HP Integrity Superdome - Dual-Core Itanium2/1.6 GHz/18MB iL3 - 64p/128c
using
HP-UX 11i v3 64-bit
and
Oracle Database 10g Release 2 Enterprise Edition with Partitioning and Oracle Automatic Storage Management

TPC Benchmark™ H
Full Disclosure Report

First Edition

November 30, 2006

First Edition - November 30, 2006

Hewlett-Packard Company, the sponsor of this benchmark test, believes that the information in this document is accurate as of the publication date. The information in this document is subject to change without notice. The sponsors assume no responsibility for any errors that may appear in this document. The pricing information in this document is believed to accurately reflect the current prices as of the publication date. However, the sponsors provide no warranty of the pricing information in this document.

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

All performance data contained in this report was obtained in a rigorously controlled environment. Results obtained in other operating environments may vary significantly. No warranty of system performance or price/performance is expressed or implied in this report.

Printed in U.S.A., November 30, 2006.

Overview

This report documents the methodology and results of the TPC Benchmark™ H test conducted on the HP Integrity Superdome - Dual-Core Itanium2/1.6 GHz/18MB iL3 - 64p/128c, in conformance with the requirements of the TPC Benchmark™ H Standard Specification, Revision 2.3.0. The operating system used for the benchmark was HP-UX 11i v3 64-bit; the DBMS was Oracle Database 10g Release 2 Enterprise Edition with Partitioning and Oracle Automatic Storage Management.

Standard and Executive Summary Statements

The pages following this preface contain the Executive Summary and Numerical Quantities Summary of the benchmark results.

Auditor

The benchmark configuration, environment and methodology used to produce and validate the test results and the pricing model used to calculate the cost per QphH was audited by Francois Raab, InfoSizing, to verify compliance with the relevant TPC specifications.

TPC Benchmark H Overview

The TPC Benchmark ™ H (TPC-H) is a decision support benchmark. It consists of a suite of business oriented ad-hoc queries and concurrent data 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(e.g., via a point and click GUI interface);
- 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.

The TPC-H operations are modeled as follows:

- The database is continuously available 24 hours a day, 7 days a week, for ad-hoc queries from multiple end users and updates against all tables, except possibly during infrequent (e.g., once a month) maintenance sessions;
- The TPC-H database tracks, possibly with some delay, the state of the OLTP database through on-going updates which batch together a number of modifications impacting some part of the decision support database;
- Due to the world-wide nature of the business data stored in the TPC-H database, the queries and the updates may be executed against the database at any time, especially in relation to each other. In addition, this mix of queries and updates is subject to specific ACIDity requirements, since queries and updates may execute concurrently;

- To achieve the optimal compromise between performance and operational requirements the database administrator can set, once and for all, the locking levels and the concurrent scheduling rules for queries and updates.

The minimum database required to run the benchmark holds business data from 10,000 suppliers. It contains almost ten million rows representing a raw storage capacity of about 1 GB. Compliant benchmark implementations may also use one of the larger permissible database populations (e.g. 10000 GB), as defined in Clause 4.1.3.

The performance metrics reported by TPC-H measure multiple aspects of the capability of the system to process queries. The TPC-H metric at the selected size (QphH@Size) is the performance metric. To be compliant with the TPC-H standard, all references to TPC-H results for a given configuration must include all required reporting components (see Clause 5.4.7). The TPC believes that comparisons of TPC-H results measured against different database sizes are misleading and discourages such comparisons.

The TPC-H database must be implemented using a commercially available database management system (DBMS), and the queries executed via an interface using dynamic SQL. The specification provides for variants of SQL, as implementers are not required to have implemented a specific SQL standard in full. TPC-D uses terminology and metrics that are similar to other benchmarks, originated by the TPC and others. Such similarity in terminology does not in any way imply that TPC-H results are comparable to other benchmarks. The only benchmark results comparable to TPC-H are other TPC-H results compliant with the same revision.

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

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

Benchmark sponsors are permitted several possible system designs, provided that they adhere to the model described in Clause 6. A full disclosure report (FDR) of the implementation details, as specified in Clause 8, must be made available along with the reported results.

General Implementation Guidelines

The purpose of TPC benchmarks is to provide relevant, objective performance data to industry users. To achieve that purpose, TPC benchmark specifications require that benchmark tests be implemented with systems, products, technologies and pricing that:

- Are generally available to users;
- Are relevant to the market segment that the individual TPC benchmark models or represents (e.g. TPC-H models and represents complex, high data volume, decision support environments);
- Would plausibly be implemented by a significant number of users in the market segment the benchmark models or represents.

Hewlett-Packard Company does not warrant or represent that a user can or will achieve performance similar to the benchmark results contained in this report. No warranty of system performance or price/performance is expressed or implied by this report

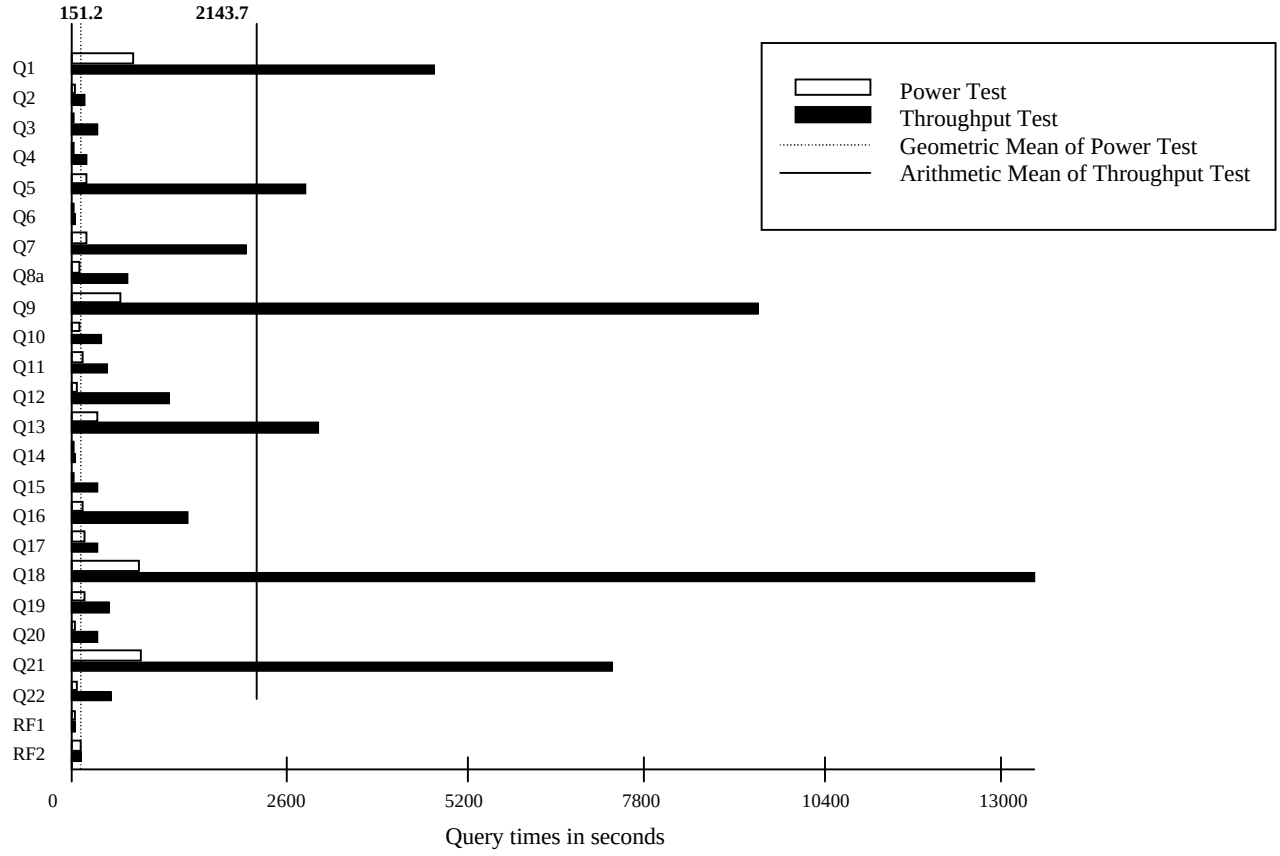
HP Integrity Superdome - Dual-Core Itanium2/1.6 GHz/18MB iL3 - 64p/128c

TPC-H Rev 2.3.0

Report Date: November 30, 2006

Total System Cost	Composite Query per Hour Metric	Price/Performance
$6,681,114 USD	**171,380.0 QphH@10000GB**	**$38.98 USD QphH@10000GB**

Database Size	Database Manager	Operating System	Other Software	Availability Date
10000 GB*	**Oracle Database 10g Release 2 Enterprise Edition with Partitioning and Oracle Automatic Storage Management**	**HP-UX 11i v3 64-bit**	**None**	**April 1, 2007**



Database Load Time = 05:51:44	Load Includes Backup: N	Total Data Storage/Database Size = 11.07	
RAID (Base Tables Only): N	RAID (Base Tables and Auxiliary Data Structures): N	RAID (All): Y	

System Configuration

Processors/Cores/Threads/Type:	64/128/128 / Dual-Core Intel Itanium2 9040 1.6GHz, 9MB iL3 cache per core
Memory:	512 GB
Disk Drives:	1 HP Surestore Disk System 2120 with 4 36GB disks and 256 HP StorageWorks MSA1000 (with a total of 3072 36GB 15K RPM disks)
Total Disk Storage	110736GB (In this number one GB is defined as 1024*1024*1024 bytes)
Lan Controllers	1 PCI 1000BT Lan Adapter

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

HP Integrity Superdome - Dual-Core Itanium2/1.6 GHz/18MB iL3 - 64p/128c

TPC-H Rev 2.3.0

Report Date: November 30, 2006

Description	Part Number	Source	Reference Price	Qty	Extended Price	3 yr. Maint. Price
Server Hardware						
Superdome left chassis	A9834A, Opt 429	1	235,950	1	235,950	
Superdome right chassis	A9835A, Opt 429	1	249,950	1	249,950	
Superdome sx2000 Cell Board	A9837A	1	19,250	16	308,000	
24x7x4hr - 3 Year Svc & Support Price (Hardware and Software)						$1,379,874
256GBBundle SDRAM (128x2GB DIMMS)	A9856A	1	720,896	2	1,441,792	
12-Slot PCI-X I/O Chassis	A9836A	1	16,950	16	271,200	
Dual-Core Intel Itanium2 9040/1.6GHz/18MB L3	AB406A	1	23,000	64	1,472,000	
PCI 1000BT Lan Adapter	A6847A, Opt. 0D1	1	1,325	1	1,325	
PCI 4GB Fibre Channel Adapter (dual port)	AB379A	1	3,995	128	511,360	
PCI Dual Channel Ultra320 SCSI Adapter	A7173A	1	795	1	795	
HPDisk System 2120	A7382A	1	1,070	1	1,070	
1-36GB LP 15K HDD	A7527A	1	966	4	3,864	
HP Universal Rack 10642 G2 Pallet Rack	AF001A	1	1,249	1	1,249	
Modular Power Dist Unit for std racks	A5137AZ	1	145	1	145	
200-240 volts North America	A5137AZ, Opt AW4	1	94	1	94	
HP Tape Array 5300 (DVD and DAT tape)	C7508B	1	729	1	729	
HP rx2620 Server (inc mem/disk/monitor/keyboard/mouse)	AB333A	1	5,315	1	5,315	
I/O Chassis Enclsoure for 12-Slot PCI-X Chassis	A9852A	1	25,750	4	103,000	
Graphite I/O expansion power subsystem	A5861D	1	34,860	2	69,720	
				Subtotal	**4,677,558**	**1,379,874**
Server Software						
Oracle Database 10g Release 2 Enterprise Edition and Oracle Automatic Storage Management, Named User Plus for 3 years		2	10,000	64	640,000	
Partitioning, Named User Plus for 3 years		2	2,500	64	160,000	
Oracle Database Server Support Package for 3 years:		2	6,000	1		6,000
HPUX 11i v3 Foundation Operating Environment	B9429AC	1	2,370	128	303,360	
HP-UX 11i v3 HP9000/Integrity FOE Media	BA489AA	1	565	1	565	
					1,103,925	**6,000**
Storage						
5m Fibre Channel Cables	221692-B22	1	82	256	20,992	
HP StorageWorks MSA 1000 (256 + 26 spares)	201723-B22	1	6,995	282	1,972,590	
3 Yr Support Price for MSA1000 and disks						Included
36GB 15K Ultra320 Hard Drive (3072 + 308 spares)	286776-B22	1	269	3,380	909,220	
HP Universal Rack 10642 G2 Pallet Rack	AF001A	1	1,249	28	35,222	
Modular Power Dist Unit for std racks	A5137AZ	1	145	113	16,356	
200-240 volts North America	A5137AZ, Opt AW4	1	94	113	10,603	
ProLiant Cluster HA/200 for MSA1000	252409-B22	1	4,007	1	4,007	
				Subtotal	**2,968,990**	**0**
				Total	**8,750,473**	**1,385,874**
Oracle Mandatory E-Business Discount on (Licenses and Support)					(161,200)	
35% Large Configuration Discount and Support Prepayment*					(2,755,882)	(538,151)
				Grand Total	**5,833,391**	**847,723**

3-yr Cost of Ownership:	**6,681,114**
QphH@10000GB:	**171,380.0**
$/QphH@10000GB:	**38.98**

Source: 1=HP,
2=Oracle (Pricing Contact: Herve Lejeune
email: Herve.Lejeune@oracle.com; phone number: (650-506-1894)

*All discounts are based on US list prices and for similar quantities and configurations

Audited By: Francois Raab for InfoSizing (www.sizing.com)

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

Measurement Results

Database Scaling (SF/size)	10000
Total Data Storage/Database Size	11.07
Start of Database Load Time	2006-11-16 14:01:06
End of Database Load Time	2006-11-16 19:52:50
Database Load Time	05:51:44
Query Streams for Throughput Test (S)	9
TPC-H Power	238,063.3
TPC-H Throughput	123,375.2
TPC-H Composite Query-per-Hour Metric (QphH@10000GB)	171,380.0
Total System Price Over 3 Years	6,681,114
TPC-H Price/Performance Metric ($/QphH@10000GB)	$39

Measurement Intervals

Measurement Interval in Throughput Test (Ts)	57,775

Duration of Stream Execution:

Power Stream	Seed	RF1 Start Time RF1 End Time	Query Start Time Query End Time	RF2 Start Time RF2 End Time	Duration (sec)
	1116195250	11/17/06 13:33:55 11/17/06 13:34:58	11/17/2006 13:34:58 11/17/2006 15:17:13	11/17/2006 15:17:13 11/17/2006 15:19:52	6,357

Throughput Stream	Seed	Query Start Time Query End Time	Duration (sec)	RF1 Start Time RF1 End Time	RF2 Start Time RF2 End Time
1	1116195251	11/17/2006 15:19:53 11/18/2006 05:56:49	52,617	11/18/2006 06:49:56 11/18/2006 06:51:00	11/18/2006 06:51:00 11/18/2006 06:53:32
2	1116195252	11/17/2006 15:19:53 11/18/2006 05:56:06	52,573	11/18/2006 06:53:32 11/18/2006 06:54:36	11/18/2006 06:54:36 11/18/2006 06:57:11
3	1116195253	11/17/2006 15:19:53 11/18/2006 06:18:09	53,897	11/18/2006 06:57:11 11/18/2006 06:58:16	11/18/2006 06:58:16 11/18/2006 07:00:48
4	1116195254	11/17/2006 15:19:53 11/18/2006 04:56:37	49,005	11/18/2006 07:00:48 11/18/2006 07:01:53	11/18/2006 07:01:53 11/18/2006 07:04:27
5	1116195255	11/17/2006 15:19:53 11/18/2006 06:49:55	55,803	11/18/2006 07:04:27 11/18/2006 07:05:34	11/18/2006 07:05:34 11/18/2006 07:08:06
6	1116195256	11/17/2006 15:19:53 11/18/2006 05:31:25	51,092	11/18/2006 07:08:06 11/18/2006 07:09:13	11/18/2006 07:09:13 11/18/2006 07:11:47
7	1116195257	11/17/2006 15:19:53 11/18/2006 06:26:43	54,411	11/18/2006 07:11:47 11/18/2006 07:12:52	11/18/2006 07:12:52 11/18/2006 07:15:27
8	1116195258	11/17/2006 15:19:53 11/18/2006 06:37:30	55,057	11/18/2006 07:15:27 11/18/2006 07:16:33	11/18/2006 07:16:33 11/18/2006 07:19:09
9	1116195259	11/17/2006 15:19:53 11/18/2006 06:17:48	53,876	11/18/2006 07:19:09 11/18/2006 07:20:15	11/18/2006 07:20:15 11/18/2006 07:22:48

HP Integrity Superdome - Dual-Core Itanium2/1.6 GHz/18MB iL3 - 64p/128c

TPC-H Rev 2.3.0

Report Date November 30, 2006

TPC-H Timing Intervals (in seconds)

Duration of stream execution:

Stream ID	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8a	Q9	Q10	Q11	Q12
Stream 00	912.5	71.5	50.8	47.8	234.5	33.0	251.8	130.1	730.6	120.9	176.9	108.2
Stream 01	5229.7	166.6	325.1	231.8	1898.6	138.4	1974.1	666.2	7933.2	837.7	1490.7	658.9
Stream 02	4633.0	121.0	309.7	360.1	1863.6	167.6	2125.2	545.9	8968.7	1234.0	2136.5	1218.5
Stream 03	5426.2	165.7	134.3	453.2	2171.1	87.4	3168.8	934.3	7435.2	1098.9	4071.0	763.7
Stream 04	4869.8	162.7	351.3	323.7	653.8	74.3	2961.3	622.1	6914.4	885.0	3625.2	1704.2
Stream 05	5354.4	338.8	319.4	194.3	2628.4	67.3	1954.5	738.3	2787.3	737.8	3267.6	1391.9
Stream 06	4730.7	200.9	396.6	247.5	2835.9	69.2	1972.9	806.7	9435.0	448.2	535.0	1440.9
Stream 07	5003.7	128.0	292.3	193.0	2111.5	34.6	1874.4	531.3	8886.6	465.4	2883.9	218.4
Stream 08	5710.2	148.3	135.5	186.6	1973.3	45.7	1864.9	842.6	8115.8	747.2	191.1	940.8
Stream 09	5163.7	223.0	355.9	328.1	370.1	80.6	2833.0	1255.1	12365.0	1028.2	2226.7	1220.9
Minimum	4633.0	121.0	134.3	186.6	370.1	34.6	1864.9	531.3	2787.3	448.2	191.1	218.4
Average	5124.6	183.9	291.1	279.8	1834.0	85.0	2303.2	771.4	8093.5	831.4	2269.7	1062.0
Maximum	5710.2	338.8	396.6	453.2	2835.9	167.6	3168.8	1255.1	12365.0	1234.0	4071.0	1704.2

Stream ID	Q13	Q14	Q15a	Q16	Q17	Q18	Q19	Q20	Q21	Q22	RF1	RF2
Stream 00	380.4	26.0	43.7	190.8	219.5	996.6	221.4	75.8	1013.6	97.7	63.2	159.1
Stream 01	2725.4	100.0	329.1	1521.5	781.7	11011.2	688.2	633.0	12549.5	726.2	64.0	152.7
Stream 02	2186.3	84.0	313.6	1773.7	1628.9	12512.8	558.5	781.6	8536.6	513.3	63.7	155.2
Stream 03	2674.2	75.1	317.6	1207.3	898.1	12837.5	1183.8	429.0	7894.6	469.9	64.3	152.6
Stream 04	2981.2	78.0	272.0	1387.0	464.3	12251.6	515.3	435.5	6794.9	676.9	65.0	153.1
Stream 05	3319.7	94.4	202.8	1285.1	453.6	14344.0	983.9	693.1	13969.4	676.8	67.3	152.1
Stream 06	3047.2	80.7	379.0	1710.5	393.2	13498.4	567.2	389.2	7313.6	593.6	67.1	153.5
Stream 07	3345.5	32.0	51.5	589.0	382.5	17105.1	1339.3	381.3	8059.1	502.2	65.0	155.1
Stream 08	2468.4	67.8	379.2	1393.7	1202.0	15265.0	1932.2	292.7	10669.7	484.4	65.7	156.3
Stream 09	3092.1	66.4	289.2	837.7	463.8	12627.8	1094.1	560.6	6851.3	542.0	66.1	152.2
Minimum	2186.3	32.0	51.5	589.0	382.5	11011.2	515.3	292.7	6794.9	469.9	63.7	152.1
Average	2871.1	75.4	281.5	1300.6	740.9	13494.8	984.7	510.7	9182.1	576.2	65.4	153.6
Maximum	3345.5	100.0	379.2	1773.7	1628.9	17105.1	1932.2	781.6	13969.4	726.2	67.3	156.3

INFO SIZING

TPC TRANSACTION PROCESSING PERFORMANCE COUNCIL CERTIFIED AUDITOR

Benchmark Sponsor: Sharada Bose
Performance Manager BCS
Hewlett-Packard
Pruneridge Avenue, MS4105
94065 Cupertino, CA 95014

November 29, 2006

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

Platform: **HP Integrity Superdome Enterprise Server**

Database Manager: **Oracle Database 10g R2 Enterprise Edition w/ Partitioning**

Operating System: **HP-UX 11.i v3 64-bit**

The results were:

CPU (Speed)	Memory	Disks	QphH@10000GB
HP Integrity Superdome Enterprise Server			
64 x Itanium2 (1.6GHz, dual-core)	18 MB Cache/cpu 512 GB Main	3072 x 36GB ext. 4 x 18GB int.	171,380.0

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:

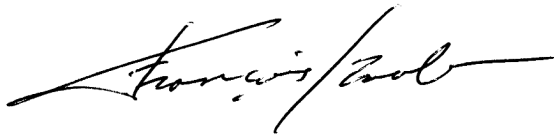
- The database records were defined with the proper layout and size
- The database population was generated using DBGEN
- The database was properly scaled to 10,000GB and populated accordingly
- The compliance of the database auxiliary data structures was verified

- 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 between the longest and the 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:

None.

Respectfully Yours,



François Raab
President

1 General Items

1.1 Benchmark Sponsor

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

Hewlett-Packard Company is the test sponsor of this TPC Benchmark H benchmark.

1.2 Parameter Settings

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

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 HP-UX and Oracle Database 10g Release 2 Enterprise Edition with Partitioning and Oracle Automatic Storage Management parameters used in this benchmark.

1.3 Configuration Diagrams

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

Measured Configuration:

- 64 1.6GHz Dual-Core Intel Itanium2 9040 CPUs each with 9MB iL3 cache per core.
- 512 GB Memory
- 128 PCI Fibre Channel 4GB Adapter (dual-port) Cards
- 2 I/O Expansion Cabinet
- 1 HP 1000 BaseSX PCI Lan Adapters
- 256 HP StorageWorks MSA1000 (with a total of 3072 36GB Disks)
- 1 High Availability Storage Systems (with a total of 4 18GB 10K Disks)
- 1 DVD ROM
- 1 SCSI Card
-

Priced Configuration:

- 64 1.6GHz Dual-Core Intel Itanium2 9040 CPUs each with 9MB iL3 cache per core.
- 512 GB Memory
- 128 PCI Fibre Channel 4GB Adapter (dual-port)
- 2 I/O Expansion Cabinet
- 1 HP 1000 BaseSX PCI Lan Adapters
- 256 HP StorageWorks MSA1000 (with a total of 3072 36GB Disks)

- 1 HP Surestore Disk System 2120 (with a total of 4 36GB 15K 80U4 HDD Disks)
- 1 DVD ROM
- 1 SCSI Cards
-
- The difference between measured and priced is a High Availability Storage System for the root disk which currently is obsolete. For the priced system a Surestore Disk System 2120 was substituted.

Terminal

Keyboard

Mouse

HP Integrity Superdome Enterprise Server

WITH:

64p/128c – 1.6GHz Itanium2 Processors
512GB Memory
128 4GB Fibre Channel Adapters (dual-port)
2 I/O Expansion Cabinet
1 HP 1000 BaseSX PCI Lan Adapter
1 DVD ROM
1 SCSI Card
1 HP Surestore Disk System 2120 with 4 36GB Disks

Priced Configuration

256 HP StorageWorks MSA1000
with 3072 36GB 15k RPM Disks

(26 Cabinets with 10 msa1000 in each)

Terminal

Keyboard

Mouse

HP Integrity Superdome Enterprise Server

WITH:

64p/128c – 1.6GHz Itanium2 Processors
512GB Memory
128 4GB Fibre Channel Adapters (dual-port)
2 I/O Expansion Cabinet
1 HP 1000 BaseSX PCI Lan Adapter
1 DVD ROM
1 SCSI Card
1 High Availability Storage System with 4 18GB Disks

Measured Configuration

256 HP StorageWorks MSA1000
with 3072 36GB 15k RPM Disks

(26 Cabinets with 10 msa1000 in each)

2 Clause 1 Logical Database Design Related Items

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 describes the scripts that define, create, and analyze the tables and indices 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. Columns were reordered in the tables – please refer to the table create statements for the ordering.

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

Section 5.2 describes the distribution of tables and logs across all media.

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.

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 definition 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 actual 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 E 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 the isolation level set to "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 implementation-specific layer/driver code included in Appendix F.

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 ACID test is included in Appendix C.

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.

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.

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.

5. The total price from the ORDERS table and the extended price from the LINEITEM table were retrieved for a randomly selected order key.
6. The ACID Transaction was performed using the order key from step 1. The transaction was stopped prior to the commit.
7. The ACID Transaction was ROLLED BACK.
8. 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.

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.

9. The consistency of the ORDERS and LINEITEM tables was verified based on a sample of order keys.
10. 100 ACID Transactions were submitted from each of 10 execution streams.
11. 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 some order.

Read-Write Conflict with Commit

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

12. An ACID Transaction was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID Transaction was suspended prior to COMMIT.
13. An ACID Query was started for the same O_KEY used in step 1. The ACID Query blocked and did not see any uncommitted changes made by the ACID Transaction.
14. The ACID Transaction was resumed, and COMMITTED.
15. The ACID Query completed. It returned the data as committed by the ACID Transaction.

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.

16. An ACID Transaction was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID Transaction was suspended prior to ROLLBACK.
17. 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.
18. The ACID Transaction was ROLLED BACK.
19. The ACID Query completed.

Write-Write Conflict with Commit

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

20. An ACID Transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID transaction T1 was suspended prior to COMMIT.
21. Another ACID Transaction, T2, was started using the same O_KEY and L_KEY and a randomly selected DELTA.
22. T2 waited.
23. T1 was allowed to COMMIT and T2 completed.
24. It was verified that T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE +(DELTA1*(T1.L_EXTENDEDPRICE/T1.L_QUANTITY))

Write-Write Conflict with Rollback

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

25. An ACID Transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID transaction T1 was suspended prior to ROLLBACK.
26. Another ACID Transaction, T2, was started using the same O_KEY and L_KEY and a randomly selected DELTA.
27. T2 waited.
28. T1 was allowed to ROLLBACK and T2 completed.
29. It was verified that T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE.

Concurrent Progress of Read and Write on Different Tables

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

30. An ACID Transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. T1 was suspended prior to COMMIT.

31. Another ACID transaction, T2 was started using random values for PS_PARTKEY and PS_SUPPKEY, all columns of the PARTSUPP table for which PS_PARTKEY and PS_SUPPKEy are equal are returned.
32. ACID Transaction T2 completed.
33. T1 was allowed to COMMIT.
34. It was verified that the appropriate rows in the ORDER, LINEITEM, and HISTORY tables have been changed.

Read-Only Query Conflict with Update Transactions

Demonstrates 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.

35. A Transaction, T1, was started which executed Q21 against the qualification database, was started using a randomly selected DELTA.
36. An ACID Transaction, T2, was started for a randomly selected O_KEY, L_KEY and DELTA.
37. T2 completed and appropriate rows in the ORDERS, LINEITEM and HISTORY tables had been changed.
38. Transaction T1 completed executing Q21.

4.5 Durability

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

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.

39. The disks containing TPC-H tables and log files were on RAID 1/0 protected disk groups. During the durability test, one disk was removed from each RAID group containing the data and the log. The test continued uninterrupted, because of the RAID protection.

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 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 and the counts were verified.

Memory Failure

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

See the previous section.

5 Clause 4 Scaling and Database Population

5.1 Ending Cardinality of Tables

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

Table	Cardinality
ORDER	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

Distribution of tables and logs across media:

Each MSA array (with 12 disks) was configured as a single Raid-1/0 array group. Each array group was divided into 3 luns.

LUN1 for Oracle/ASM use (eg. tables, indexes, logs)

LUN2 for flat-file data

LUN3 for swap and ACID/qual database tests

OS root and the Oracle home directory were configured on two external disks.

256 LUNs, one from each MSA1000 array, were allocated for Oracle ASM use and a single disk group was built across all LUNs. All tables, indexes, temp space and other Oracle files were configured in this disk group.

5.3 Database Partition/Replication Mapping

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

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

Section 5.2 describes the distribution of tables and logs across all media..

5.4 RAID Feature

Implementation 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.

RAID1/0 was used for all data.

5.5 DBGEN Modification

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 not modified 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 05:51:44.

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:

Type	Quantity	Disk Size	Total
1 HP Surestore Disk System 2120	4	36	144
256 HP StorageWorks MSA1000	3072	36	110,592.0
TOTAL			**110,736.0**
Scale Factor			**10,000**
Storage Ratio			**11.07**

5.8 Database Load Mechanism Details and Illustration

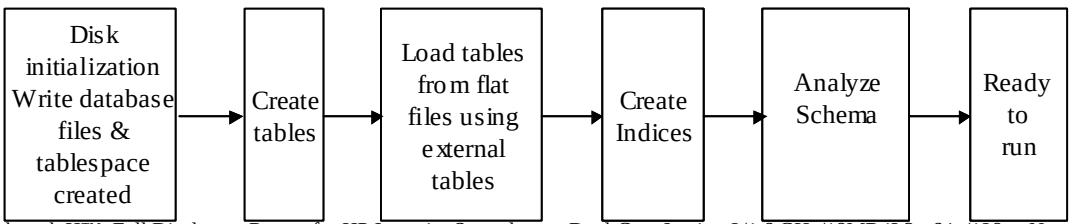
The details of the database load must be described, including a block diagram illustrating the overall process.

The database was loaded using data generation stored on the flat files all on the tested and priced configuration

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 changes to adjust for the database scale factor.



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.

A script was run to display the hardware configurations of the SUT.

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

The database was restarted.

All scripts and queries used are included in Appendix E.

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:

40. Database started
41. RF1 Refresh Transaction
42. Stream 00 Execution
43. RF2 Refresh Transaction

6.3 Timing Intervals for Each Query and Refresh Functions

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

The timing intervals for each query and both update functions are given in the Numerical Quantities Summary earlier in this document.

6.4 Number of Streams for the Throughput Test

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

9 streams were used for the throughput test.

6.5 Start and End Date/Time of 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 given in the Numerical Quantities 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 given in the Numerical Quantities Summary earlier in this document.

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

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

Power Stream	Seed	RF1 Start Time RF1 End Time	RF2 Start Time RF2 End Time
	1116195250	11/17/06 13:33:55 11/17/06 13:34:58	11/17/2006 15:17:13 11/17/2006 15:19:52

Throughput Stream	RF1 Start Time RF1 End Time	RF2 Start Time RF2 End Time
1	11/18/2006 06:49:56 11/18/2006 06:51:00	11/18/2006 06:51:00 11/18/2006 06:53:32
2	11/18/2006 06:53:32 11/18/2006 06:54:36	11/18/2006 06:54:36 11/18/2006 06:57:11
3	11/18/2006 06:57:11 11/18/2006 06:58:16	11/18/2006 06:58:16 11/18/2006 07:00:48
4	11/18/2006 07:00:48 11/18/2006 07:01:53	11/18/2006 07:01:53 11/18/2006 07:04:27
5	11/18/2006 07:04:27 11/18/2006 07:05:34	11/18/2006 07:05:34 11/18/2006 07:08:06
6	11/18/2006 07:08:06 11/18/2006 07:09:13	11/18/2006 07:09:13 11/18/2006 07:11:47
7	11/18/2006 07:11:47 11/18/2006 07:12:52	11/18/2006 07:12:52 11/18/2006 07:15:27
8	11/18/2006 07:15:27 11/18/2006 07:16:33	11/18/2006 07:16:33 11/18/2006 07:19:09
9	11/18/2006 07:19:09 11/18/2006 07:20:15	11/18/2006 07:20:15 11/18/2006 07:22:48

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

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

The timing intervals for each query and each update function are given in the Numerical Quantities 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, is given in the Numerical Quantities 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:

	QppH@10000GB	QthH@10000GB	QphH@10000GB
Reported Run	238,063.3	123,375.2	171,380.0
Reproducibility Run	237,286.0	126,405.4	173,188.4
% Difference	0.3%	2.5%	1.1%

6.11 System Activity Between Performance Tests

Any activity on the SUT that takes place between the conclusion of the Reported Run and the beginning of Reproducibility Run must be disclosed.

No activity between the two runs.

7 Clause 6 SUT and Driver Implementation Related Items

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.

All stream executions are performed by a single script. QGEN is used to produce query text.

For each power-test run:

- The SQL for RF1 is submitted to the database
- Then the queries as generated by QGEN are submitted in the order defined by Clause 5.3.5.4
- The SQL for RF2 is submitted to the database**.**

7.2 Implementation-Specific Layer (ISL)

If an implementation specific layer is used, then a detailed description of how it performs its functions must be provided. All related source code, scripts and configuration files must be disclosed. The information provided should be sufficient for an independent reconstruction of the implementation specific layer.

The source code for the "qexec" utility can be found in Appendix E.

7.3 Profile-Directed Optimization

If profile-directed optimization as described in Clause 5.2. is used, such use must be disclosed..

Profile-directed optimization subject to the requirements of *5.2.9* and *5.2.10* was not used.

8 Clause 7 Pricing

8.1 Hardware and Software Used in the Priced System

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

A detailed list of hardware and software used in the priced system is included in the pricing sheet in the executive summary. All prices are currently effective.

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.

A detailed pricing sheet of all the hardware and software used in this configuration and the 3-year maintenance costs, demonstrating the computation of the total 3-year price of the configuration, is included in the executive summary at the beginning of this document.

8.3 Availability Date

The committed delivery date for general availability of products used in the priced 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.

Availability Dates:

Server Hardware	Now
Server Software	April 1, 2007
Storage	Now
Database Manager (Oracle Database 10g Release 2 Enterprise Edition with Partitioning and Oracle Automatic Storage Management)	January 1, 2007

9 Clause 8 Auditor's Information and Attestation Letter

9.1 Auditor's Report

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

This implementation of the TPC Benchmark H was audited by Francois Raab for InfoSizing Further information regarding the audit process may be obtained from:

Francois Raab
InfoSizing
1373 N. Franklin Steet
Colorado Springs, CO 80903
(719) 473-7555
(719) 473-7554

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

Appendix A Parameter Settings

A.1 10 TB-init.ora

```
instance_type                 = rdbms
aq_tm_processes               = 0
audit_trail                   = FALSE
compatible                    = 10.1.0.2
control_files                 = (+DG1/control1,+DG1/control2)
cpu_count                     = 64
db_block_checksum             = false
db_block_size                 = 32768
db_cache_size                 = 30g
db_file_multiblock_read_count = 64
db_files                      = 2400
db_name                       = 10tb
db_writer_processes           = 16
dml_locks                     = 40000
global_names                  = FALSE
hpux_sched_noage              = 180
instance_name                 = tpch
job_queue_processes           = 0
log_buffer                    = 268435456
log_checkpoints_to_alert      = true
log_checkpoint_interval       = 18000
max_dump_file_size            = unlimited
nls_date_format               = YYYY-MM-DD
open_cursors                  = 1024
optimizer_features_enable     = 10.2.0.1.1
optimizer_index_cost_adj      = 200
optimizer_mode                = CHOOSE
parallel_adaptive_multi_user  = true
parallel_execution_message_size = 65535
parallel_max_servers          = 1600
parallel_min_servers          = 1600
parallel_threads_per_cpu      = 3
pga_aggregate_target          = 150g
processes                     = 5000
recovery_parallelism          = 32
replication_dependency_tracking = false
session_cached_cursors        = 0
shared_pool_size              = 50g
statistics_level              = basic
undo_management               = auto
undo_retention                = 200000
```

A.2 system

```
*
* Created on Mon Nov 13 09:55:06 2006
*
version 1
configuration current "jazz @ 20060922.09:18:24PDT; jmkvw -proj
kern1 -RW -c Task: ruemmler_i80_io_master8
ruemmler_i80_io_master8 i80(I80_BL2006_0828)
cup2_ruemmler_i80_io_master8(auto) ; FLAVOR=perf " [4558b17a]
*
* Module entries
*
module acpi best 1.0.[4527358F]
module acpi_node best 1.0.[4527357E]
module asio0 best 1.0.[4527355C]
module asyncdsk best 1.0.[45273560]
module audio best 1.0.[4527354C]
module autofs best 1.0.[452734F0]
module beep best 1.0.[4527351C]
module btlan best 1.0.[45273546]
module c8xx best 1.0.[45273553]
module cachefs best 1.0.[4527351D]
module cdfs auto 0.1.[45273517]
module cec_gen best 1.0.[452734FD]
module cec_hp best 1.0.[452734F6]
module cec_psm best 1.0.[4527351E]
module clone best 1.0.[451A54C0]
module cn best 1.0.[452734D7]
module devswerr best 0.1.[452734B6]
module diag2 best 1.0.[452734D2]
module dlkm best 1.0.[452734A1]
module dmem best 1.0.[4528847E]
module echo best 1.0.[4527346E]
module eschgr best 1.0.[45273454]
module esctl best 1.0.[45273467]
module esdisk best 1.0.[4527344C]
module eslpt best 1.0.[451A5439]
module estape best 1.0.[45273444]
module estp best 1.0.[451A5429]
module esvroot best 1.0.[451A5423]
module fcd best 1.0.[4528867D]
module fcp static 1.0.[45273420]
module fcparray static 1.0.[4527341D]
module fcpdev static 1.0.[4527341A]
module foundation_psm best 1.0.[45273418]
module gelan best 1.0.[452724C9]
module gh2p best 1.0.[452733DD]
module graph3 best 1.0.[452733B3]
module gvid best 1.0.[452733D4]
module hcd best 1.0.[452733A9]
module hid best 1.0.[452733A9]
module hp_apa best 1.0.[45273E3F]
module hpstreams best 1.0.[452733DB]
module hub best 1.0.[4527338D]
module ia64_psm best 1.0.[45273390]
module iether best 1.0.[452729DC]
module igelan best 1.0.[452728F5]
module ciss best 1.0.[45273B75]
module inet best 1.0.[4527336F]
module intl100 best 1.0.[4527337A]
module iomem best 1.0.[4527333B]
module ipmi best 1.0.[45273333]
module ipmi_psm best 1.0.[4527331F]
module ixgbe best 1.0.[452749F9]
module kepd best 1.0.[45273306]
module kgssapi best 1.0.[452734D8]
module klmmod best 1.0.[452734DE]
module ktest best 1.0.[452732FB]
module ktestio best 1.0.[452732BC]
module ktracer best 1.0.[452732C2]
module kwdb best 2.0.[452732FE]
module kwdb_testdlkm best 0.2.[452732B2]
module lbabest 1.0.[452732B7]
module ldterm best 1.0.[452732B0]
module livedump_drv best 1.0.[4527329B]
module local_sapic best 1.0.[45273290]
module lvm best 1.0.[452892CB]
module lvmcore best 1.0.[45273287]
module mcadev best 0.1.[45273279]
module mm best 1.0.[45273260]
module mpt best 1.0.[45272B01]
module netdiag1 best 1.0.[45273264]
module nfsbest 1.0.[452734C1]
module nfs_client best 1.0.[4527351D]
module nfs_client_pv2 best 1.0.[45273485]
module nfs_client_pv3 best 1.0.[45273479]
```

```
module nfs_client_pv4           best        1.0.[45273486]
module nfssrv           best        1.0.[4527349B]
module nfswrp           best        1.0.[45273470]
module nms              best        1.0.[4527324E]
module pckt             best        1.0.[4527321A]
module pdh              best        1.0.[4527322A]
module pid              best        1.0.[45273212]
module plat_intr        best        1.0.[45273206]
module prm              best        1.0.[45273170]
module procsm           best        1.0.[4527315E]
module ptem             best        1.0.[45273150]
module ptm              best        1.0.[45273148]
module ptsbest          1.0.[451A50DB]
module ptym             best        1.0.[4527314B]
module ptys             best        1.0.[451A50D9]
module ramdisc          best        1.0.[4527312D]
module rmp3f01          best        1.0.[4527312B]
module rng              best        0.1.[45273128]
module rpc              best        1.0.[4527347D]
module rpcmod           best        1.0.[4527345D]
module rpcsec           best        1.0.[4527344B]
module rpcsec_gss       best        1.0.[45273352]
module sac              best        1.0.[45273118]
module sad              best        1.0.[45273123]
module sapic            best        1.0.[4527310F]
module sasd             best        1.0.[45274B8B]
module sba              best        1.0.[4527311B]
module schgr            best        1.0.[451A5094]
module sctl             best        1.0.[451A5094]
module sdisk            best        1.0.[451A508F]
module side             best        1.0.[452730DF]
module smbios_psm       best        1.0.[452730C2]
module stape            best        1.0.[451A504E]
module sy  best         1.0.[45273080]
module telm             best        1.0.[45273067]
module tels             best        1.0.[451A4F90]
module td  best         1.0.[4527237D]
module tgt best         1.0.[451A4F49]
module tun              best        1.0.[45273008]
module ufsbest          1.0.[4527303E]
module usbd             best        1.0.[45272FFB]
module vmm_cdio         best        1.0.[452728FA]
module vm_arena_test            best        1.0.[45272979]
module vm_fs_test       best        1.0.[4527295C]
module vcn              best        1.0.[45272FE2]
module vol              best        1.0.[45272F56]
module vols             best        1.0.[45272E56]
module vpar             best        1.0.[45272906]
module vpar_driver      best        1.0.[452727A8]
module vxdmp            best        1.0.[45272E78]
module vxfs             static      41.0.[454912DC]
module vxportal         static      41.0.[454912E3]
module wxb_hp           best        1.0.[45272607]
module gvid_info        loaded      0.1.[45413E7D]
module gvid_him_cons            auto        0.1.[45413E80]
*
* Dump entries
*
dump lvol
*
* Tunables entries
*
tunable as_isolation_level          1
tunable o_sync_is_o_dsync           1
tunable nkthread        11488
tunable filecache_max               3%
tunable process_id_min              0
tunable cmc_plat_poll 15
tunable maxfiles_lim  4096
tunable ninode          120000
tunable nswapdev        100
tunable swchunk         65536
tunable user:unlockable_mem         1
tunable vxfs_ifree_timelag          3600000
tunable semmnu          4092
tunable semmns          8192
tunable semmni          4096
tunable maxuprc         3277
tunable vps_ceiling     64
tunable user:swapmem_on             0
tunable STRMSGSZ  65535
tunable shmseg          512
tunable shmmni          2048
tunable shmmax          0x40000000000
tunable semume          512
tunable nstrpty         200
tunable npty            200
tunable msgtql          5120
tunable msgmnb          65536
tunable maxtsiz_64bit 4294967296
tunable maxtsiz         1073741824
tunable maxssiz_64bit 268435456
tunable maxssiz         0x10000000
tunable maxdsiz_64bit               0x80000000
tunable maxdsiz         0x40000000
tunable max_thread_proc             2048
tunable hfs_revra_per_disk          256
tunable hfs_ra_per_disk             256
tunable hfs_max_revra_blocks        20
tunable hfs_max_ra_blocks           20
tunable create_fastlinks            1
tunable timezone        480
tunable semvmx          32768
tunable nproc           7168
tunable nfile           2000000
tunable msgmni          512
tunable maxfiles        4096
tunable pagezero_daemon_enabled             0
tunable filecache_min 1%
```

A.3 env

```
########## MACHINE PARAMETERS ####################
#export RAC_NODES="titan1 titan2"
########## PATHS #################################
export KIT_DIR=/dbms/oracle10i/kit
export SCHEMA_DIR=$KIT_DIR/schema
export PERL=/opt/perl/bin/perl
export UTILS=$KIT_DIR/utils
export TEST_DB=/tmp
export QUAL_DB=$TEST_DB
export DBGEN=$KIT_DIR/dbgen
export ACID_DIR=$KIT_DIR/acid
export QEXEC=$KIT_DIR/utils
export QUERIES=$KIT_DIR/queries
export ANSWERS=$KIT_DIR/answers
export ANS2VAL=/dbms/oracle10i/kit/acid/answers2validate
export ACID_OUT=$KIT_DIR/out
export DSS_CONFIG=$DBGEN
export DSS_QUERY=$KIT_DIR/queries
export DSS_PATH=$ADE_VIEW_ROOT
export MAINT=$KIT_DIR/maintenance
export CC=/opt/ansic/bin/cc
export FRAME=$KIT_DIR/frame
export FRAME_DIR=/dbms/oracle10i/frame
export SCALE_FACTOR=10000
export UPDATE_1_DOP=64
export UPDATE_2_DOP=128
```

```
############ FRAME STUFF
export FRAME_PATH=$KIT_DIR/frame

#export ORACORE3INCL=/vobs/oracore3/include
#export ORACORE3PUBL=/vobs/oracore3/public
export ORACORE3INCL=$ORACLE_HOME/rdbms/demo
export ORACORE3PUBL=$ORACLE_HOME/rdbms/public
#export RDBMSPUBL=/vobs/rdbms/public
export RDBMSPUBL=$ORACLE_HOME/rdbms/public
#export NETWORKPUBL=/vobs/network_src/public
export NETWORKPUBL=$ORACLE_HOME/network/public
export RDBMSDEMO=$ORACLE_HOME/rdbms/demo
export PLSQLDEMO=$ORACLE_HOME/plsql/demo
export PLSQLPUBL=$ORACLE_HOME/plsql/public
export O=$ORACLE_HOME
export
PATH=./:${BUMPX_DIR}:${UTILS}:${DBGEN}:${MAINT}:${ACI
D_DIR}:${FRAME}/bin:${FRAME}/bin:${REGR_TEST}:${PATH}
#
########### ENVIRONMENT VARIABLES ##################
export WORKLOAD=TPCH
export HOST=
#export OPTLEVEL=X02
export GETOPT=-DSTDLIB_HAS_GETOPT
export PLATFORM=
#export INITORA=$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=5;;
  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
```

A.4 profile

```
stty erase "^H" kill "^x" intr "^C" eof "^D" susp "^z"
export EDITOR=/usr/bin/vi
export ORACLE_HOME=/oracle

#export ORACLE_SID=ASM
#echo 'ORACLE_SID is ASM'

export ORACLE_SID=tpch
echo 'ORACLE_SID is tpch'

#export ORACLE_SID=qual
#echo 'ORACLE_SID is qual'


export KIT_DIR=/dbms/oracle10i/kit

export
SHLIB_PATH=$ORACLE_HOME/lib:$ORACLE_HOME/lib32:$OR
ACLE_HOME/rdbms/lib:$ORACLE_HOME/network/lib
export
LD_LIBRARY_PATH=$ORACLE_HOME/lib:$ORACLE_HOME/lib6
4:$ORACLE_HOME/rdbms/lib:$ORACLE_HOME/network/lib64
export SAVEHIST=2049
export FRAME_PATH=/dbms/oracle10i/frame
export O=$ORACLE_HOME
export ORACLE_PATH=/dbms/oracle10i/frame/tools
export PS1="`whoami`-(`hostname`)> "
export skgxp_trace_path=/tmp/srq.tpch1
export ASYNC_BUF_CONF=256
#export ASYNC_BUF_CONF=6000   #MAX_CONCURRENT in
async.h is 5000
echo "export ASYNC_BUF_CONF=$ASYNC_BUF_CONF"

export
PATH=./:$ORACLE_HOME/bin:/usr/local/bin:$ORACLE_HOME:$O
RACLE_HOME/lib:/opt/perf_tools/bin:/tools:/tpch/run_power:/tpch:/db
ms/oracle10i/frame/bin:/dbms/oracle10i/frame:/dbms/oracle10i/tools/bin
:/tools/Tusc:/dbms/tpcd_v8/bumpx/bumpx:/dbms/tpcd_v8/bumpx/dbgen
:/dbms/tpcd_v8/out/scripts:/opt/ansic/bin:/opt/langtools/bin:/sbin:/usr/sbi
n:.:/bin:/usr/bin:/usr/local/bin:/usr/contrib/bin:/etc:/usr/include:/dbms/ora
cle10i/kit:/dbms/oracle10i/kit/bumpx:/dbms/oracle10i/local/TestIO:/usr/
ccs/bin/:/opt/caliper/bin:/opt/rdma/bin:~/bin

alias ltt="ls -ltr |tail -30"
alias cd_frame="cd /dbms/oracle10i/frame"
alias cd_stats="cd /dbms/oracle10i/frame/stats"
alias cd_q="cd /dbms/oracle10i/frame/queries/queries_tpch"
alias cd_log="cd /oracle/rdbms/log"
alias cd_u="cd /dbms/oracle10i/frame/queries/queries_tpch/updates"
alias ltm="ls -lt |more"
alias cdbin="cd /dbms/tpcd_v8/bin"
alias cdlog="cd /oracle/rdbms/log"
alias cdload="cd /dbms/oracle10i/kit/audit/10tb.ASM"
alias cdtools="cd /dbms/oracle10i/tools/bin"
alias cdq="cd /tpch/tpch/run_power"
alias pso="ps -ef | grep ora | grep -v sleep"
alias pso_hc="ps -fu oracle | sort -n -k2"
alias setterm="TERM=dtterm;export TERM"
alias taillog="tail -f /oracle/rdbms/log/alert_tpch.log"
#alias taillog="tail -f /oracle/rdbms/log/alert_$ORACLE_SID.log"
alias cdlog="cd $ORACLE_HOME/rdbms/log"
umask 002
iosum(){
if [ "$1" -eq "" ] ; then
   echo usage:  iosum iterations
else
 sar -d 5 $1 | ${FRAME_PATH}/bin/io.pl
fi
}
```

A.5 initasm.ora

```
instance_type=asm
shared_pool_size=4G
db_cache_size=2g
asm_diskgroups=DG1
ASM.instance_number=1
instance_number=1
processes=500
ASM_DISKSTRING='/dbms/links/roradsk*'
lock_sga=TRUE
background_dump_dest='/opt/app/admin/ASM/log'
core_dump_dest='/opt/app/admin/ASM/log'
user_dump_dest='/opt/app/admin/ASM/log'
```

Appendix B Build Programs and Scripts

B.1 dbcre.sh

```
#!/bin/ksh

echo START CREATE DB at `date`
export ORACLE_SID=tpch

sqlplus  /NOLOG <<!
connect / as sysdba
set timing on
set echo on

shutdown abort;


startup pfile=/oracle/dbs/10TB_init.ora nomount;
create database
  controlfile reuse
  logfile '+DG1' size 120000m reuse,
          '+DG1' size 120000m reuse
  datafile '+DG1' size 5000m reuse
  sysaux datafile '+DG1' size 5000m reuse
  undo tablespace ts_undo1
       datafile  '+DG1' size 32000m reuse
  maxdatafiles 3000
  maxinstances 2
;

set termout off
set echo off
spool /tmp/cat
@?/rdbms/admin/catalog.sql;
@?/rdbms/admin/catparr.sql;
@?/rdbms/admin/catproc.sql;
connect system/manager
@?/sqlplus/admin/pupbld.sql;
@?/rdbms/admin/utlxplan.sql;
spool off
!
echo END CREATE DB at `date`
```

B.2 sctso.sh

```
#!/bin/ksh


echo CREATE TABLESPACES at `date`
export ORACLE_SID=tpch


(( i = 1 ))
while (( i <= 6 ))
do
sqlplus  / as sysdba <<! &
set timing on
set echo on

alter tablespace ts_undo1
add datafile '+DG1' size 128000m reuse;
;
!
(( i = $i + 1 ))
done

sqlplus  / as sysdba <<! &
set timing on
set echo on

--drop tablespace ts_default including contents;
create tablespace ts_default
datafile '+DG1' size 15000m reuse
extent management local autoallocate nologging;
!

sqlplus  / as sysdba <<! &
set timing on
set echo on

--drop tablespace ts_temp including contents;
create temporary tablespace ts_temp
tempfile '+DG1' size 128000m reuse
extent management local
uniform size 5M
;
!

wait

(( i = 1 ))

while (( i <= 82 ))
do

sqlplus  / as sysdba <<! &

set timing on
set echo on
alter tablespace ts_temp
  add tempfile '+DG1' size 128000m reuse;
!
(( i = $i + 1 ))
done

wait


(( i = 1 ))
while (( i <= 84 ))
do
sqlplus  / as sysdba <<! &
set timing on
set echo on
--drop tablespace ts_l${i} including contents;
create tablespace ts_l${i}
datafile '+DG1' size 128000m reuse
extent management dictionary
default storage (initial 100m next 100m maxextents unlimited
pctincrease 0)
nologging
;
!
(( i = $i + 1 ))
done

wait

(( i = 1 ))
while (( i <= 84 ))
do
sqlplus  / as sysdba <<! &
set timing on
set echo on
```

```
--drop tablespace ts_o${i} including contents;
create tablespace ts_o${i}
datafile '+DG1' size 32000m reuse
extent management dictionary
default storage (initial 100m next 100m maxextents unlimited
pctincrease 0)
nologging
;
!
(( i= $i + 1 ))
done

wait

sqlplus  / as sysdba <<! &
set timing on
set echo on
--drop tablespace ts_c including contents;
create bigfile tablespace ts_c
datafile '+DG1' size 280000m  reuse
extent management local autoallocate nologging;
!

sqlplus  / as sysdba <<! &
set timing on
set echo on

--drop tablespace ts_p including contents;
create bigfile tablespace ts_p
datafile '+DG1' size 280000m reuse
extent management local autoallocate nologging;
!

sqlplus  / as sysdba <<! &
set timing on
set echo on

--drop tablespace ts_okey including contents;
create bigfile tablespace ts_okey
datafile '+DG1' size 400000m reuse
extent management local autoallocate nologging;
!


sqlplus  / as sysdba <<! &
set timing on
set echo on
--drop tablespace ts_s including contents;
create tablespace ts_s
datafile '+DG1' size 18000M reuse
extent management local autoallocate nologging;
;
!

sqlplus  / as sysdba <<! &
set timing on
set echo on

--drop tablespace ts_custkey including contents;
create tablespace ts_custkey
datafile '+DG1' size 50000m reuse
extent management local autoallocate nologging;
!

sqlplus  / as sysdba <<! &
set timing on
set echo on

--drop tablespace ts_lokey including contents;
create tablespace ts_lokey
datafile '+DG1' size 128000m reuse
extent management local autoallocate nologging;
!

sqlplus  / as sysdba <<! &
set timing on
set echo on

--drop tablespace ts_psupp including contents;
create tablespace ts_psupp
datafile '+DG1' size 128000m reuse
extent management dictionary
default storage (initial 500m next 500m maxextents unlimited
pctincrease 0)
nologging
;
!
wait

(( i = 1 ))

while (( i <= 11 ))
do
sqlplus  / as sysdba <<! &
set timing on
set echo on

alter tablespace ts_psupp
add datafile '+DG1' size 128000m reuse;
!

(( i = $i + 1 ))
done


wait

(( i = 1 ))

while (( i <= 14 ))
do
sqlplus  / as sysdba <<! &
set timing on
set echo on

alter tablespace ts_lokey
add datafile '+DG1' size 128000m reuse;
!

(( i = $i + 1 ))
done


wait


echo END CREATE TABLESPACES at `date`
```

B.3 dapop.sh

```
#!/bin/ksh

echo START TABLE CREATION at `date`
export ORACLE_SID=tpch;

/dbms/oracle10i/frame/bin/tshut
/dbms/oracle10i/frame/bin/tshut.asm
/dbms/oracle10i/frame/bin/tstart.asm
/dbms/oracle10i/frame/bin/tstart
```

```
sqlplus /NOLOG <<!
connect / 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;

create directory data_dir1 as '/flat1/';
create directory data_dir2 as '/flat2/';
create directory data_dir3 as '/flat3/';
create directory data_dir4 as '/flat4/';
create directory data_dir5 as '/flat5/';
create directory data_dir6 as '/flat6/';
create directory data_dir7 as '/flat7/';
create directory data_dir8 as '/flat8/';
create directory data_dir9 as '/flat9/';
create directory data_dir10 as '/flat10/';
create directory data_dir11 as '/flat11/';
create directory data_dir12 as '/flat12/';
create directory data_dir13 as '/flat13/';
create directory data_dir14 as '/flat14/';
create directory data_dir15 as '/flat15/';
create directory data_dir16 as '/flat16/';



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
        nobadfile
        nologfile
                fields terminated by '|'
                missing field values are null
         )
         location (
         data_dir1:'lineitem.tbl.1',
         data_dir1:'lineitem.tbl.2',
         data_dir1:'lineitem.tbl.3',
         data_dir1:'lineitem.tbl.4',
         data_dir1:'lineitem.tbl.5',
         data_dir2:'lineitem.tbl.6',
         data_dir2:'lineitem.tbl.7',
         data_dir2:'lineitem.tbl.8',
         data_dir2:'lineitem.tbl.9',
         data_dir2:'lineitem.tbl.10',
         data_dir3:'lineitem.tbl.11',
         data_dir3:'lineitem.tbl.12',
         data_dir3:'lineitem.tbl.13',
         data_dir3:'lineitem.tbl.14',
         data_dir3:'lineitem.tbl.15',
         data_dir4:'lineitem.tbl.16',
         data_dir4:'lineitem.tbl.17',
         data_dir4:'lineitem.tbl.18',
         data_dir4:'lineitem.tbl.19',
         data_dir4:'lineitem.tbl.20',
         data_dir5:'lineitem.tbl.21',
         data_dir5:'lineitem.tbl.22',
         data_dir5:'lineitem.tbl.23',
         data_dir5:'lineitem.tbl.24',
         data_dir5:'lineitem.tbl.25',
         data_dir6:'lineitem.tbl.26',
         data_dir6:'lineitem.tbl.27',
         data_dir6:'lineitem.tbl.28',
         data_dir6:'lineitem.tbl.29',
         data_dir6:'lineitem.tbl.30',
         data_dir7:'lineitem.tbl.31',
         data_dir7:'lineitem.tbl.32',
         data_dir7:'lineitem.tbl.33',
         data_dir7:'lineitem.tbl.34',
         data_dir7:'lineitem.tbl.35',
         data_dir8:'lineitem.tbl.36',
         data_dir8:'lineitem.tbl.37',
         data_dir8:'lineitem.tbl.38',
         data_dir8:'lineitem.tbl.39',
         data_dir8:'lineitem.tbl.40',
         data_dir9:'lineitem.tbl.41',
         data_dir9:'lineitem.tbl.42',
         data_dir9:'lineitem.tbl.43',
         data_dir9:'lineitem.tbl.44',
         data_dir9:'lineitem.tbl.45',
         data_dir10:'lineitem.tbl.46',
         data_dir10:'lineitem.tbl.47',
         data_dir10:'lineitem.tbl.48',
         data_dir10:'lineitem.tbl.49',
         data_dir10:'lineitem.tbl.50',
         data_dir11:'lineitem.tbl.51',
         data_dir11:'lineitem.tbl.52',
         data_dir11:'lineitem.tbl.53',
         data_dir11:'lineitem.tbl.54',
         data_dir11:'lineitem.tbl.55',
         data_dir12:'lineitem.tbl.56',
         data_dir12:'lineitem.tbl.57',
         data_dir12:'lineitem.tbl.58',
         data_dir12:'lineitem.tbl.59',
         data_dir12:'lineitem.tbl.60',
         data_dir13:'lineitem.tbl.61',
```

```
            data_dir13:'lineitem.tbl.62',
            data_dir13:'lineitem.tbl.63',
            data_dir13:'lineitem.tbl.64',
            data_dir13:'lineitem.tbl.65',
            data_dir14:'lineitem.tbl.66',
            data_dir14:'lineitem.tbl.67',
            data_dir14:'lineitem.tbl.68',
            data_dir14:'lineitem.tbl.69',
            data_dir14:'lineitem.tbl.70',
            data_dir15:'lineitem.tbl.71',
            data_dir15:'lineitem.tbl.72',
            data_dir15:'lineitem.tbl.73',
            data_dir15:'lineitem.tbl.74',
            data_dir15:'lineitem.tbl.75',
            data_dir16:'lineitem.tbl.76',
            data_dir16:'lineitem.tbl.77',
            data_dir16:'lineitem.tbl.78',
            data_dir16:'lineitem.tbl.79',
            data_dir16:'lineitem.tbl.80',
            data_dir1:'lineitem.tbl.81',
            data_dir2:'lineitem.tbl.82',
            data_dir3:'lineitem.tbl.83',
            data_dir4:'lineitem.tbl.84'
))
reject limit unlimited parallel;


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
          nobadfile
          nologfile
                    fields terminated by '|'
                    missing field values are null
           )
           location (
      data_dir1:'orders.tbl.1',
      data_dir1:'orders.tbl.2',
      data_dir1:'orders.tbl.3',
      data_dir1:'orders.tbl.4',
      data_dir1:'orders.tbl.5',
      data_dir2:'orders.tbl.6',
      data_dir2:'orders.tbl.7',
      data_dir2:'orders.tbl.8',
      data_dir2:'orders.tbl.9',
      data_dir2:'orders.tbl.10',
      data_dir3:'orders.tbl.11',
      data_dir3:'orders.tbl.12',
      data_dir3:'orders.tbl.13',
      data_dir3:'orders.tbl.14',
      data_dir3:'orders.tbl.15',
      data_dir4:'orders.tbl.16',
      data_dir4:'orders.tbl.17',
      data_dir4:'orders.tbl.18',
      data_dir4:'orders.tbl.19',
      data_dir4:'orders.tbl.20',
      data_dir5:'orders.tbl.21',
      data_dir5:'orders.tbl.22',
      data_dir5:'orders.tbl.23',
      data_dir5:'orders.tbl.24',
      data_dir5:'orders.tbl.25',
      data_dir6:'orders.tbl.26',
      data_dir6:'orders.tbl.27',
      data_dir6:'orders.tbl.28',
      data_dir6:'orders.tbl.29',
      data_dir6:'orders.tbl.30',
      data_dir7:'orders.tbl.31',
      data_dir7:'orders.tbl.32',
      data_dir7:'orders.tbl.33',
      data_dir7:'orders.tbl.34',
      data_dir7:'orders.tbl.35',
      data_dir8:'orders.tbl.36',
      data_dir8:'orders.tbl.37',
      data_dir8:'orders.tbl.38',
      data_dir8:'orders.tbl.39',
      data_dir8:'orders.tbl.40',
      data_dir9:'orders.tbl.41',
      data_dir9:'orders.tbl.42',
      data_dir9:'orders.tbl.43',
      data_dir9:'orders.tbl.44',
      data_dir9:'orders.tbl.45',
      data_dir10:'orders.tbl.46',
      data_dir10:'orders.tbl.47',
      data_dir10:'orders.tbl.48',
      data_dir10:'orders.tbl.49',
      data_dir10:'orders.tbl.50',
      data_dir11:'orders.tbl.51',
      data_dir11:'orders.tbl.52',
      data_dir11:'orders.tbl.53',
      data_dir11:'orders.tbl.54',
      data_dir11:'orders.tbl.55',
      data_dir12:'orders.tbl.56',
      data_dir12:'orders.tbl.57',
      data_dir12:'orders.tbl.58',
      data_dir12:'orders.tbl.59',
      data_dir12:'orders.tbl.60',
      data_dir13:'orders.tbl.61',
      data_dir13:'orders.tbl.62',
      data_dir13:'orders.tbl.63',
      data_dir13:'orders.tbl.64',
      data_dir13:'orders.tbl.65',
      data_dir14:'orders.tbl.66',
      data_dir14:'orders.tbl.67',
      data_dir14:'orders.tbl.68',
      data_dir14:'orders.tbl.69',
      data_dir14:'orders.tbl.70',
      data_dir5:'orders.tbl.71',
      data_dir6:'orders.tbl.72',
      data_dir7:'orders.tbl.73',
      data_dir8:'orders.tbl.74',
      data_dir9:'orders.tbl.75',
      data_dir16:'orders.tbl.76',
      data_dir16:'orders.tbl.77',
      data_dir16:'orders.tbl.78',
      data_dir16:'orders.tbl.79',
      data_dir16:'orders.tbl.80',
      data_dir5:'orders.tbl.81',
      data_dir6:'orders.tbl.82',
      data_dir7:'orders.tbl.83',
      data_dir8:'orders.tbl.84'
))
reject limit unlimited parallel;

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
            nobadfile
            nologfile
                        fields terminated by '|'
                        missing field values are null
             )
             location (
      data_dir1:'partsupp.tbl.1',
      data_dir1:'partsupp.tbl.2',
      data_dir1:'partsupp.tbl.3',
      data_dir1:'partsupp.tbl.4',
      data_dir2:'partsupp.tbl.5',
      data_dir2:'partsupp.tbl.6',
      data_dir2:'partsupp.tbl.7',
      data_dir2:'partsupp.tbl.8',
      data_dir3:'partsupp.tbl.9',
      data_dir3:'partsupp.tbl.10',
      data_dir3:'partsupp.tbl.11',
      data_dir3:'partsupp.tbl.12',
      data_dir4:'partsupp.tbl.13',
      data_dir4:'partsupp.tbl.14',
      data_dir4:'partsupp.tbl.15',
      data_dir4:'partsupp.tbl.16',
      data_dir5:'partsupp.tbl.17',
      data_dir5:'partsupp.tbl.18',
      data_dir5:'partsupp.tbl.19',
      data_dir5:'partsupp.tbl.20',
      data_dir6:'partsupp.tbl.21',
      data_dir6:'partsupp.tbl.22',
      data_dir6:'partsupp.tbl.23',
      data_dir6:'partsupp.tbl.24',
      data_dir7:'partsupp.tbl.25',
      data_dir7:'partsupp.tbl.26',
      data_dir7:'partsupp.tbl.27',
      data_dir7:'partsupp.tbl.28',
      data_dir8:'partsupp.tbl.29',
      data_dir8:'partsupp.tbl.30',
      data_dir8:'partsupp.tbl.31',
      data_dir8:'partsupp.tbl.32',
      data_dir9:'partsupp.tbl.33',
      data_dir9:'partsupp.tbl.34',
      data_dir9:'partsupp.tbl.35',
      data_dir9:'partsupp.tbl.36',
      data_dir10:'partsupp.tbl.37',
      data_dir10:'partsupp.tbl.38',
      data_dir10:'partsupp.tbl.39',
      data_dir10:'partsupp.tbl.40',
      data_dir11:'partsupp.tbl.41',
      data_dir11:'partsupp.tbl.42',
      data_dir11:'partsupp.tbl.43',
      data_dir11:'partsupp.tbl.44',
      data_dir12:'partsupp.tbl.45',
      data_dir12:'partsupp.tbl.46',
      data_dir12:'partsupp.tbl.47',
      data_dir12:'partsupp.tbl.48',
      data_dir13:'partsupp.tbl.49',
      data_dir13:'partsupp.tbl.50',
      data_dir13:'partsupp.tbl.51',
      data_dir13:'partsupp.tbl.52',
      data_dir14:'partsupp.tbl.53',
      data_dir14:'partsupp.tbl.54',
      data_dir14:'partsupp.tbl.55',
      data_dir14:'partsupp.tbl.56',
      data_dir1:'partsupp.tbl.57',
      data_dir2:'partsupp.tbl.58',
      data_dir3:'partsupp.tbl.59',
      data_dir4:'partsupp.tbl.60',
      data_dir16:'partsupp.tbl.61',
      data_dir16:'partsupp.tbl.62',
      data_dir16:'partsupp.tbl.63',
      data_dir16:'partsupp.tbl.64'
))
reject limit unlimited parallel;


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
            nobadfile
            nologfile
                        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'
))
reject limit unlimited parallel;

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
            nobadfile
            nologfile
                        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'
))
reject limit unlimited parallel;

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
            nobadfile
            nologfile
            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'
             ))
reject limit unlimited parallel;

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
            nobadfile
            nologfile
                        fields terminated by '|'
                        missing field values are null
             )
             location (
             data_dir9:'nation.tbl'))
reject limit unlimited;

drop table r_et;
create table r_et(
   r_regionkey        number ,
   r_name             char(25) ,
   r_comment          varchar(152)
)
organization external (
type ORACLE_LOADER
default directory data_dir1
access parameters
(
                        records delimited by newline
            nobadfile
            nologfile
                        fields terminated by '|'
                        missing field values are null
             )
             location (
             data_dir9:'region.tbl'))
reject limit unlimited;


drop table  lineitem;
create table lineitem(
   l_shipdate         ,
   l_orderkey         NOT NULL,
   l_discount         NOT NULL,
   l_extendedprice    NOT NULL,
   l_suppkey          NOT NULL,
   l_quantity         NOT NULL,
   l_returnflag       ,
   l_partkey          NOT NULL,
   l_linestatus       ,
   l_tax              NOT NULL,
   l_commitdate       ,
   l_receiptdate      ,
   l_shipmode         ,
   l_linenumber       NOT NULL,
   l_shipinstruct     ,
   l_comment
)
pctfree 1
pctused 99
initrans 10
storage (freelist groups 4 freelists 84)
parallel
nologging
partition by range (l_shipdate)
subpartition by hash(l_partkey)
subpartitions 128
(
```

```
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_l22
,
partition item23 values less than (to_date('1993-11-01','YYYY-MM-
DD'))
tablespace ts_l23
,
partition item24 values less than (to_date('1993-12-01','YYYY-MM-
DD'))
tablespace ts_l24
,
partition item25 values less than (to_date('1994-01-01','YYYY-MM-
DD'))
tablespace ts_l25
,
partition item26 values less than (to_date('1994-02-01','YYYY-MM-
DD'))
tablespace ts_l26
,
partition item27 values less than (to_date('1994-03-01','YYYY-MM-
DD'))
tablespace ts_l27
,
partition item28 values less than (to_date('1994-04-01','YYYY-MM-
DD'))
tablespace ts_l28
,
partition item29 values less than (to_date('1994-05-01','YYYY-MM-
DD'))
tablespace ts_l29
,
partition item30 values less than (to_date('1994-06-01','YYYY-MM-
DD'))
tablespace ts_l30
,
partition item31 values less than (to_date('1994-07-01','YYYY-MM-
DD'))
tablespace ts_l31
,
partition item32 values less than (to_date('1994-08-01','YYYY-MM-
DD'))
tablespace ts_l32
,
partition item33 values less than (to_date('1994-09-01','YYYY-MM-
DD'))
tablespace ts_l33
,
partition item34 values less than (to_date('1994-10-01','YYYY-MM-
DD'))
tablespace ts_l34
,
partition item35 values less than (to_date('1994-11-01','YYYY-MM-
DD'))
tablespace ts_l35
,
partition item36 values less than (to_date('1994-12-01','YYYY-MM-
DD'))
```

```
tablespace ts_l36
,
partition item37 values less than (to_date('1995-01-01','YYYY-MM-
DD'))
tablespace ts_l37
,
partition item38 values less than (to_date('1995-02-01','YYYY-MM-
DD'))
tablespace ts_l38
,
partition item39 values less than (to_date('1995-03-01','YYYY-MM-
DD'))
tablespace ts_l39
,
partition item40 values less than (to_date('1995-04-01','YYYY-MM-
DD'))
tablespace ts_l40
,
partition item41 values less than (to_date('1995-05-01','YYYY-MM-
DD'))
tablespace ts_l41
,
partition item42 values less than (to_date('1995-06-01','YYYY-MM-
DD'))
tablespace ts_l42
,
partition item43 values less than (to_date('1995-07-01','YYYY-MM-
DD'))
tablespace ts_l43
,
partition item44 values less than (to_date('1995-08-01','YYYY-MM-
DD'))
tablespace ts_l44
,
partition item45 values less than (to_date('1995-09-01','YYYY-MM-
DD'))
tablespace ts_l45
,
partition item46 values less than (to_date('1995-10-01','YYYY-MM-
DD'))
tablespace ts_l46
,
partition item47 values less than (to_date('1995-11-01','YYYY-MM-
DD'))
tablespace ts_l47
,
partition item48 values less than (to_date('1995-12-01','YYYY-MM-
DD'))
tablespace ts_l48
,
partition item49 values less than (to_date('1996-01-01','YYYY-MM-
DD'))
tablespace ts_l49
,
partition item50 values less than (to_date('1996-02-01','YYYY-MM-
DD'))
tablespace ts_l50
,
partition item51 values less than (to_date('1996-03-01','YYYY-MM-
DD'))
tablespace ts_l51
,
partition item52 values less than (to_date('1996-04-01','YYYY-MM-
DD'))
tablespace ts_l52
,
partition item53 values less than (to_date('1996-05-01','YYYY-MM-
DD'))
tablespace ts_l53
,
partition item54 values less than (to_date('1996-06-01','YYYY-MM-
DD'))
tablespace ts_l54
,
partition item55 values less than (to_date('1996-07-01','YYYY-MM-
DD'))
tablespace ts_l55
,
partition item56 values less than (to_date('1996-08-01','YYYY-MM-
DD'))
tablespace ts_l56
,
partition item57 values less than (to_date('1996-09-01','YYYY-MM-
DD'))
tablespace ts_l57
,
partition item58 values less than (to_date('1996-10-01','YYYY-MM-
DD'))
tablespace ts_l58
,
partition item59 values less than (to_date('1996-11-01','YYYY-MM-
DD'))
tablespace ts_l59
,
partition item60 values less than (to_date('1996-12-01','YYYY-MM-
DD'))
tablespace ts_l60
,
partition item61 values less than (to_date('1997-01-01','YYYY-MM-
DD'))
tablespace ts_l61
,
partition item62 values less than (to_date('1997-02-01','YYYY-MM-
DD'))
tablespace ts_l62
,
partition item63 values less than (to_date('1997-03-01','YYYY-MM-
DD'))
tablespace ts_l63
,
partition item64 values less than (to_date('1997-04-01','YYYY-MM-
DD'))
tablespace ts_l64
,
partition item65 values less than (to_date('1997-05-01','YYYY-MM-
DD'))
tablespace ts_l65
,
partition item66 values less than (to_date('1997-06-01','YYYY-MM-
DD'))
tablespace ts_l66
,
partition item67 values less than (to_date('1997-07-01','YYYY-MM-
DD'))
tablespace ts_l67
,
partition item68 values less than (to_date('1997-08-01','YYYY-MM-
DD'))
tablespace ts_l68
,
partition item69 values less than (to_date('1997-09-01','YYYY-MM-
DD'))
tablespace ts_l69
,
partition item70 values less than (to_date('1997-10-01','YYYY-MM-
DD'))
tablespace ts_l70
,
partition item71 values less than (to_date('1997-11-01','YYYY-MM-
DD'))
tablespace ts_l71
```

```
,
partition item72 values less than (to_date('1997-12-01','YYYY-MM-
DD'))
tablespace ts_l72
,
partition item73 values less than (to_date('1998-01-01','YYYY-MM-
DD'))
tablespace ts_l73
,
partition item74 values less than (to_date('1998-02-01','YYYY-MM-
DD'))
tablespace ts_l74
,
partition item75 values less than (to_date('1998-03-01','YYYY-MM-
DD'))
tablespace ts_l75
,
partition item76 values less than (to_date('1998-04-01','YYYY-MM-
DD'))
tablespace ts_l76
,
partition item77 values less than (to_date('1998-05-01','YYYY-MM-
DD'))
tablespace ts_l77
,
partition item78 values less than (to_date('1998-06-01','YYYY-MM-
DD'))
tablespace ts_l78
,
partition item79 values less than (to_date('1998-07-01','YYYY-MM-
DD'))
tablespace ts_l79
,
partition item80 values less than (to_date('1998-08-01','YYYY-MM-
DD'))
tablespace ts_l80
,
partition item81 values less than (to_date('1998-09-01','YYYY-MM-
DD'))
tablespace ts_l81
,
partition item82 values less than (to_date('1998-10-01','YYYY-MM-
DD'))
tablespace ts_l82
,
partition item83 values less than (to_date('1998-11-01','YYYY-MM-
DD'))
tablespace ts_l83
,
partition item84 values less than (MAXVALUE)
tablespace ts_l84 )
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;

drop table  orders;
create table orders(
   o_orderdate         ,
   o_orderkey          NOT NULL,
   o_custkey           NOT NULL,
   o_orderpriority     ,
   o_shippriority      ,
   o_clerk             ,
   o_orderstatus       ,
   o_totalprice        ,
   o_comment
)
pctfree 1
pctused 99
initrans 10
storage (freelist groups 4 freelists 99)
parallel
nologging
partition by range (o_orderdate)
subpartition by hash(o_custkey)
subpartitions 128
(
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_o22
,
partition ord23 values less than (to_date('1993-11-01','YYYY-MM-
DD'))
tablespace ts_o23
,
partition ord24 values less than (to_date('1993-12-01','YYYY-MM-
DD'))
tablespace ts_o24
,
partition ord25 values less than (to_date('1994-01-01','YYYY-MM-
DD'))
tablespace ts_o25
,
partition ord26 values less than (to_date('1994-02-01','YYYY-MM-
DD'))
tablespace ts_o26
,
partition ord27 values less than (to_date('1994-03-01','YYYY-MM-
DD'))
tablespace ts_o27
,
partition ord28 values less than (to_date('1994-04-01','YYYY-MM-
DD'))
tablespace ts_o28
,
partition ord29 values less than (to_date('1994-05-01','YYYY-MM-
DD'))
tablespace ts_o29
,
partition ord30 values less than (to_date('1994-06-01','YYYY-MM-
DD'))
tablespace ts_o30
,
partition ord31 values less than (to_date('1994-07-01','YYYY-MM-
DD'))
tablespace ts_o31
,
partition ord32 values less than (to_date('1994-08-01','YYYY-MM-
DD'))
tablespace ts_o32
,
partition ord33 values less than (to_date('1994-09-01','YYYY-MM-
DD'))
tablespace ts_o33
,
partition ord34 values less than (to_date('1994-10-01','YYYY-MM-
DD'))
tablespace ts_o34
,
partition ord35 values less than (to_date('1994-11-01','YYYY-MM-
DD'))
tablespace ts_o35
,
partition ord36 values less than (to_date('1994-12-01','YYYY-MM-
DD'))
tablespace ts_o36
,
partition ord37 values less than (to_date('1995-01-01','YYYY-MM-
DD'))
tablespace ts_o37
,
partition ord38 values less than (to_date('1995-02-01','YYYY-MM-
DD'))
tablespace ts_o38
,
partition ord39 values less than (to_date('1995-03-01','YYYY-MM-
DD'))
tablespace ts_o39
,
partition ord40 values less than (to_date('1995-04-01','YYYY-MM-
DD'))
tablespace ts_o40
,
partition ord41 values less than (to_date('1995-05-01','YYYY-MM-
DD'))
tablespace ts_o41
,
partition ord42 values less than (to_date('1995-06-01','YYYY-MM-
DD'))
tablespace ts_o42
,
partition ord43 values less than (to_date('1995-07-01','YYYY-MM-
DD'))
tablespace ts_o43
,
partition ord44 values less than (to_date('1995-08-01','YYYY-MM-
DD'))
tablespace ts_o44
,
partition ord45 values less than (to_date('1995-09-01','YYYY-MM-
DD'))
tablespace ts_o45
,
partition ord46 values less than (to_date('1995-10-01','YYYY-MM-
DD'))
tablespace ts_o46
,
partition ord47 values less than (to_date('1995-11-01','YYYY-MM-
DD'))
tablespace ts_o47
,
partition ord48 values less than (to_date('1995-12-01','YYYY-MM-
DD'))
tablespace ts_o48
,
partition ord49 values less than (to_date('1996-01-01','YYYY-MM-
DD'))
tablespace ts_o49
,
partition ord50 values less than (to_date('1996-02-01','YYYY-MM-
DD'))
tablespace ts_o50
```

```
,
partition ord51 values less than (to_date('1996-03-01','YYYY-MM-
DD'))
tablespace ts_o51
,
partition ord52 values less than (to_date('1996-04-01','YYYY-MM-
DD'))
tablespace ts_o52
,
partition ord53 values less than (to_date('1996-05-01','YYYY-MM-
DD'))
tablespace ts_o53
,
partition ord54 values less than (to_date('1996-06-01','YYYY-MM-
DD'))
tablespace ts_o54
,
partition ord55 values less than (to_date('1996-07-01','YYYY-MM-
DD'))
tablespace ts_o55
,
partition ord56 values less than (to_date('1996-08-01','YYYY-MM-
DD'))
tablespace ts_o56
,
partition ord57 values less than (to_date('1996-09-01','YYYY-MM-
DD'))
tablespace ts_o57
,
partition ord58 values less than (to_date('1996-10-01','YYYY-MM-
DD'))
tablespace ts_o58
,
partition ord59 values less than (to_date('1996-11-01','YYYY-MM-
DD'))
tablespace ts_o59
,
partition ord60 values less than (to_date('1996-12-01','YYYY-MM-
DD'))
tablespace ts_o60
,
partition ord61 values less than (to_date('1997-01-01','YYYY-MM-
DD'))
tablespace ts_o61
,
partition ord62 values less than (to_date('1997-02-01','YYYY-MM-
DD'))
tablespace ts_o62
,
partition ord63 values less than (to_date('1997-03-01','YYYY-MM-
DD'))
tablespace ts_o63
,
partition ord64 values less than (to_date('1997-04-01','YYYY-MM-
DD'))
tablespace ts_o64
,
partition ord65 values less than (to_date('1997-05-01','YYYY-MM-
DD'))
tablespace ts_o65
,
partition ord66 values less than (to_date('1997-06-01','YYYY-MM-
DD'))
tablespace ts_o66
,
partition ord67 values less than (to_date('1997-07-01','YYYY-MM-
DD'))
tablespace ts_o67
,
partition ord68 values less than (to_date('1997-08-01','YYYY-MM-
DD'))
tablespace ts_o68
,
partition ord69 values less than (to_date('1997-09-01','YYYY-MM-
DD'))
tablespace ts_o69
,
partition ord70 values less than (to_date('1997-10-01','YYYY-MM-
DD'))
tablespace ts_o70
,
partition ord71 values less than (to_date('1997-11-01','YYYY-MM-
DD'))
tablespace ts_o71
,
partition ord72 values less than (to_date('1997-12-01','YYYY-MM-
DD'))
tablespace ts_o72
,
partition ord73 values less than (to_date('1998-01-01','YYYY-MM-
DD'))
tablespace ts_o73
,
partition ord74 values less than (to_date('1998-02-01','YYYY-MM-
DD'))
tablespace ts_o74
,
partition ord75 values less than (to_date('1998-03-01','YYYY-MM-
DD'))
tablespace ts_o75
,
partition ord76 values less than (to_date('1998-04-01','YYYY-MM-
DD'))
tablespace ts_o76
,
partition ord77 values less than (to_date('1998-05-01','YYYY-MM-
DD'))
tablespace ts_o77
,
partition ord78 values less than (to_date('1998-06-01','YYYY-MM-
DD'))
tablespace ts_o78
,
partition ord79 values less than (to_date('1998-07-01','YYYY-MM-
DD'))
tablespace ts_o79
,
partition ord80 values less than (to_date('1998-08-01','YYYY-MM-
DD'))
tablespace ts_o80
,
partition ord81 values less than (to_date('1998-09-01','YYYY-MM-
DD'))
tablespace ts_o81
,
partition ord82 values less than (to_date('1998-10-01','YYYY-MM-
DD'))
tablespace ts_o82
,
partition ord83 values less than (to_date('1998-11-01','YYYY-MM-
DD'))
tablespace ts_o83
,
partition ord84 values less than (MAXVALUE)
tablespace ts_o84 )
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;


drop table  partsupp;
create table partsupp(
   ps_partkey          NOT NULL,
   ps_suppkey          NOT NULL,
   ps_supplycost       NOT NULL,
   ps_availqty         ,
   ps_comment
)
parallel
nologging
partition by hash(ps_partkey)
partitions 128
tablespace ts_psupp
as select
   ps_partkey          ,
   ps_suppkey          ,
   ps_supplycost       ,
   ps_availqty         ,
   ps_comment
from ps_et;


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
parallel
nologging
partition by hash (c_custkey)
partitions 128
tablespace ts_c
storage (initial 100m)
as select
   c_custkey           ,
   c_mktsegment        ,
   c_nationkey         ,
   c_name              ,
   c_address           ,
   c_phone             ,
   c_acctbal           ,
   c_comment
from c_et;


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
parallel
nologging
partition by hash (p_partkey)
partitions 128
tablespace ts_p
storage (initial 100m)
as select
   p_partkey           ,
   p_type              ,
   p_size              ,
   p_brand             ,
   p_name              ,
   p_container         ,
   p_mfgr              ,
   p_retailprice       ,
   p_comment
from p_et;


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
parallel
nologging
partition by hash (s_suppkey)
partitions 128
tablespace ts_s
as select
   s_suppkey           ,
   s_nationkey         ,
   s_comment           ,
   s_name              ,
   s_address           ,
   s_phone             ,
   s_acctbal
from s_et;

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

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 DONE TABLE CREATION at `date`
```

B.4 ixcre.sh

```
#!/bin/ksh

echo START INDEX at `date`
export ORACLE_SID=tpch

sqlplus  tpch/tpch <<!
set echo on
set timing on
set termout on

drop index i_l_orderkey;
create index i_l_orderkey
on lineitem (l_orderkey)
global partition by hash (l_orderkey)
partitions 128
pctfree 10
initrans 10
tablespace ts_lokey
storage (freelist groups 4 freelists 99)
parallel
compute statistics
nologging;


drop index i_o_orderkey;
create unique index i_o_orderkey
on orders (o_orderkey)
global partition by hash (o_orderkey)
partitions 128
pctfree 10
initrans 10
tablespace ts_okey
storage (freelist groups 4 freelists 99 )
parallel
compute statistics
nologging;


drop index i_c_custkey;
create unique index i_c_custkey
on customer (c_custkey)
pctfree 2
initrans 10
tablespace ts_custkey
storage (freelists 99)
parallel
compute statistics
nologging;


drop index i_ps_pkey_skey;
create index i_ps_pkey_skey
on partsupp (ps_partkey,ps_suppkey)
global partition by hash (ps_partkey)
partitions 128
pctfree 5
initrans 10
tablespace ts_lokey
storage (freelist groups 4 freelists 99)
parallel
compute statistics
nologging;
!

echo DONE INDEX at `date`
```

B.5 anl.sh

```
#!/bin/ksh

echo START ANALYZE at `date`
export ORACLE_SID=tpch;

sqlplus  tpch/tpch <<!
set timing on
set echo on
set termout on

execute dbms_stats.gather_schema_stats('TPCH' , estimate_percent =>
1, degree => 64 , granularity => 'GLOBAL', method_opt => 'for all
columns size 1' );
connect / as sysdba
execute dbms_stats.gather_system_stats;
exec dbms_scheduler.disable('GATHER_STATS_JOB');
alter system switch logfile;
!

/dbms/oracle10i/frame/bin/tshut
/dbms/oracle10i/frame/bin/tshut.asm
/dbms/oracle10i/frame/bin/tstart.asm
/dbms/oracle10i/frame/bin/tstart


echo END ANALYZE at `date`
```

B.6 loadasm

```
#!/bin/ksh

echo START LOADASM at `date`
export ORACLE_SID=ASM

sqlplus /NOLOG <<!
connect / as sysdba;
shutdown abort;
startup pfile=/oracle/dbs/initasm.ora ;
alter diskgroup all mount;
drop diskgroup dg1 including contents;
create diskgroup dg1 External REDUNDANCY
DISK
'/dbms/links/roradsk1' SIZE 143353M,
'/dbms/links/roradsk2' SIZE 143353M,
'/dbms/links/roradsk3' SIZE 143353M,
'/dbms/links/roradsk4' SIZE 143353M,
'/dbms/links/roradsk5' SIZE 143353M,
'/dbms/links/roradsk6' SIZE 143353M,
'/dbms/links/roradsk7' SIZE 143353M,
'/dbms/links/roradsk8' SIZE 143353M,
'/dbms/links/roradsk9' SIZE 143353M,
'/dbms/links/roradsk10' SIZE 143353M,
```

'/dbms/links/roradsk11' SIZE 143353M,
'/dbms/links/roradsk12' SIZE 143353M,
'/dbms/links/roradsk13' SIZE 143353M,
'/dbms/links/roradsk14' SIZE 143353M,
'/dbms/links/roradsk15' SIZE 143353M,
'/dbms/links/roradsk16' SIZE 143353M,
'/dbms/links/roradsk17' SIZE 143353M,
'/dbms/links/roradsk18' SIZE 143353M,
'/dbms/links/roradsk19' SIZE 143353M,
'/dbms/links/roradsk20' SIZE 143353M,
'/dbms/links/roradsk21' SIZE 143353M,
'/dbms/links/roradsk22' SIZE 143353M,
'/dbms/links/roradsk23' SIZE 143353M,
'/dbms/links/roradsk24' SIZE 143353M,
'/dbms/links/roradsk25' SIZE 143353M,
'/dbms/links/roradsk26' SIZE 143353M,
'/dbms/links/roradsk27' SIZE 143353M,
'/dbms/links/roradsk28' SIZE 143353M,
'/dbms/links/roradsk29' SIZE 143353M,
'/dbms/links/roradsk30' SIZE 143353M,
'/dbms/links/roradsk31' SIZE 143353M,
'/dbms/links/roradsk32' SIZE 143353M,
'/dbms/links/roradsk33' SIZE 143353M,
'/dbms/links/roradsk34' SIZE 143353M,
'/dbms/links/roradsk35' SIZE 143353M,
'/dbms/links/roradsk36' SIZE 143353M,
'/dbms/links/roradsk37' SIZE 143353M,
'/dbms/links/roradsk38' SIZE 143353M,
'/dbms/links/roradsk39' SIZE 143353M,
'/dbms/links/roradsk40' SIZE 143353M,
'/dbms/links/roradsk41' SIZE 143353M,
'/dbms/links/roradsk42' SIZE 143353M,
'/dbms/links/roradsk43' SIZE 143353M,
'/dbms/links/roradsk44' SIZE 143353M,
'/dbms/links/roradsk45' SIZE 143353M,
'/dbms/links/roradsk46' SIZE 143353M,
'/dbms/links/roradsk47' SIZE 143353M,
'/dbms/links/roradsk48' SIZE 143353M,
'/dbms/links/roradsk49' SIZE 143353M,
'/dbms/links/roradsk50' SIZE 143353M,
'/dbms/links/roradsk51' SIZE 143353M,
'/dbms/links/roradsk52' SIZE 143353M,
'/dbms/links/roradsk53' SIZE 143353M,
'/dbms/links/roradsk54' SIZE 143353M,
'/dbms/links/roradsk55' SIZE 143353M,
'/dbms/links/roradsk56' SIZE 143353M,
'/dbms/links/roradsk57' SIZE 143353M,
'/dbms/links/roradsk58' SIZE 143353M,
'/dbms/links/roradsk59' SIZE 143353M,
'/dbms/links/roradsk60' SIZE 143353M,
'/dbms/links/roradsk61' SIZE 143353M,
'/dbms/links/roradsk62' SIZE 143353M,
'/dbms/links/roradsk63' SIZE 143353M,
'/dbms/links/roradsk64' SIZE 143353M,
'/dbms/links/roradsk65' SIZE 143353M,
'/dbms/links/roradsk66' SIZE 143353M,
'/dbms/links/roradsk67' SIZE 143353M,
'/dbms/links/roradsk68' SIZE 143353M,
'/dbms/links/roradsk69' SIZE 143353M,
'/dbms/links/roradsk70' SIZE 143353M,
'/dbms/links/roradsk71' SIZE 143353M,
'/dbms/links/roradsk72' SIZE 143353M,
'/dbms/links/roradsk73' SIZE 143353M,
'/dbms/links/roradsk74' SIZE 143353M,
'/dbms/links/roradsk75' SIZE 143353M,
'/dbms/links/roradsk76' SIZE 143353M,
'/dbms/links/roradsk77' SIZE 143353M,
'/dbms/links/roradsk78' SIZE 143353M,
'/dbms/links/roradsk79' SIZE 143353M,
'/dbms/links/roradsk80' SIZE 143353M,
'/dbms/links/roradsk81' SIZE 143353M,
'/dbms/links/roradsk82' SIZE 143353M,
'/dbms/links/roradsk83' SIZE 143353M,
'/dbms/links/roradsk84' SIZE 143353M,
'/dbms/links/roradsk85' SIZE 143353M,
'/dbms/links/roradsk86' SIZE 143353M,
'/dbms/links/roradsk87' SIZE 143353M,
'/dbms/links/roradsk88' SIZE 143353M,
'/dbms/links/roradsk89' SIZE 143353M,
'/dbms/links/roradsk90' SIZE 143353M,
'/dbms/links/roradsk91' SIZE 143353M,
'/dbms/links/roradsk92' SIZE 143353M,
'/dbms/links/roradsk93' SIZE 143353M,
'/dbms/links/roradsk94' SIZE 143353M,
'/dbms/links/roradsk95' SIZE 143353M,
'/dbms/links/roradsk96' SIZE 143353M,
'/dbms/links/roradsk97' SIZE 143353M,
'/dbms/links/roradsk98' SIZE 143353M,
'/dbms/links/roradsk99' SIZE 143353M,
'/dbms/links/roradsk100' SIZE 143353M,
'/dbms/links/roradsk101' SIZE 143353M,
'/dbms/links/roradsk102' SIZE 143353M,
'/dbms/links/roradsk103' SIZE 143353M,
'/dbms/links/roradsk104' SIZE 143353M,
'/dbms/links/roradsk105' SIZE 143353M,
'/dbms/links/roradsk106' SIZE 143353M,
'/dbms/links/roradsk107' SIZE 143353M,
'/dbms/links/roradsk108' SIZE 143353M,
'/dbms/links/roradsk109' SIZE 143353M,
'/dbms/links/roradsk110' SIZE 143353M,
'/dbms/links/roradsk111' SIZE 143353M,
'/dbms/links/roradsk112' SIZE 143353M,
'/dbms/links/roradsk113' SIZE 143353M,
'/dbms/links/roradsk114' SIZE 143353M,
'/dbms/links/roradsk115' SIZE 143353M,
'/dbms/links/roradsk116' SIZE 143353M,
'/dbms/links/roradsk117' SIZE 143353M,
'/dbms/links/roradsk118' SIZE 143353M,
'/dbms/links/roradsk119' SIZE 143353M,
'/dbms/links/roradsk120' SIZE 143353M,
'/dbms/links/roradsk121' SIZE 143353M,
'/dbms/links/roradsk122' SIZE 143353M,
'/dbms/links/roradsk123' SIZE 143353M,
'/dbms/links/roradsk124' SIZE 143353M,
'/dbms/links/roradsk125' SIZE 143353M,
'/dbms/links/roradsk126' SIZE 143353M,
'/dbms/links/roradsk127' SIZE 143353M,
'/dbms/links/roradsk128' SIZE 143353M,
'/dbms/links/roradsk129' SIZE 143353M,
'/dbms/links/roradsk130' SIZE 143353M,
'/dbms/links/roradsk131' SIZE 143353M,
'/dbms/links/roradsk132' SIZE 143353M,
'/dbms/links/roradsk133' SIZE 143353M,
'/dbms/links/roradsk134' SIZE 143353M,
'/dbms/links/roradsk135' SIZE 143353M,
'/dbms/links/roradsk136' SIZE 143353M,
'/dbms/links/roradsk137' SIZE 143353M,
'/dbms/links/roradsk138' SIZE 143353M,
'/dbms/links/roradsk139' SIZE 143353M,
'/dbms/links/roradsk140' SIZE 143353M,
'/dbms/links/roradsk141' SIZE 143353M,
'/dbms/links/roradsk142' SIZE 143353M,
'/dbms/links/roradsk143' SIZE 143353M,
'/dbms/links/roradsk144' SIZE 143353M,
'/dbms/links/roradsk145' SIZE 143353M,
'/dbms/links/roradsk146' SIZE 143353M,
'/dbms/links/roradsk147' SIZE 143353M,
'/dbms/links/roradsk148' SIZE 143353M,
'/dbms/links/roradsk149' SIZE 143353M,
'/dbms/links/roradsk150' SIZE 143353M,
'/dbms/links/roradsk151' SIZE 143353M,
'/dbms/links/roradsk152' SIZE 143353M,

```
'/dbms/links/roradsk153' SIZE 143353M,
'/dbms/links/roradsk154' SIZE 143353M,
'/dbms/links/roradsk155' SIZE 143353M,
'/dbms/links/roradsk156' SIZE 143353M,
'/dbms/links/roradsk157' SIZE 143353M,
'/dbms/links/roradsk158' SIZE 143353M,
'/dbms/links/roradsk159' SIZE 143353M,
'/dbms/links/roradsk160' SIZE 143353M,
'/dbms/links/roradsk161' SIZE 143353M,
'/dbms/links/roradsk162' SIZE 143353M,
'/dbms/links/roradsk163' SIZE 143353M,
'/dbms/links/roradsk164' SIZE 143353M,
'/dbms/links/roradsk165' SIZE 143353M,
'/dbms/links/roradsk166' SIZE 143353M,
'/dbms/links/roradsk167' SIZE 143353M,
'/dbms/links/roradsk168' SIZE 143353M,
'/dbms/links/roradsk169' SIZE 143353M,
'/dbms/links/roradsk170' SIZE 143353M,
'/dbms/links/roradsk171' SIZE 143353M,
'/dbms/links/roradsk172' SIZE 143353M,
'/dbms/links/roradsk173' SIZE 143353M,
'/dbms/links/roradsk174' SIZE 143353M,
'/dbms/links/roradsk175' SIZE 143353M,
'/dbms/links/roradsk176' SIZE 143353M,
'/dbms/links/roradsk177' SIZE 143353M,
'/dbms/links/roradsk178' SIZE 143353M,
'/dbms/links/roradsk179' SIZE 143353M,
'/dbms/links/roradsk180' SIZE 143353M,
'/dbms/links/roradsk181' SIZE 143353M,
'/dbms/links/roradsk182' SIZE 143353M,
'/dbms/links/roradsk183' SIZE 143353M,
'/dbms/links/roradsk184' SIZE 143353M,
'/dbms/links/roradsk185' SIZE 143353M,
'/dbms/links/roradsk186' SIZE 143353M,
'/dbms/links/roradsk187' SIZE 143353M,
'/dbms/links/roradsk188' SIZE 143353M,
'/dbms/links/roradsk189' SIZE 143353M,
'/dbms/links/roradsk190' SIZE 143353M,
'/dbms/links/roradsk191' SIZE 143353M,
'/dbms/links/roradsk192' SIZE 143353M,
'/dbms/links/roradsk193' SIZE 143353M,
'/dbms/links/roradsk194' SIZE 143353M,
'/dbms/links/roradsk195' SIZE 143353M,
'/dbms/links/roradsk196' SIZE 143353M,
'/dbms/links/roradsk197' SIZE 143353M,
'/dbms/links/roradsk198' SIZE 143353M,
'/dbms/links/roradsk199' SIZE 143353M,
'/dbms/links/roradsk200' SIZE 143353M,
'/dbms/links/roradsk201' SIZE 143353M,
'/dbms/links/roradsk202' SIZE 143353M,
'/dbms/links/roradsk203' SIZE 143353M,
'/dbms/links/roradsk204' SIZE 143353M,
'/dbms/links/roradsk205' SIZE 143353M,
'/dbms/links/roradsk206' SIZE 143353M,
'/dbms/links/roradsk207' SIZE 143353M,
'/dbms/links/roradsk208' SIZE 143353M,
'/dbms/links/roradsk209' SIZE 143353M,
'/dbms/links/roradsk210' SIZE 143353M,
'/dbms/links/roradsk211' SIZE 143353M,
'/dbms/links/roradsk212' SIZE 143353M,
'/dbms/links/roradsk213' SIZE 143353M,
'/dbms/links/roradsk214' SIZE 143353M,
'/dbms/links/roradsk215' SIZE 143353M,
'/dbms/links/roradsk216' SIZE 143353M,
'/dbms/links/roradsk217' SIZE 143353M,
'/dbms/links/roradsk218' SIZE 143353M,
'/dbms/links/roradsk219' SIZE 143353M,
'/dbms/links/roradsk220' SIZE 143353M,
'/dbms/links/roradsk221' SIZE 143353M,
'/dbms/links/roradsk222' SIZE 143353M,
'/dbms/links/roradsk223' SIZE 143353M,
'/dbms/links/roradsk224' SIZE 143353M,
'/dbms/links/roradsk225' SIZE 143353M,
'/dbms/links/roradsk226' SIZE 143353M,
'/dbms/links/roradsk227' SIZE 143353M,
'/dbms/links/roradsk228' SIZE 143353M,
'/dbms/links/roradsk229' SIZE 143353M,
'/dbms/links/roradsk230' SIZE 143353M,
'/dbms/links/roradsk231' SIZE 143353M,
'/dbms/links/roradsk232' SIZE 143353M,
'/dbms/links/roradsk233' SIZE 143353M,
'/dbms/links/roradsk234' SIZE 143353M,
'/dbms/links/roradsk235' SIZE 143353M,
'/dbms/links/roradsk236' SIZE 143353M,
'/dbms/links/roradsk237' SIZE 143353M,
'/dbms/links/roradsk238' SIZE 143353M,
'/dbms/links/roradsk239' SIZE 143353M,
'/dbms/links/roradsk240' SIZE 143353M,
'/dbms/links/roradsk241' SIZE 143353M,
'/dbms/links/roradsk242' SIZE 143353M,
'/dbms/links/roradsk243' SIZE 143353M,
'/dbms/links/roradsk244' SIZE 143353M,
'/dbms/links/roradsk245' SIZE 143353M,
'/dbms/links/roradsk246' SIZE 143353M,
'/dbms/links/roradsk247' SIZE 143353M,
'/dbms/links/roradsk248' SIZE 143353M,
'/dbms/links/roradsk249' SIZE 143353M,
'/dbms/links/roradsk250' SIZE 143353M,
'/dbms/links/roradsk251' SIZE 143353M,
'/dbms/links/roradsk252' SIZE 143353M,
'/dbms/links/roradsk253' SIZE 143353M,
'/dbms/links/roradsk254' SIZE 143353M,
'/dbms/links/roradsk255' SIZE 143353M,
'/dbms/links/roradsk256' SIZE 143353M;
alter diskgroup dg1 rebalance power 0;
!

sqlplus /NOLOG <<!
connect / as sysdba;
shutdown normal;
!

export ORACLE_SID=ASM
sqlplus /NOLOG <<!
connect / as sysdba
startup pfile=/oracle/dbs/initasm.ora mount
!
echo END LOADASM at `date`
```

Appendix C Acid Scripts

C.1 a_query.sql

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

set serverout on;

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

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

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

exit;
```

C.2 a_query2.sql

```
Rem
Rem $Header: aquery2.sql 07-aug-99.23:54:47 mpoess Exp $
Rem
Rem aquery2.sql
Rem
Rem  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
Rem
Rem    NAME
Rem      aquery2.sql - <one-line expansion of the name>
Rem
Rem  DESCRIPTION
Rem       Performs query on PARTSUPP for TPC-D benchmark
Rem       Isolation Test 5.
Rem       Asks user to input values for ps_partkey and ps_suppkey
Rem       The range for ps_partkey is 1 to 20000
Rem       The range for ps_suppkey is 1 to 1000
Rem       A valid combination is 46 and 47
Rem  Usage:   sqlplus tpcd/tpcd @a_query2 <ps_partkey>
<ps_suppkey>
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    mpoess      08/07/99 - Creation
Rem    mpoess      08/07/99 - Created
Rem
rem  DESCRIPTION
rem       Performs query on PARTSUPP for TPC-D benchmark
rem       Isolation Test 5.
rem       Asks user to input values for ps_partkey and ps_suppkey
rem       The range for ps_partkey is 1 to 20000
rem       The range for ps_suppkey is 1 to 1000
rem       A valid combination is 46 and 47

set serverout on;

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

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

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

exit;
```

C.3 atom.sh

```
#!/bin/ksh
#
# $Header: atom.sh 08-aug-99.13:48:02 mpoess Exp $
#
# atom.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#    NAME
#      atom.sh - <one-line expansion of the name>
#
#  DESCRIPTION
#      Performs atomicity tests.
#      Usage: atom.sh [-n iter] [-p prog] [-u usr/pswd] -h
#
#      Options: See usage below
#
#    NOTES
#      <other useful comments, qualifications, etc.>
#
#    MODIFIED   (MM/DD/YY)
#    mpoess      08/08/99 - Creation
#    mpoess      08/08/99 - Creation
#

. $KIT_DIR/env
```

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

usage() {

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

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

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

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

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

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

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

$KIT_DIR/utils/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`..."
```

C.4 atrans.sql

```
Rem
Rem $Header: atrans.sql 07-aug-99.21:27:13 mpoess Exp $
Rem
Rem atrans.sql
Rem
Rem  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
Rem
Rem    NAME
Rem      atrans.sql - <one-line expansion of the name>
Rem
Rem   DESCRIPTION
Rem       Creates ACID Transaction Package for TPC-D benchmark.
Rem       Asks user to input values for o_key, delta and output file.
Rem
Rem    NOTES
Rem      <other useful comments, qualifications, etc.>
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    mpoess      08/07/99 - Creation
Rem    mpoess      08/07/99 - Created
Rem



set serverout on;
set termout on;
set echo on;

CREATE OR REPLACE PACKAGE d_atrans
IS
PROCEDURE doatrans
(
                l_key                   IN OUT integer,
                o_key                   IN OUT integer,
                delta                   IN OUT integer,
                l_pkey          IN OUT integer,
                l_skey          IN OUT integer,
                l_quan          IN OUT integer,
                l_newquan       IN OUT integer,
                l_tax           IN OUT number,
                l_disc          IN OUT number,
                l_eprice        IN OUT number,
                l_neweprice     IN OUT number,
                o_tprice     IN OUT number,
                o_newtprice             IN OUT number,
                rprice                  IN OUT number,
                cost                    IN OUT number
);
END;
/

CREATE OR REPLACE PACKAGE BODY d_atrans
IS
PROCEDURE doatrans
(
                l_key                   IN OUT integer,
                o_key                   IN OUT integer,
                delta                   IN OUT integer,
                l_pkey          IN OUT integer,
                l_skey          IN OUT integer,
                l_quan          IN OUT integer,
                l_newquan       IN OUT integer,
                l_tax           IN OUT number,
                l_disc          IN OUT number,
                l_eprice        IN OUT number,
                l_neweprice     IN OUT number,
                o_tprice     IN OUT number,
                o_newtprice             IN OUT number,
                rprice                  IN OUT number,
                cost                    IN OUT number
)
IS
        ototal number;
        not_serializable EXCEPTION;
        PRAGMA EXCEPTION_INIT(not_serializable,-8177);
BEGIN
  -- EXECUTE IMMEDIATE 'ALTER SESSION SET
ISOLATION_LEVEL = SERIALIZABLE';
  LOOP BEGIN

    select o_totalprice
```

```
            into o_tprice
            from orders
            where o_orderkey = o_key;

   select l_quantity, l_extendedprice, l_partkey, l_suppkey, l_tax,
l_discount
       into l_quan, l_eprice, l_pkey, l_skey, l_tax, l_disc
       from lineitem
       where l_orderkey = o_key
       and   l_linenumber = l_key;

   ototal := o_tprice - trunc((trunc((l_eprice * (1.0-l_disc)),2) *
(1.0+l_tax)),2);
   rprice := trunc((l_eprice/l_quan), 2);
   cost := trunc((rprice * delta), 2);
   l_neweprice := l_eprice + cost;
   o_newtprice := trunc((l_neweprice * (1.0 - l_disc)), 2);
   o_newtprice := ototal + trunc((o_newtprice * (1.0 + l_tax)), 2);
   l_newquan := l_quan + delta;

   update lineitem
      set l_extendedprice = l_neweprice,
      l_quantity = l_newquan
      where l_orderkey = o_key
      and l_linenumber = l_key;

   update orders
      set o_totalprice = o_newtprice
      where o_orderkey = o_key;

    insert into history (h_p_key, h_s_key, h_o_key, h_l_key, h_delta,
h_date_t)
       values (l_pkey, l_skey, o_key, l_key, delta, sysdate);

 -- dbms_lock.sleep(30);
 --   commit;
   EXIT;

  EXCEPTION
    WHEN not_serializable THEN
      ROLLBACK;
  END;

  END LOOP;

END doatrans;
END;
/

exit;
```

C.5 atranspl.c

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

/*

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

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

  NOTES
    <other useful comments, qualifications, etc.>

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

*/

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

#include "atranspl.h"

/* Declare error handling functions */

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

/* declarations for ORDERS */

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

/* declarations for LINEITEM */

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

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

sb2 l_npricei;

/* other declarations */

int delta = 0;
double rprice;
double cost;

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

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

char buf[WRITE_BUF_LEN];  /* buffer to write                    */

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

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

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

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

time_t curr_time;         /* Current Time                      */

/* OCI handles */

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

/* OCI bind handles */

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

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

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

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

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 tcpd/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) OCIErrorGet(errhp,1,NULL, (sb4 *) &errcode, (text*) msg,
                         2048,OCI_HTYPE_ERROR);
    else
      (void) OCIErrorGet(errhp,1,NULL, (sb4 *) &errcode, (text*) msg,
                         2048,OCI_HTYPE_ENV);
    fprintf(stderr,"%s\n",msg);
    break;
  case OCI_INVALID_HANDLE:
    fprintf(stderr, "Error: Invalid Handle.\n");
```

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

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

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

 ACIDexit();

 exit(1);
}

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

 /* Initialize the cursors etc. */

 (void) ACIDinit();

 /* sleep for some time (triggering) */

 sleep(trig);

 /* start doing the ACID transactions */

 tr_start = gettime();

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

 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\n", o_tprice);
   }

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

   /* Shall we commit? */

   if (BIT(flag, COMMIT)) {
     need_commit = 1;
     while (need_commit) {
       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\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
,errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,lname,strlen(lname),OCI_ATT
R_USERNAME,
        errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(passwd),OCI_A
TTR_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_SESSIO
N,errhp);

 /* Enable session parallel dml */

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

 /* Enable session parallel ddl */

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


 /* Make session serializable */

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

 /* Set optimizer_index_cost_adj = 25 */

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

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


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

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



OCIbbname(curi,&l_keyi_bp,errhp,":l_key",ADR(l_key),SIZ(l_key),SQ
LT_INT);

OCIbbname(curi,&o_keyi_bp,errhp,":o_key",ADR(o_key),SIZ(o_key),
SQLT_INT);

#endif /* NOLKEY */

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

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

 /* bind variables */


OCIbbname(curr,l_key_bp,errhp,":l_key",ADR(l_key),SIZ(l_key),SQL
T_INT);

OCIbbname(curr,o_key_bp,errhp,":o_key",ADR(o_key),SIZ(o_key),SQ
LT_INT);

OCIbbname(curr,delta_bp,errhp,":delta",ADR(delta),SIZ(delta),SQLT_I
NT);

OCIbbname(curr,l_pkey_bp,errhp,":l_pkey",ADR(l_pkey),SIZ(l_pkey),
SQLT_INT);

OCIbbname(curr,l_skey_bp,errhp,":l_skey",ADR(l_skey),SIZ(l_skey),S
QLT_INT);
```

```
OCIbbname(curr,l_quan_bp,errhp,":l_quan",ADR(l_quan),SIZ(l_quan),
SQLT_INT);
 OCIbbname(curr,l_newquan_bp,errhp,":l_newquan",ADR(l_newquan),
        SIZ(l_newquan),SQLT_INT);

OCIbbname(curr,l_tax_bp,errhp,":l_tax",ADR(l_tax),SIZ(l_tax),SQLT_
FLT);

OCIbbname(curr,l_disc_bp,errhp,":l_disc",ADR(l_disc),SIZ(l_disc),SQ
LT_FLT);

OCIbbname(curr,l_eprice_bp,errhp,":l_eprice",ADR(l_eprice),SIZ(l_epr
ice),
        SQLT_FLT);

OCIbbname(curr,l_neweprice_bp,errhp,":l_neweprice",ADR(l_newepric
e),
        SIZ(l_neweprice),SQLT_FLT);


OCIbbname(curr,o_tprice_bp,errhp,":o_tprice",ADR(o_tprice),SIZ(o_tp
rice),
        SQLT_FLT);

OCIbbname(curr,o_newtprice_bp,errhp,":o_newtprice",ADR(o_newtpri
ce),
        SIZ(o_newtprice), SQLT_FLT);
 OCIbbname(curr,rprice_bp,errhp,":rprice",ADR(rprice),SIZ(rprice),
SQLT_FLT);
 OCIbbname(curr,cost_bp,errhp,":cost",ADR(cost),SIZ(cost),
SQLT_FLT);

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

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

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

 /* bind variables */


OCIbbname(cure1,l_neweprice1_bp,errhp,":l_neweprice",ADR(l_newep
rice),
        SIZ(l_neweprice),SQLT_FLT);

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

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

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


OCIbbname(cure2,o_newtprice2_bp,errhp,":o_newtprice",ADR(o_newt
price),
        SIZ(o_newtprice),SQLT_FLT);

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

}
```

C.6 atranspl.h

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

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

  DESCRIPTION

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

*/
#ifndef ATRANSPL_H

#define ATRANSPL_H

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

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

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

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

extern int errno;

#ifndef NULL
#define NULL 0
#endif

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

#ifndef DISCARD
# define DISCARD (void)
#endif

#ifndef sword
# define sword int
#endif

#ifndef ub1
#define ub1 unsigned char
#endif

#define UNAME_LEN 64
```

```
#define WRITE_BUF_LEN 1024

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

#define ADR(object) ((ub1 *)&(object))
#define SIZ(object) ((sword)sizeof(object))
#define BIS(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_DE
FAULT)) != OCI_SUCCESS) \
      sql_error(errh,status,1); \
   else \
      DISCARD 0

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

#define OCIbbnamei(stmh,bindp,errh,sqlvar,progv,progvl,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,progvl,ftype,indp,0,0,0,0,OCI_DEFAULT)) !=
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(svcp,errh) \
   if((status=OCITransRollback(svcp,errh,OCI_DEFAULT)) !=
OCI_SUCCESS) \
      sql_error(errh,status,1); \
   else \
      DISCARD 0

#define ISOTXT "alter session set isolation_level = serializable"
#define PDMLTXT "alter session force parallel dml parallel (degree 4)"
#define PDDLTXT "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_linenumber) INTO :l_key FROM lineitem \
WHERE l_orderkey = :o_key; END;"

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

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

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

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

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

#endif /* ATRANSPL_H */
```

C.7 ckpt.sh

```
#!/bin/ksh
#
# $Header: ckpt.sh 08-aug-99.17:37:07 mpoess Exp $
#
# ckpt.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#   NAME
#     ckpt.sh - <one-line expansion of the name>
#
#  DESCRIPTION
#     Usage: ckpt.sh
```

```
#     Start database checkpoint
#
#   NOTES
#     <other useful comments, qualifications, etc.>
#
#   MODIFIED   (MM/DD/YY)
#   mpoess      08/08/99 - Creation
#   mpoess      08/08/99 - Creation
#

. $KIT_DIR/env

sqlplus -s /NOLOG << !

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

C.8 cnt_hist.sql

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

C.9 consist.sh

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

. $KIT_DIR/env

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

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

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

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

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

usage() {

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

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

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


if [ $ITER -lt 100 ]
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 @/dbms/oracle10i/kit/acid/consistency/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/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 @/dbms/oracle10i/kit/acid/consistency/consist $j >>
$CON2
   echo "------------------------------" >> $CON2
done
   i=`expr $i + 1`
done
```

C.10 consist.sql

```
Rem
Rem $Header: consist.sql 08-aug-99.16:59:17 mpoess Exp $
Rem
Rem consist.sql
Rem
Rem  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
Rem
Rem    NAME
Rem      consist.sql - <one-line expansion of the name>
Rem
Rem  DESCRIPTION
Rem       Verifies the consistency of TPC-D database using the
Rem       consistency condition.
Rem
Rem Usage:   sqlplus tpcd/tpcd @consist
Rem
Rem NOTE
Rem REQUIRES PACKAGES prvtotpt and dbmsotpt
rem
Rem    MODIFIED   (MM/DD/YY)
Rem    mpoess      08/08/99 - Creation
Rem    mpoess      08/08/99 - Created
Rem

set verify off
rem set termout on
rem set echo on


REM
REM Get today's date.
REM

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

set serverout on;

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


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

   diff := l_tprice - o_tprice;

   dbms_output.put_line('O_TOTALPRICE: ' ||
TO_CHAR(trunc(o_tprice,2)));
```

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

END;
.
/


spool off
exit
```

C.11 count_tx.sh

```
#!/bin/ksh

STEM=$1
ITER=$2
OUT=$3
FIN=FALSE
while [ "$FIN" = "FALSE" ]
do
 s=0
 FIN=TRUE
 while [ $s -lt $STEM ]
 do
   nt=`grep "Transaction Completed" $OUT/dura${s} | wc -l`
   if [ $nt -lt $ITER ];then
     FIN=FALSE
   fi
   s=`expr $s + 1`
 done
 sleep 5
done
echo all streams have commited $ITER transactions
```

C.12 d_hist.sql

```
Rem
Rem $Header: d_hist.sql 07-aug-99.21:33:08 mpoess Exp $
Rem
Rem d_hist.sql
Rem
Rem  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
Rem
Rem    NAME
Rem      d_hist.sql - <one-line expansion of the name>
Rem
Rem    DESCRIPTION
Rem      Creates a history table for ACID test purpose.
Rem
Rem    NOTES
Rem      <other useful comments, qualifications, etc.>
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    mpoess      08/07/99 - Creation
Rem    mpoess      08/07/99 - Created
Rem


set termout on;
set serverout on;
set echo on;

drop table history;

create table history
(
        h_p_key    number,
        h_s_key    number,
        h_o_key number,
        h_l_key    number,
        h_delta number,
        h_date_t date
);

exit;
```

C.13 end_acid.sh

```
#!/bin/ksh
#
# $Header: end_acid.sh 08-aug-99.17:06:20 mpoess Exp $
#
# end_acid.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#   NAME
#     end_acid.sh - <one-line expansion of the name>
#
#  DESCRIPTION
#     end_cons.sh <pid of the durability run>
#     Options: See usage below
#
#   NOTES
#     <other useful comments, qualifications, etc.>
#
#   MODIFIED   (MM/DD/YY)
#   mpoess      08/08/99 - Creation
#   mpoess      08/08/99 - Creation
#

. $KIT_DIR/env

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

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

# get history count

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

# perform the consistency

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

```
  do
    sqlplus tpch/tpch @consist $j >> $DURA_DIR/duraconsa
  done
  i=`expr $i + 1`
done

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

cat $ORACLE_HOME/rdbms/log/alert_qual.log >
${DURA_DIR}/alert_qual.log.post_dura 2>&1
cat $ORACLE_HOME/rdbms/log/alert_ASM.log >
${DURA_DIR}/alert_ASM.log.post_dura 2>&1
```

C.14 iso.sh

```
#!/bin/ksh
#
# $Header: iso.sh 17-aug-99.15:44:51 mpoess Exp $
#
# iso.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#   NAME
#     iso.sh
#
#   DESCRIPTION
#    This script triggers all 6 isolation tests. In addition,
#    it creates more readable formats of the isolation test output.
#   NOTES
#     <other useful comments, qualifications, etc.>
#
#   MODIFIED   (MM/DD/YY)
#   mpoess      08/17/99 - Creation
#   mpoess      08/17/99 - Creation
#

for iso in iso1 iso2 iso3 iso4 iso5 iso6;do
              echo Running isolation test $iso
              /dbms/oracle10i/kit/acid/isolation/${iso}.sh
#             echo Creating nicely formated output of ACID test $iso
#             /dbms/oracle10i/kit/acid/isolation/xiso.pl -o
${ACID_OUT}/${iso}
done
```

C.15 iso1.sh

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

. $KIT_DIR/env

# May need to change the following:
RSH=rsh

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

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

USER=$DATABASE_USER
PROG=atranspl


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

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

usage() {

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

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

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


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

# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

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

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

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

sleep 1

# start ACID transaction, Sleep for 60 second before COMMIT

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

# let's sleep 10 seconds before starting ACID query

sleep 10

# start ACID query with the same OKEY

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}  sqlplus $USER @$ACID_DIR/isolation/a_query
$OKEY >> $TXN2FILE
else
sqlplus $USER @$ACID_DIR/isolation/a_query $OKEY >>
$TXN2FILE
fi

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

C.16 iso2.sh

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

. $KIT_DIR/env

RSH=rsh

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

DURA_DIR=$ACID_DIR/dura

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

USER=$DATABASE_USER
PROG=atranspl

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

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

usage() {

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

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

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


# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

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

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

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

sleep 1

# start ACID transaction, Sleep for 30 second before ROLLBACK
```

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

# let's sleep 10 seconds before starting ACID query

sleep 10

# start ACID query with the same OKEY

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 ${HOST}  sqlplus "$USER"
@$ACID_DIR/isolation/a_query $OKEY >> $TXN2FILE
else
sqlplus $USER @$ACID_DIR/isolation/a_query $OKEY >>
$TXN2FILE
fi

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

C.17 iso3.sh

```
#!/bin/ksh
#
# $Header: iso3.sh 04-aug-99.09:20:35 mpoess Exp $
#
# iso3.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#   NAME
#     iso3.sh - <one-line expansion of the name>
#
#  DESCRIPTION
#     Usage: iso3.sh [-u user/password] [-n remote_node] -h
#     Options: See usage below
#  NOTES
#      For a cross node isolation test, assume the local node is
#      one of the participating nodes.  The other node can be
#      specified by the -n option.
#             We need to make sure the remote node has access to the
#             file system on the local node.  Otherwise, we need to rcp
#             the keyfile to the remote system.
#      You need to set the environment variable TPCD_KIT_DIR
#
#   MODIFIED   (MM/DD/YY)
#   mpoess     08/04/99 - Creation
#   mpoess     08/04/99 - Creation
#

. $KIT_DIR/env

# May need to change the following:
RSH=rsh

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

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

USER=$DATABASE_USER
PROG=atranspl

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

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

usage() {

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

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

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


# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE
if [ "$HOST" != "" ]
then
  rcp $KEYFILE ${HOST}:$KEYFILE
fi

sleep 1

# start ACID transaction, Sleep for 30 second before COMMIT

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

# let's sleep 10 seconds before starting second ACID transaction

sleep 10

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

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

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

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

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

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

C.18 iso4.sh

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

. $KIT_DIR/env

# May need to change the following:
RSH=rsh

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

DURA_DIR=$ACID_DIR/dura

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

USER=$DATABASE_USER
PROG=atranspl

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

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

usage() {

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

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

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


# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

if [ "$HOST" != "" ]
then
  rcp $KEYFILE ${HOST}:$KEYFILE
fi

sleep 1

# start ACID transaction, Sleep for 30 second before ROLLBACK

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

# let's sleep 10 seconds before starting second ACID transaction

sleep 10

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

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

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

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

C.19 iso5.sh

```
#!/bin/ksh
#
# $Header: iso5.sh 04-aug-99.09:21:45 mpoess Exp $
#
# iso5.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#   NAME
#     iso5.sh - <one-line expansion of the name>
#
#  DESCRIPTION
#     Usage: iso5.sh [-u user/password] [-n remote_node] -h
#     Options: See usage below
#  NOTES
#      For a cross node isolation test, assume the local node is
```

```
#       one of the participating nodes.  The other node can be
#       specified by the -n option.
#       You need to set the environment variable TPCD_KIT_DIR
#
#    MODIFIED   (MM/DD/YY)
#    mpoess      08/04/99 - Creation
#    mpoess      08/04/99 - Creation
#

. $KIT_DIR/env

# May need to change the following:
RSH=rsh

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

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

USER=$DATABASE_USER
PROG=atranspl

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

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

usage() {

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

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

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


# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

if [ "$HOST" != "" ]
then
   rcp $KEYFILE ${HOST}:$KEYFILE
fi

sleep 1


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

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

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

sleep 1

# start ACID transaction, Sleep for 60 second before COMMIT

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

# let's sleep 5 seconds before starting PARTSUPP query

sleep 5

# First generate PS_PARTKEY and PS_SUPPKEY

PSKEY=`randpsup 1`

echo "Running PARTSUPP query 5 seconds AFTER the start of ACID
Transaction" \
>> $TXN2FILE
echo "`date`" >> $TXN2FILE
echo "PS_PARTKEY and PS_SUPPKEY are: $PSKEY" >>
$TXN2FILE

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

wait

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

C.20 iso6.sh

```
#!/bin/ksh
#
# $Header: iso6.sh 04-aug-99.09:22:12 mpoess Exp $
#
# iso6.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#    NAME
#      iso6.sh - <one-line expansion of the name>
#
#  DESCRIPTION
#      Usage: iso6.sh [-u user/password] [-n remote_node] -h
#      Options: See usage below
#  NOTES
#       For a cross node isolation test, assume the local node is
#       one of the participating nodes.  The other node can be
#       specified by the -n option.
```

```
#       We need to make sure the remote node has access to the
#       file system on the local node.  Otherwise, we need to rcp
#       the keyfile to the remote system.
#       You need to set the environment variable TPCD_KIT_DIR
#
#    MODIFIED   (MM/DD/YY)
#    mpoess     08/04/99 - Creation
#    mpoess     08/04/99 - Creation
#

. $KIT_DIR/env

# 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


# generate key files

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

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

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

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

sleep 1

# start Query 1, use 0 as the delta

echo "Running Query 21 at `date`" >> $TXN1FILE
sqlplus $USER @$KIT_DIR/acid/isolation/q21 >> $TXN1FILE &

# sleep 2 seconds before starting ACID transaction

sleep 2

# start ACID transaction, COMMIT after one second

echo "Starting ACID 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 21 at `date`" >> $TXN3FILE
sqlplus $USER @$KIT_DIR/acid/isolation/q21 >> $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
```

C.21 prepare4acid.sh

```
#!/bin/ksh
#
# $Header: prepare4acid.sh 12-aug-99.17:09:18 mpoess Exp $
#
# prepare4acid.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#    NAME
#      prepare4acid.sh
#
#    DESCRIPTION
#     Prepares the qualification database for the acid tests.
#
#    NOTES
#
#    MODIFIED   (MM/DD/YY)
#    mpoess     08/12/99 - Creation
#    mpoess     08/12/99 - Creation
```

```
#
. $KIT_DIR/env

sqlplus $DATABASE_USER @d_hist
sqlplus $DATABASE_USER @atrans
```

C.22 q1.sql

```
Rem
Rem $Header: template.sql 06-feb-96.13:23:14 mpoess   Exp $
Rem
Rem q1.sql
Rem
Rem  Copyright (c) Oracle Corporation 2001. All Rights Reserved.
Rem
Rem    NAME
Rem      q1.sql - <one-line expansion of the name>
Rem
Rem    DESCRIPTION
Rem      used in isolation test 6
Rem
Rem    NOTES
Rem      <other useful comments, qualifications, etc.>
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    mpoess      02/13/01 - Created
Rem

set serverout on;

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

select
      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')  - 0
group by
      l_returnflag,
      l_linestatus
order by
      l_returnflag,
      l_linestatus;

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

exit;
```

C.23 q21.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 * 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 <= 10;

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

exit;
```

C.24 randkey.c

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

/*
```

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

  DESCRIPTION
    Generate random keys for ACID transactions:
    O_ORDERKEY unique random (1..SF*150000*4) and only
    first 8 keys out of every 32 are populated.
      and
    L_ORDERKEY based on Clause 3.1.6.2
    DELTA random (1..100)
*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "atranspl.h"

#define ORDERCNT 150000.0

/* MK_SPARSE adopted from dss.h */

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

void sql_error();
void usage();
void ACIDinit();
long atol();
void srand48();
long lrand48();

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

typedef struct aciddef {
  long okey;
  long lkey;
  int delta;
} adef;

long l_key = 0;
long o_key = 0;
char lname[UNAME_LEN];
char *passwd;

/* OCI handles */

OCIEnv *tpcenv;
OCIServer *tpcsrv;
OCIError *errhp;
OCISvcCtx *tpcsvc;
OCISession *tpcusr;
OCIStmt *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':
    strncpy((char *) lname, ++(argv[0]), UNAME_LEN);
    if (strchr((char *) lname, '/') == NULL) {
      usage();
      exit(-1);
    }
    break;
  default:
    fprintf(stderr, "Unknown argument %s\n", argv[0]);
    usage();
    break;
  }
 }

 ACIDinit();

 /* initialize array for random numbers */

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

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

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

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

  /* Obtain l_key from l_key query */

  OCIsexec(tpcsvc,curi,errhp,1);

  /* l_key is the highest l_linenumber 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_ATT
R_USERNAME,
        errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(passwd),OCI_A
TTR_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_SESSIO
N,errhp);

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

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


OCIbbname(curi,l_key_bp,errhp,":l_key",ADR(l_key),SIZ(l_key),SQL
T_INT);

OCIbbname(curi,o_key_bp,errhp,":o_key",ADR(o_key),SIZ(o_key),SQ
LT_INT);
 }
```

C.25 randpsup.c

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

/*

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

  DESCRIPTION
    Generate random keys for ACID PARTSUPP transactions:
    (Clause 4.2.3)
    PS_PARTKEY random within [SF*200000]
      and
    PS_SUPPKEY = (PS_PARTKEY + (i * ((S/4) +
(int)(PS_PARTKEY - 1)
                /S))) % S + 1
    where i random within [0..3] and S = SF * 10000

MODIFIED
  mpoess     10/23/02 - mpoess_update_from_visa
  mpoess     01/04/01 - Creation

*/

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

#define PS_PER_SF 200000.0
#define S_PER_SF 10000.0
#define SUPP_PER_PART 4

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

#define PART_SUPP_BRIDGE(tgt, p, s) \
   { \
   long tot_scnt = (long) (S_PER_SF * sf); \
   tgt = (p + s *  (tot_scnt / SUPP_PER_PART +  \
      (long) ((p - 1) / tot_scnt))) % tot_scnt + 1; \
   }

void usage();
double atof();
void srand48();
long lrand48();


main(argc, argv)
    int argc;
    char **argv;
{

 double sf = 0.1;          /* scale factor      */
 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");
}
```

C.26 run_acid.sh

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

. $KIT_DIR/env

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

usage() {

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

```
  exit 1;
}

ITER=1000
STEM=${NUM_STREAMS}
let STEM="$STEM + 1" # add one for the update stream
SF=1
PROG=atranspl
IN=${ACID_DIR}/acid_in
DURA_DIR=$ACID_OUT/dura
OUT=$DURA_DIR/drate
DURA=$DURA_DIR/dura
KEY=${DURA_DIR}/key$$_
echo "$$" > ${DURA_DIR}/shellpid
USER=tpch/tpch
TRIG=1
HCNT=duracntb

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

# get all the options

while :
do
  case "$1" in
  -n) shift; ITER=$1;;
  -s) shift; STEM=$1;;
  -p) shift; PROG=$1;;
  -i) shift; IN=$1;;
  -o) shift; OUT=$1;;
  -d) shift; DURA=$1;;
  -u) shift; USER=$1;;
  -h) usage; exit 0;;
  -t) shift; TRIG=$1;;
  -f) shift; SF=$1;;
  --) break;;
  esac
  shift;
done

#collect system info before durability start
cat /var/adm/syslog/syslog.log > ${DURA_DIR}/syslog_pre_dura 2>&1
ps -ef > ${DURA_DIR}/ps.out.pre_dura 2>&1
cat $ORACLE_HOME/rdbms/log/alert_qual.log >
${DURA_DIR}/alert_qual.log.pre_dura 2>&1
cat $ORACLE_HOME/rdbms/log/alert_ASM.log >
${DURA_DIR}/alert_ASM.log.pre_dura 2>&1

echo "Starting ACID run..."

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

# Get history count before the run

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

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

wait
# perform the consistency

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

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

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

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

done

wait

echo "ACID run completed"
```

C.27 sample.sh

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

# $1 durability output file

. $KIT_DIR/env

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



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

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

i=1
while [ $i -le 6 ]
do

j=`cat /tmp/6keys$$ | tail -${i} | head -1`
sqlplus tpch/tpch @sample $j
```

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

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

C.28 sample.sql

```
Rem
Rem $Header: sample.sql 08-aug-99.17:10:34 mpoess Exp $
Rem
Rem sample.sql
Rem
Rem  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
Rem
Rem    NAME
Rem      sample.sql - <one-line expansion of the name>
Rem
Rem    DESCRIPTION
Rem      <short description of component this file declares/defines>
Rem
Rem    NOTES
Rem      <other useful comments, qualifications, etc.>
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    mpoess      08/08/99 - Creation
Rem    mpoess      08/08/99 - Created
Rem

alter session set nls_date_format = 'YYYY-MM-DD HH:MI:SS';
select * from history where  h_o_key = &&1 and h_l_key = &&2;

exit;
```

Appendix D Query text and Output

D.1 qryqual

Begin Execution at Mon Nov 20 09:10:47 2006

```
-- using default substitutions
-- @(#)1.sql              2.1.6.2
-- 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 1.36 seconds.


-- @(#)2.sql              2.1.6.2
-- 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 acros
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

<deleted>


7852.45               Supplier#000005864     RUSSIA
8363.00               Manufacturer#4
WCNfBPZeSXh3h,c                         32-454-883-3821
blithely regular deposits
7850.66               Supplier#000001518     UNITED KINGDOM
86501.00              Manufacturer#1
ONda3YJiHKJOC                           33-730-383-3892
```

```
furiously final accounts wake carefully idle requests. even dolphins
wake acc
7843.52             Supplier#000006683       FRANCE
11680.00            Manufacturer#4
2Z0JGkiv01Y00oCFwUGfviIbhzCdy          16-464-517-8943
carefully bold accounts doub


100 rows processed.
Query Processed in 1.31 seconds.


-- @(#)3.sql             2.1.6.2
-- 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          REVENUE             O_ORDERDATE
O_SHIPPRIORITY
2456423.00          406181.01           1995-03-05  0.00
3459808.00          405838.70           1995-03-04  0.00
492164.00           390324.06           1995-02-19  0.00
1188320.00          384537.94           1995-03-09  0.00
2435712.00          378673.06           1995-02-26  0.00
4878020.00          378376.80           1995-03-12  0.00
5521732.00          375153.92           1995-03-13  0.00
2628192.00          373133.31           1995-02-22  0.00
993600.00           371407.46           1995-03-05  0.00
2300070.00          367371.15           1995-03-13  0.00


10 rows processed.
Query Processed in 1.59 seconds.


-- @(#)4.sql             2.1.6.2
-- 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 1.48 seconds.


-- @(#)5.sql             2.1.6.2
-- 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 4.04 seconds.


-- @(#)6.sql              2.1.6.2
-- 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 0.23 seconds.


-- @(#)7.sql    2.1.6.2
-- 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 3.70 seconds.


-- @(#)8a.sql           2.1.6.2
-- TPC-H/TPC-R National Market Share Query (Q8)
-- 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 2.48 seconds.

```
-- @(#)9.sql           2.1.6.2
-- 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
ALGERIA	1993.00	54942718.13

<deleted>

VIETNAM	1994.00	50596057.26
VIETNAM	1993.00	50953919.15
VIETNAM	1992.00	49613838.32

175 rows processed.
Query Processed in 4.49 seconds.

```
-- @(#)10.sql          2.1.6.2
-- TPC-H/TPC-R Returned Item Reporting Query (Q10)
-- Functional Query Definition
-- Approved February 1998


select  *  from  (
select
c_custkey,
c_name,
sum(l_extendedprice * (1 - l_discount)) as revenue,
c_acctbal,
n_name,
c_address,
c_phone,
c_comment
from
customer,
orders,
lineitem,
nation
where
c_custkey = o_custkey
and l_orderkey = o_orderkey
and o_orderdate >= to_date ('1993-10-01',  'YYYY-MM-DD')
and o_orderdate < add_months(  to_date( '1993-10-01',  'YYYY-MM-
DD'),  3)
and l_returnflag = 'R'
and c_nationkey = n_nationkey
group by
c_custkey,
c_name,
c_acctbal,
c_phone,
n_name,
c_address,
c_comment
order by
revenue desc)
where rownum <= 20
```

C_CUSTKEY	C_NAME	REVENUE
C_ACCTBAL	N_NAME	
C_ADDRESS	C_PHONE	
C_COMMENT		
57040.00	Customer#000057040	734235.25
632.87	JAPAN	
Eioyzjf4pp	22-895-641-3466	
requests sleep blithely about the furiously i		
143347.00	Customer#000143347	721002.69
2557.47	EGYPT	
1aReFYv,Kw4	14-742-935-3718	
fluffily bold excuses haggle finally after the u		
60838.00	Customer#000060838	679127.31
2454.77	BRAZIL	
64EaJ5vMAHWJlBOxJklpNc2RJiWE	12-913-494-9813	
furiously even pinto beans integrate under the ruthless foxes; ironic, even dolphins across the slyl		
101998.00	Customer#000101998	637029.57
3790.89	UNITED KINGDOM	
01c9CILnNtfOQYmZj	33-593-865-6378	
accounts doze blithely! enticing, final deposits sleep blithely special accounts. slyly express accounts pla		
125341.00	Customer#000125341	633508.09
4983.51	GERMANY	
S29ODD6bceU8QSuuEJznkNaK	17-582-695-5962	
quickly express requests wake quickly blithely		
25501.00	Customer#000025501	620269.78
7725.04	ETHIOPIA	
W556MXuoiaYCCZamJI,Rn0B4ACUGdkQ8DZ	15-874-808-6793	
quickly special requests sleep evenly among the special deposits. special deposi		
115831.00	Customer#000115831	596423.87
5098.10	FRANCE	
rFeBbEEyk dl ne7zV5fDrmiq1oK09wV7pxqCgIc	16-715-386-3788	
carefully bold excuses sleep alongside of the thinly idle		
84223.00	Customer#000084223	594998.02

```
528.65                UNITED KINGDOM
nAVZCs6BaWap rrM27N 2qBnzc5WBauxbA       33-442-824-8191
pending, final ideas haggle final requests. unusual, regular asymptotes
affix according to the even foxes.
54289.00              Customer#000054289        585603.39
5583.02               IRAN
vXCxoCsU0Bad5JQI ,oobkZ                  20-834-292-4707
express requests sublate blithely regular requests. regular, even ideas
solve.
39922.00              Customer#000039922        584878.11
7321.11               GERMANY
Zgy4s50l2GKN4pLDPBU8m342gIw6R            17-147-757-8036
even pinto beans haggle. slyly bold accounts inte
6226.00               Customer#000006226        576783.76
2230.09               UNITED KINGDOM
8gPu8,NPGkfyQQ0hcIYUGPIBWc,ybP5g,        33-657-701-3391
quickly final requests against the regular instructions wake blithely final
instructions. pa
922.00                Customer#000000922        576767.53
3869.25               GERMANY
Az9RFaut7NkPnc5zSD2PwHgVwr4jRzq          17-945-916-9648
boldly final requests cajole blith
147946.00             Customer#000147946        576455.13
2030.13               ALGERIA
iANyZHjqhyy7Ajah0pTrYyhJ                 10-886-956-3143
furiously even accounts are blithely above the furiousl
115640.00             Customer#000115640        569341.19
6436.10               ARGENTINA
Vtgfia9qI 7EpHgecU1X                     11-411-543-4901
final instructions are slyly according to the
73606.00              Customer#000073606        568656.86
1785.67               JAPAN
xuR0Tro5yChDfOCrjkd2ol                   22-437-653-6966
furiously bold orbits about the furiously busy requests wake across the
furiously quiet theodolites. d
110246.00             Customer#000110246        566842.98
7763.35               VIETNAM
7KzflgX MDOq7sOkI                        31-943-426-9837
dolphins sleep blithely among the slyly final
142549.00             Customer#000142549        563537.24
5085.99               INDONESIA
ChqEoK43OysjdHbtKCp6dKqjNyvvi9           19-955-562-2398
regular, unusual dependencies boost slyly; ironic attainments nag fluffily
into the unusual packages?
146149.00             Customer#000146149        557254.99
1791.55               ROMANIA
s87fvzFQpU                               29-744-164-6487
silent, unusual requests detect quickly slyly regul
52528.00              Customer#000052528        556397.35
551.79                ARGENTINA
NFztyTOR10UOJ                            11-208-192-3205
unusual requests detect. slyly dogged theodolites use slyly. deposit
23431.00              Customer#000023431        554269.54
3381.86               ROMANIA
HgiV0phqhaIa9aydNoIlb                    29-915-458-2654
instructions nag quickly. furiously bold accounts cajol


20 rows processed.
Query Processed in 1.92 seconds.


-- @(#)11.sql            2.1.6.2
-- TPC-H/TPC-R Important Stock Identification Query (Q11)
-- Functional Query Definition
-- Approved February 1998


select
ps_partkey,
sum(ps_supplycost * ps_availqty) as value
from
partsupp,
supplier,
nation
where
ps_suppkey = s_suppkey
and s_nationkey = n_nationkey
and n_name = 'GERMANY'
group by
ps_partkey having
sum(ps_supplycost * ps_availqty) > (
select
sum(ps_supplycost * ps_availqty) * 0.0001000000
from
partsupp,
supplier,
nation
where
ps_suppkey = s_suppkey
and s_nationkey = n_nationkey
and n_name = 'GERMANY'
)
order by
value desc

PS_PARTKEY            VALUE
129760.00             17538456.86
166726.00             16503353.92
191287.00             16474801.97
161758.00             16101755.54
34452.00              15983844.72
139035.00             15907078.34
9403.00               15451755.62
154358.00             15212937.88
38823.00              15064802.86
85606.00              15053957.15
33354.00              14408297.40
154747.00             14407580.68
82865.00              14235489.78
76094.00              14094247.04
222.0 13937777.74
223.0

<deleted>


101674.00             7879324.60
51968.00              7879102.21
72073.00              7877736.11
5182.00               7874521.73


1048 rows processed.
Query Processed in 5.61 seconds.


-- @(#)12.sql            2.1.6.2
-- 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 1.25 seconds.


-- @(#)13.sql           2.1.6.2
-- 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 1.18 seconds.


-- @(#)14.sql           2.1.6.2
-- 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 1.02 seconds.


-- @(#)15.sql   2.1.6.2
-- 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 7.38 seconds.
```

```
-- @(#)16.sql           2.1.6.2
-- 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              SUPPLIER_CNT
Brand#41  MEDIUM BRUSHED TIN      3.00                28.00
Brand#54  STANDARD BRUSHED COPPER  14.00               27.00
Brand#11  STANDARD BRUSHED TIN    23.00               24.00
Brand#11  STANDARD BURNISHED BRASS 36.00              24.00
Brand#15  MEDIUM ANODIZED NICKEL   3.00               24.00
Brand#15  SMALL ANODIZED BRASS    45.00               24.00
Brand#15  SMALL BURNISHED NICKEL  19.00               24.00
Brand#21  MEDIUM ANODIZED COPPER   3.00               24.00
Brand#22  SMALL BRUSHED NICKEL    3.00                24.00
Brand#22  SMALL BURNISHED BRASS   19.00               24.00
Brand#25  MEDIUM BURNISHED COPPER  36.00              24.00

<deleted>


Brand#52  MEDIUM BRUSHED BRASS    45.00               3.00
Brand#53  MEDIUM BRUSHED TIN      45.00               3.00
Brand#54  ECONOMY POLISHED BRASS  9.00                3.00
Brand#55  PROMO PLATED BRASS      19.00               3.00
Brand#55  STANDARD PLATED TIN     49.00               3.00


18314 rows processed.
Query Processed in 1.54 seconds.
```

```
-- @(#)17.sql           2.1.6.2
-- 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 1.42 seconds.
```

```
-- @(#)18.sql           2.1.6.2
-- 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

<deleted>


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 1.41 seconds.
```

```
-- @(#)19.sql           2.1.6.2
-- 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 2.29 seconds.
```

```
-- @(#)20.sql           2.1.6.2
-- 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
YV45D7TkfdQanOOZ7q9QxkyGUapU1oOWU6q3
Supplier#000000197      YC2Acon6kjY3zj3Fbxs2k4Vdf7X0cd2F
Supplier#000000226      83qOdU2EYRdPQAQhEtn GRZEd
Supplier#000000285      Br7e1nnt1yxrw6ImgpJ7YdhFDjuBf
Supplier#000000378      FfbhyCxWvcPrO8ltp9
Supplier#000000402      i9Sw4DoyMhzhKXCH9By,AYSgmD
Supplier#000000530      0qwCMwobKY OcmLyfRXlagA8ukENJv,

<deleted>



Supplier#000009862      rJzweWeN58
Supplier#000009868      ROjGgx5gvtkmnUUoeyy7v
Supplier#000009869
ucLqxzrpBTRMewGSM29t0rNTM30g1Tu3Xgg3mKag
Supplier#000009899      7XdpAHrzr1t,UQFZE
Supplier#000009974      7wJ,J5DKcxSU4Kp1cQLpbcAvB5AsvKT


204 rows processed.
Query Processed in 1.80 seconds.


-- @(#)21.sql           2.1.6.2
-- 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#000000486      15.00
Supplier#000000565      15.00
Supplier#000001046      15.00
Supplier#000001047      15.00

<deleted>

Supplier#000001925      12.00
Supplier#000002039      12.00
Supplier#000002357      12.00
Supplier#000002483      12.00


100 rows processed.
Query Processed in 5.14 seconds.


-- @(#)22.sql   2.1.4.2
-- TPC-H/TPC-R 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 1.33 seconds.

Ended Executing this Stream at Mon Nov 20 09:11:41 2006

Stream Started at 1164042647.06
Stream Ended at 1164042701.03
Stream Processed in 53.96 seconds

SQL statements processed: 22

Appendix E Seed and Input Parameters

E.1 Seed

1116195250

E.2 qp2.0

14 1993-07-01
2 26 COPPER EUROPE
9 pale
20 orange 1994-01-01 BRAZIL
6 1993-01-01 0.04 25
17 Brand#51 SM CASE
18 312
8 JORDAN MIDDLE EAST LARGE BRUSHED STEEL
21 INDONESIA
13 unusual packages
3 HOUSEHOLD 1995-03-24
22 12 18 23 16 22 13 25
16 Brand#35 STANDARD BRUSHED 1 9 19 28 17 45 10 12
4 1996-09-01
11 FRANCE 0.0000000100
15 1996-08-01
1 79
10 1993-08-01
19 Brand#14 Brand#21 Brand#55 7 12 26
5 EUROPE 1993-01-01
7 INDONESIA JORDAN
12 AIR TRUCK 1993-01-01

E.3 qp2.1

21 ARGENTINA
3 AUTOMOBILE 1995-03-09
18 314
5 MIDDLE EAST 1994-01-01
11 ROMANIA 0.0000000100
7 ALGERIA ETHIOPIA
6 1994-01-01 0.02 24
20 beige 1993-01-01 PERU
17 Brand#53 SM JAR
12 REG AIR MAIL 1993-01-01
16 Brand#25 LARGE ANODIZED 5 3 12 26 1 37 43 17
15 1994-04-01
13 unusual packages
10 1994-05-01
2 14 STEEL AMERICA
8 ETHIOPIA AFRICA LARGE PLATED STEEL
14 1993-10-01
19 Brand#11 Brand#14 Brand#44 2 13 22
9 moccasin
22 27 12 28 20 18 25 10
1 87
4 1994-06-01

E.4 qp2.2

6 1994-01-01 0.07 25
17 Brand#55 SM CAN
14 1994-02-01
16 Brand#55 PROMO PLATED 8 24 37 6 36 18 12 7
19 Brand#13 Brand#42 Brand#43 7 14 30
10 1993-02-01
9 maroon
2 2 BRASS MIDDLE EAST
15 1996-11-01
8 RUSSIA EUROPE LARGE ANODIZED COPPER
5 AFRICA 1994-01-01
22 17 31 24 21 23 32 10
12 SHIP MAIL 1994-01-01
7 PERU RUSSIA
13 unusual packages
18 315
1 95
4 1997-01-01
20 lawn 1996-01-01 GERMANY
3 FURNITURE 1995-03-26
11 GERMANY 0.0000000100
21 CHINA

E.5 qp2.3

8 KENYA AFRICA MEDIUM POLISHED COPPER
5 AMERICA 1994-01-01
4 1994-10-01
6 1994-01-01 0.05 25
17 Brand#52 LG CASE
7 INDONESIA KENYA
1 103
18 313
22 18 23 20 15 22 24 12
14 1994-05-01
9 lawn
10 1993-11-01
15 1994-08-01
11 SAUDI ARABIA 0.0000000100
20 snow 1995-01-01 RUSSIA
2 39 NICKEL AMERICA
21 IRAN
19 Brand#25 Brand#35 Brand#43 3 15 26
13 unusual requests
16 Brand#35 SMALL POLISHED 21 45 40 17 32 3 1 43
12 FOB MAIL 1994-01-01
3 AUTOMOBILE 1995-03-11

E.6 qp2.4

5 ASIA 1994-01-01
21 BRAZIL
14 1994-08-01
19 Brand#22 Brand#13 Brand#32 8 16 22
15 1997-02-01
17 Brand#54 LG JAR
12 TRUCK MAIL 1994-01-01

6 1994-01-01 0.02 24
4 1997-05-01
9 hot
8 FRANCE EUROPE MEDIUM BURNISHED COPPER
16 Brand#25 ECONOMY ANODIZED 15 13
6 24 5 14 2 30
11 INDIA 0.0000000100
2 27 COPPER MIDDLE EAST
10 1994-09-01
18 314
1 111
13 unusual requests
7 ARGENTINA FRANCE
22 32 24 16 19 11 15
17
3 FURNITURE 1995-03-28
20 floral 1993-01-01 IRAQ

E.7 qp2.5

Appendix F 21 SAUDI ARABIA
Appendix G 15 1994-11-01
Appendix H 4 1995-02-01
Appendix I 6 1994-01-01 0.08 25
Appendix J 7 CHINA UNITED KINGDOM
Appendix K 16 Brand#55 STANDARD BURNISHED 40
3 32 4 42 48 14
43
Appendix L 19 Brand#24 Brand#51 Brand#31 3 17
29
Appendix M 18 312
Appendix N 14 1994-11-01
Appendix O 22 11 22 13 16 20
21 29
Appendix P 11 VIETNAM 0.0000000100
Appendix Q 13 unusual requests
Appendix R 3 MACHINERY 1995-03-13
Appendix S 1 119
Appendix T 2 15 STEEL ASIA
Appendix U 5 MIDDLE EAST 1994-01-01
Appendix V 8 UNITED KINGDOM EUROPE SMALL BRUSHED COPPER
Appendix W 20 powder 1997-01-01 ARGENTINA
Appendix X 12 RAIL FOB 1994-01-01
Appendix Y 17 Brand#51 LG CAN
Appendix Z 10 1993-06-01
Appendix AA 9 gainsboro

E.8 qp2.6

10 1994-03-01
3 FURNITURE 1995-03-30
15 1997-06-01
13 express requests
6 1995-01-01 0.05 25
8 MOROCCO AFRICA SMALL PLATED COPPER
9 dodger
7 IRAN MOROCCO
4 1997-09-01
11 INDONESIA 0.0000000100
22 22 30 31 28 13 24
14
18 313
12 AIR FOB 1995-01-01
1 66
5 AFRICA 1995-01-01
16 Brand#35 MEDIUM POLISHED 19 15
24 17 32 49 40 12
2 3 BRASS MIDDLE EAST
14 1995-02-01
19 Brand#31 Brand#34 Brand#35 8 18 25
20 burnished 1995-01-01 MOZAMBIQUE
17 Brand#53 MED CASE
21 JAPAN

E.9 qp2.7

18 315
8 GERMANY EUROPE SMALL BURNISHED TIN
20 metallic 1993-01-01 FRANCE
21 EGYPT
2 40 NICKEL ASIA
4 1995-06-01
22 10 21 30 17 15 25
26
17 Brand#55 MED JAR
1 74
11 RUSSIA 0.0000000100
9 cornsilk
19 Brand#33 Brand#12 Brand#25 4 19 21
3 MACHINERY 1995-03-16
13 express requests
5 AMERICA 1995-01-01
7 BRAZIL GERMANY
10 1994-12-01
16 Brand#25 ECONOMY BRUSHED 25 3
12 15 31 34 16 6
6 1995-01-01 0.02 24
14 1995-06-01
15 1995-03-01
12 REG AIR FOB 1995-01-01

E.10 qp2.8

19 Brand#35 Brand#55 Brand#24 9 20 29
1 82
15 1997-09-01
17 Brand#52 MED CAN
5 ASIA 1995-01-01
8 UNITED STATES AMERICA STANDARD BRUSHED TIN
9 burnished
12 SHIP FOB 1995-01-01
14 1995-09-01
7 ROMANIA UNITED STATES
4 1993-02-01
3 BUILDING 1995-03-01
20 wheat 1997-01-01 SAUDI ARABIA
16 Brand#55 SMALL BURNISHED 14 49
44 45 15 34 12 2
6 1995-01-01 0.08 25
22 30 22 26 20 34 29
18
10 1993-09-01
13 express accounts
2 28 TIN AFRICA
21 VIETNAM
18 312

11 IRAN 0.0000000100

E.11 qp2.9

8 MOZAMBIQUE AFRICA STANDARD PLATED TIN
13 express accounts
2 16 STEEL ASIA
20 honeydew 1995-01-01 IRAN
17 Brand#53 JUMBO CASE
3 HOUSEHOLD 1995-03-18
6 1995-01-01 0.05 25
21 JORDAN
18 314
11 UNITED KINGDOM 0.0000000100
19 Brand#42 Brand#33 Brand#23 4 10 25
10 1994-07-01
15 1995-06-01
4 1995-09-01
22 20 23 22 24 27 11 25
1 90
7 IRAQ MOZAMBIQUE
12 FOB SHIP 1995-01-01
9 black
14 1995-12-01
5 EUROPE 1995-01-01
16 Brand#35 LARGE PLATED 31 20 17 10 11 1 18 28

Appendix F Benchmark Scripts

F.1 dbtables.sql

```
set echo on
set numwidth 25
spool rdbtablest
SELECT /*+ full(A) */ COUNT(*) FROM LINEITEM A;

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 /*+ full(A) */ COUNT(*) FROM ORDERS A;

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;
spool off
exit;
```

F.2 firstten.sql

```
set echo on
set numwidth 25
spool count.out
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;
spool off
exit;
```

F.3 gen_seed.sh

```
#!/bin/ksh

SEED_FILE=$1

#Generate the seed
echo "Setting the random number seed"
PSEED=`date +%m:%d:%H:%M:%S | sed -e 's/://g'`
echo "Using ${PSEED} as seed0"
echo ${PSEED} >  $SEED_FILE
echo "Done setting the random number seed"
```

F.4 gtime.c

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

/*

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

  DESCRIPTION
    <short description of facility this file declares/defines>

  EXPORT FUNCTION(S)
    <external functions defined for use outside package - one-line
descriptions>

  INTERNAL FUNCTION(S)
    <other external functions defined - one-line descriptions>

  STATIC FUNCTION(S)
    <static functions defined - one-line descriptions>

  NOTES
    <other useful comments, qualifications, etc.>

  MODIFIED   (MM/DD/YY)
  mpoess      10/23/02 - mpoess_update_from_visa
  mpoess      08/29/01 - Creation

*/
#include<stdio.h>
#include<stdlib.h>

# include <sys/time.h>

main ()
{
  struct timeval tv;

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

  printf ("%.2f\n", ((double) tv.tv_sec + (1.0e-6 * (double) tv.tv_usec)) )
;

}

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

F.5 qexecpl.c

```
#ifdef RCSID
static char *RCSid =
   "$Header: qexecpl.c 17-oct-2001.09:29:47 mpoess Exp $ ";
#endif /* RCSID */

/* Copyright (c) 1999, 2001, Oracle Corporation.  All rights reserved.  */

/*

  NAME
    qexecpl.c - <one-line expansion of the name>

  DESCRIPTION
    SQL Execution Engine, Oracle v8,  OCI version

  PRIVATE FUNCTION(S)
    <list of static functions defined in .c file - with one-line descriptions>

  MODIFIED   (MM/DD/YY)
     mpoess    10/17/01 - add serialization level in SQLinit
     mpoess    02/22/01 - add linux changes
     mpoess    08/05/99 - make compile
     mpoess    11/13/98 - fix pddl statement
     pswong    02/19/97 - migrating to version 8
     pswong    04/02/96 - more polishing
     pswong    03/25/96 - polish up
     pswong    03/06/96 - created

*/

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

#include "qexecpl.h"

/* Function Prototypes */

extern double 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
*/

sltype slist[MAX_SEL_LIST];  /* Array for describing Select List
*/
dltype *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;
OCIServer *tpcsrv = NULL;
OCIError *errhp = NULL;
OCISvcCtx *tpcsvc = NULL;
OCISession *tpcusr = NULL;
OCIStmt *curq = NULL;
OCIStmt *cur_dml = NULL;
OCIStmt *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: qexec username/password [q<path name for
query template file>]\n");
  fprintf(stderr,"               [l<path name for log>] [r<path name for
reports>]\n\n");
  fprintf(stderr,"Options:\n");
  fprintf(stderr,"q<path for query>         : full path name for the query
template file.\n");
  fprintf(stderr,"                          (default is stdin)\n");
  fprintf(stderr,"l<path name for log>      : full path name for log
files\n");
  fprintf(stderr,"                          (default is stdout)\n");
  fprintf(stderr,"r<path name for reports>  : full path name for
reports\n");
  fprintf(stderr,"                          (default is stdout)\n");
  exit(-1);
}



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

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

  switch(status) {
  case OCI_SUCCESS_WITH_INFO:
    fprintf(stderr, "Error: Statement returned with info.\n");
    if (type)
      (void) OCIErrorGet(errhp,1,NULL,(sb4*)&errcode,(text*)msg,
                                        2048,OCI_HTYPE_ERROR);
    else
      (void) OCIErrorGet(errhp,1,NULL,(sb4*)&errcode,(text*)msg,
                                        2048,OCI_HTYPE_ENV);
    fprintf(stderr,"%s\n",msg);
    break;
  case OCI_ERROR:
    fprintf(stderr, "Error: OCI call error.\n");
    if (type)
      (void) OCIErrorGet(errhp,1,NULL,(sb4*)&errcode,(text*)msg,
                                        2048,OCI_HTYPE_ERROR);
    else
      (void) OCIErrorGet(errhp,1,NULL,(sb4*)&errcode,(text*)msg,
                                        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 = gettime();

 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] = (dltype *) memalloc (sizeof(dltype));
     dlist[i]->defhdl = NULL;
/*     OCIhalloc(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);

 OCIhalloc(tpcenv,&errhp,OCI_HTYPE_ERROR);
 OCIhalloc(tpcenv,&curq,OCI_HTYPE_STMT);
 OCIhalloc(tpcenv,&cur_dml,OCI_HTYPE_STMT);
 OCIhalloc(tpcenv,&cur_ddl,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(logname, '/');
 *passwd = '\0';
 passwd++;

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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcsrv,0,OCI_ATTR_SERVER
,errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,logname,strlen(logname),OCI_
ATTR_USERNAME,
            errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(passwd),OCI_A
TTR_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_SESSIO
N,errhp);

/*
```

```
  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 = gettime();
    fprintf(logfile,"Query Processed in %.2f seconds.\n\n",
     (s_tr_end - s_tr_start));

    /* print comments for this query that we have saved */

    /* fprintf(logfile, "%s\n", cmnt); */

    /*fprintf(rep, "Query %s : Execution time %.2f\n", qnp,(s_tr_end -
s_tr_start));*/
    fprintf(rep, "Query %s: Execution Time: %.2f started %.2f ended
%.2f\n",
              qnp,(s_tr_end - s_tr_start),s_tr_start,s_tr_end);

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

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

  s_tr_start = gettime();

  /* prepare the statement */

  if ((status = OCIStmtPrepare(curq, errhp, (text*) stmt, (ub4)
strlen(stmt),
                                        OCI_NTV_SYNTAX,
OCI_DEFAULT)) != OCI_SUCCESS)
    sql_error(errhp,status,1);

  /* Prints the query text and comment to the logfile */

  fprintf(logfile, "\n%s\n", cmnt);
  cmnt[0]=0;
  fprintf(logfile, "\n%s\n", stmt);

  /* if this is a DDL or DML statement, execute it right away */
  /* only worries about SELECT statements right now, cannot   */
  /* execute a stored PL/SQL procedure in thie version        */


OCIaget(curq,OCI_HTYPE_STMT,&stmttyp,NULL,OCI_ATTR_STM
T_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 statments will screw it up */


OCIaget(curq,OCI_HTYPE_STMT,&rows_ret,NULL,OCI_ATTR_RO
W_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] = (dltype *) memalloc(sizeof(dltype));
      dlist[i]->defhdl = NULL;
    }

    /* Let's check the sizes and types for this select list item */

    switch (slist[i].dbtype) {

    case OCI_TYPECODE_NUMBER:

      /* The odescr will not give a good estimate to the scale if */
      /* no scale was given in the Oracle table definition.       */

#ifdef HAVE_SCALE
      if (slist[i].scale != 0) {
                defbuf = (double *) dlist[i]->fbuf;
                deflen = FLT;
                deftype = OCI_TYPECODE_DOUBLE;
                slist[i].dbtype = OCI_TYPECODE_DOUBLE;
      } else {
                defbuf = (int *) dlist[i]->ibuf;
                deflen = INT;
                deftype = OCI_TYPECODE_INTEGER;
                slist[i].dbtype = OCI_TYPECODE_INTEGER;
      }
#else
      defbuf = (double *) dlist[i]->fbuf;
      deflen = FLT;
      deftype = OCI_TYPECODE_FLOAT;
      slist[i].dbtype = OCI_TYPECODE_FLOAT;
#endif /* HAVE_SCALE */

      break;

    default:

      /* default is character string */

      defbuf = (char **) dlist[i]->sbuf;
      deflen = MAX_STR_LEN;
      deftype = SQLT_STR;
/*      deftype = OCI_TYPECODE_CHAR; */
      break;
    }

    /* Define the column */

    if ((status=OCIDefineByPos(curq,&(dlist[i]->defhdl),errhp,POS(i),
                                          defbuf,deflen,deftype,NULL,
                                          dlist[i]-
>rlen,NULL,OCI_DEFAULT))!=OCI_SUCCESS)
      sql_error(errhp,status,1);
  }
  return i;
}


/* process_select_list(): Fetch rows from a query.                  */

void process_select_list(num)
    int num;        /* number of select list items */
{

  int i,j;
  int ntf;
  int num_so_far;
  sword stats = OCI_SUCCESS;

  /* Print the headers for the query execution result */

  print_header(num);

  /* See if we need to limit the rows to fetch */

  ntf = (num_to_fetch >= 0) ? num_to_fetch : MAX_ARRAY;

  /* Fetch the rows and print them out */

  if ((ntf > MAX_ARRAY) || (num_to_fetch == -1)) {

    stats = OCIStmtFetch(curq, errhp, MAX_ARRAY,
OCI_FETCH_NEXT, OCI_DEFAULT);


OCIaget(curq,OCI_HTYPE_STMT,&rows_ret,NULL,OCI_ATTR_RO
W_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 =
OCIStmtFetch(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;
    } else {
            ntf -= MAX_ARRAY;

            while ((stats = OCIStmtFetch(curq,errhp,
                                                    ((ntf>MAX_ARRAY) ?
MAX_ARRAY:ntf),
                                                    OCI_FETCH_NEXT,
OCI_DEFAULT)) ==
                  OCI_SUCCESS) {
              ntf -= MAX_ARRAY;
              OCIaget(curq,OCI_HTYPE_STMT,&num_so_far,NULL,
                            OCI_ATTR_ROW_COUNT,errhp);
              print_rows(num,(num_so_far-rows_ret));
              rows_ret = num_so_far;
              if (ntf <= 0) break;
            }
            OCIaget(curq,OCI_HTYPE_STMT,&num_so_far,NULL,
                          OCI_ATTR_ROW_COUNT,errhp);
            print_rows(num,(num_so_far-rows_ret));
            rows_ret = num_so_far;
    }
   }
  } else {
   OCIStmtFetch(curq, errhp, ntf, OCI_FETCH_NEXT,
OCI_DEFAULT);

OCIaget(curq,OCI_HTYPE_STMT,&rows_ret,NULL,OCI_ATTR_RO
W_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 = ' ';
}
```

F.6 qexecpl.h

```
/*
 * $Header: qexecpl.h 13-nov-2001.17:52:35 mpoess Exp $
 */

/* Copyright (c) 1999, 2001, Oracle Corporation.  All rights reserved.  */

/* NOTE:  See 'header_template.doc' in the 'doc' dve under the 'forms'
      directory for the header file template that includes instructions.
*/

/*
   NAME
     qexecpl.h

   DESCRIPTION
     SQL statement execution front-end header file.

   PUBLIC FUNCTION(S)
     <list of external functions declared/defined - with one-line
descriptions>

   PRIVATE FUNCTION(S)
     <list of static functions defined in .c file - with one-line descriptions>

   EXAMPLES

   NOTES
     <other useful comments, qualifications, etc.>

   MODIFIED   (MM/DD/YY)
   mpoess      11/13/01 - change DOP to 84 for DML and DDL
   mpoess      02/22/01 - add linux changes
   mpoess      08/05/99 - make compile
   mpoess      07/15/99 - Creation
   mpoess      07/15/99 - Creation

*/

/*
# ifndef S_ORACLE
#  include <s.h>
# endif
*/
#ifndef QSTREAMPL_H

#define QSTREAMPL_H

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

```
#include <sys/param.h>
#include <sys/types.h>
#include <time.h>
#include <errno.h>
#include <math.h>

#include <oratypes.h>

#include <oratypes.h>

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

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

/* some basic definitions */

#define UNAME_LEN 64
#define MAX_FILE_PATH_LEN 128

#ifndef TRUE
#define TRUE 1
#endif /* TRUE */

#ifndef FALSE
#define FALSE 1
#endif /* FALSE */
#ifndef LINUX
#define MAX(x,y) ((x >= y) ? x : y)
#define MIN(x,y) ((x <= y) ? x : y)
#endif
/* defines and typedefs for parsing */

#define CRT_TBL  1
#define INS_STMT 3
#define SEL_STMT 4
#define UPD_STMT 5
#define DRP_VIEW 7
#define DRP_TBL  8
#define DEL_STMT 9
#define CRT_VIEW 10

/* 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=OCIStmtExecute(svch,stmh,errh,iter,0,NULL,NULL,OCI_DE
FAULT)) != 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 PDDLTXT "alter session force parallel ddl parallel (degree 84)"

#endif /* QSTREAMPL_H */
```

F.7 runTPCHall

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

ECHO=echo

sqlplus=$ORACLE_HOME/bin/sqlplus
GTIME=${KIT_DIR}/utils/gtime

RUN_ID_FILE=${KIT_DIR}/audit/r_id

if [ ! -f $RUN_ID_FILE ]
then
  echo "0" > $RUN_ID_FILE
fi

RUN_ID=`cat $RUN_ID_FILE`
RUN_ID=`expr $RUN_ID + 1`
echo $RUN_ID > $RUN_ID_FILE

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

SCRIPT_LOG_FILE=${OUT_DIR}/main.out
RDB_TABLES=${OUT_DIR}/rdbtablest
FIRST_TEN=${OUT_DIR}/firstten

LD0LOADASM=${OUT_DIR}/Ld0loadasm
LD1DBCRE=${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.$R
UN_ID
mv $ORACLE_HOME/rdbms/log/alert_ASM.log
$ORACLE_HOME/rdbms/log/alert_ASM.log.preAudit.$RUN_ID
touch $ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log
touch $ORACLE_HOME/rdbms/log/alert_ASM.log


echo "Start: load database `date`" >> $SCRIPT_LOG_FILE
loadasm > $LD0LOADASM
dbcre.sh > $LD1DBCRE
sctso.sh > $LD2SCTSO
STIME=`$GTIME`
echo "Start: timed load portion `date`" >> $SCRIPT_LOG_FILE
dapop.sh > $LD3DAPOP
ixcre.sh > $LD4IXCRE
anl.sh > $LD5ANLYZ
echo "End: timed load portion `date`" >> $SCRIPT_LOG_FILE

$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
# call the auditor: don't 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
```

F.8 runTPCHpt

```
#!/bin/ksh
. $KIT_DIR/env
#set -x
#ECHO=/bin/echo
SCRIPT_DIR=${KIT_DIR}/scripts
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>] [-u1 <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 "-u1 <program>    : Program for UF1."
echo "                   Default is $U1PROG."
echo "-u2 <program>    : Program for UF2."
echo "                   Default is $U2PROG."
echo "-o               : Collect Oracle statistics."
echo "-s               : Collect System statistics."
echo "-u <user/passwd> : User/Password. Default is tpch/tpch."
echo "-h               : Displays this message."
}
set -- `getopt "p:u1:u2:osu:h" "$@"` || usage

while :
do
  case "$1" in
  -u1) shift; U1PROG=$1;;
  -u2) shift; U2PROG=$1;;
  -p) shift; QPROG=$1;;
# not needed ?    -o) OSTAT=1;;
# not needed ?   -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}tstrcnt

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 UF1: Execution Time: $UF1_TIME >> $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}  > $DF  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`

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}


echo "End UF2 $UF2_END, $EDATE" >>  $SCRIPT_LOG_FILE
echo UF2: Execution Time: $UF2_TIME >> $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

#${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 throughtput run
(scnt.sh $PARA $RUN_ID > $STREAM_COUNT_LOG &)

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
PIDS=`ps -fu oracle | grep scnt.sh | grep -v grep | awk '{print $2}'`
kill -9 $PIDS
#calculate the metric
#analyze_streams.pl -f p -n $RUN_ID >
${TPCD_RPT}/tpch_metric.${RUN_ID}.${HID}.rpt
```

F.9 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
```

F.10 runuf1.sh

```
#!/bin/ksh
#
# $Header: runuf1.sh 25-oct-2001.15:56:04 mpoess Exp $
#
# runuf1.sh
#
# Copyright (c) 1999, 2001, Oracle Corporation.  All rights reserved.
#
#   NAME
#     runuf1.sh - <one-line expansion of the name>
#
# DESCRIPTION
#   runuf1.sh -l [<path name for reports>] -u [<uid/passwd>]
#           -p [<program>] <run_id> <scale factor> <pair number>
#           <parallelism>
# USAGE
#   To execute UF1.
#
#   NOTES
#     <other useful comments, qualifications, etc.>
#
#   MODIFIED   (MM/DD/YY)
#
#
. $KIT_DIR/env
O=${ORACLE_HOME}
UPDATE_DIR=${KIT_DIR}/update
SCRIPT_DIR=${UPDATE_DIR}/scripts
UTILS_DIR=${KIT_DIR}/utils
LOG_DIR=${UPDATE_DIR}/log
GTIME=${UTILS_DIR}/gtime
SF=${SCALE_FACTOR}
PAR_HINT=${UPDATE_1_DOP}

LOGPATH=.
PASSWD=${DATABASE_USER}

if [ $# -lt 1 ];
then
  echo runuf1.sh setnum
  exit 1
fi
SETNUM=$1
i=1
```

```
PID=""

# perform the update function 1

START=`$GTIME`

# first create the temp tables

sqlplus /NOLOG << !

connect $PASSWD;
set timing on
set serveroutput on
set echo on

drop directory data_dir;
create directory data_dir as '/flat15/updates';

drop table temp_l_et;
create table temp_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_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  ${PAR_HINT};

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  ${PAR_HINT};


alter session force parallel dml parallel (degree ${PAR_HINT});
alter session set isolation_level = serializable;
alter session set  optimizer_index_cost_adj=10;


insert into orders
select
   o_orderdate         ,
   o_orderkey          ,
   o_custkey           ,
   o_orderpriority     ,
   o_shippriority      ,
   o_clerk             ,
   o_orderstatus       ,
   o_totalprice        ,
   o_comment
from temp_o_et;


insert into  lineitem
select
   l_shipdate          ,
   l_orderkey          ,
   l_discount          ,
   l_extendedprice     ,
   l_suppkey           ,
   l_quantity          ,
   l_returnflag        ,
   l_partkey           ,
   l_linestatus        ,
   l_tax               ,
   l_commitdate        ,
   l_receiptdate       ,
   l_shipmode          ,
   l_linenumber        ,
   l_shipinstruct      ,
   l_comment
from temp_l_et;

commit;

rem drop table temp_l_et;
rem drop table temp_o_et;

exit;
!

END=`$GTIME`

# Done

echo ""
echo "Update Function 1 Set $SETNUM done!"
echo "Elapsed Time is `echo $END - $START | bc`"
echo ""
```

F.11 runuf2.sh

```
#!/bin/ksh
#
# $Header: runuf2.sh 25-oct-2001.15:56:05 mpoess Exp $
#
# runuf2.sh
#
# Copyright (c) 1999, 2001, Oracle Corporation.  All rights reserved.
#
#   NAME
#     runuf2.sh - <one-line expansion of the name>
#
# DESCRIPTION
#    runuf2.sh [-u <uid/passwd to login>] [-p <program>] <run_id>
#              <scale factor> <pair number> <parallelism>
# USAGE
#    To execute UF2.
#
#    NOTES
#      <other useful comments, qualifications, etc.>
#
#
. $KIT_DIR/env
UPDATE_DIR=${KIT_DIR}/update
SCRIPT_DIR=${UPDATE_DIR}/scripts
UTILS_DIR=${KIT_DIR}/utils
GTIME=${UTILS_DIR}/gtime
LOG_DIR=${UPDATE_DIR}/log
PAR_HINT=${UPDATE_2_DOP}
SF=${SCALE_FACTOR}
PASSWD=${DATABASE_USER}

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

SETNUM=$1

i=1
PID=""

START=`$GTIME`
# first create the temp tables

sqlplus /NOLOG << !

connect $PASSWD;
set timing  on
set serveroutput on
set echo on

drop directory data_dir;
create directory data_dir as '/flat15/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
              nobadfile
              nologfile
              fields terminated by '|'
              missing field values are null
       )
       location (
       'delete.${SETNUM}'))
reject limit unlimited parallel 16;


create table temp_okey (t_orderkey, constraint tokey1 primary
key(t_orderkey))
organization index parallel  16 nologging as select * from
temp_okey_et;
execute dbms_stats.gather_table_stats('tpch' , 'temp_okey',
estimate_percent => 1, degree => 16)


alter session  force parallel dml parallel ${PAR_HINT};
alter session set isolation_level=serializable;
alter session set optimizer_index_cost_adj=1;


delete from (select /*+ use_nl(o) */ o.rowid from orders o, temp_okey t
where o.o_orderkey = t.t_orderkey order by 1);

delete from (select /*+ use_nl(l) */ l.rowid from lineitem l,temp_okey t
where l.l_orderkey = t.t_orderkey order by 1);

commit;

drop table temp_okey;
drop table temp_okey_et;
exit;
!

END=`$GTIME`

# Done

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

F.12 scnt.sh

```
#!/bin/ksh

echo Process count for TPC-H RUN:$1 SEQUENCE:$2
while [ 1 = 1 ]; do
  cnt=`ps -ef | egrep "qexec|runTPCHus" | grep -v grep | wc -l`
              echo
       echo `date` : $cnt
              ps -ef | egrep "qexec|runTPCHus" | grep -v grep
              sleep 30
done
```

F.13 set_queue

```
#!/sbin/sh

#set -x

#
# set queue_depth
#

for i in `ls -1 /dev/rdisk/*`
```

```
do
  #scsimgr set_attr  -C disk -I $i -a max_q_depth=128
  #scsimgr set_attr  -D $i -a max_q_depth=128
  scsimgr set_attr  -D $i -a max_q_depth=128 > /dev/null 2>&1

done


exit
```

F.14 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 abort
exit
!
fi

sleep 5

exit
```

F.15 tshut.asm

```
#!/bin/ksh


export ORACLE_SID=ASM


sqlplus /NOLOG<< !
connect / as sysdba;
shutdown normal;
exit
!

sleep 5

exit
```

F.16 tstart

```
#!/bin/ksh

export ORACLE_SID=tpch

mpsched -P RR sqlplus /NOLOG << !
connect / as sysdba
startup pfile=/oracle/dbs/10TB_init.ora
execute dbms_scheduler.disable('AUTO_TASKS_JOB_CLASS');
!
sleep 5
/Lvm/set_queue;
exit
```

F.17 tstart.asm

```
#!/bin/ksh

export ORACLE_SID=ASM

mpsched -P RR sqlplus /NOLOG << !
connect / as sysdba
startup pfile=/oracle/dbs/initasm.ora mount
!

exit
```

Price Quotes

The following pages contain the price quotes for the hardware included in this FDR.

Sharada Bose
HP
Cupertino, CA 95014
November 30, 2006

HP Unix Sales Development
19111 Pruneridge Avenue
Cupertino, CA 95014
(408) 447-2320

Description	Part Number	Source	Reference Price	Qty	Extended Price	3 yr. Maint. Price
Server Hardware						
Superdome left chassis	A9834A, Opt 429	1	235,950	1	235,950	
Superdome right chassis	A9835A, Opt 429	1	249,950	1	249,950	
Superdome sx2000 Cell Board	A9837A	1	19,250	16	308,000	
24x7x4hr - 3 Year Svc & Support Price (Hardware and Software)						$1,379,874
256GBBundle SDRAM (128x2GB DIMMS)	A9856A	1	720,896	2	1,441,792	
12-Slot PCI-X I/O Chassis	A9836A	1	16,950	16	271,200	
Dual-Core Intel Itanium2 9040/1.6GHz/18MB L3	AB406A	1	23,000	64	1,472,000	
PCI 1000BT Lan Adapter	A6847A, Opt. 0D1	1	1,325	1	1,325	
PCI 4GB Fibre Channel Adapter (dual port)	AB379A	1	3,995	128	511,360	
PCI Dual Channel Ultra320 SCSI Adapter	A7173A	1	795	1	795	
HPDisk System 2120	A7382A	1	1,070	1	1,070	
1-36GB LP 15K HDD	A7527A	1	966	4	3,864	
HP Universal Rack 10642 G2 Pallet Rack	AF001A	1	1,249	1	1,249	
Modular Power Dist Unit for std racks	A5137AZ	1	145	1	145	
200-240 volts North America	A5137AZ, Opt AW4	1	94	1	94	
HP Tape Array 5300 (DVD and DAT tape)	C7508B	1	729	1	729	
HP rx2620 Server Solution (inc memory, disk, monitor, k	AB333A	1	5,315	1	5,315	
I/O Chassis Enclsoure for 12-Slot PCI-X Chassis	A9852A	1	25,750	4	103,000	
Graphite I/O expansion power subsystem	A5861D	1	34,860	2	69,720	
				Subtotal	**4,677,558**	**1,379,874**
Server Software						
HPUX 11i, v3 Foundation Operating Environment	B9429AC	1	2,370	128	303,360	
HP-UX 11i v3 HP9000/Integrity FOE Media	BA489AA	1	565	1	565	
					303,925	**0**
Storage						
5m Fibre Channel Cables	221692-B22	1	82	256	20,992	
HP StorageWorks MSA 1000 (256 + 26 spares)	201723-B22	1	6,995	282	1,972,590	
3 Yr Support Price for MSA1000 and disks						Included
36GB 15K Ultra320 Hard Drive (3072 + 308 spares)	286776-B22	1	269	3,380	909,220	
HP Universal Rack 10642 G2 Pallet Rack	AF001A	1	1,249	28	35,222	
Modular Power Dist Unit for std racks	A5137AZ	1	145	113	16,356	
200-240 volts North America	A5137AZ, Opt AW4	1	94	113	10,603	
ProLiant Cluster HA/200 for MSA1000	252409-B22	1	1	4007	4,007	
				Subtotal	**2,968,990**	**0**
				Total	**7,950,473**	**1,379,874**
Large Configuration Discount and Support Prepayment*					(2,755,882)	(538,531)
				Grand Total	**5,194,591**	**841,343**
					3-yr Cost of Ownership:	**6,035,934**

Maintenance support price is for 24 hours, 7 days with 4 hour response time.

This quote is valid for 90 days

From: Mary.Beth Pierantoni [mailto:mary.beth.pierantoni@oracle.com]
Sent: Thursday, November 02, 2006 12:25 PM
To: Boushey, Lucille
Cc: Glasstone,Ray
Subject: Oracle Quote

Product	Price	Qty	Extended Price
Oracle Database 10g Release 2 Enterprise Edition with Automatic Storage Management for 3 years, Named User Plus	**$10,000**	**64***	**$640,000**
Partitioning for 3 years, Named User Plus	**$2,500**	**64***	**$160,000**
Database Server Support Package for 3 years	**$6,000**	**1**	**$6,000**
Oracle Mandatory E-Business Discount			**<$161,200>**
Oracle TOTAL			**$644,800**

(* 64 = 0.50 * 128). Explanation: For the purposes of counting the number of processors which require licensing, an Intel multicore chip with "n" cores shall be determined by multiplying "n" cores by a factor of 0.50).
Contact: Herve Lejeune, Herve_Lejeune@oracle.com , 650-506-1894

This quote is valid for 60 days.