TPC Benchmark™ H
Full Disclosure Report for

Bull Novascale 5160

Using

Oracle Database 10g Release 2

On

Microsoft® Windows® Server 2003, Datacenter Edition for 64-bit Itanium-based Systems

First Edition

Submitted for Review

June 20, 2005

First Edtion - June 16, 2005

Bull S.A. and Oracle Corporation, the sponsors of this benchmark test, believe that the information in this document is accurate as of the publication date. The information in this document is subject to change without notice. Bull and Oracle Corporation assume no responsibility for any errors that may appear in this document.

The pricing information in this document is believed to reflect accurately the current prices as of the publication date. However, we provide no warranty on the pricing information in this document.

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 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, and therefore results obtained in other operating environments may vary significantly. We do not warrant or represent that a user can or will achieve similar performance expressed in composite query-per-hour ratings. No warranty of system performance or price/performance is expressed or implied with this document.

Bull and Oracle Corporation reserve the right to make changes in specifications and other information contained in this document without prior notice, and the reader should in all cases consult Bull or/and Oracle Corporation to determine whether any such changes have been made.

Printed in FRANCE, June 2005

The following terms used in this publication are trademarks of their respective companies:

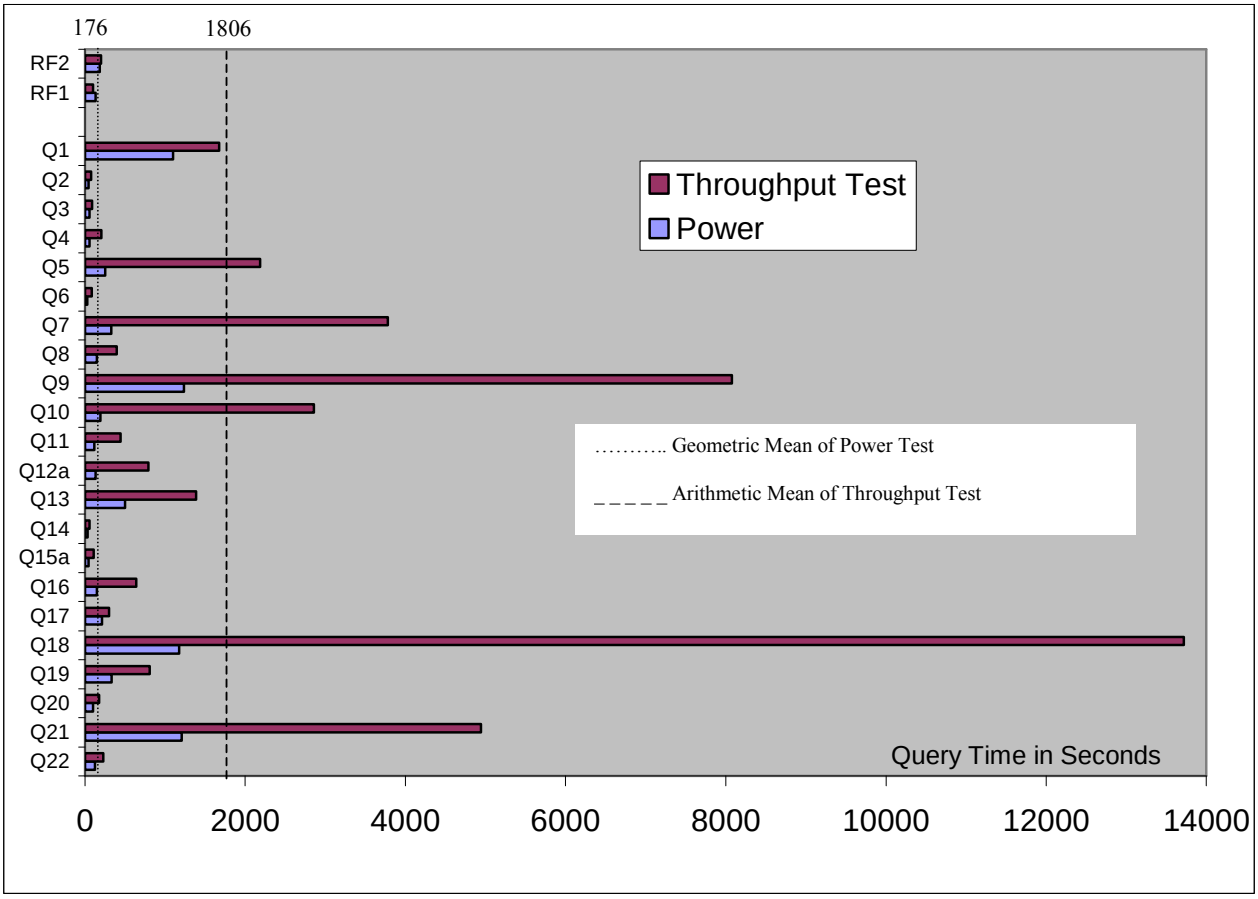
TPC Benchmark	Trademark of the Transaction Processing Performance Council
TPC-H, QppH, QthH and QphH	Trademark of the Transaction Processing Performance Council
Novascale 5160	Trademark of the Bull company
Windows® Server 2003	Trademark of the Microsoft® Corporation
Itanium	Trademark of the Intel Corporation

Other product names mentioned in this document may be trademarks and/or registered trademarks of their respective companies.

Bull / ORACLE	Bull Novascale 5160	TPC-H Rev. 2.1.0 Report Date June 20, 2005

Total System Cost	Composite Query per Hour Rating	Price Performance
$667,962	**15069.9 QphH @ 1000GB**	**$44.33 / QphH @ 1000GB**

Database size	Database Manager	Operating System	Other Software	Availability Date
1000 GB*	Oracle Database 10g Release 2 with Oracle Automatic Storage Management	Microsoft Windows Server 2003, Datacenter Edition for 64-bit Itanium-based Systems	gcc-c++-3.2.3-20	**December 20, 2005**



Database Load Time = 11:33:52	Load included backup: N	Total Data Storage / Database Size = 12.64
RAID (Base tables): Y	RAID (Base tables and Auxilieary Data Structures): Y	RAID (All): N

System Configuration			
Processors	16 x 1.6 GHz Intel® Itanium2™ with 6MB Level 3 Cache		
Memory	64 GB Main Memory		
Disk Controllers	15 LSI 7202XP -LC		
Disk Drives	2 184	73GB FB 10K73GB SCSI 15K *(1GB = 10^9byte)*	
Total Disk Storage		12645 GB *(1GB=2^30 bytes)*	

* Database size includes only raw data (eg, no temp, index, redundant storage space, etc.)

Bull / ORACLE	Bull Novascale 5160				TPC-H REV 2.1.0 Report Date: June 20, 2005		
Description	**Part Number**	**Brand**	**Pricing**	**Unit Price**	**Qty**	**Extended Price**	**3 yr. Maint. Price**
Server Hardware							
NovaScale 5160 System Quad Itanium 2 1.6GHz/6MB	CPXU404-1606	Bull	1	100 138	1	100 138	36 501
Including:							
1 Rack 36U/19" with Power Distribution Unit (PDU)	Included	Bull					
1 Platform Administrator Processor (PAP)	Included	Bull					
1 Keyboard/Video/Mouse (KVM) 8-port switch	Included	Bull					
1 LCD console drawer with keyboard and mouse	Included	Bull					
1 CPU kit: 4 x Itanium2 1.6GHz/6MB L3	Included	Bull					
1 I/O Box (11 PCI-X slots)	Included	Bull					
CPU Kit: 4x Itanium 2 1.6GHz/6MB L3, 1 CPU/M Box	CPKU307-1606	Bull	1	29 108	3	87 324	27 658
16GB (16x1024MB DIMMS, SDRAM DDR) kit for one CPU/Memory box	CMMU303-0000	Bull	1	18 767	4	75 068	0
One additionnal I/O Box including 11 add'l PCI slots	DRWU300-0000	Bull	1	15 400	1	15 400	5 006
PCI fibre optical 2 Gb/s host connection kit, w/o cable - Windows and HPC Linux	CKTU213-0000	Bull	1	1 817	1	1 817	574
FDA 1300 Fiber RAID 15-slot disks subsystem	DSSU070-1300	Bull	1	8 712	1	8 712	2 753
FDA 1300 Storage Manager Base	EXSU400-WK00	Bull	1	1 980	1	1 980	959
73GB Raid 2G Fiber Disk (10Krpm) For FDA	MSUU302-0FDA	Bull	1	517	2	1 034	584
One dummy disk module for FDA	MSUU30D-0FDA	Bull	1	40	13	525	
Additionnal Power Distribution Unit	PSSU200-0000	Bull	1	924	1	924	0
					Subtotal	**292 923**	**74 036**
Storage Hardware							
Rack 1100H 40U/19" w/o Power Distribution Unit	RCKU209-1140	Bull	1	3 681	1	3 681	1 163
7-outlet Power Distribution Unit (7x C13)	PSSU211-00M7	Bull	1	266	4	1 063	336
Dual Channel HBA PCI-X to fiber	LSI 7402	LSI	3	1 132	17	19 244	
16-slot fiber JBOD disk rack drawer	NS39-S05A134	Bull	1	3 807	15	57 105	18 047
SFP for disk drawer	NS39-S05A136	Bull	1	92	33	3 036	
FC multimode optical LC to LC 2 Gb SAN connection cable - 5	CBLH008-0500	Bull	1	200	30	6 006	
73GB/15k Fiber disk	NS39-S05A135	Bull	1	708	203	143 724	
					Subtotal	**233 859**	**19 546**
Server Software							
Novascale Master	Included	Bull	1				
Windows Server 2003 DTC Ed. (up to 16-CPU) & 5CAL	EXSU264-NA0H	Microsoft	1	29 543	1	29 543	3 445
Oracle Database 10g Release 2, Enterprise Edition, Named User Plus for 3 years		Oracle	4	10 000	16	160 000	*Inc. Below*
Partitioning, Named User Plus for 3 years		Oracle	4	2 500	16	40 000	*Inc. Below*
Database Server Support Package for 3 years		Oracle	4	2 000	3		6 000
Visual C++ Standard Edition	254-00170	Microsoft	2	109	1	109	*Inc. Below*
Microsoft Problem Resolution Service		Microsoft	2		1		245
					Subtotal	**229 652**	**9 690**
Large Volume HW, SW & Maintenance Discount					**Discounts**	157 784	33 960
					TOTAL	**598 649**	**69 313**

Notes:

Pricing : 1-Bull, 2-Microsoft, 3-LSI, 4-Oracle

(*) Spares = 10% spare disks, SFP and PCI-X HBA added in place of onsite service

Used 15 HBA, 30 SFP, 184x73Gb disks - Spares are 2 HBA, 3 SFP, 19x73Gb disks

Result and methodology audited by François Raab of InfoSizing Inc. (www.sizing.com)

Three-Year Cost of Ownership : $667 962

QphH @ 1000GB : 15 069

$ / QphH@1000GB : 44,33

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

2

Numerical Quantities Summary

Measurement Results

Scale Factor	1000 GB
Total Data Storage / Database Size	12.64
Start of Database Load	08/06/05 14:37:36
End of Database Load	09/06/05 02:11:23
Database Load Time	11:33:47
Query Streams for Throughput Test	7
TPC-H Power	20385.3
TPC-H Throughput	11140.4
Composite Query per Hour Rating (QphH@1000GB)	15069.9
Total System Price Over 3 Years	$667,962
TPC-H Price Performance Metric	$44.33

Measurement Intervals

Measurement Interval in Throughput Test (Ts)	49765 seconds (13:49:25)

Duration of Stream Execution:

	Seed	Query Start Date/Time	RF1 Start Date/Time	RF2 Start Date/Time	Duration
		Query End Date/Time	RF1 End Date/Time	RF2 End Date/Time	
Stream 00	609021123	06/09/05 03:28:59 2005	06/09/05 03:26:45 2005	06/09/05 05:35:29 2005	2:11:52
		06/09/05 05:35:29 2005	06/09/05 03:28:58 2005	06/09/05 05:38:37 2005	
Stream 01	609021124	06/09/05 05:38:37 2005	06/09/05 18:52:41 2005	06/09/05 18:54:17 2005	11:43:40
		06/09/05 17:22:17 2005	06/09/05 18:54:17 2005	06/09/05 18:57:20 2005	
Stream 02	609021125	06/09/05 05:38:42 2005	06/09/05 18:57:20 2005	06/09/05 18:59:05 2005	13:12:55
		06/09/05 18:51:37 2005	06/09/05 18:59:05 2005	06/09/05 19:02:22 2005	
Stream 03	609021126	06/09/05 05:38:56 2005	06/09/05 19:02:22 2005	06/09/05 19:04:02 2005	9:40:06
		06/09/05 15:19:02 2005	06/09/05 19:04:02 2005	06/09/05 19:07:40 2005	
Stream 04	609021127	06/09/05 05:38:55 2005	06/09/05 19:07:40 2005	06/09/05 19:09:20 2005	12:34:45
		06/09/05 18:13:40 2005	06/09/05 19:09:20 2005	06/09/05 19:12:50 2005	
Stream 05	609021128	06/09/05 05:39:02 2005	06/09/05 19:12:50 2005	06/09/05 19:14:34 2005	12:39:00
		06/09/05 18:18:02 2005	06/09/05 19:14:34 2005	06/09/05 19:18:10 2005	
Stream 06	609021129	06/09/05 05:39:10 2005	06/09/05 19:18:10 2005	06/09/05 19:19:49 2005	13:13:31
		06/09/05 18:52:41 2005	06/09/05 19:19:49 2005	06/09/05 19:22:58 2005	
Stream 07	609021130	06/09/05 05:39:08 2005	06/09/05 19:22:58 2005	06/09/05 19:24:36 2005	10:38:40
		06/09/05 16:17:48 2005	06/09/05 19:24:36 2005	06/09/05 19:28:02 2005	

TPC-H Timing Intervals (in seconds)

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8
Stream 00	1100.0	46.6	58.6	57.0	251.2	28.8	330.0	147.9
Stream 01	1455.9	91.7	109.8	112.0	2726.4	78.9	2814.6	258.5
Stream 02	1355.0	82.0	95.8	137.8	2078.3	58.6	2061.6	249.0
Stream 03	2121.7	67.7	99.3	346.6	1964.5	118.4	3937.9	792.0
Stream 04	1517.7	98.5	73.2	260.6	2664.5	64.8	4246.7	518.7
Stream 05	1701.8	67.6	73.3	177.4	1146.4	65.8	4357.8	217.0
Stream 06	1962.7	57.3	95.9	197.3	2865.4	145.6	4649.4	470.2
Stream 07	1631.6	78.5	91.2	204.4	1865.0	53.6	4392.6	259.3
Min Qi	1355.0	57.3	73.2	112.0	1146.4	53.6	2061.6	217.0
Avg Qi	1678.0	77.6	91.2	205.1	2187.2	83.7	3780.1	394.9
Max Qi	2121.7	98.5	109.8	346.6	2865.4	145.6	4649.4	792.0
	Q9	**Q10**	**Q11**	**Q12**	**Q13**	**Q14**	**Q15**	**Q16**
Stream 00	1237.3	195.8	117.4	130.4	501.4	34.8	46.4	148.4
Stream 01	11587.1	2273.7	330.2	283.3	936.5	47.9	111.8	661.2
Stream 02	12001.6	4183.9	662.9	480.1	1172.4	71.1	96.2	747.1
Stream 03	3097.7	2581.6	343.2	352.8	1831.9	76.2	119.1	529.1
Stream 04	5997.5	2201.1	406.9	738.3	2087.8	54.6	97.7	674.9
Stream 05	6704.5	2850.8	452.9	497.9	1127.4	52.7	134.2	835.7
Stream 06	9923.2	4207.3	194.5	1608.2	1495.8	62.7	129.6	601.9
Stream 07	7222.1	1716.1	712.7	1603.9	1070.2	43.3	76.6	439.6
Min Qi	3097.7	1716.1	194.5	283.3	936.5	43.3	76.6	439.6
Avg Qi	8076.2	2859.2	443.3	794.9	1388.8	58.3	109.3	641.4
Max Qi	12001.6	4207.3	712.7	1608.2	2087.8	76.2	134.2	835.7
	Q17	**Q18**	**Q19**	**Q20**	**Q21**	**Q22**	**RF1**	**RF2**
Stream 00	213.7	1176.5	333.3	99.9	1211.1	124.0	133.0	187.8
Stream 01	241.4	10604.2	760.9	191.7	6364.6	177.6	95.9	183.4
Stream 02	492.3	17086.9	1154.4	165.2	2944.5	197.5	105.0	196.4
Stream 03	292.5	9928.8	862.9	174.2	4974.2	193.9	100.0	218.1
Stream 04	256.8	14804.9	687.6	173.0	7273.9	384.6	100.1	210.7
Stream 05	250.5	16994.9	909.0	183.8	6475.9	262.1	103.6	215.8
Stream 06	302.1	15029.5	608.6	166.8	2632.2	204.4	99.2	188.7
Stream 07	270.5	11598.5	669.8	179.6	3941.2	199.5	98.4	205.7
Min Qi	241.4	9928.8	608.6	165.2	2632.2	177.6	95.9	183.4
Avg Qi	300.9	13721.1	807.6	176.3	4943.8	231.4	100.3	202.7
Max Qi	492.3	17086.9	1154.4	191.7	7273.9	384.6	105.0	218.1

INFO SIZING

TPC TRANSACTION PROCESSING PERFORMANCE COUNCIL CERTIFIED AUDITOR

Benchmark Sponsor: Jean-François Lemerre
Bull S.A.
1, rue de Provence
38432 Echirolles
France

June 17, 2005

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

Platform: **Bull Novascale 5160**

Database Manager: **Oracle Database 10g Release 2 With Automatic Storage Mgt.**

Operating System: **Microsoft Windows Server 2003 Datacenter Edition for 64-bit**

The results were:

CPU (Speed)	Memory	Disks	**QphH@1,000GB**
Bull Novascale 5160			
16 x Intel Itanium 2 (1.6 GHz)	6 MB L3 Cache 64 GB Main	184 x 73 GB ext. 2 x 73 GB int.	**15,069.9**

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:

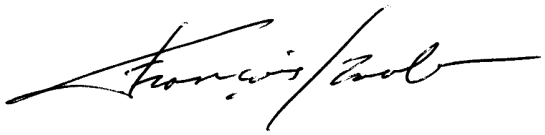
- The database records were defined with the proper layout and size
- The database population was generated using DBGEN
- The database was properly scaled to 1TB (1,000 GB) and populated accordingly
- The compliance of the database auxiliary data structures was verified
- The database load time was correctly 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 two query variants
- 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

Third Party Quotes

Microsoft Corporation Tel 425 882 8080
One Microsoft Way Fax 425 936 7329
Redmond, WA 98052-6399 http://www.microsoft.com/

Microsoft

June 15, 2005

Bull
Francy Jourdan
Ave. Jean Jaures - BP 68
Les Clayes-sous-Bois, 78340

Mr. Jourdan:

Here is the information you requested regarding pricing for several Microsoft products to be used in conjunction with your TPC-H benchmark testing.

All pricing shown is in US Dollars ($).

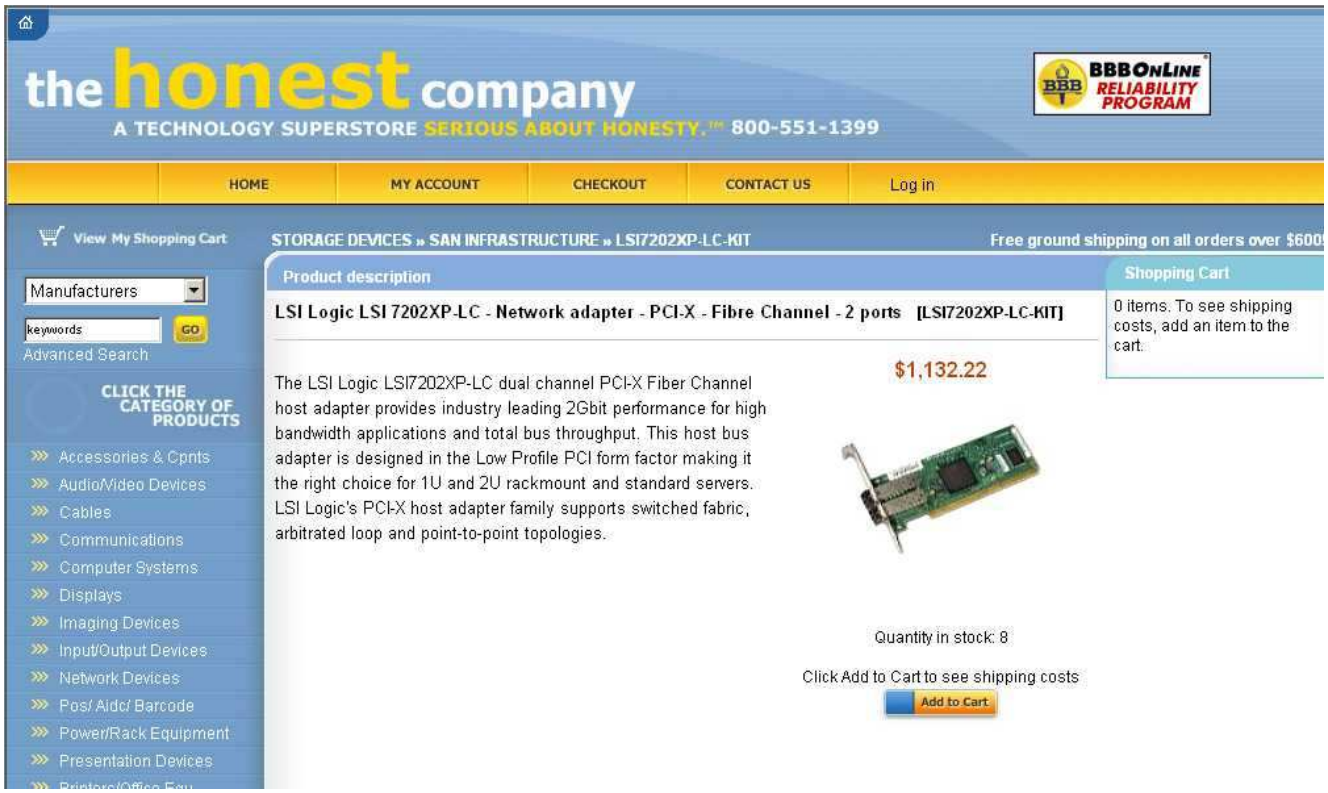
Part Number	Description	Unit Price	Quantity	Price
254-00170	**Visual C++ Standard Edition** *Discount Schedule: No Discounts Applied*	$109	1	$109
	Microsoft Problem Resolution Services *Professional Support (1 incident)*	$245	1	$245

All products are currently orderable through Microsoft's normal distribution channels.

This quote is valid for the next 90 days.

If we can be of any further assistance, please contact Jamie Reding at (425) 703-0510 or jamiere@microsoft.com.

Reference ID: PHfrjo0515063126.
Please include this Reference ID in any correspondence regarding this price quote.



De : MaryBeth Pierantoni [mailto:mary.beth.pierantoni@oracle.com]
Envoyé : lundi 23 mai 2005 22:07
À : Eric Thomas
Objet : Oracle Pricing

Product	Price	Quantity	Extended Price
Oracle Database 10g Enterprise Edition Release 2, Named User Plus for 3 years	$10,000	16	$160,000
Partitioning, Named User Plus for 3 years	$2,500	16	$40,000
Database Server Support Package for 3 years	$2,000	3	$6,000
Oracle Mandatory E-Business Discount			<$30,900>
TOTAL			$175,100

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

PREFACE

Document Overview

This report documents the methodology and results of the TPC Benchmark™ H (TPC-H) test conducted on the Bull Novascale 5160 using Oracle Database 10g, Enterprise Edition, Release 2, in conformance with the requirements of the TPC Benchmark™H Standard Specification Revision 2.1.0. The tests documented in this report were sponsored by Bull and Oracle Corporation. The operating system used for the benchmark was Microsoft Windows Server 2003, Datacenter Edition for 64-bit Itanium-based Systems.

The Transaction Processing Performance Council (TPC) developed the TPC-H Benchmark. The TPC Benchmark™ H Standard represents an effort by Bull, Oracle Corporation and other members of the Transaction Processing Performance Council (TPC) to create an industry-wide benchmark for evaluating the performance and price/performance of decision support systems, and to disseminate objective, verifiable performance data to the data processing industry.

A certified audit of these measurements and the reported results was performed by François RAAB of Info Sizing Inc. He has verified compliance with the relevant TPC Benchmark™ H specifications; audited the benchmark configuration, environment, and methodology used to produce and validate the test results; and audited the pricing model used to calculate the price/ performance. The auditor's letter of attestation is attached to the Executive Summary and precedes this section.

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 updates. 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:

- he 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 gigabyte. Compliant benchmark implementations may also use one of the larger permissible database populations (e.g., 1000 gigabytes), as defined in Clause 4.1.3.

The performance metric reported by TPC-H is called the TPC-H Composite Query-per-Hour Performance Metric (QphH@Size), and reflects multiple aspects of the capability of the system to process queries. These aspects include the selected database size against which the queries are executed, the query processing power when queries are submitted by a single stream, and the query throughput when queries are submitted by multiple concurrent users. The TPC-H Price/Performance metric is expressed as $/QphH@Size. To be compliant with the TPC-H standard, all references to TPCH results for a given configuration must include all required reporting components. *The TPC believes that comparisons of TPC-H results measured against diferent 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-H 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.

A Table of Contents follows after this page.

Related Product Information

The TPC Benchmark™ H Standard requires that test sponsors provide a Full Disclosure Report in addition to published results. You can obtain copies of the test results as well as additional copies of this full disclosure report by sending a request to the following address:

Bull S.A.
Rue Jean Jaurès
B.P.68
78340 Les Clayes-sous-Bois
France

Table of contents

1. GENERAL ITEMS

1.1 Benchmark Sponsor

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

This TPC benchmark H was sponsored by Bull S.A. and Oracle Corporation. The benchmark test was developed by Bull and Oracle Corporation. The benchmark was conducted at Bull in Les Clayes sous Bois, France.

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:

- *Data Base 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.*

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

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

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

1.3 Configuration Diagrams

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

- *Number and type of processors;*
- *Size of allocated memory, and any specific mapping/partitioning of memory unique to the test;*
- *Number and type of disk units (and controllers, if applicable);*
- *Number of channels or bus connections to disk units, including their protocol type;*
- *Number of LAN (e.g. Ethernet) connections, including routers, work stations, terminals, etc., that were physically used in the test or are incorporated into the pricing structure;*
- *Type and run-time execution location of software components (e.g., DBMS, query processing tools/languages, middle-ware components, software drivers, etc.).*

The server System Under Test (SUT), a Bull Novascale 5160 is shown in the following diagram :

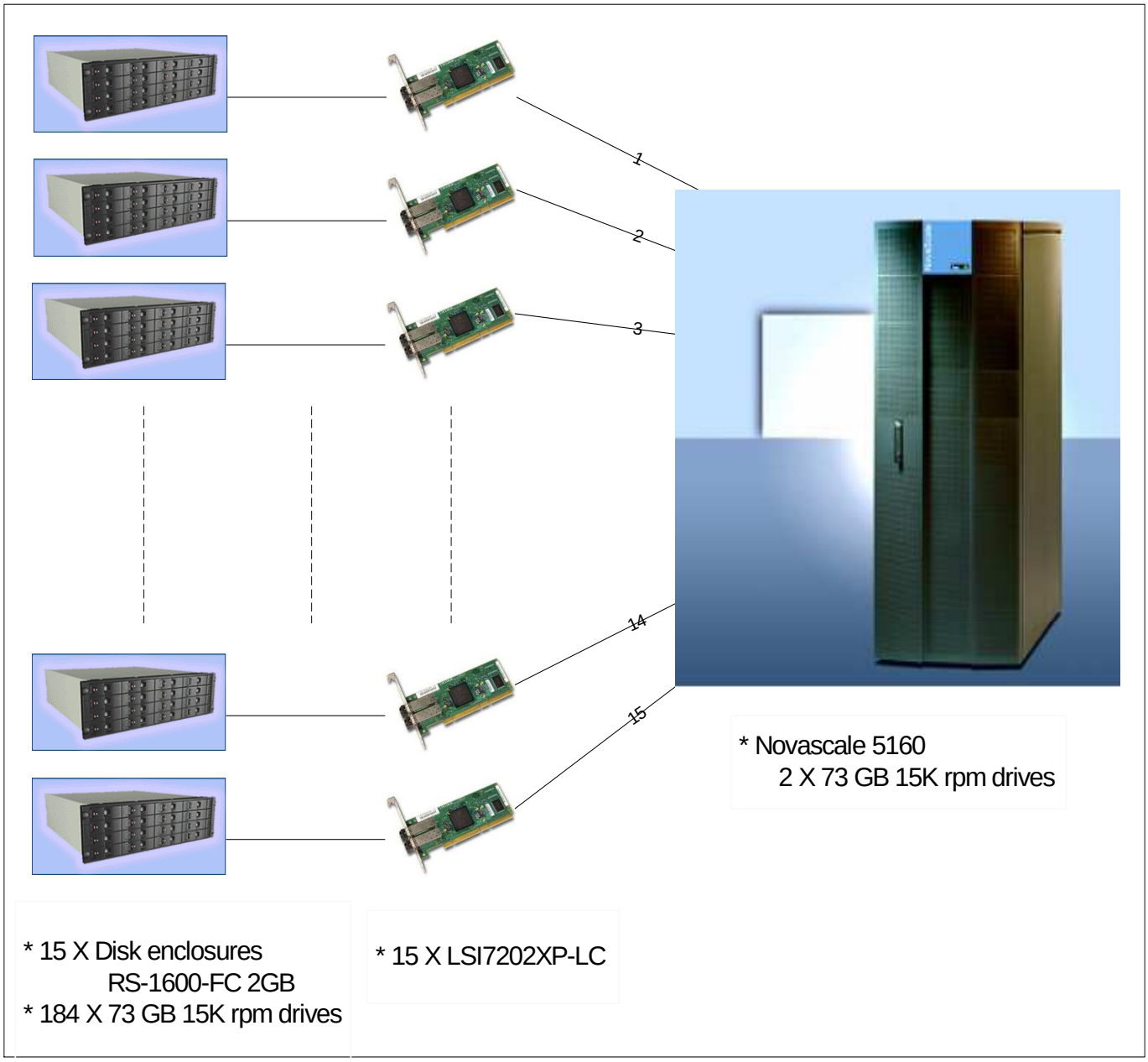


Figure 1.1 Benchmark and Priced Configuration for Novascale 5160 (16 SMP)

2 CLAUSE 1: LOGICAL DATA BASE DESIGN

2.1 Table Definitions

Listings must be provided for all table definition statements and all other statements used to setup the test and qualification databases. Appendix B contains the scripts that define, create, and analyze the tables and indexes for the TPC-H database.

2.2 Database Organization

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

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

2.3. Horizontal Partitioning

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

Horizontal partitioning was used for all tables except NATION and REGION. Refer to the table/index create statements in Appendix B for more details. Similar partitioning was used in the qualification database size.

2.4 Vertical Partitioning

Vertical partitioning of tables is not allowed. For example, groups of columns of one row shall not be assigned to files, disks, or areas different from those storing the other columns of that row. The row must be processed as an atomic series of contiguous columns.

Comment: *The effect of vertical partitioning is to reduce the effective row size accessed by the system. Given the synthetic nature of this benchmark, the effect of vertical partitioning is achieved by the choice of row sizes. No further vertical partitioning of the data set is allowed. Specifically, the above Clause prohibits assigning one or more of the columns not accessed by the TPC-H query set to a vertical partition.*

Vertical partitioning was not used. See Appendix B, which contains the database and table creation statements.

2.5 Replication

Any replication of physical objects must be disclosed and must conform to the requirements of Clause1.5.6.

No replication was used. See Appendix B, which contains the database and table creation statements.

3 CLAUSE 2: QUERIES AND UPDATE 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 Random Number Generation

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

DBGEN Version 1.3.0 and QGEN version 1.3.0 were used to generate random numbers for all database populations.

3.3 Substitution Parameters

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

The supplied QGEN version 1.3.0 was used to generate the substitution parameters.

3.4 Query Text and Output Data from Qualification Database

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

The query text was produced using minor modifications and two query variants:

- Variant A for Q12
- Variant A for Q15

Appendix C contains the query text.

3.5 Query Substitution Parameters and Seeds

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

Appendix D contains the seed and query substitution parameters.

3.6 Query Isolation Level

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

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

3.7 Source Code of Refresh Functions

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

Appendix E contains the source code for the refresh functions.

4. CLAUSE 3: DATABASE SYSTEM PROPERTIES

4.1 Atomicity

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

4.1.1 Completed Transaction

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

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

4.1.2 Aborted Transaction

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

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

4.2 Consistency

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

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

O_TOTALPRICE = SUM(L_EXTENDEDPRICE – L_DISCOUNT) * (1 + L_TAX)
For each ORDER and LINEITEM defined by (O_ORDERKEY = L_ORDERKEY)

4.2.1 Consistency Test

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

The consistency of the ORDER and LINEITEM tables was verified based on randomly selected values of

the column O_ORDERKEY.

1. Acid queries were executed to verify the initial consistent state of the ORDER and LINEITEM tables.
2. More than 100 Acid transactions were submitted from each of two execution streams.
3. Acid queries were re-executed to verify the consistent state of the ORDER and LINEITEM tables after the Acid transaction streams.
4. The consistency of the ORDER and LINEITEM tables was re-verified.

To guarantee arithmetic function portability and consistency of results, the following expression was executed to verify the consistency between the ORDER and LINEITEM tables:

O_TOTALPRICE = SUM(ROUND((ROUND((L_EXTENDEDPRICE * (1 - L_DISCOUNT)),2,1) * (1 + L_TAX)),2,1))

4.3 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.

4.3.1 Read-Write Conflict with Commit

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

1. An Acid transaction was started for a randomly selected O_KEY, L_KEY, and DELTA. The Acid transaction was suspended prior to COMMIT.

2. An ACID query was started for the same O_KEY used in Step 1. The ACID query completed and did not see the uncommitted changes made by the Acid transaction.

3. The Acid transaction was COMMITTED.

4.3.2 Read-Write Conflict with Rollback

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

1. An ACID transaction was started for a randomly selected O_KEY, L_KEY, and DELA. The ACID transaction was suspended prior to ROLLBACK.

2. An ACID query was started for the same O_KEY used in Step 1. The ACID query did not see the uncommitted changes made by the ACID transaction.

3. The ACID transaction was ROLLED BACK.

4. The ACID query completed.

4.3.3 Write-Write Conflict with Commit

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

1. An ACID transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID transaction 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 was calculated correctly.
   T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE +(DELTA1*(T1.L_EXTENDEDPRICE/T1.L_QUANTITY))

4.3.4 Write-Write Conflict with Rollback

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

1. An Acid transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. The Acid transaction was suspended prior to ROLLBACK.

2. Another Acid transaction, T2, was started using the same O_KEY and L_KEY and a different randomly selected DELA.

3. T2 waited

4. T1 was allowed to ROLLBACK and T2 completed

5. It was verified that T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE.

4.3.5 Concurrent Progress of Read and Write on Different Tables

Demonstrate the ability of read and write transactions afecting diferent 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.

3. ACID Transaction T2 completed.

4. ACID transaction T1 completed and the appropriate rows in the ORDER, LINEITEM, and HISTORY tables were changed.

4.3.6 Updates not Indefinitely Delayed by Reads on Same Table

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

1. An ACID transaction, T1, was started, executing Q1 against the qualification database. The substitution parameter was chosen from the interval [0..2159] so that the query ran for a sufficient length of time.

2. Before T1 completed, an ACID transaction, T2, was started using randomly selected values of O_KEY, L_KEY and DELTA.

3. T2 completed before T1 completed. Verified that the appropriate rows in ORDER, LINEITEM and HISTORY tables have been changed.

4.4 Durability

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

4.4.1 Failure of a Durable Medium and System Crash

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

The database tables and logs were placed on Automatic Storage Management (ASM) disk groups. The partitions were created on 4 logical drives of the same characteristics as the drives used for the Test database, and the four drives were on two separate controllers.

1. The data files and log files were created into one disk group using ASM's normal redundancy mirroring option.

2. Eight streams of ACID transactions were started.

3. After at least 100 transactions had occurred on each stream and the streams were still running, one ASM disk drive was removed.

4. After it was determined that the test would still run with the data and log drive removed, and after running at least another 100 transactions on each stream, the Oracle database was powered off.

5. The data and log drive was inserted back in the system.

6. When power was restored the system rebooted and the database was restarted.

7. The data files were restored to their state prior to the ACID transaction streams.

8. The database ran through its recovery mode.

9. The counts in the success files and the HISTORY table count were compared and the counts matched.

5. CLAUSE 4: SCALING AND DATABASE POPULATION

5.1 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	# of ROWS
Orders	1,500,000,000
Lineitem	5,999,989,709
Customer	150,000,000
Parts	200,000,000
Supplier	10,000,000
Partsupp	800,000,000
Nation	25
Region	5

5.2 Distribution of Tables and Logs Across Media

The distribution of tables and logs across all media must be explicitly described using a format similar to that shown in the following example for both the tested and priced systems.

The SUT and the priced system have 184 external drives and 2 internal drives.
Utilization of the drives:

- • 180 physical/logical drives for the 1000GB database. See Appendix B for exact disk configuration.
- • Two Oracle ASM diskgroups namely *groupe1* and *groupe2* were created on the SUT. *groupe1* consists of 180 external drives and *groupe2* consists of 4 external drives.
- • The Test Database, its Log and Temp files were created on ASM diskgroup *groupe1*. The Test Database and its Log were mirrored by Oracle ASM. Temp was not mirrored. For ASM scripts, refer to APPENDIX B.
- • The ACID database was created on Oracle ASM diskgroup *groupe2*.
- • The operating system, Microsoft Windows Server 2003, Datacenter Edition for 64-bit Itanium-based systems, and Oracle Database 10g Release 2 Enterprise Edition, as well as part of the operating system page file, were installed on the internal drive.

5.3 Partitions/Replications Mapping

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

Comment: *The intent is to provide sufficient detail about partitioning and replication to allow independent reconstruction of the test database.*

The database was not replicated.
Horizontal partitioning was used for base tables LINEITEM, ORDERS, PARTSUPP, PART, SUPPLIER and CUSTOMER. The
details for this partitioning can be understood by examining the syntax of the table and index definition statements in Appendix B.

5.4 Use of RAID

Implementations may use some form of RAID . The RAID level used must be disclosed for each device.

No hardware RAID was used in the implementation. The test database and its log, as well as the qualification database and its log were mirrored using Oracle ASM. Temp was not mirrored.

5.5 DBGEN Modifications

The version number, release number, modification number, and patch level of DBGEN must be disclosed. Any modifications to the DBGEN 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 1.3.0 was used for populating the database.

5.6 Database Load Time

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

The Numerical Quantities summary (p. 5) contains the database load time, which was 11:33:52

5.7 Data Storage Ratio

The data storage ratio must be disclosed. It is computed by dividing the total data storage of the priced configuration (expressed in GB) by the size chosen for the test database. The ratio must be reported to the nearest 1/100th, rounded up. For example, a system configured with 96 disks of 2.1 GB capacity for a 100GB test database has a data storage ratio of 2. 02.

Comment: *For the reporting of configured disk capacity, gigabyte (GB) is defined to be 2^30 bytes. Since disk manufacturers typically report disk size using base ten (i.e., GB = 10^9), it may be necessary to convert the advertised size from base ten to base two.*

Disk Type	# of Disks	Space Per Disk*	Subtotal Disk Space**
Internal	2	73 GB	135 GB
External	184	73 GB	12509 GB
		Total	12644 GB

**Disk manufacturer definition of one GB is 10^9 byte* ***In this calculation one GB is defined as 2^30 bytes*

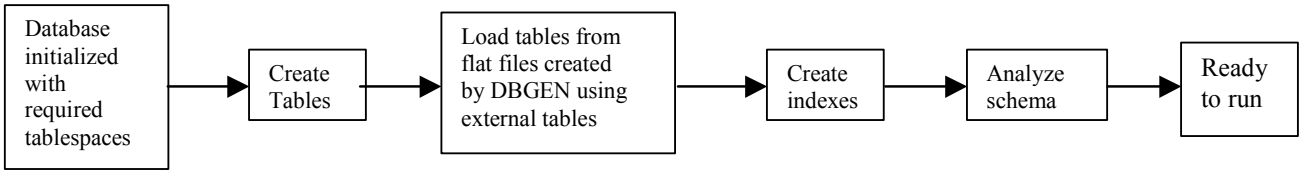
The Numerical Quantities summary (p. 5) contains the data storage ratio (12.64) for the system used.

5.8 Database Loading

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

The following steps were used to load the database:

1) DBGEN version 1.3.0 was used to create flat files.
2) The database was created and loaded from those flat files using the scripts in Appendix B.



5.9 Qualification Database Configuration

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

The qualification database was created using scripts identical to those of the test database, except for variances due to the sizes of the two databases.

6. Clause 5: Performance Metrics and Execution Rules

6.1 System Activity Between Load and Performance Tests

Any system activity on the SUT which takes place between the conclusion of the load test and the beginning of the performance test must be fully disclosed including listings of scripts or command logs.
Auditor requested queries were run against the database to verify the completeness and correctness of the database load. All scripts and queries used are included in Appendix E

6.2 Power Test Implementation

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 followed to run the power test.

1. RF 1 refresh transaction
2. Stream 00 execution
3. RF2 refresh transaction

6.3 Timing Intervals and Reporting

The timing intervals for each query and 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. For convenience, it is repeated in Section 6.10.

6.4 Number of Streams in the Throughput Test

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

Seven streams were run for the throughput test.

6.5 Start and End Date/Time for Each Query Stream

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

The throughput test start time and finish time for each stream are given in the Numerical Quantity Summary earlier in this document. For convenience, it is repeated in Section 6.10.

6.6 Total Elapsed Time for 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 Quantity Summary earlier in this document. For convenience, it is repeated in Section 6.10.

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

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

The start and finish times for each refresh function in the refresh stream are given in the Numerical Quantity Summary earlier in this document. For convenience, it is repeated in Section 6.10.

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 refresh function for the throughput test are given in the Numerical Quantity Summary earlier in this document. For convenience, it is repeated in Section 6.10.

6.9 Performance Metrics

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

The performance metrics, and the numbers on which they are based, are given in the Numerical Quantity Summary earlier in this document. For convenience, it is repeated in Section 6.10.

6.10 The Performance Metric and Numerical Quantities from Both Runs

*The performance metric (QphH) and the numerical quantities (TPC-*H Power@Size and *TPC-H Throughput@Size) from both of the runs must be disclosed.*

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

Run ID	**QppH@1000GB**	**QthH@1000GB**	**QphH@1000GB**
Run 1	20385.3	11140.4	15069.9
Run 2	20939.0	11177.0	15298.2
% Difference	+2.72%	+0.33%	+1.51%

(Run 1 was reported.)

Tables from Numerical Quantities pages in the front of this report, reprinted here for reader's convenience:
See next page

Numerical Quantities Summary

Measurement Results

Scale Factor	1000 GB
Total Data Storage / Database Size	12.64
Start of Database Load	08/06/05 14:37:36
End of Database Load	09/06/05 02:11:23
Database Load Time	11:33:47
Query Streams for Throughput Test	7
TPC-H Power	20385.3
TPC-H Throughput	11140.4
Composite Query per Hour Rating (QphH@1000GB)	15069.9
Total System Price Over 3 Years	$667,962
TPC-H Price Performance Metric	$44.33

Measurement Intervals

Measurement Interval in Throughput Test (Ts)	49765 seconds (13:49:25)

Duration of Stream Execution:

	Seed	Query Start Date/Time	RF1 Start Date/Time	RF2 Start Date/Time	Duration
		Query End Date/Time	RF1 End Date/Time	RF2 End Date/Time	
Stream 00	609021123	06/09/05 03:28:59 2005	06/09/05 03:26:45 2005	06/09/05 05:35:29 2005	2:11:52
		06/09/05 05:35:29 2005	06/09/05 03:28:58 2005	06/09/05 05:38:37 2005	
Stream 01	609021124	06/09/05 05:38:37 2005	06/09/05 18:52:41 2005	06/09/05 18:54:17 2005	11:43:40
		06/09/05 17:22:17 2005	06/09/05 18:54:17 2005	06/09/05 18:57:20 2005	
Stream 02	609021125	06/09/05 05:38:42 2005	06/09/05 18:57:20 2005	06/09/05 18:59:05 2005	13:12:55
		06/09/05 18:51:37 2005	06/09/05 18:59:05 2005	06/09/05 19:02:22 2005	
Stream 03	609021126	06/09/05 05:38:56 2005	06/09/05 19:02:22 2005	06/09/05 19:04:02 2005	9:40:06
		06/09/05 15:19:02 2005	06/09/05 19:04:02 2005	06/09/05 19:07:40 2005	
Stream 04	609021127	06/09/05 05:38:55 2005	06/09/05 19:07:40 2005	06/09/05 19:09:20 2005	12:34:45
		06/09/05 18:13:40 2005	06/09/05 19:09:20 2005	06/09/05 19:12:50 2005	
Stream 05	609021128	06/09/05 05:39:02 2005	06/09/05 19:12:50 2005	06/09/05 19:14:34 2005	12:39:00
		06/09/05 18:18:02 2005	06/09/05 19:14:34 2005	06/09/05 19:18:10 2005	
Stream 06	609021129	06/09/05 05:39:10 2005	06/09/05 19:18:10 2005	06/09/05 19:19:49 2005	13:13:31
		06/09/05 18:52:41 2005	06/09/05 19:19:49 2005	06/09/05 19:22:58 2005	
Stream 07	609021130	06/09/05 05:39:08 2005	06/09/05 19:22:58 2005	06/09/05 19:24:36 2005	10:38:40
		06/09/05 16:17:48 2005	06/09/05 19:24:36 2005	06/09/05 19:28:02 2005	

TPC-H Timing Intervals (in seconds)

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8
Stream 00	1100.0	46.6	58.6	57.0	251.2	28.8	330.0	147.9
Stream 01	1455.9	91.7	109.8	112.0	2726.4	78.9	2814.6	258.5
Stream 02	1355.0	82.0	95.8	137.8	2078.3	58.6	2061.6	249.0
Stream 03	2121.7	67.7	99.3	346.6	1964.5	118.4	3937.9	792.0
Stream 04	1517.7	98.5	73.2	260.6	2664.5	64.8	4246.7	518.7
Stream 05	1701.8	67.6	73.3	177.4	1146.4	65.8	4357.8	217.0
Stream 06	1962.7	57.3	95.9	197.3	2865.4	145.6	4649.4	470.2
Stream 07	1631.6	78.5	91.2	204.4	1865.0	53.6	4392.6	259.3
Min Qi	1355.0	57.3	73.2	112.0	1146.4	53.6	2061.6	217.0
Avg Qi	1678.0	77.6	91.2	205.1	2187.2	83.7	3780.1	394.9
Max Qi	2121.7	98.5	109.8	346.6	2865.4	145.6	4649.4	792.0
	Q9	**Q10**	**Q11**	**Q12**	**Q13**	**Q14**	**Q15**	**Q16**
Stream 00	1237.3	195.8	117.4	130.4	501.4	34.8	46.4	148.4
Stream 01	11587.1	2273.7	330.2	283.3	936.5	47.9	111.8	661.2
Stream 02	12001.6	4183.9	662.9	480.1	1172.4	71.1	96.2	747.1
Stream 03	3097.7	2581.6	343.2	352.8	1831.9	76.2	119.1	529.1
Stream 04	5997.5	2201.1	406.9	738.3	2087.8	54.6	97.7	674.9
Stream 05	6704.5	2850.8	452.9	497.9	1127.4	52.7	134.2	835.7
Stream 06	9923.2	4207.3	194.5	1608.2	1495.8	62.7	129.6	601.9
Stream 07	7222.1	1716.1	712.7	1603.9	1070.2	43.3	76.6	439.6
Min Qi	3097.7	1716.1	194.5	283.3	936.5	43.3	76.6	439.6
Avg Qi	8076.2	2859.2	443.3	794.9	1388.8	58.3	109.3	641.4
Max Qi	12001.6	4207.3	712.7	1608.2	2087.8	76.2	134.2	835.7
	Q17	**Q18**	**Q19**	**Q20**	**Q21**	**Q22**	**RF1**	**RF2**
Stream 00	213.7	1176.5	333.3	99.9	1211.1	124.0	133.0	187.8
Stream 01	241.4	10604.2	760.9	191.7	6364.6	177.6	95.9	183.4
Stream 02	492.3	17086.9	1154.4	165.2	2944.5	197.5	105.0	196.4
Stream 03	292.5	9928.8	862.9	174.2	4974.2	193.9	100.0	218.1
Stream 04	256.8	14804.9	687.6	173.0	7273.9	384.6	100.1	210.7
Stream 05	250.5	16994.9	909.0	183.8	6475.9	262.1	103.6	215.8
Stream 06	302.1	15029.5	608.6	166.8	2632.2	204.4	99.2	188.7
Stream 07	270.5	11598.5	669.8	179.6	3941.2	199.5	98.4	205.7
Min Qi	241.4	9928.8	608.6	165.2	2632.2	177.6	95.9	183.4
Avg Qi	300.9	13721.1	807.6	176.3	4943.8	231.4	100.3	202.7
Max Qi	492.3	17086.9	1154.4	191.7	7273.9	384.6	105.0	218.1

6.11 System Activity

Any activity on the SUT that takes place between the conclusion of Run1 and the beginning of Run2 must be fully disclosed including listings of scripts or command logs along with any system reboots or database restarts.

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

7. Clause 6: SUT and Driver Implementation Related Items

7.1 Driver

A detailed textual description of how the driver performs its functions, how its various components interact and any product functionalities or environmental setting on which it relies must be provided. All related source code, scripts and configuration files must be disclosed. The information provided should be suficient for an independent reconstruction of the driver.

The Power Test and Throughput Test are performed by a shell script called runTPCpt. QGEN is first called with a stream id of 0 to generate the QET for the Power Test. UF1 is then started by executing the runuf1.sh script.
Query submission follows, with the qexecpl.c program. The execution of the UF2 script runuf2.sh rounds out the Power Test execution.
Following the Power Test, QGEN is again called with the subsequent 7 stream ids to generate new QET for each Throughput Test. qexecpl.c is called simultaneously for all 7 streams to execute the queries as above. Then the runTPCHus script is called to run all 7 update pairs to finish the throughput run.

7.2 Implementation-Specific Layer (ISL)

If an implementation specific layer is used, then a detailed description of how it performs its functions, how its various components interact and any product functionalities or environmental setting on which it relies must be provided. All related source code, scripts and configuration files must be disclosed. The information provided should be suficient for an independent reconstruction of the implementation specific layer.

The source code for the “qexec” Utility can be found in Appendix E.

7.3 Profiled-Directed optimization

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

Profiled-Directed optimization subject to the requirements of 5.2.9 and 5.2.10 was not used.

8. Clause 7: Pricing Related Items

8.1 Hardware and Software Used

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

The components (hardware and software) and the maintenance supplied by Bull are discounted by 25% and 35% respectively from list price, only for similar quantities and configuration and in case of a prepay cash. The software supplied by Microsoft are discounted by 15%, and $30900 on Oracle (Oracle Mandatory E-Business Discount). The detailed list of all hardware and software for the priced configuration is provided in Executive Summary at the front of this report.

8.2 Three-Year Cost of System Configuration

The total 3-year price of the entire configuration must be reported, including: hardware, software, and maintenance charges. Separate component pricing is required.

The three years support pricing for Bull S.A. consists of one year warranty (two years for disks) included in the system package price and two years support price, defined as full care in the “GlobalCare” Bull maintenance offer.
This offer is presented in Three levels of service: Bronze, Silver and Gold. The customer may opt, according to their requirements, for the level of their choice. The level of service chosen must be the same for all of the products of the configuration.

Contents of the Gold service:
The Gold service offers a high quality of service, permanent prevention and accelerated response time. It is composed of the following services:
Hardware maintenance
• The reception of 'break-down' calls 24hrs 7/7 through a number allocated by Bull or upon the automatic telephone server or being recorded on the web.
• Telephone product assistance from 8 AM to 8 PM (*) Monday to Friday except bank holidays.
• The customer is called back by an expert within 1 hour of the call being received, not included during the intervention stages.
• Remote maintenance and remote diagnostics of the system (*)
• System controlled remotely (*)
• Access to technical product data via the web.
• Breakdown intervention carried out during working days, between 8 AM and 8 PM Monday to Friday with a delay inferior to 2 hours for a subsystem stops, or inferior to 4 hours for a partly non-functioning system. This consists of the on-site repair and replacement if necessary of parts or faulty equipment broken due to ordinary usage, by new or equivalent parts or equipment. Carrying out these interventions is included in the price of the service.
• The correction of faulty products is carried out within 8 hours of receiving the initial call from the customer.
• The creation and implementation of hardware technical status applicable to Bull customers.
In choosing the Gold level of service, the equivalent level of software support is chosen automatically.
Other extended coverage options are available
(*)The 'Alarm' (Autocalls) service authorises warning messages to be sent by the server to a Bull remote maintenance centre. These alarms can be received 24h 7/7, but the reaction depends wholly upon the contractual service cover period. This means that if action is necessary following an alarm, action will only be taken during the periods stated in the maintenance contract. It is mandatory to have the material necessary to alarming and remote maintenance to benefit this functionnality.

The Gold Service is choosen for the system configuration. The pricing summary sheet on page 4 in the front of this report contains all details.

8.3 Availability Dates

The committed delivery date for general availability (availability date) of products used in the priced calculations must be reported. When the priced system includes products with diferent availability dates, the single availability date reported on the first page of the executive summary must be the date by which all components are committed to being available. The full disclosure report must report availability dates individually for at least each of the categories for which a pricing subtotal must be provided (see Clause 7.3.1.4). All availability dates, whether for individual components or for the SUT as a whole, must be disclosed to a precision of 1 day, but the precise format is left to the test sponsor.

Summary by category from the measured and priced configuration:

Category	Available
Server Hardware	Now (date of publication)
Storage	Now (date of publication)
Server Software except for Oracle Database 10g Release 2 for 64-bit Itanium-based Systems	Now (date of publication)
Oracle Database 10g Release 2 for 64-bit Itanium-based Systems	December 20, 2005

9. Clause 8: Audit Related Items

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
francois@sizing.com
1373 N. Franklin St. 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: System and Database Tunable Parameters

Software levels:

Microsoft Windows Server 2003, Datacenter Edition
Oracle Database 10g Release 2 Enterprise Edition

System Information:

Item	Value
OS Name	Microsoft(R) Windows(R) Server 2003, Datacenter Edition
Version	5.2.3790 Build 3790
OS Manufacturer	Microsoft Corporation
System Name	TPCH-16V
System Manufacturer	BULL
System Model	Novascale
System Type	Itanium (TM) -based System
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
Processor	ia64 Family 31 Model 2 Stepping 2 GenuineIntel ~1600 Mhz
BIOS Version/Date	BULL. B756.001.03/18/2005.09:21:54., 3/18/2005
SMBIOS Version	2.3
Windows Directory	C:\WINDOWS
System Directory	C:\WINDOWS\system32
Boot Device	\Device\HarddiskVolume101
Locale	United States
Hardware Abstraction Layer	Version = "5.2.3790.0 (srv03_rtm.030324-2048)"
User Name	TPCH-16V\Administrator
Time Zone	Pacific Daylight Time
Total Physical Memory	65,536.00 MB
Available Physical Memory	37.93 GB
Total Virtual Memory	146.79 GB
Available Virtual Memory	94.74 GB
Page File Space	82.80 GB
Page File	C:\pagefile.sys

Oracle Database 10g Release 2 Entreprise Edition parameters

Inittpch.ora

```
INSTANCE_TYPE= RDBMS
DB_CREATE_FILE_DEST= '+groupe1'
db_name= tpch
undo_management= auto
control_files= '+GROUPE1/tpch/controlfile/current.256.560011433'
statistics_level = BASIC
audit_trail = FALSE
compatible = 10.2.0.0
db_block_checksum = false
db_block_size = 8192
db_cache_size = 8g
aq_tm_processes = 0
db_file_multiblock_read_count = 128
db_files = 512
db_name = tpch
db_writer_processes = 4
dml_locks = 40000
global_names = FALSE
shared_pool_size = 4G
large_pool_size = 8G
log_buffer = 335544320
log_checkpoints_to_alert = true
max_dump_file_size = unlimited
timed_statistics = false
nls_date_format = YYYY-MM-DD
open_cursors = 600
optimizer_dynamic_sampling = 3
optimizer_index_cost_adj = 450
optimizer_mode = CHOOSE
optimizer_features_enable = 10.2.0.1
parallel_adaptive_multi_user = TRUE
parallel_execution_message_size = 65535
parallel_max_servers = 320
parallel_min_servers = 320
pga_aggregate_target = 32G
processes = 600

query_rewrite_enabled = true
recovery_parallelism = 8
sessions = 500
transactions = 100
undo_management = auto
undo_retention = 100000

remote_login_passwordfile = exclusive

log_checkpoint_timeout=0
BACKGROUND_DUMP_DEST= C:\oracle\download\10GR2\oh_03_03_05\rdbms\trace\bdump
CORE_DUMP_DEST= C:\oracle\download\10GR2\oh_03_03_05\rdbms\trace\cdump
USER_DUMP_DEST= C:\oracle\download\10GR2\oh_03_03_05\rdbms\trace\udump
job_queue_processes=1
```

Inittpchasm.ora

```
INSTANCE_TYPE = ASM
db_unique_name = tpchasm
ASM_DISKSTRING = 'c:\asm\disk*'
asm_diskgroups='GROUPE1' #Manual Mount
shared_pool_size = 100M
large_pool_size = 100M
processes = 100
remote_login_passwordfile=SHARED
background_dump_dest=C:\oracle\download\10GR2\oh_asm\rdbms\trace\bdump
core_dump_dest=C:\oracle\download\10GR2\oh_asm\rdbms\trace\cdump
user_dump_dest=C:\oracle\download\10GR2\oh_asm\rdbms\trace\udump
```

APPENDIX B: Program and Scripts

bumpx.pl

```
#!/usr/bin/perl
#
# $Header: bumpxlite.pl 23-oct-2002.13:15:45 mpoess Exp $
#
# bumpxlite.pl
#
# Copyright (c) 2001, 2002, Oracle Corporation.  All rights reserved.
#
#   NAME
#     bumpxlite.pl - <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      10/23/02 - mpoess_update_from_visa
#   mpoess      09/24/01 - take out readfile subroutine
#   mpoess      08/10/01 - Creation
#

$os = $ENV{'OS'};
if (($os cmp 'Windows_NT') != 0) { # os is UNIX
   $os = "unix";$nt = 0;$unix = 1;
} else {
   $os = "nt"; $nt = 1; $unix = 0;
}
$| = 1;
$verbose = 0;
if (($os cmp "unix")==0) {
   $defphases = "dbcre,sctso,scuto,dbgen,dapop,anlyz,ixcre";
} else {
   $defphases                                                    =
"sdgen,shutd,start,dbgen,plcre,dbcre,sctso,scuto,dapop,scuvo,anlyz,ixcr
e,chob";
}
$allbmtypes = "tpcd,wisc";
$bmtype = "tpcd" if !defined $bmtype;
$pdfile =  "$ENV{'BUMPX_DIR'}/param.txt";  # This file contains the
description of all possible parameters.
while ($arg = shift(@ARGV)) {
   if ($arg !~ /-(i|o|t|p|d|a|s|h)/){
      $error =  "** Error: Bad argument to $0: $arg\n";
      &usage;
   }
   if ($arg =~ /-h/) { &usage;ext(0);}
   $runsilent = 1 if ($arg =~ /-s/);
   $outfile = shift(@ARGV) if ($arg =~ /-o/);
   $bmtype = shift(@ARGV) if ($arg =~ /-t/);
   $phaselist = shift(@ARGV) if ($arg =~ /-p/);
   if ($arg =~ /-d/) {
            $defpar = shift(@ARGV);
            @keys = keys %params;
            while ($#keys >=0) {
               $key = pop(@keys);
               if (($defpar cmp "") ==0) {
                        print $key, "=", $params{$key}, "\n";
               } else {
                        print $key, "=", $params{$key}, "\n" if ($key =~
/$defpar/);
               }
            }
            exit(0);
   }
}
$outfile = "$ENV{'BUMPX_DIR'}/bumpx.dat" if !defined $outfile;
if ($nt) {
   $listdir = $filedir."list/";
   if (!-e $listfile) {
            system ("mkdir $listdir");
   }
}
if (($os cmp "nt") == 0) {          ## NT Port (Use tmpfile to buffer
   $tmpfile = "tmp.txt";       ## commands and nruntpb to synchronize
them)
   $tmpfile = $filedir.$tmpfile;
   $nruntpb =  "nruntpb.exe";
} ## NT End
if (!-e $outfile) {
   $error = "** Error: -o file, $outfile, does not exist\n";
   &usage;
}
$phaselist = $defphases if !defined $phaselist;
@phases = split(/,/, $phaselist);
## NT Port (Use tmpfile to buffer commands for nruntpb)
open (TMPFILE, ">$tmpfile") if ( (($os cmp "nt") == 0));
## NT End
&doexecute;
## NT Port
close(TMPFILE) if ( (($os cmp "nt") == 0));
## NT End
exit(0);

sub doexecute { # First, do preprocessing stuff
   print "Execution pass begun." if $verbose;
   open (INFILE, $outfile);
 WLOOP1:
   while ($line = <INFILE>)
   {
      study $line;
      next WLOOP1 if $line =~ /^\s*\#/;
      next WLOOP1 if $line =~ /^\s*\n/;
      if ($line =~ /^%b-preproc/)
      {
         $insection = 1;
         next WLOOP1;
      }
      next WLOOP1 if ($insection != 1);
      if ($line =~ /^%e-preproc/)
      {
         $insection = 0;
         $commands{$shortcmd} = $longcmd if defined $shortcmd;
         last WLOOP1;
      }
      if ($line =~ /^\*/)
      {
         $commands{$shortcmd} = $longcmd if defined $shortcmd;
         $line =~ /^(\*.*\S+)\s*\n$/;
         $shortcmd = $1;
         $longcmd = "";
         next WLOOP1;
      }
      if ($line =~ /^\\/)
      {
         #$line =~ /\\(.*\n)/;
         $line =~ /^\\(.*\n)/;
         $longcmd = $longcmd . $1;
         next WLOOP1;
      }
      print "Illegal entry in preproc stage:\n   $line";
```

```
    }
    close (INFILE);

# Then, do all of the requested phases
    $execctr = 0;
    foreach $phase (@phases)
    {
            $phase_cmd_num = 0;
            print "\n  Executing phase \'$phase\'" if $verbose;
            $bg = 0;
       open (INFILE, $outfile);
      WLOOP2:
       while ($line = <INFILE>)
       {
          study $line;
          next WLOOP2 if $line =~ /^\s*\#/;
          next WLOOP2 if $line =~ /^\s*\n/;
          if ($line =~ /^\*ignon/)
          {
             $ignon = 1;
             next WLOOP2;
          }
          if ($line =~ /^\*ignoff/)
          {
             $ignon = 0;
             next WLOOP2;
          }
          next WLOOP2 if ($ignon == 1);
          if ($line =~ /^%b-$phase/)
          {
             $insection = 1;
             $execcmd = "";
             next WLOOP2;
          }
          next WLOOP2 if ($insection != 1);
          if ($line =~ /^%e-$phase/)
          {
             $insection = 0;
             &execute ($execcmd);
             last WLOOP2;
          }
          if ($line =~ /^\*(.*)/)
          {
             &execute ($execcmd);
             if (($1 =~ /bgo/) || ($1 =~ /wait/) || ($1 =~ /ignore/))
             {
                $execcmd = $line;
                next WLOOP2;
             }
             $line =~ /^(\*.*\S+)\s*\n$/;
             $execcmd = $commands{$1};
             next WLOOP2;
          }
          if ($line =~ /^\{(.*)\}/)
          {
             $insert = "";
             $insert = $1;
             $execcmd =~ s/\{\}/$insert/;
             next WLOOP2;
          }
          if ($line =~ /^\{(.*)$/)
          {
             $insubsection = 1;
             $insert = "";
             $insert = $1;
             next WLOOP2;
          }
          if ($line =~ /^(.*)\}/)
          {
             $insubsection = 0;
             $insert = $insert . $1;
                     if (($os cmp "nt") == 0){ ## NT Port (Ignore
'\n')
                        $insert =~ /(.*)\n$/s;
                        $insert = $1;
                     } ## NT End
             $execcmd =~ s/\{\}/$insert/;
             next WLOOP2;
          }
          $insert = $insert . $line if ($insubsection == 1);
       }
       close (INFILE);
    }
    print "\nExecution pass complete.\n" if $verbose;
}

sub execute
{
    $cmd = shift(@_);
    if ($cmd)
    {
       return if ($cmd =~ /^\*ignore/);
       if ($cmd =~ /^\*bgon=(.*)/)
       {
          $bgmax = $1;
          $bg = 1;
          $bgrun = 0;
          return;
       }
       if ($cmd =~ /^\*bgoff/)
       {
          $bg = 0;
          return;
       }

            if ($cmd =~ /^\*time=(.*)/)  ##NT only
       {
                print $1 . "\n";
                print localtime(time) . "\n";
          return;
       }
       if ($cmd =~ /^copy (.*)/) ## NT only
       {
                system($cmd);
          ## Quit if failed
          if ($?) {
                        print "system copy command
failed:\n$cmd\nreason: $? ($!)\n";
                   exit(-1);
                }
          return;
       }
            if ($cmd =~ /^del (.*)/) ## NT only
       {
                system($cmd);
          ## Quit if failed
          if ($?) {
                        print "system del command
failed:\n$cmd\nreason: $? ($!)\n";
                   exit(-1);
                }
          return;
       }

            if ($cmd =~ /^\*wait/) ## This deals with main differences
between NT and UNIX
       {
                if (($os cmp "unix") == 0)
                {
                        while ($fpid = shift(@wpids))
```

```
                        {
                           waitpid($fpid, 0);
                        }
                }
                else
                {       ## NT Port (Start background tasks if any. nruntpb
will wait until all tasks are done)
                        if ($bgrun >= 1)
                        {
                           close(TMPFILE);
                           system("cat $tmpfile >> $listdir$phase.lst");
                           system("vi $tmpfile") if $debug;
                           system("$nruntpb -p < $tmpfile") if !$debug;
                           if ($?)
                           {
                                   print "system command
failed:\n$nruntpb < $tmpfile\n";
                                   print "reason: $? ($!)\n";
                                   print "Please check the contents in
the input file.\n";
                                   exit(-1);
                           }
                           open(TMPFILE, ">$tmpfile");
                        }
                }
           $bgrun = 0;
           return;
        }
        if ($cmd =~ /(s|g)etenv/)
        {
                @lines = split(/\n/, $cmd);
           $cmd = "";
           foreach $line (@lines)
           {
              while (1)
              {
                 last if ($line !~ /getenv/);
                 $line =~ /(.*)\*getenv\(([^(\)|\*)]*)\)(.*)/;
                 $line = $1 . $ENV{$2} . $3;
              }
              if ($line =~ /jojo/) #we do not want to use this for now
              {
                 $line =~ /setenv\s+(\S+)\s+(\S+)/;
                 $ENV{$1} = $2;
              }
              else
              {
                 $cmd = $cmd . $line. "\n";
              }
           }
        }
        return if ($cmd !~ /\S+/); # return if nothing left to execute
                $execctr++;
        $ENV{'BUMPX_CTR'} = $$.'-'.$execctr;
                if (($os cmp "unix") == 0)
                {
                   if ($bg == 1)
                   {
                           print "." if $verbose;
                           $fpid = fork;
                           if ($fpid == 0)
                           {
                              exec ($cmd);
                              print "exec\'d command
failed:\n$cmd\nreason: $!\n";
                              exit(-1);
                           }
                           unshift (@wpids, $fpid);
                           $bgrun = $bgrun + 1;
                           &execute ("*wait") if (($bgrun >= $bgmax) &&
($bgmax >= 0));
                   }
                   else
                   {
                           system ($cmd);
                           print "system\'d command
failed:\n$cmd\nreason: $? ($!)\n" if $?;
                   }
                }
                else ## NT support
                {
           ## NT Port (Submit background tasks if there are bgmax of them,
otherwise write to tmpfile)
                   if ($bg == 1)
                   {
                           print "." if $verbose;
                           if ($bgrun < $bgmax)
                           {
                              $cmd =~
s/phase\#.lst/$listdir$phase\_$phase_cmd_num.lst/;
                              ++$phase_cmd_num;
                              print TMPFILE $cmd;
                              $bgrun = $bgrun + 1;
                           }
                           else
                           {
                              close(TMPFILE);
                              system("cat $tmpfile >> $listdir$phase.lst");
                              system("$nruntpb -p < $tmpfile");
                              if ($?) {
                                      print "system command
failed:\n$nruntpb < $tmpfile\nreason: $? ($!)\n";
                                      print "Please check the contents in
the input file.\n";
                                      exit(-1);
                              }
                              open(TMPFILE, ">$tmpfile");
                              $cmd =~
s/phase\#.lst/$listdir$phase\_$phase_cmd_num.lst/;
                              ++$phase_cmd_num;
                              print TMPFILE $cmd;
                              $bgrun = 1;
                           }
                   }
                   else
                   {
                           $cmd =~
s/phase\#.lst/$listdir$phase\_$phase_cmd_num.lst/;
                           ++$phase_cmd_num;
                           print TMPFILE $cmd;
                           close(TMPFILE);
                           system("cat $tmpfile >> $listdir$phase.lst");
                           system ("sh $tmpfile");
                           if ($?) {
                              print "system\'d command failed:\nsh
$tmpfile\nreason: $? ($!)\n";
                              print "Please check the contents in the shell
script.\n";
                              exit(-1);
                           }
                           open(TMPFILE, ">$tmpfile");
                   }
                } ## NT support End
   }
}

sub usage
{
   print "Usage:\n";
   print "This is a lite version of bumpx.pl.  It can only be used to
```

```
execute a .dat file\n";
   print " $0 [-o outfile] [-p phaselist] [-t type]\n";
   print "   -o : intermediary file to be created and/or used\n";
   print "        defaults to bumpx.dat in \$BUMPX_DIR or \$CWD\n";
   print "   -p : list of phases to create/execute\n";
   print "        phaselist is a comma separated list of phases in order\n";
   print "        possible phases are:\n";
   print "          sdgen = seed file generation\n";
   print "          dbgen = data flat file generation\n";
   print "          plcre = NT raw partition and links creation\n";
   print "          dbcre = database creation\n";
   print "          shutd = shutdown database (on all instances)\n";
   print "          start = startup database (on all instances)\n";
   print "          sccre = schema creation\n";
   print "          sctso = schema creation (tablespaces only)\n";
   print "          scuto = schema creation (user and tables only)\n";
   print "          scuvo = schema creation (views only)\n";
   print "          dapop = data population\n";
   print "          ixcre = index creation (including constraints)\n";
   print "          anlyz = analyze objects\n";
   print "          chob  = change parameters of objects\n";
   print "          expln = create explain plans\n";
   print "          query = run and time queries\n";
   print "        defaults to $defphases\n";
   print "   -t : type of benchmark\n";
   print "        enables benchmark-specific defaults\n";
   print "        current possibilties are: $allbmtypes\n";
   print "        defaults to tpcd\n";
   print "   -s : run silent (no parameter checking is done)\n";
   print "\n";
   print "Examples:\n";
   print " $0 -p dapop\n";
   print "      Executes data population phase of intermediary file
bumpx.dat.\n";
   print "\n";
   print "$error\n";
   exit(-1);
}
```

1TB.dat

```
############################################################
##############
# preprocessing-like directives

%b-preproc

*sql
\echo "{}" > script*getenv(BUMPX_CTR).sql
\sqlplus /NOLOG <<!
\set echo on;
\set timing on;
\set termout on;
\connect sys/bull@tpch as sysdba;
\select to_char(sysdate, 'MM-DD-YYYY HH24:MI:SS') now from
dual;
\@script*getenv(BUMPX_CTR).sql;
\select to_char(sysdate, 'MM-DD-YYYY HH24:MI:SS') now from
dual;
\exit;
\!
\/bin/rm script*getenv(BUMPX_CTR).sql;


*load1
\sqlldr {}

*mknod
\mknod {}

*dbgen
\dbgen {}

*sh
\{}

%e-preproc
%b-dbcre
*bgon=1
############################################################
##################
# Undo Tablespace Creation Phase
#*sql
#{
#create  undo tablespace ts_undo2
#  datafile size 20G
#;
#}



*bgoff
%e-dbcre
%b-sctso
*bgon=300
############################################################
##################
# Schema Creation Phase - datafiles only (no tables or users)
# creating data tablespaces, datafiles
# creating tpch's ts_one tablespace
*sql
{
drop tablespace ts_default including contents;
create tablespace ts_default datafile
size 1G
extent management local
autoallocate
;
}

*wait
*sql
{
drop tablespace ts_l1 including contents;
create tablespace ts_l1 datafile
size 32000M
extent management local uniform size 32m
;
}

*wait
*sql
{
alter tablespace ts_l1 add datafile
size 32000M, size 32000M, size 32000M, size 32000M,
size 32000M, size 32000M, size 32000M, size 32000M
;
}

*sql
{
alter tablespace ts_l1 add datafile
size 32000M, size 32000M, size 32000M, size 32000M,
size 32000M, size 32000M, size 32000M, size 32000M
;
}

*sql
{
alter tablespace ts_l1 add datafile
```

```
size 32000M, size 32000M, size 32000M, size 32000M,
size 32000M, size 32000M, size 32000M, size 32000M
;
}

*sql
{
alter tablespace ts_l1 add datafile
size 32000M, size 32000M, size 32000M, size 32000M,
size 32000M, size 32000M, size 32000M
;
}


*wait
*sql
{
drop tablespace ts_o1 including contents;
create tablespace ts_o1 datafile
size 16G
extent management local uniform size 8m
;
}

*wait
*sql
{
alter tablespace ts_o1 add datafile
size 16G, size 16G, size 16G, size 16G,
size 16G, size 16G, size 16G, size 16G
;
}

*sql
{
alter tablespace ts_o1 add datafile
size 16G, size 16G, size 16G, size 16G,
size 16G, size 16G, size 16G
;
}

*wait
*sql
{
drop tablespace ts_psupp including contents;
create tablespace ts_psupp datafile
size 14g
extent management local uniform size 128m
;
}

*wait
*sql
{
alter tablespace ts_psupp add datafile
size 14G, size 14G, size 14G, size 14G,
size 14G, size 14G, size 14G, size 14G
;
}

*sql
{
alter tablespace ts_psupp add datafile
size 14G, size 14G, size 14G, size 14G,
size 14G, size 14G, size 14G
;
}


*wait
```

```
*sql
{
drop tablespace ts_rest including contents;
create tablespace ts_rest datafile
size 7G
extent management local uniform size 32m
;
}

*wait
*sql
{
alter tablespace ts_rest add datafile
size 7G, size 7G, size 7G, size 7G,
size 7G, size 7G, size 7G, size 7G
;
}

*sql
{
alter tablespace ts_rest add datafile
size 7G, size 7G, size 7G, size 7G,
size 7G, size 7G, size 7G
;
}



*wait
*sql
{
drop tablespace ts_index including contents;
create tablespace ts_index datafile
size 21G
extent management local uniform size 128m
;
}

*wait
*sql
{
alter tablespace ts_index add datafile
size 21G, size 21G, size 21G, size 21G,
size 21G, size 21G, size 21G, size 21G
;
}

*sql
{
alter tablespace ts_index add datafile
size 21G, size 21G, size 21G, size 21G,
size 21G, size 21G, size 21G
;
}



# creating tpch's ts_temp tablespace
#*wait
#*sql
#{
#drop tablespace ts_temp including contents;
#create temporary tablespace ts_temp tempfile
#size 24G
#extent management local uniform size 30m
#;
#}

*wait
*sql
```

```
{
alter tablespace ts_temp add tempfile
'+GROUPE1/tpch/tempfile/ts_temp_file02' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file03' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file04' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file05' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file06' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file07' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file08' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file09' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file10' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file11' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file12' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file13' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file14' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file15' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file16' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file17' size 24G
;
}

*sql
{
alter tablespace ts_temp add tempfile
'+GROUPE1/tpch/tempfile/ts_temp_file18' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file19' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file20' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file21' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file22' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file23' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file24' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file25' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file26' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file27' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file28' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file29' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file30' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file31' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file32' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file33' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file34' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file35' size 24G,
'+GROUPE1/tpch/tempfile/ts_temp_file36' size 24G
;
}

*wait
*sql
{
ALTER DATABASE DEFAULT TEMPORARY TABLESPACE
ts_temp;
}


*bgoff
%e-sctso
%b-dapop
*bgon=1
############################################################
####################
# Schema Creation Phase - User and Tables
#  AND  Database Population Phase
#
# creating tpch user
*sql
{
drop user tpch cascade;
grant DBA
to tpch identified by tpch;
}

*sql
{
connect tpch/tpch@tpch;
drop directory data_dir_li;
create directory data_dir_li as 'D:/';
drop directory data_dir;
create directory data_dir as 'D:/';
}
*wait
*sql
{
connect tpch/tpch@tpch;
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_dir_li
access parameters
(
                        records delimited by newline
                        nobadfile
                        nologfile
                        fields terminated by '|'
                        missing field values are null
                        (
                        l_orderkey,
                        l_partkey,
                        l_suppkey,
                        l_linenumber,
                        l_quantity,
                        l_extendedprice,
                        l_discount,
                        l_tax,
                        l_returnflag,
                        l_linestatus,
                        l_shipdate              DATE 'YYYY-MM-DD',
                        l_commitdate                    DATE
'YYYY-MM-DD',
                        l_receiptdate                   DATE
'YYYY-MM-DD',
                        l_shipinstruct,
                        l_shipmode,
                        l_comment
                        )
                )
                location (
'lineitem.tbl.1','lineitem.tbl.2','lineitem.tbl.3','lineitem.tbl.4','lineitem.tbl
.5','lineitem.tbl.6','lineitem.tbl.7','lineitem.tbl.8','lineitem.tbl.9',
'lineitem.tbl.10','lineitem.tbl.11','lineitem.tbl.12','lineitem.tbl.13','lineite
m.tbl.14','lineitem.tbl.15','lineitem.tbl.16','lineitem.tbl.17','lineitem.tbl.1
8','lineitem.tbl.19',
'lineitem.tbl.20','lineitem.tbl.21','lineitem.tbl.22','lineitem.tbl.23','lineite
```

```
m.tbl.24','lineitem.tbl.25','lineitem.tbl.26','lineitem.tbl.27','lineitem.tbl.2
8','lineitem.tbl.29',
'lineitem.tbl.30','lineitem.tbl.31','lineitem.tbl.32'
))
reject limit unlimited;
}


*wait
*sql
{
connect tpch/tpch@tpch;
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_dir
access parameters
(
                        records delimited by newline
                        nobadfile
                        nologfile
                        fields terminated by '|'
                        missing field values are null
                        (
                        o_orderkey,
                        o_custkey,
                        o_orderstatus,
                        o_totalprice,
                        o_orderdate DATE 'YYYY-MM-DD',
                        o_orderpriority,
                        o_clerk,
                        o_shippriority,
                        o_comment
                        )
            )
            location (
'orders.tbl.1','orders.tbl.2','orders.tbl.3','orders.tbl.4','orders.tbl.5','orders
.tbl.6','orders.tbl.7','orders.tbl.8','orders.tbl.9',
'orders.tbl.10','orders.tbl.11','orders.tbl.12','orders.tbl.13','orders.tbl.14','
orders.tbl.15','orders.tbl.16','orders.tbl.17','orders.tbl.18','orders.tbl.19',
'orders.tbl.20','orders.tbl.21','orders.tbl.22','orders.tbl.23','orders.tbl.24','
orders.tbl.25','orders.tbl.26','orders.tbl.27','orders.tbl.28','orders.tbl.29',
'orders.tbl.30','orders.tbl.31','orders.tbl.32'
))
reject limit unlimited;
}
*wait
*sql
{
connect tpch/tpch@tpch;
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_dir
access parameters
(
                        records delimited by newline
                        nobadfile
                        nologfile
                        fields terminated by '|'
                        missing field values are null
            )
            location (
'partsupp.tbl.1','partsupp.tbl.2','partsupp.tbl.3','partsupp.tbl.4','partsupp.t
bl.5','partsupp.tbl.6','partsupp.tbl.7','partsupp.tbl.8','partsupp.tbl.9',
'partsupp.tbl.10','partsupp.tbl.11','partsupp.tbl.12','partsupp.tbl.13','parts
upp.tbl.14','partsupp.tbl.15','partsupp.tbl.16','partsupp.tbl.17','partsupp.t
bl.18','partsupp.tbl.19',
'partsupp.tbl.20','partsupp.tbl.21','partsupp.tbl.22','partsupp.tbl.23','parts
upp.tbl.24','partsupp.tbl.25','partsupp.tbl.26','partsupp.tbl.27','partsupp.t
bl.28','partsupp.tbl.29',
'partsupp.tbl.30','partsupp.tbl.31','partsupp.tbl.32'
))
reject limit unlimited;
}
*wait
*sql
{
connect tpch/tpch@tpch;
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_dir
access parameters
(
                        records delimited by newline
                        nobadfile
                        nologfile
                        fields terminated by '|'
                        missing field values are null
            )
            location (
'part.tbl.1','part.tbl.2','part.tbl.3','part.tbl.4','part.tbl.5','part.tbl.6','part.tbl.
7','part.tbl.8','part.tbl.9',
'part.tbl.10','part.tbl.11','part.tbl.12','part.tbl.13','part.tbl.14','part.tbl.15','
part.tbl.16','part.tbl.17','part.tbl.18','part.tbl.19',
'part.tbl.20','part.tbl.21','part.tbl.22','part.tbl.23','part.tbl.24','part.tbl.25','
part.tbl.26','part.tbl.27','part.tbl.28','part.tbl.29',
'part.tbl.30','part.tbl.31','part.tbl.32'
))
reject limit unlimited;
}
*wait
*sql
{
connect tpch/tpch@tpch;
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_dir
access parameters
(
                        records delimited by newline
                        nobadfile
                        nologfile
                        fields terminated by '|'
                        missing field values are null
               )
               location (
'customer.tbl.1','customer.tbl.2','customer.tbl.3','customer.tbl.4','custom
er.tbl.5','customer.tbl.6','customer.tbl.7','customer.tbl.8','customer.tbl.9',
'customer.tbl.10','customer.tbl.11','customer.tbl.12','customer.tbl.13','cu
stomer.tbl.14','customer.tbl.15','customer.tbl.16','customer.tbl.17','custo
mer.tbl.18','customer.tbl.19',
'customer.tbl.20','customer.tbl.21','customer.tbl.22','customer.tbl.23','cu
stomer.tbl.24','customer.tbl.25','customer.tbl.26','customer.tbl.27','custo
mer.tbl.28','customer.tbl.29',
'customer.tbl.30','customer.tbl.31','customer.tbl.32'
))
reject limit unlimited;
}
*wait
*sql
{
connect tpch/tpch@tpch;
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_dir
access parameters
(
                        records delimited by newline
                        nobadfile
                        nologfile
                        fields terminated by '|'
                        missing field values are null
               )
               location (
'supplier.tbl.1','supplier.tbl.2','supplier.tbl.3','supplier.tbl.4','supplier.tbl.
5','supplier.tbl.6','supplier.tbl.7','supplier.tbl.8','supplier.tbl.9',
'supplier.tbl.10','supplier.tbl.11','supplier.tbl.12','supplier.tbl.13','supplie
r.tbl.14','supplier.tbl.15','supplier.tbl.16','supplier.tbl.17','supplier.tbl.18'
,'supplier.tbl.19',
'supplier.tbl.20','supplier.tbl.21','supplier.tbl.22','supplier.tbl.23','supplie
r.tbl.24','supplier.tbl.25','supplier.tbl.26','supplier.tbl.27','supplier.tbl.28'
,'supplier.tbl.29',
'supplier.tbl.30','supplier.tbl.31','supplier.tbl.32'
               ))
reject limit unlimited;
}
*wait
*sql
{
connect tpch/tpch@tpch;
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_dir
access parameters
(
                        records delimited by newline
                        nobadfile
                        nologfile
                        fields terminated by '|'
                        missing field values are null
               )
               location (
               'nation.tbl'))
reject limit unlimited;
}
*wait
*sql
{
connect tpch/tpch@tpch;
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_dir
access parameters
(
                        records delimited by newline
                        nobadfile
                        nologfile
                        fields terminated by '|'
                        missing field values are null
               )
               location (
               'region.tbl'))
reject limit unlimited;
}
*wait
*sql
{
connect tpch/tpch@tpch;
alter table l_et parallel;
alter table o_et parallel;
alter table ps_et parallel;
alter table p_et parallel;
alter table c_et parallel;
alter table s_et parallel;
}
# altering tpch's default and temporary tablespace
*sql
{
alter user tpch default tablespace ts_default;
alter user tpch temporary tablespace ts_temp;
}
*wait
*sql
{
connect tpch/tpch@tpch
```

```
@?/rdbms/admin/utlxplan.sql;
}

*wait
*sql
{
set timing on
set echo on
!date
connect tpch/tpch@tpch;

drop table lineitem purge;
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 (freelists 99)
compress
parallel
nologging
partition by range (l_shipdate)
subpartition by hash(l_partkey)
subpartitions 32
(
partition item1 values less than (to_date('1992-01-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item2 values less than (to_date('1992-02-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item3 values less than (to_date('1992-03-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item4 values less than (to_date('1992-04-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item5 values less than (to_date('1992-05-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item6 values less than (to_date('1992-06-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item7 values less than (to_date('1992-07-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item8 values less than (to_date('1992-08-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item9 values less than (to_date('1992-09-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item10 values less than (to_date('1992-10-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item11 values less than (to_date('1992-11-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item12 values less than (to_date('1992-12-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item13 values less than (to_date('1993-01-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item14 values less than (to_date('1993-02-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item15 values less than (to_date('1993-03-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item16 values less than (to_date('1993-04-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item17 values less than (to_date('1993-05-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item18 values less than (to_date('1993-06-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item19 values less than (to_date('1993-07-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item20 values less than (to_date('1993-08-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item21 values less than (to_date('1993-09-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item22 values less than (to_date('1993-10-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item23 values less than (to_date('1993-11-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item24 values less than (to_date('1993-12-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item25 values less than (to_date('1994-01-01','YYYY-MM-DD'))
store in (ts_l1)
```

```
,
partition item26 values less than (to_date('1994-02-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item27 values less than (to_date('1994-03-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item28 values less than (to_date('1994-04-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item29 values less than (to_date('1994-05-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item30 values less than (to_date('1994-06-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item31 values less than (to_date('1994-07-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item32 values less than (to_date('1994-08-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item33 values less than (to_date('1994-09-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item34 values less than (to_date('1994-10-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item35 values less than (to_date('1994-11-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item36 values less than (to_date('1994-12-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item37 values less than (to_date('1995-01-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item38 values less than (to_date('1995-02-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item39 values less than (to_date('1995-03-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item40 values less than (to_date('1995-04-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item41 values less than (to_date('1995-05-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item42 values less than (to_date('1995-06-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item43 values less than (to_date('1995-07-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item44 values less than (to_date('1995-08-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item45 values less than (to_date('1995-09-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item46 values less than (to_date('1995-10-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item47 values less than (to_date('1995-11-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item48 values less than (to_date('1995-12-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item49 values less than (to_date('1996-01-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item50 values less than (to_date('1996-02-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item51 values less than (to_date('1996-03-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item52 values less than (to_date('1996-04-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item53 values less than (to_date('1996-05-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item54 values less than (to_date('1996-06-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item55 values less than (to_date('1996-07-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item56 values less than (to_date('1996-08-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item57 values less than (to_date('1996-09-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item58 values less than (to_date('1996-10-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item59 values less than (to_date('1996-11-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item60 values less than (to_date('1996-12-01','YYYY-MM-DD'))
store in (ts_l1)
```

```
,
partition item61 values less than (to_date('1997-01-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item62 values less than (to_date('1997-02-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item63 values less than (to_date('1997-03-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item64 values less than (to_date('1997-04-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item65 values less than (to_date('1997-05-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item66 values less than (to_date('1997-06-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item67 values less than (to_date('1997-07-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item68 values less than (to_date('1997-08-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item69 values less than (to_date('1997-09-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item70 values less than (to_date('1997-10-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item71 values less than (to_date('1997-11-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item72 values less than (to_date('1997-12-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item73 values less than (to_date('1998-01-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item74 values less than (to_date('1998-02-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item75 values less than (to_date('1998-03-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item76 values less than (to_date('1998-04-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item77 values less than (to_date('1998-05-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item78 values less than (to_date('1998-06-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item79 values less than (to_date('1998-07-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item80 values less than (to_date('1998-08-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item81 values less than (to_date('1998-09-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item82 values less than (to_date('1998-10-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item83 values less than (to_date('1998-11-01','YYYY-MM-DD'))
store in (ts_l1)
,
partition item84 values less than (MAXVALUE)
store in (ts_l1)
)
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;
!date
}

*wait
*sql
{
connect tpch/tpch@tpch;
set timing on
set echo on
!date

drop table orders purge;
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 (freelists 99)
compress
parallel
nologging
partition by range (o_orderdate)
subpartition by hash(o_custkey)
subpartitions 32
(
partition ord1 values less than (to_date('1992-01-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord2 values less than (to_date('1992-02-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord3 values less than (to_date('1992-03-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord4 values less than (to_date('1992-04-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord5 values less than (to_date('1992-05-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord6 values less than (to_date('1992-06-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord7 values less than (to_date('1992-07-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord8 values less than (to_date('1992-08-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord9 values less than (to_date('1992-09-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord10 values less than (to_date('1992-10-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord11 values less than (to_date('1992-11-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord12 values less than (to_date('1992-12-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord13 values less than (to_date('1993-01-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord14 values less than (to_date('1993-02-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord15 values less than (to_date('1993-03-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord16 values less than (to_date('1993-04-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord17 values less than (to_date('1993-05-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord18 values less than (to_date('1993-06-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord19 values less than (to_date('1993-07-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord20 values less than (to_date('1993-08-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord21 values less than (to_date('1993-09-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord22 values less than (to_date('1993-10-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord23 values less than (to_date('1993-11-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord24 values less than (to_date('1993-12-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord25 values less than (to_date('1994-01-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord26 values less than (to_date('1994-02-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord27 values less than (to_date('1994-03-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord28 values less than (to_date('1994-04-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord29 values less than (to_date('1994-05-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord30 values less than (to_date('1994-06-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord31 values less than (to_date('1994-07-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord32 values less than (to_date('1994-08-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord33 values less than (to_date('1994-09-01','YYYY-MM-DD'))
store in (ts_o1)
,
```

```
partition ord34 values less than (to_date('1994-10-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord35 values less than (to_date('1994-11-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord36 values less than (to_date('1994-12-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord37 values less than (to_date('1995-01-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord38 values less than (to_date('1995-02-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord39 values less than (to_date('1995-03-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord40 values less than (to_date('1995-04-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord41 values less than (to_date('1995-05-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord42 values less than (to_date('1995-06-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord43 values less than (to_date('1995-07-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord44 values less than (to_date('1995-08-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord45 values less than (to_date('1995-09-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord46 values less than (to_date('1995-10-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord47 values less than (to_date('1995-11-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord48 values less than (to_date('1995-12-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord49 values less than (to_date('1996-01-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord50 values less than (to_date('1996-02-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord51 values less than (to_date('1996-03-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord52 values less than (to_date('1996-04-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord53 values less than (to_date('1996-05-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord54 values less than (to_date('1996-06-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord55 values less than (to_date('1996-07-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord56 values less than (to_date('1996-08-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord57 values less than (to_date('1996-09-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord58 values less than (to_date('1996-10-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord59 values less than (to_date('1996-11-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord60 values less than (to_date('1996-12-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord61 values less than (to_date('1997-01-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord62 values less than (to_date('1997-02-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord63 values less than (to_date('1997-03-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord64 values less than (to_date('1997-04-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord65 values less than (to_date('1997-05-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord66 values less than (to_date('1997-06-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord67 values less than (to_date('1997-07-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord68 values less than (to_date('1997-08-01','YYYY-MM-DD'))
store in (ts_o1)
,
```

```
partition ord69 values less than (to_date('1997-09-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord70 values less than (to_date('1997-10-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord71 values less than (to_date('1997-11-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord72 values less than (to_date('1997-12-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord73 values less than (to_date('1998-01-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord74 values less than (to_date('1998-02-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord75 values less than (to_date('1998-03-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord76 values less than (to_date('1998-04-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord77 values less than (to_date('1998-05-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord78 values less than (to_date('1998-06-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord79 values less than (to_date('1998-07-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord80 values less than (to_date('1998-08-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord81 values less than (to_date('1998-09-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord82 values less than (to_date('1998-10-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord83 values less than (to_date('1998-11-01','YYYY-MM-DD'))
store in (ts_o1)
,
partition ord84 values less than (MAXVALUE)
store in (ts_o1)
)
as select
   o_orderdate        ,
   o_orderkey         ,
   o_custkey          ,
   o_orderpriority    ,
   o_shippriority     ,
   o_clerk            ,
   o_orderstatus      ,
   o_totalprice       ,
   o_comment
from o_et;
!date
}

*wait
*sql
{
connect tpch/tpch@tpch
set timing on
set echo on

!date

drop table partsupp purge;
create table partsupp(
   ps_partkey          NOT NULL,
   ps_suppkey          NOT NULL,
   ps_supplycost       NOT NULL,
   ps_availqty        ,
   ps_comment
)
partition by hash(ps_partkey)
partitions 32
compress
parallel
nologging
tablespace ts_psupp
as select
   ps_partkey         ,
   ps_suppkey         ,
   ps_supplycost      ,
   ps_availqty        ,
   ps_comment
from ps_et;
!date
}

*wait
*sql
{
connect tpch/tpch@tpch
set timing on
set echo on

!date
drop table customer purge;
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
compress
parallel
nologging
partition by hash (c_custkey)
partitions 32
tablespace ts_rest
as select
   c_custkey          ,
   c_mktsegment       ,
```

```
   c_nationkey          ,
   c_name               ,
   c_address            ,
   c_phone              ,
   c_acctbal            ,
   c_comment
from c_et;
!date
}

*wait
*sql
{
connect tpch/tpch@tpch
set timing on
set echo on

!date
drop table part purge;

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
compress
parallel
nologging
partition by hash (p_partkey)
partitions 32
tablespace ts_rest
as select
   p_partkey           ,
   p_type              ,
   p_size              ,
   p_brand             ,
   p_name              ,
   p_container         ,
   p_mfgr              ,
   p_retailprice       ,
   p_comment
from p_et;
!date
}

*wait
*sql
{
connect tpch/tpch@tpch;
set timing on
set echo on

drop table supplier purge;
create table supplier(
   s_suppkey           NOT NULL,
   s_nationkey         ,
   s_comment           ,
   s_name              ,
   s_address           ,
   s_phone             ,
   s_acctbal
)
pctfree 0
pctused 99
compress
parallel
nologging
partition by hash (s_suppkey)
partitions 32
tablespace ts_rest
as select
   s_suppkey           ,
   s_nationkey         ,
   s_comment           ,
   s_name              ,
   s_address           ,
   s_phone             ,
   s_acctbal
from s_et;
}

*wait
*sql
{
connect tpch/tpch@tpch;
set echo on
set timing on

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

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


*bgoff
%e-scuto

*wait
*sql
{
connect tpch/tpch@tpch
set timing on
set echo on

!date
drop table l_et purge;
drop table o_et purge;
drop table ps_et purge;
drop table p_et purge;
drop table c_et purge;
drop table s_et purge;
drop table n_et purge;
drop table r_et purge;
}
*bgoff
%e-dapop
%b-ixcre
*bgon=1
#########################################################
```

```
####################
# Index Creation Phase
*sql
{
connect tpch/tpch@tpch;
!date
set echo on
set timing on
drop index i_l_orderkey;
create index i_l_orderkey
on lineitem (l_orderkey)
pctfree 5
initrans 10
tablespace ts_index
storage (freelist groups 4 freelists 84)
parallel
compute statistics
nologging;
alter index  i_l_orderkey   allocate extent;
alter index  i_l_orderkey   allocate extent;
alter index  i_l_orderkey   allocate extent;
}
*wait
*sql
{
connect tpch/tpch@tpch;
!date
set echo on
set timing on
drop index i_o_orderkey;
create unique index i_o_orderkey
on orders (o_orderkey)
pctfree 5
initrans 10
tablespace ts_index
storage (freelist groups 4 freelists 84 )
parallel
compute statistics
nologging;
alter index  i_o_orderkey   allocate extent;
alter index  i_o_orderkey   allocate extent;
alter index  i_o_orderkey   allocate extent;
}
*wait
*sql
{
connect tpch/tpch@tpch;
!date
set echo on
set timing on

drop index i_c_custkey;
create unique index i_c_custkey
on customer (c_custkey)
pctfree 0
initrans 10
tablespace ts_index
storage (freelists 84)
parallel
compute statistics
nologging;
}
*wait
*sql
{
connect tpch/tpch@tpch;
!date
set echo on
set timing on

drop index i_ps_partkey_suppkey;
create unique index i_ps_partkey_suppkey
      on partsupp(ps_partkey, ps_suppkey)
global partition by hash(ps_partkey)
partitions 32
pctfree 5
initrans 10
tablespace ts_index
storage (freelists 84)
parallel
nologging
compute statistics;
}

*bgoff
%e-ixcre
%b-anlyz
*bgon=1
############################################################
####################
# Analyze Phase
*wait
*sql
{
connect tpch/tpch@tpch;
!date
set timing on
execute dbms_stats.gather_schema_stats('TPCH' , estimate_percent =>
1, degree => 32 , granularity => 'GLOBAL' );
connect sys/bull@tpch as sysdba
execute dbms_stats.gather_system_stats;
execute dbms_scheduler.disable('GATHER_STATS_JOB');
execute dbms_scheduler.disable('AUTO_SPACE_ADVISOR_JOB');
execute dbms_scheduler.disable('AUTO_TASKS_JOB_CLASS');
alter system switch logfile;
!date
}

*bgoff
%e-anlyz
```

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
ATOM_COMMIT_CHK=$ACID_OUT/atom_commit_chk

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 > ${OUT_DIR}/rankey_commit.out

KEYS=`head -10 ${OUT_DIR}/rankey_commit.out | awk '{printf "%d ", $1}'`
echo "The keys to check for atomicity BEFORE the test"
echo "$KEYS"
for j in $KEYS
do
   sqlplus $USER @chk_atom $j >> $ATOM_COMMIT_CHK
   echo "------------------------------" >> $ATOM_COMMIT_CHK
done

$PROG 1 1 1 0 u$USER i${OUT_DIR}/rankey_commit.out > ${OUT}c 2>&1

echo "The keys to check for atomicity AFERT the test"
echo "$KEYS"
for j in $KEYS
do
   sqlplus $USER @chk_atom $j >> $ATOM_COMMIT_CHK
   echo "------------------------------" >> $ATOM_COMMIT_CHK
done

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 > ${OUT_DIR}/rankey_rollback.out
KEYS=`head -10 ${OUT_DIR}/rankey_rollback.out | awk '{printf "%d ", $1}'`
echo "The keys to check for atomicity BEFORE rollback the test"
echo "$KEYS"
for j in $KEYS
do
   sqlplus $USER @chk_atom $j >> $ATOM_COMMIT_CHK
   echo "------------------------------" >> $ATOM_COMMIT_CHK
done

$PROG 1 1 0 0 u$USER i${OUT_DIR}/rankey_rollback.out > ${OUT}r 2>&1
echo "The keys to check for atomicity AFERT rollback the test"
echo "$KEYS"
for j in $KEYS
do
   sqlplus $USER @chk_atom $j >> $ATOM_COMMIT_CHK
   echo "------------------------------" >> $ATOM_COMMIT_CHK
done

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

atranspl.c

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

/*

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

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

  NOTES
    <other useful comments, qualifications, etc.>

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

*/

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

#include "atranspl.h"

/* Declare error handling functions */

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@dbname combo
*/
char *passwd;           /* pointer to password             */
char *dbname;                                    /* pointer to dbname
                                                 */

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 and dbname*/

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

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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcsrv,0,OCI_ATTR_SERVE
R,errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,lname,strlen(lname),OCI_AT
TR_USERNAME,
        errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(passwd),OCI_
ATTR_PASSWORD,
        errhp);

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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcusr,0,OCI_ATTR_SESSI
ON,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),S
QLT_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),SQ
LT_INT);

OCIbbname(curr,o_key_bp,errhp,":o_key",ADR(o_key),SIZ(o_key),S
QLT_INT);

OCIbbname(curr,delta_bp,errhp,":delta",ADR(delta),SIZ(delta),SQLT
_INT);

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),
SQLT_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),S
QLT_FLT);

OCIbbname(curr,l_eprice_bp,errhp,":l_eprice",ADR(l_eprice),SIZ(l_e
price),
        SQLT_FLT);

OCIbbname(curr,l_neweprice_bp,errhp,":l_neweprice",ADR(l_newepr
ice),
        SIZ(l_neweprice),SQLT_FLT);


OCIbbname(curr,o_tprice_bp,errhp,":o_tprice",ADR(o_tprice),SIZ(o_t
price),
        SQLT_FLT);

OCIbbname(curr,o_newtprice_bp,errhp,":o_newtprice",ADR(o_newtpr
ice),
        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_new
eprice),
        SIZ(l_neweprice),SQLT_FLT);

OCIbbname(cure1,l_newquan1_bp,errhp,":l_newquan",ADR(l_newqu
an),
        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_new
tprice),
        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_D
EFAULT)) != 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
2)"
#define PDDLTXT "alter session force parallel ddl parallel (degree 2)"
#define OICATXT "alter session set optimizer_index_cost_adj=25"

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

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

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

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

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

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

#endif /* ATRANSPL_H */
```

ckpt.sh

```
#!/bin/ksh
#
# $Header: ckpt.sh 08-aug-99.17:37:07 mpoess Exp $
#
# ckpt.sh
#
#  Copyright (c) Oracle Corporation 1999. All Rights Reserved.
#
#   NAME
#     ckpt.sh - <one-line expansion of the name>
#
#  DESCRIPTION
#     Usage: ckpt.sh
#     Start database checkpoint
#
#   NOTES
#     <other useful comments, qualifications, etc.>
#
#   MODIFIED   (MM/DD/YY)
#   mpoess      08/08/99 - Creation
#   mpoess      08/08/99 - Creation
#

. $KIT_DIR/env

sqlplus -s /NOLOG << !

              connect sys/bull@tpch 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 @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

get_alert_log.sh

cat alert_tpchtest.log >> $CHK

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

consist.sql

```
set verify off
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
```

dura.sh

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

. $KIT_DIR/env

# Create history table

# Count number of entries in the history table

SERVER="ultraperf2"

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

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

sleep 1200

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

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

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

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

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

sleep 120

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


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

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

sleep 120

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

# Power Off
```

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}_
TRIG=1
HCNT=duracnta

# get history count

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

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

gettime.c

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

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

/*
```

```
  NAME
    gettime.c

  DESCRIPTION
    get wall clock time.
    get cpu time.

  FUNCTIONS
    get wall clock time.
    get cpu time.

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

*/


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

#define SUN_OS5

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

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

#if defined(aix) || defined(AIXRIOS)
# define TIME_W_GETTIME
# define CPU_W_TIMES
# define GETRU_STATS
#endif /* AIXRIOS */

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

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

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

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

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

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

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

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

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

# include <stdio.h>

#ifdef GETRU_STATS
struct rusage selfru;
struct rusage kidsru;
#endif /* GETRU_STATS */

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



double gettime ()

{

#ifdef TIME_W_GETTIME
  struct timeval tv;

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

#ifdef TIME_W_TIMES
  struct tms buf;

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

}



double getcpu ()

{

#ifdef CPU_W_TIMES
  struct tms buf;
```

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

#ifdef CPU_W_GETRU
  struct rusage ru;
  double usecs;

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

}



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

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

{

#ifdef GETRU_STATS
  struct rusage ru;

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

#ifdef GET_P_STATS
  timeval_t tv;
  struct process_stats ru;

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

}



getru1 (kids)

int kids;

{

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

#ifdef GET_P_STATS
  timeval_t tv;

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

}



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

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

{

#ifdef GETRU_STATS
  struct rusage ru;

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

#ifdef GET_P_STATS
  timeval_t tv;
  struct process_stats ru;

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

}



#ifdef GETRU_STATS
```

```
print_ru (fp, ru)

FILE *fp;
struct rusage *ru;

{

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

}



diffru (ru2, ru)

struct rusage *ru2;
struct rusage *ru;

{

  ru2->ru_utime.tv_sec -= ru->ru_utime.tv_sec;
  ru2->ru_utime.tv_usec -= ru->ru_utime.tv_usec;
  ru2->ru_stime.tv_sec -= ru->ru_stime.tv_sec;
  ru2->ru_stime.tv_usec -= ru->ru_stime.tv_usec;
  ru2->ru_maxrss -= ru->ru_maxrss;
  ru2->ru_ixrss -= ru->ru_ixrss;
  ru2->ru_idrss -= ru->ru_idrss;
  ru2->ru_minflt -= ru->ru_minflt;
  ru2->ru_majflt -= ru->ru_majflt;
  ru2->ru_nswap -= ru->ru_nswap;
  ru2->ru_inblock -= ru->ru_inblock;
  ru2->ru_oublock -= ru->ru_oublock;
  ru2->ru_msgsnd -= ru->ru_msgsnd;
  ru2->ru_msgrcv -= ru->ru_msgrcv;
  ru2->ru_nsignals -= ru->ru_nsignals;
  ru2->ru_nvcsw -= ru->ru_nvcsw;
  ru2->ru_nivcsw -= ru->ru_nivcsw;

}

#endif /* GETRU_STATS */



#ifdef GET_P_STATS

print_ru (fp, ps)

FILE *fp;
struct process_stats *ps;

{

  fprintf (fp, "%lu ", ps->ps_utime.tv_sec * 1000 +
                (ps->ps_utime.tv_usec/1000));
  fprintf (fp, "%lu ", ps->ps_stime.tv_sec * 1000 +
                 (ps->ps_stime.tv_usec/1000));
  fprintf (fp, "%lu ", ps->ps_maxrss);
  fprintf (fp, "%lu ", ps->ps_pagein);
  fprintf (fp, "%lu ", ps->ps_reclaim);
  fprintf (fp, "%lu ", ps->ps_zerofill);
  fprintf (fp, "%lu ", ps->ps_pffincr);
  fprintf (fp, "%lu ", ps->ps_pffdecr);
  fprintf (fp, "%lu ", ps->ps_swap);
  fprintf (fp, "%lu ", ps->ps_syscall);
  fprintf (fp, "%lu ", ps->ps_volcsw);
  fprintf (fp, "%lu ", ps->ps_involcsw);
  fprintf (fp, "%lu ", ps->ps_signal);
  fprintf (fp, "%lu ", ps->ps_lread);
  fprintf (fp, "%lu ", ps->ps_lwrite);
  fprintf (fp, "%lu ", ps->ps_bread);
  fprintf (fp, "%lu ", ps->ps_bwrite);
  fprintf (fp, "%lu ", ps->ps_phread);
  fprintf (fp, "%lu",  ps->ps_phwrite);

}



diffru (ru2, ru)

struct process_stats *ru2;
struct process_stats *ru;

{

  ru2->ps_utime.tv_sec -= ru->ps_utime.tv_sec;
  ru2->ps_utime.tv_usec -= ru->ps_utime.tv_usec;
  ru2->ps_stime.tv_sec -= ru->ps_stime.tv_sec;
  ru2->ps_stime.tv_usec -= ru->ps_stime.tv_usec;
  ru2->ps_maxrss -= ru->ps_maxrss;
  ru2->ps_pagein -= ru->ps_pagein;
  ru2->ps_reclaim -= ru->ps_reclaim;
  ru2->ps_zerofill -= ru->ps_zerofill;
  ru2->ps_pffincr -= ru->ps_pffincr;
  ru2->ps_pffdecr -= ru->ps_pffdecr;
  ru2->ps_swap -= ru->ps_swap;
  ru2->ps_syscall -= ru->ps_syscall;
  ru2->ps_volcsw -= ru->ps_volcsw;
  ru2->ps_involcsw -= ru->ps_involcsw;
  ru2->ps_signal -= ru->ps_signal;
  ru2->ps_lread -= ru->ps_lread;
  ru2->ps_lwrite -= ru->ps_lwrite;
  ru2->ps_bread -= ru->ps_bread;
  ru2->ps_bwrite -= ru->ps_bwrite;
  ru2->ps_phread -= ru->ps_phread;
  ru2->ps_phwrite -= ru->ps_phwrite;

}

#endif /* GET_P_STATS */
```

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

HOST=

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 15

# start ACID query with the same OKEY

echo "Running ACID query 15 seconds AFTER the start of ACID
Transaction" \
>> $TXN2FILE
echo "`date`" >> $TXN2FILE
if [ "$HOST" != "" ]
then
echo "Starting ACID query on node $HOST" >> $TXN2FILE
echo ${HOST}
${RSH} -n ${HOST} sqlplus $USER
@$ACID_DIR/isolation/a_query $OKEY >> $TXN2FILE
else
sqlplus $USER @$ACID_DIR/isolation/a_query $OKEY >>
$TXN2FILE
fi
echo "END TXN2----------" >>
$TXN2FILE
wait
echo "END TXN1----------" >>
$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
HOST=

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 15 seconds before starting ACID query

sleep 15

# start ACID query with the same OKEY

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

echo "END TXN2------------------------------------------------" >>
$TXN2FILE
wait
echo "END TXN1------------------------------------------------" >>
$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
HOST=

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
else
echo continue
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 15 seconds before starting second ACID transaction

sleep 15

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

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

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

```
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 15 seconds before starting second ACID transaction

sleep 15

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

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

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

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

iso5.sh

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

. $KIT_DIR/env

# May need to change the following:
RSH=rsh
HOST=

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

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
HOST=

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

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

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

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

sleep 1

# start Query 1, use 0 as the delta

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

# sleep 2 seconds before starting ACID transaction

sleep 2

# start ACID transaction, COMMIT after one second

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

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

# start Query 17

sleep 2

echo "Running 2nd Query 17b at `date`" >> $TXN3FILE
sqlplus $USER @q1 >> $TXN3FILE &

# wait for everyone to finish

wait

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

cat $TXN1FILE $TXN2FILE $TXN3FILE >> $ISOFILE

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

randkey.c

```
/* 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;
char *dbname;

/* 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, "tpch/tpch@tpchtest");

 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@basename>\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 and dbname*/

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


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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcsrv,0,OCI_ATTR_SERVE
R,errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,lname,strlen(lname),OCI_AT
TR_USERNAME,
        errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(passwd),OCI_
ATTR_PASSWORD,
        errhp);

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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcusr,0,OCI_ATTR_SESSI
ON,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),SQ
LT_INT);

OCIbbname(curi,o_key_bp,errhp,":o_key",ADR(o_key),SIZ(o_key),S
QLT_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");
}
```

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 $DATABASE_USER @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.>
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    mpoess      08/08/99 - Creation
Rem    mpoess      08/08/99 - Created
Rem

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

exit;
```

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
  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);

    EXIT;

  EXCEPTION
    WHEN not_serializable THEN
      ROLLBACK;
  END;

  END LOOP;

END doatrans;
END;
/
exit;
```

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
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

echo "Starting ACID run..."

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

# Get history count before the run

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

while [ $i -lt $STEM ]
do
  randkey $ITER ${SF} u${DATABASE_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 $DATABASE_USER @consist $j >>
$DURA_DIR/duraconsb
  done
  i=`expr $i + 1`
done

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

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

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

done

wait

echo "ACID run completed"
```

Disk Configuration Details

Scipt to create 1TB ASM group

```
create diskgroup groupe1 NORMAL REDUNDANCY
FAILGROUP ss_grp1 DISK
-- cab 1a
'c:/asm/disk52',
'c:/asm/disk53',
'c:/asm/disk54',
'c:/asm/disk55',
'c:/asm/disk56',
'c:/asm/disk57',
-- cab 1b
'c:/asm/disk36',
'c:/asm/disk37',
'c:/asm/disk38',
'c:/asm/disk39',
'c:/asm/disk40',
'c:/asm/disk41',
-- cab 3a
'c:/asm/disk84',
```

```
'c:/asm/disk85',
'c:/asm/disk86',
'c:/asm/disk87',
'c:/asm/disk88',
'c:/asm/disk89',
-- cab 3b
'c:/asm/disk68',
'c:/asm/disk69',
'c:/asm/disk70',
'c:/asm/disk71',
'c:/asm/disk72',
'c:/asm/disk73',
-- cab 5a
'c:/asm/disk132',
'c:/asm/disk133',
'c:/asm/disk134',
'c:/asm/disk135',
'c:/asm/disk136',
'c:/asm/disk137',
-- cab 5b
'c:/asm/disk148',
'c:/asm/disk149',
'c:/asm/disk150',
'c:/asm/disk151',
'c:/asm/disk152',
'c:/asm/disk153',
-- cab 7a
'c:/asm/disk196',
'c:/asm/disk197',
'c:/asm/disk198',
'c:/asm/disk199',
'c:/asm/disk200',
'c:/asm/disk201',
-- cab 7b
'c:/asm/disk210',
'c:/asm/disk211',
'c:/asm/disk212',
'c:/asm/disk213',
'c:/asm/disk214',
'c:/asm/disk215',
-- cab 9a
'c:/asm/disk234',
'c:/asm/disk235',
'c:/asm/disk236',
'c:/asm/disk237',
'c:/asm/disk238',
'c:/asm/disk239',
-- cab9b
'c:/asm/disk226',
'c:/asm/disk227',
'c:/asm/disk228',
'c:/asm/disk229',
'c:/asm/disk230',
'c:/asm/disk231',
-- cab 11a
'c:/asm/disk140',
'c:/asm/disk141',
'c:/asm/disk142',
'c:/asm/disk143',
'c:/asm/disk144',
'c:/asm/disk145',
-- cab 11b
'c:/asm/disk156',
'c:/asm/disk157',
'c:/asm/disk158',
'c:/asm/disk159',
'c:/asm/disk160',
'c:/asm/disk161',
-- cab13a
'c:/asm/disk92',
'c:/asm/disk93',
'c:/asm/disk94',
'c:/asm/disk95',
'c:/asm/disk96',
'c:/asm/disk97',
-- cab13b
'c:/asm/disk76',
'c:/asm/disk77',
'c:/asm/disk78',
'c:/asm/disk79',
'c:/asm/disk80',
'c:/asm/disk81',
-- cab 15a
'c:/asm/disk28',
'c:/asm/disk29',
'c:/asm/disk30',
'c:/asm/disk31',
'c:/asm/disk32',
'c:/asm/disk33'
FAILGROUP ss_grp2 DISK
-- cab 2a
'c:/asm/disk20',
'c:/asm/disk21',
'c:/asm/disk22',
'c:/asm/disk23',
'c:/asm/disk24',
'c:/asm/disk25',
-- cab 2b
'c:/asm/disk4',
'c:/asm/disk5',
'c:/asm/disk6',
'c:/asm/disk7',
'c:/asm/disk8',
'c:/asm/disk9',
-- cab 4a
'c:/asm/disk116',
'c:/asm/disk117',
'c:/asm/disk118',
'c:/asm/disk119',
'c:/asm/disk120',
'c:/asm/disk121',
-- cab 4b
'c:/asm/disk100',
'c:/asm/disk101',
'c:/asm/disk102',
'c:/asm/disk103',
'c:/asm/disk104',
'c:/asm/disk105',
-- cab 6a
'c:/asm/disk164',
'c:/asm/disk165',
'c:/asm/disk166',
'c:/asm/disk167',
'c:/asm/disk168',
'c:/asm/disk169',
-- cab 6b
'c:/asm/disk180',
'c:/asm/disk181',
'c:/asm/disk182',
'c:/asm/disk183',
'c:/asm/disk184',
'c:/asm/disk185',
-- cab 8a
'c:/asm/disk203',
'c:/asm/disk204',
'c:/asm/disk205',
'c:/asm/disk206',
'c:/asm/disk207',
'c:/asm/disk208',
-- cab 8b
```

```
'c:/asm/disk219',
'c:/asm/disk220',
'c:/asm/disk221',
'c:/asm/disk222',
'c:/asm/disk223',
'c:/asm/disk224',
-- cab 10a
'c:/asm/disk172',
'c:/asm/disk173',
'c:/asm/disk174',
'c:/asm/disk175',
'c:/asm/disk176',
'c:/asm/disk177',
-- cab 10b
'c:/asm/disk188',
'c:/asm/disk189',
'c:/asm/disk190',
'c:/asm/disk191',
'c:/asm/disk192',
'c:/asm/disk193',
-- cab12a
'c:/asm/disk108',
'c:/asm/disk109',
'c:/asm/disk110',
'c:/asm/disk111',
'c:/asm/disk112',
'c:/asm/disk113',
-- cab 12b
'c:/asm/disk124',
'c:/asm/disk125',
'c:/asm/disk126',
'c:/asm/disk127',
'c:/asm/disk128',
'c:/asm/disk129',
-- cab 14a
'c:/asm/disk60',
'c:/asm/disk61',
'c:/asm/disk62',
'c:/asm/disk63',
'c:/asm/disk64',
'c:/asm/disk65',
-- cab 14b
'c:/asm/disk44',
'c:/asm/disk45',
'c:/asm/disk46',
'c:/asm/disk47',
'c:/asm/disk48',
'c:/asm/disk49',
-- cab 15b
'c:/asm/disk12',
'c:/asm/disk13',
'c:/asm/disk14',
'c:/asm/disk15',
'c:/asm/disk16',
'c:/asm/disk17'
;
```

Scipt to create ACID ASM group

```
create diskgroup groupe2 NORMAL REDUNDANCY
FAILGROUP ss_grp1 DISK
'c:/asm/disk209',
'c:/asm/disk217'
FAILGROUP ss_grp2 DISK
'c:/asm/disk225',
'c:/asm/disk233'
;
```

APPENDIX C: Query Text

```
-- @(#)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 4.27 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

……

7871.50 Supplier#000007206 RUSSIA
104695.00 Manufacturer#1
3w fNCnrVmvJjE95sgWZzvW 32-432-452-7731
furiously dogged pinto beans cajole. bold, express notornis until the slyly pending
7852.45 Supplier#000005864 RUSSIA
8363.00 Manufacturer#4
WCNfBPZeSXh3h,c 32-454-883-3821
blithely regular deposits
7850.66 Supplier#000001518 UNITED KINGDOM
86501.00 Manufacturer#1
ONda3YJiHKJOC 33-730-383-3892
furiously final accounts wake carefully idle requests. even dolphins wake acc
7843.52 Supplier#000006683 FRANCE
11680.00 Manufacturer#4
2Z0JGkiv01Y00oCFwUGfviIbhzCdy 16-464-517-8943
carefully bold accounts doub

100 rows processed.
Query Processed in 5.08 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	05-MAR-95	0.00
3459808.00	405838.70	04-MAR-95	0.00
492164.00	390324.06	19-FEB-95	0.00
1188320.00	384537.94	09-MAR-95	0.00
2435712.00	378673.06	26-FEB-95	0.00
4878020.00	378376.80	12-MAR-95	0.00
5521732.00	375153.92	13-MAR-95	0.00
2628192.00	373133.31	22-FEB-95	0.00
993600.00	371407.46	05-MAR-95	0.00
2300070.00	367371.15	13-MAR-95	0.00

10 rows processed.
Query Processed in 3.24 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 4.23 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 6.69 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 0.77 seconds.

-- @(#)7.sql    2.1.6.2
-- TPC-H/TPC-R Volume Shipping Query (Q7)
-- Functional Query Definition
-- Approved February 1998

select
supp_nation,
cust_nation,
l_year,
sum(volume) as revenue
from
(
select
n1.n_name as supp_nation,
n2.n_name as cust_nation,
                                        to_number (to_char (l_shipdate,'yyyy')) as l_year,
l_extendedprice * (1 - l_discount) as volume
from
supplier,
lineitem,
orders,
customer,
nation n1,
nation n2
where
s_suppkey = l_suppkey
and o_orderkey = l_orderkey
and c_custkey = o_custkey
and s_nationkey = n1.n_nationkey
and c_nationkey = n2.n_nationkey
and (
(n1.n_name = 'FRANCE' and n2.n_name = 'GERMANY')
or (n1.n_name = 'GERMANY' and n2.n_name = 'FRANCE')
)
and l_shipdate between to_date( '1995-01-01', 'YYYY-MM-DD') and to_date( '1996-12-31', 'YYYY-MM-DD')
) shipping
group by
supp_nation,
cust_nation,
```

```
l_year
order by
supp_nation,
cust_nation,
l_year

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


4 rows processed.
Query Processed in 3.12 seconds.

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


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

O_YEAR	MKT_SHARE
1995.00	0.03
1996.00	0.04

2 rows processed.
Query Processed in 11.23 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
CHINA	1995.00	49287534.62
CHINA	1994.00	50851090.07
CHINA	1993.00	54229629.83
CHINA	1992.00	52400529.37
EGYPT	1998.00	29054433.39
EGYPT	1997.00	50627611.45
EGYPT	1996.00	49542212.84
EGYPT	1995.00	48311550.32
EGYPT	1994.00	49790644.74
EGYPT	1993.00	48904292.97
EGYPT	1992.00	49434932.62
ETHIOPIA	1998.00	28040717.27
ETHIOPIA	1997.00	47455009.87
ETHIOPIA	1996.00	46491097.57
ETHIOPIA	1995.00	46804449.30
ETHIOPIA	1994.00	48516143.92
ETHIOPIA	1993.00	46551891.56
ETHIOPIA	1992.00	44934648.64
FRANCE	1998.00	32226407.84
FRANCE	1997.00	47121485.86
FRANCE	1996.00	47263135.50
FRANCE	1995.00	47275997.57
FRANCE	1994.00	47067209.33
FRANCE	1993.00	51163370.11
FRANCE	1992.00	47846235.33
GERMANY	1998.00	28624942.66
GERMANY	1997.00	49309074.88
GERMANY	1996.00	49918683.17
GERMANY	1995.00	52650718.72
GERMANY	1994.00	50346900.42
GERMANY	1993.00	50991895.81
GERMANY	1992.00	48274126.10
INDIA	1998.00	29943144.35
INDIA	1997.00	50665453.23
INDIA	1996.00	50283092.29
INDIA	1995.00	50006774.64
INDIA	1994.00	48995190.76
INDIA	1993.00	50286902.85
INDIA	1992.00	50850329.40
INDONESIA	1998.00	27672340.00
INDONESIA	1997.00	50512145.73
INDONESIA	1996.00	51653060.12
INDONESIA	1995.00	51508779.59
INDONESIA	1994.00	52817950.32
INDONESIA	1993.00	47959994.96
INDONESIA	1992.00	51776605.03
IRAN	1998.00	29065736.24
IRAN	1997.00	50042063.05
IRAN	1996.00	50926653.19
IRAN	1995.00	51249667.65
IRAN	1994.00	50337085.87
IRAN	1993.00	51730763.49
IRAN	1992.00	49955856.56

IRAQ	1998.00	31624551.00
IRAQ	1997.00	55121749.02
IRAQ	1996.00	55897663.79
IRAQ	1995.00	54815472.52
IRAQ	1994.00	54408516.13
IRAQ	1993.00	53633167.98
IRAQ	1992.00	55891939.34
JAPAN	1998.00	27934179.67
JAPAN	1997.00	44517162.55
JAPAN	1996.00	42545606.12
JAPAN	1995.00	43749356.40
JAPAN	1994.00	44840243.07
JAPAN	1993.00	44660015.53
JAPAN	1992.00	45410249.12
JORDAN	1998.00	26901488.58
JORDAN	1997.00	45471878.41
JORDAN	1996.00	46794325.79
JORDAN	1995.00	45178828.58
JORDAN	1994.00	45333636.51
JORDAN	1993.00	47971496.10
JORDAN	1992.00	44717239.18
KENYA	1998.00	28597614.34
KENYA	1997.00	47949733.73
KENYA	1996.00	46886924.62
KENYA	1995.00	46072338.76
KENYA	1994.00	45772061.17
KENYA	1993.00	46308728.23
KENYA	1992.00	47257780.84
MOROCCO	1998.00	26732115.58
MOROCCO	1997.00	45637304.25
MOROCCO	1996.00	45558221.75
MOROCCO	1995.00	47851318.89
MOROCCO	1994.00	46272172.94
MOROCCO	1993.00	46764326.18
MOROCCO	1992.00	48122783.58
MOZAMBIQUE	1998.00	30712392.01
MOZAMBIQUE	1997.00	50316528.76
MOZAMBIQUE	1996.00	51640320.25
MOZAMBIQUE	1995.00	50693774.51
MOZAMBIQUE	1994.00	49253277.63
MOZAMBIQUE	1993.00	49153016.54
MOZAMBIQUE	1992.00	48247551.85
PERU	1998.00	29326102.32
PERU	1997.00	49753780.40
PERU	1996.00	50935170.29
PERU	1995.00	53309883.41
PERU	1994.00	50643531.80
PERU	1993.00	51584622.00
PERU	1992.00	47523899.05
ROMANIA	1998.00	30368667.40
ROMANIA	1997.00	50365683.85
ROMANIA	1996.00	49598999.01
ROMANIA	1995.00	47537642.87
ROMANIA	1994.00	51455283.01
ROMANIA	1993.00	50407136.89
ROMANIA	1992.00	48185385.13
RUSSIA	1998.00	28322384.03
RUSSIA	1997.00	50106685.18
RUSSIA	1996.00	51753342.43
RUSSIA	1995.00	49215820.36
RUSSIA	1994.00	52205666.44
RUSSIA	1993.00	51860230.03

RUSSIA	1992.00	53251677.15
SAUDI ARABIA	1998.00	31541259.81
SAUDI ARABIA	1997.00	52438750.81
SAUDI ARABIA	1996.00	52543737.82
SAUDI ARABIA	1995.00	52938696.53
SAUDI ARABIA	1994.00	51389601.97
SAUDI ARABIA	1993.00	52937508.88
SAUDI ARABIA	1992.00	54843459.64
UNITED KINGDOM	1998.00	28494874.00
UNITED KINGDOM	1997.00	49381810.90
UNITED KINGDOM	1996.00	51386853.96
UNITED KINGDOM	1995.00	51509586.79
UNITED KINGDOM	1994.00	48086499.71
UNITED KINGDOM	1993.00	49166827.22
UNITED KINGDOM	1992.00	49349122.08
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 8.76 seconds.

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


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

C_CUSTKEY            C_NAME                 REVENUE
C_ACCTBAL            N_NAME
C_ADDRESS                          C_PHONE
C_COMMENT
57040.00             Customer#000057040       734235.25
632.87               JAPAN
Eioyzjf4pp                         22-895-641-3466
requests sleep blithely about the furiously i
143347.00            Customer#000143347       721002.69
2557.47              EGYPT
1aReFYv,Kw4                        14-742-935-3718
fluffily bold excuses haggle finally after the u
60838.00             Customer#000060838       679127.31
2454.77              BRAZIL
64EaJ5vMAHWJlBOxJklpNc2RJiWE           12-913-494-9813
furiously even pinto beans integrate under the ruthless foxes; ironic, even dolphins across the slyl
101998.00            Customer#000101998       637029.57
3790.89              UNITED KINGDOM
01c9CILnNtfOQYmZj                  33-593-865-6378
accounts doze blithely! enticing, final deposits sleep blithely special accounts. slyly express accounts pla
125341.00            Customer#000125341       633508.09
4983.51              GERMANY
```

S29ODD6bceU8QSuuEJznkNaK 17-582-695-5962
quickly express requests wake quickly blithely
25501.00 Customer#000025501 620269.78
7725.04 ETHIOPIA
W556MXuoiaYCCZamJI,Rn0B4ACUGdkQ8DZ 15-874-808-6793
quickly special requests sleep evenly among the special deposits. special deposi
115831.00 Customer#000115831 596423.87
5098.10 FRANCE
rFeBbEEyk dl ne7zV5fDrmiq1oK09wV7pxqCgIc 16-715-386-3788
carefully bold excuses sleep alongside of the thinly idle
84223.00 Customer#000084223 594998.02
528.65 UNITED KINGDOM
nAVZCs6BaWap rrM27N 2qBnzc5WBauxbA 33-442-824-8191
pending, final ideas haggle final requests. unusual, regular asymptotes affix according to the even foxes.
54289.00 Customer#000054289 585603.39
5583.02 IRAN
vXCxoCsU0Bad5JQI ,oobkZ 20-834-292-4707
express requests sublate blithely regular requests. regular, even ideas solve.
39922.00 Customer#000039922 584878.11
7321.11 GERMANY
Zgy4s50l2GKN4pLDPBU8m342gIw6R 17-147-757-8036
even pinto beans haggle. slyly bold accounts inte
6226.00 Customer#000006226 576783.76
2230.09 UNITED KINGDOM
8gPu8,NPGkfyQQ0hcIYUGPIBWc,ybP5g, 33-657-701-3391
quickly final requests against the regular instructions wake blithely final instructions. pa
922.00 Customer#000000922 576767.53
3869.25 GERMANY
Az9RFaut7NkPnc5zSD2PwHgVwr4jRzq 17-945-916-9648
boldly final requests cajole blith
147946.00 Customer#000147946 576455.13
2030.13 ALGERIA
iANyZHjqhyy7Ajah0pTrYyhJ 10-886-956-3143
furiously even accounts are blithely above the furiousl
115640.00 Customer#000115640 569341.19
6436.10 ARGENTINA
Vtgfia9qI 7EpHgecU1X 11-411-543-4901
final instructions are slyly according to the
73606.00 Customer#000073606 568656.86
1785.67 JAPAN
xuR0Tro5yChDfOCrjkd2ol 22-437-653-6966
furiously bold orbits about the furiously busy requests wake across the furiously quiet theodolites. d
110246.00 Customer#000110246 566842.98
7763.35 VIETNAM
7KzflgX MDOq7sOkI 31-943-426-9837
dolphins sleep blithely among the slyly final
142549.00 Customer#000142549 563537.24
5085.99 INDONESIA
ChqEoK43OysjdHbtKCp6dKqjNyvvi9 19-955-562-2398
regular, unusual dependencies boost slyly; ironic attainments nag fluffily into the unusual packages?
146149.00 Customer#000146149 557254.99
1791.55 ROMANIA
s87fvzFQpU 29-744-164-6487
silent, unusual requests detect quickly slyly regul
52528.00 Customer#000052528 556397.35
551.79 ARGENTINA
NFztyTOR10UOJ 11-208-192-3205
unusual requests detect. slyly dogged theodolites use slyly. deposit
23431.00 Customer#000023431 554269.54
3381.86 ROMANIA
HgiV0phqhaIa9aydNoIlb 29-915-458-2654
instructions nag quickly. furiously bold accounts cajol

20 rows processed.
Query Processed in 6.72 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
96493.00	13339057.83
56548.00	13329014.97
55576.00	13306843.35
159751.00	13306614.48
92406.00	13287414.50
182636.00	13223726.74
199969.00	13135288.21
62865.00	13001926.94
7284.00	12945298.19
197867.00	12944510.52
11562.00	12931575.51
75165.00	12916918.12
97175.00	12911283.50
140840.00	12896562.23
65241.00	12890600.46
166120.00	12876927.22
9035.00	12863828.70
144616.00	12853549.30
176723.00	12832309.74
170884.00	12792136.58
29790.00	12723300.33
95213.00	12555483.73
183873.00	12550533.05
171235.00	12476538.30
21533.00	12437821.32
17290.00	12432159.50
156397.00	12260623.50
122611.00	12222812.98
139155.00	12220319.25
146316.00	12215800.61
171381.00	12199734.52
198633.00	12078226.95
167417.00	12046637.62
59512.00	12043468.76
31688.00	12034893.64
159586.00	12001505.84
8993.00	11963814.30
120302.00	11857707.55
43536.00	11779340.52
9552.00	11776909.16
86223.00	11772205.08
53776.00	11758669.65
131285.00	11616953.74
91628.00	11611114.83
169644.00	11567959.72
182299.00	11567462.05
33107.00	11453818.76
104184.00	11436657.44
67027.00	11419127.14
176869.00	11371451.71
30885.00	11369674.79
54420.00	11345076.88
72240.00	11313951.05
178708.00	11294635.17
81298.00	11273686.13
158324.00	11243442.72
117095.00	11242535.24

176793.00 11237733.38
86091.00 11177793.79
116033.00 11145434.36
129058.00 11119112.20
193714.00 11104706.39
117195.00 11077217.96
49851.00 11043701.78
19791.00 11030662.62
75800.00 11012401.62
161562.00 10996371.69
10119.00 10980015.75
39185.00 10970042.56
47223.00 10950022.13
175594.00 10942923.05
111295.00 10893675.61
155446.00 10852764.57

128435.00 8004242.88
92516.00 8003836.80
30802.00 8003710.88
107418.00 8000430.30
46620.00 7999778.35
191803.00 7994734.15
106343.00 7993087.76
59362.00 7990397.46
8329.00 7990052.90
75133.00 7988244.00
179023.00 7986829.62
135899.00 7985726.64
5824.00 7985340.02
148579.00 7984889.56
95888.00 7984735.72
9791.00 7982699.79
170437.00 7982370.72
39782.00 7977858.24
20605.00 7977556.00
28682.00 7976960.00
42172.00 7973399.00
56137.00 7971405.40
64729.00 7970769.72
98643.00 7968603.73
153787.00 7967535.58
8932.00 7967222.19
20134.00 7965713.28
197635.00 7963507.58
80408.00 7963312.17
37728.00 7961875.68
26624.00 7961772.31
44736.00 7961144.10
29763.00 7960605.03
36147.00 7959463.68
146040.00 7957587.66
115469.00 7957485.14
142276.00 7956790.63
181280.00 7954037.35
115096.00 7953047.55
109650.00 7952258.73
93862.00 7951992.24
158325.00 7950728.30

55952.00	7950387.06
122397.00	7947106.27
28114.00	7946945.72
11966.00	7945197.48
47814.00	7944083.00
85096.00	7943691.06
51657.00	7943593.77
196680.00	7943578.89
13141.00	7942730.34
193327.00	7941036.25
152612.00	7940663.71
139680.00	7939242.36
31134.00	7938318.30
45636.00	7937240.85
56694.00	7936015.95
8114.00	7933921.88
71518.00	7932261.69
72922.00	7930400.64
146699.00	7929167.40
92387.00	7928972.67
186289.00	7928786.19
95952.00	7927972.78
196514.00	7927180.70
4403.00	7925729.04
2267.00	7925649.37
45924.00	7925047.68
11493.00	7916722.23
104478.00	7916253.60
166794.00	7913842.00
161995.00	7910874.27
23538.00	7909752.06
41093.00	7909579.92
112073.00	7908617.57
92814.00	7908262.50
88919.00	7907992.50
79753.00	7907933.88
108765.00	7905338.98
146530.00	7905336.60
71475.00	7903367.58
36289.00	7901946.50
61739.00	7900794.00
52338.00	7898638.08
194299.00	7898421.24
105235.00	7897829.94
77207.00	7897752.72
96712.00	7897575.27
10157.00	7897046.25
171154.00	7896814.50
79373.00	7896186.00
113808.00	7893353.88
27901.00	7892952.00
128820.00	7892882.72
25891.00	7890511.20
122819.00	7888881.02
154731.00	7888301.33
101674.00	7879324.60
51968.00	7879102.21
72073.00	7877736.11
5182.00	7874521.73

1048 rows processed.

Query Processed in 4.23 seconds.

```
-- @(#)12.sql          2.1.6.2
-- TPC-H/TPC-R Shipping Modes and Order Priority Query(Q12)
-- Variant A
-- Approved February 1998


select
l_shipmode,
sum(decode(o_orderpriority,  '1-URGENT',  1, '2-HIGH',  1,  0))
as high_line_count,
sum(decode(o_orderpriority,  '1-URGENT',  0, '2-HIGH',  0,  1))
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       0.00                   15526.00
SHIP       0.00                   15462.00
```

2 rows processed.
Query Processed in 4.35 seconds.

```
-- @(#)13.sql          2.1.6.2
-- TPC-H/TPC-R Customer Distribution Query (Q13)
-- Functional Query Definition
-- Approved February 1998

select
c_count,
count(*) as custdist
from
(
select
c_custkey,
count(o_orderkey)  as  c_count
from
customer, orders where
c_custkey = o_custkey(+)
and o_comment(+) not like '%special%requests%'
group by
c_custkey
) c_orders
group by
c_count
order by
custdist desc,
c_count desc
```

C_COUNT	CUSTDIST
0.00	50004.00
9.00	6641.00
10.00	6566.00
11.00	6058.00
8.00	5949.00
12.00	5553.00
13.00	4989.00
19.00	4748.00
7.00	4707.00
18.00	4625.00
15.00	4552.00
17.00	4530.00
14.00	4484.00
20.00	4461.00
16.00	4323.00
21.00	4217.00
22.00	3730.00
6.00	3334.00
23.00	3129.00
24.00	2622.00
25.00	2079.00
5.00	1972.00
26.00	1593.00
27.00	1185.00
4.00	1033.00
28.00	869.00
29.00	559.00
3.00	398.00
30.00	373.00
31.00	235.00
2.00	144.00
32.00	128.00
33.00	71.00
34.00	48.00
35.00	33.00
1.00	23.00
36.00	17.00
37.00	7.00
40.00	4.00
38.00	4.00
39.00	2.00
41.00	1.00

42 rows processed.
Query Processed in 1.85 seconds.

```
-- using 609021123 as a seed to the RNG
-- @(#)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 0.60 seconds.

-- @(#)15.sql          2.1.6.2
-- TPC-H/TPC-R Top Supplier Query (Q15)
-- Variant A
-- Approved February 1998

with   revenue0
as (select
l_suppkey supplier_no,
sum(l_extendedprice * (1 - l_discount)) total_revenue
from
lineitem
where
                l_shipdate >= date '1996-01-01'
                and l_shipdate < date '1996-01-01' + interval '3' month
group by
l_suppkey)
select
s_suppkey,
s_name,
s_address,
s_phone,
total_revenue
from
supplier,
revenue0
where
s_suppkey = supplier_no
and total_revenue = (
select
max(total_revenue)
from
revenue0  )
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 42.88 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
Brand#25   MEDIUM BURNISHED COPPER   36.00                24.00
Brand#31   PROMO POLISHED COPPER     36.00                24.00
Brand#33   LARGE POLISHED TIN        23.00                24.00
Brand#33   PROMO POLISHED STEEL      14.00                24.00
```

Brand#35	PROMO BRUSHED NICKEL	14.00	24.00
Brand#41	ECONOMY BRUSHED STEEL	9.00	24.00
Brand#41	ECONOMY POLISHED TIN	19.00	24.00
Brand#41	LARGE PLATED COPPER	36.00	24.00
Brand#42	ECONOMY PLATED BRASS	3.00	24.00
Brand#42	STANDARD POLISHED TIN	49.00	24.00
Brand#43	PROMO BRUSHED TIN	3.00	24.00
Brand#43	SMALL ANODIZED COPPER	36.00	24.00
Brand#44	STANDARD POLISHED NICKEL	3.00	24.00
Brand#52	ECONOMY PLATED TIN	14.00	24.00
Brand#52	STANDARD BURNISHED NICKEL	3.00	24.00
Brand#53	MEDIUM ANODIZED STEEL	14.00	24.00
Brand#14	PROMO ANODIZED NICKEL	45.00	23.00
Brand#32	ECONOMY PLATED BRASS	9.00	23.00
Brand#52	SMALL ANODIZED COPPER	3.00	23.00
Brand#11	ECONOMY BRUSHED COPPER	45.00	20.00
Brand#11	ECONOMY PLATED BRASS	23.00	20.00
Brand#11	LARGE BRUSHED COPPER	49.00	20.00
Brand#11	LARGE POLISHED COPPER	49.00	20.00
Brand#12	STANDARD ANODIZED TIN	49.00	20.00
Brand#12	STANDARD PLATED BRASS	19.00	20.00
Brand#13	ECONOMY BRUSHED BRASS	9.00	20.00
Brand#13	ECONOMY BURNISHED STEEL	14.00	20.00
Brand#13	LARGE BURNISHED NICKEL	19.00	20.00
Brand#13	MEDIUM BURNISHED COPPER	36.00	20.00
Brand#13	SMALL BRUSHED TIN	45.00	20.00
Brand#13	STANDARD ANODIZED COPPER	3.00	20.00
Brand#13	STANDARD PLATED NICKEL	23.00	20.00
Brand#14	ECONOMY ANODIZED COPPER	14.00	20.00
Brand#14	ECONOMY PLATED TIN	36.00	20.00
Brand#14	ECONOMY POLISHED NICKEL	3.00	20.00
Brand#14	MEDIUM ANODIZED NICKEL	3.00	20.00
Brand#14	SMALL POLISHED TIN	14.00	20.00
Brand#15	MEDIUM ANODIZED COPPER	9.00	20.00
Brand#15	MEDIUM PLATED TIN	23.00	20.00
Brand#15	PROMO PLATED BRASS	14.00	20.00
Brand#15	SMALL ANODIZED COPPER	45.00	20.00
Brand#15	SMALL PLATED COPPER	49.00	20.00
Brand#15	STANDARD PLATED TIN	3.00	20.00
Brand#21	LARGE ANODIZED COPPER	36.00	20.00
Brand#21	LARGE BRUSHED TIN	3.00	20.00
Brand#21	MEDIUM ANODIZED COPPER	14.00	20.00
Brand#21	PROMO BRUSHED TIN	36.00	20.00
Brand#21	PROMO POLISHED NICKEL	45.00	20.00
Brand#21	SMALL ANODIZED COPPER	9.00	20.00
Brand#21	SMALL POLISHED NICKEL	23.00	20.00
Brand#22	LARGE ANODIZED COPPER	36.00	20.00
Brand#22	LARGE BRUSHED COPPER	49.00	20.00
Brand#22	PROMO ANODIZED TIN	49.00	20.00
Brand#22	PROMO POLISHED BRASS	45.00	20.00
Brand#22	SMALL BURNISHED STEEL	45.00	20.00
Brand#23	MEDIUM ANODIZED STEEL	45.00	20.00
Brand#23	PROMO POLISHED STEEL	23.00	20.00
Brand#23	STANDARD BRUSHED TIN	14.00	20.00
Brand#23	STANDARD PLATED NICKEL	36.00	20.00
Brand#24	PROMO PLATED COPPER	49.00	20.00
Brand#24	PROMO PLATED STEEL	49.00	20.00
Brand#24	PROMO POLISHED STEEL	9.00	20.00
Brand#24	STANDARD BRUSHED TIN	36.00	20.00
Brand#25	LARGE ANODIZED BRASS	3.00	20.00
Brand#25	PROMO BURNISHED TIN	3.00	20.00
Brand#31	ECONOMY POLISHED NICKEL	3.00	20.00

Brand#31	MEDIUM PLATED TIN	45.00	20.00
Brand#31	SMALL ANODIZED STEEL	14.00	20.00
Brand#32	ECONOMY ANODIZED COPPER	36.00	20.00
Brand#32	ECONOMY BRUSHED NICKEL	49.00	20.00
Brand#32	LARGE ANODIZED TIN	19.00	20.00
Brand#32	MEDIUM BURNISHED COPPER	19.00	20.00
Brand#32	SMALL ANODIZED STEEL	45.00	20.00
Brand#33	ECONOMY POLISHED COPPER	19.00	20.00
Brand#33	PROMO PLATED NICKEL	14.00	20.00
Brand#33	SMALL POLISHED TIN	9.00	20.00
Brand#33	STANDARD ANODIZED BRASS	49.00	20.00
Brand#33	STANDARD BURNISHED BRASS	45.00	20.00
Brand#34	ECONOMY BRUSHED NICKEL	49.00	20.00
Brand#34	LARGE BRUSHED BRASS	19.00	20.00
Brand#34	SMALL BRUSHED TIN	3.00	20.00
Brand#34	STANDARD PLATED COPPER	9.00	20.00
Brand#35	LARGE ANODIZED NICKEL	3.00	20.00
Brand#35	MEDIUM ANODIZED BRASS	45.00	20.00
Brand#35	MEDIUM ANODIZED STEEL	23.00	20.00
Brand#35	PROMO ANODIZED COPPER	49.00	20.00
Brand#35	SMALL POLISHED COPPER	14.00	20.00
Brand#41	LARGE ANODIZED STEEL	3.00	20.00
Brand#41	LARGE BRUSHED NICKEL	23.00	20.00
Brand#41	LARGE BURNISHED COPPER	3.00	20.00
Brand#41	MEDIUM PLATED STEEL	19.00	20.00

Brand#55	SMALL POLISHED COPPER	45.00	4.00
Brand#55	SMALL POLISHED COPPER	49.00	4.00
Brand#55	SMALL POLISHED NICKEL	9.00	4.00
Brand#55	SMALL POLISHED NICKEL	14.00	4.00
Brand#55	SMALL POLISHED NICKEL	19.00	4.00
Brand#55	SMALL POLISHED NICKEL	23.00	4.00
Brand#55	SMALL POLISHED NICKEL	45.00	4.00
Brand#55	SMALL POLISHED NICKEL	49.00	4.00
Brand#55	SMALL POLISHED STEEL	19.00	4.00
Brand#55	SMALL POLISHED STEEL	45.00	4.00
Brand#55	SMALL POLISHED TIN	14.00	4.00
Brand#55	SMALL POLISHED TIN	23.00	4.00
Brand#55	SMALL POLISHED TIN	45.00	4.00
Brand#55	STANDARD ANODIZED BRASS	9.00	4.00
Brand#55	STANDARD ANODIZED BRASS	23.00	4.00
Brand#55	STANDARD ANODIZED BRASS	49.00	4.00
Brand#55	STANDARD ANODIZED COPPER	9.00	4.00
Brand#55	STANDARD ANODIZED COPPER	14.00	4.00
Brand#55	STANDARD ANODIZED COPPER	45.00	4.00
Brand#55	STANDARD ANODIZED NICKEL	3.00	4.00
Brand#55	STANDARD ANODIZED NICKEL	14.00	4.00
Brand#55	STANDARD ANODIZED NICKEL	45.00	4.00
Brand#55	STANDARD ANODIZED NICKEL	49.00	4.00
Brand#55	STANDARD ANODIZED STEEL	3.00	4.00
Brand#55	STANDARD ANODIZED STEEL	14.00	4.00
Brand#55	STANDARD ANODIZED TIN	14.00	4.00
Brand#55	STANDARD ANODIZED TIN	36.00	4.00
Brand#55	STANDARD ANODIZED TIN	45.00	4.00
Brand#55	STANDARD BRUSHED BRASS	9.00	4.00
Brand#55	STANDARD BRUSHED BRASS	19.00	4.00
Brand#55	STANDARD BRUSHED COPPER	14.00	4.00
Brand#55	STANDARD BRUSHED COPPER	19.00	4.00
Brand#55	STANDARD BRUSHED NICKEL	3.00	4.00
Brand#55	STANDARD BRUSHED NICKEL	36.00	4.00

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

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

18314 rows processed.
Query Processed in 1.02 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 4.51 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	07-APR-94	544089.09	323.00
Customer#000144617	144617.00	3043270.00	12-FEB-97	530604.44	317.00
Customer#000013940	13940.00	2232932.00	13-APR-97	522720.61	304.00
Customer#000066790	66790.00	2199712.00	30-SEP-96	515531.82	327.00
Customer#000046435	46435.00	4745607.00	03-JUL-97	508047.99	309.00
Customer#000015272	15272.00	3883783.00	28-JUL-93	500241.33	302.00
Customer#000146608	146608.00	3342468.00	12-JUN-94	499794.58	303.00
Customer#000096103	96103.00	5984582.00	16-MAR-92	494398.79	312.00
Customer#000024341	24341.00	1474818.00	15-NOV-92	491348.26	302.00
Customer#000137446	137446.00	5489475.00	23-MAY-97	487763.25	311.00
Customer#000107590	107590.00	4267751.00	04-NOV-94	485141.38	301.00
Customer#000050008	50008.00	2366755.00	09-DEC-96	483891.26	302.00
Customer#000015619	15619.00	3767271.00	07-AUG-96	480083.96	318.00
Customer#000077260	77260.00	1436544.00	12-SEP-92	479499.43	307.00
Customer#000109379	109379.00	5746311.00	10-OCT-96	478064.11	302.00
Customer#000054602	54602.00	5832321.00	09-FEB-97	471220.08	307.00
Customer#000105995	105995.00	2096705.00	03-JUL-94	469692.58	307.00
Customer#000148885	148885.00	2942469.00	31-MAY-92	469630.44	313.00
Customer#000114586	114586.00	551136.00	19-MAY-93	469605.59	308.00
Customer#000105260	105260.00	5296167.00	06-SEP-96	469360.57	303.00
Customer#000147197	147197.00	1263015.00	02-FEB-97	467149.67	320.00
Customer#000064483	64483.00	2745894.00	04-JUL-96	466991.35	304.00
Customer#000136573	136573.00	2761378.00	31-MAY-96	461282.73	301.00
Customer#000016384	16384.00	502886.00	12-APR-94	458378.92	312.00
Customer#000117919	117919.00	2869152.00	20-JUN-96		

456815.92 317.00
Customer#000012251 12251.00 735366.00 24-NOV-93
455107.26 309.00
Customer#000120098 120098.00 1971680.00 14-JUN-95
453451.23 308.00
Customer#000066098 66098.00 5007490.00 07-AUG-92
453436.16 304.00
Customer#000117076 117076.00 4290656.00 05-FEB-97
449545.85 301.00
Customer#000129379 129379.00 4720454.00 07-JUN-97
448665.79 303.00
Customer#000126865 126865.00 4702759.00 07-NOV-94
447606.65 320.00
Customer#000088876 88876.00 983201.00 30-DEC-93
446717.46 304.00
Customer#000036619 36619.00 4806726.00 17-JAN-95
446704.09 328.00
Customer#000141823 141823.00 2806245.00 29-DEC-96
446269.12 310.00
Customer#000053029 53029.00 2662214.00 13-AUG-93
446144.49 302.00
Customer#000018188 18188.00 3037414.00 25-JAN-95
443807.22 308.00
Customer#000066533 66533.00 29158.00 21-OCT-95
443576.50 305.00
Customer#000037729 37729.00 4134341.00 29-JUN-95
441082.97 309.00
Customer#000003566 3566.00 2329187.00 04-JAN-98
439803.36 304.00
Customer#000045538 45538.00 4527553.00 22-MAY-94
436275.31 305.00
Customer#000081581 81581.00 4739650.00 04-NOV-95
435405.90 305.00
Customer#000119989 119989.00 1544643.00 20-SEP-97
434568.25 320.00
Customer#000003680 3680.00 3861123.00 03-JUL-98
433525.97 301.00
Customer#000113131 113131.00 967334.00 15-DEC-95
432957.75 301.00
Customer#000141098 141098.00 565574.00 24-SEP-95
430986.69 301.00
Customer#000093392 93392.00 5200102.00 22-JAN-97
425487.51 304.00
Customer#000015631 15631.00 1845057.00 12-MAY-94
419879.59 302.00
Customer#000112987 112987.00 4439686.00 17-SEP-96
418161.49 305.00
Customer#000012599 12599.00 4259524.00 12-FEB-98
415200.61 304.00
Customer#000105410 105410.00 4478371.00 05-MAR-96
412754.51 302.00
Customer#000149842 149842.00 5156581.00 30-MAY-94
411329.35 302.00
Customer#000010129 10129.00 5849444.00 21-MAR-94
409129.85 309.00
Customer#000069904 69904.00 1742403.00 19-OCT-96
408513.00 305.00
Customer#000017746 17746.00 6882.00 09-APR-97
408446.93 303.00
Customer#000013072 13072.00 1481925.00 15-MAR-98
399195.47 301.00
Customer#000082441 82441.00 857959.00 07-FEB-94

```
382579.74           305.00
Customer#000088703      88703.00            2995076.00          30-JAN-94
363812.12           302.00


57 rows processed.
Query Processed in 4.81 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 4.32 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
Supplier#000000761     zlSLelQUj2XrvTTFnv7WAcYZGvvMTx882d4
Supplier#000000884     bmhEShejaS
Supplier#000000887     urEaTejH5POADP2ARrf
Supplier#000000935     ij98czM 2KzWe7dDTOxB8sq0UfCdvrX
Supplier#000000975     ,AC e,tBpNwKb5xMUzeohxlRn, hdZJo73gFQF8y
Supplier#000001263     rQWr6nf8ZhB2TAiIDIvo5Io
```

Supplier#000001399	LmrocnIMSyYOWuANx7
Supplier#000001446	lch9HMNU1R7a0LIybsUodVknk6
Supplier#000001454	TOpimgu2TVXIjhiL93h,
Supplier#000001500	wDmF5xLxtQch9ctVu,
Supplier#000001602	uKNWIeafaM644
Supplier#000001626	UhxNRzUu1dtFmp0
Supplier#000001682	pXTkGxrTQVyH1Rr
Supplier#000001699	Q9C4rfJ26oijVPqqcqVXeRI
Supplier#000001700	7hMlCof1Y5zLFg
Supplier#000001726	TeRY7TtTH24sEword7yAaSkjx8
Supplier#000001730	Rc8e,1Pybn r6zo0VJIEiD0UD vhk
Supplier#000001746	qWsendlOekQG1aW4uq06uQaCm51se8lirv7 hBRd
Supplier#000001752	Fra7outx41THYJaRThdOGiBk
Supplier#000001856	jXcRgzYF0ah05iR8p6w5SbJJLcUGyYiURPvFwUWM
Supplier#000001931	FpJbMU2h6ZR2eBv8I9NIxF
Supplier#000001939	Nrk,JA4bfReUs
Supplier#000001990	DSDJkCgBJzuPg1yuM,CUdLnsRliOxkkHezTCA
Supplier#000002020	jB6r1d7MxP6co
Supplier#000002022	dwebGX7Id2pc25YvY33
Supplier#000002036	20ytTtVObjKUUI2WCB0A
Supplier#000002204	uYmlr46C06udCqanj0KiRsoTQakZsEyssL
Supplier#000002243	nSOEV3JeOU79
Supplier#000002245	hz2qWXWVjOyKhqPYMoEwz6zFkrTaDM
Supplier#000002282	ES21K9dxoW1I1TzWCj7ekdlNwSWnv1Z 6mQ,BKn
Supplier#000002303	nCoWfpB6YOymbgOht7ltfklpkHl
Supplier#000002373	RzHSxOTQmElCjxIBiVA52Z JB58rJhPRylR
Supplier#000002419	qydBQd14I5l5mVXa4fYY
Supplier#000002481	nLKHUOn2Ml9TOA06Znq9GEMcIlMO2
Supplier#000002571	JZUugz04c iJFLrlGsz9O N,W 1rVHNIReyq
Supplier#000002585	CsPoKpw2QuTY4AV1NkWuttneIa4SN
Supplier#000002630	ZIQAvjNUY9KH5ive zm7k VlPiDl7CCo21
Supplier#000002719	4nnzQI2CbqREQUuIsXTBVUkaP4mNS3
Supplier#000002721	HVdFAN2JHMQSpKm
Supplier#000002730	lIFxR4fzm31C6,muzJwl84z
Supplier#000002775	yDclaDaBD4ihH
Supplier#000002853	rTNAOItXka
Supplier#000002875	6JgMi 9Qt6VmwL3Ltt1SRlKww0keLQ,RAZa
Supplier#000002934	m,trBENywSArwg3DhB
Supplier#000002941	Naddba 8YTEKekZyP0
Supplier#000002960	KCPCEsRGGo6vx8TygHh60nAYf9rStQT2T
Supplier#000002980	B9k9yVsyaXvWktOSHezqHiAEp9id0SKzkw
Supplier#000003062	LSQNgqY1xnOzz9zBCapy7HwOZQ
Supplier#000003087	ANwe8QsZ4rgj1HSqVz991eWQ
Supplier#000003089	s5b VCIZqMSZVa r g7LTdcg29GbTE7rI1x
Supplier#000003095	HxON3jJhUi3zjt,r mTD
Supplier#000003201	E87yws6I,t0qNs4QW7UzExKiJnJDZWue
Supplier#000003213	pxrRP4irQ1VoyfQ,dTf3
Supplier#000003241	j06SU,LS9O3mwjAMOViANeIhb
Supplier#000003275	9xO4nyJ2QJcX6vGf
Supplier#000003288	EDdfNt7E5Uc,xLTupoIgYL4yY7ujh,
Supplier#000003313	El2I7we,049SPrvomUm4hZwJoOhZkvLxLJXgVH
Supplier#000003314	jnisU8MzqO4iUB3zsPcrysMw3DDUojS4q7LD
Supplier#000003380	jPv0V,pszouuFT3YsAqlP,kxT3u,gTFiEbRt,x
Supplier#000003403	e3X2o ,KCG9tsHji8A XXCxiF2hZWBw
Supplier#000003421	Sh3dt9W5oeofFWovnFhrg,
Supplier#000003441	zvFJIzS,oUuShHjpcX
Supplier#000003590	sy79CMLxqb,Cbo
Supplier#000003607	lNqFHQYjwSAkf
Supplier#000003625	qY588W0Yk5iaUy1RXTgNrEKrMAjBYHcKs
Supplier#000003656	eEYmmO2gmD JdfG32XtDgJV,db56
Supplier#000003782	iVsPZg7bk06TqNMwi0LKbLUrC1zmrg
Supplier#000003918	meRvRCsJoAbfqd0Re4

Supplier#000003941	Pmb05mQfBMS618O7WKqZJ 9vyv
Supplier#000003994	W00LZp3NjK0
Supplier#000004005	V723F1wCy2eA4OgIu8TjBtOVUHp
Supplier#000004033	ncsAhv9Je,kFXTNjfb2
Supplier#000004140	0hL7DJyYjcHL
Supplier#000004165	wTJ2dZNQA8P2oi99N6DT47ndHy,XKD2
Supplier#000004207	tF64pwiOM4IkWjN3mS,e06WuAjLx
Supplier#000004236	dl,HPtJmGipxYsSqn9wmqkuWjst,mCeJ8O6T
Supplier#000004246	Xha aXQF7u4qU3LsHD
Supplier#000004278	bBddbpBxIVp Di9
Supplier#000004343	GK3sbopqrQEkWLMvVBFCG
Supplier#000004346	S3076LEOwo
Supplier#000004388	VfZ l1J,mwp4aS
Supplier#000004406	Ah0ZaLu6VwufPWUz,7kbXgYZhauEaHqGIg
Supplier#000004430	yvSsKNSTL5HLXBET4luOsPNLxKzAMk
Supplier#000004522	xXtCKwsZDArxIBGDfzX2PgobGZsBg
Supplier#000004527	p pVXCnxgcklWF6A1o3OHY3qW6
Supplier#000004542	NJSbLJDroYG2y1r3rDiKg
Supplier#000004574	1HvGwnVueZ5CIndc
Supplier#000004655	67NqBc4 t3PG3F8aO IsqWNq4kGaPowYL
Supplier#000004701	6jX4u47URzIMHf
Supplier#000004711	bEzjp1QdQu ls2ERMxv0km vn6bu2zXlL1
Supplier#000004987	UFx1upJ8MvOvgFjA8
Supplier#000005000	DeX804 w0H8FrCUvahgy ilbuzBX3NK
Supplier#000005100	OfvYPs3Io,wEvvLHNaLuCX
Supplier#000005192	JDp4rhXiDw0kf6RH
Supplier#000005195	Woi3b2ZaicPh ZSfu1EfXhE
Supplier#000005283	5fxYXxwXy,TQX,MqDC2hxzyQ
Supplier#000005300	gXG28YqpxU
Supplier#000005386	Ub6AAfHpWLWP
Supplier#000005426	9Dz2OVT1q sb4BK71ljQ1XjPBYRPvO
Supplier#000005484	saFdOR qW7AFY,3asPqiiAa11Mo22pCoN0BtPrKo
Supplier#000005505	d2sbjG43KwMPX
Supplier#000005506	On f5ypzoWgB
Supplier#000005516	XsN99Ks9wEvcohU6jRD2MeebQFf76mD8vovuY
Supplier#000005536	Nzo9tGkpgbHT,EZ4D,77MYKl4ah1C
Supplier#000005605	7Vj6Eil0mThqkM
Supplier#000005631	14TVrjlzo2SJEBYCDgpMwTlvwSqC
Supplier#000005730	5rkb0PSews HvxkL8JaD41UpnSF2cg8H1
Supplier#000005736	2dq XTYhtYWSfp
Supplier#000005737	dmEWcS32C3kx,d,B95 OmYn48
Supplier#000005797	,o,OebwRbSDmVl9gN9fpWPCiqB UogvlSR
Supplier#000005836	tx3SjPD2ZuWGFBRH,
Supplier#000005875	lK,sYiGzB94hSyHy9xvSZFbVQNCZe2LXZuGbS
Supplier#000005974	REhR5jE,lLusQXvf54SwYySgsSSVFhu
Supplier#000005989	rjFY,5kgLpBu7c
Supplier#000006059	4m0cv8MwJ9yX2vlwI Z
Supplier#000006065	UiI2Cy3W4Tu5sLk LuvXLRy6KihlGv
Supplier#000006070	TalC5m0pDrO6DZbngfmGmqe
Supplier#000006109	rY5gbfh3dKHnylcQUTPGCwnbe
Supplier#000006121	S92ycWwEzYYw4GspCBJN1WMuHhoZ
Supplier#000006215	j2iEbTsl,5PWdqWZ7k1yiISb7qtiiZljDIPEo
Supplier#000006217	RVN23SYT9jenUeaWGXUd
Supplier#000006274	S3yTZWqxTKUq g QQgcW9 AqhCkNZsW51hHuwU
Supplier#000006435	xIgE69XszYbnO4Eon7cHHO8y
Supplier#000006463	7 wkdj2EO49iotley2kmIM ADpLSszGV3RNWj
Supplier#000006493	ojV f,sNaB6Hm7r,fknDVTL63raJgAjZK
Supplier#000006521	b9 2zjHzxR
Supplier#000006607	3F 2e2gqD5u5B
Supplier#000006706	Ak4ga,ePu1QZ6C3qkrqjosaX0gxvqS9vkbe
Supplier#000006761	n4jhxGMqB5prD1HhpLvwrWStOLlla
Supplier#000006808	HGd2Xo 9nEcHJhZvXjXxWKIpApT

Supplier#000006858 fnlINT885vBBhsWwTGiZ0o22thwGY16h GHJj21
Supplier#000006872 XIDPiA7PLXCWK6SeEcld
Supplier#000006949 mLxYUJhsGcLtKe ,GFirNu183AvT
Supplier#000006985 PrUUiboQpy,OtgJ01Z4BxJQUyrw9c3I
Supplier#000007072 2tRyX9M1a 4Rcm57s779F1ANG9jlpK
Supplier#000007098 G3j8g0KC4OcbAu2OVoPHrXQWMCUdjq8wgCHOExu
Supplier#000007135 ls DoKV7V5ulfQy9V
Supplier#000007160 TqDGBULB3cTqIT6FKDvm9BS4e4v,zwYiQPb
Supplier#000007169 tEc95D2moN9S84nd55O,dlnW
Supplier#000007322 wr7dgte5q MAjiY0uwmi3MyDkSMX1
Supplier#000007365 51xhROLvQMJ05DndtZWt
Supplier#000007398 V8eE6oZ00OFNU,
Supplier#000007402 4UVv58ery1rjmqSR5
Supplier#000007448 yhhpWiJi7EJ6Q5VCaQ
Supplier#000007477 9m9j0wfhWzCvVHxkU,PpAxwSH0h
Supplier#000007509 q8,V6LJRoHJjHcOuSG7aLTMg
Supplier#000007561 rMcFg2530VC
Supplier#000007789 rQ7cUcPrtudOyO3svNSkimqH6qrfWT2Sz
Supplier#000007801 69fi,U1r6enUb
Supplier#000007818 yhhc2CQec Jrvc8zqBi83
Supplier#000007885 u3sicchh5ZpyTUpN1cJKNcAoabIWgY
Supplier#000007918 r,v9mBQ6LoEYyj1
Supplier#000007926 ErzCF80K9Uy
Supplier#000007957 ELwnio14ssoU1 dRyZIL OK3Vtzb
Supplier#000007965 F7Un5lJ7p5hhj
Supplier#000007968 DsF9UlZ2Fo6HXN9aErvyg1ikHoD582HSGZpP
Supplier#000007998 LnASFBfYRFOo9d6d,asBvVq9Lo2P
Supplier#000008168 aOa82a8ZbKCnfDLX
Supplier#000008231 IK7eGw Yj90sTdpsP,vcqWxLB
Supplier#000008243 2AyePMkDqmzVzjGTizXthFLo8h EiudCMxOmIIG
Supplier#000008275 BlbNDfWg,gpXKQlLN
Supplier#000008323 75I18sZmASwm POeheRMdj9tmpyeQ,BfCXN5BIAb
Supplier#000008366 h778cEj14BuW9OEKlvPTWq4iwASR6EBBXN7zeS8
Supplier#000008423 RQhKnkAhR0DAr3Ix4Q1weMMn00hNe Kq
Supplier#000008480 4sSDA4ACReklNjEm5T6b
Supplier#000008532 Uc29q4,5xVdDOF87UZrxhr4xWS0ihEUXuh
Supplier#000008595 MH0iB73GQ3z UW3O DbCbqmc
Supplier#000008610 SgVgP90vP452sUNTgzL9zKwXHXAzV6tV
Supplier#000008705 aE,trRNdPx,4yinTD9O3DebDIp
Supplier#000008742 HmPlQEzKCPEcTUL14,kKq
Supplier#000008841 I 85Lu1sekbg2xrSIzm0
Supplier#000008895 2cH4okfaLSZTTg8sKRbbJQxkmeFu2Esj
Supplier#000008967 2kwEHyMG 7FwozNImAUE6mH0hYtqYculJM
Supplier#000008972 w2vF6 D5YZO3visPXsqVfLADTK
Supplier#000009032 qK,trB6Sdy4Dz1BRUFNy
Supplier#000009147 rOAuryHxpZ9eOvx
Supplier#000009252 F7cZaPUHwh1 ZKyj3xmAVWC1XdP ue1p5m,i
Supplier#000009278 RqYTzgxj93CLX 0mcYfCENOefD
Supplier#000009327 uoqMdf7e7Gj9dbQ53
Supplier#000009430 igRqmneFt
Supplier#000009567 r4Wfx4c3xsEAjcGj71HHZByornl D9vrztXlv4
Supplier#000009601 51m637bO,Rw5DnHWFUvLacRx9
Supplier#000009709 rRnCbHYgDgl9PZYnyWKVYSUW0vKg
Supplier#000009753 wLhVEcRmd7PkJF4FBnGK7Z
Supplier#000009796 z,y4Idmr15DOvPUqYG
Supplier#000009799 4wNjXGa4OKWl
Supplier#000009811 E3iuyq7UnZxU7oPZIe2Gu6
Supplier#000009812 APFRMy3lCbgFga53n5t9DxzFPQPgnjrGt32
Supplier#000009862 rJzweWeN58
Supplier#000009868 ROjGgx5gvtkmnUUoeyy7v
Supplier#000009869 ucLqxzrpBTRMewGSM29t0rNTM30g1Tu3Xgg3mKag
Supplier#000009899 7XdpAHrzr1t,UQFZE

```
Supplier#000009974        7wJ,J5DKcxSU4Kp1cQLpbcAvB5AsvKT


204 rows processed.
Query Processed in 1.69 seconds.


-- @(#)21.sql          2.1.6.2
-- TPC-H/TPC-R Suppliers Who Kept Orders Waiting Query (Q21)
-- Functional Query Definition
-- Approved February 1998

select * from (
select
s_name,
count(*)  numwait
from
supplier,
lineitem l1,
orders,
nation
where
s_suppkey = l1.l_suppkey
and o_orderkey = l1.l_orderkey
and o_orderstatus = 'F'
and l1.l_receiptdate > l1.l_commitdate
and exists (
select
*
from
lineitem l2
where
l2.l_orderkey = l1.l_orderkey
and l2.l_suppkey <> l1.l_suppkey
)
and not exists (
select
*
from
lineitem l3
where
l3.l_orderkey = l1.l_orderkey
and l3.l_suppkey <> l1.l_suppkey
and l3.l_receiptdate > l3.l_commitdate
)
and s_nationkey = n_nationkey
and n_name = 'SAUDI ARABIA'
group by
s_name
order by
numwait desc,
s_name)
where rownum <= 100

S_NAME                 NUMWAIT
Supplier#000002829       20.00
Supplier#000005808       18.00
Supplier#000000262       17.00
Supplier#000000496       17.00
Supplier#000002160       17.00
Supplier#000002301       17.00
Supplier#000002540       17.00
```

Supplier#000003063	17.00
Supplier#000005178	17.00
Supplier#000008331	17.00
Supplier#000002005	16.00
Supplier#000002095	16.00
Supplier#000005799	16.00
Supplier#000005842	16.00
Supplier#000006450	16.00
Supplier#000006939	16.00
Supplier#000009200	16.00
Supplier#000009727	16.00
Supplier#000000486	15.00
Supplier#000000565	15.00
Supplier#000001046	15.00
Supplier#000001047	15.00
Supplier#000001161	15.00
Supplier#000001336	15.00
Supplier#000001435	15.00
Supplier#000003075	15.00
Supplier#000003335	15.00
Supplier#000005649	15.00
Supplier#000006027	15.00
Supplier#000006795	15.00
Supplier#000006800	15.00
Supplier#000006824	15.00
Supplier#000007131	15.00
Supplier#000007382	15.00
Supplier#000008913	15.00
Supplier#000009787	15.00
Supplier#000000633	14.00
Supplier#000001960	14.00
Supplier#000002323	14.00
Supplier#000002490	14.00
Supplier#000002993	14.00
Supplier#000003101	14.00
Supplier#000004489	14.00
Supplier#000005435	14.00
Supplier#000005583	14.00
Supplier#000005774	14.00
Supplier#000007579	14.00
Supplier#000008180	14.00
Supplier#000008695	14.00
Supplier#000009224	14.00
Supplier#000000357	13.00
Supplier#000000436	13.00
Supplier#000000610	13.00
Supplier#000000788	13.00
Supplier#000000889	13.00
Supplier#000001062	13.00
Supplier#000001498	13.00
Supplier#000002056	13.00
Supplier#000002312	13.00
Supplier#000002344	13.00
Supplier#000002596	13.00
Supplier#000002615	13.00
Supplier#000002978	13.00
Supplier#000003048	13.00
Supplier#000003234	13.00
Supplier#000003727	13.00
Supplier#000003806	13.00
Supplier#000004472	13.00
Supplier#000005236	13.00

```
Supplier#000005906      13.00
Supplier#000006241      13.00
Supplier#000006326      13.00
Supplier#000006384      13.00
Supplier#000006394      13.00
Supplier#000006624      13.00
Supplier#000006629      13.00
Supplier#000006682      13.00
Supplier#000006737      13.00
Supplier#000006825      13.00
Supplier#000007021      13.00
Supplier#000007417      13.00
Supplier#000007497      13.00
Supplier#000007602      13.00
Supplier#000008134      13.00
Supplier#000008234      13.00
Supplier#000009435      13.00
Supplier#000009436      13.00
Supplier#000009564      13.00
Supplier#000009896      13.00
Supplier#000000379      12.00
Supplier#000000673      12.00
Supplier#000000762      12.00
Supplier#000000811      12.00
Supplier#000000821      12.00
Supplier#000001337      12.00
Supplier#000001916      12.00
Supplier#000001925      12.00
Supplier#000002039      12.00
Supplier#000002357      12.00
Supplier#000002483      12.00


100 rows processed.
Query Processed in 13.31 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 2.61 seconds.

APPENDIX D: Seed & Query Substitution Parameters

This Appendix contains Seed values and substitution parameters for each stream.

Substitution Parameters for Stream 00 (seed :609021123)

```
14      1993-02-01
2       44      NICKEL          AMERICA
9       chartreuse
20      almond  1995-01-01      IRAQ
6       1997-01-01      0.03    25
17      Brand#11        MED CAN
18      312
8       UNITED KINGDOM  EUROPE          STANDARD BRUSHED NICKEL
21      JORDAN
13      unusual  packages
3       FURNITURE       1995-03-08
22      23      20      11      31      32      29      17
16      Brand#31        SMALL ANODIZED  25      28      22      6       45      5       44      15
4       1994-03-01
11      ETHIOPIA        0.0000001000
15      1995-12-01
1       87
10      1994-11-01
19      Brand#31        Brand#33        Brand#55        9       10      28
5       ASIA    1997-01-01
7       ETHIOPIA        UNITED KINGDOM
12      AIR     MAIL    1993-01-01
```

Substitution Parameters for Stream 01 (seed :609021124)

```
21      ETHIOPIA
3       MACHINERY       1995-03-24
18      314
5       EUROPE          1993-01-01
11      CHINA   0.0000001000
7       RUSSIAMOROCCO
6       1993-01-01      0.09    25
20      ivory   1994-01-01      ARGENTINA
17      Brand#13        JUMBO CASE
12      REG AIR         MAIL    1993-01-01
16      Brand#21        LARGE PLATED    23      21      46      11      42      38      41      20
15      1993-09-01
13      unusual  packages
10      1993-08-01
2       32      TIN     EUROPE
8       MOROCCO         AFRICA          STANDARD PLATED BRASS
14      1993-05-01
19      Brand#43        Brand#21        Brand#54        4       11      24
9       blanched
22      11      30      28      33      15      20      32
1       95
4       1996-10-01
```

Substitution Parameters for Stream 02 (seed :609021125)

6 1993-01-01 0.06 24
17 Brand#15 JUMBO JAR
14 1993-09-01
16 Brand#51 PROMO POLISHED 35 3 46 9 20 19 47 1
19 Brand#45 Brand#54 Brand#43 10 12 20
10 1994-06-01
9 antique
2 19 STEEL AMERICA
15 1996-04-01
8 GERMANY EUROPE STANDARD ANODIZED BRASS
5 MIDDLE EAST 1993-01-01
22 28 14 19 31 15 13 30
12 SHIP MAIL 1993-01-01
7 KENYA GERMANY
13 unusual packages
18 315
1 103
4 1994-06-01
20 seashell 1997-01-01 MOROCCO
3 BUILDING 1995-03-10
11 FRANCE 0.0000001000
21 RUSSIA

Substitution Parameters for Stream 03 (seed :609021126)

8 UNITED STATES AMERICA PROMO POLISHED BRASS
5 AMERICA 1993-01-01
4 1997-01-01
6 1993-01-01 0.04 25
17 Brand#12 JUMBO CAN
7 FRANCE UNITED STATES
1 111
18 313
22 25 27 21 12 22 30 33
14 1993-12-01
9 turquoise
10 1993-03-01
15 1993-12-01
11 ROMANIA 0.0000001000
20 dim 1995-01-01 ETHIOPIA
2 7 BRASS MIDDLE EAST
21 KENYA
19 Brand#42 Brand#32 Brand#43 5 13 28
13 unusual packages
16 Brand#41 MEDIUM ANODIZED 26 25 45 12 35 23 6 17
12 FOB MAIL 1993-01-01
3 HOUSEHOLD 1995-03-26

Substitution Parameters for Stream 04 (seed :609021127)

5 ASIA 1993-01-01
21 FRANCE
14 1994-03-01
19 Brand#54 Brand#25 Brand#42 10 14 24
15 1996-07-01

17 Brand#14 WRAP CASE
12 MAIL FOB 1994-01-01
6 1993-01-01 0.09 25
4 1994-10-01
9 snow
8 MOZAMBIQUE AFRICA PROMO BURNISHED BRASS
16 Brand#21 ECONOMY BURNISHED 38 34 14 11 32 31 13 10
11 GERMANY 0.0000001000
2 45 NICKEL AMERICA
10 1993-12-01
18 314
1 119
13 unusual packages
7 UNITED KINGDOM MOZAMBIQUE
22 30 12 18 15 20 19 34
3 BUILDING 1995-03-12
20 papaya 1994-01-01 SAUDI ARABIA

Substitution Parameters for Stream 05 (seed :609021128)

21 UNITED STATES
15 1994-04-01
4 1997-05-01
6 1994-01-01 0.07 24
7 MOROCCO INDIA
16 Brand#51 STANDARD POLISHED 37 12 49 44 28 7 26 6
19 Brand#51 Brand#52 Brand#31 6 15 20
18 312
14 1994-06-01
22 18 23 13 22 12 24 20
11 SAUDI ARABIA 0.0000001000
13 express requests
3 HOUSEHOLD 1995-03-28
1 66
2 33 TIN MIDDLE EAST
5 EUROPE 1994-01-01
8 INDIA ASIA ECONOMY BRUSHED BRASS
20 blue 1997-01-01 INDONESIA
12 RAIL FOB 1994-01-01
17 Brand#11 WRAP JAR
10 1994-09-01
9 sandy

Substitution Parameters for Stream 06 (seed :609021129)

10 1993-06-01
3 AUTOMOBILE 1995-03-14
15 1996-11-01
13 express requests
6 1994-01-01 0.04 24
8 ALGERIA AFRICA ECONOMY PLATED STEEL
9 red
7 GERMANY ALGERIA
4 1995-02-01
11 INDIA 0.0000001000

22 17 22 10 32 14 27 12
18 313
12 AIR FOB 1994-01-01
1 74
5 MIDDLE EAST 1994-01-01
16 Brand#41 MEDIUM BRUSHED 38 7 34 8 39 49 28 41
2 21 COPPER ASIA
14 1994-09-01
19 Brand#53 Brand#45 Brand#35 1 16 27
20 linen 1996-01-01 UNITED STATES
17 Brand#13 WRAP CAN
21 MOZAMBIQUE

Substitution Parameters for Stream 07 (seed :609021130)

18 315
8 PERU AMERICA ECONOMY ANODIZED STEEL
20 tan 1994-01-01 KENYA
21 INDIA
2 8 BRASS MIDDLE EAST
4 1997-09-01
22 24 14 30 22 17 18 28
17 Brand#15 SM BOX
1 82
11 VIETNAM 0.0000001000
9 peru
19 Brand#15 Brand#23 Brand#35 6 17 23
3 HOUSEHOLD 1995-03-30
13 express requests
5 AFRICA 1994-01-01
7 UNITED STATES PERU
10 1994-04-01
16 Brand#21 PROMO BURNISHED 2 47 17 10 21 44 23 35
6 1994-01-01 0.09 25
14 1995-01-01
15 1994-07-01
12 REG AIR FOB 1994-01-01

APPENDIX E: Implementation-Specific Layer/Driver Code

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=${OUT_DIR}/firstten
LD1DBCRE=${OUT_DIR}/Ld1dbcre
LD2SCTSO=${OUT_DIR}/Ld2sctso
LD3DAPOP=${OUT_DIR}/Ld4dapop
LD4IXCRE=${OUT_DIR}/Ld5ixcre
LD5ANLYZ=${OUT_DIR}/Ld5anlyz
DAT_FILE=${KIT_DIR}/audit/1TB.dat

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

save_alert_log.exp preAudit $RUN_ID

echo "Start: load database `date`" >> $SCRIPT_LOG_FILE
#bumpx.pl -s -o ${DAT_FILE} -p dbcre > $LD1DBCRE
#bumpx.pl -s -o ${DAT_FILE} -p sctso > $LD2SCTSO
STIME=`$GTIME`
echo "Start: timed load portion `date`" >> $SCRIPT_LOG_FILE
bumpx.pl -s -o ${DAT_FILE} -p dapop > $LD3DAPOP
bumpx.pl -s -o ${DAT_FILE} -p ixcre > $LD4IXCRE
bumpx.pl -s -o ${DAT_FILE} -p anlyz > $LD5ANLYZ
echo "End: timed load portion `date`" >> $SCRIPT_LOG_FILE

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

echo "reboot and restart the database before run `date`" >>
$SCRIPT_LOG_FILE
restart_server_db.sh
echo "server and database come up at `date`" >>
$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

#2shut >> $SCRIPT_LOG_FILE
#2start >> $SCRIPT_LOG_FILE
#ckpnt.sh
runTPCHpt ${SCALE_FACTOR} 1 ${RUN_ID}

#2shut >> $SCRIPT_LOG_FILE
#2start >> $SCRIPT_LOG_FILE
#ckpnt.sh
runTPCHpt ${SCALE_FACTOR} 2 ${RUN_ID}

sleep 600
#2shut >> $SCRIPT_LOG_FILE

save_alert_log.exp Audit $RUN_ID

echo "End TPC-H Benchmark SEQUENCE NUMBER: $RUN_ID
`date`" >> $SCRIPT_LOG_FILE
```

runTPCHpt

```
#!/bin/ksh
. $KIT_DIR/env
#set -x
#ECHO=/bin/echo
SCRIPT_DIR=${KIT_DIR}/scripts
SQL_DIR=${KIT_DIR}/sql
UPD_DIR=${KIT_DIR}/update
SRC_DIR=${KIT_DIR}/utils
QRY_DIR=${KIT_DIR}/queries # this is the location of the query
template file
QGEN_DIR=${KIT_DIR}/dbgen
QGEN=${QGEN_DIR}/qgen
QEXEC=${SRC_DIR}

DSS_QUERY=${KIT_DIR}/queries
export DSS_QUERY

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

TPCD_BIN=${KIT_DIR}/audit/bin

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

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

# The defaults

QPROG=${QEXEC}/qexec
```

```
usage () {

echo " "
echo "Usage: $0 [-p <program for query stream>] [-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;;
  -o) OSTAT=1;;
  -s) SSTAT=1;;
  -h) usage; exit 0;;
  --) shift; break;;
  esac
  shift;
done


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

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


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

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

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

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 &
sleep 5
 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
```

runTPCHus

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

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

DSS_QUERY=${KIT_DIR}/queries
export DSS_QUERY

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

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


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

```
GTIME=${SRC_DIR}/gtime
HID=1

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


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

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

 # Execute UF1

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

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

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

 # Execute UF2

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

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

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

print > /tmp/th_pipe2
```

runuf1.sh

```
#!/bin/ksh
#
# $Header: runuf1.sh 25-oct-2001.15:56:04 mpoess Exp $
#
# runuf1.sh
#
# Copyright (c) 1999, 2001, Oracle Corporation.  All rights reserved.
#
#   NAME
#     runuf1.sh - <one-line expansion of the name>
#
# DESCRIPTION
#   runuf1.sh -l [<path name for reports>] -u [<uid/passwd>]
#           -p [<program>] <run_id> <scale factor> <pair number>
#           <parallelism>
# USAGE
#   To execute UF1.
#
#   NOTES
#     <other useful comments, qualifications, etc.>
#
#   MODIFIED   (MM/DD/YY)
#   mpoess     10/25/01 - change default directory for update sets
#   mpoess     10/17/01 - add support for external tables
#   mpoess     08/15/99 - Creation
#   mpoess     08/15/99 - Creation
#
#
. $KIT_DIR/env
O=${ORACLE_HOME}
UPDATE_DIR=${KIT_DIR}/update
SCRIPT_DIR=${UPDATE_DIR}/scripts
UTILS_DIR=${KIT_DIR}/utils
LOG_DIR=${UPDATE_DIR}/log
GTIME=${UTILS_DIR}/gtime
SF=${SCALE_FACTOR}
PAR_HINT=${UPDATE_DOP_1}

LOGPATH=.
PASSWD=${DATABASE_USER}

if [ $# -lt 1 ];
then
  echo runuf1.sh setnum
  exit 1
fi
SETNUM=$1
if [ $# -eq 2 ]
then
   PAR_HINT=$2
fi

i=1
PID=""

# perform the update function 1

START=`$GTIME`

# first create the temp tables

sqlplus $PASSWD << !
spool runuf1.log
set timing on
set serveroutput on
set echo on

drop directory data_dir;
create directory data_dir as 'D:/';

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
                (
                l_orderkey,
                l_partkey,
                l_suppkey,
                l_linenumber,
                l_quantity,
                l_extendedprice,
                l_discount,
                l_tax,
                l_returnflag,
                l_linestatus,
                l_shipdate              DATE 'YYYY-MM-DD',
                l_commitdate                    DATE
'YYYY-MM-DD',
                l_receiptdate                   DATE
'YYYY-MM-DD',
                l_shipinstruct,
                l_shipmode,
                l_comment
                )
    )
    location (
    'lineitem.tbl.u${SETNUM}.1',
    'lineitem.tbl.u${SETNUM}.2',
    'lineitem.tbl.u${SETNUM}.3',
    'lineitem.tbl.u${SETNUM}.4',
    'lineitem.tbl.u${SETNUM}.5',
    'lineitem.tbl.u${SETNUM}.6',
    'lineitem.tbl.u${SETNUM}.7',
    'lineitem.tbl.u${SETNUM}.8',
    'lineitem.tbl.u${SETNUM}.9',
    'lineitem.tbl.u${SETNUM}.10',
    'lineitem.tbl.u${SETNUM}.11',
    'lineitem.tbl.u${SETNUM}.12',
        'lineitem.tbl.u${SETNUM}.13',
        'lineitem.tbl.u${SETNUM}.14',
        'lineitem.tbl.u${SETNUM}.15',
    'lineitem.tbl.u${SETNUM}.16'
))
reject limit 100;

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
                (
                o_orderkey,
                o_custkey,
                o_orderstatus,
                o_totalprice,
                o_orderdate DATE 'YYYY-MM-DD',
                o_orderpriority,
                o_clerk,
                o_shippriority,
                o_comment
                )
    )
    location (
    'orders.tbl.u${SETNUM}.1',
    'orders.tbl.u${SETNUM}.2',
    'orders.tbl.u${SETNUM}.3',
    'orders.tbl.u${SETNUM}.4',
    'orders.tbl.u${SETNUM}.5',
    'orders.tbl.u${SETNUM}.6',
    'orders.tbl.u${SETNUM}.7',
    'orders.tbl.u${SETNUM}.8',
    'orders.tbl.u${SETNUM}.9',
    'orders.tbl.u${SETNUM}.10',
    'orders.tbl.u${SETNUM}.11',
    'orders.tbl.u${SETNUM}.12',
    'orders.tbl.u${SETNUM}.13',
    'orders.tbl.u${SETNUM}.14',
    'orders.tbl.u${SETNUM}.15',
    'orders.tbl.u${SETNUM}.16'
))
reject limit 100;
alter table temp_l_et parallel ${PAR_HINT};
alter table temp_o_et parallel ${PAR_HINT};

prompt DEGRE OF PARALLELISM = ${PAR_HINT}
alter session force parallel dml parallel (degree ${PAR_HINT});
alter session set isolation_level = serializable;
alter session set optimizer_index_cost_adj = 25;

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 ;

spool off
exit;
!

END=`$GTIME`

# Done

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

#if  [ $SETNUM -eq 1 ]
#then
#       $KIT_DIR/audit/ckpnt.sh
#fi
```

runuf2.sh

```
#!/bin/ksh
#
# $Header: runuf2.sh 25-oct-2001.15:56:05 mpoess Exp $
#
# runuf2.sh
#
# Copyright (c) 1999, 2001, Oracle Corporation.  All rights reserved.
#
#    NAME
#      runuf2.sh - <one-line expansion of the name>
#
# DESCRIPTION
#    runuf2.sh [-u <uid/passwd to login>] [-p <program>] <run_id>
#              <scale factor> <pair number> <parallelism>
# USAGE
#    To execute UF2.
#
#    NOTES
#      <other useful comments, qualifications, etc.>
#
#    MODIFIED   (MM/DD/YY)
#    mpoess      10/25/01 - change default directory for update sets
#    mpoess      10/17/01 - add support for external tables
#    mpoess      08/15/99 - Creation
#    mpoess      08/15/99 - Creation
#
. $KIT_DIR/env
UPDATE_DIR=${KIT_DIR}/update
SCRIPT_DIR=${UPDATE_DIR}/scripts
UTILS_DIR=${KIT_DIR}/utils
GTIME=${UTILS_DIR}/gtime
LOG_DIR=${UPDATE_DIR}/log
PAR_HINT=${UPDATE_DOP_2}
SF=${SCALE_FACTOR}
PASSWD=${DATABASE_USER}

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

SETNUM=$1

if [ $# -eq 2 ]
then
   PAR_HINT=$2
fi

i=1
PID=""

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

sqlplus $PASSWD << !

set timing  on
set serveroutput on
set echo on

drop directory data_dir;
create directory data_dir as 'D:/';

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.u${SETNUM}.1',
       'delete.u${SETNUM}.2',
       'delete.u${SETNUM}.3',
       'delete.u${SETNUM}.4',
       'delete.u${SETNUM}.5',
       'delete.u${SETNUM}.6',
       'delete.u${SETNUM}.7',
       'delete.u${SETNUM}.8',
       'delete.u${SETNUM}.9',
       'delete.u${SETNUM}.10',
              'delete.u${SETNUM}.11',
       'delete.u${SETNUM}.12',
       'delete.u${SETNUM}.13',
       'delete.u${SETNUM}.14',
       'delete.u${SETNUM}.15',
       'delete.u${SETNUM}.16'
       ))
reject limit unlimited;

alter table temp_okey_et parallel 16;
```

```
create table temp_okey parallel 16 nologging as select * from temp_okey_et;

create unique index i_temp_okey on temp_okey (t_orderkey) parallel 16 nologging compute statistics;

alter session force parallel dml parallel ${PAR_HINT};
alter session set isolation_level=serializable;
alter session set optimizer_dynamic_sampling = 2;
alter session set optimizer_index_cost_adj = 25;

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

delete from (select /*+ ordered index(l) 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 ""

#$KIT_DIR/audit/ckpnt.sh
```

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@dbname combo
*/
char *passwd;
char *dbname;

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 and dbname*/

  passwd = strchr(logname, '/');
  *passwd = '\0';
  passwd++;
  dbname = strchr(passwd, '@');
  *dbname = '\0';
  dbname++;

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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcsrv,0,OCI_ATTR_SERVE
R,errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,logname,strlen(logname),OCI
_ATTR_USERNAME,
                errhp);

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(passwd),OCI_
ATTR_PASSWORD,
                errhp);

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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcusr,0,OCI_ATTR_SESSI
ON,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 */

    /* Fix for Q15 */
    strcpy(qnp,qn);

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

  s_tr_start = gettime();

  /* prepare the statement */

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

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

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

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


OCIaget(curq,OCI_HTYPE_STMT,&stmttyp,NULL,OCI_ATTR_ST
MT_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_R
OW_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_R
OW_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_R
OW_COUNT,errhp);
    print_rows(num,rows_ret);
  }

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

}


int get_statement()
{

  char line[128];
  char *pos, *str;

  /* Reset statement buffer */

  stmt[0] = '\0';

  while (fgets(line, 127, qtemp) != NULL) {

    /* skip blank lines */
    if (line[0] == '\n')
      continue;

    /* remove blanks */

    str = line;

    while (*str == ' ') str++;

    /* Let's get the line together first */

    strcat(stmt, str);

    /* if this is a comment line */
    if ((str[0] == '-') && (str[1] == '-'))
      return COMMENT;

    /* see if this is a set_fetchrows line */
    if (strncmp(str, "set_fetchrows", 13) == 0) {
      pos = strchr(str, ';');
      *pos = '\0';
      pos = strchr(str, '=');
      num_to_fetch = atol(++pos);
      return SET_FETCHROW;
    }

    /* if this is the end of the current statement */
    if ((pos = strchr(stmt, ';')) != NULL) {
      *pos = '\0';
      return SQL_STMT;
    }
  }
  return END_OF_FILE;
}


/* memalloc(): Allocates memory, exit program if we have a problem.
*/

void *memalloc(size)
     int size;
{

  void *tmp;

  if ((tmp = (void *) malloc(size)) == NULL) {
    fprintf(stderr, "Error in malloc\n");
    SQLexit();
    return NULL;       /* should never reach here */
  } else {
    return tmp;
  }
}



void print_header(nsel)
     int nsel;          /* Number of select list items */
{

  int i, diff;
  char colname[MAX_COLNAME_SIZE];
  int len = 0;          /* Running column length */
  int cwid = 0;

  fprintf(logfile, "\n");

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

    /* extract the column name */

    strncpy((char *)colname, (char *)slist[i].buf, slist[i].buflen);
    colname[slist[i].buflen] = '\0';

    /* format the output a little */

    cwid = MAX(slist[i].dbsize, slist[i].buflen);

    /* do a little bit of formatting */

    if (cwid > 80) {
      fprintf(logfile,"\n");
      len = 0;
    } else if ((len += cwid) > 80) {
      fprintf(logfile,"\n");
      len = cwid;
    }
#ifdef FORMAT1
    if ((slist[i].dbtype == INT_TYPE) || (slist[i].dbtype == FLT_TYPE))
      fprintf(logfile, "%*s ", cwid, slist[i].buf);
    else  /* string type */
      fprintf(logfile, "%*s ", -cwid, slist[i].buf);
#else
      fprintf(logfile, "%*s ", -cwid, colname);
#endif /* FORMAT1 */
```

```
  }

  fprintf(logfile,"\n");
}


void print_rows(ncol, nrow)
    int ncol;
    int nrow;
{

  int i,j;
  int len;
  int diff;
  int cwid;

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

    len = 0;

    for (j=0;j<ncol;j++) {

      cwid = MAX(slist[j].dbsize, slist[j].buflen);

      /* do a little bit of formatting */

      if (cwid > 80) {
                fprintf(logfile,"\n");
                len = 0;
      } else if ((len += cwid) > 80) {
                fprintf(logfile,"\n");
                len = cwid;
      }

      switch(slist[j].dbtype) {
      case INT_TYPE:
#ifdef HAVE_SCALE
                fprintf(logfile, "%*ld|", cwid,       (dlist[j]->ibuf)[i]);
                break;
#endif /* HAVE_SCALE */
      case FLT_TYPE:
#ifdef FORMAT1
                fprintf(logfile,"%*.2f ", cwid, (dlist[j]->fbuf)[i]);
#else
                fprintf(logfile,"%*.2f ", -cwid, (dlist[j]->fbuf)[i]);
#endif /* FORMAT1 */
                break;
      default:
                fprintf(logfile, "%*s ", -(cwid), (dlist[j]->sbuf)[i]);
                break;
      }
    }
    fprintf(logfile, "\n");
  }
}


/* remove_newline(): Remove newline character from str. */

void remove_newline(str)
    char *str;
{

  char *p;

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

qexecpl.h

```
/*
 * $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_D
EFAULT)) != 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 */
```

gtime.c

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

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

/*

  NAME
    gettime.c

  DESCRIPTION
    get wall clock time.
    get cpu time.

  FUNCTIONS
    get wall clock time.
    get cpu time.

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

*/


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

#define SUN_OS5

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

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

#if defined(aix) || defined(AIXRIOS)
# define TIME_W_GETTIME
# define CPU_W_TIMES
# define GETRU_STATS
#endif /* AIXRIOS */

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

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

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

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

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

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

```
*/

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

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

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

# include <stdio.h>

#ifdef GETRU_STATS
struct rusage selfru;
struct rusage kidsru;
#endif /* GETRU_STATS */

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



double gettime ()

{

#ifdef TIME_W_GETTIME
  struct timeval tv;

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

#ifdef TIME_W_TIMES
  struct tms buf;

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

}



double getcpu ()

{

#ifdef CPU_W_TIMES
  struct tms buf;

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

#ifdef CPU_W_GETRU
  struct rusage ru;
  double usecs;

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

}



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

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

{

#ifdef GETRU_STATS
  struct rusage ru;

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

#ifdef GET_P_STATS
  timeval_t tv;
  struct process_stats ru;

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

}



getru1 (kids)

int kids;

{

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

#ifdef GET_P_STATS
  timeval_t tv;

  if (kids) {
    memset (&kidsru, 0, sizeof (kidsru));
    get_process_stats (&tv, PS_SELF, (struct process_stats *) 0,
&kidsru);
  }
```

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

}



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

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

{

#ifdef GETRU_STATS
  struct rusage ru;

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

#ifdef GET_P_STATS
  timeval_t tv;
  struct process_stats ru;

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

}



#ifdef GETRU_STATS

print_ru (fp, ru)

FILE *fp;
struct rusage *ru;

{

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

}



diffru (ru2, ru)

struct rusage *ru2;
struct rusage *ru;

{

  ru2->ru_utime.tv_sec -= ru->ru_utime.tv_sec;
  ru2->ru_utime.tv_usec -= ru->ru_utime.tv_usec;
  ru2->ru_stime.tv_sec -= ru->ru_stime.tv_sec;
  ru2->ru_stime.tv_usec -= ru->ru_stime.tv_usec;
  ru2->ru_maxrss -= ru->ru_maxrss;
  ru2->ru_ixrss -= ru->ru_ixrss;
  ru2->ru_idrss -= ru->ru_idrss;
  ru2->ru_minflt -= ru->ru_minflt;
  ru2->ru_majflt -= ru->ru_majflt;
  ru2->ru_nswap -= ru->ru_nswap;
  ru2->ru_inblock -= ru->ru_inblock;
  ru2->ru_oublock -= ru->ru_oublock;
  ru2->ru_msgsnd -= ru->ru_msgsnd;
  ru2->ru_msgrcv -= ru->ru_msgrcv;
  ru2->ru_nsignals -= ru->ru_nsignals;
  ru2->ru_nvcsw -= ru->ru_nvcsw;
  ru2->ru_nivcsw -= ru->ru_nivcsw;

}

#endif /* GETRU_STATS */



#ifdef GET_P_STATS

print_ru (fp, ps)

FILE *fp;
struct process_stats *ps;

{

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

```
  fprintf (fp, "%lu ", ps->ps_pffincr);
  fprintf (fp, "%lu ", ps->ps_pffdecr);
  fprintf (fp, "%lu ", ps->ps_swap);
  fprintf (fp, "%lu ", ps->ps_syscall);
  fprintf (fp, "%lu ", ps->ps_volcsw);
  fprintf (fp, "%lu ", ps->ps_involcsw);
  fprintf (fp, "%lu ", ps->ps_signal);
  fprintf (fp, "%lu ", ps->ps_lread);
  fprintf (fp, "%lu ", ps->ps_lwrite);
  fprintf (fp, "%lu ", ps->ps_bread);
  fprintf (fp, "%lu ", ps->ps_bwrite);
  fprintf (fp, "%lu ", ps->ps_phread);
  fprintf (fp, "%lu",  ps->ps_phwrite);

}



diffru (ru2, ru)

struct process_stats *ru2;
struct process_stats *ru;

{

  ru2->ps_utime.tv_sec -= ru->ps_utime.tv_sec;
  ru2->ps_utime.tv_usec -= ru->ps_utime.tv_usec;
  ru2->ps_stime.tv_sec -= ru->ps_stime.tv_sec;
  ru2->ps_stime.tv_usec -= ru->ps_stime.tv_usec;
  ru2->ps_maxrss -= ru->ps_maxrss;
  ru2->ps_pagein -= ru->ps_pagein;
  ru2->ps_reclaim -= ru->ps_reclaim;
  ru2->ps_zerofill -= ru->ps_zerofill;
  ru2->ps_pffincr -= ru->ps_pffincr;
  ru2->ps_pffdecr -= ru->ps_pffdecr;
  ru2->ps_swap -= ru->ps_swap;
  ru2->ps_syscall -= ru->ps_syscall;
  ru2->ps_volcsw -= ru->ps_volcsw;
  ru2->ps_involcsw -= ru->ps_involcsw;
  ru2->ps_signal -= ru->ps_signal;
  ru2->ps_lread -= ru->ps_lread;
  ru2->ps_lwrite -= ru->ps_lwrite;
  ru2->ps_bread -= ru->ps_bread;
  ru2->ps_bwrite -= ru->ps_bwrite;
  ru2->ps_phread -= ru->ps_phread;
  ru2->ps_phwrite -= ru->ps_phwrite;

}

#endif /* GET_P_STATS */
```

APPENDIX F: Misc database scripts

Activity between Database Load and Run1. When the load finished, the runTPCHall script automatically selected a seed value and saved it.
The database was restarted.
Then the 2 auditor scripts count.sql and dbtables.sql were run to validate that the database structure was correct.

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;
```

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;

SELECT COUNT(*) FROM CUSTOMER;

SELECT * from customer
where c_custkey in (831,2651,2492,3784,9021,100023)
order by c_custkey;


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_LINENUMBE
R)
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;
```