HP Integrity rx8640
using
HP-UX 11.i v2 64-bit
and
Oracle Database 10g Release 2 Enterprise Edition with Partitioning

TPC Benchmark™ H
Full Disclosure Report

First Edition

August 4, 2006

First Edition - August 4, 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., August 4, 2006.

Overview

This report documents the methodology and results of the TPC Benchmark™ H test conducted on the HP Integrity rx8640 , 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 11.i v2 64-bit; the DBMS was Oracle 10g Release 2.

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. 1000 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

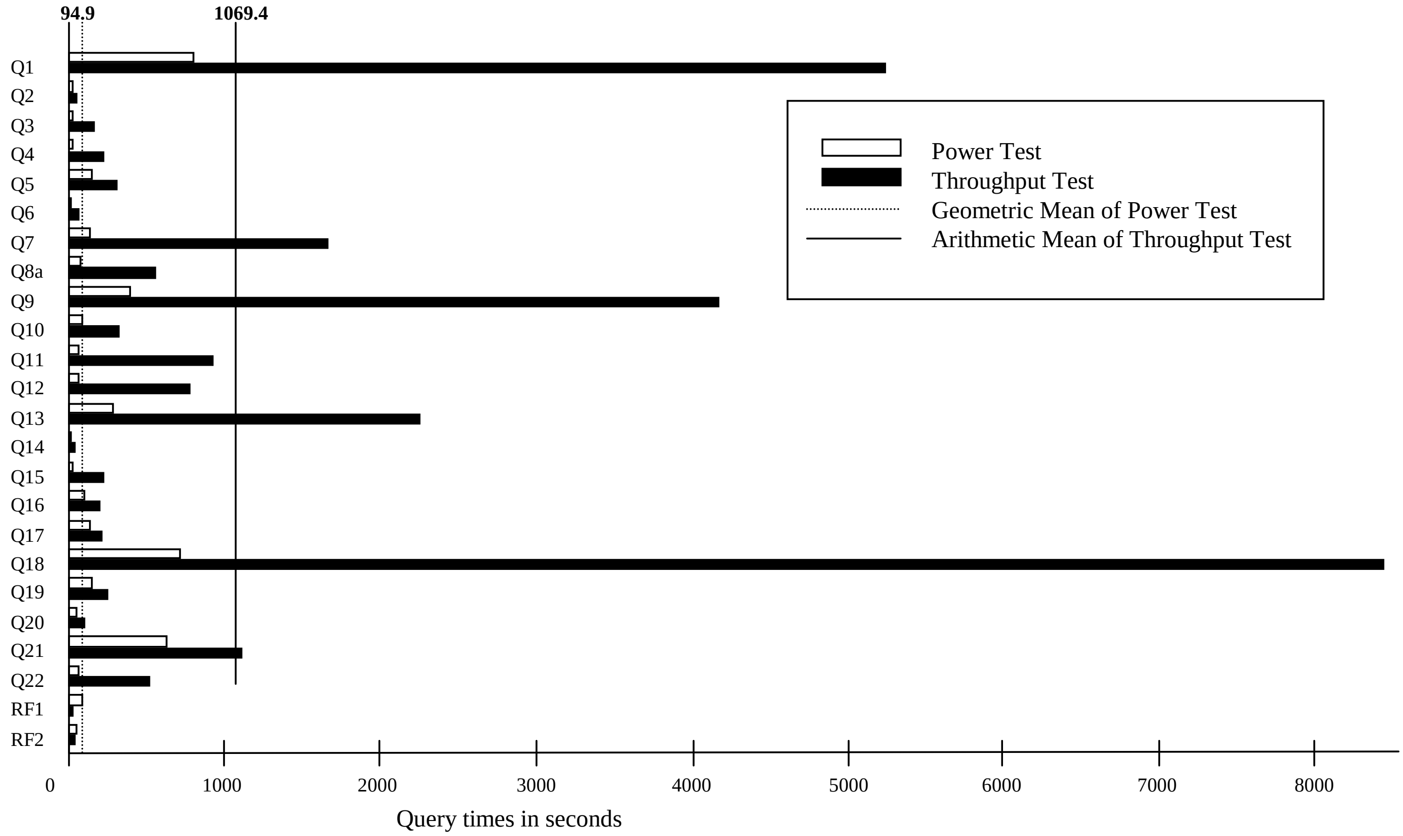
The purpose of TPC benchmarks is to provide relevant, objective performance data to industry users. To achieve that purpose, TPC benchmark specifications require that benchmark tests be implemented with systems, products, technologies and pricing that:

- 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 rx8640	TPC-H Rev 2.3.0 Report Date: August 4, 2006
Total System Cost	Composite Query per Hour Metric	Price/Performance
$984,810	**27,143.6** **QphH@1000GB**	**$36** **QphH@1000GB**

Database Size	Database Manager	Operating System	Other Software	Availability Date
1000 GB*	**Oracle Database 10g Release 2 Enterprise Edition with Partitioning**	**HP-UX 11.i v2 64-bit**	**None**	**January 1, 2007**



Database Load Time = 04:27:42	Load Includes Backup: N	Total Data Storage/Database Size = 17.39
RAID (Base Tables Only): N	RAID (Base Tables and Auxiliary Data Structures): N	RAID (All): Y

System Configuration

Processors:	8 Dual-core Intel Itanium2 Processors 9050 1.6GHz, 18MB
Memory:	128 GB
Disk Drives:	3 Internal disks (36GB each) disks plus 40 HP StorageWorks MSA1000 (with a total of 480 36GB 15K RPM disks)
Total Disk Storage	17388GB (In this calculation 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.)

TPC-H Rev 2.3.0

HP Integrity rx8640

Report Date: August 4, 2006

Description	Part Number	Source	Reference Price	Qty	Extended Price	3 yr. Maint. Price
Server Hardware						
HP Integrity rx8640 SMP Base System with 8 1.6GHz/18MB Dual-core Processor Module	AB444A#002**	1	191,108	1	191,108	
4GB high-density DDR memory module (uses 2 DIMM)	AB454A	1	8,000	32	256,000	
HP Integrity rx8640 I/O Backplane	AD160A	1	7,700	2	15,400	
3 Year Svc & Support Price (Hardware and Software)	HA110A3-6KT**	1				$106,561
HP Integrity rx8640 Sys. Expansion Unit	AB301A	1	35,785	1	35,785	
HP Rack kit for rx86xx Server	J1528B	1	582	1	582	
HP Rack Kit for SEU Server	J1530C	1	709	1	709	
DVD+RW Drive	AB351B	1	850	1	850	
PCI 1000 Base T Dual Port LAN Adapter Card	A7012A		1,495	1	1,495	
PCI 2GB Fibre Channel Adapter (dual port)	A6826A	1	4,395	20	87,900	
36 GB HotPlug Ultra 320 SCSI Low Profile Disk (15k)	AD146A	1	1,200	3	3,600	
HP Server Thin Client (Console)	AB300B	1	1,250	1	1,250	
Rack Model 5642	358254-B21	1	689	1	689	
				Subtotal	**595,368**	**106,561**
Server Software						
Oracle Database 10g Release 2 Enterprise Edition, Named User Plus		2		8*	80,000	
Partitioning for 3 years, Named User Plus		2		8*	20,000	
Oracle Database Server Support Package for 3 years:		2		3		6,000
HPUX 11i v2 Foundation Operating Environment	B9429AC**	1	2,370	16	37,920	
HPUX Fndn OE Media	B9106AA, Opt 0D1	1	199	1	199	
				Subtotal	**138,119**	**6,000**
Storage						
16 meter Fibre Optic Cable	221692-B22	1	82	40	3,280	
HP StorageWorks MSA 1000 (40+4 spares)	201723-B22	1	6,995	44	307,780	Included
36GB 15K Ultra320 Hard Drive (480 + 48 spares)	286776-B22	1	269	528	142,032	Included
10642 (42U) Rack Cabinet	245161-B21		1,359	4	5,436	
ProLiant Cluster HA/200 for MSA100	252409-B22	1	4,007	1	4,007	
				Subtotal	**462,535**	**0**
				Total	**1,196,022**	**112,561**
Oracle Mandatory E-Business Discount on (Licenses and Support)					(15,900)	
Large Configuration Discount and Support Prepayment*					(276,437)	(31,435)
				Grand Total	**903,685**	**81,126**

3-yr Cost of Ownership:	**984,810**
QphH@1000GB:	**27,143.6**
$/QphH@1000GB:	**$ 36**

Source: 1=HP,
2=Oracle (Pricing Contact: MaryBeth Pierantoni; email: mary.beth.pierantori@oracle.com; phone number: (916-315-5081)

*8 = 0.50 * 16. 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.

** These components are not immediately orderable. See the FDR for more information.

A 25.6% discount was based on the overall value of the specific components from HP (Price Key) in this single quotation. Discounts for similarly sized configurations will be similar to those quoted here, but may vary based on the components in the quotation.

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.

HP Integrity rx8640

TPC-H Rev 2.3.0

Report Date: August 4, 2006

Measurement Results

Database Scaling (SF/size)	1000
Total Data Storage/Database Size	17.39
Start of Database Load Time	2006-07-29 22:43:01
End of Database Load Time	2006-07-30 03:10:43
Database Load Time	04:27:42
Query Streams for Throughput Test (S)	7
TPC-H Power	37,931.2
TPC-H Throughput	19,424.0
TPC-H Composite Query-per-Hour Metric (QphH@1000GB)	27,143.6
Total System Price Over 3 Years	984,810
TPC-H Price/Performance Metric ($/QphH@1000GB)	$36

Measurement Intervals

Measurement Interval in Throughput Test (Ts)	28,542

Duration of Stream Execution:

	SEED	Start Date/Time	End Date/Time	Duration
Stream 00	730031043	30-Jul-2006 12:31:26	30-Jul-2006 13:42:44	1:11:18
Stream 01	730031044	30-Jul-2006 13:42:45	30-Jul-2006 20:43:54	7:01:09
Stream 02	730031045	30-Jul-2006 13:42:46	30-Jul-2006 18:16:42	4:33:56
Stream 03	730031046	30-Jul-2006 13:42:46	30-Jul-2006 18:50:31	5:07:45
Stream 04	730031047	30-Jul-2006 13:42:46	30-Jul-2006 20:54:15	7:11:29
Stream 05	730031048	30-Jul-2006 13:42:46	30-Jul-2006 21:29:38	7:46:52
Stream 06	730031049	30-Jul-2006 13:42:46	30-Jul-2006 21:26:45	7:43:59
Stream 07	730031050	30-Jul-2006 13:42:46	30-Jul-2006 20:02:24	6:19:38
Refresh		30-Jul-2006 21:29:38	30-Jul-2006 21:38:27	0:08:49

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	801.8	32.6	25.4	27.9	156.8	20.1	140.0	83.4	403.0	91.2	61.2	65.4
Stream 01	1765.8	45.8	162.2	75.1	2155.9	51.1	435.6	237.1	947.8	258.6	295.4	172.8
Stream 02	1653.8	109.6	68.9	83.9	450.9	44.6	259.7	539.7	3736.4	1404.7	360.6	288.0
Stream 03	2126.7	66.7	94.9	90.1	2391.6	42.8	383.7	305.4	3911.1	1359.6	722.0	213.5
Stream 04	5079.2	104.2	84.3	260.8	478.5	51.8	240.8	596.3	3983.5	1163.2	567.7	365.2
Stream 05	4144.2	63.3	168.9	440.3	289.6	72.9	1421.5	151.6	845.1	222.7	418.6	240.8
Stream 06	5233.1	47.8	166.5	229.0	319.0	67.1	1658.6	562.8	4165.8	321.9	919.4	779.9
Stream 07	6285.7	79.5	52.9	94.4	484.1	40.2	477.8	531.4	1274.0	277.7	2342.5	335.0
Minimum	1653.8	45.8	52.9	75.1	289.6	40.2	240.8	151.6	845.1	222.7	295.4	172.8
Average	3755.5	73.8	114.1	182.0	938.5	52.9	696.8	417.8	2694.8	715.5	803.8	342.2
Maximum	6285.7	109.6	168.9	440.3	2391.6	72.9	1658.6	596.3	4165.8	1404.7	2342.5	779.9

Stream ID	Q13	Q14	Q15a	Q16	Q17	Q18	Q19	Q20	Q21	Q22	RF1	RF2
Stream 00	293.6	19.9	29.4	109.5	136.2	731.4	155.7	50.5	636.3	62.2	88.5	55.3
Stream 01	914.2	49.3	175.3	608.9	275.7	7382.9	453.1	112.5	8508.7	185.8	24.7	45.2
Stream 02	654.3	62.3	388.0	1459.6	660.5	1619.2	500.9	150.8	1779.4	160.6	24.5	45.8
Stream 03	753.4	95.9	313.9	476.1	320.8	1776.9	484.5	156.5	1802.6	576.2	25.4	45.6
Stream 04	962.9	61.1	236.2	1388.1	229.4	7994.7	461.8	71.9	1318.1	188.9	26.0	45.8
Stream 05	2016.7	65.6	301.4	905.6	180.6	7384.2	512.3	94.1	7645.0	424.8	25.7	77.2
Stream 06	2261.2	44.6	228.8	209.3	214.2	8417.9	255.0	98.9	1108.9	529.7	25.5	45.6
Stream 07	648.5	50.0	214.3	658.2	353.4	6093.3	382.2	680.3	1240.9	181.4	26.1	45.5
Minimum	648.5	44.6	175.3	209.3	180.6	1619.2	255.0	71.9	1108.9	160.6	24.5	45.2
Average	1173.0	61.2	265.4	815.1	319.2	5809.9	435.7	195.0	3343.4	321.0	25.4	50.1
Maximum	2261.2	95.9	388.0	1459.6	660.5	8417.9	512.3	680.3	8508.7	576.2	26.1	77.2

INFO SIZING

Benchmark Sponsor:

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

Ray Glasstone
Manger, DSS Performance
Oracle Corporation
100 Oracle Parkway 1911
Redwood Shores, CA

August 3, 2006

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

Platform: **HP Integrity rx8640**

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

Operating System: HP-UX 11.i V2 64-bit

The results were:

CPU (Speed)	Memory	Disks	**QphH@1000GB**
HP Integrity rx8640			
8 x Itanium2 9050 (1.6GHz, dual-core)	18 MB Cache/cpu 128 GB Main	480 x 36GB ext. 3 x36GB int.	**27,143.6**

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 1,000GB 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 7 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 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.

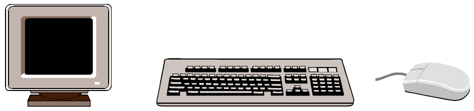
Measured Configuration:

- 8 1.6GHz Dual-core Intel Itanium2 Processors 9050 CPUs each with 18MB
- 128 GB Memory
- 3 36GB 15K Ultra 320 SCSI Internal disks
- 20 PCI Fibre Channel 2X (dual-port) Cards
- 1 HP 1000 BaseSX PCI Lan Adapters
- 40 HP StorageWorks MSA1000 (with a total of 480 36GB Disks)
- 1 DVD/RW
- 1 SCSI Card

Priced Configuration:

- 8 1.6GHz Dual-core Intel Itanium2 Processors 9050 CPUs each with 18MB
- 128 GB Memory
- 3 36GB 15K Ultra 320 SCSI Internal disks
- 20 PCI Fibre Channel 2X (dual-port) Cards
- 1 HP 1000 BaseSX PCI Lan Adapters
- 40 HP StorageWorks MSA1000 (with a total of 480 36GB Disks)
- 1 DVD/RW
- 1 SCSI Card

Priced Configuration



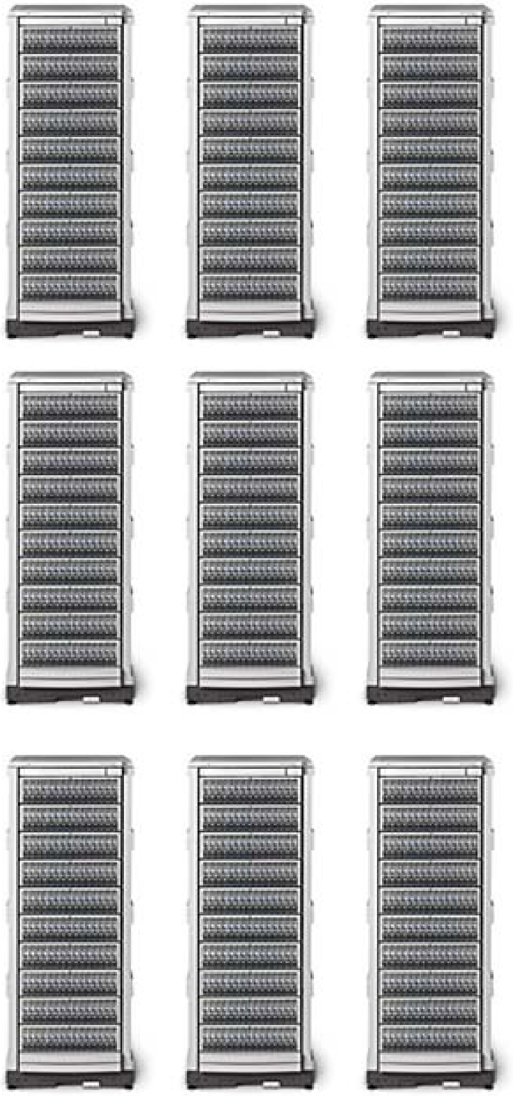
HP Integrity rx8640 Server

summary

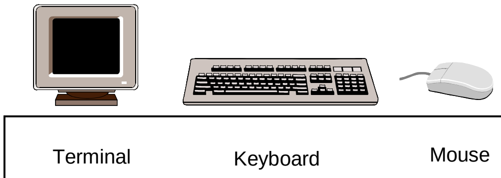
WITH:

8 – 1.6GHz\18MB Dual-Core Itanium2 Processors 9050
128GB Memory
20 PCI 2GB Fibre Channel Adapter (dual-port)
1 HP 1000 BaseSX PCI Lan Adapter
3 36 GB Ultra 320 SCSI Low Profile Disk (15k)
1 HP Integrity rx8640 Sys. Expansion Unit
1 DVD+RW Drive

40 HP StorageWorks MSA1000
with 480 36GB 15k RPM Disks



Measured Configuration



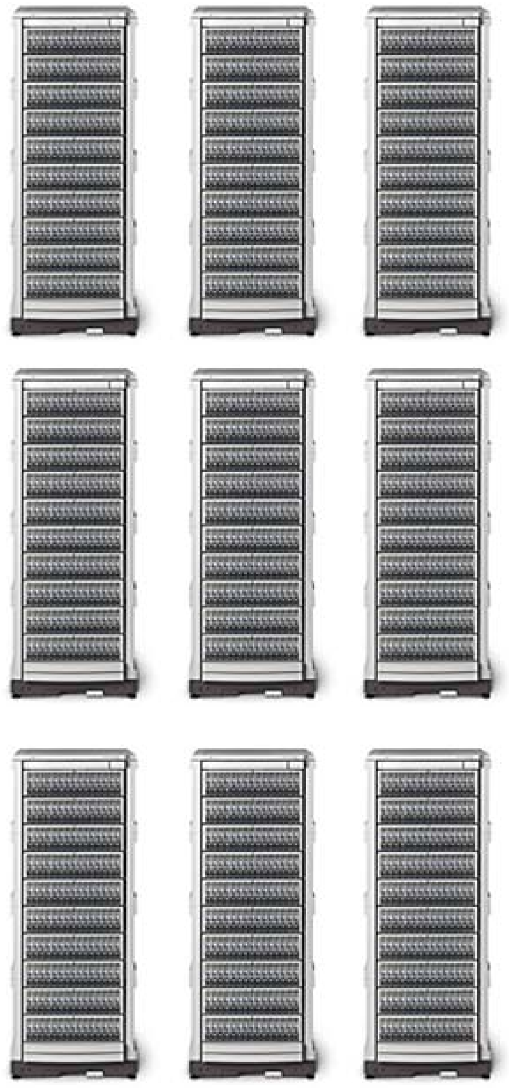
HP Integrity rx8640 Server

summary

WITH:

8 – 1.6GHz\18MB Dual-Core Itanium2 Processors 9050
128GB Memory
20 PCI 2GB Fibre Channel Adapter (dual-port)
1 HP 1000 BaseSX PCI Lan Adapter
3 36 GB Ultra 320 SCSI Low Profile Disk (15k)
1 HP Integrity rx8640 Sys. Expansion Unit
1 DVD+RW Drive

40 HP StorageWorks MSA1000
with 480 36GB 15k RPM Disks



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.

1. The total price from the ORDERS table and the extended price from the LINEITEM table were retrieved for a randomly selected order key.
2. The ACID Transaction was performed using the order key from step 1. The transaction was stopped prior to the commit.
3. The ACID Transaction was ROLLED BACK.
4. The total price from the ORDERS table and the extended price from the LINEITEM table were retrieved for the same order key. It was verified that the appropriate rows had not been changed.

4.3 Consistency

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

Consistency Test

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

1. The consistency of the ORDERS and LINEITEM tables was verified based on a sample of order keys.
2. 100 ACID Transactions were submitted from each of 7 execution streams.
3. The consistency of the ORDERS and LINEITEM tables was re-verified.

4.4 Isolation

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

Read-Write Conflict with Commit

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

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

Read-Write Conflict with Rollback

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

1. An ACID Transaction was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID Transaction was suspended prior to ROLLBACK.
2. An ACID Query was started for the same O_KEY used in step 1. The ACID Query did not see the uncommitted changes made by the ACID Transaction.
3. The ACID Transaction was ROLLED BACK.
4. The ACID Query completed.

Write-Write Conflict with Commit

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

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

Write-Write Conflict with Rollback

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

1. An ACID Transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID transaction T1 was suspended prior to ROLLBACK.
2. Another ACID Transaction, T2, was started using the same O_KEY and L_KEY and a randomly selected DELTA.
3. T2 waited.
4. T1 was allowed to ROLLBACK and T2 completed.
5. It was verified that T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE.

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.

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

2. Another ACID transaction, T2 was started using 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.
3. ACID Transaction T2 completed.
4. T1 was allowed to COMMIT.
5. 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.

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

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 RAID groups 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 matched.

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	1,500,000,000
LINEITEM	5,999,989,709
CUSTOMER	150,000,000
PART	200,000,000
SUPPLIER	10,000,000
PARTSUPP	800,000,000
NATION	25
REGION	5

5.2 Distribution of Tables and Logs Across Media

Distribution of tables and logs across media:

Each MSA1000 array (with 12 disks) was configured into 4 Raid-1/0 luns.

LUN1 for TPCH/Oracle ASM use
LUN2 for flat files
LUN3 for swap
LUN4 qual/acid database

Forty luns, one from each MSA1000 array, were allocated for Oracle ASM use and a single disk group was built across all 40 luns. All tables, indexes, temp space and other Oracle files were configured in this disk group.

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

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 log, data, temp, index, and all other files.

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 04:27:42.

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
Internal	3	36	108
40 HP StorageWorks MSA1000	480	36	17,280.0
TOTAL			**17,388.0**
Scale Factor			**1,000**
Storage Ratio			**17.39**

5.8 Database Load Mechanism Details and Illustration

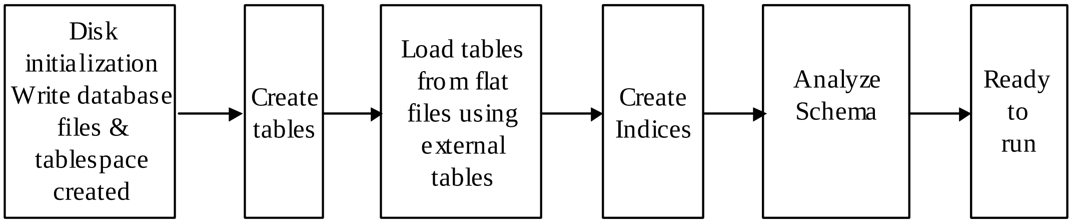
The details of the database load must be described, including a block diagram illustrating the overall process.

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.


Disk initialization Write database files & tablespace created
Create tables
Load tables from flat files using external tables
Create Indices
Analyze Schema
Ready to run

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:

1. Database started
2. RF1 Refresh Transaction
3. Stream 00 Execution
4. 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.

7 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.

		RF1		RF2	
Number	Date	Start	End	Start	End
1	7/30/2006	21:29:38	21:30:03	21:30:03	21:30:48
2	7/30/2006	21:30:48	21:31:13	21:31:13	21:31:58
3	7/20/2006	21:31:59	21:32:24	21:32:24	21:33:10
4	7/30/2006	21:33:10	21:33:36	21:33:36	21:34:21
5	7/30/2006	21:34:21	21:34:47	21:34:47	21:36:04
6	7/20/2006	21:36:04	21:36:30	21:36:30	21:37:16
7	7/30/2006	21:37:16	21:37:42	21:37:42	21:38:27

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@1000GB	QthH@1000GB	QphH@1000GB
Reported Run	37,931.2	19,424.0	27,143.6
Reproducibility Run	38,905.1	19,594.3	27,610.1
% Difference	2.6%	0.9%	1.7%

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.

The database was restarted 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 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.

Oracle is available now

Description	Part Number	Order ability Date	Availability Date
HP Integrity rx8640 SMP Base System with 8 1.6GHz/18MB Dual-core Processor Module	AB444A#002**	August 1, 2006	January 1, 2007
3 Year Svc & Support Price (Hardware and Software)	HA110A3-6KT**	August 1, 2006	January 1, 2007
HP-UX 11i v2 Foundation Operating Environment	B9429AC	December 1, 2006	January 1, 2007

For HP pricing verification, please contact HP Unix Sales Development at 408-447-2320

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

1TB-init.ora

```
aq_tm_processes              = 0
audit_trail                  = FALSE
compatible                   = 10.1.0.2
control_files                =
(/dbms/10gR2/ctrl/rctrl1,/dbms/10gR2/ctrl/rctrl2)
cpu_count                    = 32
db_block_checksum            = false
db_block_size                = 16384
db_cache_size                = 10g
db_file_multiblock_read_count   = 128
db_files                     = 600
db_name                      = 1000gb
db_writer_processes          = 4
dml_locks                    = 40000
global_names                 = FALSE
hpux_sched_noage             = 180
instance_name                = tpch
log_buffer                   = 33554432
log_checkpoint_interval      = 2147483647
log_checkpoints_to_alert     = true
max_dump_file_size           = unlimited
nls_date_format              = YYYY-MM-DD
open_cursors                 = 1024
optimizer_features_enable    = 10.2.0.1.1
optimizer_index_cost_adj=200
optimizer_mode               = CHOOSE
parallel_execution_message_size = 65535
parallel_max_servers         = 640
parallel_min_servers         = 640
pga_aggregate_target               = 50g
processes                    = 1024
recovery_parallelism         = 32
replication_dependency_tracking = false
shared_pool_size             = 15g
statistics_level             = basic
undo_management                    = auto
undo_retention               = 200000
```

initASM.ora

```
ASM.instance_number=1
ASM_DISKSTRING='/dev/rdsk/*t0d0'
asm_diskgroups=DG3
background_dump_dest='/oracle/rdbms/log'
core_dump_dest='/oracle/rdbms/log'
db_cache_size=500m
instance_number=1
instance_type=asm
lock_sga=TRUE
processes=500
shared_pool_size=500m
user_dump_dest='/oracle/rdbms/log'
```

system

```
*
* Module entries
*
module msrset loaded 0.1.0
module root best [44772574]
module cell best [44772574]
module sba best [443FE98D]
module lba best [443FE9A0]
module tgt best [443FE990]
module sdisk best [443FE990]
module sctl best [443FE990]
module asio0 best [443FE9C4]
module azusa_psm best [44772574]
module pty0 best [443FE953]
module pty1 best [443FE953]
module LCentIf best [443FE9C4]
module acpi_node best [443FE9A4]
module sac best [443FE9C4]
module wxb_hp best [443FE9C4]
module ia64_psm best [44772574]
module lion_psm best [44772574]
module pdh best [44772574]
module c8xx best [443FE990]
module diag2 best [443FE990]
module dmem best [443FE9A4]
module dev_config best [443FE98D]
module cdfs best 0.1.0
module rng loaded 0.1.0
module inet best [443FE9B7]
module uipc best [443FE9B3]
module tun best [4133B744]
module telm best [412E8D79]
module tels best [412E8D79]
module netdiag1 best [443FE9A9]
module btlan best [412E8A46]
module intl100 best [412E8A84]
module dlpi best [412E9113]
module token_arp best [412E9113]
module nms best [443FE9B7]
module hpstreams best [41C37402]
module clone best [41C37402]
module strlog best [41C37402]
module sad best [41C37402]
module echo best [41C37402]
module sc best [41C37402]
module timod best [41C37402]
module tirdwr best [41C37402]
module pipedev best [41C37402]
module pipemod best [41C37402]
module ffs best [41C37402]
module ldterm best [443FE96E]
module ptem best [443FE96E]
module pts best [443FE96E]
module ptm best [443FE96E]
module pckt best [443FE96E]
module nfs_core best [41F61CCE]
module nfs_server best [412E8CC4]
module nfs_client best [41F61CCE]
module nfsm best [412E8CC4]
module rpcmod best [41F5F18C]
module autofsc best [42E01411]
module cachefsc best [42279EC7]
module cifs best [426980D7]
module td best [435D5614]
module fcd best [435516DF]
module fcd_fcp best [435516DF]
module fcd_vbus best [435516DF]
module fddi4 best [41237311]
module gelan best [435D560A]
module iether best [43FC182D]
module igelan best [43FC1850]
module vxfs best [443FE999]
module vxportal best [443FE999]
module lvm best [443FE997]
module lv best [443FE997]
module ipmi best [443FE99B]
module ipmi_psm best [443FE99B]
module mip6mod best [42A90340]
module asyncdsk best [443FE9C4]
module ciss best [43FC1873]
module sasd best [43FC1138]
module sasd_vbus best [43FC1138]
module vxvm best [42931793]
module vxdmp best [4292E92B]
module vol best [42931793]
module vols best [42931793]
module dmpaa best 0.1.0
module dmpaaa best 0.1.0
module dmpap best 0.1.0
module dmpapg best 0.1.0
module dmpapf best 0.1.0
module dmpjbod best 0.1.0
module dmphpalua best 0.1.0
module dmphdsalua best 0.1.0
module mpt best [43FC1893]
module pfil auto 0.1.0
module ipf loaded 0.1.0
*
* Swap entries
*
*
* Dump entries
*
dump lvol
*
* Driver binding entries
*
*
* Tunables entries
*
tunable ncsize 34816
tunable max_async_ports 1024
tunable swchunk 65536
tunable dbc_max_pct 3
tunable dbc_min_pct 3
tunable vps_ceiling 64
tunable STRMSGSZ 65535
tunable shmseg 512
tunable shmmni 2048
tunable semume 512
tunable semmnu 4092
tunable semmns 8192
tunable semmni 4096
tunable nswapdev 25
tunable npty 200
tunable ninode 120000
tunable msgtql 5120
tunable msgssz 128
tunable msgseg 32767
tunable msgmnb 65536
tunable msgmax 32768
tunable msgmap 5122
tunable maxvgs 200
tunable maxuprc 3687
tunable maxtsiz 1073741824
tunable maxssiz 0x10000000
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 nstrpty 200
tunable cmc_plat_poll 15
tunable maxssiz_64bit 1073741824
tunable user:maxswapchunks 16384
tunable user:semmap 4098
tunable msgmni 4096
tunable vxfs_ifree_timelag 3600000
tunable unlockable_mem 1
tunable timezone 480
tunable swapmem_on 0
tunable semvmx 32768
tunable nproc 4096
tunable nfile 2000000
tunable maxtsiz_64bit 4294967296
tunable maxfiles_lim 4096
tunable maxfiles 4096
tunable maxdsiz_64bit 0x80000000
tunable maxdsiz 0x40000000
tunable eqmemsize 512
tunable bufpages 1000000
tunable pagezero_daemon_enabled 0
tunable shmmax 0xc80000000
```

env

```
########### MACHINE PARAMETERS #####################
#export RAC_NODES="titan1 titan2"
########### PATHS ##################################
export KIT_DIR=/dbms/10gR2/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/10gR2/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/10gR2/frame
export SCALE_FACTOR=1000
export UPDATE_1_DOP=32
export UPDATE_2_DOP=64
############ FRAME STUFF
export FRAME_PATH=$KIT_DIR/frame

export ORACORE3INCL=$ORACLE_HOME/rdbms/demo
export ORACORE3PUBL=$ORACLE_HOME/rdbms/public
export RDBMSPUBL=$ORACLE_HOME/rdbms/public
export NETWORKPUBL=$ORACLE_HOME/network/public
export RDBMSDEMO=$ORACLE_HOME/rdbms/demo
export PLSQLDEMO=$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
```

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 ORACLE_SID=1gtpch1
#echo 'ORACLE_SID is 1gtpch1'


export KIT_DIR=/dbms/10gR2/kit

export
SHLIB_PATH=$ORACLE_HOME/lib:$ORACLE_HOME/lib32:$ORACLE_HOME/rdbms/lib:$ORACLE_HOME/network/lib
export
LD_LIBRARY_PATH=$ORACLE_HOME/lib:$ORACLE_HOME/lib64:$ORACLE_HOME/rdbms/lib:$ORACLE_HOME/network/lib64
export SAVEHIST=2049
export FRAME_PATH=/dbms/10gR2/frame
export O=$ORACLE_HOME
export ORACLE_PATH=/dbms/10gR2/frame/tools
export PS1="`whoami`-(`hostname`)> "
export skgxp_trace_path=/tmp/srq.tpch1
export ASYNC_BUF_CONF=256
echo "export ASYNC_BUF_CONF=$ASYNC_BUF_CONF"

export
PATH=./:$ORACLE_HOME/bin:/usr/local/bin:$ORACLE_HOME:$ORACLE_HOME/lib:/tools:/tpch/run_power:/tpch:/dbms/10gR2/frame/bin:/dbms/10gR2/frame:/dbms/10gR2/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/sbin:.:/bin:/usr/bin:/usr/local/bin:/usr/contrib/bin:/etc:/usr/include:/dbms/10gR2/kit:/dbms/10gR2/kit/bumpx:/dbms/10gR2/local/TestIO:/usr/ccs/bin/:/opt/caliper/bin:/opt/rdma/bin:~/bin:/dbms/10gR2/frame/bin/

alias j="cd /dbms/10gR2/jobs"
alias cd_load="cd /dbms/10gR2/kit/audit/"
alias ltt="ls -ltr |tail -30"
alias cd_frame="cd /dbms/10gR2/frame"
alias cd_stats="cd /dbms/10gR2/frame/stats"
alias cd_q="cd /dbms/10gR2/frame/queries/queries_tpch"
alias cd_u="cd /dbms/10gR2/frame/queries/queries_tpch/updates"
alias cd_kit="cd /dbms/10gR2/kit/"
alias cd_audit="cd /dbms/10gR2/kit/audit/"
alias cd_acid="cd /dbms/10gR2/kit/acid/"
alias ltm="ls -lt |more"
alias cdtools="cd /dbms/10gR2/tools/bin"
alias cdq="cd /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_$ORACLE_SID.log"
alias cdlog="cd $ORACLE_HOME/rdbms/log"
alias oldstats="cd /dbms/10gR2/frame/stats_saved"
umask 002
iosum(){
if [ "$1" -eq "" ] ; then
   echo usage:  iosum iterations
else
 sar -d 5 $1 | ${FRAME_PATH}/bin/io.pl
fi
}
```

Appendix B Build Programs and Scripts

B.1 dbcre.sh

```
#!/bin/ksh



export ORACLE_SID=tpch

echo START CREATE DB at `date`

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

shutdown abort;


startup pfile=/oracle/dbs/1TB_init.ora nomount;
create database
  controlfile reuse
  logfile '+DG3' size 10000m reuse,
          '+DG3' size 10000m reuse
  datafile '+DG3' size 5000m reuse
  sysaux datafile '+DG3' size 5000m reuse
  undo tablespace ts_undo1
       datafile  '+DG3' size 32000m reuse
  maxdatafiles 1000
  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


/dbms/10gR2/frame/bin/tshut abort
/dbms/10gR2/frame/bin/tshut.asm
/dbms/10gR2/frame/bin/tstart.asm
/dbms/10gR2/frame/bin/tstart

export ORACLE_SID=tpch

echo CREATE TABLESPACES at `date`

(( i = 1 ))
while (( i <= 4 ))
do
sqlplus  / as sysdba <<! &
set timing on
set echo on

alter tablespace ts_undo1
add datafile '+DG3' size 20g reuse;
;
!
(( i = $i + 1 ))
done
wait


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

--drop tablespace ts_default including contents;
create tablespace ts_default
datafile '+DG3' size 8g reuse
extent management local
autoallocate
;
!

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

--drop tablespace ts_temp including contents;
create temporary tablespace ts_temp
tempfile '+DG3' size 32256M reuse
extent management local
uniform size 5M
;
!

wait

(( i = 1 ))

while (( i <= 30 ))
do

sqlplus  / as sysdba <<! &

set timing on
set echo on
alter tablespace ts_temp
  add tempfile '+DG3' size 32256M 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 '+DG3' size 18G reuse
extent management dictionary
default storage (initial 35m next 35m 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 '+DG3' size 4000M reuse
extent management dictionary
default storage (initial 12m next 12m 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 tablespace ts_c
datafile '+DG3' size 25000M reuse
extent management dictionary
default storage (initial 15m next 15m maxextents unlimited pctincrease
0)
nologging
;
!

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

--drop tablespace ts_p including contents;
create tablespace ts_p
datafile '+DG3' size 28500M reuse
extent management dictionary
default storage (initial 20m next 20m maxextents unlimited pctincrease
0)
nologging
;
!

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

--drop tablespace ts_s including contents;
create tablespace ts_s
datafile '+DG3' size 1800M reuse
extent management dictionary
default storage (initial 2m next 2m maxextents unlimited pctincrease 0)
nologging
;
!

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

--drop tablespace ts_psupp including contents;
create tablespace ts_psupp
datafile '+DG3' size 27800M reuse
extent management dictionary
default storage (initial 80m next 80m maxextents unlimited pctincrease
0)
nologging
;
!
wait

(( i = 1 ))

while (( i <= 4 ))
do
sqlplus  / as sysdba <<! &
set timing on
set echo on

alter tablespace ts_psupp
add datafile '+DG3' size 28500M reuse;
!

(( i = $i + 1 ))
done
wait

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

--drop tablespace ts_okey including contents;
create tablespace ts_okey
datafile '+DG3' size 17000M 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 '+DG3' size 3200M 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 '+DG3' size 28000M reuse
extent management local
autoallocate nologging
;
!
```

```
wait

(( i = 1 ))

while (( i <= 5 ))
do
sqlplus  / as sysdba <<! &
set timing on
set echo on

alter tablespace ts_lokey
add datafile '+DG3' size 28000M reuse;
!

(( i = $i + 1 ))
done

(( i = 1 ))

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

alter tablespace ts_okey
add datafile '+DG3' size 17000M reuse;
!

(( i = $i + 1 ))
done

wait

echo END CREATE TABLESPACES at `date`
```

B.3 dapop.sh

```
#!/bin/ksh


export ORACLE_SID=tpch

echo START TABLE CREATION at `date`

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;

create directory data_dir1 as '/flat1/';
create directory data_dir2 as '/flat2/';
create directory data_dir3 as '/flat3/';
create directory data_dir4 as '/flat4/';


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_dir1:'lineitem.tbl.6',
              data_dir1:'lineitem.tbl.7',
              data_dir1:'lineitem.tbl.8',
              data_dir1:'lineitem.tbl.9',
              data_dir1:'lineitem.tbl.10',
              data_dir1:'lineitem.tbl.11',
              data_dir1:'lineitem.tbl.12',
              data_dir1:'lineitem.tbl.13',
              data_dir1:'lineitem.tbl.14',
              data_dir1:'lineitem.tbl.15',
              data_dir1:'lineitem.tbl.16',
              data_dir1:'lineitem.tbl.17',
              data_dir1:'lineitem.tbl.18',
              data_dir1:'lineitem.tbl.19',
              data_dir1:'lineitem.tbl.20',
              data_dir1:'lineitem.tbl.21',
              data_dir2:'lineitem.tbl.22',
              data_dir2:'lineitem.tbl.23',
              data_dir2:'lineitem.tbl.24',
              data_dir2:'lineitem.tbl.25',
              data_dir2:'lineitem.tbl.26',
              data_dir2:'lineitem.tbl.27',
              data_dir2:'lineitem.tbl.28',
              data_dir2:'lineitem.tbl.29',
              data_dir2:'lineitem.tbl.30',
              data_dir2:'lineitem.tbl.31',
              data_dir2:'lineitem.tbl.32',
              data_dir2:'lineitem.tbl.33',
              data_dir2:'lineitem.tbl.34',
              data_dir2:'lineitem.tbl.35',
            data_dir2:'lineitem.tbl.36',
            data_dir2:'lineitem.tbl.37',
            data_dir2:'lineitem.tbl.38',
            data_dir2:'lineitem.tbl.39',
            data_dir2:'lineitem.tbl.40',
            data_dir2:'lineitem.tbl.41',
            data_dir2:'lineitem.tbl.42',
            data_dir3:'lineitem.tbl.43',
            data_dir3:'lineitem.tbl.44',
            data_dir3:'lineitem.tbl.45',
            data_dir3:'lineitem.tbl.46',
            data_dir3:'lineitem.tbl.47',
            data_dir3:'lineitem.tbl.48',
            data_dir3:'lineitem.tbl.49',
            data_dir3:'lineitem.tbl.50',
            data_dir3:'lineitem.tbl.51',
            data_dir3:'lineitem.tbl.52',
            data_dir3:'lineitem.tbl.53',
            data_dir3:'lineitem.tbl.54',
            data_dir3:'lineitem.tbl.55',
            data_dir3:'lineitem.tbl.56',
            data_dir3:'lineitem.tbl.57',
            data_dir3:'lineitem.tbl.58',
            data_dir3:'lineitem.tbl.59',
            data_dir3:'lineitem.tbl.60',
            data_dir3:'lineitem.tbl.61',
            data_dir3:'lineitem.tbl.62',
            data_dir3:'lineitem.tbl.63',
            data_dir4:'lineitem.tbl.64',
            data_dir4:'lineitem.tbl.65',
            data_dir4:'lineitem.tbl.66',
            data_dir4:'lineitem.tbl.67',
            data_dir4:'lineitem.tbl.68',
            data_dir4:'lineitem.tbl.69',
            data_dir4:'lineitem.tbl.70',
            data_dir4:'lineitem.tbl.71',
            data_dir4:'lineitem.tbl.72',
            data_dir4:'lineitem.tbl.73',
            data_dir4:'lineitem.tbl.74',
            data_dir4:'lineitem.tbl.75',
            data_dir4:'lineitem.tbl.76',
            data_dir4:'lineitem.tbl.77',
            data_dir4:'lineitem.tbl.78',
            data_dir4:'lineitem.tbl.79',
            data_dir4:'lineitem.tbl.80',
            data_dir4:'lineitem.tbl.81',
            data_dir4:'lineitem.tbl.82',
            data_dir4:'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_dir1:'orders.tbl.6',
data_dir1:'orders.tbl.7',
data_dir1:'orders.tbl.8',
data_dir1:'orders.tbl.9',
data_dir1:'orders.tbl.10',
data_dir1:'orders.tbl.11',
data_dir1:'orders.tbl.12',
data_dir1:'orders.tbl.13',
data_dir1:'orders.tbl.14',
data_dir1:'orders.tbl.15',
data_dir1:'orders.tbl.16',
data_dir1:'orders.tbl.17',
data_dir1:'orders.tbl.18',
data_dir1:'orders.tbl.19',
data_dir1:'orders.tbl.20',
data_dir1:'orders.tbl.21',
data_dir2:'orders.tbl.22',
data_dir2:'orders.tbl.23',
data_dir2:'orders.tbl.24',
data_dir2:'orders.tbl.25',
data_dir2:'orders.tbl.26',
data_dir2:'orders.tbl.27',
data_dir2:'orders.tbl.28',
data_dir2:'orders.tbl.29',
data_dir2:'orders.tbl.30',
data_dir2:'orders.tbl.31',
data_dir2:'orders.tbl.32',
data_dir2:'orders.tbl.33',
data_dir2:'orders.tbl.34',
data_dir2:'orders.tbl.35',
data_dir2:'orders.tbl.36',
data_dir2:'orders.tbl.37',
data_dir2:'orders.tbl.38',
data_dir2:'orders.tbl.39',
data_dir2:'orders.tbl.40',
data_dir2:'orders.tbl.41',
data_dir2:'orders.tbl.42',
data_dir3:'orders.tbl.43',
data_dir3:'orders.tbl.44',
data_dir3:'orders.tbl.45',
data_dir3:'orders.tbl.46',
data_dir3:'orders.tbl.47',
data_dir3:'orders.tbl.48',
data_dir3:'orders.tbl.49',
data_dir3:'orders.tbl.50',
data_dir3:'orders.tbl.51',
data_dir3:'orders.tbl.52',
data_dir3:'orders.tbl.53',
data_dir3:'orders.tbl.54',
data_dir3:'orders.tbl.55',
data_dir3:'orders.tbl.56',
data_dir3:'orders.tbl.57',
data_dir3:'orders.tbl.58',
data_dir3:'orders.tbl.59',
data_dir3:'orders.tbl.60',
data_dir3:'orders.tbl.61',
data_dir3:'orders.tbl.62',
data_dir3:'orders.tbl.63',
data_dir4:'orders.tbl.64',
data_dir4:'orders.tbl.65',
data_dir4:'orders.tbl.66',
        data_dir4:'orders.tbl.67',
        data_dir4:'orders.tbl.68',
        data_dir4:'orders.tbl.69',
        data_dir4:'orders.tbl.70',
        data_dir4:'orders.tbl.71',
        data_dir4:'orders.tbl.72',
        data_dir4:'orders.tbl.73',
        data_dir4:'orders.tbl.74',
        data_dir4:'orders.tbl.75',
        data_dir4:'orders.tbl.76',
        data_dir4:'orders.tbl.77',
        data_dir4:'orders.tbl.78',
        data_dir4:'orders.tbl.79',
        data_dir4:'orders.tbl.80',
        data_dir4:'orders.tbl.81',
        data_dir4:'orders.tbl.82',
        data_dir4:'orders.tbl.83',
        data_dir4:'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_dir1:'partsupp.tbl.5',
        data_dir1:'partsupp.tbl.6',
        data_dir1:'partsupp.tbl.7',
        data_dir1:'partsupp.tbl.8',
        data_dir1:'partsupp.tbl.9',
        data_dir1:'partsupp.tbl.10',
        data_dir1:'partsupp.tbl.11',
        data_dir1:'partsupp.tbl.12',
        data_dir1:'partsupp.tbl.13',
        data_dir1:'partsupp.tbl.14',
        data_dir1:'partsupp.tbl.15',
        data_dir1:'partsupp.tbl.16',
        data_dir2:'partsupp.tbl.17',
        data_dir2:'partsupp.tbl.18',
        data_dir2:'partsupp.tbl.19',
        data_dir2:'partsupp.tbl.20',
        data_dir2:'partsupp.tbl.21',
        data_dir2:'partsupp.tbl.22',
        data_dir2:'partsupp.tbl.23',
        data_dir2:'partsupp.tbl.24',
        data_dir2:'partsupp.tbl.25',
        data_dir2:'partsupp.tbl.26',
        data_dir2:'partsupp.tbl.27',
        data_dir2:'partsupp.tbl.28',
        data_dir2:'partsupp.tbl.29',
        data_dir2:'partsupp.tbl.30',
        data_dir2:'partsupp.tbl.31',
        data_dir2:'partsupp.tbl.32',
        data_dir3:'partsupp.tbl.33',
        data_dir3:'partsupp.tbl.34',
        data_dir3:'partsupp.tbl.35',
        data_dir3:'partsupp.tbl.36',
        data_dir3:'partsupp.tbl.37',
        data_dir3:'partsupp.tbl.38',
        data_dir3:'partsupp.tbl.39',
        data_dir3:'partsupp.tbl.40',
        data_dir3:'partsupp.tbl.41',
        data_dir3:'partsupp.tbl.42',
        data_dir3:'partsupp.tbl.43',
        data_dir3:'partsupp.tbl.44',
        data_dir3:'partsupp.tbl.45',
        data_dir3:'partsupp.tbl.46',
        data_dir3:'partsupp.tbl.47',
        data_dir3:'partsupp.tbl.48',
        data_dir4:'partsupp.tbl.49',
        data_dir4:'partsupp.tbl.50',
        data_dir4:'partsupp.tbl.51',
        data_dir4:'partsupp.tbl.52',
        data_dir4:'partsupp.tbl.53',
        data_dir4:'partsupp.tbl.54',
        data_dir4:'partsupp.tbl.55',
        data_dir4:'partsupp.tbl.56',
        data_dir4:'partsupp.tbl.57',
        data_dir4:'partsupp.tbl.58',
        data_dir4:'partsupp.tbl.59',
        data_dir4:'partsupp.tbl.60',
        data_dir4:'partsupp.tbl.61',
        data_dir4:'partsupp.tbl.62',
        data_dir4:'partsupp.tbl.63',
        data_dir4:'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_dir1:'part.tbl.2',
        data_dir1:'part.tbl.3',
        data_dir1:'part.tbl.4',
        data_dir2:'part.tbl.5',
        data_dir2:'part.tbl.6',
        data_dir2:'part.tbl.7',
        data_dir2:'part.tbl.8',
        data_dir3:'part.tbl.9',
        data_dir3:'part.tbl.10',
        data_dir3:'part.tbl.11',
        data_dir3:'part.tbl.12',
        data_dir4:'part.tbl.13',
        data_dir4:'part.tbl.14',
        data_dir4:'part.tbl.15',
        data_dir4:'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_dir1:'customer.tbl.2',
        data_dir1:'customer.tbl.3',
        data_dir1:'customer.tbl.4',
        data_dir2:'customer.tbl.5',
        data_dir2:'customer.tbl.6',
        data_dir2:'customer.tbl.7',
        data_dir2:'customer.tbl.8',
        data_dir3:'customer.tbl.9',
        data_dir3:'customer.tbl.10',
        data_dir3:'customer.tbl.11',
        data_dir3:'customer.tbl.12',
        data_dir4:'customer.tbl.13',
        data_dir4:'customer.tbl.14',
        data_dir4:'customer.tbl.15',
        data_dir4:'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'
                  ))
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_dir1:'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_dir1:'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 64
(
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 64
(
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 64
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 64
tablespace ts_c
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 64
tablespace ts_p
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 64
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

export ORACLE_SID=tpch

echo START INDEX at `date`
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 32
pctfree 5
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 32
pctfree 2
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 32
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`

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

execute dbms_stats.gather_schema_stats('TPCH' , estimate_percent =>
1, degree => 16 , 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;
!

echo END ANALYZE at `date`
```

B.6 loadasm

```
#!/bin/ksh

set -x

export ORACLE_SID=ASM
echo $ORACLE_SID

sqlplus /NOLOG <<!
set echo ON;
set timing ON;
connect / as sysdba;
shutdown abort;
startup pfile=/oracle/dbs/initASM.ora ;
alter diskgroup all mount;
drop diskgroup DG3 including contents;
create diskgroup DG3 External REDUNDANCY
DISK
'/dev/rdsk/c21t0d0' SIZE 117750M,
'/dev/rdsk/c25t0d0' SIZE 117750M,
'/dev/rdsk/c9t0d0' SIZE 117750M,
'/dev/rdsk/c11t0d0' SIZE 117750M,
'/dev/rdsk/c13t0d0' SIZE 117750M,
'/dev/rdsk/c15t0d0' SIZE 117750M,
'/dev/rdsk/c28t0d0' SIZE 117750M,
'/dev/rdsk/c17t0d0' SIZE 117750M,
'/dev/rdsk/c19t0d0' SIZE 117750M,
'/dev/rdsk/c23t0d0' SIZE 117750M,
'/dev/rdsk/c45t0d0' SIZE 117750M,
'/dev/rdsk/c49t0d0' SIZE 117750M,
'/dev/rdsk/c33t0d0' SIZE 117750M,
'/dev/rdsk/c35t0d0' SIZE 117750M,
'/dev/rdsk/c37t0d0' SIZE 117750M,
'/dev/rdsk/c39t0d0' SIZE 117750M,
'/dev/rdsk/c52t0d0' SIZE 117750M,
'/dev/rdsk/c41t0d0' SIZE 117750M,
'/dev/rdsk/c43t0d0' SIZE 117750M,
'/dev/rdsk/c47t0d0' SIZE 117750M,
'/dev/rdsk/c66t0d0' SIZE 117750M,
'/dev/rdsk/c68t0d0' SIZE 117750M,
'/dev/rdsk/c56t0d0' SIZE 117750M,
'/dev/rdsk/c58t0d0' SIZE 117750M,
'/dev/rdsk/c60t0d0' SIZE 117750M,
'/dev/rdsk/c62t0d0' SIZE 117750M,
'/dev/rdsk/c70t0d0' SIZE 117750M,
'/dev/rdsk/c64t0d0' SIZE 117750M,
'/dev/rdsk/c72t0d0' SIZE 117750M,
'/dev/rdsk/c94t0d0' SIZE 117750M,
'/dev/rdsk/c90t0d0' SIZE 117750M,
'/dev/rdsk/c92t0d0' SIZE 117750M,
'/dev/rdsk/c84t0d0' SIZE 117750M,
'/dev/rdsk/c86t0d0' SIZE 117750M,
'/dev/rdsk/c88t0d0' SIZE 117750M,
'/dev/rdsk/c74t0d0' SIZE 117750M,
'/dev/rdsk/c76t0d0' SIZE 117750M,
'/dev/rdsk/c78t0d0' SIZE 117750M,
'/dev/rdsk/c80t0d0' SIZE 117750M,
'/dev/rdsk/c82t0d0' SIZE 117750M;
alter diskgroup DG3 rebalance power 0;
!

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

export ORACLE_SID=ASM
echo $ORACLE_SID
sqlplus /NOLOG <<!
connect / as sysdba
startup pfile=/oracle/dbs/initASM.ora mount
!
```

Acid Scripts

a_query.sql

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

set serverout on;

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

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

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

exit;
```

a_query2.sql

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

set serverout on;

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

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

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

exit;
```

atom.sh

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

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;
```

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);

}
```

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 */
```

ckpt.sh

```
#!/bin/ksh
#
# $Header: ckpt.sh 08-aug-99.17:32:22 mpoess Exp $
#
# ckpt.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#    NAME
#      ckpt.sh - <one-line expansion of the name>
#
#    DESCRIPTION
#      <short description of component this file declares/defines>
#
#    NOTES
#      <other useful comments, qualifications, etc.>
#
#    MODIFIED   (MM/DD/YY)
#    mpoess      08/08/99 - Creation
#    mpoess      08/08/99 - Creation
#
. $KIT_DIR/env
sqlplus -s /NOLOG << !

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

cnt_hist.sql

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

consist.sh

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

. $KIT_DIR/env

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

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

consist.sql

```
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 /*+ index(lineitem,i_l_orderkey) */
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
```

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
```

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;
```

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_1g.log >
${DURA_DIR}/alert_1g.log.post_dura 2>&1
```

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
```

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
```

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
```

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
```

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
```

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
```

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
```

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
```

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;
```

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;
```

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);
}
```

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");
}
```

run_acid.sh

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

. $KIT_DIR/env

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

usage() {

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

ITER=600
STEM=${NUM_STREAMS}
let STEM="$STEM + 1" # add one for the update stream
SF=1
PROG=atranspl
IN=${ACID_DIR}/acid_in
DURA_DIR=$ACID_OUT/dura
OUT=$DURA_DIR/drate
DURA=$DURA_DIR/dura
KEY=${DURA_DIR}/key$$_
echo "$$" > ${DURA_DIR}/shellpid
USER=tpch/tpch
TRIG=1
HCNT=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_1g.log >
${DURA_DIR}/alert_1g.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"
```

sample.sh

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

# $1 durability output file

. $KIT_DIR/env

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



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

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

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

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

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

sample.sql

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

Appendix D Query text and Output

qryqual

Begin Execution at Fri Jul 28 15:05:32 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 1.42 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
B3rqp0xbSEim4Mpy2RH J                  33-320-228-2957
blithely special packages are. stealthily express deposits across the
closely final instructi
9923.77             Supplier#000002324       GERMANY
29821.00             Manufacturer#4
y3OD9UywSTOk                          17-779-299-1839
quickly express packages breach quiet pinto beans. requ
9871.22             Supplier#000006373       GERMANY
43868.00             Manufacturer#5
J8fcXWsTqM                            17-813-485-8637
never silent deposits integrate furiously blit
```

```
9870.78          Supplier#000001286       GERMANY
81285.00          Manufacturer#2
YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH    17-516-924-4574
final theodolites cajole slyly special,
9870.78          Supplier#000001286       GERMANY
181285.00          Manufacturer#4
YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH    17-516-924-4574
final theodolites cajole slyly special,
9852.52          Supplier#000008973       RUSSIA
18972.00          Manufacturer#2
t5L67YdBYYH6o,Vz24jpDyQ9            32-188-594-7038
quickly regular instructions wake-- carefully unusual braids into the
expres

-------
-------
<deleted>
-------
-------


100 rows processed.
Query Processed in 2.42 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 1.49 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 1.68 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 3.86 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 0.21 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 1.80 seconds.
```

```
-- @(#)8a.sql             2.1.6.2
-- TPC-H/TPC-R National Market Share Query (Q8)
-- Variant A
-- Approved February 1998


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

```
O_YEAR          MKT_SHARE
1995.00         0.03
1996.00         0.04


2 rows processed.
Query Processed in 8.40 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
ALGERIA          1992.00         54628034.71
ARGENTINA           1998.00           30211185.71
ARGENTINA           1997.00           50805741.75
ARGENTINA           1996.00           51923746.58
ARGENTINA           1995.00           49298625.77
ARGENTINA           1994.00           50835610.11
ARGENTINA           1993.00           51646079.18
ARGENTINA           1992.00           50410314.99
BRAZIL          1998.00         27217924.38
BRAZIL          1997.00         48378669.20
BRAZIL          1996.00         50482870.36
BRAZIL          1995.00         47623383.63
BRAZIL          1994.00         47840165.73
BRAZIL          1993.00         49054694.04
BRAZIL          1992.00         48667639.08
CANADA            1998.00           30379833.77
CANADA            1997.00           50465052.31
CANADA            1996.00           52560501.39
CANADA            1995.00           52375332.81
CANADA            1994.00           52600364.66
CANADA            1993.00           52644504.07
CANADA            1992.00           53932871.70
CHINA           1998.00         31075466.16
CHINA           1997.00         50551874.45
CHINA           1996.00         51039293.88

-------
-------
<deleted>
-------
-------


UNITED STATES          1998.00            25126238.95
UNITED STATES          1997.00            50077306.42
UNITED STATES          1996.00            48048649.47
UNITED STATES          1995.00            48809032.42
UNITED STATES          1994.00            49296747.18
UNITED STATES          1993.00            48029946.80
UNITED STATES          1992.00            48671944.50
VIETNAM           1998.00           30442736.06
VIETNAM           1997.00           50309179.79
VIETNAM           1996.00           50488161.41
VIETNAM           1995.00           49658284.61
VIETNAM           1994.00           50596057.26
VIETNAM           1993.00           50953919.15
VIETNAM           1992.00           49613838.32


175 rows processed.
Query Processed in 5.33 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
-------
-------
<deleted>
-------
-------



20 rows processed.
Query Processed in 1.83 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
222.00               13937777.74
121271.00            13908336.00
55221.00             13716120.47
22819.00             13666434.28
76281.00             13646853.68
85298.00             13581154.93
85158.00             13554904.00
139684.00            13535538.72
31034.00             13498025.25
87305.00             13482847.04
10181.00             13445148.75
62323.00             13411824.30
26489.00             13377256.38
-------
-------
<deleted>
-------
-------

25891.00             7890511.20
122819.00            7888881.02
154731.00            7888301.33
101674.00            7879324.60
51968.00             7879102.21
72073.00             7877736.11
5182.00              7874521.73


1048 rows processed.
Query Processed in 4.88 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 1.67 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 0.85 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 0.25 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 1.88 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
-------
-------
<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 0.63 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 1.75 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
500241.33              302.00
Customer#000146608     146608.00        3342468.00        1994-
06-12
499794.58              303.00
Customer#000096103     96103.00         5984582.00        1992-
03-16
494398.79              312.00
Customer#000024341     24341.00         1474818.00        1992-
11-15
491348.26              302.00
Customer#000137446     137446.00        5489475.00        1997-
05-23
487763.25              311.00
Customer#000107590     107590.00        4267751.00        1994-
11-04
485141.38              301.00
Customer#000050008     50008.00         2366755.00        1996-
12-09
483891.26              302.00
Customer#000015619     15619.00         3767271.00        1996-
08-07
480083.96              318.00
Customer#000077260     77260.00         1436544.00        1992-
09-12
479499.43              307.00
Customer#000109379     109379.00        5746311.00        1996-
10-10
478064.11              302.00
Customer#000054602     54602.00         5832321.00        1997-
02-09
471220.08              307.00
Customer#000105995     105995.00        2096705.00        1994-
07-03
469692.58              307.00
-------
-------
<deleted>
-------
-------

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 2.08 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 1.65 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,
Supplier#000000688       D fw5ocppmZpYBBIPI718hCihLDZ5KhKX
Supplier#000000710       f19YPvOyb QoYwjKC,oPycpGfieBAcwKJo
Supplier#000000736
l6i2nMwVuovfKnuVgaSGK2rDy65DlAFLegiL7
-------
-------
<deleted>
-------
-------

Supplier#000009709       rRnCbHYgDgl9PZYnyWKVYSUW0vKg
Supplier#000009753       wLhVEcRmd7PkJF4FBnGK7Z
Supplier#000009796       z,y4Idmr15DOvPUqYG
Supplier#000009799        4wNjXGa4OKWl
Supplier#000009811       E3iuyq7UnZxU7oPZIe2Gu6
Supplier#000009812       APFRMy3lCbgFga53n5t9DxzFPQPgnjrGt32
Supplier#000009862       rJzweWeN58
Supplier#000009868       ROjGgx5gvtkmnUUoeyy7v
Supplier#000009869
ucLqxzrpBTRMewGSM29t0rNTM30g1Tu3Xgg3mKag
Supplier#000009899       7XdpAHrzr1t,UQFZE
Supplier#000009974       7wJ,J5DKcxSU4Kp1cQLpbcAvB5AsvKT


204 rows processed.
Query Processed in 2.70 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
-------
-------
<deleted>
-------
-------

Supplier#000000821      12.00
Supplier#000001337      12.00
Supplier#000001916      12.00
Supplier#000001925      12.00
Supplier#000002039      12.00
Supplier#000002357      12.00
Supplier#000002483      12.00


100 rows processed.
Query Processed in 9.67 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 1.01 seconds.


Ended Executing this Stream at Fri Jul 28 15:06:30 2006


Stream Started at 1154124332.92
Stream Ended at 1154124390.38
Stream Processed in 57.47 seconds


SQL statements processed: 22
```

Seed and Input Parameters

Seed

0730031043

qp1.0

```
14          1994-01-01
2           23          BRASS       AFRICA
9           plum
20          grey        1995-01-01              RUSSIA
6           1994-01-01              0.04        25
17          Brand#53    LG CAN
18          313
8           MOZAMBIQUE              AFRICA      ECONOMY BURNISHED NICKEL
21          JAPAN
13          express     packages
3           AUTOMOBILE              1995-03-01
22          17          31          20          25          13          26          29
16          Brand#55    MEDIUM PLATED           11          1           12          43          9           46          39          6
4           1993-02-01
11          VIETNAM                 0.0000001000
15          1996-01-01
1           100
10          1994-04-01
19          Brand#22    Brand#35    Brand#44    1           11          20
5           AFRICA      1994-01-01
7           GERMANY                 MOZAMBIQUE
12          RAIL        SHIP        1993-01-01
```

qp1.1

```
21          EGYPT
3           HOUSEHOLD               1995-03-17
18          315
5           AMERICA                 1994-01-01
11          INDONESIA               0.0000001000
7           UNITED STATES           INDIA
6           1994-01-01              0.09        24
20          royal       1993-01-01              IRAQ
17          Brand#55    MED BOX
12          AIR         SHIP        1994-01-01
16          Brand#35    ECONOMY BRUSHED                     27          17          47          24          1           45          31          12
15          1993-10-01
13          express     packages
10          1993-02-01
2           11          TIN         ASIA
8           INDIA       ASIA        LARGE BRUSHED NICKEL
14          1994-04-01
19          Brand#34    Brand#13    Brand#43    7           12          28
9           orchid
22          33          25          10          11          24          23          18
1           109
4           1995-09-01
```

qp1.2

```
6           1994-01-01              0.07        24
17          Brand#51    MED PACK
```

```
14        1994-07-01
16        Brand#25  SMALL ANODIZED 7        11        45        19        27        6         8         16
19        Brand#32  Brand#51  Brand#32  2         13        24
10        1993-11-01
9         misty
2         49        COPPER    AFRICA
15        1996-05-01
8         ALGERIA AFRICA    LARGE PLATED NICKEL
5         EUROPE    1994-01-01
22        11        23        33        13        26        29        19
12        SHIP      RAIL      1996-01-01
7         MOZAMBIQUE        ALGERIA
13        express   requests
18        312
1         117
4         1993-06-01
20        cornsilk  1997-01-01          ARGENTINA
3         AUTOMOBILE        1995-03-03
11        RUSSIA    0.0000001000
21        RUSSIA
```

qp1.3

```
8         PERU      AMERICA             LARGE ANODIZED NICKEL
5         MIDDLE EAST       1994-01-01
4         1995-12-01
6         1994-01-01          0.04      25
17        Brand#53  MED CAN
7         INDIA     PERU
1         64
18        314
22        26        28        17        27        29        31        16
14        1994-11-01
9         magenta
10        1994-08-01
15        1994-01-01
11        IRAN      0.0000001000
20        olive     1995-01-01          MOZAMBIQUE
2         37        STEEL     EUROPE
21        KENYA
19        Brand#34  Brand#34  Brand#31  7         14        20
13        special   requests
16        Brand#55  LARGE PLATED        38        21        17        46        25        4         34        49
12        FOB       REG AIR   1994-01-01
3         FURNITURE           1995-03-19
```

qp1.4

```
5         AFRICA    1995-01-01
21        FRANCE
14        1995-02-01
19        Brand#41  Brand#22  Brand#31  2         15        27
15        1996-08-01
17        Brand#55  JUMBO BOX
12        MAIL      REG AIR   1995-01-01
6         1995-01-01          0.02      24
4         1993-09-01
9         lavender
8         INDONESIA           ASIA      MEDIUM POLISHED BRASS
16        Brand#35  PROMO POLISHED 24       19        46        16        37        32        10        25
11        UNITED KINGDOM 0.0000001000
2         25        BRASS     AFRICA
10        1993-05-01
18        315
1         72
13        special   requests
```

```
7         ALGERIA INDONESIA
22        17        20        24        14        23        19        13
3         AUTOMOBILE          1995-03-05
20        azure     1994-01-01          FRANCE
```

qp1.5

```
21        UNITED KINGDOM
15        1994-05-01
4         1996-04-01
6         1995-01-01          0.07      24
7         PERU      ARGENTINA
16        Brand#25  MEDIUM ANODIZED               15        47        48        16        17        44        37        10
19        Brand#43  Brand#55  Brand#25  8         16        23
18        313
14        1995-05-01
22        20        18        23        17        26        34        21
11        IRAQ      0.0000001000
13        special   requests
3         FURNITURE           1995-03-21
1         80
2         12        NICKEL    EUROPE
5         AMERICA             1995-01-01
8         ARGENTINA           AMERICA             MEDIUM BURNISHED BRASS
20        lawn      1997-01-01          SAUDI ARABIA
12        TRUCK     REG AIR   1995-01-01
17        Brand#52  JUMBO PACK
10        1994-02-01
9         honeydew
```

qp1.6

```
10        1994-12-01
3         MACHINERY           1995-03-07
15        1996-12-01
13        special   accounts
6         1995-01-01          0.05      25
8         CHINA     ASIA      SMALL BRUSHED BRASS
9         frosted
7         INDONESIA           CHINA
4         1994-01-01
11        UNITED STATES       0.0000001000
22        12        16        27        19        26        17        10
18        314
12        RAIL      REG AIR   1995-01-01
1         88
5         ASIA      1995-01-01
16        Brand#55  ECONOMY BURNISHED             35        46        32        22        28        30        26        7
2         50        COPPER    AMERICA
14        1995-08-01
19        Brand#45  Brand#43  Brand#24  3         17        20
20        smoke     1996-01-01          IRAN
17        Brand#54  JUMBO CAN
21        MOROCCO
```

qp1.7

```
18        312
8         IRAN      MIDDLE EAST         SMALL PLATED BRASS
20        floral    1994-01-01          ALGERIA\]
```

```
21        GERMANY
2         38        STEEL     EUROPE
4         1996-08-01
22        10        20        22        21        30        16        17
17        Brand#51  WRAP BOX
1         96
11        JAPAN     0.0000001000
9         dim
19        Brand#52  Brand#21  Brand#23  8         18        27
3         BUILDING            1995-03-23
13        special   accounts
5         EUROPE    1995-01-01
7         ARGENTINA           IRAN
10        1993-09-01
16        Brand#35  STANDARD POLISHED       47        42        48        12        3         40        23        20
6         1995-01-01          0.02      24
14        1995-11-01
15        1994-08-01
12        REG AIR   AIR       1995-01-01
```

Appendix F Benchmark Scripts

F.2 dbtables.sql

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

SELECT * FROM LINEITEM
WHERE L_ORDERKEY IN
 ( 4, 26598,  148577,  387431, 56704, 517442, 600000)
AND L_LINENUMBER = 1
ORDER BY L_ORDERKEY;



SELECT * FROM REGION;

SELECT COUNT(*) FROM NATION;

SELECT * FROM NATION
WHERE N_NATIONKEY IN (3,10,14,20)
ORDER BY N_NATIONKEY;



SELECT COUNT(*) FROM ORDERS;

SELECT * FROM ORDERS
WHERE O_ORDERKEY IN ( 7, 44065, 287590, 411111, 483876,
599942 )
ORDER BY O_ORDERKEY;



SELECT COUNT(*) FROM PART;

SELECT * FROM PART
WHERE P_PARTKEY IN (1,984,8743,9028,13876,17899,20000)
ORDER BY P_PARTKEY;



SELECT COUNT(*) FROM PARTSUPP;

SELECT* FROM PARTSUPP
  WHERE PS_PARTKEY = 3398
  AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
    FROM PARTSUPP WHERE PS_PARTKEY = 3398);

SELECT* FROM PARTSUPP
  WHERE PS_PARTKEY =15873
  AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
    FROM PARTSUPP WHERE PS_PARTKEY = 15873);

SELECT* FROM PARTSUPP
  WHERE PS_PARTKEY = 11394
  AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
    FROM PARTSUPP WHERE PS_PARTKEY = 11394);

SELECT* FROM PARTSUPP
  WHERE PS_PARTKEY = 6743
  AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
    FROM PARTSUPP WHERE PS_PARTKEY = 6743);

SELECT* FROM PARTSUPP
  WHERE PS_PARTKEY = 19763
  AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
    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.3 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.4 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.5 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.6 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.7 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.8 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
$FRAME_DIR/bin/tshut >>  $SCRIPT_LOG_FILE
$FRAME_DIR/bin/tstart >>  $SCRIPT_LOG_FILE
dapop.sh > $LD3DAPOP
ixcre.sh > $LD4IXCRE
anl.sh > $LD5ANLYZ
$FRAME_DIR/bin/tshut
$FRAME_DIR/bin/tshut.asm
$FRAME_DIR/bin/tstart.asm
$FRAME_DIR/bin/tstart
$KIT_DIR/audit/ckpnt.sh
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

$FRAME_DIR/bin/tshut >>  $SCRIPT_LOG_FILE
$FRAME_DIR/bin/tshut.asm >>  $SCRIPT_LOG_FILE
$FRAME_DIR/bin/tstart.asm >>  $SCRIPT_LOG_FILE
$FRAME_DIR/bin/tstart >>  $SCRIPT_LOG_FILE
$KIT_DIR/audit/ckpnt.sh
runTPCHpt ${SCALE_FACTOR} 1 ${RUN_ID}

$FRAME_DIR/bin/tshut >>  $SCRIPT_LOG_FILE
$FRAME_DIR/bin/tshut.asm >>  $SCRIPT_LOG_FILE
$FRAME_DIR/bin/tstart.asm >>  $SCRIPT_LOG_FILE
$FRAME_DIR/bin/tstart >>  $SCRIPT_LOG_FILE
$KIT_DIR/audit/ckpnt.sh
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.9 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.10 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.11 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 '/flat1/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;

drop table temp_l_et;
drop table temp_o_et;

exit;
!
```

```
END=`$GTIME`

# Done

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

F.12 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 '/flat1/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=10;


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

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

commit;

drop table temp_okey;
drop table temp_okey_et;
exit;
!

END=`$GTIME`

# Done

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

F.13 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.14 set_queue

```
for i in 21 25 9 11 13 15 28 17 19 23 45 49 33 35 37 39
52 41 43 47 \
        66 68 56 58 60 62 70 64 72 94 90 92 84 86 88 74 76
78 80 82
do
             /usr/sbin/scsictl -m queue_depth=128
/dev/rdsk/c${i}t0d0
             /usr/sbin/scsictl -m queue_depth=128
/dev/rdsk/c${i}t0d3
done
exit
```

F.15 tshut

```
#!/bin/ksh

export ORACLE_SID=tpch

sqlplus /NOLOG<< !
connect / as sysdba
shutdown abort
exit
!

exit
```

F.16 tshut.asm

```
#!/bin/ksh

export ORACLE_SID=ASM

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

exit
```

F.17 tstart

```
#!/bin/ksh

export ORACLE_SID=tpch

mpsched -P RR sqlplus /NOLOG << !
connect / as sysdba
startup pfile=/oracle/dbs/1TB_init.ora
execute dbms_scheduler.disable('AUTO_TASKS_JOB_CLASS');
exit
!

/home/Lvm/set_queue.sh

exit
```

F.18 tstart.asm

```
#!/bin/ksh

export ORACLE_SID=ASM

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

exit
```

Price Quotes

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

Sharada Bose
HP
Cupertino, CA 95014
August 3, 2006

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

HP Integrity rx8640

							3 yr. Maint.
Description	**Part Number**	**Sourc**	**Reference**	**Qty**	**Extended**	**Pri**	**Price**
Server Hardware							
HP Integrity rx8640 SMP Base System with 8 1.6GH	AB444A#002**	1	191,108	1	191,108		
4GB high-density DDR memory module (uses 2 DIMM	AB454A	1	8,000	32	256,000		
HP Integrity rx8640 I/O Backplane	AD160A	1	7,700	2	15,400		
3 Year Svc & Support Price (Hardware and Software)	HA110A3-6KT**	1					106,561
HP Integrity rx8640 Sys. Expansion Unit	AB301A	1	35,785	1	35,785		
HP Rack kit for rx86xx Server	J1528B	1	582	1	582		
HP Rack Kit for SEU Server	J1530C	1	709	1	709		
DVD+RW Drive	AB351B	1	850	1	850		
PCI 1000 Base T Dual Port LAN Adapter Card	A7012A		1,495	1	1,495		
PCI 2GB Fibre Channel Adapter (dual port)	A6826A	1	4,395	20	87,900		
36 GB HotPlug Ultra 320 SCSI Low Profile Disk (15k)	AD146A	1	1,200	3	3,600		
HP Server Thin Client (Console)	AB300B	1	1,250	1	1,250		
Rack Model 5642	358254-B21	1	689	1	689		
				Subtotal	**595,368**		**106,561**
Server Software							
HPUX 11i v2 Foundation Operating Environment	B9429AC**	1	2,370	16	37,920		
HPUX Fndn OE Media	B9106AA, Opt 0D1	1	199	1	199		
				Subtotal	**38,119**		**0**
Storage							
16 meter Fibre Optic Cable	221692-B22	1	82	40	3,280		
HP StorageWorks MSA 1000 (40+4 spares)	201723-B22	1	6,995	44	307,780	Included	
36GB 15K Ultra320 Hard Drive (480 + 48 spares)	286776-B22	1	269	528	142,032	Included	
10642 (42U) Rack Cabinet	245161-B21		1,359	4	5,436		
ProLiant Cluster HA/200 for MSA100	252409-B22	1	4,007	1	4,007		
				Subtotal	**462,535**		**0**
				Total	**1,096,022**		**106,561**
Large Configuration Discount and Support Prepayment*					**-276,437**		**-31,435**
				Grand Tot	**819,585**		**75,126**

This quote is valid for 90 days

From: Mary.Beth Pierantoni [mailto:mary.beth.pierantoni@oracle.com]
Sent: Thursday, July 27, 2006 11:27 AM
To: Boushey, Lucille
Subject: Oracle Pricing

Product	Price	Quantity	Extended Price
Oracle Database 10g Enterprise Edition, Named User Plus for 3 years	$10,000	*8	$80,000
Partitioning, Named User Plus for 3 years	$2,500	*8	$20,000
Oracle Database Server Support Package for 3 years	$2,000	3	6,000
Oracle Mandatory E-Business Discount			<$15,900>
Oracle TOTAL			$90,100

* 8 = 0.50 * 16. 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.

Oracle pricing contact: MaryBeth Pierantoni, mary.beth.pierantoni@oracle.com, 916-315-5081

Quote is valid for 90 days