &
set echo on;
set timing on;
drop tablespace ts_p;
create tablespace ts_p
datafile '+dg_tpch' size 200000m reuse
extent management local autoallocate nologging ;
exit;
!
```

```
wait
# create ts_l tablespaces
let i=1
while ((i<=21))
do
sqlplus / as sysdba<<! &
set echo on;
set timing on;
drop tablespace ts_l${i};
create tablespace ts_l${i}
datafile '+dg_tpch' size 450000m reuse
extent management local uniform size 20m nologging ;
exit;
!
let i+=1
done
wait
# create tpch's ts_i_lorderkey tablespace
sqlplus / as sysdba<<! &
set echo on;
set timing on;
drop tablespace ts_i_lorderkey;
create tablespace ts_i_lorderkey
datafile '+dg_tpch' size 1600000m reuse
extent management local autoallocate nologging ;
exit;
!
```

```
# create tpch's ts_i_oorderkey tablespace
```

```

sqlplus / as sysdba<<! &
set echo on;
set timing on;
drop tablespace ts_i_oorderkey;
create tablespace ts_i_oorderkey
datafile '+dg_tpch' size 400000m reuse
extent management local autoallocate nologging ;
exit;
!

# creating tpch's ts_i_ccustkey tablespace
sqlplus / as sysdba<<! &
set echo on;
set timing on;
drop tablespace ts_i_ccustkey;
create tablespace ts_i_ccustkey
datafile '+dg_tpch' size 36000m reuse
extent management local autoallocate nologging ;
exit;
!

# adding tpch's ts_undo datafiles
let i=1
while ((i<=8))
do
sqlplus / as sysdba<<! &
set echo on;
set timing on;
alter tablespace ts_undo
add datafile '+dg_tpch' size 128000m reuse;
exit;
!
let i+=1
done

wait

echo END Tablespace Creation at `date`

```

## datapop.sh

```

#####
#!/bin/ksh
#####
# Schema Creation Phase - User and Tables and Database
Population Phase
#####
echo Start Data Loading at `date`

sqlplus / as sysdba <<!
set timing on
set echo on;
set termout on;

drop user tpch cascade;
grant DBA
to tpch identified by tpch;

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

connect tpch/tpch;
drop directory data_dir1;
drop directory data_dir2;
drop directory data_dir3;
drop directory data_dir4;
drop directory data_dir5;
drop directory data_dir6;
drop directory data_dir7;
drop directory data_dir8;
drop directory data_dir9;
drop directory data_dir10;
drop directory data_dir11;
drop directory data_dir12;
drop directory data_dir13;
drop directory data_dir14;
drop directory data_dir15;
drop directory data_dir16;
drop directory data_dir17;
drop directory data_dir18;
drop directory data_dir19;

```

```

drop directory data_dir20;
drop directory data_dir21;
drop directory data_dir22;
drop directory data_dir23;
drop directory data_dir24;
drop directory data_dir25;
drop directory data_dir26;
drop directory data_dir27;
drop directory data_dir28;
drop directory data_dir29;
drop directory data_dir30;
drop directory data_dir31;
drop directory data_dir32;
drop directory data_dir33;
drop directory data_dir34;
drop directory data_dir35;
drop directory data_dir36;
drop directory data_dir37;
drop directory data_dir38;
drop directory data_dir39;
drop directory data_dir40;
drop directory data_dir41;
drop directory data_dir42;
drop directory data_dir43;
drop directory data_dir44;
drop directory data_dir45;
drop directory data_dir46;
drop directory data_dir47;
drop directory data_dir48;
drop directory data_dir49;
drop directory data_dir50;
drop directory data_dir51;
drop directory data_dir52;
drop directory data_dir53;
drop directory data_dir54;
drop directory data_dir55;
drop directory data_dir56;
drop directory data_dir57;
drop directory data_dir58;
drop directory data_dir59;
drop directory data_dir60;
drop directory data_dir61;
drop directory data_dir62;
drop directory data_dir63;
drop directory data_dir64;
drop directory data_dir65;
drop directory data_dir66;
drop directory data_dir67;
drop directory data_dir68;
drop directory data_dir69;
drop directory data_dir70;
drop directory data_dir71;
drop directory data_dir72;
drop directory data_dir73;
drop directory data_dir74;
drop directory data_dir75;
drop directory data_dir76;
drop directory data_dir77;
drop directory data_dir78;
drop directory data_dir79;
drop directory data_dir80;
drop directory data_dir81;
drop directory data_dir82;
drop directory data_dir83;
drop directory data_dir84;
drop directory data_dir85;
drop directory data_dir86;
drop directory data_dir87;
drop directory data_dir88;
drop directory data_dir89;
drop directory data_dir90;
drop directory data_dir91;
drop directory data_dir92;
drop directory data_dir93;
drop directory data_dir94;
drop directory data_dir95;
drop directory data_dir96;
create directory data_dir1 as '/ff1';
create directory data_dir2 as '/ff2';
create directory data_dir3 as '/ff3';
create directory data_dir4 as '/ff4';
create directory data_dir5 as '/ff5';
create directory data_dir6 as '/ff6';
create directory data_dir7 as '/ff7';
create directory data_dir8 as '/ff8';
create directory data_dir9 as '/ff9';

```



```

create directory data_dir10 as '/ff10';
create directory data_dir11 as '/ff11';
create directory data_dir12 as '/ff12';
create directory data_dir13 as '/ff13';
create directory data_dir14 as '/ff14';
create directory data_dir15 as '/ff15';
create directory data_dir16 as '/ff16';
create directory data_dir17 as '/ff17';
create directory data_dir18 as '/ff18';
create directory data_dir19 as '/ff19';
create directory data_dir20 as '/ff20';
create directory data_dir21 as '/ff21';
create directory data_dir22 as '/ff22';
create directory data_dir23 as '/ff23';
create directory data_dir24 as '/ff24';
create directory data_dir25 as '/ff25';
create directory data_dir26 as '/ff26';
create directory data_dir27 as '/ff27';
create directory data_dir28 as '/ff28';
create directory data_dir29 as '/ff29';
create directory data_dir30 as '/ff30';
create directory data_dir31 as '/ff31';
create directory data_dir32 as '/ff32';
create directory data_dir33 as '/ff33';
create directory data_dir34 as '/ff34';
create directory data_dir35 as '/ff35';
create directory data_dir36 as '/ff36';
create directory data_dir37 as '/ff37';
create directory data_dir38 as '/ff38';
create directory data_dir39 as '/ff39';
create directory data_dir40 as '/ff40';
create directory data_dir41 as '/ff41';
create directory data_dir42 as '/ff42';
create directory data_dir43 as '/ff43';
create directory data_dir44 as '/ff44';
create directory data_dir45 as '/ff45';
create directory data_dir46 as '/ff46';
create directory data_dir47 as '/ff47';
create directory data_dir48 as '/ff48';
create directory data_dir49 as '/ff49';
create directory data_dir50 as '/ff50';
create directory data_dir51 as '/ff51';
create directory data_dir52 as '/ff52';
create directory data_dir53 as '/ff53';
create directory data_dir54 as '/ff54';
create directory data_dir55 as '/ff55';
create directory data_dir56 as '/ff56';
create directory data_dir57 as '/ff57';
create directory data_dir58 as '/ff58';
create directory data_dir59 as '/ff59';
create directory data_dir60 as '/ff60';
create directory data_dir61 as '/ff61';
create directory data_dir62 as '/ff62';
create directory data_dir63 as '/ff63';
create directory data_dir64 as '/ff64';
create directory data_dir65 as '/ff65';
create directory data_dir66 as '/ff66';
create directory data_dir67 as '/ff67';
create directory data_dir68 as '/ff68';
create directory data_dir69 as '/ff69';
create directory data_dir70 as '/ff70';
create directory data_dir71 as '/ff71';
create directory data_dir72 as '/ff72';
create directory data_dir73 as '/ff73';
create directory data_dir74 as '/ff74';
create directory data_dir75 as '/ff75';
create directory data_dir76 as '/ff76';
create directory data_dir77 as '/ff77';
create directory data_dir78 as '/ff78';
create directory data_dir79 as '/ff79';
create directory data_dir80 as '/ff80';
create directory data_dir81 as '/ff81';
create directory data_dir82 as '/ff82';
create directory data_dir83 as '/ff83';
create directory data_dir84 as '/ff84';
create directory data_dir85 as '/ff85';
create directory data_dir86 as '/ff86';
create directory data_dir87 as '/ff87';
create directory data_dir88 as '/ff88';
create directory data_dir89 as '/ff89';
create directory data_dir90 as '/ff90';
create directory data_dir91 as '/ff91';
create directory data_dir92 as '/ff92';
create directory data_dir93 as '/ff93';
create directory data_dir94 as '/ff94';
create directory data_dir95 as '/ff95';

```

```

create directory data_dir96 as '/ff96';

rem 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 ,
  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_dir1
access parameters
(
  records delimited by newline
  badfile 'l_et.bad'
  logfile 'l_et.log'
  fields terminated by '|'
  missing field values are null
)
location (
data_dir1:'lineitem.tbl.1',
data_dir2:'lineitem.tbl.2',
data_dir3:'lineitem.tbl.3',
data_dir4:'lineitem.tbl.4',
data_dir5:'lineitem.tbl.5',
data_dir6:'lineitem.tbl.6',
data_dir7:'lineitem.tbl.7',
data_dir8:'lineitem.tbl.8',
data_dir9:'lineitem.tbl.9',
data_dir10:'lineitem.tbl.10',
data_dir11:'lineitem.tbl.11',
data_dir12:'lineitem.tbl.12',
data_dir13:'lineitem.tbl.13',
data_dir14:'lineitem.tbl.14',
data_dir15:'lineitem.tbl.15',
data_dir16:'lineitem.tbl.16',
data_dir17:'lineitem.tbl.17',
data_dir18:'lineitem.tbl.18',
data_dir19:'lineitem.tbl.19',
data_dir20:'lineitem.tbl.20',
data_dir21:'lineitem.tbl.21',
data_dir22:'lineitem.tbl.22',
data_dir23:'lineitem.tbl.23',
data_dir24:'lineitem.tbl.24',
data_dir25:'lineitem.tbl.25',
data_dir26:'lineitem.tbl.26',
data_dir27:'lineitem.tbl.27',
data_dir28:'lineitem.tbl.28',
data_dir29:'lineitem.tbl.29',
data_dir30:'lineitem.tbl.30',
data_dir31:'lineitem.tbl.31',
data_dir32:'lineitem.tbl.32',
data_dir33:'lineitem.tbl.33',
data_dir34:'lineitem.tbl.34',
data_dir35:'lineitem.tbl.35',
data_dir36:'lineitem.tbl.36',
data_dir37:'lineitem.tbl.37',
data_dir38:'lineitem.tbl.38',
data_dir39:'lineitem.tbl.39',
data_dir40:'lineitem.tbl.40',
data_dir41:'lineitem.tbl.41',
data_dir42:'lineitem.tbl.42',
data_dir43:'lineitem.tbl.43',
data_dir44:'lineitem.tbl.44',
data_dir45:'lineitem.tbl.45',
data_dir46:'lineitem.tbl.46',
data_dir47:'lineitem.tbl.47',
data_dir48:'lineitem.tbl.48',
data_dir49:'lineitem.tbl.49',
data_dir50:'lineitem.tbl.50',
data_dir51:'lineitem.tbl.51',
data_dir52:'lineitem.tbl.52',
data_dir53:'lineitem.tbl.53',

```

```

data_dir54:'lineitem.tbl.54',
data_dir55:'lineitem.tbl.55',
data_dir56:'lineitem.tbl.56',
data_dir57:'lineitem.tbl.57',
data_dir58:'lineitem.tbl.58',
data_dir59:'lineitem.tbl.59',
data_dir60:'lineitem.tbl.60',
data_dir61:'lineitem.tbl.61',
data_dir62:'lineitem.tbl.62',
data_dir63:'lineitem.tbl.63',
data_dir64:'lineitem.tbl.64',
data_dir65:'lineitem.tbl.65',
data_dir66:'lineitem.tbl.66',
data_dir67:'lineitem.tbl.67',
data_dir68:'lineitem.tbl.68',
data_dir69:'lineitem.tbl.69',
data_dir70:'lineitem.tbl.70',
data_dir71:'lineitem.tbl.71',
data_dir72:'lineitem.tbl.72',
data_dir73:'lineitem.tbl.73',
data_dir74:'lineitem.tbl.74',
data_dir75:'lineitem.tbl.75',
data_dir76:'lineitem.tbl.76',
data_dir77:'lineitem.tbl.77',
data_dir78:'lineitem.tbl.78',
data_dir79:'lineitem.tbl.79',
data_dir80:'lineitem.tbl.80',
data_dir81:'lineitem.tbl.81',
data_dir82:'lineitem.tbl.82',
data_dir83:'lineitem.tbl.83',
data_dir84:'lineitem.tbl.84',
data_dir85:'lineitem.tbl.85',
data_dir86:'lineitem.tbl.86',
data_dir87:'lineitem.tbl.87',
data_dir88:'lineitem.tbl.88',
data_dir89:'lineitem.tbl.89',
data_dir90:'lineitem.tbl.90',
data_dir91:'lineitem.tbl.91',
data_dir92:'lineitem.tbl.92',
data_dir93:'lineitem.tbl.93',
data_dir94:'lineitem.tbl.94',
data_dir95:'lineitem.tbl.95',
data_dir96:'lineitem.tbl.96'
))
reject limit unlimited
parallel 144;

rem 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 data_dir1
access parameters
(
      records delimited by newline
      badfile 'o_et.bad'
      logfile 'o_et.log'
      fields terminated by '|'
      missing field values are null
)
location (
data_dir1:'orders.tbl.1',
data_dir2:'orders.tbl.2',
data_dir3:'orders.tbl.3',
data_dir4:'orders.tbl.4',
data_dir5:'orders.tbl.5',
data_dir6:'orders.tbl.6',
data_dir7:'orders.tbl.7',
data_dir8:'orders.tbl.8',
data_dir9:'orders.tbl.9',
data_dir10:'orders.tbl.10',
data_dir11:'orders.tbl.11',
data_dir12:'orders.tbl.12',
data_dir13:'orders.tbl.13',
data_dir14:'orders.tbl.14',
data_dir15:'orders.tbl.15',
data_dir16:'orders.tbl.16',
data_dir17:'orders.tbl.17',
data_dir18:'orders.tbl.18',
data_dir19:'orders.tbl.19',
data_dir20:'orders.tbl.20',
data_dir21:'orders.tbl.21',
data_dir22:'orders.tbl.22',
data_dir23:'orders.tbl.23',
data_dir24:'orders.tbl.24',
data_dir25:'orders.tbl.25',
data_dir26:'orders.tbl.26',
data_dir27:'orders.tbl.27',
data_dir28:'orders.tbl.28',
data_dir29:'orders.tbl.29',
data_dir30:'orders.tbl.30',
data_dir31:'orders.tbl.31',
data_dir32:'orders.tbl.32',
data_dir33:'orders.tbl.33',
data_dir34:'orders.tbl.34',
data_dir35:'orders.tbl.35',
data_dir36:'orders.tbl.36',
data_dir37:'orders.tbl.37',
data_dir38:'orders.tbl.38',
data_dir39:'orders.tbl.39',
data_dir40:'orders.tbl.40',
data_dir41:'orders.tbl.41',
data_dir42:'orders.tbl.42',
data_dir43:'orders.tbl.43',
data_dir44:'orders.tbl.44',
data_dir45:'orders.tbl.45',
data_dir46:'orders.tbl.46',
data_dir47:'orders.tbl.47',
data_dir48:'orders.tbl.48',
data_dir49:'orders.tbl.49',
data_dir50:'orders.tbl.50',
data_dir51:'orders.tbl.51',
data_dir52:'orders.tbl.52',
data_dir53:'orders.tbl.53',
data_dir54:'orders.tbl.54',
data_dir55:'orders.tbl.55',
data_dir56:'orders.tbl.56',
data_dir57:'orders.tbl.57',
data_dir58:'orders.tbl.58',
data_dir59:'orders.tbl.59',
data_dir60:'orders.tbl.60',
data_dir61:'orders.tbl.61',
data_dir62:'orders.tbl.62',
data_dir63:'orders.tbl.63',
data_dir64:'orders.tbl.64',
data_dir65:'orders.tbl.65',
data_dir66:'orders.tbl.66',
data_dir67:'orders.tbl.67',
data_dir68:'orders.tbl.68',
data_dir69:'orders.tbl.69',
data_dir70:'orders.tbl.70',
data_dir71:'orders.tbl.71',
data_dir72:'orders.tbl.72',
data_dir73:'orders.tbl.73',
data_dir74:'orders.tbl.74',
data_dir75:'orders.tbl.75',
data_dir76:'orders.tbl.76',
data_dir77:'orders.tbl.77',
data_dir78:'orders.tbl.78',
data_dir79:'orders.tbl.79',
data_dir80:'orders.tbl.80',
data_dir81:'orders.tbl.81',
data_dir82:'orders.tbl.82',
data_dir83:'orders.tbl.83',
data_dir84:'orders.tbl.84',
data_dir85:'orders.tbl.85',
data_dir86:'orders.tbl.86',
data_dir87:'orders.tbl.87',
data_dir88:'orders.tbl.88',
data_dir89:'orders.tbl.89',
data_dir90:'orders.tbl.90',
data_dir91:'orders.tbl.91',
data_dir92:'orders.tbl.92',
data_dir93:'orders.tbl.93',
data_dir94:'orders.tbl.94',
data_dir95:'orders.tbl.95',
data_dir96:'orders.tbl.96'
))
reject limit unlimited
parallel 144;

rem 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 data_dir1
access parameters
(
  records delimited by newline
  badfile 'ps_et.bad'
  logfile 'ps_et.log'
  fields terminated by '|'
  missing field values are null
)
location (
data_dir1:'partsupp.tbl.1',
data_dir2:'partsupp.tbl.2',
data_dir3:'partsupp.tbl.3',
data_dir4:'partsupp.tbl.4',
data_dir5:'partsupp.tbl.5',
data_dir6:'partsupp.tbl.6',
data_dir7:'partsupp.tbl.7',
data_dir8:'partsupp.tbl.8',
data_dir9:'partsupp.tbl.9',
data_dir10:'partsupp.tbl.10',
data_dir11:'partsupp.tbl.11',
data_dir12:'partsupp.tbl.12',
data_dir13:'partsupp.tbl.13',
data_dir14:'partsupp.tbl.14',
data_dir15:'partsupp.tbl.15',
data_dir16:'partsupp.tbl.16',
data_dir17:'partsupp.tbl.17',
data_dir18:'partsupp.tbl.18',
data_dir19:'partsupp.tbl.19',
data_dir20:'partsupp.tbl.20',
data_dir21:'partsupp.tbl.21',
data_dir22:'partsupp.tbl.22',
data_dir23:'partsupp.tbl.23',
data_dir24:'partsupp.tbl.24',
data_dir25:'partsupp.tbl.25',
data_dir26:'partsupp.tbl.26',
data_dir27:'partsupp.tbl.27',
data_dir28:'partsupp.tbl.28',
data_dir29:'partsupp.tbl.29',
data_dir30:'partsupp.tbl.30',
data_dir31:'partsupp.tbl.31',
data_dir32:'partsupp.tbl.32',
data_dir33:'partsupp.tbl.33',
data_dir34:'partsupp.tbl.34',
data_dir35:'partsupp.tbl.35',
data_dir36:'partsupp.tbl.36',
data_dir37:'partsupp.tbl.37',
data_dir38:'partsupp.tbl.38',
data_dir39:'partsupp.tbl.39',
data_dir40:'partsupp.tbl.40',
data_dir41:'partsupp.tbl.41',
data_dir42:'partsupp.tbl.42',
data_dir43:'partsupp.tbl.43',
data_dir44:'partsupp.tbl.44',
data_dir45:'partsupp.tbl.45',
data_dir46:'partsupp.tbl.46',
data_dir47:'partsupp.tbl.47',
data_dir48:'partsupp.tbl.48',
data_dir49:'partsupp.tbl.49',
data_dir50:'partsupp.tbl.50',
data_dir51:'partsupp.tbl.51',
data_dir52:'partsupp.tbl.52',
data_dir53:'partsupp.tbl.53',
data_dir54:'partsupp.tbl.54',
data_dir55:'partsupp.tbl.55',
data_dir56:'partsupp.tbl.56',
data_dir57:'partsupp.tbl.57',
data_dir58:'partsupp.tbl.58',
data_dir59:'partsupp.tbl.59',
data_dir60:'partsupp.tbl.60',
data_dir61:'partsupp.tbl.61',
data_dir62:'partsupp.tbl.62',
data_dir63:'partsupp.tbl.63',
data_dir64:'partsupp.tbl.64',
data_dir65:'partsupp.tbl.65',
data_dir66:'partsupp.tbl.66',
data_dir67:'partsupp.tbl.67',

```

```

data_dir68:'partsupp.tbl.68',
data_dir69:'partsupp.tbl.69',
data_dir70:'partsupp.tbl.70',
data_dir71:'partsupp.tbl.71',
data_dir72:'partsupp.tbl.72',
data_dir73:'partsupp.tbl.73',
data_dir74:'partsupp.tbl.74',
data_dir75:'partsupp.tbl.75',
data_dir76:'partsupp.tbl.76',
data_dir77:'partsupp.tbl.77',
data_dir78:'partsupp.tbl.78',
data_dir79:'partsupp.tbl.79',
data_dir80:'partsupp.tbl.80',
data_dir81:'partsupp.tbl.81',
data_dir82:'partsupp.tbl.82',
data_dir83:'partsupp.tbl.83',
data_dir84:'partsupp.tbl.84',
data_dir85:'partsupp.tbl.85',
data_dir86:'partsupp.tbl.86',
data_dir87:'partsupp.tbl.87',
data_dir88:'partsupp.tbl.88',
data_dir89:'partsupp.tbl.89',
data_dir90:'partsupp.tbl.90',
data_dir91:'partsupp.tbl.91',
data_dir92:'partsupp.tbl.92',
data_dir93:'partsupp.tbl.93',
data_dir94:'partsupp.tbl.94',
data_dir95:'partsupp.tbl.95',
data_dir96:'partsupp.tbl.96'
)

```

```

reject limit unlimited
parallel 144;

```

```

rem 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 data_dir1
access parameters
(

```

```

  records delimited by newline
  badfile 'p_et.bad'
  logfile 'p_et.log'
  fields terminated by '|'
  missing field values are null
)

```

```

location (
data_dir1:'part.tbl.1',
data_dir2:'part.tbl.2',
data_dir3:'part.tbl.3',
data_dir4:'part.tbl.4',
data_dir5:'part.tbl.5',
data_dir6:'part.tbl.6',
data_dir7:'part.tbl.7',
data_dir8:'part.tbl.8',
data_dir9:'part.tbl.9',
data_dir10:'part.tbl.10',
data_dir11:'part.tbl.11',
data_dir12:'part.tbl.12',
data_dir13:'part.tbl.13',
data_dir14:'part.tbl.14',
data_dir15:'part.tbl.15',
data_dir16:'part.tbl.16',
data_dir17:'part.tbl.17',
data_dir18:'part.tbl.18',
data_dir19:'part.tbl.19',
data_dir20:'part.tbl.20',
data_dir21:'part.tbl.21',
data_dir22:'part.tbl.22',
data_dir23:'part.tbl.23',
data_dir24:'part.tbl.24',
data_dir25:'part.tbl.25',
data_dir26:'part.tbl.26',
data_dir27:'part.tbl.27',
data_dir28:'part.tbl.28',
data_dir29:'part.tbl.29',

```

```

data_dir30:'part.tbl.30',
data_dir31:'part.tbl.31',
data_dir32:'part.tbl.32',
data_dir33:'part.tbl.33',
data_dir34:'part.tbl.34',
data_dir35:'part.tbl.35',
data_dir36:'part.tbl.36',
data_dir37:'part.tbl.37',
data_dir38:'part.tbl.38',
data_dir39:'part.tbl.39',
data_dir40:'part.tbl.40',
data_dir41:'part.tbl.41',
data_dir42:'part.tbl.42',
data_dir43:'part.tbl.43',
data_dir44:'part.tbl.44',
data_dir45:'part.tbl.45',
data_dir46:'part.tbl.46',
data_dir47:'part.tbl.47',
data_dir48:'part.tbl.48',
data_dir49:'part.tbl.49',
data_dir50:'part.tbl.50',
data_dir51:'part.tbl.51',
data_dir52:'part.tbl.52',
data_dir53:'part.tbl.53',
data_dir54:'part.tbl.54',
data_dir55:'part.tbl.55',
data_dir56:'part.tbl.56',
data_dir57:'part.tbl.57',
data_dir58:'part.tbl.58',
data_dir59:'part.tbl.59',
data_dir60:'part.tbl.60',
data_dir61:'part.tbl.61',
data_dir62:'part.tbl.62',
data_dir63:'part.tbl.63',
data_dir64:'part.tbl.64',
data_dir65:'part.tbl.65',
data_dir66:'part.tbl.66',
data_dir67:'part.tbl.67',
data_dir68:'part.tbl.68',
data_dir69:'part.tbl.69',
data_dir70:'part.tbl.70',
data_dir71:'part.tbl.71',
data_dir72:'part.tbl.72',
data_dir73:'part.tbl.73',
data_dir74:'part.tbl.74',
data_dir75:'part.tbl.75',
data_dir76:'part.tbl.76',
data_dir77:'part.tbl.77',
data_dir78:'part.tbl.78',
data_dir79:'part.tbl.79',
data_dir80:'part.tbl.80',
data_dir81:'part.tbl.81',
data_dir82:'part.tbl.82',
data_dir83:'part.tbl.83',
data_dir84:'part.tbl.84',
data_dir85:'part.tbl.85',
data_dir86:'part.tbl.86',
data_dir87:'part.tbl.87',
data_dir88:'part.tbl.88',
data_dir89:'part.tbl.89',
data_dir90:'part.tbl.90',
data_dir91:'part.tbl.91',
data_dir92:'part.tbl.92',
data_dir93:'part.tbl.93',
data_dir94:'part.tbl.94',
data_dir95:'part.tbl.95',
data_dir96:'part.tbl.96'
))
reject limit unlimited
parallel 144;

rem 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 data_dir1
access parameters
(
  records delimited by newline
  badfile 'c_et.bad'
  logfile 'c_et.log'
  fields terminated by '|'
  missing field values are null
)
location (
data_dir1:'customer.tbl.1',
data_dir2:'customer.tbl.2',
data_dir3:'customer.tbl.3',
data_dir4:'customer.tbl.4',
data_dir5:'customer.tbl.5',
data_dir6:'customer.tbl.6',
data_dir7:'customer.tbl.7',
data_dir8:'customer.tbl.8',
data_dir9:'customer.tbl.9',
data_dir10:'customer.tbl.10',
data_dir11:'customer.tbl.11',
data_dir12:'customer.tbl.12',
data_dir13:'customer.tbl.13',
data_dir14:'customer.tbl.14',
data_dir15:'customer.tbl.15',
data_dir16:'customer.tbl.16',
data_dir17:'customer.tbl.17',
data_dir18:'customer.tbl.18',
data_dir19:'customer.tbl.19',
data_dir20:'customer.tbl.20',
data_dir21:'customer.tbl.21',
data_dir22:'customer.tbl.22',
data_dir23:'customer.tbl.23',
data_dir24:'customer.tbl.24',
data_dir25:'customer.tbl.25',
data_dir26:'customer.tbl.26',
data_dir27:'customer.tbl.27',
data_dir28:'customer.tbl.28',
data_dir29:'customer.tbl.29',
data_dir30:'customer.tbl.30',
data_dir31:'customer.tbl.31',
data_dir32:'customer.tbl.32',
data_dir33:'customer.tbl.33',
data_dir34:'customer.tbl.34',
data_dir35:'customer.tbl.35',
data_dir36:'customer.tbl.36',
data_dir37:'customer.tbl.37',
data_dir38:'customer.tbl.38',
data_dir39:'customer.tbl.39',
data_dir40:'customer.tbl.40',
data_dir41:'customer.tbl.41',
data_dir42:'customer.tbl.42',
data_dir43:'customer.tbl.43',
data_dir44:'customer.tbl.44',
data_dir45:'customer.tbl.45',
data_dir46:'customer.tbl.46',
data_dir47:'customer.tbl.47',
data_dir48:'customer.tbl.48',
data_dir49:'customer.tbl.49',
data_dir50:'customer.tbl.50',
data_dir51:'customer.tbl.51',
data_dir52:'customer.tbl.52',
data_dir53:'customer.tbl.53',
data_dir54:'customer.tbl.54',
data_dir55:'customer.tbl.55',
data_dir56:'customer.tbl.56',
data_dir57:'customer.tbl.57',
data_dir58:'customer.tbl.58',
data_dir59:'customer.tbl.59',
data_dir60:'customer.tbl.60',
data_dir61:'customer.tbl.61',
data_dir62:'customer.tbl.62',
data_dir63:'customer.tbl.63',
data_dir64:'customer.tbl.64',
data_dir65:'customer.tbl.65',
data_dir66:'customer.tbl.66',
data_dir67:'customer.tbl.67',
data_dir68:'customer.tbl.68',
data_dir69:'customer.tbl.69',
data_dir70:'customer.tbl.70',
data_dir71:'customer.tbl.71',
data_dir72:'customer.tbl.72',
data_dir73:'customer.tbl.73',
data_dir74:'customer.tbl.74',
data_dir75:'customer.tbl.75',
data_dir76:'customer.tbl.76',
data_dir77:'customer.tbl.77',
data_dir78:'customer.tbl.78',

```

```

data_dir79:'customer.tbl.79',
data_dir80:'customer.tbl.80',
data_dir81:'customer.tbl.81',
data_dir82:'customer.tbl.82',
data_dir83:'customer.tbl.83',
data_dir84:'customer.tbl.84',
data_dir85:'customer.tbl.85',
data_dir86:'customer.tbl.86',
data_dir87:'customer.tbl.87',
data_dir88:'customer.tbl.88',
data_dir89:'customer.tbl.89',
data_dir90:'customer.tbl.90',
data_dir91:'customer.tbl.91',
data_dir92:'customer.tbl.92',
data_dir93:'customer.tbl.93',
data_dir94:'customer.tbl.94',
data_dir95:'customer.tbl.95',
data_dir96:'customer.tbl.96'
))
reject limit unlimited
parallel 144;

rem drop table s_et;
create table s_et(
  s_suppkey          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 data_dir1
access parameters
(
  records delimited by newline
  badfile 's_et.bad'
  logfile 's_et.log'
  fields terminated by '|'
  missing field values are null
)
location (
data_dir1:'supplier.tbl.1',
data_dir2:'supplier.tbl.2',
data_dir3:'supplier.tbl.3',
data_dir4:'supplier.tbl.4',
data_dir5:'supplier.tbl.5',
data_dir6:'supplier.tbl.6',
data_dir7:'supplier.tbl.7',
data_dir8:'supplier.tbl.8',
data_dir9:'supplier.tbl.9',
data_dir10:'supplier.tbl.10',
data_dir11:'supplier.tbl.11',
data_dir12:'supplier.tbl.12',
data_dir13:'supplier.tbl.13',
data_dir14:'supplier.tbl.14',
data_dir15:'supplier.tbl.15',
data_dir16:'supplier.tbl.16',
data_dir17:'supplier.tbl.17',
data_dir18:'supplier.tbl.18',
data_dir19:'supplier.tbl.19',
data_dir20:'supplier.tbl.20',
data_dir21:'supplier.tbl.21',
data_dir22:'supplier.tbl.22',
data_dir23:'supplier.tbl.23',
data_dir24:'supplier.tbl.24',
data_dir25:'supplier.tbl.25',
data_dir26:'supplier.tbl.26',
data_dir27:'supplier.tbl.27',
data_dir28:'supplier.tbl.28',
data_dir29:'supplier.tbl.29',
data_dir30:'supplier.tbl.30',
data_dir31:'supplier.tbl.31',
data_dir32:'supplier.tbl.32',
data_dir33:'supplier.tbl.33',
data_dir34:'supplier.tbl.34',
data_dir35:'supplier.tbl.35',
data_dir36:'supplier.tbl.36',
data_dir37:'supplier.tbl.37',
data_dir38:'supplier.tbl.38',
data_dir39:'supplier.tbl.39',
data_dir40:'supplier.tbl.40',
data_dir41:'supplier.tbl.41',
data_dir42:'supplier.tbl.42',
data_dir43:'supplier.tbl.43',
data_dir44:'supplier.tbl.44',
data_dir45:'supplier.tbl.45',
data_dir46:'supplier.tbl.46',
data_dir47:'supplier.tbl.47',
data_dir48:'supplier.tbl.48',
data_dir49:'supplier.tbl.49',
data_dir50:'supplier.tbl.50',
data_dir51:'supplier.tbl.51',
data_dir52:'supplier.tbl.52',
data_dir53:'supplier.tbl.53',
data_dir54:'supplier.tbl.54',
data_dir55:'supplier.tbl.55',
data_dir56:'supplier.tbl.56',
data_dir57:'supplier.tbl.57',
data_dir58:'supplier.tbl.58',
data_dir59:'supplier.tbl.59',
data_dir60:'supplier.tbl.60',
data_dir61:'supplier.tbl.61',
data_dir62:'supplier.tbl.62',
data_dir63:'supplier.tbl.63',
data_dir64:'supplier.tbl.64',
data_dir65:'supplier.tbl.65',
data_dir66:'supplier.tbl.66',
data_dir67:'supplier.tbl.67',
data_dir68:'supplier.tbl.68',
data_dir69:'supplier.tbl.69',
data_dir70:'supplier.tbl.70',
data_dir71:'supplier.tbl.71',
data_dir72:'supplier.tbl.72',
data_dir73:'supplier.tbl.73',
data_dir74:'supplier.tbl.74',
data_dir75:'supplier.tbl.75',
data_dir76:'supplier.tbl.76',
data_dir77:'supplier.tbl.77',
data_dir78:'supplier.tbl.78',
data_dir79:'supplier.tbl.79',
data_dir80:'supplier.tbl.80',
data_dir81:'supplier.tbl.81',
data_dir82:'supplier.tbl.82',
data_dir83:'supplier.tbl.83',
data_dir84:'supplier.tbl.84',
data_dir85:'supplier.tbl.85',
data_dir86:'supplier.tbl.86',
data_dir87:'supplier.tbl.87',
data_dir88:'supplier.tbl.88',
data_dir89:'supplier.tbl.89',
data_dir90:'supplier.tbl.90',
data_dir91:'supplier.tbl.91',
data_dir92:'supplier.tbl.92',
data_dir93:'supplier.tbl.93',
data_dir94:'supplier.tbl.94',
data_dir95:'supplier.tbl.95',
data_dir96:'supplier.tbl.96'
))
reject limit unlimited
parallel 144;

rem 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 data_dir1
access parameters
(
  records delimited by newline
  badfile 'n_et.bad'
  logfile 'n_et.log'
  fields terminated by '|'
  missing field values are null
)
location (
data_dir96:'nation.tbl')
reject limit unlimited;

rem 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 data_dir1
access parameters
(
    records delimited by newline
    badfile 'r_et.bad'
    logfile 'r_et.log'
    fields terminated by '|'
    missing field values are null
)
    location (
        data_dir96:'region.tbl'))
reject limit unlimited;

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 2
pctused 98
intrans 10
storage (initial 20m freelist groups 4 freelists 84)
parallel 144
nologging
compress
partition by range (l_shipdate)
subpartition by hash (l_partkey)
subpartitions 144
(
    partition item1 values less than (to_date('1992-01-01','YYYY-MM-DD'))
    tablespace ts_l1
    ,
    partition item2 values less than (to_date('1992-02-01','YYYY-MM-DD'))
    tablespace ts_l2
    ,
    partition item3 values less than (to_date('1992-03-01','YYYY-MM-DD'))
    tablespace ts_l3
    ,
    partition item4 values less than (to_date('1992-04-01','YYYY-MM-DD'))
    tablespace ts_l4
    ,
    partition item5 values less than (to_date('1992-05-01','YYYY-MM-DD'))
    tablespace ts_l5
    ,
    partition item6 values less than (to_date('1992-06-01','YYYY-MM-DD'))
    tablespace ts_l6
    ,
    partition item7 values less than (to_date('1992-07-01','YYYY-MM-DD'))
    tablespace ts_l7
    ,
    partition item8 values less than (to_date('1992-08-01','YYYY-MM-DD'))
    tablespace ts_l8
    ,
    partition item9 values less than (to_date('1992-09-01','YYYY-MM-DD'))
    tablespace ts_l9
    ,
    partition item10 values less than (to_date('1992-10-01','YYYY-MM-DD'))
    tablespace ts_l10
    ,
    partition item11 values less than (to_date('1992-11-01','YYYY-MM-DD'))
    tablespace ts_l11
    ,
    partition item12 values less than (to_date('1992-12-01','YYYY-MM-DD'))
    tablespace ts_l12
    ,
    partition item13 values less than (to_date('1993-01-01','YYYY-MM-DD'))
    tablespace ts_l13
    ,
    partition item14 values less than (to_date('1993-02-01','YYYY-MM-DD'))
    tablespace ts_l14
    ,
    partition item15 values less than (to_date('1993-03-01','YYYY-MM-DD'))
    tablespace ts_l15
    ,
    partition item16 values less than (to_date('1993-04-01','YYYY-MM-DD'))
    tablespace ts_l16
    ,
    partition item17 values less than (to_date('1993-05-01','YYYY-MM-DD'))
    tablespace ts_l17
    ,
    partition item18 values less than (to_date('1993-06-01','YYYY-MM-DD'))
    tablespace ts_l18
    ,
    partition item19 values less than (to_date('1993-07-01','YYYY-MM-DD'))
    tablespace ts_l19
    ,
    partition item20 values less than (to_date('1993-08-01','YYYY-MM-DD'))
    tablespace ts_l20
    ,
    partition item21 values less than (to_date('1993-09-01','YYYY-MM-DD'))
    tablespace ts_l21
    ,
    partition item22 values less than (to_date('1993-10-01','YYYY-MM-DD'))
    tablespace ts_l1
    ,
    partition item23 values less than (to_date('1993-11-01','YYYY-MM-DD'))
    tablespace ts_l2
    ,
    partition item24 values less than (to_date('1993-12-01','YYYY-MM-DD'))
    tablespace ts_l3
    ,
    partition item25 values less than (to_date('1994-01-01','YYYY-MM-DD'))
    tablespace ts_l4
    ,
    partition item26 values less than (to_date('1994-02-01','YYYY-MM-DD'))
    tablespace ts_l5
    ,
    partition item27 values less than (to_date('1994-03-01','YYYY-MM-DD'))
    tablespace ts_l6
    ,
    partition item28 values less than (to_date('1994-04-01','YYYY-MM-DD'))
    tablespace ts_l7
    ,
    partition item29 values less than (to_date('1994-05-01','YYYY-MM-DD'))
    tablespace ts_l8
    ,
    partition item30 values less than (to_date('1994-06-01','YYYY-MM-DD'))
    tablespace ts_l9
    ,
    partition item31 values less than (to_date('1994-07-01','YYYY-MM-DD'))
    tablespace ts_l10
    ,
    partition item32 values less than (to_date('1994-08-01','YYYY-MM-DD'))
    tablespace ts_l11
)

```

```

partition item33 values less than (to_date('1994-09-
01','YYYY-MM-DD'))
tablespace ts_l12
partition item34 values less than (to_date('1994-10-
01','YYYY-MM-DD'))
tablespace ts_l13
partition item35 values less than (to_date('1994-11-
01','YYYY-MM-DD'))
tablespace ts_l14
partition item36 values less than (to_date('1994-12-
01','YYYY-MM-DD'))
tablespace ts_l15
partition item37 values less than (to_date('1995-01-
01','YYYY-MM-DD'))
tablespace ts_l16
partition item38 values less than (to_date('1995-02-
01','YYYY-MM-DD'))
tablespace ts_l17
partition item39 values less than (to_date('1995-03-
01','YYYY-MM-DD'))
tablespace ts_l18
partition item40 values less than (to_date('1995-04-
01','YYYY-MM-DD'))
tablespace ts_l19
partition item41 values less than (to_date('1995-05-
01','YYYY-MM-DD'))
tablespace ts_l20
partition item42 values less than (to_date('1995-06-
01','YYYY-MM-DD'))
tablespace ts_l21
partition item43 values less than (to_date('1995-07-
01','YYYY-MM-DD'))
tablespace ts_l1
partition item44 values less than (to_date('1995-08-
01','YYYY-MM-DD'))
tablespace ts_l2
partition item45 values less than (to_date('1995-09-
01','YYYY-MM-DD'))
tablespace ts_l3
partition item46 values less than (to_date('1995-10-
01','YYYY-MM-DD'))
tablespace ts_l4
partition item47 values less than (to_date('1995-11-
01','YYYY-MM-DD'))
tablespace ts_l5
partition item48 values less than (to_date('1995-12-
01','YYYY-MM-DD'))
tablespace ts_l6
partition item49 values less than (to_date('1996-01-
01','YYYY-MM-DD'))
tablespace ts_l7
partition item50 values less than (to_date('1996-02-
01','YYYY-MM-DD'))
tablespace ts_l8
partition item51 values less than (to_date('1996-03-
01','YYYY-MM-DD'))
tablespace ts_l9
partition item52 values less than (to_date('1996-04-
01','YYYY-MM-DD'))
tablespace ts_l10
partition item53 values less than (to_date('1996-05-
01','YYYY-MM-DD'))
tablespace ts_l11
partition item54 values less than (to_date('1996-06-
01','YYYY-MM-DD'))
tablespace ts_l12
partition item55 values less than (to_date('1996-07-
01','YYYY-MM-DD'))
tablespace ts_l13
partition item56 values less than (to_date('1996-08-
01','YYYY-MM-DD'))
tablespace ts_l14
partition item57 values less than (to_date('1996-09-
01','YYYY-MM-DD'))
tablespace ts_l15
partition item58 values less than (to_date('1996-10-
01','YYYY-MM-DD'))
tablespace ts_l16
partition item59 values less than (to_date('1996-11-
01','YYYY-MM-DD'))
tablespace ts_l17
partition item60 values less than (to_date('1996-12-
01','YYYY-MM-DD'))
tablespace ts_l18
partition item61 values less than (to_date('1997-01-
01','YYYY-MM-DD'))
tablespace ts_l19
partition item62 values less than (to_date('1997-02-
01','YYYY-MM-DD'))
tablespace ts_l20
partition item63 values less than (to_date('1997-03-
01','YYYY-MM-DD'))
tablespace ts_l21
partition item64 values less than (to_date('1997-04-
01','YYYY-MM-DD'))
tablespace ts_l1
partition item65 values less than (to_date('1997-05-
01','YYYY-MM-DD'))
tablespace ts_l2
partition item66 values less than (to_date('1997-06-
01','YYYY-MM-DD'))
tablespace ts_l3
partition item67 values less than (to_date('1997-07-
01','YYYY-MM-DD'))
tablespace ts_l4
partition item68 values less than (to_date('1997-08-
01','YYYY-MM-DD'))
tablespace ts_l5
partition item69 values less than (to_date('1997-09-
01','YYYY-MM-DD'))
tablespace ts_l6
partition item70 values less than (to_date('1997-10-
01','YYYY-MM-DD'))
tablespace ts_l7
partition item71 values less than (to_date('1997-11-
01','YYYY-MM-DD'))
tablespace ts_l8
partition item72 values less than (to_date('1997-12-
01','YYYY-MM-DD'))
tablespace ts_l9
partition item73 values less than (to_date('1998-01-
01','YYYY-MM-DD'))
tablespace ts_l10
partition item74 values less than (to_date('1998-02-
01','YYYY-MM-DD'))
tablespace ts_l11
partition item75 values less than (to_date('1998-03-
01','YYYY-MM-DD'))
tablespace ts_l12

```

```

partition item76 values less than (to_date('1998-04-
01','YYYY-MM-DD'))
tablespace ts_l13
partition item77 values less than (to_date('1998-05-
01','YYYY-MM-DD'))
tablespace ts_l14
partition item78 values less than (to_date('1998-06-
01','YYYY-MM-DD'))
tablespace ts_l15
partition item79 values less than (to_date('1998-07-
01','YYYY-MM-DD'))
tablespace ts_l16
partition item80 values less than (to_date('1998-08-
01','YYYY-MM-DD'))
tablespace ts_l17
partition item81 values less than (to_date('1998-09-
01','YYYY-MM-DD'))
tablespace ts_l18
partition item82 values less than (to_date('1998-10-
01','YYYY-MM-DD'))
tablespace ts_l19
partition item83 values less than (to_date('1998-11-
01','YYYY-MM-DD'))
tablespace ts_l20
partition item84 values less than (MAXVALUE)
tablespace ts_l21
)
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_linenumber,l_shipinstruct,l_comment from l_et
order by l_orderkey;

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 2
pctused 98
intrans 10
storage (initial 10m freelist groups 4 freelists 84)
parallel 144
nologging
compress
partition by range (o_orderdate)
subpartition by hash (o_custkey)
subpartitions 144
(
partition ord1 values less than (to_date('1992-01-
01','YYYY-MM-DD'))
tablespace ts_o1
partition ord2 values less than (to_date('1992-02-
01','YYYY-MM-DD'))
tablespace ts_o2
partition ord3 values less than (to_date('1992-03-
01','YYYY-MM-DD'))
tablespace ts_o3
partition ord4 values less than (to_date('1992-04-
01','YYYY-MM-DD'))
tablespace ts_o4
partition ord5 values less than (to_date('1992-05-
01','YYYY-MM-DD'))
tablespace ts_o5
partition ord6 values less than (to_date('1992-06-
01','YYYY-MM-DD'))
tablespace ts_o6
partition ord7 values less than (to_date('1992-07-
01','YYYY-MM-DD'))
tablespace ts_o7
partition ord8 values less than (to_date('1992-08-
01','YYYY-MM-DD'))
tablespace ts_o8
partition ord9 values less than (to_date('1992-09-
01','YYYY-MM-DD'))
tablespace ts_o9
partition ord10 values less than (to_date('1992-10-
01','YYYY-MM-DD'))
tablespace ts_o10
partition ord11 values less than (to_date('1992-11-
01','YYYY-MM-DD'))
tablespace ts_o11
partition ord12 values less than (to_date('1992-12-
01','YYYY-MM-DD'))
tablespace ts_o12
partition ord13 values less than (to_date('1993-01-
01','YYYY-MM-DD'))
tablespace ts_o13
partition ord14 values less than (to_date('1993-02-
01','YYYY-MM-DD'))
tablespace ts_o14
partition ord15 values less than (to_date('1993-03-
01','YYYY-MM-DD'))
tablespace ts_o15
partition ord16 values less than (to_date('1993-04-
01','YYYY-MM-DD'))
tablespace ts_o16
partition ord17 values less than (to_date('1993-05-
01','YYYY-MM-DD'))
tablespace ts_o17
partition ord18 values less than (to_date('1993-06-
01','YYYY-MM-DD'))
tablespace ts_o18
partition ord19 values less than (to_date('1993-07-
01','YYYY-MM-DD'))
tablespace ts_o19
partition ord20 values less than (to_date('1993-08-
01','YYYY-MM-DD'))
tablespace ts_o20
partition ord21 values less than (to_date('1993-09-
01','YYYY-MM-DD'))
tablespace ts_o21
partition ord22 values less than (to_date('1993-10-
01','YYYY-MM-DD'))
tablespace ts_o1
partition ord23 values less than (to_date('1993-11-
01','YYYY-MM-DD'))
tablespace ts_o2
partition ord24 values less than (to_date('1993-12-
01','YYYY-MM-DD'))
tablespace ts_o3
partition ord25 values less than (to_date('1994-01-
01','YYYY-MM-DD'))
tablespace ts_o4
partition ord26 values less than (to_date('1994-02-
01','YYYY-MM-DD'))
tablespace ts_o5
partition ord27 values less than (to_date('1994-03-
01','YYYY-MM-DD'))
tablespace ts_o6

```



```

partition ord28 values less than (to_date('1994-04-
01','YYYY-MM-DD'))
tablespace ts_o7
partition ord29 values less than (to_date('1994-05-
01','YYYY-MM-DD'))
tablespace ts_o8
partition ord30 values less than (to_date('1994-06-
01','YYYY-MM-DD'))
tablespace ts_o9
partition ord31 values less than (to_date('1994-07-
01','YYYY-MM-DD'))
tablespace ts_o10
partition ord32 values less than (to_date('1994-08-
01','YYYY-MM-DD'))
tablespace ts_o11
partition ord33 values less than (to_date('1994-09-
01','YYYY-MM-DD'))
tablespace ts_o12
partition ord34 values less than (to_date('1994-10-
01','YYYY-MM-DD'))
tablespace ts_o13
partition ord35 values less than (to_date('1994-11-
01','YYYY-MM-DD'))
tablespace ts_o14
partition ord36 values less than (to_date('1994-12-
01','YYYY-MM-DD'))
tablespace ts_o15
partition ord37 values less than (to_date('1995-01-
01','YYYY-MM-DD'))
tablespace ts_o16
partition ord38 values less than (to_date('1995-02-
01','YYYY-MM-DD'))
tablespace ts_o17
partition ord39 values less than (to_date('1995-03-
01','YYYY-MM-DD'))
tablespace ts_o18
partition ord40 values less than (to_date('1995-04-
01','YYYY-MM-DD'))
tablespace ts_o19
partition ord41 values less than (to_date('1995-05-
01','YYYY-MM-DD'))
tablespace ts_o20
partition ord42 values less than (to_date('1995-06-
01','YYYY-MM-DD'))
tablespace ts_o21
partition ord43 values less than (to_date('1995-07-
01','YYYY-MM-DD'))
tablespace ts_o1
partition ord44 values less than (to_date('1995-08-
01','YYYY-MM-DD'))
tablespace ts_o2
partition ord45 values less than (to_date('1995-09-
01','YYYY-MM-DD'))
tablespace ts_o3
partition ord46 values less than (to_date('1995-10-
01','YYYY-MM-DD'))
tablespace ts_o4
partition ord47 values less than (to_date('1995-11-
01','YYYY-MM-DD'))
tablespace ts_o5
partition ord48 values less than (to_date('1995-12-
01','YYYY-MM-DD'))
tablespace ts_o6
partition ord49 values less than (to_date('1996-01-
01','YYYY-MM-DD'))
tablespace ts_o7
partition ord50 values less than (to_date('1996-02-
01','YYYY-MM-DD'))
tablespace ts_o8
partition ord51 values less than (to_date('1996-03-
01','YYYY-MM-DD'))
tablespace ts_o9
partition ord52 values less than (to_date('1996-04-
01','YYYY-MM-DD'))
tablespace ts_o10
partition ord53 values less than (to_date('1996-05-
01','YYYY-MM-DD'))
tablespace ts_o11
partition ord54 values less than (to_date('1996-06-
01','YYYY-MM-DD'))
tablespace ts_o12
partition ord55 values less than (to_date('1996-07-
01','YYYY-MM-DD'))
tablespace ts_o13
partition ord56 values less than (to_date('1996-08-
01','YYYY-MM-DD'))
tablespace ts_o14
partition ord57 values less than (to_date('1996-09-
01','YYYY-MM-DD'))
tablespace ts_o15
partition ord58 values less than (to_date('1996-10-
01','YYYY-MM-DD'))
tablespace ts_o16
partition ord59 values less than (to_date('1996-11-
01','YYYY-MM-DD'))
tablespace ts_o17
partition ord60 values less than (to_date('1996-12-
01','YYYY-MM-DD'))
tablespace ts_o18
partition ord61 values less than (to_date('1997-01-
01','YYYY-MM-DD'))
tablespace ts_o19
partition ord62 values less than (to_date('1997-02-
01','YYYY-MM-DD'))
tablespace ts_o20
partition ord63 values less than (to_date('1997-03-
01','YYYY-MM-DD'))
tablespace ts_o21
partition ord64 values less than (to_date('1997-04-
01','YYYY-MM-DD'))
tablespace ts_o1
partition ord65 values less than (to_date('1997-05-
01','YYYY-MM-DD'))
tablespace ts_o2
partition ord66 values less than (to_date('1997-06-
01','YYYY-MM-DD'))
tablespace ts_o3
partition ord67 values less than (to_date('1997-07-
01','YYYY-MM-DD'))
tablespace ts_o4
partition ord68 values less than (to_date('1997-08-
01','YYYY-MM-DD'))
tablespace ts_o5
partition ord69 values less than (to_date('1997-09-
01','YYYY-MM-DD'))
tablespace ts_o6
partition ord70 values less than (to_date('1997-10-
01','YYYY-MM-DD'))
tablespace ts_o7

```

```

partition ord71 values less than (to_date('1997-11-
01','YYYY-MM-DD'))
tablespace ts_o8
',
partition ord72 values less than (to_date('1997-12-
01','YYYY-MM-DD'))
tablespace ts_o9
',
partition ord73 values less than (to_date('1998-01-
01','YYYY-MM-DD'))
tablespace ts_o10
',
partition ord74 values less than (to_date('1998-02-
01','YYYY-MM-DD'))
tablespace ts_o11
',
partition ord75 values less than (to_date('1998-03-
01','YYYY-MM-DD'))
tablespace ts_o12
',
partition ord76 values less than (to_date('1998-04-
01','YYYY-MM-DD'))
tablespace ts_o13
',
partition ord77 values less than (to_date('1998-05-
01','YYYY-MM-DD'))
tablespace ts_o14
',
partition ord78 values less than (to_date('1998-06-
01','YYYY-MM-DD'))
tablespace ts_o15
',
partition ord79 values less than (to_date('1998-07-
01','YYYY-MM-DD'))
tablespace ts_o16
',
partition ord80 values less than (to_date('1998-08-
01','YYYY-MM-DD'))
tablespace ts_o17
',
partition ord81 values less than (to_date('1998-09-
01','YYYY-MM-DD'))
tablespace ts_o18
',
partition ord82 values less than (to_date('1998-10-
01','YYYY-MM-DD'))
tablespace ts_o19
',
partition ord83 values less than (to_date('1998-11-
01','YYYY-MM-DD'))
tablespace ts_o20
',
partition ord84 values less than (MAXVALUE)
tablespace ts_o21
)
as select o_orderdate, o_orderkey, o_custkey,
o_orderpriority, o_shippriority, o_clerk,
o_orderstatus, o_totalprice, o_comment from o_et
order by o_orderkey;

rem drop table partsupp;
create table partsupp(
    ps_partkey          NOT NULL,
    ps_suppkey          NOT NULL,
    ps_supplycost       ,
    ps_availqty         ,
    ps_comment          ,
)
pctfree 0
pctused 99
parallel 144
nologging
compress
partition by hash (ps_partkey)
partitions 144
tablespace ts_ps
--storage (initial 1500m)
storage (initial 20m)
as select ps_partkey, ps_suppkey, ps_supplycost,
ps_availqty, ps_comment from ps_et;

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
tablespace ts_p
storage (freelists 99)
parallel 144
nologging
compress
partition by hash (p_partkey)
partitions 144
as select p_partkey, p_type, p_size, p_brand, p_name,
p_container, p_mfgr, p_retailprice, p_comment from
p_et;

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
tablespace ts_c
storage (freelists 99)
parallel 144
nologging
compress
partition by hash (c_custkey)
partitions 144
as select c_custkey, c_mktsegment, c_nationkey,
c_name, c_address, c_phone, c_acctbal, c_comment from
c_et;

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
tablespace ts_s
storage (freelists 99)
parallel 144
nologging
compress
partition by hash (s_suppkey)
partitions 144
as select s_suppkey, s_nationkey, s_comment, s_name,
s_address, s_phone, s_acctbal from s_et;

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

rem drop table region;
create table region(
    r_regionkey        ,
    r_name             ,
    r_comment          )
tablespace ts_default
as select * from r_et;

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;
```

```
!
```

```
echo END Data Loading at `date`
```

```
=====
```

## ixcre.sh

```
#!/bin/ksh
#####
# Index Creation Phase
#####
echo START Index Creation at `date`

sqlplus / as sysdba<<EOF
connect tpch/tpch;

set echo on
set timing on

rem drop index i_l_orderkey;
create index i_l_orderkey
on lineitem (l_orderkey)
global partition by hash (l_orderkey)
partitions 144
pctfree 5
initrans 10
tablespace ts_i_lorderkey
storage (freelist groups 4 freelists 84)
parallel 144
compute statistics
nologging ;

rem drop index i_o_orderkey;
create unique index i_o_orderkey
on orders (o_orderkey)
global partition by hash (o_orderkey)
partitions 144
pctfree 5
initrans 10
tablespace ts_i_oorderkey
storage (freelist groups 4 freelists 84 )
parallel 144
compute statistics
nologging ;

rem drop index i_c_custkey;
create unique index i_c_custkey
on customer (c_custkey)
pctfree 0
initrans 10
tablespace ts_i_ccustkey
storage (freelists 84)
parallel 144
compute statistics
nologging ;

rem drop index ps_pkey_skey;
create unique index ps_pkey_skey
on partsupp (ps_partkey,ps_suppkey)
global partition by hash (ps_partkey)
partitions 144
pctfree 0
initrans 10
tablespace ts_ps
parallel 144
compute statistics
nologging ;

alter index i_o_orderkey allocate extent (size 100m
instance 1);
alter index i_l_orderkey allocate extent (size 40m
instance 1);
EOF
```

```
echo END Index Creation at `date`
```

```
=====
```

## anlyz.sh

```
#!/bin/ksh
#####
# Analyze Phase
#####
```

```
echo START Analyze at `date`
sqlplus / as sysdba<<EOF
connect tpch/tpch;
set echo on
set timing on
execute dbms_stats.gather_schema_stats('TPCH' ,
estimate_percent => 1, degree => 144
, granularity => 'GLOBAL', method_opt => 'for all
columns size 1' );
connect / as sysdba
execute dbms_stats.gather_system_stats;
execute dbms_scheduler.disable
('AUTO_TASKS_JOB_CLASS');
EOF
```

```
echo END Analyze at `date`
```

```
=====
```

## ACID Test Source Code

```
=====
```

```
=====
```

### atranspl.c

```

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

#include "atranspl.h"

/* Declare error handling functions */

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;
OCIError *errhp = NULL;
OCISvcCtx *tpcsvc = 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;

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>\n[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\n");
    fprintf(stderr, "    delta      :1 to generate new
random delta, otherwise obtain delta from input\n\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 tcpcd/tpcd\n");
    fprintf(stderr, "    t<trigger>      :
Trigger Time - sleep <trigger> seconds before
start\n\n");
    fprintf(stderr, "    s<sleep>      :
Sleep Time - sleep <sleep> seconds before commit or
rollback\n\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) OCIErrGet(errhp,1,NULL, (sb4 *)
&errcode, (text*) msg,
                2048,OCI_HTYPE_ERROR);
    else
        (void) OCIErrGet(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) OCIErrGet(errhp,1,NULL, (sb4 *)
&errcode, (text*) msg,
                    2048,OCI_HTYPE_ERROR);
            else
                (void) OCIErrGet(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
void main(argc,argv)
#endif
{
    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\n");

        /* Initialize the cursors etc. */

        (void) ACIDinit();

        /* sleep for some time (triggering) */
        sleep(trig);

        /* start doing the ACID transactions */
        tr_start = gettimeofday();

        /* The number of iteration we will run depends on
the number of */
        /* input lines
        */
        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_eprice);
        FPRTF1(outfile, "l_quantity:      %d\n", (int)
l_quan);
        FPRTF1(outfile, "o_totalprice:    %.2f\n",
o_tprice);
    }

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

    /* Shall we commit? */

    if (BIT(flag, COMMIT)) {
        need_commit = 1;
        while (need_commit) {
            if((status=OCITransCommit
(tpcsvc,errhp,OCI_DEFAULT)) != OCI_SUCCESS) {
                OCIrol(tpcsvc,errhp);
                OCIsexec(tpcsvc,curr,errhp,1);
            } else {
                need_commit = 0;
                curr_time = time(NULL);
                FPRTF2(outfile, "ACID Transaction
iteration %d COMMITED 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: %.2lf\n",
l_neweprice);
        FPRTF1(outfile, "l_quantity:      %d\n", (int)
l_newquan);
        FPRTF1(outfile, "o_totalprice:    %.2lf\n",
o_newtprice);
        FPRTF1(outfile, "l_tax:          %.2lf\n",
l_tax);
        FPRTF1(outfile, "l_discount:     %.2lf\n",
l_disc);
        FPRTF1(outfile, "rprice:        %.2lf\n",
rprice);
        FPRTF1(outfile, "cost:          %.2lf\n",
cost);
        FPRTF(outfile,
"-----
\n");
    }
}

    tr_end = gettime();

    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,errh
p);
OCIaset(tpcusr,OCI_HTYPE_SESSION,lname,strlen
(lname),OCI_ATTR_USERNAME,
errhp);
OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen
(passwd),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,err
hp);

/* Enable session parallel dml */

sprintf((char *) sqlstmt, PDMLTXT);
OCIStmtPrepare(cur_i, errhp, (text *)sqlstmt, strlen
((char *)sqlstmt),
OCI_NTV_SYNTAX,OCI_DEFAULT);
OCIExec(tpcsvc, cur_i, errhp, 1);

/* Enable session parallel ddl */

/*sprintf((char *) sqlstmt, PDDLTX);
OCIStmtPrepare(cur_i, errhp, (text *)sqlstmt, strlen
((char *)sqlstmt),
OCI_NTV_SYNTAX,OCI_DEFAULT);
OCIExec(tpcsvc, cur_i, errhp, 1);*/

/* Make session serializable */

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

/* Set optimizer_index_cost_adj = 25 */

sprintf((char *) sqlstmt, OICATXT);
OCIStmtPrepare(cur_i, errhp, (text *)sqlstmt, strlen
((char *)sqlstmt),
OCI_NTV_SYNTAX,OCI_DEFAULT);
OCIExec(tpcsvc, cur_i, 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(cur_i, errhp, sqlstmt, strlen((char *)
sqlstmt),OCI_NTV_SYNTAX,OCI_DEFAULT);

OCIbname(cur_i, &l_keyi_bp, errhp, ":l_key", ADR(l_key),
SIZ(l_key), SQT_INT);
OCIbname(cur_i, &o_keyi_bp, errhp, ":o_key", ADR(o_key),
SIZ(o_key), SQT_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), SQT_INT);
OCIbname(curr, o_key_bp, errhp, ":o_key", ADR(o_key),
SIZ(o_key), SQT_INT);
OCIbname(curr, delta_bp, errhp, ":delta", ADR(delta),
SIZ(delta), SQT_INT);
OCIbname(curr, l_pkey_bp, errhp, ":l_pkey", ADR
(l_pkey), SIZ(l_pkey), SQT_INT);
OCIbname(curr, l_skey_bp, errhp, ":l_skey", ADR
(l_skey), SIZ(l_skey), SQT_INT);
OCIbname(curr, l_quan_bp, errhp, ":l_quan", ADR
(l_quan), SIZ(l_quan), SQT_INT);
OCIbname(curr, l_newquan_bp, errhp, ":l_newquan", ADR
(l_newquan),
SIZ(l_newquan), SQT_INT);
OCIbname(curr, l_tax_bp, errhp, ":l_tax", ADR(l_tax),
SIZ(l_tax), SQT_FLT);
OCIbname(curr, l_disc_bp, errhp, ":l_disc", ADR
(l_disc), SIZ(l_disc), SQT_FLT);
OCIbname(curr, l_eprice_bp, errhp, ":l_eprice", ADR
(l_eprice), SIZ(l_eprice),
SQT_FLT);
OCIbname
(curr, l_neweprice_bp, errhp, ":l_neweprice", ADR
(l_neweprice),
SIZ(l_neweprice), SQT_FLT);

OCIbname(curr, o_tprice_bp, errhp, ":o_tprice", ADR
(o_tprice), SIZ(o_tprice),
SQT_FLT);
OCIbname
(curr, o_newtprice_bp, errhp, ":o_newtprice", ADR
(o_newtprice),
SIZ(o_newtprice), SQT_FLT);
OCIbname(curr, rprice_bp, errhp, ":rprice", ADR
(rprice), SIZ(rprice), SQT_FLT);
OCIbname(curr, cost_bp, errhp, ":cost", ADR(cost), SIZ
(cost), SQT_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",ADR
(l_neweprice),
SIZ(l_neweprice),SQLT_FLT);
OCIbname(cure1,l_newquan1_bp,errhp,":l_newquan",ADR
(l_newquan),
SIZ(l_newquan),SQLT_INT);
OCIbname(cure1,o_key1_bp,errhp,":o_key",ADR(o_key),
SIZ(o_key),SQLT_INT);
OCIbname(cure1,l_key1_bp,errhp,":l_key",ADR(l_key),
SIZ(l_key),SQLT_INT);

OCIbname
(cure2,o_newtprice2_bp,errhp,":o_newtprice",ADR
(o_newtprice),
SIZ(o_newtprice),SQLT_FLT);
OCIbname(cure2,o_key2_bp,errhp,":o_key",ADR(o_key),
SIZ(o_key),SQLT_INT);
}

```

## atranspl.h

```

#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(flg,mask) (unsigned) (flg |=(unsigned)
mask)
#define BIT(flg,mask) (unsigned) ((unsigned) flg &
(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 \
DISCARD 0

#define OCIhfree(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=OCIStmtExecute
(svch,stmh,errh,iter,0,NULL,NULL,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), \
progv,progv1,ftype,0,0,0,0,OCI_DEFAULT
T)) != 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), \
progv,progv1,ftype,indp,0,0,0,0,OCI_DEF
AULT)) != OCI_SUCCESS) \
sql_error(errh,status,1); \
else \
DISCARD 0

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

```



```

#define OCIrol(svc, errh) \
    if((status=OCITransRollback
(svc, 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 4)"
#define PDDLTX "alter session force parallel ddl
parallel (degree 4)"
#define OICATXT "alter session set
optimizer_index_cost_adj=25"

#define SQLTXT1 "BEGIN SELECT /*+ index
(lineitem, i_l_orderkey) */ MAX(l_linenum) 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_linenum = :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_linenum = :l_key; END;"

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

#endif /* ATRANSPL_H */

=====
randpsup.c
=====
#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 */
    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");
}

=====
randkey.c
=====
#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)
            (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':
            strcpy((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());

    /* 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
    (passwd),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);
    OCISmtPrepare(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);
}

=====
gettime.c
=====
#define SUN_OS5

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

#if defined(sequent) || defined(SEQ_PXS)
#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_GETRU
#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_GETRU) && !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_GETRU) ||
defined(GETRU_STATS)
#include <sys/time.h>
#endif /* TIME_W_GETTIME || CPU_W_GETRU || GETRU_STATS */

#if defined(CPU_W_GETRU) || defined(GETRU_STATS)
#include <sys/resource.h>
#endif /* CPU_W_GETRU || 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;
#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);
    if (kids)
        diffru (&ru, &kidsru);
    else
        diffru (&ru, &selfru);
    print_ru (fp, &ru);
    fprintf (fp, "\n");
#endif /* GET_P_STATS */
}

getrul (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;
    struct process_stats ru;

    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 */
}

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 */
}

#ifdef GETRU_STATS
    print_ru (fp, ru)
#endif

FILE *fp;
struct rusage *ru;
{
    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;
    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_utime.tv_sec * 1000 +
            (ps->ps_utime.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_utime.tv_sec == ru->ps_utime.tv_sec;
    ru2->ps_utime.tv_usec == ru->ps_utime.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 */

```

### a\_query.sql

```

=====
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

```

=====
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

. $KIT_DIR/env

ITER=3
SF=1
PROG=atranspl
OUT=${ACID_OUT}/atom
USER=${DATABASE_USER}

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

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 ""

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.sh

```

=====
#!/bin/ksh
ade useview acid2 << END
echo I am in atranspl.sh
echo my arguments are $1 $2 $3 $4 $5 $6 $7
/private/tpcd/acid/kit/utils/atranspl $1 $2 $3 $4 $5
$6 $7
exit
END

```

## ckpt.sh

```

=====
#!/bin/ksh

. $KIT_DIR/env

sqlplus -s /NOLOG<< !

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

!

```

## cnt\_hist.sql

```

=====
select count(*) from history;
exit;

```

## consist.sh

```

=====
#!/bin/ksh

. $KIT_DIR/env

KEY=${ACID_OUT}/key$$_
OUTFILE=${ACID_OUT}/consrte
CON1=${ACID_OUT}/conb
CON2=${ACID_OUT}/cona

```

```

CHK=${ACID_OUT}/consckpt
SF=1

/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 "-h                  : 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 ]
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
```

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

```
(sleep $CK; $ACID_DIR/consistency/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

```
set verify off
```

```
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
```

## end acid.sh

```
#!/bin/ksh
```

```
. $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
```

```
ITER=100
STEM=8
PROG=${ACID_DIR}/atranspl
IN=${ACID_DIR}/acid_in
DURA=${DURA_DIR}/drate
OUT=${DURA_DIR}/drate
DSMPL=${DURA_DIR}/durasmpl
KEY=${DURA_DIR}/key${1}_
USER=${DATABASE_USER}
TRIG=1
HCNT=duracnta
```

```
sqlplus $USER @cnt_hist > $DURA_DIR/$HCNT
```

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

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

## isol.sh

```

#!/bin/ksh

. $KIT_DIR/env

RSH=rsh

OH=$ORACLE_HOME
OUT_DIR=$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

randkey 1 1 u"$USER" > $KEYFILE
echo -----
ls -l $KEYFILE
cat $KEYFILE
echo ++++++
OKEY=`cat $KEYFILE | awk '{print $1}'`
echo "o_key is "$OKEY

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
echo before PROG

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

echo PROG 1 1 1 0 i$KEYFILE u$USER s60

sleep 10

echo "Running ACID query 10 seconds AFTER the start of
ACID Transaction" \
>> $TXN2FILE
echo "`date`" >> $TXN2FILE
if [ "$HOST" != "" ]
then
echo "Starting ACID query on node $HOST" >> $TXN2FILE
@$ACID_DIR/isolation/a_query $OKEY >> $TXN2FILE
${RSH} -n ${HOST} a_query.sh $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

. $KIT_DIR/env

RSH=rsh

OH=$ORACLE_HOME
OUT_DIR=$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

randkey 1 1 u"$USER" > $KEYFILE

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

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

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

```



```

sleep 10

echo "Running ACID query 10 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} a_query.sh $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
. $KIT_DIR/env

RSH=rsh

OH=$ORACLE_HOME
OUT_DIR=$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=a_transpl

/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

randkey 1 1 u"$USER" > $KEYFILE
k1=`cat $KEYFILE | awk '{print $1}'`
k2=`cat $KEYFILE | awk '{print $2}'`
k3=`cat $KEYFILE | awk '{print $3}'`
sleep 1

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

sleep 10

if [ "$HOST" != "" ]
then
echo "Starting TXN2 on node $HOST" >> $TXN2FILE
${RSH} -n ${HOST} a_transpl.sh 2 2 1 1 $k1 $k2 $k3
u$USER s1 >> $TXN2FILE &
else
$PROG 2 2 1 1 i$KEYFILE u$USER s1 >> $TXN2FILE &
fi

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

=====
iso4.sh
=====
#!/bin/ksh
. $KIT_DIR/env

RSH=rsh

OH=$ORACLE_HOME
OUT_DIR=$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=a_transpl

/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

randkey 1 1 u"$USER" > $KEYFILE

k1=`cat $KEYFILE | awk '{print $1}'`
k2=`cat $KEYFILE | awk '{print $2}'`
k3=`cat $KEYFILE | awk '{print $3}'`

sleep 1

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

sleep 10

if [ "$HOST" != "" ]
then

```

```

echo "Starting TXN2 on node $HOST" >> $TXN2FILE
${RSH} -n ${HOST} atranspl.sh 2 2 1 1 $k1 $k2 $k3
u$USER s1 >> $TXN2FILE &
else
$PROG 2 2 1 1 i$KEYFILE u$USER s1 >> $TXN2FILE &
fi

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

=====
iso5.sh
=====
#!/bin/ksh
. $KIT_DIR/env

RSH=rsh

OH=$ORACLE_HOME
OUT_DIR=$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 ""
    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

randkey 1 1 u"$USER" > $KEYFILE

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

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

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

```

```

sleep 5
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} a_query2.sh ${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
. $KIT_DIR/env

# May need to change the following:
RSH=rsh

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

randkey 1 1 u"$USER" > $KEYFILE

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

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

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

```

```

# generate key files
randkey 1 0.1 u"$USER" > $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 1 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 1

sleep 2

echo "Running 2nd Query 1 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

=====
run_acid.sh
=====
#!/bin/ksh

. $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
OUT=${DURA_DIR}/drate
DURA=${DURA_DIR}/dura
KEY=${DURA_DIR}/key$$_
USER=${DATABASE_USER}
TRIG=1
HCNT=duracontb

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

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

i=0
while [ $i -lt $STEM ]
do
for j in `head -10 ${KEY}${i} | awk '{printf "%d
", $1}'`
do
sqlplus ${USER} @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
. $KIT_DIR/env

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

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 ${DATABASE_USER} @sample $j
    i=`expr $i + 1`
done

/bin/rm -f /tmp/*key*

```

### sample.sql

```

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;

```

## Disk Configuration Details

#### Basic Assumptions:

1. Use hardware RAID-1 capabilities of StorEdge 3510 FC Arrays
2. Use Oracle Automatic Storage Manager (ASM) for striping

#### Disk Config:

```

1 x StorEdge 6120. Used for oracle software and tpc-h
kit
1 x s1 12 x 73GB. Used for boot tray
108 x StorEdge 3510 FC RAID Arrays w/ 12 10Krpm 73.4G
drives. Used for all database objects (tables,
indexes, controlfiles, redo logs, temporary
tablespace, undo tablespace, system tablespace)

```

Mirroring was implemented by creating 6 x 1+1 RAID-1 logical drives on each StorEdge 3510 FC Array. The logical drives were then partitioned and presented to the SUT as LUNs. Solaris Volume Manager was used to create a metadevice on each LUN.

Striping was implemented using Oracle Automatic Storage Manager (ASM). An ASM diskgroup was created comprised of the SVM metadevices described above. Oracle managed striping the tables, indexes, controlfiles, redo logs, undo tablespace, temporary tablespace and system tablespace across the ASM diskgroup.

#### Solaris Volume Manager information

```

d100      s  43GB c1t40d0s6
d101      s  43GB c2t40d0s6
d102      s  43GB c3t40d0s6
d103      s  43GB c4t40d0s6
d104      s  43GB c5t40d0s6
d105      s  43GB c6t40d0s6
d106      s  43GB c9t40d0s6
d107      s  43GB c10t40d0s6
d108      s  43GB c11t40d0s6
d109      s  43GB c12t40d0s6
d110      s  43GB c13t40d0s6
d111      s  43GB c14t40d0s6
d112      s  43GB c15t40d0s6
d113      s  43GB c16t40d0s6
d114      s  43GB c17t40d0s6
d115      s  43GB c18t40d0s6
d116      s  43GB c19t40d0s6
d117      s  43GB c20t40d0s6
d118      s  43GB c21t40d0s6
d119      s  43GB c22t40d0s6
d120      s  43GB c23t40d0s6
d121      s  43GB c24t40d0s6
d122      s  43GB c25t40d0s6
d123      s  43GB c26t40d0s6
d124      s  43GB c27t40d0s6
d125      s  43GB c28t40d0s6
--- d126-d207, d300-d407, d500-d607, d700-d807,
    d900-d1007, d1100-d1174
d1175     s  43GB c78t40d5s6
d1176     s  43GB c79t40d5s6
d1177     s  43GB c80t40d5s6
d1178     s  43GB c81t40d5s6
d1180     s  43GB c83t40d5s6
d1181     s  43GB c84t40d5s6
d1182     s  43GB c85t40d5s6
d1183     s  43GB c86t40d5s6
d1184     s  43GB c87t40d5s6
d1185     s  43GB c88t40d5s6
d1186     s  43GB c89t40d5s6
d1187     s  43GB c90t40d5s6
d1188     s  43GB c91t40d5s6
d1189     s  43GB c92t40d5s6
d1190     s  43GB c93t40d5s6
d1191     s  43GB c94t40d5s6
d1192     s  43GB c95t40d5s6
d1193     s  43GB c96t40d5s6
d1194     s  43GB c97t40d5s6
d1195     s  43GB c98t40d5s6
d1196     s  43GB c99t40d5s6
d1197     s  43GB c100t40d5s6
d1198     s  43GB c101t40d5s6
d1199     s  43GB c102t40d5s6
d1200     s  43GB c103t40d5s6
d1201     s  43GB c104t40d5s6
d1202     s  43GB c105t40d5s6
d1203     s  43GB c106t40d5s6
d1204     s  43GB c109t40d5s6
d1205     s  43GB c110t40d5s6
d1206     s  43GB c111t40d5s6
d1207     s  43GB c112t40d5s6

```

#### ASM instance information:

```

# initasm.ora
instance_type = asm
processes      = 1024
shared_pool_size = 2g
sort_area_size = 10485760
ASM_DISKSTRING = '/links/asm/d*'
ASM_DISKGROUPS = dg_tpch

# create dg_tpch diskgroup
CREATE DISKGROUP dg_tpch EXTERNAL REDUNDANCY
DISK
  '/links/asm/d100' size 44000m,
  '/links/asm/d101' size 44000m,

```

```

'/links/asm/d102' size 44000m,
'/links/asm/d103' size 44000m,
'/links/asm/d104' size 44000m,
'/links/asm/d105' size 44000m,
'/links/asm/d106' size 44000m,
'/links/asm/d107' size 44000m,
'/links/asm/d108' size 44000m,
'/links/asm/d109' size 44000m,
'/links/asm/d110' size 44000m,
'/links/asm/d111' size 44000m,
'/links/asm/d112' size 44000m,
'/links/asm/d113' size 44000m,
'/links/asm/d114' size 44000m,
'/links/asm/d115' size 44000m,
'/links/asm/d116' size 44000m,
'/links/asm/d117' size 44000m,
'/links/asm/d118' size 44000m,
'/links/asm/d119' size 44000m,
'/links/asm/d120' size 44000m,
'/links/asm/d121' size 44000m,
'/links/asm/d122' size 44000m,
'/links/asm/d123' size 44000m,
'/links/asm/d124' size 44000m,
'/links/asm/d125' size 44000m,
--- d126-d207, d300-d407, d500-d607, d700-d807,
    d900-d1007, d1100-d1174
'/links/asm/d1175' size 44000m,
'/links/asm/d1176' size 44000m,
'/links/asm/d1177' size 44000m,
'/links/asm/d1178' size 44000m,
'/links/asm/d1179' size 44000m,
'/links/asm/d1180' size 44000m,
'/links/asm/d1181' size 44000m,
'/links/asm/d1182' size 44000m,
'/links/asm/d1183' size 44000m,
'/links/asm/d1184' size 44000m,
'/links/asm/d1185' size 44000m,
'/links/asm/d1186' size 44000m,
'/links/asm/d1187' size 44000m,
'/links/asm/d1188' size 44000m,
'/links/asm/d1189' size 44000m,
'/links/asm/d1190' size 44000m,
'/links/asm/d1191' size 44000m,
'/links/asm/d1192' size 44000m,
'/links/asm/d1193' size 44000m,
'/links/asm/d1194' size 44000m,
'/links/asm/d1195' size 44000m,
'/links/asm/d1196' size 44000m,
'/links/asm/d1197' size 44000m,
'/links/asm/d1198' size 44000m,
'/links/asm/d1199' size 44000m,
'/links/asm/d1200' size 44000m,
'/links/asm/d1201' size 44000m,
'/links/asm/d1202' size 44000m,
'/links/asm/d1203' size 44000m,
'/links/asm/d1204' size 44000m,
'/links/asm/d1205' size 44000m,
'/links/asm/d1206' size 44000m,
'/links/asm/d1207' size 44000m
;
```

## Appendix C. Query Text and Query Output

### qual01

```

=====
-- TPC-H/TPC-R Pricing Summary Report Query (Q1)
-- Functional Query Definition
-- Approved February 1998

```

```

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 5.16 seconds.

```

### qual02

```

=====
-- TPC-H/TPC-R Minimum Cost Supplier Query (Q2)
-- Functional Query Definition
-- Approved February 1998

```

```

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
where
p_partkey = ps_partkey
and s_suppkey = ps_suppkey
and p_size = 15
and p_type like '%BRASS'
and s_nationkey = n_nationkey
and n_regionkey = r_regionkey
and r_name = 'EUROPE'
and ps_supplycost = (
select
min(ps_supplycost)
from
partsupp,
supplier,
nation,
region
where
p_partkey = ps_partkey
and s_suppkey = ps_suppkey
and s_nationkey = n_nationkey
and n_regionkey = r_regionkey
and r_name = 'EUROPE'
)
order by
s_acctbal desc,
n_name,
s_name,
p_partkey
)
where rownum <= 100

```

S_ACCTBAL	S_NAME	N_NAME	P_PARTKEY	P_MFGR	S_ADDRESS	S_PHONE	S_COMMENT
9938.53	Supplier#000005359	UNITED KINGDOM	185358.00	Manufacturer#4	QKuHYh,vZGiwu2FWEJoLDx04	33-429-790-6131	blithely silent pinto beans are furiously. slyly final deposits across
9937.84	Supplier#000005969	ROMANIA	108438.00	Manufacturer#1	ANDENSOSmk,miq23Xfb5Rwt6dvUcvt6Qa	29-520-692-3537	carefully slow deposits use furiously. slyly ironic platelets above the ironic
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 final instructi
29821.00	Manufacturer#4	GERMANY	43868.00	Manufacturer#5	y3OD9UyWSTok	17-779-299-1839	quickly express packages breach quiet pinto beans. requ
9871.22	Supplier#000006373	GERMANY	43868.00	Manufacturer#5	J8fcXWsTqM	17-813-485-8637	never silent deposits integrate furiously blit
9870.78	Supplier#000001286	GERMANY	81285.00	Manufacturer#2	YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH	17-516-924-4574	final theodolites cajole slyly special,
9870.78	Supplier#000001286	GERMANY	181285.00	Manufacturer#4	YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH	17-516-924-4574	

final theodolites cajole slyly special, 9852.52 RUSSIA 18972.00 t5L67YdBYH6o,Vz24jpDyQ9 7038	Supplier#000008973 Manufacturer#2 32-188-594-	kkYvL6IuvojJgTNG IKKaXQDYgx8ILohj 8014 17-627-663-
quickly regular instructions wake-- carefully unusual braids into the expres 9847.83 RUSSIA 130557.00 xMe97bpE69NzdWLoX 3593	Supplier#000008097 Manufacturer#2 32-375-640-	quickly regular requests are furiously. pending deposits wake 7937.93 Supplier#000009012 ROMANIA 83995.00 Manufacturer#2 iUiTziH,Ek3i4lwSgunXMgrcTzwdb 29-250-925- 9690 blithely bold ideas haggle quickly final, regular request 7914.45 Supplier#000001013 RUSSIA 125988.00 Manufacturer#2 riRcntps4KEDtYScjpMIWeYf6mNnR 32-194-698- 3365
slyly regular dependencies sleep slyly furiously express dep 9847.57 FRANCE 86344.00 VSt3rzk3qG698u6ld8HhOByvrTcSTsvQldQDag 16-886-766- 7945	Supplier#000006345 Manufacturer#1 16-886-766-	final, ironic theodolites alongside of the ironic 7912.91 Supplier#000004211 GERMANY 159180.00 Manufacturer#5 2wQRVovHrm3,v03IKzfTd,1PYsFXQFFOG 17-266-947- 7315
silent pinto beans should have to snooze carefully along the final reques 9847.57 FRANCE 173827.00 VSt3rzk3qG698u6ld8HhOByvrTcSTsvQldQDag 16-886-766- 7945	Supplier#000006345 Manufacturer#2 16-886-766-	final requests integrate slyly above the silent, even 7912.91 Supplier#000004211 GERMANY 184210.00 Manufacturer#4 2wQRVovHrm3,v03IKzfTd,1PYsFXQFFOG 17-266-947- 7315
silent pinto beans should have to snooze carefully along the final reques 9836.93 RUSSIA 4841.00 JOLK7C1,7xrEZSSow 5385	Supplier#000007342 Manufacturer#4 32-399-414-	final requests integrate slyly above the silent, even 7894.56 Supplier#000007981 GERMANY 85472.00 Manufacturer#4 NSJ96vMROAbeXP 17-963-404- 3760
final accounts haggle. bold accounts are furiously dugouts. furiously silent asymptotes are slyly 9817.10 RUSSIA 124815.00 4LfOHUZjggjEbAKw TgdKcgOc4D4uCYw 32-551-831- 1437	Supplier#000002352 Manufacturer#2 32-551-831-	regular, even theodolites integrate carefully. bold, special theodolites are slyly fluffily iron 7887.08 Supplier#000009792 GERMANY 164759.00 Manufacturer#3 Y28ITVeYriT3kIGdV2K8fSZ V2UqT5H1Otz 17-988-938- 4296
blithely pending packages across the ironic accounts grow slyly after the furiously 9817.10 RUSSIA 152351.00 4LfOHUZjggjEbAKw TgdKcgOc4D4uCYw 32-551-831- 1437	Supplier#000002352 Manufacturer#3 32-551-831-	pending, ironic packages sleep among the carefully ironic accounts. quickly final accounts 7871.50 Supplier#000007206 RUSSIA 104695.00 Manufacturer#1 3w fNCnrVmvJjE95sgWZzvW 32-432-452- 7731
blithely pending packages across the ironic accounts grow slyly after the furiously 9739.86 FRANCE 138357.00 o,Z3v4POifevE k9U1b 6JlucX,I 16-494-913- 5925	Supplier#000003384 Manufacturer#2 16-494-913-	furiously dogged pinto beans cajole. bold, express notornis until the slyly pending 7852.45 Supplier#000005864 RUSSIA 8363.00 Manufacturer#4 WCNfBPZeSxh3h,c 32-454-883- 3821
slyly ironic theodolites hag 9721.95 UNITED KINGDOM 156241.00 Atg6Gnm4dT2 2995	Supplier#000008757 Manufacturer#3 33-821-407- 2995	blithely regular deposits 7850.66 Supplier#000001518 UNITED KINGDOM 86501.00 Manufacturer#1 ONda3YJiHKJOC 33-730-383- 3892
ironic, even dolphins above the furiously ironic foxes sleep slyly around the caref -----delete lines ----- 8042.09 RUSSIA 150729.00 Dh8Ikg39onrbOL4DyTfGw8a9oKUX3d9Y 32-836-132- 8872	Supplier#000003245 Manufacturer#1 32-836-132-	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 regular instructions integrate blithely silent foxes. furiously express instructions hagg 7992.40 FRANCE 118574.00 8tBydnTDwUqfBfFV413 8937	Supplier#000006108 Manufacturer#1 16-974-998-	carefully bold accounts doub 100 rows processed. Query Processed in 0.44 seconds.
regular pinto beans are after 7980.65 FRANCE 13784.00 zE,7HgVPrCn 8247	Supplier#000001288 Manufacturer#4 16-646-464-	=====
unusual pinto beans cajole furiously according t 7950.37 GERMANY 33094.00 Manufacturer#5	Supplier#000008101 Manufacturer#5	=====

### qual03

```

-- TPC-H/TPC-R Shipping Priority Query (Q3)
-- Functional Query Definition
-- Approved February 1998

```

```

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
revenue desc,
o_orderdate)
where rownum <= 10

```

L_ORDERKEY	O_ORDERDATE	O_SHIPRIORITY	REVENUE
2456423.00	05 0.00		406181.01
3459808.00	04 0.00		405838.70
492164.00	19 0.00		390324.06
1188320.00	09 0.00		384537.94
2435712.00	26 0.00		378673.06
4878020.00	12 0.00		378376.80
5521732.00	13 0.00		375153.92
2628192.00	22 0.00		373133.31
993600.00	05 0.00		371407.46
2300070.00	13 0.00		367371.15

10 rows processed.  
Query Processed in 1.97 seconds.

### qual04

```

-- TPC-H/TPC-R Order Priority Checking Query (Q4)
-- Functional Query Definition
-- Approved February 1998

```

```

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

```

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

5 rows processed.  
Query Processed in 2.43 seconds.

### qual05

```

-- TPC-H/TPC-R Local Supplier Volume Query (Q5)
-- Functional Query Definition
-- Approved February 1998

```

```

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

```

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

5 rows processed.  
Query Processed in 2.73 seconds.

### qual06

```

-- TPC-H/TPC-R Forecasting Revenue Change Query (Q6)
-- Functional Query Definition
-- Approved February 1998

```

```

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

```

REVENUE
123141078.23

1 row processed.  
Query Processed in 0.38 seconds.

### qual07

```

-- TPC-H/TPC-R Volume Shipping Query (Q7)
-- Functional Query Definition

```



-- Approved February 1998

```

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( '1996-12-31', '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 1.81 seconds.

### qual08

```

-- TPC-H/TPC-R National Market Share Query (Q8)
-- Variant A
-- Approved February 1998

```

```

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
1995.00	0.03
1996.00	0.04

2 rows processed.  
Query Processed in 2.78 seconds.

### qual09

```

-- TPC-H/TPC-R Product Type Profit Measure Query (Q9)
-- Functional Query Definition
-- Approved February 1998

```

```

select
nation,
o_year,
sum(amount) as sum_profit
from
(
select
n_name as nation,
to_number (to_char (o_orderdate, 'yyyy')) as o_year,
l_extendedprice * (1 - l_discount) - ps_supplycost *
l_quantity as amount
from
part,
supplier,
lineitem,
partsupp,
orders,
nation
where
s_suppkey = l_suppkey
and ps_suppkey = l_suppkey
and ps_partkey = l_partkey
and p_partkey = l_partkey
and o_orderkey = l_orderkey
and s_nationkey = n_nationkey
and p_name like '%green%'
) profit
group by
nation,
o_year
order by
nation,
o_year desc

```

NATION	O_YEAR	SUM_PROFIT
ALGERIA	1998.00	31342867.23
ALGERIA	1997.00	57138193.02
ALGERIA	1996.00	56140140.13
ALGERIA	1995.00	53051469.65
ALGERIA	1994.00	

53867582.13		46551891.56	
ALGERIA	1993.00	ETHIOPIA	1992.00
54942718.13		44934648.64	
ALGERIA	1992.00	FRANCE	1998.00
54628034.71		32226407.84	
ARGENTINA	1998.00	FRANCE	1997.00
30211185.71		47121485.86	
ARGENTINA	1997.00	FRANCE	1996.00
50805741.75		47263135.50	
ARGENTINA	1996.00	FRANCE	1995.00
51923746.58		47275997.57	
ARGENTINA	1995.00	FRANCE	1994.00
49298625.77		47067209.33	
ARGENTINA	1994.00	----- delete lines -----	
50835610.11		PERU	1994.00
ARGENTINA	1993.00	50643531.80	
51646079.18		PERU	1993.00
ARGENTINA	1992.00	51584622.00	
50410314.99		PERU	1992.00
BRAZIL	1998.00	47523899.05	
27217924.38		ROMANIA	1998.00
BRAZIL	1997.00	30368667.40	
48378669.20		ROMANIA	1997.00
BRAZIL	1996.00	50365683.85	
50482870.36		ROMANIA	1996.00
BRAZIL	1995.00	49598999.01	
47623383.63		ROMANIA	1995.00
BRAZIL	1994.00	47537642.87	
47840165.73		ROMANIA	1994.00
BRAZIL	1993.00	51455283.01	
49054694.04		ROMANIA	1993.00
BRAZIL	1992.00	50407136.89	
48667639.08		ROMANIA	1992.00
CANADA	1998.00	48185385.13	
30379833.77		RUSSIA	1998.00
CANADA	1997.00	28322384.03	
50465052.31		RUSSIA	1997.00
CANADA	1996.00	50106685.18	
52560501.39		RUSSIA	1996.00
CANADA	1995.00	51753342.43	
52375332.81		RUSSIA	1995.00
CANADA	1994.00	49215820.36	
52600364.66		RUSSIA	1994.00
CANADA	1993.00	52205666.44	
52644504.07		RUSSIA	1993.00
CANADA	1992.00	51860230.03	
53932871.70		RUSSIA	1992.00
CHINA	1998.00	53251677.15	
31075466.16		SAUDI ARABIA	1998.00
CHINA	1997.00	31541259.81	
50551874.45		SAUDI ARABIA	1997.00
CHINA	1996.00	52438750.81	
51039293.88		SAUDI ARABIA	1996.00
CHINA	1995.00	52543737.82	
49287534.62		SAUDI ARABIA	1995.00
CHINA	1994.00	52938696.53	
50851090.07		SAUDI ARABIA	1994.00
CHINA	1993.00	51389601.97	
54229629.83		SAUDI ARABIA	1993.00
CHINA	1992.00	52937508.88	
52400529.37		SAUDI ARABIA	1992.00
EGYPT	1998.00	54843459.64	
29054433.39		UNITED KINGDOM	1998.00
EGYPT	1997.00	28494874.00	
50627611.45		UNITED KINGDOM	1997.00
EGYPT	1996.00	49381810.90	
49542212.84		UNITED KINGDOM	1996.00
EGYPT	1995.00	51386853.96	
48311550.32		UNITED KINGDOM	1995.00
EGYPT	1994.00	51509586.79	
49790644.74		UNITED KINGDOM	1994.00
EGYPT	1993.00	48086499.71	
48904292.97		UNITED KINGDOM	1993.00
EGYPT	1992.00	49166827.22	
49434932.62		UNITED KINGDOM	1992.00
ETHIOPIA	1998.00	49349122.08	
28040717.27		UNITED STATES	1998.00
ETHIOPIA	1997.00	25126238.95	
47455009.87		UNITED STATES	1997.00
ETHIOPIA	1996.00	50077306.42	
46491097.57		UNITED STATES	1996.00
ETHIOPIA	1995.00	48048649.47	
46804449.30		UNITED STATES	1995.00
ETHIOPIA	1994.00	48809032.42	
48516143.92		UNITED STATES	1994.00
ETHIOPIA	1993.00	49296747.18	



110246.00	Customer#000110246	33354.00	14408297.40
566842.98		154747.00	14407580.68
7763.35	VIETNAM	82865.00	14235489.78
7KzflgX MDOq7sOkI	31-943-426-	76094.00	14094247.04
9837		222.00	13937777.74
dolphins sleep blithely among the slyly final		121271.00	13908336.00
142549.00	Customer#000142549	55221.00	13716120.47
563537.24		22819.00	13666434.28
5085.99	INDONESIA	76281.00	13646853.68
ChqEoK43OysjdHbtKCp6dKqjNyvvi9	19-955-562-	85298.00	13581154.93
2398		85158.00	13554904.00
regular, unusual dependencies boost slyly; ironic		139684.00	13535538.72
attainments nag fluffily into the unusual packages?		31034.00	13498025.25
146149.00	Customer#000146149	87305.00	13482847.04
557254.99		10181.00	13445148.75
1791.55	ROMANIA	62323.00	13411824.30
s87fvzFQpU	29-744-164-	26489.00	13377256.38
6487		96493.00	13339057.83
silent, unusual requests detect quickly slyly regul		56548.00	13329014.97
52528.00	Customer#000052528	55576.00	13306843.35
556397.35		159751.00	13306614.48
551.79	ARGENTINA	92406.00	13287414.50
NFztyTOR10UOJ	11-208-192-	182636.00	13223726.74
3205		199969.00	13135288.21
unusual requests detect. slyly dogged theodolites use		62865.00	13001926.94
slyly. deposit		7284.00	12945298.19
23431.00	Customer#000023431	197867.00	12944510.52
554269.54		11562.00	12931575.51
3381.86	ROMANIA	75165.00	12916918.12
HgiV0phqhaIa9aydNoIlb	29-915-458-	97175.00	12911283.50
2654		140840.00	12896562.23
instructions nag quickly. furiously bold accounts		65241.00	12890600.46
cajol		166120.00	12876927.22
		9035.00	12863828.70
		144616.00	12853549.30
20 rows processed.		176723.00	12832309.74
Query Processed in 2.87 seconds.		170884.00	12792136.58
		29790.00	12723300.33
=====		95213.00	12555483.73
<b>qual11</b>		183873.00	12550533.05
=====		171235.00	12476538.30
-- TPC-H/TPC-R Important Stock Identification Query		21533.00	12437821.32
(Q11)		17290.00	12432159.50
-- Functional Query Definition		156397.00	12260623.50
-- Approved February 1998		122611.00	12222812.98
		139155.00	12220319.25
		146316.00	12215800.61
select		171381.00	12199734.52
ps_partkey,		198633.00	12078226.95
sum(ps_supplycost * ps_availqty) as value		167417.00	12046637.62
from		59512.00	12043468.76
partsupp,			
supplier,			
nation		98643.00	7968603.73
where		153787.00	7967535.58
ps_suppkey = s_suppkey		8932.00	7967222.19
and s_nationkey = n_nationkey		20134.00	7965713.28
and n_name = 'GERMANY'		197635.00	7963507.58
group by		80408.00	7963312.17
ps_partkey having		37728.00	7961875.68
sum(ps_supplycost * ps_availqty) > (		26624.00	7961772.31
select		44736.00	7961144.10
sum(ps_supplycost * ps_availqty) * 0.0001000000		29763.00	7960605.03
from		36147.00	7959463.68
partsupp,		146040.00	7957587.66
supplier,		115469.00	7957485.14
nation		142276.00	7956790.63
where		181280.00	7954037.35
ps_suppkey = s_suppkey		115096.00	7953047.55
and s_nationkey = n_nationkey		109650.00	7952258.73
and n_name = 'GERMANY'		93862.00	7951992.24
)		158325.00	7950728.30
order by		55952.00	7950387.06
value desc		122397.00	7947106.27
		28114.00	7946945.72
		11966.00	7945197.48
PS_PARTKEY	VALUE	47814.00	7944083.00
129760.00	17538456.86	85096.00	7943691.06
166726.00	16503353.92	51657.00	7943593.77
191287.00	16474801.97	196680.00	7943578.89
161758.00	16101755.54	13141.00	7942730.34
34452.00	15983844.72	193327.00	7941036.25
139035.00	15907078.34	152612.00	7940663.71
9403.00	15451755.62	139680.00	7939242.36
154358.00	15212937.88	31134.00	7938318.30
38823.00	15064802.86	45636.00	7937240.85
85606.00	15053957.15	56694.00	7936015.95

```

8114.00      7933921.88
71518.00    7932261.69
72922.00    7930400.64
146699.00   7929167.40
92387.00    7928972.67
186289.00   7928786.19
95952.00    7927972.78
196514.00   7927180.70
4403.00     7925729.04
2267.00     7925649.37
45924.00    7925047.68
11493.00    7916722.23
104478.00   7916253.60
166794.00   7913842.00
161995.00   7910874.27
23538.00    7909752.06
41093.00    7909579.92
112073.00   7908617.57
92814.00    7908262.50
88919.00    7907992.50
79753.00    7907933.88
108765.00   7905338.98
146530.00   7905336.60
71475.00    7903367.58
36289.00    7901946.50
61739.00    7900794.00
52338.00    7898638.08
194299.00   7898421.24
105235.00   7897829.94
77207.00    7897752.72
96712.00    7897575.27
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
101674.00   7879324.60
51968.00    7879102.21
72073.00    7877736.11
5182.00     7874521.73

```

1048 rows processed.  
Query Processed in 2.78 seconds.

### qual12

```

-- TPC-H/TPC-R Shipping Modes and Order Priority Query (Q12)
-- Functional Query Definition
-- Approved February 1998

```

```

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 1.98 seconds.

### qual13

```

-- TPC-H/TPC-R Customer Distribution Query (Q13)
-- Functional Query Definition
-- Approved February 1998

```

```

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
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 1.76 seconds.

=====  
**qual14**  
=====

-- TPC-H/TPC-R Promotion Effect Query (Q14)  
-- Functional Query Definition  
-- Approved February 1998

```
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 0.12 seconds.

=====  
**qual15**  
=====

-- TPC-H/TPC-R Top Supplier Query (Q15)  
-- Functional Query Definition  
-- Approved February 1998

```
with revenue as (
select
l_suppkey supplier_no,
sum(l_extendedprice * (1-l_discount)) total_revenue
from
lineitem
where
l_shipdate >= to_date ('1996-01-01', 'YYYY-MM-DD')
and l_shipdate < add_months(to_date ('1996-01-01', 'YYYY-MM-DD'), 3)
group by
l_suppkey
)
select
s_suppkey,
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 1.61 seconds.

=====  
**qual16**  
=====

-- TPC-H/TPC-R Parts/Supplier Relationship Query (Q16)  
-- Functional Query Definition  
-- Approved February 1998

```
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
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#22	SMALL BURNISHED BRASS	19.00
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		
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	Brand#22	PROMO ANODIZED TIN	49.00
24.00			20.00		
Brand#43	PROMO BRUSHED TIN	3.00	Brand#22	PROMO POLISHED BRASS	45.00
24.00			20.00		
Brand#43	SMALL ANODIZED COPPER	36.00	Brand#22	SMALL BURNISHED STEEL	45.00
24.00			20.00		
Brand#44	STANDARD POLISHED NICKEL	3.00	Brand#23	MEDIUM ANODIZED STEEL	45.00
24.00			20.00		
Brand#52	ECONOMY PLATED TIN	14.00	Brand#23	PROMO POLISHED STEEL	23.00
24.00			20.00		
Brand#52	STANDARD BURNISHED NICKEL	3.00	Brand#23	STANDARD BRUSHED TIN	14.00
24.00			20.00		
Brand#53	MEDIUM ANODIZED STEEL	14.00	Brand#23	STANDARD PLATED NICKEL	36.00
24.00			20.00		
Brand#14	PROMO ANODIZED NICKEL	45.00	Brand#24	PROMO PLATED COPPER	49.00
23.00			20.00		
Brand#32	ECONOMY PLATED BRASS	9.00	Brand#24	PROMO PLATED STEEL	49.00
23.00			20.00		
Brand#52	SMALL ANODIZED COPPER	3.00	Brand#24	PROMO POLISHED STEEL	9.00
23.00			20.00		
Brand#11	ECONOMY BRUSHED COPPER	45.00	Brand#24	STANDARD BRUSHED TIN	36.00
20.00			20.00		
Brand#11	ECONOMY PLATED BRASS	23.00	Brand#25	LARGE ANODIZED BRASS	3.00
20.00			20.00		
Brand#11	LARGE BRUSHED COPPER	49.00	Brand#25	PROMO BURNISHED TIN	3.00
20.00			20.00		
Brand#11	LARGE POLISHED COPPER	49.00	Brand#31	ECONOMY POLISHED NICKEL	3.00
20.00			20.00		
Brand#12	STANDARD ANODIZED TIN	49.00	Brand#31	MEDIUM PLATED TIN	45.00
20.00			20.00		
Brand#12	STANDARD PLATED BRASS	19.00	Brand#31	SMALL ANODIZED STEEL	14.00
20.00			20.00		
Brand#13	ECONOMY BRUSHED BRASS	9.00	Brand#32	ECONOMY ANODIZED COPPER	36.00
20.00			20.00		
Brand#13	ECONOMY BURNISHED STEEL	14.00	Brand#32	ECONOMY BRUSHED NICKEL	49.00
20.00			20.00		
Brand#13	LARGE BURNISHED NICKEL	19.00	Brand#32	LARGE ANODIZED TIN	19.00
20.00			20.00		
Brand#13	MEDIUM BURNISHED COPPER	36.00	Brand#32	MEDIUM BURNISHED COPPER	19.00
20.00			20.00		
Brand#13	SMALL BRUSHED TIN	45.00	Brand#32	SMALL ANODIZED STEEL	45.00
20.00			20.00		
Brand#13	STANDARD ANODIZED COPPER	3.00	Brand#33	ECONOMY POLISHED COPPER	19.00
20.00			20.00		
Brand#13	STANDARD PLATED NICKEL	23.00	Brand#33	PROMO PLATED NICKEL	14.00
20.00			20.00		
Brand#14	ECONOMY ANODIZED COPPER	14.00	Brand#33	SMALL POLISHED TIN	9.00
20.00			20.00		
Brand#14	ECONOMY PLATED TIN	36.00	Brand#33	STANDARD ANODIZED BRASS	49.00
20.00			20.00		
Brand#14	ECONOMY POLISHED NICKEL	3.00	Brand#33	STANDARD BURNISHED BRASS	45.00
20.00			20.00		
Brand#14	MEDIUM ANODIZED NICKEL	3.00	Brand#34	ECONOMY BRUSHED NICKEL	49.00
20.00			20.00		
Brand#14	SMALL POLISHED TIN	14.00	Brand#34	LARGE BRUSHED BRASS	19.00
20.00			20.00		
Brand#15	MEDIUM ANODIZED COPPER	9.00	Brand#34	SMALL BRUSHED TIN	3.00
20.00			20.00		
Brand#15	MEDIUM PLATED TIN	23.00	Brand#34	STANDARD PLATED COPPER	9.00
20.00			20.00		
Brand#15	PROMO PLATED BRASS	14.00	Brand#35	LARGE ANODIZED NICKEL	3.00
20.00			20.00		
Brand#15	SMALL ANODIZED COPPER	45.00	Brand#35	MEDIUM ANODIZED BRASS	45.00
20.00			20.00		
Brand#15	SMALL PLATED COPPER	49.00	Brand#35	MEDIUM ANODIZED STEEL	23.00
20.00			20.00		
Brand#15	STANDARD PLATED TIN	3.00	Brand#35	PROMO ANODIZED COPPER	49.00
20.00			20.00		
Brand#21	LARGE ANODIZED COPPER	36.00	Brand#35	SMALL POLISHED COPPER	14.00
20.00			20.00		
Brand#21	LARGE BRUSHED TIN	3.00	Brand#41	LARGE ANODIZED STEEL	3.00
20.00			20.00		
Brand#21	MEDIUM ANODIZED COPPER	14.00	Brand#41	LARGE BRUSHED NICKEL	23.00
20.00			20.00		
Brand#21	PROMO BRUSHED TIN	36.00	Brand#41	LARGE BURNISHED COPPER	3.00
20.00			20.00		
Brand#21	PROMO POLISHED NICKEL	45.00		----- delete lines -----	
20.00			Brand#55	STANDARD ANODIZED BRASS	23.00
Brand#21	SMALL ANODIZED COPPER	9.00	4.00		
20.00			Brand#55	STANDARD ANODIZED BRASS	49.00
Brand#21	SMALL POLISHED NICKEL	23.00	4.00		
20.00			Brand#55	STANDARD ANODIZED COPPER	9.00
Brand#22	LARGE ANODIZED COPPER	36.00	4.00		
20.00			Brand#55	STANDARD ANODIZED COPPER	14.00
Brand#22	LARGE BRUSHED COPPER	49.00	4.00		
20.00			Brand#55	STANDARD ANODIZED COPPER	45.00

4.00				4.00			
Brand#55	STANDARD	ANODIZED NICKEL	3.00	Brand#55	STANDARD	PLATED NICKEL	45.00
4.00				4.00			
Brand#55	STANDARD	ANODIZED NICKEL	14.00	Brand#55	STANDARD	PLATED STEEL	14.00
4.00				4.00			
Brand#55	STANDARD	ANODIZED NICKEL	45.00	Brand#55	STANDARD	PLATED STEEL	23.00
4.00				4.00			
Brand#55	STANDARD	ANODIZED NICKEL	49.00	Brand#55	STANDARD	PLATED STEEL	49.00
4.00				4.00			
Brand#55	STANDARD	ANODIZED STEEL	3.00	Brand#55	STANDARD	PLATED TIN	9.00
4.00				4.00			
Brand#55	STANDARD	ANODIZED STEEL	14.00	Brand#55	STANDARD	PLATED TIN	14.00
4.00				4.00			
Brand#55	STANDARD	ANODIZED TIN	14.00	Brand#55	STANDARD	PLATED TIN	36.00
4.00				4.00			
Brand#55	STANDARD	ANODIZED TIN	36.00	Brand#55	STANDARD	POLISHED BRASS	3.00
4.00				4.00			
Brand#55	STANDARD	ANODIZED TIN	45.00	Brand#55	STANDARD	POLISHED BRASS	9.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED BRASS	9.00	Brand#55	STANDARD	POLISHED BRASS	23.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED BRASS	19.00	Brand#55	STANDARD	POLISHED COPPER	3.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED COPPER	14.00	Brand#55	STANDARD	POLISHED COPPER	23.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED COPPER	19.00	Brand#55	STANDARD	POLISHED COPPER	45.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED NICKEL	3.00	Brand#55	STANDARD	POLISHED NICKEL	3.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED NICKEL	36.00	Brand#55	STANDARD	POLISHED NICKEL	23.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED STEEL	9.00	Brand#55	STANDARD	POLISHED NICKEL	36.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED STEEL	14.00	Brand#55	STANDARD	POLISHED NICKEL	45.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED STEEL	19.00	Brand#55	STANDARD	POLISHED NICKEL	49.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED STEEL	49.00	Brand#55	STANDARD	POLISHED STEEL	14.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED TIN	19.00	Brand#55	STANDARD	POLISHED STEEL	23.00
4.00				4.00			
Brand#55	STANDARD	BRUSHED TIN	49.00	Brand#55	STANDARD	POLISHED TIN	9.00
4.00				4.00			
Brand#55	STANDARD	BURNISHED BRASS	9.00	Brand#55	STANDARD	POLISHED TIN	19.00
4.00				4.00			
Brand#55	STANDARD	BURNISHED BRASS	19.00	Brand#55	STANDARD	POLISHED TIN	36.00
4.00				4.00			
Brand#55	STANDARD	BURNISHED BRASS	23.00	Brand#11	SMALL	BRUSHED TIN	19.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED BRASS	36.00	Brand#15	LARGE	PLATED NICKEL	45.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED COPPER	3.00	Brand#15	LARGE	POLISHED NICKEL	9.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED NICKEL	9.00	Brand#21	PROMO	BURNISHED STEEL	45.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED NICKEL	49.00	Brand#22	STANDARD	PLATED STEEL	23.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED STEEL	19.00	Brand#25	LARGE	PLATED STEEL	19.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED STEEL	23.00	Brand#32	STANDARD	ANODIZED COPPER	23.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED STEEL	36.00	Brand#33	SMALL	ANODIZED BRASS	9.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED STEEL	45.00	Brand#35	MEDIUM	ANODIZED TIN	19.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED TIN	9.00	Brand#51	SMALL	PLATED BRASS	23.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED TIN	19.00	Brand#52	MEDIUM	BRUSHED BRASS	45.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED TIN	36.00	Brand#53	MEDIUM	BRUSHED TIN	45.00
4.00				3.00			
Brand#55	STANDARD	BURNISHED TIN	49.00	Brand#54	ECONOMY	POLISHED BRASS	9.00
4.00				3.00			
Brand#55	STANDARD	PLATED BRASS	9.00	Brand#55	PROMO	PLATED BRASS	19.00
4.00				3.00			
Brand#55	STANDARD	PLATED BRASS	45.00	Brand#55	STANDARD	PLATED TIN	49.00
4.00				3.00			
Brand#55	STANDARD	PLATED BRASS	49.00				
4.00							
Brand#55	STANDARD	PLATED COPPER	9.00				
4.00							
Brand#55	STANDARD	PLATED COPPER	45.00				
4.00							
Brand#55	STANDARD	PLATED NICKEL	3.00				
4.00							
Brand#55	STANDARD	PLATED NICKEL	19.00				

18314 rows processed.  
Query Processed in 0.69 seconds.

=====  
**qual17**  
=====



-- TPC-H/TPC-R Small-Quantity-Order Revenue Query (Q17)  
 -- Functional Query Definition  
 -- Approved February 1998

```
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
)
```

AVG\_YEARLY  
 348406.05

1 row processed.  
 Query Processed in 2.55 seconds.

=====  
**qual18**  
 =====

-- TPC-H/TPC-R Large Volume Customer Query (Q18)  
 -- Function Query Definition  
 -- Approved February 1998

```
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
)
where rownum <= 100
```

C_NAME	C_CUSTKEY	O_ORDERKEY	O_ORDERDATE	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
Customer#000046435	46435.00	4745607.00	1997-07-03	508047.99	309.00
Customer#000015272	15272.00	3883783.00	1993-07-28	500241.33	302.00
Customer#000146608	146608.00	3342468.00	1994-06-12	499794.58	303.00
Customer#000096103	96103.00	5984582.00	1992-03-16	494398.79	312.00
Customer#000024341	24341.00	1474818.00	1992-11-15	491348.26	302.00
Customer#000137446	137446.00	5489475.00	1997-05-23	487763.25	311.00
Customer#000107590	107590.00	4267751.00	1994-11-04	485141.38	301.00
Customer#000050008	50008.00	2366755.00	1996-12-09	483891.26	302.00
Customer#000015619	15619.00	3767271.00	1996-08-07	480083.96	318.00
Customer#000077260	77260.00	1436544.00	1992-09-12	479499.43	307.00
Customer#000109379	109379.00	5746311.00	1996-10-10	478064.11	302.00
Customer#000054602	54602.00	5832321.00	1997-02-09	471220.08	307.00
Customer#000105995	105995.00	2096705.00	1994-07-03	469692.58	307.00
Customer#000148885	148885.00	2942469.00	1992-05-31	469630.44	313.00
Customer#000114586	114586.00	551136.00	1993-05-19	469605.59	308.00
Customer#000105260	105260.00	5296167.00	1996-09-06	469360.57	303.00
Customer#000147197	147197.00	1263015.00	1997-02-02	467149.67	320.00
Customer#000064483	64483.00	2745894.00	1996-07-04	466991.35	304.00
Customer#000136573	136573.00	2761378.00	1996-05-31	461282.73	301.00
Customer#000016384	16384.00	502886.00	1994-04-12	458378.92	312.00
Customer#000117919	117919.00	2869152.00	1996-06-20	456815.92	317.00
Customer#000012251	12251.00	735366.00	1993-11-24	455107.26	309.00
Customer#000120098	120098.00	1971680.00	1995-06-14	453451.23	308.00
Customer#000066098	66098.00	5007490.00	1992-08-07	453436.16	304.00
Customer#000117076	117076.00	4290656.00	1997-02-05	449545.85	301.00
Customer#000129379	129379.00	4720454.00	1997-06-07	448665.79	303.00
Customer#000126865	126865.00	4702759.00	1994-11-07		

```

447606.65      320.00
Customer#000088876      88876.00
983201.00      1993-12-30
446717.46      304.00
Customer#000036619      36619.00
4806726.00      1995-01-17
446704.09      328.00
Customer#000141823      141823.00
2806245.00      1996-12-29
446269.12      310.00
Customer#000053029      53029.00
2662214.00      1993-08-13
446144.49      302.00
Customer#000018188      18188.00
3037414.00      1995-01-25
443807.22      308.00
Customer#000066533      66533.00
29158.00      1995-10-21
443576.50      305.00
Customer#000037729      37729.00
4134341.00      1995-06-29
441082.97      309.00
Customer#000003566      3566.00
2329187.00      1998-01-04
439803.36      304.00
Customer#000045538      45538.00
4527553.00      1994-05-22
436275.31      305.00
Customer#000081581      81581.00
4739650.00      1995-11-04
435405.90      305.00
Customer#000119989      119989.00
1544643.00      1997-09-20
434568.25      320.00
Customer#000003680      3680.00
3861123.00      1998-07-03
433525.97      301.00
Customer#000113131      113131.00
967334.00      1995-12-15
432957.75      301.00
Customer#000141098      141098.00
565574.00      1995-09-24
430986.69      301.00
Customer#000093392      93392.00
5200102.00      1997-01-22
425487.51      304.00
Customer#000015631      15631.00
1845057.00      1994-05-12
419879.59      302.00
Customer#000112987      112987.00
4439686.00      1996-09-17
418161.49      305.00
Customer#000012599      12599.00
4259524.00      1998-02-12
415200.61      304.00
Customer#000105410      105410.00
4478371.00      1996-03-05
412754.51      302.00
Customer#000149842      149842.00
5156581.00      1994-05-30
411329.35      302.00
Customer#000010129      10129.00
5849444.00      1994-03-21
409129.85      309.00
Customer#000069904      69904.00
1742403.00      1996-10-19
408513.00      305.00
Customer#000017746      17746.00
6882.00      1997-04-09
408446.93      303.00
Customer#000013072      13072.00
1481925.00      1998-03-15
399195.47      301.00
Customer#000082441      82441.00
857959.00      1994-02-07
382579.74      305.00
Customer#000088703      88703.00
2995076.00      1994-01-30
363812.12      302.00

```

```

57 rows processed.
Query Processed in 3.06 seconds.

```

```
=====
```

## qual19

```

=====
-- TPC-H/TPC-R Discounted Revenue Query (Q19)
-- Functional Query Definition
-- Approved February 1998

```

```

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'
)
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 2.43 seconds.

```

```
=====
```

## qual20

```

=====
-- TPC-H/TPC-R Potential Part Promotion Query (Q20)
-- Function Query Definition
-- Approved February 1998

```

```

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 > (
select
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,RmTymrZVYaFZva2SH,j
Supplier#000000091   YV45D7TkfdQanOOZ7q9QxkyGUapUloOWU6q3
Supplier#000000197   YC2Acon6kjY3zj3Fbxs2k4Vdf7X0cd2F
Supplier#000000226   83qOdU2EYRdPQAQhEtn GRZED
Supplier#000000285   Br7elnntlyxrw6ImpgJ7YdhFDjuBf
Supplier#000000378   FfbhyCxWvcPr08ltp9
Supplier#000000402   i9Sw4DoyMhzhKXCH9By,AYSgmD
Supplier#000000530   OcmLyfRXlagA8ukENJv,
Supplier#000000688   D
fw5ocppmZpYBBIPI718hCihLDZ5KhKX
Supplier#000000710   fl19YPvOyb
QoYwjKC,oPypcGfieBAcwKJo
Supplier#000000736   l612nMwVuovfKnuVgaSGK2rDy65DlAFLegiL7
Supplier#000000761   zLSLeLQUj2XrvTTFnv7WAcYZGvMTx882d4
Supplier#000000884   bmhEShejaS
Supplier#000000887   urEaTejH5POADP2ARrf
Supplier#000000935   ij98czM
2KzWe7dTOxB8sq0UfCdvrX
Supplier#000000975   ,ac e,tBpNwKb5xMUzeohxlRn,
hdZJo73gFQF8y
Supplier#000001263   rQWrf6nf8ZhB2TAiIDivo5Io
Supplier#000001399   LmrocnIMSyYOWuANx7
Supplier#000001446   lch9HMNUlR7a0LlybsUodVknk6
Supplier#000001454   TOPimgu2TVXI jhiL93h,
Supplier#000001500   wDmF5xLxtQch9ctVu,
Supplier#000001602   uKNWleafam644
Supplier#000001626   UhxNRzUuldtFmp0
Supplier#000001682   pXTkGxrTQVYHlRr
Supplier#000001699   Q9C4rfJ26oiJVPqqcVXeRI
Supplier#000001700   7hmlCoflY5zLFg
Supplier#000001726   TeRY7tTH24sEword7yAaSkjx8
Supplier#000001730   Rc8e,1Pybn r6zo0VJIEiD0UD
vhk
Supplier#000001746   qWsendlOekQGLaW4uq06uQaCm5lse8lirv7 hBRd
Supplier#000001752   Fra7outx41THYJaRThdOGiBk
Supplier#000001856   jXcRgzYF0ah05iR8p6w5SbJLcUGyYiURPvFwUWM
Supplier#000001931   FpJbMU2h6ZR2eBv8I9NIxF
Supplier#000001939   NrK,JA4bfReUs
Supplier#000001990   DSDJkCgBJzuPglyuM,CuDlnsRliOxkkHezTCA
Supplier#000002020   jB6rld7MxP6co
Supplier#000002022   dwebGX7Id2pc25YvY33
Supplier#000002036   20ytTtVobjKUUI2WCB0A
Supplier#000002204   uYmlr46C06udCqanj0KiRsoTQakZsEyssL
Supplier#000002243   nSOEV3JeOU79
Supplier#000002245   hz2qWXWVjOyKhqPYMcEwz6zFkrTaDM
Supplier#000002282   ES2lK9dxoWl1lTzWCj7ekdlNwSwnv1Z 6mQ,BKn
Supplier#000002303   nCovfpB6YOymbgOht7lftklpkHl
Supplier#000002373   RzHsxOTQmElcJxIBiVA52Z
JB58rJhPrYlR
Supplier#000002419   qydBQd14I515mVXA4fYY
Supplier#000002481   nLKHUOn2M19TOA06Znq9GEMcIlMO2
Supplier#000002571   JZUgz04c iJFLrLrGsz90 N,W
lrVHNIReyq

```

```

Supplier#000002585   CsPoKpw2QuTY4AV1NkWuttneIa4SN
Supplier#000002630   ZIQAvjNUY9KH5ive zm7k
VlPiDl7CCo2l
Supplier#000002719   4nnzQI2CbqREQUuIsXTBVUkaP4mNS3
Supplier#000002721   HVdFAN2JHMqSpKm
Supplier#000002730   lIFxR4fzm3lC6,muzJw184z
Supplier#000002775   yDclaDaBD4ihH
Supplier#000002853   rTNAOIitXka
Supplier#000002875   6JgMi
9Qt6VmwL3Ltt1SRlKww0keLQ,RAZA
Supplier#000002934   m,trBENywsARwg3DhB
Supplier#000002941   Naddba 8YTEKekZyP0
Supplier#000002960   KCPCEsRGGo6vx8TygHh60nAyf9rStQT2T
Supplier#000002980   B9k9yVsyaXvWktOSHezqHiAEp9id0SKzkw
Supplier#000003062   LSQNgqYlXnOzz9zBCapy7HwOZQ
Supplier#000003087   ANwe8QsZ4rgjIHSqVz991eWQ
Supplier#000003089   s5b VCIZqMSZVA r
g7LTdcg29GbTE7rIlx
----- delete lines -----
Supplier#000006949   mLxYUJhsGcLtKe ,GFirNul83AvT
Supplier#000006985   PrUUIboQpy,OtgJ01Z4BxJQUYrwr9c3I
Supplier#000007072   2tRyX9M1a
4Rcm57s779F1ANG9jlpK
Supplier#000007098   G3j8g0KC4OcbAu2OVOPHrXQWMCUDjq8wgCHOExu
Supplier#000007135   ls DoKv7V5ulFqy9V
Supplier#000007160   TqDGBULB3cTqIT6FKDvm9BS4e4v,zwYiQPb
Supplier#000007169   tEc95D2moN9S84nd550,dlnW
Supplier#000007322   wr7dgte5q
MAjiY0uwm13MyDkSMX1
Supplier#000007365   51xhROLvQMj05DndtZWt
Supplier#000007398   V8eE6oZ00OFNU,
Supplier#000007402   4UVv58erylrmqSR5
Supplier#000007448   yhhpWiJi7EJ6Q5VCAQ
Supplier#000007477   9m9j0wfhwZCvVHxkU,PPaxwSH0h
Supplier#000007509   q8,V6LJRohJjHcOusG7aLTMg
Supplier#000007561   rMcFg2530VC
Supplier#000007789   rQ7cUcPrtudOy03svNSkimqH6qrfWT2Sz
Supplier#000007801   69fi,U1r6enUb
Supplier#000007818   yhhc2CQec Jrvc8zqBi83
Supplier#000007885   u3sicchh5ZpyTUpNlcJKNCaobIWgY
Supplier#000007918   r,v9mBQ6LoEYyjl
Supplier#000007926   ErzCF80K9Uy
Supplier#000007957   ELwnio14ssoUl dRyZIL OK3Vtzb
Supplier#000007965   F7Un5lJ7p5hhj
Supplier#000007968   DsF9UlZ2Fo6HXN9aErvyglikHoD582HSGZpP
Supplier#000007998   LnASFBfYRFOo9d6d,asBvVq9Lo2P
Supplier#000008168   aOa82a8ZbkCnfDLX
Supplier#000008231   IK7eGw Yj90sTdpS,vcqWxLB
Supplier#000008243   2AyePMkDqmqzVzjGTixZthFlO8h
EiudCMxOmIIG
Supplier#000008275   BlbnDFwg,gpXKQLLN
Supplier#000008323   75l18sZmASwm
POeherMdj9tmpyeQ,BfCXN5BIAb
Supplier#000008366   h778cejl4BuW9OEKlvPTWq4iwASR6EBBXN7zeS8
Supplier#000008423   RQhKnkAhRODar3Lx4Q1wEMn00hNe Kq
Supplier#000008480   4sSDA4ACReklNjEm5T6b
Supplier#000008532   Uc29q4,5xVdDOP87UZrxhr4xWS0ihEUxuh
Supplier#000008595   MH0iE7XQ3z UW30 DbCbqmc
Supplier#000008610   SgVgP90vP452sUNTgzL9zKwXHXAZv6tV
Supplier#000008705   aE,trRNdPx,4yinTD903DebDIp
Supplier#000008742   HmlPQEzKCPECtUL14,kKq
Supplier#000008841   I 85Lulsekbg2xrSIzm0
Supplier#000008895   2cH4okfaLSZTTg8SKRbbJQxkmeFu2Esj
Supplier#000008967   2kwEHyMG
7FwozNImAUE6mH0hYtqYculJM
Supplier#000008972   w2vF6 D5YZO3visPXsqVfLADTK
Supplier#000009032   qK,trB6Sdy4Dz1BRUFNy
Supplier#000009147   rOAuryHxpZ9eOvx
Supplier#000009252   F7czZaPUHwhl ZKyj3xmAVWC1XdP
uelp5m,i
Supplier#000009278   RqYtZgzj93CLX OmcyfCENOfd

```

```

Supplier#000009327      uoqMdf7e7Gj9dbQ53
Supplier#000009430      igRqmneFt
Supplier#000009567      r4Wfx4c3xsEAjcgJ71HHZByornl
D9vrztXlv4
Supplier#000009601      51m637b0, Rw5DnHWFUvLacRx9
Supplier#000009709      rRnCbHYgdgl9PZYnyWKVYSUW0vKg
Supplier#000009753      wLhVEcRmd7PkJF4FBnGK7Z
Supplier#000009796      z, y4Idmr15DOvPUqYG
Supplier#000009799      4wNjXGa4OKWl
Supplier#000009811      E3iuyq7UnZxU7oPZIE2Gu6
Supplier#000009812
APFRMy3lCbgFga53n5t9DxzFPQPgnjrGt32
Supplier#000009862      rJzweWeN58
Supplier#000009868      ROjGgX5gvtkmmUUoeyy7v
Supplier#000009869
ucLqxzrpBTRMewGSM29t0rNTM30g1Tu3Xgg3mKag
Supplier#000009899      7XdpAhrzrlt, UQFZE
Supplier#000009974
7wJ, J5DKcxSU4Kp1cQLpbcAvB5AsvKt

```

204 rows processed.  
Query Processed in 1.18 seconds.

## qual21

```

-----
-- TPC-H/TPC-R Suppliers Who Kept Orders Waiting Query
(Q21)
-- Functional Query Definition
-- Approved February 1998

```

```

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)
where rownum <= 100

```

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#00000486	15.00
Supplier#00000565	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#00000633	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
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
```

```
100 rows processed.
Query Processed in 9.80 seconds.
```

```
=====
qual22
=====
```

```
-- TPC-D Global Sales Opportunity Query (Q22)
-- Functional Query Definition
-- Approved February 1998
```

```
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
```

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

```
7 rows processed.
Query Processed in 1.35 seconds.
```

## Appendix D. Seed and Query Substitution Parameters

This Appendix contains Seed values and substitution parameters for each stream

### seed values

```

=====
session 00 1110165648
session 01 1110165649
session 02 1110165650
session 03 1110165651
session 04 1110165652
session 05 1110165653
session 06 1110165654
session 07 1110165655
session 08 1110165656
session 09 1110165657
=====

```

### stream 00 substitution parameters

```

=====
14 1996-08-01
2 8 NICKEL AMERICA
9 linen
20 green 1994-01-01 INDONESIA
6 1997-01-01 0.04 25
17 Brand#52 WRAP PKG
18 313
8 KENYA AFRICA LARGE ANODIZED TIN
21 FRANCE
13 pending accounts
3 MACHINERY 1995-03-12
22 19 12 13 34 20 22
24
16 Brand#55 MEDIUM BURNISHED 47
30 38 2 5 24 8 28
4 1994-10-01
11 FRANCE 0.0000000100
15 1994-12-01
1 109
10 1993-10-01
19 Brand#52 Brand#22 Brand#12
9 15 25
5 MIDDLE EAST 1997-01-01
7 GERMANY KENYA
12 SHIP TRUCK 1996-01-01
=====

```

### stream 01 substitution parameters

```

=====
21 UNITED KINGDOM
3 BUILDING 1995-03-28
18 315
5 AFRICA 1997-01-01
11 ROMANIA 0.0000000100
7 UNITED STATES FRANCE
6 1997-01-01 0.09 24
20 rosy 1997-01-01 UNITED STATES
17 Brand#54 WRAP DRUM
12 FOB TRUCK 1996-01-01
16 Brand#35 ECONOMY PLATED 50 5
19 3 40 16 4 32
15 1997-06-01
13 unusual deposits
10 1994-07-01
2 46 TIN MIDDLE EAST
8 FRANCE EUROPE MEDIUM POLISHED TIN
14 1996-11-01
19 Brand#54 Brand#55 Brand#11
4 16 21
9 lace
22 11 12 22 18 28 25
10
1 117
4 1997-05-01
=====

```

### stream 02 substitution parameters

```

=====
6 1997-01-01 0.07 25
17 Brand#51 SM BAG
14 1997-03-01
16 Brand#25 STANDARD BRUSHED 22
26 19 28 48 4 44 40
19 Brand#11 Brand#43 Brand#51
9 17 29
10 1993-05-01
9 grey
2 33 COPPER ASIA
15 1995-03-01
8 UNITED KINGDOM EUROPE MEDIUM BURNISHED TIN
5 ASIA 1997-01-01
22 32 17 23 29 28 21
19
12 MAIL TRUCK 1997-01-01
7 MOZAMBIQUE UNITED KINGDOM
13 unusual deposits
18 312
1 64
4 1995-02-01
20 cornsilk 1996-01-01 JORDAN
3 MACHINERY 1995-03-14
11 GERMANY 0.0000000100
21 MOROCCO
=====

```

### stream 03 substitution parameters

```

=====
8 MOROCCO AFRICA SMALL BRUSHED NICKEL
5 EUROPE 1997-01-01
4 1997-09-01
6 1997-01-01 0.04 25
17 Brand#53 SM PKG
7 INDIA MOROCCO
1 72
18 314
22 12 32 31 29 30 27
16
14 1997-06-01
9 forest
10 1994-02-01
15 1997-10-01
11 SAUDI ARABIA 0.0000000100
20 navy 1994-01-01 CANADA
2 21 STEEL MIDDLE EAST
21 GERMANY
19 Brand#14 Brand#21 Brand#55
4 18 25
13 unusual deposits
16 Brand#55 LARGE ANODIZED 19 31
3 44 45 10 26 41
12 RAIL MAIL 1997-01-01
3 BUILDING 1995-03-30
=====

```

### stream 04 substitution parameters

```

=====
5 MIDDLE EAST 1993-01-01
21 UNITED STATES
14 1997-09-01
19 Brand#11 Brand#54 Brand#54
10 19 21
15 1995-06-01
17 Brand#55 SM DRUM
12 AIR MAIL 1997-01-01
6 1993-01-01 0.02 24
4 1995-06-01
9 deep
8 GERMANY EUROPE SMALL PLATED NICKEL
16 Brand#35 PROMO PLATED 24 2
21 37 27 11 45 46
11 INDIA 0.0000000100
2 9 BRASS ASIA
10 1994-11-01
18 315
1 lace
13 unusual deposits
7 ALGERIA GERMANY
22 17 30 29 15 24 16
23
3 HOUSEHOLD 1995-03-16
20 azure 1997-01-01 CHINA
=====

```

=====  
**stream 05 substitution parameters**  
=====

21 PERU  
15 1993-03-01  
4 1993-03-01  
6 1993-01-01 0.07 25  
7 PERU UNITED STATES  
16 Brand#25 SMALL POLISHED 8 23  
47 19 1 29 31 11  
19 Brand#23 Brand#42 Brand#43  
5 10 28  
18 313  
14 1997-12-01  
22 14 25 15 10 17 18  
32  
11 VIETNAM 0.000000100  
13 unusual deposits  
3 BUILDING 1995-03-01  
1 88  
2 47 TIN AFRICA  
5 AFRICA 1993-01-01  
8 UNITED STATES AMERICA SMALL ANODIZED NICKEL  
20 lavender 1996-01-01 GERMANY  
12 REG AIR MAIL 1997-01-01  
17 Brand#52 LG BAG  
10 1993-08-01  
9 coral

=====  
**stream 06 substitution parameters**  
=====

10 1994-06-01  
3 HOUSEHOLD 1995-03-18  
15 1995-10-01  
13 unusual packages  
6 1993-01-01 0.05 25  
8 MOZAMBIQUE AFRICA STANDARD POLISHED  
NICKEL  
9 brown  
7 INDONESIA MOZAMBIQUE  
4 1995-09-01  
11 INDONESIA 0.000000100  
22 32 16 22 18 19 17  
11  
18 314  
12 SHIP MAIL 1993-01-01  
1 96  
5 AMERICA 1993-01-01  
16 Brand#55 ECONOMY ANODIZED 17  
44 37 10 9 20 12 35  
2 35 COPPER ASIA  
14 1993-03-01  
19 Brand#25 Brand#25 Brand#42  
10 11 24  
20 smoke 1994-01-01 RUSSIA  
17 Brand#54 LG PKG  
21 INDONESIA

=====  
**stream 07 substitution parameters**  
=====

18 312  
8 INDIA ASIA STANDARD BURNISHED NICKEL  
20 firebrick 1993-01-01 JAPAN  
21 ARGENTINA  
2 22 STEEL AFRICA  
4 1993-06-01  
22 22 29 17 10 30 16  
28  
17 Brand#51 LG DRUM  
1 104  
11 RUSSIA 0.000000100  
9 beige  
19 Brand#22 Brand#13 Brand#42  
6 12 21  
3 AUTOMOBILE 1995-03-03  
13 unusual packages  
5 ASIA 1993-01-01  
7 ARGENTINA INDIA  
10 1993-03-01  
16 Brand#35 STANDARD BURNISHED 14  
47 12 36 45 13 32 10  
6 1993-01-01 0.02 24  
14 1993-07-01  
15 1993-07-01

12 FOB REG AIR 1993-01-01  
=====

**stream 08 substitution parameters**  
=====

19 Brand#34 Brand#41 Brand#31  
1 13 28  
1 112  
15 1996-01-01  
17 Brand#53 MED BAG  
5 EUROPE 1994-01-01  
8 ALGERIA AFRICA PROMO BRUSHED BRASS  
9 white  
12 MAIL FOB 1993-01-01  
14 1993-10-01  
7 CHINA ALGERIA  
4 1996-01-01  
3 FURNITURE 1995-03-20  
20 pink 1996-01-01 BRAZIL  
16 Brand#25 MEDIUM POLISHED 24 7  
23 5 6 27 36 34  
6 1994-01-01 0.07 24  
22 13 29 10 12 11 26  
22  
10 1993-12-01  
13 express packages  
2 10 BRASS EUROPE  
21 CHINA  
18 313  
11 IRAN 0.000000100

=====  
**stream 09 substitution parameters**  
=====

8 PERU AMERICA PROMO PLATED BRASS  
13 express packages  
2 48 TIN AFRICA  
20 burlywood 1995-01-01 MOZAMBIQUE  
17 Brand#55 MED PKG  
3 AUTOMOBILE 1995-03-05  
6 1994-01-01 0.05 25  
21 IRAN  
18 315  
11 UNITED KINGDOM 0.000000100  
19 Brand#31 Brand#34 Brand#35  
6 14 24  
10 1994-09-01  
15 1993-10-01  
4 1993-10-01  
22 29 11 25 15 19 10  
17  
1 120  
7 IRAN PERU  
12 RAIL FOB 1993-01-01  
9 tan  
14 1994-01-01  
5 MIDDLE EAST 1994-01-01  
16 Brand#55 ECONOMY BRUSHED 10 6  
15 34 18 47 2 38

## Appendix E. Implementation-Specific Layer/Driver Code

### runTPCHall

```
#####
runTPCHall
#####
#!/bin/ksh
. $KIT_DIR/env

ECHO=echo

sqlplus=$ORACLE_HOME/bin/sqlplus
svrmgrl=$ORACLE_HOME/bin/svrmgrl
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
LD4IXCRE=${OUT_DIR}/Ld4ixcre
LD5ANLYZ=${OUT_DIR}/Ld5anlyz

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
STIME=`SGTIME`
echo "Start: timed load portion `date`" >>
$SCRIPT_LOG_FILE
dbcre.sh > $LD1DBCRE
sctso.sh > $LD2SCTSO
dapop.sh > $LD3DAPOP
ixcre.sh > $LD4IXCRE
anlyz.sh > $LD5ANLYZ

$KIT_DIR/audit/gen_seed.sh $KIT_DIR/audit/seed
echo Generated seed: `cat $KIT_DIR/audit/seed` >>
$SCRIPT_LOG_FILE

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}

runTPCHpt ${SCALE_FACTOR} 2 ${RUN_ID}
```

```
sleep 600
tshut >> $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

#####
runTPCHpt
#####
#!/bin/ksh
. $KIT_DIR/env
SCRIPT_DIR=${KIT_DIR}/scripts
SQL_DIR=${KIT_DIR}/sql
UPD_DIR=${KIT_DIR}/update
SRC_DIR=${KIT_DIR}/utils
QRY_DIR=${KIT_DIR}/queries # this is the location of
the query template file
QGEN_DIR=${KIT_DIR}/dbgen
QGEN=${QGEN_DIR}/qgen
QEXEC=${SRC_DIR}
DSS_QUERY=${KIT_DIR}/queries
export DSS_QUERY

UPD_SQL=${UPD_DIR}/sql
UPD_SPT=${UPD_DIR}/scripts
UPD_SRC=${UPD_DIR}/source
UPD_DAT=${UPD_DIR}/data

TPCD_BIN=${KIT_DIR}/audit/bin

GTIME=${SRC_DIR}/gtime
SEED_FILE=${KIT_DIR}/audit/seed

DF=/dev/null
HID=1
INTERVAL=60
COUNT=1200

# The defaults
QPROG=${QEXEC}/qexec

usage () {
echo " "
echo "Usage: $0 [-p <program for query stream>] [-ul
<program for UF1>]"
echo "          [-u2 <program for UF2>] [-o] [-s] [-h]
[-u <user/password>]"
echo "          <scale factor> <run_number>"
echo " "
echo "scale factor      : The scale factor of the run."
echo "update ||ism     : The parallelism to use for
the UFs."
echo " "
echo "-p <program>      : Program for Query Stream."
echo "                  Default is $QPROG."
echo "-ul <program>     : Program for UF1."
echo "                  Default is $U1PROG."
echo "-u2 <program>     : Program for UF2."
echo "                  Default is $U2PROG."
echo "-o                : Collect Oracle statistics."
echo "-s                : Collect System statistics."
echo "-u <user/passwd> : User/Password. Default is
tpch/tpch."
echo "-h                : Displays this message."
}
set -- `getopt "p:ul:u2:osu:h" "$@"` || usage

while :
do
    case "$1" in
        -u1) shift; U1PROG=$1;;
        -u2) shift; U2PROG=$1;;
        -p) shift; QPROG=$1;;
        -o) OSTAT=1;;
        -s) SSTAT=1;;
        -h) usage; exit 0;;
        --) shift; break;;
        esac
    shift;
done
```



```

if [ "$#" -ne "3" ]
then
    usage
    exit 1
fi

SF=$1
PARA=$2
RUN_ID=$3

OUT_DIR=${KIT_DIR}/audit/tests/${RUN_ID}
if [ ! -d $OUT_DIR ]
then
    mkdir $OUT_DIR
fi

TPCD_LOG=${OUT_DIR}
TPCD_RPT=${OUT_DIR}
OUT=${OUT_DIR}

let UF_SET="($PARA-1)*($NUM_STREAMS+1)+1"
START_SET=1
let STOP_SET=$NUM_STREAMS
let START_SET_UPDATE="($PARA-1)*($NUM_STREAMS+1)+2"
let STOP_SET_UPDATE="$START_SET_UPDATE+$NUM_STREAMS-1"

TPCD_LOG_FILE=${TPCD_LOG}/m${PARA}s0
TPCD_RPT_FILE=${TPCD_RPT}/m${PARA}s0inter
QRY_FILE=${TPCD_RPT}/qtemp.${PARA}s0
QUERY_PARAMETER=${TPCD_LOG}/qp${PARA}.0
SCRIPT_LOG_FILE=${TPCD_LOG}/m${PARA}timing
UF1_LOG=${TPCD_LOG}/m${PARA}s0rf1
UF2_LOG=${TPCD_LOG}/m${PARA}s0rf2
STREAM_COUNT_LOG=${TPCD_LOG}/m${PARA}tstcrnt

echo "TPC-H Test - RUN:${PARA} SEQUENCE:${RUN_ID}
`date`" > $SCRIPT_LOG_FILE
echo "TPC-H Test - RUN:${PARA} SEQUENCE:${RUN_ID}
`date`" > $TPCD_RPT_FILE
echo "Generates query template file with seed: `cat
$SEED_FILE` for stream 0" >> $SCRIPT_LOG_FILE
echo >> $SCRIPT_LOG_FILE

${QGEN} -c -r `cat $SEED_FILE` -p 0 -s ${SF} -l
$QUERY_PARAMETER > ${QRY_FILE}

START=${GTIME}
echo "Start Power Test - RUN:${PARA} SEQUENCE:${
{RUN_ID} Execution Starts $START, `date`" >>
$SCRIPT_LOG_FILE
echo "" >> $SCRIPT_LOG_FILE

# Execute UF1

SDATE=`date`
UF1_START=${GTIME}
echo "Start UF1 $UF1_START, `date`" >>
$SCRIPT_LOG_FILE

${ECHO} ${UPD_SPT}/runuf1.sh ${UF_SET} >> $UF1_LOG
2>&1
# Execute Query Stream

UF1_END=${GTIME}
E1DATE=`date`

UF1_TIME=`echo $UF1_END - $UF1_START | bc`
echo UF1: Execution Time: $UF1_TIME >>
${TPCD_RPT_FILE}
echo Start Time: $UF1_START, $SDATE >>
${TPCD_RPT_FILE}
echo End Time: $UF1_END, $E1DATE >> ${TPCD_RPT_FILE}
echo "" >> ${TPCD_RPT_FILE}

echo "End UF1 $UF1_END, ${E1DATE}" >> $SCRIPT_LOG_FILE
echo >> $SCRIPT_LOG_FILE

echo "Start Query Part `${GTIME}`, `date`" >>
$SCRIPT_LOG_FILE

${QPROG} ${DATABASE_USER} q${QRY_FILE} l$
{TPCD_LOG_FILE} r${TPCD_RPT_FILE} > $DFD 2>&1

# Execute UF2

UF2_START=${GTIME}
E2DATE=`date`

echo "End Query Part `${GTIME}`, ${E2DATE}" >>
$SCRIPT_LOG_FILE
echo "" >> $SCRIPT_LOG_FILE

echo "Start UF2 $UF2_START, `date`" >>
$SCRIPT_LOG_FILE
${ECHO} ${UPD_SPT}/runuf2.sh ${UF_SET} >> $UF2_LOG
2>&1
UF2_END=${GTIME}
END=${GTIME}
EDATE=`date`

echo "End UF2 $UF2_END, $EDATE" >> $SCRIPT_LOG_FILE
echo >> $SCRIPT_LOG_FILE

echo "End TPC-H Power Test - RUN:${PARA} SEQUENCE:${
{RUN_ID}, $END, $EDATE" >> $SCRIPT_LOG_FILE
MEA_INT=`echo $END - $START | bc`
echo "Elapsed Time for TPC-H Power Test - RUN:${PARA}
SEQUENCE:${RUN_ID} is $MEA_INT" >> $SCRIPT_LOG_FILE
echo >> $SCRIPT_LOG_FILE

UF2_TIME=`echo $UF2_END - $UF2_START | bc`
echo UF2: Execution Time: $UF2_TIME >>
${TPCD_RPT_FILE}echo Start Time: $UF2_START, $E2DATE
>> ${TPCD_RPT_FILE}
echo End Time: $UF2_END, $EDATE >> ${TPCD_RPT_FILE}

${KIT_DIR}/audit/abridge.pl ${TPCD_LOG_FILE}

i=$START_SET
PSEED=`cat $SEED_FILE`

while [ $i -le $STOP_SET ]; do
    TPCD_LOG_FILE=${TPCD_LOG}/mt${RUN_ID}_${i}.log
    TPCD_RPT_FILE=${TPCD_RPT}/mt${RUN_ID}_${i}.rpt
    QUERY_PARAMETER=${TPCD_LOG}/qp${PARA}.${i}
    QRY_FILE=${TPCD_RPT}/qtemp.${PARA}s${i}

    PSEED=`expr $PSEED + 1`
    ${QGEN} -c -r ${PSEED} -p ${i} -s ${SF} -l
    $QUERY_PARAMETER > ${QRY_FILE}

    i=`expr $i + 1`
done

TH_START_D=`date`
TH_START_T=${GTIME}
echo >> $SCRIPT_LOG_FILE

rm -f /tmp/th_pipe1
mknod /tmp/th_pipe1 p
rm -f /tmp/th_pipe2
mknod /tmp/th_pipe2 p
i=$START_SET

echo "Start Throughput Test - RUN:${PARA} SEQUENCE:${
{RUN_ID} $TH_START_T, $TH_START_D" >>
$SCRIPT_LOG_FILE

# starts a script to count the streams during the
throughput run
(scnt.sh $PARA $RUN_ID > $STREAM_COUNT_LOG &)
scnt_PID=$!

while [ $i -le $STOP_SET ]; do
    M_SDATE=`date`
    M_STIME=${GTIME}
    TPCD_LOG_FILE=${TPCD_LOG}/m${PARA}s${i}
    TPCD_RPT_FILE=${TPCD_RPT}/m${PARA}s${i}inter
    echo "Start Query Stream $i $M_STIME, ${M_SDATE}" >>
$SCRIPT_LOG_FILE
    QRY_FILE=${TPCD_RPT}/qtemp.${PARA}s${i}
    ${QPROG} ${DATABASE_USER} q${QRY_FILE} l$
{TPCD_LOG_FILE} r${TPCD_RPT_FILE} | grep -v "Connected
to ORACLE" >> $SCRIPT_LOG_FILE &
    i=`expr $i + 1`
done
(${KIT_DIR}/audit/runTPCHus $RUN_ID $START_SET_UPDATE
$STOP_SET_UPDATE ${SF} $PARA >> $SCRIPT_LOG_FILE 2>&1
&)

```

```

wait
THQ_END_T=`$GTIME`
THQ_END_D=`date`
echo End all Query Streams $THQ_END_T, $THQ_END_D >>
$SCRIPT_LOG_FILE
print > /tmp/th_pipe1
read < /tmp/th_pipe2

TH_END_D=`date`
TH_END_T=`$GTIME`
echo End Update Stream ${TH_END_T}, ${TH_END_D} >>
$SCRIPT_LOG_FILE
echo >> $SCRIPT_LOG_FILE
echo "End Throughput Test ${TH_END_T}, ${TH_END_D}" >>
$SCRIPT_LOG_FILE
echo Execution Time Throughput Test: `echo ${TH_END_T}
- ${TH_START_T} | bc` >> $SCRIPT_LOG_FILE

i=$START_SET
while [ $i -le $STOP_SET ]; do
    TPCD_LOG_FILE=${TPCD_LOG}/m${PARA}s${i}
    ${KIT_DIR}/audit/abridge.pl ${TPCD_LOG_FILE}
    i=`expr $i + 1`
done
#kill -9 $scnt_PID
/usr/bin/pkill scnt

```

## runTPCHus

```

=====
#!/bin/ksh
. $KIT_DIR/env

SCRIPT_DIR=${KIT_DIR}/scripts
SQL_DIR=${KIT_DIR}/sql
UPD_DIR=${KIT_DIR}/update
UPD_SPT=${UPD_DIR}/scripts
SRC_DIR=${KIT_DIR}/utils
QRY_DIR=${KIT_DIR}/queries # this is the location of
the query template file
QGEN_DIR=${KIT_DIR}/dbgen
QGEN=${QGEN_DIR}/qgen

DSS_QUERY=${KIT_DIR}/queries
export DSS_QUERY

RUN_ID=$1
START_SET_UPDATE=$2
STOP_SET_UPDATE=$3
SF=$4
PARA=$5

OUT_DIR=${KIT_DIR}/audit/tests/${RUN_ID}
if [ ! -d $OUT_DIR ]
then
    mkdir $OUT_DIR
fi

TPCD_RPT=$OUT_DIR
SCRIPT_LOG_FILE=${OUT_DIR}/m${PARA}timing
OUT=$OUT_DIR

GTIME=${SRC_DIR}/gtime
HID=1

START=`$GTIME`
echo "Start Update Stream $START, `date`" >>
$SCRIPT_LOG_FILE
echo "" >> $SCRIPT_LOG_FILE

#waiting for all the query streams to finish first
read < /tmp/th_pipe1

i=$START_SET_UPDATE
j=1
while [ $i -le $STOP_SET_UPDATE ]; do

    # Execute UF1

    UF1_LOG=${OUT_DIR}/m${PARA}s${j}rf1
    UF2_LOG=${OUT_DIR}/m${PARA}s${j}rf2
    RPT_FILE=${OUT_DIR}/m${PARA}s${j}inter

```

```

SDATE=`date`
UF1_START=`$GTIME`
echo "Start UF1-${j} at ${UF1_START}, ${SDATE}"
>> ${RPT_FILE}

${UPD_SPT}/runuf1.sh ${i} >> ${UF1_LOG} 2>&1
UF1_END=`$GTIME`
EDATE=`date`
echo "End UF1-${j} at ${UF1_END}, ${EDATE}" >>
${RPT_FILE}
echo UF1-${j} Execution Time: `echo ${UF1_END}
- ${UF1_START} | bc` >> ${RPT_FILE}

# Execute UF2

SDATE=`date`
UF2_START=`$GTIME`
echo "Start UF2-${j} ${UF2_START}, ${SDATE}" >>
${RPT_FILE}

${UPD_SPT}/runuf2.sh ${i} >> ${UF2_LOG} 2>&1
UF2_END=`$GTIME`
EDATE=`date`
echo "End UF2-${j} at $UF2_END, ${EDATE}" >>
${RPT_FILE}
echo UF2-${j} Execution Time: `echo ${UF2_END}
- ${UF2_START} | bc` >> ${RPT_FILE}

i=`expr $i + 1`
j=`expr $j + 1`
done

print > /tmp/th_pipe2

```

## runuf1.sh

```

=====
#!/bin/ksh
. $KIT_DIR/env
O=${ORACLE_HOME}
UPDATE_DIR=${KIT_DIR}/update
SCRIPT_DIR=${UPDATE_DIR}/scripts
UTILS_DIR=${KIT_DIR}/utils
LOG_DIR=${UPDATE_DIR}/log
GTIME=${SCRIPT_DIR}/gtime
SF=${SCALE_FACTOR}
PAR_HINT=${UPDATE_DOP_INS}
LOGPATH=.
PASSWD=${DATABASE_USER}
if [ $# -lt 1 ];
then
echo runuf1.sh setnum
exit 1
fi
SETNUM=$1
i=1
PID=""

START=`$GTIME`

sqlplus /NOLOG << !
connect $PASSWD;
set timing on
set serveroutput on
set echo on
drop directory data_dir;
create directory data_dir as '/ffl/updates';
drop table temp_l_et;
create table temp_l_et(
    l_orderkey number ,
    l_partkey number ,
    l_suppkey number ,
    l_linenum 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 (
        'lineitem.tbl.u${SETNUM}'
    )
)
reject limit unlimited parallel 24;

drop table temp_o_et;

create table temp_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 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 parallel 24;

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 ,
l_linenum ,
l_shipinstruct ,
l_comment
from temp_l_et;

```

```

commit;
drop table temp_l_et;
drop table temp_o_et;

exit;
!

END=`$GTIME`

echo ""
echo "Update Function 1 Set $$SETNUM done!"
echo "Elapsed Time is `echo $END - $START | bc`"
echo ""
=====
runuf2.sh
=====
#!/bin/ksh
. $KIT_DIR/env
UPDATE_DIR=${KIT_DIR}/update
SCRIPT_DIR=${UPDATE_DIR}/scripts
UTILS_DIR=${KIT_DIR}/utils
GTIME=${SCRIPT_DIR}/gtime
LOG_DIR=${UPDATE_DIR}/log
PAR_HINT=${UPDATE_DOP_DEL}
SF=${SCALE_FACTOR}
PASSWD=${DATABASE_USER}

if [ $# -lt 1 ]
then
usage
exit 1
fi

SETNUM=$1

i=1
PID=""

START=`$GTIME`

sqlplus /NOLOG << !
connect $PASSWD;
set timing on
set serveroutput on
set echo on
drop directory data_dir;
create directory data_dir as '/ff1/updates';
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
        badfile 'okey.${SETNUM}.bad'
        logfile 'okey.${SETNUM}.log'
        fields terminated by '|'
        missing field values are null
    )
    location (
        'delete.${SETNUM}')
)
reject limit unlimited parallel 32;

create table temp_okey (t_orderkey, constraint tokey1
primary key (t_orderkey))
organization index parallel 32 nologging
as select * from
temp_okey_et;

execute dbms_stats.gather_table_stats('TPCH' ,
'temp_okey', estimate_percent => 1, degree => 32);

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(t o) */ o.rowid from
orders o, temp_okey t

```

```

where o.o_orderkey = t.t_orderkey );

delete from (select /*+ use_nl(t l) */ l.rowid from
lineitem l,temp_okey t
where l.l_orderkey = t.t_orderkey );

commit;

drop table temp_okey;
drop table temp_okey_et;
exit;
!

END=`$GTIME`

echo ""
echo "Update Function 2 Set $SETNUM done!"
echo "Elapsed Time is `echo $END - $START | bc`"
echo ""

```

**env**

```

##### PATHS #####
export KIT_DIR=/export/btmp/home/oracle/kit
export SCHEMA_DIR=$KIT_DIR/schema
export PERL=/opt/PERL/bin/perl
export BUMPX_DIR=$KIT_DIR/bumpx
export BUMPX_OUT=$KIT_DIR/bumpx
export UTILS=$KIT_DIR/utills
# change to a regular directory
export TEST_DB=$KIT_DIR/acid
export QUAL_DB=$TEST_DB
export DBGEN=$KIT_DIR/dbgen
export ACID_DIR=$KIT_DIR/acid
export QEXEC=$KIT_DIR/utills
export QUERIES=$KIT_DIR/queries
export ANSWERS=$KIT_DIR/answers
export
ANS2VAL=/export/btmp/home/oracle/kit/acid/answers
export ACID_OUT=$ACID_DIR/acid_out
export DSS_CONFIG=$DBGEN
export DSS_QUERY=$KIT_DIR/queries
export DSS_PATH=$ADE_VIEW_ROOT
export MAINT=$KIT_DIR/maintenance
export CC=cc
export FRAME=$KIT_DIR/frame
export REGR_TEST=$KIT_DIR/internal/regression_test
export SCALE_FACTOR=10000
export UPDATE_DOP_INS=144
export UPDATE_DOP_DEL=144
##### FRAME STUFF
export FRAME_PATH=$KIT_DIR/frame
export ORACORE3INCL=$ORACLE_HOME/rdbms/demo
export ORACORE3PUBL=$ORACLE_HOME/rdbms/public
export RDBMSPUBL=$ORACLE_HOME/rdbms/public
export NETWORKPUBL=$ORACLE_HOME/network/public
export RDBMSDEMO=$ORACLE_HOME/rdbms/demo
export PLSQLEMO=$ORACLE_HOME/plsql/demo
export PLSQLPUBL=$ORACLE_HOME/plsql/public
export O=$ORACLE_HOME
export PATH=./:${BUMPX_DIR}:${UTILS}:${DBGEN}:
${MAINT}:${ACID_DIR}:${FRAME}/bin:${FRAME}/bin:${PATH}
#
##### ENVIRONMENT VARIABLES #####
export WORKLOAD=TPCH
export HOST=
export OPTLEVEL=X02
export GETOPT=-DSTDLIB_HAS_GETOPT
export PLATFORM=
#export INITORA=$KIT_DIR/schema/test_db/testdb.ora
export INITORA=$KIT_DIR/schema/test_db/sf100.ora

##### ALIASES #####

##### RULES - do not change these #####
case "$SCALE_FACTOR" in
  1) export NUM_STREAMS=2;;
  10) export NUM_STREAMS=3;;
  100) export NUM_STREAMS=4;;
  300) export NUM_STREAMS=6;;
  1000) export NUM_STREAMS=7;;
  3000) export NUM_STREAMS=8;;
  10000) export NUM_STREAMS=9;;
esac

```

```

DATABASE_USER=tpch/tpch

=====
qexecpl.c
=====
#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 gettime();

/* 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 */

/* 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 */

```

```

sctype 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[3]; /* Number of the query being
executed */
char qnp[3]; /* 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;
OCIError *errhp = NULL;
OCIError *errhp = NULL;
OCISvcCtx *tpcsvc = NULL;
OCISession *tpcusr = NULL;
OCISstmt *curq = NULL;
OCISstmt *cur_dml = NULL;
OCISstmt *cur_ddl = NULL;
OCIParam *tpcpar = NULL;

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

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

void usage() {

    fprintf(stderr, "\nUsage: gexec username/password
[q<path name for query template file>]\n");
    fprintf(stderr, " [l<path name for
log>] [r<path name for reports>]\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,
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]== '@') {
                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;

    /* 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] = (dltyp * ) memalloc (sizeof(dltyp));
            dlist[i]->defhdl = NULL;
        }
        /* OCIhallocc(curq,&(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_dll,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,errh
p);
    OCIaset(tpcusr,OCI_HTYPE_SESSION,logname,strlen
(logname),OCI_ATTR_USERNAME,
errhp);
    OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen
(passwd),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,err
hp);

    /*
    if ((status=OCILogon((OCIEnv *)tpcenv,(OCIError *)
errhp,(OCISvcCtx *)tpcsvc,
(text *)logname, strlen(logname),
(text *)passwd,
strlen(passwd), (text *) 0, 0)) !=
OCI_SUCCESS)
        sql_error(errhp, status, 1);
    */
    printf("\nConnected to ORACLE as user: %s\n\n",
logname);
}

/* 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 = gettimeofday();
        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);

        /* Let's fflush stuff so that we can see what's
going on */

        fflush(logfile);
        fflush(rep);
    }
    else
        tr_start = gettimeofday();

    s_tr_start = gettimeofday();

    /* 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,&stmttyp,NULL,OCI_ATTR_STMT_TYPE,
errhp);

if (stmttyp != 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 statements will screw it up */

OCIaget
(curq,OCI_HTYPE_STMT,&rows_ret,NULL,OCI_ATTR_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()
{
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;
}
#endif
}
}
}

```



```

}
#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 */
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_ATTR_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 = OCISstmtFetch
(curq,errhp,MAX_ARRAY,OCI_FETCH_NEXT,
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,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,NULL,
OCI_ATTR_ROW_COUNT,errhp);
print_rows(num,(num_so_far-rows_ret));
rows_ret = num_so_far;
if (ntf <= 0) break;
}
} else {
OCISstmtFetch(curq, errhp, ntf, OCI_FETCH_NEXT,
OCI_DEFAULT);
OCIaget
(curq,OCI_HTYPE_STMT,&rows_ret,NULL,OCI_ATTR_ROW_COUNT
,errhp);
print_rows(num,rows_ret);

fprintf(logfile,"\n\n%d row%c processed.\n",
rows_ret,
rows_ret == 1 ? '\0' : 's');
}

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;
}
}
}

```

```

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 */
break;
default:
fprintf(logfile, "%*s ", -(cwid), (dlist[j]-
>sbuf)[i]);
break;
}
}

fprintf(logfile, "\n");
}

/* remove_newline(): Remove newline character from
str. */
void remove_newline(str)
char *str;
{
char *p;

while ((p = strchr(str, '\n')) != NULL)
*p = ' ';
}

=====
qexecpl.h
=====
#ifdef 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>

#ifdef OCIDFN
#include <ocidfn.h>
#endif /* OCIDFN */

#ifdef 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

/* 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; */
    OCITypeCode 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) \
    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=OCISstmtExecute
(svch,stmh,errh,iter,0,NULL,NULL,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 */
```

```
=====
```

## **gtime.c**

---

---

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

# include <sys/time.h>

main ()
{
    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 */
```

## Appendix F. Misc database scripts

Activity Between Database Load and Run1 When the load finished, the runTPCHall script automatically selected a seed value and saved it.

Then the 2 auditor scripts count.sql and dbtables.sql were run to validate that the database structure was correct.

### count.sql

```
=====
select * from lineitem where rownum < 11;
select * from orders where rownum < 11;
select * from part where rownum < 11;
select * from partsupp where rownum < 11;
select * from supplier where rownum < 11;
select * from customer where rownum < 11;
select * from nation where rownum < 11;
select * from region where rownum < 11;
=====
```

### dbtables.sql

```
=====
set numwidth 25
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)
FROM PARTSUPP WHERE PS_PARTKEY =19763);
```

```
SELECT COUNT(*) FROM SUPPLIER;
```

```
SELECT * FROM SUPPLIER
WHERE S_SUPPKEY IN (83,265,492,784,901,1000)
ORDER BY S_SUPPKEY;
```

```
DROP TABLE MINMAX;
```

```
CREATE TABLE MINMAX
(TNAME CHAR(15),
KEYMIN INTEGER,
KEYMAX INTEGER);
```

```
INSERT INTO MINMAX
SELECT 'LINEITEM_ORD',MIN(L_ORDERKEY),MAX(L_ORDERKEY)
FROM LINEITEM ;
```

```
INSERT INTO MINMAX
SELECT 'LINEITEM_NBR',MIN(L_LINENUMBER),MAX
(L_LINENUMBER)
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_PARTKEY)
FROM PARTSUPP;
```

```
INSERT INTO MINMAX
SELECT 'PARTSUPP_SUPP',MIN(PS_SUPPKEY),MAX(PS_SUPPKEY)
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;
```

### tshut

```
=====
#!/bin/ksh
export ORACLE_SID=tpch

if [ "$1" = "abort" ]; then
sqlplus /NOLOG<< !
connect / as sysdba
shutdown abort
exit
!
else
sqlplus /NOLOG<< !
connect / as sysdba
shutdown immediate
exit
!
=====
```

---

fi  
exit

---

## Appendix G. Pricing information

---

For Oracle pricing please contact:

MaryBeth Pierantoni  
650-506-2118  
[mary.beth.pierantoni@oracle.com](mailto:mary.beth.pierantoni@oracle.com)

For Sun pricing please contact:

Daryl Madura  
503-617-8588  
[daryl.madura@sun.com](mailto:daryl.madura@sun.com)