HP 9000 Superdome Enterprise Server
using
HP-UX 11.i 64-bit
and
Oracle9i Database Enterprise Edition

TPC Benchmark™ H
Full Disclosure Report

Third Edition

October 29, 2002

Third Edition - October 29, 2002

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

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

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

Printed in U.S.A., October 29, 2002.

Overview

This report documents the methodology and results of the TPC Benchmark™ H test conducted on the HP 9000 Superdome Enterprise Server, in conformance with the requirements of the TPC Benchmark™ H Standard Specification, Revision 2.0. The operating system used for the benchmark was HP-UX 11.i 64-bit; the DBMS was Oracle9i .

Standard and Executive Summary Statements

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

Auditor

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

TPC Benchmark H Overview

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

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

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

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

The TPC-H operations are modeled as follows:

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

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

The minimum database required to run the benchmark holds business data from 10,000 suppliers. It contains almost ten million rows representing a raw storage capacity of about 1 GB. Compliant benchmark implementations may also use one of the larger permissible database populations (e.g. 1000 GB), as defined in Clause 4.1.3.

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

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

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

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

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

General Implementation Guidelines

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

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

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

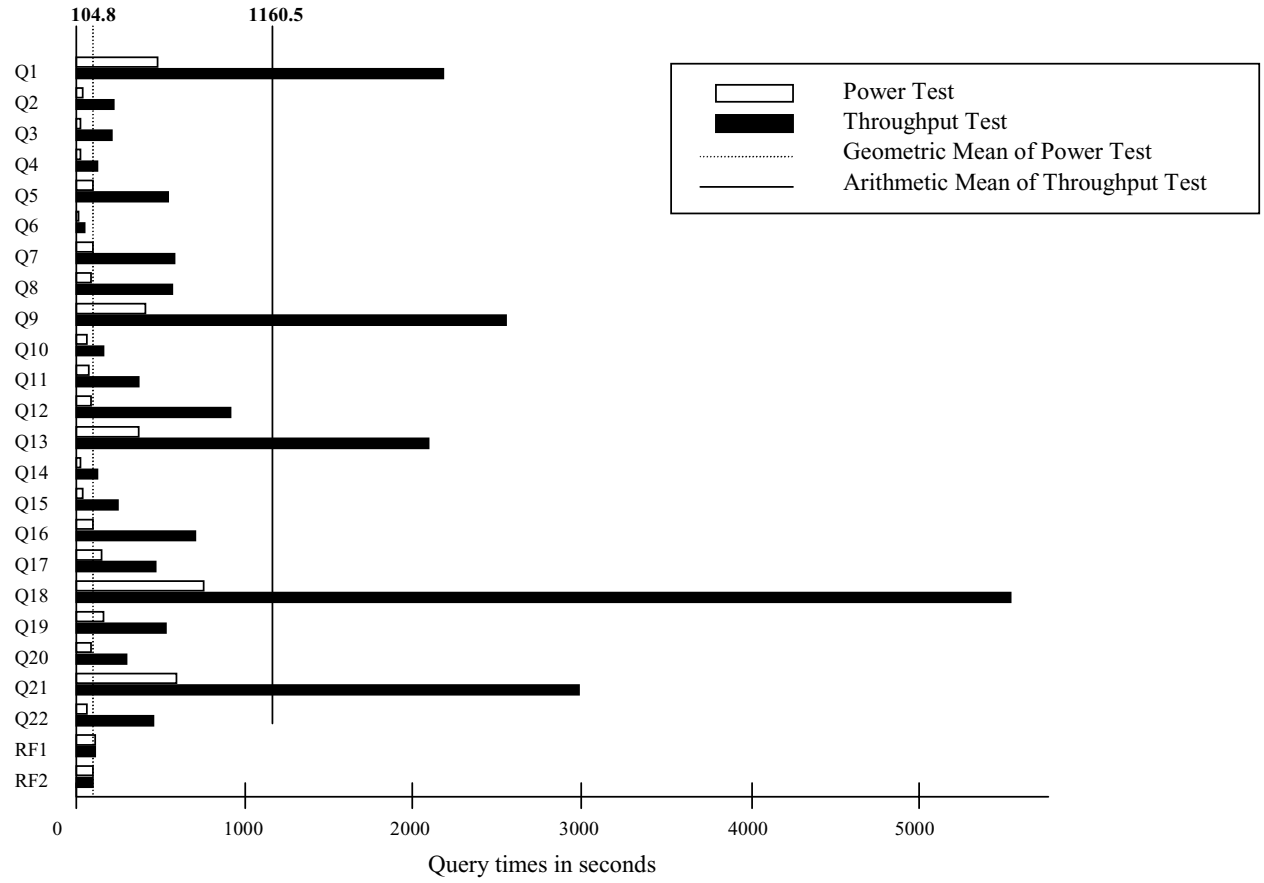
HP 9000 Superdome Enterprise Server

TPC-H Rev 2.0

Report Date: June 24, 2002
Updated October 29, 2002

Total System Cost	Composite Query per Hour Metric	Price/Performance
$5,249,167	**25,805.4** **QphH@1000GB**	**$203** **QphH@1000GB**

Database Size	Database Manager	Operating System	Other Software	Availability Date
1000 GB*	**Oracle9i Database Enterprise Edition with Partitioning v9.2.0.2.0**	**HP-UX 11.i 64-bit**	**None**	**October 30, 2002**



Database Load Time = 02:22	Load Includes Backup: N	Total Data Storage/Database Size = 22.73
RAID (Base Tables Only): N	RAID (Base Tables and Auxiliary Data Structures): N	RAID (All): Y

System Configuration

Processors:	64 PA-RISC 8700 875MHz, .75MB I-cache, 1.5MB D-cache
Memory:	128 GB
Disk Drives:	1 hp surestore disk system 2100 with 3 - 18GB Ultra 3 SCSI LVD disks and 84 SureStore VA7100 (with a total of 1260 18GB 15K RPM LVD disks)
Total Disk Storage	22734GB (In this calculation one GB is defined as 1024*1024*1024 bytes)
Lan Controllers	1 HP 10/100 Base-T Lan Adapter

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

HP 9000 Superdome Enterprise Server

TPC-H Rev 2.0

Report Date: June 24, 2002
Updated October 29, 2002

Description	Part Number	Source	Reference Price	Qty	Extended Price	3 yr. Maint. Price
Server Hardware						
Super Dome left chassis (support includes hardware, servcies, and support)	A5201A, Opt. 429	1	205,840	1	205,840	268,626
Super Dome right chassis	A5202A, Opt. 429	1	218,435	1	218,435	
Memory module - 2 GB	A5198A, Opt. 0D1	1	14,000	64	896,000	
I/O enclosures	A4856A, Opt. 0D1	1	14,805	8	118,440	
PDCA Redundant Power Source	A5800A, Opt. 0D1	1	578	2	1,156	
Cell Board with 4 PA-8700 875MHz Processors	A6862A	1	10,080	16	161,280	
ICOD right to use processor	A6885A, Opt. 104	1	23,000	64	1,472,000	403,392
Super Dome PCI Core I/O card	A6855A, Opt. 0D1	1	1,045	1	1,045	
PCI Dual FWD SCSI-2 Card	A5159A, Opt 0D1	1	1,245	2	2,490	
PCI 1000BT Lan Adapter	A4926A, Opt. 0D1	1	2,135	1	2,135	
PCI Fibre Channel 2X	A5158A, Opt 0D1	1	2,240	84	188,160	
RackInstallation Kit	A5170A. Opt 0D1	1	410	1	410	
DVD-ROM (includes SCSI-2 card, cables, enclosures)	C7499A	1	3,503	1	3,503	
HP9000 A500 Support Management Station (includes memory,CPU,lan card,disk,OS, etc)	A5570B	1	11,780	1	11,780	
hp surestore disk system 2100	A5675A	1	700	1	700	
1-18GB FWD disk module	A6537A	1	525	3	1,575	
5m SCSI-2 Cable	C2978B	1	99	1	99	
				Subtotal	**3,285,048**	**672,018**
Server Software						
Oracle9i Database Enterprise Edition Release2, Named User Plus for 3 years		2	10,000	64	640,000	
Partitioning, Named User Plus for 3 years		2	2,500	64	160,000	
Oracle Database Server Support Package for 3 years		2	6,000	1		6,000
HP-UX 11i LTU	B9088AC, Opt AAF	1		1	0	
HP-UX 11i Media	B3920EA, Opt UM9	1		1	0	
HP-UX 11i	B3920EA, Opt OD1	1		1	0	
HP-UX 11.i Factory Integration	B3920EA, Opt AAF	1	520	1	520	
				Subtotal	**800,520**	**6,000**
Storage						
Surestore VA 7100 w/dual controllers 512MB cac	A6262A	1	44,250	84	3,717,000	308,616
18GB 15K RPM FC HDD	A6191A, Opt. 0D1	1	914	1260	1,151,640	
16 meter Fibre Optic Cable	A3531AR	1	170	84	14,280	
HP9000 Std. Rack System E41	A4902A	1	1,910	7	13,370	
Modular Power Dist.	A5137AZ	1	145	28	4,060	
200-240 Volts	A5137AZ, Opt AW4	1	94	28	2,632	
				Subtotal	**4,902,982**	**308,616**
				Total	**8,988,550**	**986,634**
Oracle Mandatory E-Business Discount on Licenses and Support					(161,200)	
Large Configuration Discount and Support Prepayment*					(4,186,820)	(377,997)
				Grand Total	**4,640,530**	**608,637**
				3-yr Cost of Ownership:	**$**	**5,249,167**
2=Oracle (Pricing Contact: MaryBeth Pierantoni (See Appendix G)				**QphH@1000GB:**		**25,805.4**
*All discounts are based on US list prices and for similar quantities and configurations				**$/QphH@1000GB:**	**$**	**203**

Audited By: Brad Askins for InfoSizing (www.sizing.com)

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

HP 9000 Superdome Enterprise Server

TPC-H Rev 2.0

Report Date: June 24, 2002
Updated October 29, 2002

Measurement Results

Database Scaling (SF/size)	1000
Total Data Storage/Database Size	22.73
Start of Database Load Time	2002-06-03 21:24:20
End of Database Load Time	2002-06-03 23:46:09
Database Load Time	02:22
Query Streams for Throughput Test (S)	7
TPC-H Power	34,362.6
TPC-H Throughput	19,379.2
TPC-H Composite Query-per-Hour Metric (QphH@1000GB)	25,805.4
Total System Price Over 3 Years	$5,249,167
TPC-H Price/Performance Metric ($/QphH@1000GB)	$203

Measurement Intervals

Measurement Interval in Throughput Test (Ts)	28,608

Duration of Stream Execution:

	SEED	Start Date/Time	End Date/Time	Duration
Stream 00	603234609	06/04/2002 09:06:49	06/04/2002 10:15:51	1:09:02
Stream 01	603234610	06/04/2002 10:16:05	06/04/2002 17:09:09	6:53:04
Stream 02	603234611	06/04/2002 10:16:05	06/04/2002 17:33:08	7:17:03
Stream 03	603234612	06/04/2002 10:16:05	06/04/2002 17:26:21	7:10:16
Stream 04	603234613	06/04/2002 10:16:05	06/04/2002 17:06:01	6:49:56
Stream 05	603234614	06/04/2002 10:16:05	06/04/2002 17:49:56	7:33:51
Stream 06	603234615	06/04/2002 10:16:05	06/04/2002 16:22:17	6:06:12
Stream 07	603234616	06/04/2002 10:16:05	06/04/2002 17:22:20	7:17:15
Refresh		06/04/2002 10:16:05	06/04/2002 18.12.53	7:56:48

HP 9000 Superdome Enterprise Server

TPC-H Rev 2.0

Report Date: June 24, 2002
Updated October 29, 2002

TPC-H Timing Intervals (in seconds)

Duration of stream execution:

Stream ID	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
Stream 00	482.6	37.5	33.1	28.7	104.5	21.6	108.9	93.5	406.3	65.2	79.6	87.3
Stream 01	2189.4	284.8	188.6	79.7	468.7	44.6	548.4	404.0	2185.3	329.7	279.9	607.9
Stream 02	2573.5	297.5	94.0	216.3	535.2	86.4	572.5	492.4	2601.4	428.9	111.9	695.2
Stream 03	2357.9	233.4	50.0	121.0	513.4	78.5	623.0	233.1	2051.7	352.5	307.3	289.3
Stream 04	2286.2	300.2	167.6	190.6	831.9	49.4	389.8	447.3	2414.8	324.5	341.8	744.7
Stream 05	2408.0	193.1	217.7	120.2	441.0	56.9	498.3	340.7	1315.9	246.2	373.0	390.4
Stream 06	2174.9	230.8	217.8	128.3	552.1	53.8	584.7	573.1	2544.2	162.1	370.3	922.1
Stream 07	2219.1	308.3	219.5	246.7	293.0	22.3	324.0	532.7	4410.6	226.3	383.1	315.7
Minimum	2174.9	193.1	50.0	79.7	293.0	22.3	324.0	233.1	1315.9	162.1	111.9	289.3
Average	2315.5	264.0	165.0	157.6	519.3	56.0	505.8	431.9	2503.4	295.7	309.6	566.5
Maximum	2573.5	308.3	219.5	246.7	831.9	86.4	623.0	573.1	4410.6	428.9	383.1	922.1

Stream ID	Q13	Q14	Q15a	Q16	Q17	Q18	Q19	Q20	Q21	Q22	RF1	RF2
Stream 00	373.6	30.9	44.8	103.9	157.3	754.1	163.6	92.2	593.4	64.7	115.6	98.6
Stream 01	4877.3	136.7	452.2	696.3	514.5	5298.6	624.6	242.7	3998.6	330.9	84.8	97.2
Stream 02	6463.2	116.5	302.9	620.5	903.4	6411.9	658.1	217.0	1309.4	514.8	85.4	97.0
Stream 03	8984.6	102.6	306.5	221.6	707.8	5201.3	563.5	313.0	3070.1	333.9	99.8	96.7
Stream 04	3538.8	108.0	264.7	760.2	600.2	6171.5	625.1	279.5	3383.1	375.9	108.2	98.4
Stream 05	5278.6	161.8	294.8	1056.1	391.0	6777.6	797.0	156.5	4814.0	901.5	104.7	96.7
Stream 06	2090.5	124.6	251.2	704.6	469.2	5535.4	537.3	305.4	2979.7	459.2	109.5	97.3
Stream 07	5035.7	43.9	106.0	347.4	582.6	4976.8	761.9	340.2	4188.6	1009.8	104.7	95.9
Minimum	2090.5	43.9	106.0	221.6	391.0	4976.8	537.3	156.5	1309.4	330.9	84.8	95.9
Average	5181.2	113.4	282.6	629.5	595.5	5767.6	652.5	264.9	3391.9	560.8	99.6	97.0
Maximum	8984.6	161.8	452.2	1056.1	903.4	6777.6	797.0	340.2	4814.0	1009.8	109.5	98.4

INFO SIZING

TPC TRANSACTION PROCESSING PERFORMANCE COUNCIL CERTIFIED AUDITOR

Benchmark Sponsors:

Juergen Mueller
Performance Engineer
Hewlett Packard Company
19111 Pruneridge Avenue
Cupertino CA 95014

Ray Glasstone
Manager, DSS Performance
Oracle Corporation
100 Oracle Parkway
Redwood Shores, CA 94065

June 18, 2002

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

Platform: **HP 9000 Superdome Enterprise Server**

Database Manager**:** **Oracle 9i Database Enterprise Edition Release 2**

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

The results were:

CPU (Speed)	Memory	Disks	**QphH@1000GB**
HP 9000 Superdome Enterprise Server			
64 x PA-RISC 8700 (875 MHz)	.75MB I-cache, 1.5MB D-cache per CPU 128 GB Main	1263 x 18 GB	**25,805.4**

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:

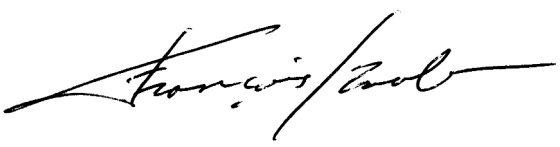
- The database records were defined with the proper layout and size
- The database population was generated using DBGEN

- The database was properly scaled to 1000GB and populated accordingly
- The compliance of the database auxiliary data structures was verified
- The database load time was correctly measured and reported
- The required ACID properties were verified and met
- The query input variables were generated by QGEN
- The query text was produced using minor modifications and one variant
- The execution of the queries against the SF1 database produced compliant answers
- A compliant implementation specific layer was used to drive the tests
- The throughput tests involved 7 query streams
- The ratio between the longest and the shortest query was such that no query timing was adjusted
- The execution times for queries and refresh functions were correctly measured and reported
- The repeatability of the measured results was verified
- The required amount of database log was configured
- The system pricing was verified for major components and maintenance
- The major pages from the FDR were verified for accuracy

Additional Audit Notes:

The measured system included 3 9GB disk drives that were substituted by 3 18GB disk drives in the priced configuration. Based on the specifications of these disks and on additional performance data collected on these disks, it is my opinion that this substitution does not have a material effect on the reported performance

Respectfully Yours,



François Raab, President

Bradley J. Askins, Auditor

1 General Items

1.1 Benchmark Sponsor

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

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

1.2 Parameter Settings

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

Database Tuning Options

Optimizer/Query execution options

Query processing tool/language configuration parameters

Recovery/commit options

Consistency/locking options

Operating system and configuration parameters

Configuration parameters and options for any other software component incorporated into the pricing structure;

Compiler optimization options.

Appendix A contains the HP-UX and Oracle9i Database Enterprise Edition with Partitioning 9.2.0.2.0 parameters used in this benchmark.

1.3 Configuration Diagrams

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

Measured Configuration:

- 64 875MHz PA-RISC 8700 CPUs each with .75MB I-cache, 1.5MB D-cache.
- 128 GB Memory
- 95 PCI Fibre Channel 2X Card
- 1 I/O Expansion Cabinet
- 1 HP 1000 BaseSX PCI Lan Adapter
- 84 SureStore VA7100 (with a total of 1260 18GB Disks)
- 1 High Availability Storage System (with a total of 3 9 GB Ultra 3 SCSI LVD Disks)
- 1 DVD ROM
- 2 SCSI Cards

Priced Configuration:

- 64 875MHz PA-RISC 8700 CPUs each with .75MB I-cache, 1.5MB D-cache.
- 128 GB Memory
- 84 PCI Fibre Channel 2X Card
- 1 HP 1000 BaseSX PCI Lan Adapter
- 84 SureStore VA7100 (with a total of 1260 18GB Disks)
- 1 hp surestore disk system 2100 (with a total of 3 18 GB Ultra 3 SCSI LVD Disks)

- 1 DVD ROM
- 2 SCSI Cards
-
-
- The difference between measured and priced was 95 PCI Fibre Channel 2Xcards were on the system but only 84 were connected to the 84 arrays and used. Also an I/O expansion cabinet was attached to the system but was not used as it had no arrays connected to it.

Measured Configuration

Terminal

Keyboard

Mouse

HP 9000 Superdome Enterprise Server

64 - 875MHz

PA-RISC 8700 Processors

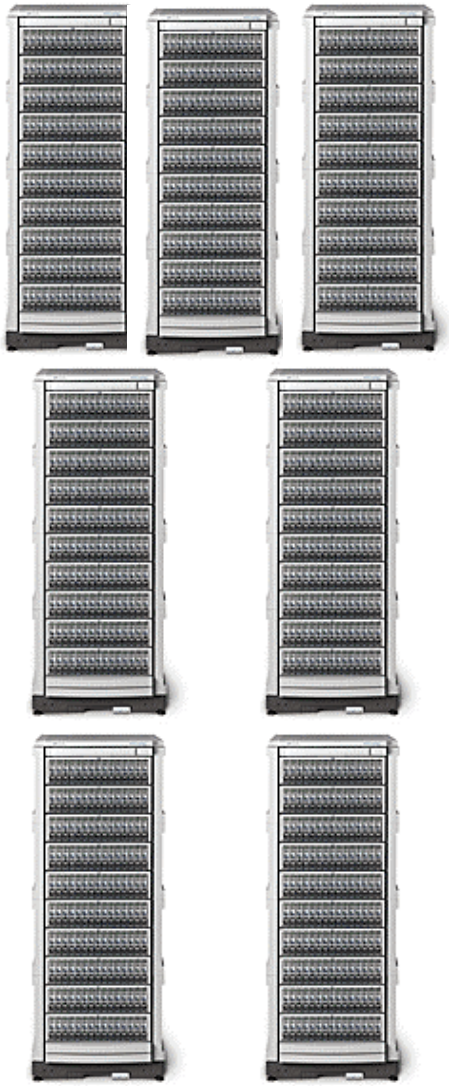
128GB Memory

.75MB I-Cache, 1.5M D-Cache

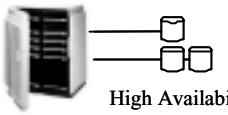
84 HP Surestore Virtual Array 7100

- 1260 18GB 15K RPM Disk

95 Fiber Channel Adapters



Additional I/O Chassis



High Availability Storage System
3 9.1GB LVD Ultra2-SCSI Disk

SCSI

LAN

SCSI

HP DVD-ROM

Terminal

Keyboard

Mouse

Priced Configuration

HP 9000 Superdome Enterprise Server

84 Fibre Channel Adapter Cards

64 - 875MHz

PA-RISC 8700 Processors

128GB Memory

.75MB I-Cache, 1.5M D-Cache

84 HP Surestore Virtual Array 7100

- 1260 18GB 15K RPM Disk

84Fiber Channel Adapters

1 hp surestore disk system 2100

With 3 – 9.1GB Disks

LAN

SCSI

SCSI

HP DVD-ROM

2 Clause 1 Logical Database Design Related Items

2.1 Database Definition Statements

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

Appendix B describes the scripts that define, create, and analyze the tables and indices for the TPC-H database.

2.2 Physical Organization

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

No record clustering or index clustering was used. Columns were reordered in the tables – please refer to the table create statements for the ordering.

2.3 Horizontal Partitioning

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

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

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

2.4 Replication

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

No replication was used.

3 Clause 2 Queries and Refresh Functions

3.1 Query Language

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

SQL was the query language used to implement all queries.

3.2 Verifying Method for Random Number Generation

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

TPC supplied versions 1.3.0 of DBGEN and QGEN were used for this TPC-H benchmark.

3.3 Generating Values for Substitution Parameters

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

QGEN version 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 definition or approved variants in order to obtain executable query text, these modifications must be disclosed and justified. The justification for a particular minor query modification can apply collectively to all queries for which it has been used. The output data for the power and throughput tests must be made available electronically upon request.

- Appendix C contains the actual query text and query output.

3.5 Query Substitution Parameters and Seeds Used

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

Appendix E contains the seed and query substitution parameters.

3.6 Query Isolation Level

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

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

3.7 Source Code of Refresh Functions

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

The refresh function is part of the implementation-specific layer/driver code included in Appendix F.

4 Clause 3 Database System Properties

4.1 ACID Properties

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

Source code for ACID test is included in Appendix C.

4.2 Atomicity

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

Completed Transaction

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

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

Aborted Transaction

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

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

4.3 Consistency

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

Consistency Test

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

1. The consistency of the ORDERS and LINEITEM tables was verified based on a sample of order keys.
2. 100 ACID Transactions were submitted from each of 8 execution streams.
3. The consistency of the ORDERS and LINEITEM tables was re-verified.

4.4 Isolation

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

Read-Write Conflict with Commit

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

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

Read-Write Conflict with Rollback

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

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

Write-Write Conflict with Commit

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

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

Write-Write Conflict with Rollback

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

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

Concurrent Progress of Read and Write on Different Tables

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

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

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

Read-Only Query Conflict with Update Transactions

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

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

4.5 Durability

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

Failure of a Durable Medium

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

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

System Crash

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

The system crash and memory failure tests were combined. Power to the servers was turned off during the durability test. When power was restored, the system rebooted and the database was restarted. The durability success file and the HISTORY table were compared and the counts matched.

Memory Failure

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

See the previous section.

5 Clause 4 Scaling and Database Population

5.1 Ending Cardinality of Tables

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

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

5.2 Distribution of Tables and Logs Across Media

On each VA7100 array, several LUNs were created in RAID 1/0 mode.

These LUNs were one each for:

- flat-files
- swap
- lineitem + other tables
- partsupp table
- custkey index
- okey index
- lkey index
- temp
- misc (logs/sys/default, etc).

12 logical volumes were created across the 84 arrays with 7 LUNs in each volume group. Each of these volume groups was divided into 8 lvols. For lineitem table, 7 of these lvols from each of the 12 volume groups was used as an independent Oracle tablespace (84 in all).

The remaining 8th lvol from each of the 12 volume groups was used as an Oracle tablespace for loading orders and other tables (12 tablespaces total).

For partsupp and indexes, 12 volume groups each across 7 LUNs were created. A single Oracle tablespace with 12 datafiles each (one from each volume group) was used for partsupp table. A similar configuration was used for the indexes, across all arrays. 1 volume group was used for temp space. This was split into 70 datafiles in one tablespace. Sys/default/undo/logs were also created out of this tablespace.

5.3 Database Partition/Replication Mapping

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

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

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

5.4 RAID Feature

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

RAID1/0 was used for log, data, temp, index, and all storage disks.

5.5 DBGEN Modification

Any modifications to the DBGEN (see clause 4.2.1) source code must be disclosed. In the event that a program other than DBGEN was used to populate the database, it must be disclosed in its entirety.

The supplied DBGEN version 1.3.0 was not modified to generate the database population for this benchmark.

5.6 Database Load Time

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

The database load time was 02:22

5.7 Data Storage Ratio

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

The data storage ratio is computed from the following information:

Type	Quantity	Disk Size	Total
1 hp surestore disk system 2100	3	18	54
84 SureStore VA7100	1260	18	22,680.0
TOTAL			**22,734.0**
Scale Factor			**1,000**
Storage Ratio			**22.73**

5.8 Database Load Mechanism Details and Illustration

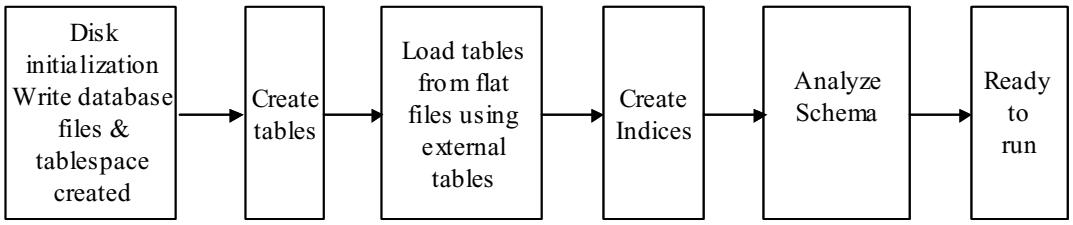
The details of the database load must be described, including a block diagram illustrating the overall process.

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

5.9 Qualification Database Configuration

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

The qualification database used identical scripts to create and load the data with changes to adjust for the database scale factor.


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

6 Clause 5 Performance Metrics and Execution-Rules

6.1 System Activity Between Load and Performance Tests

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

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

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

The database was restarted.

All scripts and queries used are included in Appendix E.

6.2 Steps in the Power Test

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

The following steps were used to implement the power test:

1. Database started
2. RF1 Refresh Transaction
3. Stream 00 Execution
4. RF2 Refresh Transaction

6.3 Timing Intervals for Each Query and Refresh Functions

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

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

6.4 Number of Streams for the Throughput Test

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

7 streams were used for the throughput test.

6.5 Start and End Date/Time of Each Query Stream

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

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

6.6 Total Elapsed Time of the Measurement Interval

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

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

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

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

The start and finish time for each refresh function in the refresh stream are given in the Numerical Quantities Summary earlier in this document.

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

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

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

6.9 Performance Metrics

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

The performance metrics, and the numbers, on which they are based, is given in the Numerical Quantities Summary earlier in this document.

6.10 The Performance Metric and Numerical Quantities from Both Runs

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

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

	QppH@1000GB	QthH@1000GB	QphH@1000GB
Reported Run	34362.6	19379.2	25805.4
Reproducibility Run	34391.2	19632.4	25984.3
% Difference	0.1%	1.3%	0.7%

6.11 System Activity Between Performance Tests

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

The database was restarted between the two runs.

7 Clause 6 SUT and Driver Implementation Related Items

7.1 Driver

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

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

For each power-test run:

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

7.2 Implementation-Specific Layer (ISL)

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

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

7.3 Profile-Directed Optimization

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

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

8 Clause 7 Pricing

8.1 Hardware and Software Used in the Priced System

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

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

8.2 Total Three Year Price

The total 3-year price of the entire configuration must be reported including: hardware, software, and maintenance charges. Separate component pricing is recommended. The basis of all discounts used must be disclosed.

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

8.3 Availability Date

The committed delivery date for general availability of products used in the priced calculations must be reported. When the priced system includes products with different availability dates, the reported availability date for the priced system must be the date at which all components are committed to be available.

Availability Dates:

Server Hardware	June 24, 2002
Server Software	Available Now
Storage	Available Now
Database Manager (Oracle9i Database Enterprise Edition with Partitioning 9.2.0.2.0)	October 30, 2002

9 Clause 8 Auditor's Information and Attestation Letter

9.1 Auditor's Report

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

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

Brad Askins
InfoSizing
18 Varick Road
(617) 984-8477
(650) 949-1352

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

10 Report Availability

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

Transaction Processing Performance Council
404 Balboa Street
San Francisco, CA 94118
Voice: 415-750-8260
Fax: 415-751-4829

or your local Hewlett-Packard sales office

Appendix A Parameter Settings

A.1 init_build.ora

```
audit_trail                     = FALSE
compatible                      = 9.0.0.0
control_files                   =
(/dbms/oracle9i/dbs/control1,/dbms/oracle9i/dbs/
control2)
db_block_buffers                = 10000
db_block_size                   = 8192
db_file_multiblock_read_count   = 32
db_files                        = 1200
db_name                         = 1tb
db_writer_processes             = 4
dml_locks                       = 40000
enqueue_resources               = 40000
global_names                    = FALSE
instance_name                   = tpch
large_pool_size                     =
2000000000
max_dump_file_size              = 25000
nls_date_format                 = YYYY-MM-DD
open_cursors                    = 1024
optimizer_mode                  = CHOOSE
parallel_execution_message_size = 16384
parallel_max_servers            = 512
parallel_min_servers            = 32
sort_area_size                = 30000000
hash_area_size                = 70000000
processes                       = 800
max_rollback_segments         = 516
sessions                        = 800
shared_pool_size                = 200000000
transaction_auditing            = FALSE
transactions                    = 512
undo_management               = auto
```

A.2 init_run.ora

```
fast_start_io_target              = 61000
statistics_level                  = BASIC
audit_trail                       = FALSE
compatible                        = 9.0.0.0
control_files                     =
(/dbms/oracle9i/dbs/control1,/dbms/oracle9i/dbs/
control2)
cpu_count                       = 64
db_block_checksum               = false
db_block_size                     = 8192
db_cache_size                    = 12g

db_file_multiblock_read_count     = 256
db_files                          = 1200
db_name                           = 1tb
db_writer_processes               = 10
dml_locks                         = 40000
enqueue_resources                 = 40000
global_names                      = FALSE
instance_name                     = tpch
large_pool_size                       =
3000000000
log_buffer                        = 33554432
log_checkpoints_to_alert          = true
max_dump_file_size                = 25000
max_rollback_segments             = 516
nls_date_format                   = YYYY-MM-DD
open_cursors                      = 1024
optimizer_features_enable       = 9.2.0.1
optimizer_index_cost_adj          = 50
optimizer_mode                    = CHOOSE
parallel_adaptive_multi_user      = TRUE
parallel_automatic_tuning       = TRUE
parallel_execution_message_size   = 16384
parallel_max_servers              = 880
parallel_min_servers              = 800
parallel_threads_per_cpu          =3
pga_aggregate_target            = 90g
processes                         = 1500
query_rewrite_enabled             = true
recovery_parallelism              = 32
replication_dependency_tracking   = false
sessions                          = 800
shared_pool_size                  = 200000000
transaction_auditing              = FALSE
transactions                      = 512
undo_management                   = auto
```

A.3 system

```
*************************************
* Source: /ux/core/kern/filesets.info/CORE-
KRN/generic
* @(#) vw: -f    selectors: --  ph_ic20_i80
'CUPI80_IC20' 'BE11.11_IC20'
*
*************************************
* Additional drivers required in every machine-
type to create a complete
* system file during cold install.  This list is
every driver that the
* master.d/ files do not force on the system or
is not identifiable by
* ioscan.
* Other CPU-type specific files can exist for
their special cases.
* see create_sysfile (1m).
*************************************
*
* Drivers/Subsystems
sba
lba
asio0
c720
sctl
sdisk
cdfs
cxperf
olar_psm
olar_psm_if
dev_olar
hd_fabric
diag0
diag1
diag2
dmem
dev_config
nfs_core
nfs_client
nfs_server
btlan
maclan
dlpi
token_arp
```

```
inet
uipc
tun
telm
tels
netdiag1
nms
hpstreams
clone
strlog
sad
echo
sc
timod
tirdwr
pipedev
pipemod
ffs
ldterm
ptem
pts
ptm
pckt
fddi4
gelan
GSCtoPCI
vxfs
vxportal
lvm
lv
nfsm
rpcmod
autofsc
cachefsc
cifs
td
dump lvol
asyncdsk
coke
*
STRMSGSZ              65535
maxvgs              99
nstrpty             60
maxfiles            4096
maxfiles_lim        4096
nflocks             2048
bufpages            50000
eqmemsize           512
maxusers            32
maxuprc             2048
max_async_ports     2048
nproc               3000
nhtbl_scale         1
swchunk             16384
maxswapchunks       32768
nswapdev            40
nfile               400000
ninode              120000
npty                20
shmmni              2048
semmni             4000
semmns              2048
semmnu              2048
semvmx              32768
shmmax             0x40000000000
shmseg             200
maxssiz             0x10000000
maxdsiz           0x40000000
timezone            480
msgmni              100
msgseg              2048
msgtql              100
unlockable_mem      1
swapmem_on          0
maxdsiz_64bit    0x80000000
maxssiz_64bit    0x80000000
```

A.4 profile

```
stty erase "^H" kill "^x" intr "^C" eof "^D"
susp "^z"
export EDITOR=/usr/bin/vi
export ORACLE_HOME=/dbms/oracle9i/oracle9i_0322
export ORACLE_SID=tpch
#export ORACLE_SID=1g
echo 'ORACLE_SID is tpch'
#echo 'ORACLE_SID is 1g'
export
SHLIB_PATH=/dbms/oracle9i/oracle9i_0322/lib32:/d
bms/oracle9i/oracle9i_0322/lib:/dbms/oracle9i/or
acle9i_0322/rdbms/lib:/dbms/oracle9i/oracle9i_03
22/network/lib
export
LD_LIBRARY_PATH=/dbms/oracle9i/oracle9i_0322/lib
64:/dbms/oracle9i/oracle9i_0322/rdbms/lib:/dbms/
oracle9i/oracle9i_0322/network/lib64
export SAVEHIST=2049
export FRAME_PATH=/dbms/oracle9i/frame
export O=/dbms/oracle9i/oracle9i_0322
export ORACLE_PATH=/dbms/oracle9i/frame/tools
export PS1="`whoami`-(hpgsp15)> "

export
PATH=./:/dbms/oracle9i/oracle9i_0322/:/dbms/orac
le9i/oracle9i_0322/lib:/dbms/oracle9i/oracle9i_0
322/bin:/dbms/oracle9i/frame/bin:/dbms/oracle9i/
frame:/dbms/oracle9i/tools/bin:/tools/Tusc:/dbms
/tpcd_v8/bumpx/bumpx:/dbms/tpcd_v8/bumpx/dbgen:/
dbms/tpcd_v8/out/scripts:/opt/ansic/bin:/opt/lan
gtools/bin:/sbin:/usr/sbin:.:/bin:/usr/bin:/usr/
local/bin:/usr/contrib/bin:/etc:/usr/include:/db
ms/oracle9i/kit:/dbms/oracle9i/kit/bumpx:/dbms/o
racle9i/local/TestIO
alias tpch="cd
/dbms/oracle9i/kit/schema/8.1.7/unix/1TB"

alias ltt="ls -ltr |tail -30"
alias cd_h="cd /dbms/oracle9i/kit/schema/9i"
alias ltm="ls -lt |more"
alias pso="ps -ef | grep ora | grep -v sleep"
alias pso_hc="ps -fu oracle | sort -n -k2"
alias setterm="TERM=dtterm;export TERM"
alias taillog="tail -f
/dbms/oracle9i/oracle9i_0322/rdbms/log/alert_tpc
h.log"

umask 002
export KIT_DIR=/dbms/oracle9i/kit
```

Appendix B Build Programs and Scripts

B.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 / 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;


*load3
\echo '#!/bin/csh' >
/export/home/oracle/kit/load/scripts/load1_*gete
nv(BUMPX_CTR).sh
\echo 'setenv ORACLE_HOME
/export/home/oracle/oracle817' >>
/export/home/oracle/kit/load/scripts/load1_*gete
nv(BUMPX_CTR).sh
\echo 'setenv ORACLE_SID  inst2' >>
/export/home/oracle/kit/load/scripts/load1_*gete
nv(BUMPX_CTR).sh
\echo 'setenv PATH
${PATH}:${ORACLE_HOME}/rdbms/bin:${ORACLE_HOME}/
bin' >>
/export/home/oracle/kit/load/scripts/load1_*gete
nv(BUMPX_CTR).sh
\echo 'setenv LD_LIBRARY_PATH
${ORACLE_HOME}/rdbms/lib:${ORACLE_HOME}/lib:"$LD
_LIBRARY_PATH"' >>
/export/home/oracle/kit/load/scripts/load1_*gete
nv(BUMPX_CTR).sh
\echo "sqlldr {}" >>
/export/home/oracle/kit/load/scripts/load1_*gete
nv(BUMPX_CTR).sh
\chmod a+x
/export/home/oracle/kit/load/scripts/load1_*gete
nv(BUMPX_CTR).sh
\rcp
/export/home/oracle/kit/load/scripts/load1_*gete
nv(BUMPX_CTR).sh
rep2:/export/home/oracle/kit/load/scripts/
\rsh -n rep2
/export/home/oracle/kit/load/scripts/load1_*gete
nv(BUMPX_CTR).sh



*load1
\sqlldr {}

*load2
\sqlldr {}


*mknod
\mknod {}

*dbgen
\dbgen {}

*sh
\{}

%e-preproc
%b-dbcre
*bgon=1
################################################
############################
# Database Creation Phase
*sql
{
shutdown abort;
}
*wait
# creating database  and undo tablespace
*sql
{
startup pfile=
/dbms/oracle9i/oracle9i_0829/dbs/init_build.ora
nomount;
create database
  controlfile reuse
  logfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/log1
' size 10558m reuse,

'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/log2
' size 10558m reuse
  datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/sys_
1' size 1000m reuse
  undo tablespace ts_undo
       datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/undo
_1' size 2600m reuse,

'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/undo
_2' size 2600m reuse,

'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/undo
_3' size 2600m reuse,

'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/undo
_4' size 2600m reuse,

'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/undo
_5' size 2600m reuse,

'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/undo
_6' size 2600m reuse,

'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/undo
_7' size 2600m reuse,

'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/undo
_8' size 2600m reuse
  maxdatafiles 4000
  maxinstances 1
;
}
*wait
# building data dictionary
*sql
```

```
{
set termout off
set echo off
spool /tmp/cat
@?/rdbms/admin/catalog.sql;
@?/rdbms/admin/catparr.sql;
@?/rdbms/admin/catproc.sql;
connect system/manager
@?/rdbms/admin/utlxplan.sql;
connect system/manager
@/dbms/oracle8/sqlplus/admin/pupbld.sql;
spool off
}
*wait
*bgoff
%e-dbcre
%b-sctso
*bgon=200
################################################
############################
# 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
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/defa
ult_1' size 760m reuse
extent management local
autoallocate
;
}
*sql
{
drop tablespace ts_l1 including contents;
create tablespace ts_l1
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_1' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l2 including contents;
create tablespace ts_l2
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_2' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l3 including contents;
create tablespace ts_l3
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_3' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l4 including contents;
create tablespace ts_l4
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_4' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l5 including contents;
create tablespace ts_l5
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_5' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l6 including contents;
create tablespace ts_l6
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_6' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l7 including contents;
create tablespace ts_l7
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_7' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l8 including contents;
create tablespace ts_l8
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_8' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l9 including contents;
create tablespace ts_l9
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_9' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l10 including contents;
create tablespace ts_l10
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_10' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l11 including contents;
create tablespace ts_l11
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_11' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l12 including contents;
create tablespace ts_l12
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_12' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l13 including contents;
create tablespace ts_l13
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_13' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l14 including contents;
create tablespace ts_l14
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_14' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l15 including contents;
create tablespace ts_l15
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_15' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l16 including contents;
create tablespace ts_l16
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_16' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l17 including contents;
create tablespace ts_l17
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_17' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l18 including contents;
create tablespace ts_l18
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_18' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l19 including contents;
create tablespace ts_l19
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_19' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l20 including contents;
create tablespace ts_l20
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_20' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l21 including contents;
create tablespace ts_l21
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_21' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l22 including contents;
create tablespace ts_l22
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_22' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l23 including contents;
create tablespace ts_l23
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_23' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l24 including contents;
create tablespace ts_l24
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_24' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l25 including contents;
create tablespace ts_l25
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_25' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l26 including contents;
create tablespace ts_l26
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_26' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l27 including contents;
create tablespace ts_l27
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_27' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l28 including contents;
create tablespace ts_l28
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_28' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l29 including contents;
create tablespace ts_l29
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_29' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l30 including contents;
create tablespace ts_l30
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_30' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l31 including contents;
create tablespace ts_l31
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_31' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l32 including contents;
create tablespace ts_l32
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_32' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l33 including contents;
create tablespace ts_l33
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_33' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l34 including contents;
create tablespace ts_l34
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_34' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l35 including contents;
create tablespace ts_l35
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_35' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l36 including contents;
create tablespace ts_l36
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_36' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l37 including contents;
create tablespace ts_l37
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_37' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l38 including contents;
create tablespace ts_l38
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_38' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l39 including contents;
create tablespace ts_l39
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_39' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l40 including contents;
create tablespace ts_l40
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_40' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l41 including contents;
create tablespace ts_l41
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_41' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l42 including contents;
create tablespace ts_l42
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_42' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l43 including contents;
create tablespace ts_l43
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_43' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l44 including contents;
create tablespace ts_l44
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_44' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l45 including contents;
create tablespace ts_l45
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_45' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l46 including contents;
create tablespace ts_l46
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_46' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l47 including contents;
create tablespace ts_l47
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_47' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l48 including contents;
create tablespace ts_l48
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_48' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l49 including contents;
create tablespace ts_l49
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_49' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l50 including contents;
create tablespace ts_l50
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_50' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l51 including contents;
create tablespace ts_l51
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_51' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l52 including contents;
create tablespace ts_l52
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_52' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l53 including contents;
create tablespace ts_l53
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_53' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l54 including contents;
create tablespace ts_l54
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_54' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l55 including contents;
create tablespace ts_l55
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_55' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l56 including contents;
create tablespace ts_l56
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_56' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l57 including contents;
create tablespace ts_l57
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_57' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l58 including contents;
create tablespace ts_l58
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_58' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l59 including contents;
create tablespace ts_l59
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_59' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l60 including contents;
create tablespace ts_l60
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_60' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l61 including contents;
create tablespace ts_l61
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_61' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l62 including contents;
create tablespace ts_l62
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_62' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l63 including contents;
create tablespace ts_l63
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_63' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l64 including contents;
create tablespace ts_l64
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_64' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l65 including contents;
create tablespace ts_l65
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_65' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l66 including contents;
create tablespace ts_l66
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_66' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l67 including contents;
create tablespace ts_l67
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_67' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l68 including contents;
create tablespace ts_l68
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_68' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l69 including contents;
create tablespace ts_l69
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_69' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l70 including contents;
create tablespace ts_l70
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_70' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l71 including contents;
create tablespace ts_l71
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_71' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l72 including contents;
create tablespace ts_l72
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_72' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l73 including contents;
create tablespace ts_l73
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_73' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l74 including contents;
create tablespace ts_l74
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_74' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l75 including contents;
create tablespace ts_l75
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_75' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l76 including contents;
create tablespace ts_l76
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_76' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l77 including contents;
create tablespace ts_l77
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_77' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l78 including contents;
create tablespace ts_l78
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_78' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l79 including contents;
create tablespace ts_l79
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_79' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l80 including contents;
create tablespace ts_l80
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_80' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l81 including contents;
create tablespace ts_l81
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_81' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_l82 including contents;
create tablespace ts_l82
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_82' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l83 including contents;
create tablespace ts_l83
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_83' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_l84 including contents;
create tablespace ts_l84
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/line
_84' size 11000m reuse
extent management dictionary default storage
(initial 640m next 20m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o1 including contents;
create tablespace ts_o1
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
1' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o2 including contents;
create tablespace ts_o2
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
2' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o3 including contents;
create tablespace ts_o3
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
3' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_o4 including contents;
create tablespace ts_o4
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
4' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o5 including contents;
create tablespace ts_o5
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
5' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o6 including contents;
create tablespace ts_o6
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
6' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o7 including contents;
create tablespace ts_o7
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
7' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o8 including contents;
create tablespace ts_o8
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
8' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o9 including contents;
create tablespace ts_o9
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
9' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
```

```
*sql
{
drop tablespace ts_o10 including contents;
create tablespace ts_o10
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
10' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o11 including contents;
create tablespace ts_o11
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
11' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_o12 including contents;
create tablespace ts_o12
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/ord_
12' size 22000m reuse
extent management dictionary default storage
(initial 130m next 10m maxextents unlimited
pctincrease 0)
;
}
*sql
{
drop tablespace ts_okey including contents;
create tablespace ts_okey
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_1' size 4100m reuse
extent management local
uniform size 100M
;
}
*sql
{
drop tablespace ts_custkey including contents;
create tablespace ts_custkey
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_1' size 1340m reuse
extent management local
uniform size 60M
;
}
*sql
{
drop tablespace ts_lokey including contents;
create tablespace ts_lokey
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_1' size 14200m reuse
extent management local
uniform size 100M
;
}
*sql
{
drop tablespace ts_psupp1 including contents;
create tablespace ts_psupp1
datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_1' size 16000m reuse
extent management dictionary default storage
(initial 1100m next 50m maxextents unlimited
pctincrease 0)
;
}
# creating tpch's ts_temp tablespace
*sql
{
drop tablespace ts_temp including contents;
create temporary tablespace ts_temp
tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_1' size 13400m reuse
extent management local
uniform size 100M
;
}
*wait
# adding tablespace datafiles
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_2' size 16000m reuse
;
}
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_3' size 16000m reuse
;
}
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_4' size 16000m reuse
;
}
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_5' size 16000m reuse
;
}
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_6' size 16000m reuse
;
}
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_7' size 16000m reuse
;
}
*sql
```

```
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_8' size 16000m reuse
;
}
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_9' size 16000m reuse
;
}
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_10' size 16000m reuse
;
}
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_11' size 16000m reuse
;
}
*sql
{
alter tablespace ts_psupp1
add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/psup
p_12' size 16000m reuse
;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_2' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_3' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_4' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_5' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_6' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_7' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_8' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_9' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_10' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_11' size 4100m reuse;
}
*sql
{
alter tablespace ts_okey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/okey
_12' size 4100m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_2' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_3' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_4' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
```

```
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_5' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_6' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_7' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_8' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_9' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_10' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_11' size 1340m reuse;
}
*sql
{
alter tablespace ts_custkey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/cust
key_12' size 1340m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_2' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_3' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_4' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_5' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_6' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_7' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_8' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_9' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_10' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_11' size 14200m reuse;
}
*sql
{
alter tablespace ts_lokey
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/loke
y_12' size 14200m reuse;
}
*sql
{
alter tablespace ts_default
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/defa
ult_2' size 760m reuse;
}
*sql
{
alter tablespace ts_default
```

```
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/defa
ult_3' size 760m reuse;
}
*sql
{
alter tablespace ts_default
  add datafile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/defa
ult_4' size 760m reuse;
}
#adding tpch's ts_temp datafiles
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_2' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_3' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_4' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_5' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_6' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_7' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_8' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_9' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_10' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_11' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_12' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_13' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_14' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_15' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_16' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_17' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_18' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_19' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
```

```
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_20' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_21' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_22' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_23' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_24' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_25' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_26' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_27' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_28' size 13400m  reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_29' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
```

```
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_30' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_31' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_32' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_33' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_34' size 13400m  reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_35' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_36' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_37' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_38' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_39' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
```

```
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_40' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_41' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_42' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_43' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_44' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_45' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_46' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_47' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_48' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_49' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
```

```
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_50' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_51' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_52' size 13400m  reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_53' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_54' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_55' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_56' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_57' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_58' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_59' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
```

```
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_60' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_61' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_62' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_63' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_64' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_65' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_66' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_67' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_68' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_69' size 13400m reuse;
}
*sql
{
alter tablespace ts_temp
  add tempfile
'/dbms/oracle9i/oracle9i_0829/dbs/datafiles/temp
_70' size 13400m reuse;
}
*wait
*bgoff
%e-sctso
%b-dapop
*bgon=1
################################################
############################
# Schema Creation Phase - User and Tables
#  AND  Database Population Phase
#
# creating tpch user
*sql
{
shutdown abort;
}
*wait
*sql
{
startup
pfile=/dbms/oracle9i/oracle9i_0829/dbs/init_run.
ora
}
*wait
*sql
{
#drop user tpch cascade;
grant DBA
to tpch identified by tpch;
}
*wait
*sql
{
connect tpch/tpch;
drop directory data_dir;
create directory data_dir as '/flat1';
}
*sql
{
connect tpch/tpch;
drop table l_et;
create table l_et(
    l_shipdate          date ,
    l_orderkey          number ,
    l_discount          number ,
    l_extendedprice     number ,
    l_suppkey           number ,
    l_quantity          number ,
    l_returnflag        char(1) ,
    l_partkey           number ,
    l_linestatus        char(1) ,
    l_tax               number ,
    l_commitdate        date ,
    l_receiptdate       date ,
    l_shipmode          char(10) ,
    l_linenumber        number ,
    l_shipinstruct      char(25) ,
    l_comment           varchar(44)
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
               records delimited by newline
               badfile 'l_et.bad'
               logfile 'l_et.log'
               fields terminated by '|'
               missing field values are null
```

```
)
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',
'lineitem.tbl.14',
'lineitem.tbl.15',
'lineitem.tbl.16',
'lineitem.tbl.17',
'lineitem.tbl.18',
'lineitem.tbl.19',
'lineitem.tbl.20',
'lineitem.tbl.21',
'lineitem.tbl.22',
'lineitem.tbl.23',
'lineitem.tbl.24',
'lineitem.tbl.25',
'lineitem.tbl.26',
'lineitem.tbl.27',
'lineitem.tbl.28',
'lineitem.tbl.29',
'lineitem.tbl.30',
'lineitem.tbl.31',
'lineitem.tbl.32',
'lineitem.tbl.33',
'lineitem.tbl.34',
'lineitem.tbl.35',
'lineitem.tbl.36',
'lineitem.tbl.37',
'lineitem.tbl.38',
'lineitem.tbl.39',
'lineitem.tbl.40',
'lineitem.tbl.41',
'lineitem.tbl.42',
'lineitem.tbl.43',
'lineitem.tbl.44',
'lineitem.tbl.45',
'lineitem.tbl.46',
'lineitem.tbl.47',
'lineitem.tbl.48',
'lineitem.tbl.49',
'lineitem.tbl.50',
'lineitem.tbl.51',
'lineitem.tbl.52',
'lineitem.tbl.53',
'lineitem.tbl.54',
'lineitem.tbl.55',
'lineitem.tbl.56',
'lineitem.tbl.57',
'lineitem.tbl.58',
'lineitem.tbl.59',
'lineitem.tbl.60',
'lineitem.tbl.61',
'lineitem.tbl.62',
'lineitem.tbl.63',
'lineitem.tbl.64',
'lineitem.tbl.65',
'lineitem.tbl.66',
'lineitem.tbl.67',
'lineitem.tbl.68',
'lineitem.tbl.69',
'lineitem.tbl.70',
'lineitem.tbl.71',
'lineitem.tbl.72',
'lineitem.tbl.73',
'lineitem.tbl.74',
'lineitem.tbl.75',
'lineitem.tbl.76',
'lineitem.tbl.77',
'lineitem.tbl.78',
'lineitem.tbl.79',
'lineitem.tbl.80',
'lineitem.tbl.81',
'lineitem.tbl.82',
'lineitem.tbl.83',
'lineitem.tbl.84'
))
reject limit unlimited;
}
*sql
{
connect tpch/tpch;
drop table o_et;
create table o_et(
    o_orderdate          date ,
    o_orderkey           number ,
    o_custkey            number ,
    o_orderpriority      char(15) ,
    o_shippriority       number ,
    o_clerk              char(15) ,
    o_orderstatus        char(1) ,
    o_totalprice         number ,
    o_comment            varchar(79)
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
               records delimited by newline
               badfile 'o_et.bad'
               logfile 'o_et.log'
               fields terminated by '|'
               missing field values are null
       )
       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',
        'orders.tbl.33',
        'orders.tbl.34',
        'orders.tbl.35',
        'orders.tbl.36',
        'orders.tbl.37',
        'orders.tbl.38',
        'orders.tbl.39',
        'orders.tbl.40',
        'orders.tbl.41',
        'orders.tbl.42',
        'orders.tbl.43',
        'orders.tbl.44',
        'orders.tbl.45',
        'orders.tbl.46',
        'orders.tbl.47',
        'orders.tbl.48',
        'orders.tbl.49',
        'orders.tbl.50',
        'orders.tbl.51',
        'orders.tbl.52',
        'orders.tbl.53',
        'orders.tbl.54',
        'orders.tbl.55',
        'orders.tbl.56',
        'orders.tbl.57',
        'orders.tbl.58',
        'orders.tbl.59',
        'orders.tbl.60',
        'orders.tbl.61',
        'orders.tbl.62',
        'orders.tbl.63',
        'orders.tbl.64',
        'orders.tbl.65',
        'orders.tbl.66',
        'orders.tbl.67',
        'orders.tbl.68',
        'orders.tbl.69',
        'orders.tbl.70',
        'orders.tbl.71',
        'orders.tbl.72',
        'orders.tbl.73',
        'orders.tbl.74',
        'orders.tbl.75',
        'orders.tbl.76',
        'orders.tbl.77',
        'orders.tbl.78',
        'orders.tbl.79',
        'orders.tbl.80',
        'orders.tbl.81',
        'orders.tbl.82',
        'orders.tbl.83',
        'orders.tbl.84'
        ))
reject limit unlimited;
}
*sql
{
connect tpch/tpch;
drop table ps_et;
create table ps_et(
    ps_partkey          number ,
    ps_suppkey          number ,
    ps_supplycost       number ,
    ps_availqty         number ,
    ps_comment          varchar(199)
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
                records delimited by newline
                badfile 'ps_et.bad'
                logfile 'ps_et.log'
                fields terminated by '|'
                missing field values are null
        )
        location (
        'partsupp.tbl.1',
        'partsupp.tbl.2',
        'partsupp.tbl.3',
        'partsupp.tbl.4',
        'partsupp.tbl.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',
        'partsupp.tbl.14',
        'partsupp.tbl.15',
        'partsupp.tbl.16',
        'partsupp.tbl.17',
        'partsupp.tbl.18',
        'partsupp.tbl.19',
        'partsupp.tbl.20',
        'partsupp.tbl.21',
        'partsupp.tbl.22',
        'partsupp.tbl.23',
        'partsupp.tbl.24',
        'partsupp.tbl.25',
        'partsupp.tbl.26',
        'partsupp.tbl.27',
        'partsupp.tbl.28',
        'partsupp.tbl.29',
        'partsupp.tbl.30',
        'partsupp.tbl.31',
        'partsupp.tbl.32',
        'partsupp.tbl.33',
        'partsupp.tbl.34',
        'partsupp.tbl.35',
        'partsupp.tbl.36',
        'partsupp.tbl.37',
        'partsupp.tbl.38',
        'partsupp.tbl.39',
        'partsupp.tbl.40',
        'partsupp.tbl.41',
        'partsupp.tbl.42',
        'partsupp.tbl.43',
        'partsupp.tbl.44',
        'partsupp.tbl.45',
        'partsupp.tbl.46',
        'partsupp.tbl.47',
        'partsupp.tbl.48',
        'partsupp.tbl.49',
        'partsupp.tbl.50',
        'partsupp.tbl.51',
        'partsupp.tbl.52',
        'partsupp.tbl.53',
        'partsupp.tbl.54',
        'partsupp.tbl.55',
        'partsupp.tbl.56',
        'partsupp.tbl.57',
        'partsupp.tbl.58',
        'partsupp.tbl.59',
        'partsupp.tbl.60',
        'partsupp.tbl.61',
        'partsupp.tbl.62',
```

```
       'partsupp.tbl.63',
       'partsupp.tbl.64'))
reject limit unlimited;
}
*sql
{
connect tpch/tpch;
drop table p_et;
create table p_et(
    p_partkey           number ,
    p_type              varchar(25) ,
    p_size              number ,
    p_brand             char(10) ,
    p_name              varchar(55) ,
    p_container         char(10) ,
    p_mfgr              char(25) ,
    p_retailprice       number ,
    p_comment           varchar(23)
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
               records delimited by newline
               badfile 'p_et.bad'
               logfile 'p_et.log'
               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'
))
reject limit unlimited;
}
*sql
{
connect tpch/tpch;
drop table c_et;
create table c_et(
    c_custkey           number ,
    c_mktsegment        char(10) ,
    c_nationkey         number ,
    c_name              varchar(25) ,
    c_address           varchar(40) ,
    c_phone             char(15) ,
    c_acctbal           number ,
    c_comment           varchar(117)
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
               records delimited by newline
               badfile 'c_et.bad'
               logfile 'c_et.log'
               fields terminated by '|'
               missing field values are null
       )
       location (
       'customer.tbl.1',
       'customer.tbl.2',
       'customer.tbl.3',
       'customer.tbl.4',
       'customer.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',
       'customer.tbl.14',
       'customer.tbl.15',
       'customer.tbl.16'
))
reject limit unlimited;
}
*sql
{
connect tpch/tpch;
drop table s_et;
create table s_et(
    s_suppkey           number ,
    s_nationkey         number ,
    s_comment           varchar(101) ,
    s_name              char(25) ,
    s_address           varchar(40) ,
    s_phone             char(15) ,
    s_acctbal           number
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
               records delimited by newline
               badfile 's_et.bad'
               logfile 's_et.log'
               fields terminated by '|'
               missing field values are null
       )
       location (
       'supplier.tbl'
       ))
reject limit unlimited;
}
*sql
{
connect 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
               badfile 'n_et.bad'
               logfile 'n_et.log'
               fields terminated by '|'
               missing field values are null
```

```
       )
       location (
       'nation.tbl'))
reject limit unlimited;
}
*sql
{
connect 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
               badfile 'r_et.bad'
               logfile 'r_et.log'
               fields terminated by '|'
               missing field values are null
       )
       location (
       'region.tbl'))
reject limit unlimited;
}
*sql
{
connect 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;
}
*sql
{
connect tpch/tpch
@?/rdbms/admin/utlxplan.sql;
}
*wait
*sql
{
set timing on
set echo on
!date
connect tpch/tpch;
connect tpch/tpch;
rem drop table lineitem;
create table lineitem(
    l_shipdate           ,
    l_orderkey           NOT NULL,
    l_discount           ,
    l_extendedprice      ,
    l_suppkey            NOT NULL,
    l_quantity           ,
    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 (initial 640m next 20m freelists 99)
parallel 128
nologging
partition by range (l_shipdate)
subpartition by hash(l_partkey)
subpartitions 16
(
partition item1 values less than (to_date('1992-
01-01','YYYY-MM-DD'))
tablespace ts_l1
,
partition item2 values less than (to_date('1992-
02-01','YYYY-MM-DD'))
tablespace ts_l2
,
partition item3 values less than (to_date('1992-
03-01','YYYY-MM-DD'))
tablespace ts_l3
,
partition item4 values less than (to_date('1992-
04-01','YYYY-MM-DD'))
tablespace ts_l4
,
partition item5 values less than (to_date('1992-
05-01','YYYY-MM-DD'))
tablespace ts_l5
,
partition item6 values less than (to_date('1992-
06-01','YYYY-MM-DD'))
tablespace ts_l6
,
partition item7 values less than (to_date('1992-
07-01','YYYY-MM-DD'))
tablespace ts_l7
,
partition item8 values less than (to_date('1992-
08-01','YYYY-MM-DD'))
tablespace ts_l8
,
partition item9 values less than (to_date('1992-
09-01','YYYY-MM-DD'))
tablespace ts_l9
,
partition item10 values less than
(to_date('1992-10-01','YYYY-MM-DD'))
tablespace ts_l10
,
partition item11 values less than
(to_date('1992-11-01','YYYY-MM-DD'))
tablespace ts_l11
,
partition item12 values less than
(to_date('1992-12-01','YYYY-MM-DD'))
tablespace ts_l12
,
partition item13 values less than
(to_date('1993-01-01','YYYY-MM-DD'))
tablespace ts_l13
,
partition item14 values less than
(to_date('1993-02-01','YYYY-MM-DD'))
tablespace ts_l14
,
```

```
partition item15 values less than
(to_date('1993-03-01','YYYY-MM-DD'))
tablespace ts_l15
,
partition item16 values less than
(to_date('1993-04-01','YYYY-MM-DD'))
tablespace ts_l16
,
partition item17 values less than
(to_date('1993-05-01','YYYY-MM-DD'))
tablespace ts_l17
,
partition item18 values less than
(to_date('1993-06-01','YYYY-MM-DD'))
tablespace ts_l18
,
partition item19 values less than
(to_date('1993-07-01','YYYY-MM-DD'))
tablespace ts_l19
,
partition item20 values less than
(to_date('1993-08-01','YYYY-MM-DD'))
tablespace ts_l20
,
partition item21 values less than
(to_date('1993-09-01','YYYY-MM-DD'))
tablespace ts_l21
,
partition item22 values less than
(to_date('1993-10-01','YYYY-MM-DD'))
tablespace ts_l22
,
partition item23 values less than
(to_date('1993-11-01','YYYY-MM-DD'))
tablespace ts_l23
,
partition item24 values less than
(to_date('1993-12-01','YYYY-MM-DD'))
tablespace ts_l24
,
partition item25 values less than
(to_date('1994-01-01','YYYY-MM-DD'))
tablespace ts_l25
,
partition item26 values less than
(to_date('1994-02-01','YYYY-MM-DD'))
tablespace ts_l26
,
partition item27 values less than
(to_date('1994-03-01','YYYY-MM-DD'))
tablespace ts_l27
,
partition item28 values less than
(to_date('1994-04-01','YYYY-MM-DD'))
tablespace ts_l28
,
partition item29 values less than
(to_date('1994-05-01','YYYY-MM-DD'))
tablespace ts_l29
,
partition item30 values less than
(to_date('1994-06-01','YYYY-MM-DD'))
tablespace ts_l30
,
partition item31 values less than
(to_date('1994-07-01','YYYY-MM-DD'))
tablespace ts_l31
,
partition item32 values less than
(to_date('1994-08-01','YYYY-MM-DD'))
tablespace ts_l32
,
partition item33 values less than
(to_date('1994-09-01','YYYY-MM-DD'))
tablespace ts_l33
,
partition item34 values less than
(to_date('1994-10-01','YYYY-MM-DD'))
tablespace ts_l34
,
partition item35 values less than
(to_date('1994-11-01','YYYY-MM-DD'))
tablespace ts_l35
,
partition item36 values less than
(to_date('1994-12-01','YYYY-MM-DD'))
tablespace ts_l36
,
partition item37 values less than
(to_date('1995-01-01','YYYY-MM-DD'))
tablespace ts_l37
,
partition item38 values less than
(to_date('1995-02-01','YYYY-MM-DD'))
tablespace ts_l38
,
partition item39 values less than
(to_date('1995-03-01','YYYY-MM-DD'))
tablespace ts_l39
,
partition item40 values less than
(to_date('1995-04-01','YYYY-MM-DD'))
tablespace ts_l40
,
partition item41 values less than
(to_date('1995-05-01','YYYY-MM-DD'))
tablespace ts_l41
,
partition item42 values less than
(to_date('1995-06-01','YYYY-MM-DD'))
tablespace ts_l42
,
partition item43 values less than
(to_date('1995-07-01','YYYY-MM-DD'))
tablespace ts_l43
,
partition item44 values less than
(to_date('1995-08-01','YYYY-MM-DD'))
tablespace ts_l44
,
partition item45 values less than
(to_date('1995-09-01','YYYY-MM-DD'))
tablespace ts_l45
,
partition item46 values less than
(to_date('1995-10-01','YYYY-MM-DD'))
tablespace ts_l46
,
partition item47 values less than
(to_date('1995-11-01','YYYY-MM-DD'))
tablespace ts_l47
,
partition item48 values less than
(to_date('1995-12-01','YYYY-MM-DD'))
tablespace ts_l48
,
partition item49 values less than
(to_date('1996-01-01','YYYY-MM-DD'))
tablespace ts_l49
,
partition item50 values less than
(to_date('1996-02-01','YYYY-MM-DD'))
tablespace ts_l50
,
```

```
partition item51 values less than
(to_date('1996-03-01','YYYY-MM-DD'))
tablespace ts_l51
,
partition item52 values less than
(to_date('1996-04-01','YYYY-MM-DD'))
tablespace ts_l52
,
partition item53 values less than
(to_date('1996-05-01','YYYY-MM-DD'))
tablespace ts_l53
,
partition item54 values less than
(to_date('1996-06-01','YYYY-MM-DD'))
tablespace ts_l54
,
partition item55 values less than
(to_date('1996-07-01','YYYY-MM-DD'))
tablespace ts_l55
,
partition item56 values less than
(to_date('1996-08-01','YYYY-MM-DD'))
tablespace ts_l56
,
partition item57 values less than
(to_date('1996-09-01','YYYY-MM-DD'))
tablespace ts_l57
,
partition item58 values less than
(to_date('1996-10-01','YYYY-MM-DD'))
tablespace ts_l58
,
partition item59 values less than
(to_date('1996-11-01','YYYY-MM-DD'))
tablespace ts_l59
,
partition item60 values less than
(to_date('1996-12-01','YYYY-MM-DD'))
tablespace ts_l60
,
partition item61 values less than
(to_date('1997-01-01','YYYY-MM-DD'))
tablespace ts_l61
,
partition item62 values less than
(to_date('1997-02-01','YYYY-MM-DD'))
tablespace ts_l62
,
partition item63 values less than
(to_date('1997-03-01','YYYY-MM-DD'))
tablespace ts_l63
,
partition item64 values less than
(to_date('1997-04-01','YYYY-MM-DD'))
tablespace ts_l64
,
partition item65 values less than
(to_date('1997-05-01','YYYY-MM-DD'))
tablespace ts_l65
,
partition item66 values less than
(to_date('1997-06-01','YYYY-MM-DD'))
tablespace ts_l66
,
partition item67 values less than
(to_date('1997-07-01','YYYY-MM-DD'))
tablespace ts_l67
,
partition item68 values less than
(to_date('1997-08-01','YYYY-MM-DD'))
tablespace ts_l68
,
partition item69 values less than
(to_date('1997-09-01','YYYY-MM-DD'))
tablespace ts_l69
,
partition item70 values less than
(to_date('1997-10-01','YYYY-MM-DD'))
tablespace ts_l70
,
partition item71 values less than
(to_date('1997-11-01','YYYY-MM-DD'))
tablespace ts_l71
,
partition item72 values less than
(to_date('1997-12-01','YYYY-MM-DD'))
tablespace ts_l72
,
partition item73 values less than
(to_date('1998-01-01','YYYY-MM-DD'))
tablespace ts_l73
,
partition item74 values less than
(to_date('1998-02-01','YYYY-MM-DD'))
tablespace ts_l74
,
partition item75 values less than
(to_date('1998-03-01','YYYY-MM-DD'))
tablespace ts_l75
,
partition item76 values less than
(to_date('1998-04-01','YYYY-MM-DD'))
tablespace ts_l76
,
partition item77 values less than
(to_date('1998-05-01','YYYY-MM-DD'))
tablespace ts_l77
,
partition item78 values less than
(to_date('1998-06-01','YYYY-MM-DD'))
tablespace ts_l78
,
partition item79 values less than
(to_date('1998-07-01','YYYY-MM-DD'))
tablespace ts_l79
,
partition item80 values less than
(to_date('1998-08-01','YYYY-MM-DD'))
tablespace ts_l80
,
partition item81 values less than
(to_date('1998-09-01','YYYY-MM-DD'))
tablespace ts_l81
,
partition item82 values less than
(to_date('1998-10-01','YYYY-MM-DD'))
tablespace ts_l82
,
partition item83 values less than
(to_date('1998-11-01','YYYY-MM-DD'))
tablespace ts_l83
,
partition item84 values less than (MAXVALUE)
tablespace ts_l84 )
as select * from l_et;
!date
}
*wait
*sql
{
connect tpch/tpch;
set timing on
set echo on
!date
```

```
rem drop table orders;
create table orders(
    o_orderdate          ,
    o_orderkey           NOT NULL,
    o_custkey            NOT NULL,
    o_orderpriority      ,
    o_shippriority       ,
    o_clerk              ,
    o_orderstatus        ,
    o_totalprice         ,
    o_comment            )
pctfree 1
pctused 99
initrans 10
storage (initial 130m next 10m freelists 99)
parallel 128
nologging
partition by range (o_orderdate)
subpartition by hash(o_custkey)
subpartitions 16
(
partition ord1 values less than (to_date('1992-
01-01','YYYY-MM-DD'))
tablespace ts_o1,
partition ord2 values less than (to_date('1992-
02-01','YYYY-MM-DD'))
tablespace ts_o2,
partition ord3 values less than (to_date('1992-
03-01','YYYY-MM-DD'))
tablespace ts_o3,
partition ord4 values less than (to_date('1992-
04-01','YYYY-MM-DD'))
tablespace ts_o4,
partition ord5 values less than (to_date('1992-
05-01','YYYY-MM-DD'))
tablespace ts_o5,
partition ord6 values less than (to_date('1992-
06-01','YYYY-MM-DD'))
tablespace ts_o6,
partition ord7 values less than (to_date('1992-
07-01','YYYY-MM-DD'))
tablespace ts_o7,
partition ord8 values less than (to_date('1992-
08-01','YYYY-MM-DD'))
tablespace ts_o8,
partition ord9 values less than (to_date('1992-
09-01','YYYY-MM-DD'))
tablespace ts_o9,
partition ord10 values less than (to_date('1992-
10-01','YYYY-MM-DD'))
tablespace ts_o10,
partition ord11 values less than (to_date('1992-
11-01','YYYY-MM-DD'))
tablespace ts_o11,
partition ord12 values less than (to_date('1992-
12-01','YYYY-MM-DD'))
tablespace ts_o12,
partition ord13 values less than (to_date('1993-
01-01','YYYY-MM-DD'))
tablespace ts_o1,
partition ord14 values less than (to_date('1993-
02-01','YYYY-MM-DD'))
tablespace ts_o2,
partition ord15 values less than (to_date('1993-
03-01','YYYY-MM-DD'))
tablespace ts_o3,
partition ord16 values less than (to_date('1993-
04-01','YYYY-MM-DD'))
tablespace ts_o4,
partition ord17 values less than (to_date('1993-
05-01','YYYY-MM-DD'))
tablespace ts_o5,
partition ord18 values less than (to_date('1993-
06-01','YYYY-MM-DD'))
tablespace ts_o6,
partition ord19 values less than (to_date('1993-
07-01','YYYY-MM-DD'))
tablespace ts_o7,
partition ord20 values less than (to_date('1993-
08-01','YYYY-MM-DD'))
tablespace ts_o8,
partition ord21 values less than (to_date('1993-
09-01','YYYY-MM-DD'))
tablespace ts_o9,
partition ord22 values less than (to_date('1993-
10-01','YYYY-MM-DD'))
tablespace ts_o10,
partition ord23 values less than (to_date('1993-
11-01','YYYY-MM-DD'))
tablespace ts_o11,
partition ord24 values less than (to_date('1993-
12-01','YYYY-MM-DD'))
tablespace ts_o12,
partition ord25 values less than (to_date('1994-
01-01','YYYY-MM-DD'))
tablespace ts_o1,
partition ord26 values less than (to_date('1994-
02-01','YYYY-MM-DD'))
tablespace ts_o2,
partition ord27 values less than (to_date('1994-
03-01','YYYY-MM-DD'))
tablespace ts_o3,
partition ord28 values less than (to_date('1994-
04-01','YYYY-MM-DD'))
tablespace ts_o4,
partition ord29 values less than (to_date('1994-
05-01','YYYY-MM-DD'))
tablespace ts_o5,
partition ord30 values less than (to_date('1994-
06-01','YYYY-MM-DD'))
tablespace ts_o6,
partition ord31 values less than (to_date('1994-
07-01','YYYY-MM-DD'))
tablespace ts_o7,
partition ord32 values less than (to_date('1994-
08-01','YYYY-MM-DD'))
tablespace ts_o8,
partition ord33 values less than (to_date('1994-
09-01','YYYY-MM-DD'))
tablespace ts_o9,
partition ord34 values less than (to_date('1994-
10-01','YYYY-MM-DD'))
tablespace ts_o10,
partition ord35 values less than (to_date('1994-
11-01','YYYY-MM-DD'))
tablespace ts_o11,
partition ord36 values less than (to_date('1994-
12-01','YYYY-MM-DD'))
tablespace ts_o12,
partition ord37 values less than (to_date('1995-
01-01','YYYY-MM-DD'))
tablespace ts_o1,
partition ord38 values less than (to_date('1995-
02-01','YYYY-MM-DD'))
tablespace ts_o2,
partition ord39 values less than (to_date('1995-
03-01','YYYY-MM-DD'))
tablespace ts_o3,
partition ord40 values less than (to_date('1995-
04-01','YYYY-MM-DD'))
tablespace ts_o4,
partition ord41 values less than (to_date('1995-
05-01','YYYY-MM-DD'))
```

```
tablespace ts_o5,
partition ord42 values less than (to_date('1995-
06-01','YYYY-MM-DD'))
tablespace ts_o6,
partition ord43 values less than (to_date('1995-
07-01','YYYY-MM-DD'))
tablespace ts_o7,
partition ord44 values less than (to_date('1995-
08-01','YYYY-MM-DD'))
tablespace ts_o8,
partition ord45 values less than (to_date('1995-
09-01','YYYY-MM-DD'))
tablespace ts_o9,
partition ord46 values less than (to_date('1995-
10-01','YYYY-MM-DD'))
tablespace ts_o10,
partition ord47 values less than (to_date('1995-
11-01','YYYY-MM-DD'))
tablespace ts_o11,
partition ord48 values less than (to_date('1995-
12-01','YYYY-MM-DD'))
tablespace ts_o12,
partition ord49 values less than (to_date('1996-
01-01','YYYY-MM-DD'))
tablespace ts_o1,
partition ord50 values less than (to_date('1996-
02-01','YYYY-MM-DD'))
tablespace ts_o2,
partition ord51 values less than (to_date('1996-
03-01','YYYY-MM-DD'))
tablespace ts_o3,
partition ord52 values less than (to_date('1996-
04-01','YYYY-MM-DD'))
tablespace ts_o4,
partition ord53 values less than (to_date('1996-
05-01','YYYY-MM-DD'))
tablespace ts_o5,
partition ord54 values less than (to_date('1996-
06-01','YYYY-MM-DD'))
tablespace ts_o6,
partition ord55 values less than (to_date('1996-
07-01','YYYY-MM-DD'))
tablespace ts_o7,
partition ord56 values less than (to_date('1996-
08-01','YYYY-MM-DD'))
tablespace ts_o8,
partition ord57 values less than (to_date('1996-
09-01','YYYY-MM-DD'))
tablespace ts_o9,
partition ord58 values less than (to_date('1996-
10-01','YYYY-MM-DD'))
tablespace ts_o10,
partition ord59 values less than (to_date('1996-
11-01','YYYY-MM-DD'))
tablespace ts_o11,
partition ord60 values less than (to_date('1996-
12-01','YYYY-MM-DD'))
tablespace ts_o12,
partition ord61 values less than (to_date('1997-
01-01','YYYY-MM-DD'))
tablespace ts_o1,
partition ord62 values less than (to_date('1997-
02-01','YYYY-MM-DD'))
tablespace ts_o2,
partition ord63 values less than (to_date('1997-
03-01','YYYY-MM-DD'))
tablespace ts_o3,
partition ord64 values less than (to_date('1997-
04-01','YYYY-MM-DD'))
tablespace ts_o4,
partition ord65 values less than (to_date('1997-
05-01','YYYY-MM-DD'))
tablespace ts_o5,
partition ord66 values less than (to_date('1997-
06-01','YYYY-MM-DD'))
tablespace ts_o6,
partition ord67 values less than (to_date('1997-
07-01','YYYY-MM-DD'))
tablespace ts_o7,
partition ord68 values less than (to_date('1997-
08-01','YYYY-MM-DD'))
tablespace ts_o8,
partition ord69 values less than (to_date('1997-
09-01','YYYY-MM-DD'))
tablespace ts_o9,
partition ord70 values less than (to_date('1997-
10-01','YYYY-MM-DD'))
tablespace ts_o10,
partition ord71 values less than (to_date('1997-
11-01','YYYY-MM-DD'))
tablespace ts_o11,
partition ord72 values less than (to_date('1997-
12-01','YYYY-MM-DD'))
tablespace ts_o12,
partition ord73 values less than (to_date('1998-
01-01','YYYY-MM-DD'))
tablespace ts_o1,
partition ord74 values less than (to_date('1998-
02-01','YYYY-MM-DD'))
tablespace ts_o2,
partition ord75 values less than (to_date('1998-
03-01','YYYY-MM-DD'))
tablespace ts_o3,
partition ord76 values less than (to_date('1998-
04-01','YYYY-MM-DD'))
tablespace ts_o4,
partition ord77 values less than (to_date('1998-
05-01','YYYY-MM-DD'))
tablespace ts_o5,
partition ord78 values less than (to_date('1998-
06-01','YYYY-MM-DD'))
tablespace ts_o6,
partition ord79 values less than (to_date('1998-
07-01','YYYY-MM-DD'))
tablespace ts_o7,
partition ord80 values less than (to_date('1998-
08-01','YYYY-MM-DD'))
tablespace ts_o8,
partition ord81 values less than (to_date('1998-
09-01','YYYY-MM-DD'))
tablespace ts_o9,
partition ord82 values less than (to_date('1998-
10-01','YYYY-MM-DD'))
tablespace ts_o10,
partition ord83 values less than (to_date('1998-
11-01','YYYY-MM-DD'))
tablespace ts_o11,
partition ord84 values less than (MAXVALUE)
tablespace ts_o12
)
as select * from o_et;
!date
}
*wait
*sql
{
connect tpch/tpch
set timing on
set echo on

!date

rem drop table partsupp;
create table partsupp(
```

```
    ps_partkey          not null ,
    ps_suppkey           not null,
    ps_supplycost        not null,
    ps_availqty          ,
    ps_comment           ,
constraint pk_partkey_suppkey_1 primary
key(ps_partkey, ps_suppkey)
)
organization index
partition by hash(ps_partkey)
partitions 128
storage (initial 1100m next 50m)
parallel 128
nologging
compress
pctthreshold 50
tablespace ts_psupp1
as select * from ps_et;
!date
}
*wait
*sql
{
connect tpch/tpch
set timing on
set echo on

!date
rem drop table customer;
create table customer(
    c_custkey            NOT NULL,
    c_mktsegment         ,
    c_nationkey          ,
    c_name               ,
    c_address            ,
    c_phone              ,
    c_acctbal            ,
    c_comment            )
pctfree 0
pctused 99
parallel 128
nologging
storage (initial 180m next 10m)
partition by hash (c_custkey)
partitions 128
store in
(ts_o1,ts_o2,ts_o3,ts_o4,ts_o5,ts_o6,ts_o7,ts_o8
,ts_o9,ts_o10,
ts_o11,ts_o12)
as select * from c_et;
!date
}
*wait
*sql
{
connect tpch/tpch
set timing on
set echo on

!date
rem drop table part;

create table part(
    p_partkey            NOT NULL,
    p_type               ,
    p_size               ,
    p_brand              ,
    p_name               ,
    p_container          ,
    p_mfgr               ,
    p_retailprice        ,
    p_comment            )
pctfree 0
pctused 99
parallel 128
nologging
storage (initial 200m next 10m)
partition by hash (p_partkey)
partitions 128
store in
(ts_o1,ts_o2,ts_o3,ts_o4,ts_o5,ts_o6,ts_o7,ts_o8
,ts_o9,ts_o10,
ts_o11,ts_o12)
as select * from p_et;
!date
}
*wait
*sql
{
connect tpch/tpch;
set timing on
set echo on
rem drop table supplier;
create table supplier(
    s_suppkey            NOT NULL,
    s_nationkey          ,
    s_comment            ,
    s_name               ,
    s_address            ,
    s_phone              ,
    s_acctbal            )
pctfree 0
pctused 99
parallel 128
nologging
storage (initial 40m next 5m)
partition by hash (s_suppkey)
partitions 128
store in
(ts_o1,ts_o2,ts_o3,ts_o4,ts_o5,ts_o6,ts_o7,ts_o8
,ts_o9,ts_o10,
ts_o11,ts_o12)
as select * from s_et;
}
*wait
*sql
{
connect tpch/tpch;
set echo on
set timing on

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

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

*wait
*bgoff
%e-scuto

*sql
```

```
{
connect tpch/tpch
set timing on
set echo on

!date
drop table l_et;
drop table o_et;
drop table ps_et;
drop table p_et;
drop table c_et;
drop table s_et;
drop table n_et;
drop table r_et;
}
*bgoff
%e-dapop
%b-ixcre
*bgon=1
################################################
##############################
# Index Creation Phase
*sql
{
connect tpch/tpch;
!date
set echo on
set timing on
rem drop index i_l_orderkey;
create index i_l_orderkey
on lineitem (l_orderkey)
pctfree 2
initrans 10
tablespace ts_lokey
storage (freelists 99)
parallel 128
compute statistics
nologging;
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
alter index i_l_orderkey allocate extent (size
300m instance 1);
}
*sql
{
connect tpch/tpch;
!date
set echo on
set timing on
rem drop index i_o_orderkey;
create unique index i_o_orderkey
on orders (o_orderkey)
pctfree 2
initrans 10
tablespace ts_okey
storage (freelists 99 )
parallel 128
compute statistics
nologging;
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
alter index i_o_orderkey allocate extent (size
300m instance 1);
}
*sql
{
connect tpch/tpch;
!date
set echo on
set timing on

rem drop index i_c_custkey;
create unique index i_c_custkey
on customer (c_custkey)
pctfree 2
initrans 10
tablespace ts_custkey
storage (freelists 99)
parallel 128
compute statistics
nologging;
}
*wait
*bgoff
%e-ixcre
%b-anlyz
*bgon=1
```

```
################################################
#############################
# Analyze Phase
*sql
{
connect tpch/tpch;
!date
set timing on
execute dbms_stats.gather_schema_stats('TPCH' ,
estimate_percent => 1, degree => 128 ,
granularity => 'GLOBAL' );
connect / as sysdba
execute dbms_stats.gather_system_stats;
alter system switch logfile;
!date
}
*wait
*bgoff
%e-anlyz
```

B.2 bumpx.pl

```
#!/usr/contrib/bin/perl
#
# $Header: bumpx.pl 13-feb-2001.14:35:03 mpoess
Exp $
#
# bumpx.pl
#
#  Copyright (c) Oracle Corporation 1999, 2000.
All Rights Reserved.
#
#    NAME
#      bumpx.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      02/13/01 - make global range
index possible for o_orderkey
#    mpoess      01/31/00 - fix bug with undo
#    mpoess      01/21/00 - fix bugs for
partitioned tablespaces
#    mpoess      01/20/00 - add OPS load one
partiton from one node support
#    mpoess      01/07/00 - add support for
flatfile distribution
#    mpoess      01/06/00 - add controlfile
generation performace improvement
#    mpoess      12/16/99 - add partitioning
support for control files
#    mpoess      10/28/99 - adjust bumpx to
fixed dbgen (84 partitioning)
#    mpoess      08/17/99 - change column
definition of base tables
#    mpoess      08/13/99 - add support for
reporting files
#    mpoess      08/08/99 - change file
atributes
#    mpoess      07/12/99 - add setpath for
dbgen
#    mpoess      07/12/99 - disable -o option
for dbgen
#    mpoess      07/07/99 - fix dbgen phase
#    mpoess      07/06/99 - change path to perl
#    mpoess      07/02/99 - add environment
variable support for conf file
#    mpoess      06/28/99 - add pathname for
param.txt file
#    mpoess      06/28/99 - Copy from TPCD
#    mpoess      06/28/99 - Creation
#    MODIFIED   (MM/DD/YY)
#    mpoess      11/18/98 - add undo rollback
segment support for OPS systems
#    mpoess      11/18/98 - add control log etc.
keywords in SQLLDR call
#    mpoess      11/06/98 - fix bug in
print_as_select
#    mpoess      11/05/98 - change name of
BUMPC_CTR variable
#    mpoess      11/04/98 - delete default
startup before index creation
#    mpoess      11/04/98 - adopt Barry Perkins
changes to dbgen
#    mpoess      11/03/98 - change as select
handling
#    mpoess      11/02/98 - change background
process handling for NT
#    mpoess      10/26/98 - modifiy dapop so
that multiple partitions can be lo
#    mpoess      10/23/98 - take *waits out from
flat file generation
#    mpoess      10/20/98 - add sdgen, dbgen and
dapop description
#    mpoess      10/12/98 - fix bug in
constraint creation
#    mpoess      10/12/98 - add support for non
partitioned tables
#    mpoess      10/08/98 - add -including new
values- clause in view log defin
#    mpoess      09/27/98 - update
links,param2html
#    mpoess      09/25/98 - adding new features
#    akarasik    07/29/98 -
#    akarasik    07/29/98 - Checking in
#     kwong     05/22/98 -  add change storage
for tables and indexes in
#                            chdop
#     kwong      05/21/98 -  fix load pipe
#     kwong      05/21/98 -  add some more
default values in init.ora
#     kwong      05/20/98 -  add parameter
nt_port
#     kwong     05/19/98 -  fix control file
generation
#     kwong      05/18/98 -  run utlxplan.sql
after created user
#     kwong      05/18/98 -  add "dbgen" phase
#     kwong      05/15/98 -  calculate partition
values for l_orderkey
#                            and l_partkey
#     kwong      05/13/98 -  add phase "chdop"
for modifying dop as the
#                            end of the build
#     kwong      05/13/98 -  fix the order of
create tablespaces and add
#                            datafiles
#     pswong     04/15/97 -  fix pipeline load
#     pswong     04/02/97 -  analyze partition
support
#     pswong     03/27/97 -  introduced ts
groups
#     pswong     03/18/97 -  named pipe loader
and dbgen support
```

```
#     pswong     02/28/97 -  more partition
support
#     pswong     02/14/97 -  Version 8
#     amozes     10/27/94 -  Creation

#*********************************************
*****************************
#*********************************************
*****************************
# Main Section
#*********************************************
*****************************
#*********************************************
*****************************

$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;
$allphases =
"dbcre,shutd,start,dbgen,sccre,scuto,scuvo,sctso
,dapop,ixcre,anlyz,chob,expln,query,sdgen,plcre"
;
if (($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,ixcre,chob";
}
$allbmtypes = "tpcd,wisc";
$bmtype = "tpcd" if !defined $bmtype;
$pdfile =  "$ENV{'BUMPX_DIR'}/param.txt";  #
This file contains the description of all
possible parameters.
&read_parameter_description;
$runsilent=0;
while ($arg = shift(@ARGV))
{
    if ($arg !~ /-(i|o|t|c|x|p|d|a|s)/)
    {
        $error =  "** Error: Bad argument to $0:
$arg\n";
        &usage;
    }
    $runsilent = 1 if ($arg =~ /-s/);
    $infile = shift(@ARGV) if ($arg =~ /-i/);
    $outfile = shift(@ARGV) if ($arg =~ /-o/);
    $bmtype = shift(@ARGV) if ($arg =~ /-t/);
    $docreate = 1 if (($arg =~ /-c/) || ($arg =~
/-xc/));
    $doexecute = 1 if (($arg =~ /-x/) || ($arg
=~ /-cx/));
    $phaselist = shift(@ARGV) if ($arg =~ /-p/);
    if ($arg =~ /-d/)
    {
        $defpar = shift(@ARGV);
        &defaults;
        @keys = keys %params;
        while ($#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);
    }
    if ($arg =~ /-a/)
    {
        $infile = "$ENV{'BUMPX_DIR'}/bumpx.conf"
if !defined $infile;
        $infile = $infile;
        &readfile;
        &defaults;
        @keys = keys %params;
        while ($#keys >=0)
        {
            $key = pop(@keys);
            print $key, "=", $params{$key}, "\n";
        }
        exit(0);
    }
}

$docreate = 0 if !defined $docreate;
$doexecute = 0 if !defined $doexecute;
if (!($docreate || $doexecute))
{
    $error = "** Error: Must specify either -c
or -x or both\n";
    &usage;
}
$infile = "$ENV{'BUMPX_DIR'}/bumpx.conf" if
!defined $infile;
$outfile = "$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
commands and nruntpb to synchronize them)
            $tmpfile = "tmp.txt";
            $tmpfile = $filedir.$tmpfile;
            $nruntpb =  "nruntpb.exe";
            ## NT End
        }

if ((!-e $infile) && !$doexecute && $docreate)
{
    $error = "** Error: -i file, $infile, does
not exist\n";
    &usage;
}
if ((!-e $outfile) && !$docreate)
{
```

```
    $error = "** Error: -o file, $outfile, does
not exist\n";
    &usage;
}
$phaselist = $defphases if !defined $phaselist;
@phases = split(/,/, $phaselist);

if ($docreate)
{
    open (OUTFILE, ">$outfile") if $docreate;
    &readfile;
    &defaults;
    &docreate;
    close (OUTFILE);
}

## NT Port (Use tmpfile to buffer commands for
nruntpb)
open (TMPFILE, ">$tmpfile") if ( (($os cmp "nt")
== 0) && $doexecute);
## NT End

&doexecute if $doexecute;

## NT Port
close(TMPFILE) if ( (($os cmp "nt") == 0) &&
$doexecute);
## NT End


exit(0);

sub readfile
{
    $matchon = 0;
    $contin = 0;
    $pkey = "";
    $pval = "";
    open (INFILE, "$infile");
  WLOOP:
    while ($line = <INFILE>)
    {
        $line = $line."\n" if $line !~ /\n/;
        study $line;
        if ($line =~ /^\*matchon/)
        {
            $matchon = 1;
            next WLOOP;
        }
        if ($line =~ /^\*matchoff/)
        {
            $matchon = 0;
            next WLOOP;
        }
        if ($matchon == 1)
        {
            &dump0($line) if $docreate;
            next WLOOP;
        }
        next WLOOP if $line =~ /^\s*\#/;
        next WLOOP if $line =~ /^\s*\n/;
        if ($contin)
        {
            if ($line =~ /(.*)\&\s*\n/) # still
continuing (changed \ to &)
            {
#                $pval = $pval . $1 . "\n";
                $pval = $pval . $1;
                next WLOOP;
            }
            $line =~ /(.*)\s*\n/; # reached the
end
            $pval = $pval . $1;
            $pval =~ s/\?/$ENV{'ORACLE_HOME'}/g;
            &key_exists($pkey);
            if ($result!=1)
            {
                print "Parameter $pkey does not
exist.\nBailing out!\n";
                exit (0);
            }
            $params{$pkey} = $pval;
            $contin = 0;
            $pkey = "";
            $pval = "";
            next WLOOP;
        }
        else
        {
            if ($line =~
/\s*(\S+)\s*=\s*(.*)\&\s*\n/)
            {
                $pkey = $1;
#                $pval = $2 . "\n";
                $pval = $2;
                $contin = 1;
                next WLOOP;
            }
            if ($line =~ /\s*(\S+)\s*=\s*\n/)
            {
                undef($params{$1});
                next WLOOP;
            }
            if ($line !~
/\s*(\S+)\s*=\s*((\S+)|(\S+.*\S+))\s*\n/)
            {
                print "Bad line: $line";
                next WLOOP;
            }
            $tkey = $1;
            $tval = $2;
            $tval =~ s/\?/$ENV{'ORACLE_HOME'}/g;
            if ($tval =~ /\$/) {  # a $ sign at
the beginning of the contents
                                    # of the
parameter value results in a environment
                                     # variable
lookup and substitution
                $tenv = $tval;
                $tval =~ s/\$//g;
                if ($unix) {
                    $tenv =~
s/(.*)\/\$(.*)\/(.*)/\$$2/g;
                }
                $tenvval = `echo $tenv`;
                $tenv =~ s/\$//g;
                chop($tenvval);
                if ($tenvval=~/^\n/) {
                    print "bumpx error:
Environment variable $tval not defined!\nbailing
out...\n";
                    exit(1);
                }
                $tval =~ s/$tenv/$tenvval/g;
            }

            &key_exists($tkey);
            if ($result!=1)
            {
                print "Parameter $tkey does not
exists.\nBailing out!\n";
                exit (0);
            }
            $params{$tkey} = $tval;
```

```
        }
    }
    close (INFILE);
}

sub docreate
{
    print "Creation pass begun." if $verbose;
    @phases_tmp = @phases; # because I will eat
the elements
    while ($phase=shift(@phases_tmp))
    {
        if ($allphases =~ /$phase/)
        {
            $nextphase = shift(@phases_tmp);
            unshift (@phases_tmp,$nextphase) if
($nextphase);
            print "\n  Creating phase \'$phase\'"
if $verbose;
            &dump0("%b-$phase\n");
            &prepmulti();

&dump0("*bgon=$params{$phase.'_max_bg'}");
            eval (&$phase);
            &dump0("*wait");
            &dump0("*bgoff");
            &dump0("%e-$phase\n");
        }
        else
        {
            print "\n  Phase \'$phase\' not
built in...assuming a \*match block being used";
        }
    }
    print "\nCreation pass complete.\n" if
$verbose;
}

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)/;
            $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 . $1;
## NT Port (Ignore '\n')
               if (($os cmp "nt") == 0)
               {
                   $insert =~ /(.*)\n$/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 " $0 [-c] [-x] [-i infile] [-o
outfile] [-p phaselist] [-t type]\n";
   print "   -c : create intermediary file
(needed for execution)\n";
   print "   -x : execute intermediary file\n";
   print "   -i : configuration file to be
used\n";
   print "        defaults to bumpx.conf in
\$BUMPX_DIR or \$CWD\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 -c -o file.out\n";
    print "   This will create an intermediary
file named file.out for the default\n";
    print "   phases using bumpx.conf which can
be executed in the future.\n";
    print " $0 -x -p dapop\n";
    print "   Executes data population phase of
intermediary file bumpx.dat.\n";
    print " $0 -cx -p dbcre,dapop\n";
    print "   This will create an intermediary
file, bumpx.dat, using configuration\n";
    print "   file, bumpx.conf, for both the
database creation phase and the data\n";
    print "   population phase, and then execute
that file.\n";
    print "\n";
    print "$error\n";
    exit(-1);
}

#*********************************************
*****************************
#*********************************************
*****************************
# All Known Phases
#*********************************************
*****************************
#*********************************************
*****************************

sub dbcre
{

&dump0("#######################################
#################################");
    &dump0("# Database Creation Phase");
    &time0("Begin database creation");

# Shut down anything that may have been running
    &shutdb();

# Create database (system tablespace/datafile,
redo logs, rollback segments)
    @sys_files = split (/,/,
$params{'ts_sys_datafiles'});
    $sys_file = shift(@sys_files);
    @log_files = split (/,/,
$params{'ts_log_datafiles'});
    &dump0("# creating database and initial
rollback segment");
    $cmd .= "startup pfile=
$params{'startupfile_dbcre'} nomount;\n";
    $cmd .= "create database\n";
    $cmd .= "  controlfile reuse\n" if
!($params{'db_ctlreuse'} =~ /false/);
    for ($i = 0; $i <
$params{'ts_log_#files_pt'}; $i++)
    {
        ($i == 0) ? ($addname = "logfile") :
($addname = "       ");
        (($i+1) == $params{'ts_log_#files_pt'})
? ($addcomma = "") : ($addcomma = ",");
        $log_file = shift(@log_files);
        if (($os cmp "unix") == 0)
        {
            $cmd .= "  $addname
'$params{'ts_log_area'}$log_file' size
$params{'ts_log_size'}
$params{'ts_log_options'}$addcomma\n";
        }
        else
        {
            $cmd .= "  $addname (
'$params{'ts_log_area'}$log_file') size
$params{'ts_log_size'}
$params{'ts_log_options'}$addcomma\n";
        }
    }
    $cmd .= "  maxlogfiles
$params{'db_maxlogfiles'}\n" if defined
$params{'db_maxlogfiles'};
    $cmd .= "  maxlogmembers
$params{'db_maxlogmembers'}\n" if defined
$params{'db_maxlogmembers'};
    $cmd .= "  maxloghistory
$params{'db_maxloghistory'}\n" if defined
$params{'db_maxloghistory'};
    $cmd .= "  datafile
'$params{'ts_sys_area'}$sys_file' size
$params{'ts_sys_size'}
$params{'ts_sys_options'}\n";
    $cmd .= "  maxdatafiles
$params{'db_maxdatafiles'}\n" if defined
$params{'db_maxdatafiles'};
    $cmd .= "  maxinstances
$params{'db_maxinstances'}\n" if defined
$params{'db_maxinstances'};
    $cmd .= "  maxarchlogs
$params{'db_maxarchlogs'}\n" if defined
$params{'db_maxarchlogs'};
    $cmd .= "  archivelog\n" if defined
$params{'db_archivelog'};
    $cmd .= "  character set
$params{'db_charset'}\n" if defined
$params{'db_charset'};
    $cmd .= ";\n";
    $cmd .= "\n";
    $cmd .= "create public rollback segment
t_rs1 ";
    $cmd .= "  storage
$params{'ts_undo_rs_storage'}" if defined
$params{'ts_undo_rs_storage'};
    $cmd .= ";\n";
    $cmd .= "\n";
    $cmd .= "alter rollback segment t_rs1
online;\n";
    $cmd .= "\n";
    $cmd .= "shutdown\n"; #^^sd
    &dump1("*sql");
    &dump0("*wait");

# This startup is in its own session because of
some weird bug I've been
# seeing on the SP2; otherwise, it's in the
previous session
#^^su
#    $cmd = "";
#    $cmd .= "startup pfile=$iofname[1];\n";
#    &dump1("*sql");
#    &dump0("*wait");
    startdb("dbcre");
    &dump0("*wait");
```

```
    @ops_nodes =
split(/,/,$params{'ops_nodes'});
    if (defined $params{'ops_nodes'}) {
        foreach $ops_node (@ops_nodes)
        {
            if ($ops_node =~ /undo/)
            {
                $ts_undo = 'undo';
            }
            else {
                $ts_undo = 'undo_'.$ops_node;
            }
            if ($params{'skip_ts'} !~ /$ts_undo/)
            {
                &add_ts_rollb ($ts_undo);
            }
        }
    } else {
        $ts_undo = 'undo';
    }


    # currently set up for the foreground -
maybe should be changed
    &dump0("# creating extra logfile threads");
    for ($i = 1; $i <
$params{'ts_log_#threads'}; $i++)
    {
        $thrno = $i + 1;
        $cmd .= "alter database add logfile
thread $thrno\n";
        for ($j = 0; $j <
$params{'ts_log_#files_pt'}; $j++)
        {
            (($j+1) ==
$params{'ts_log_#files_pt'}) ? ($addcomma = ";")
: ($addcomma = ",");
            $log_file = shift(@log_files);
            $cmd .= "
'$params{'ts_log_area'}$log_file' size
$params{'ts_log_size'}
$params{'ts_log_options'}$addcomma\n";
        }
        $cmd .= "alter database enable public
thread $thrno;\n";
    }
    &dump1("*sql");
    &dump0("*wait");

# Build data dictionary
    &dump0("# building data dictionary");
    $cmd .= "set termout off\n";
    $cmd .= "set echo off\n";
    $cmd .= '@' . "$params{'dd_sql_area'}" .
"catalog.sql;\n";
    $cmd .= '@' . "$params{'dd_sql_area'}" .
"catparr.sql;\n";
    $cmd .= '@' . "$params{'dd_sql_area'}" .
"catproc.sql;\n";
    $cmd .= "connect system/manager\n";
    $cmd .= '@' . "$params{'dd_sql_area'}" .
"utlxplan.sql;\n";
    $cmd .= '@' . "$params{'dd_sqlplus_area'}" .
"pupbld.sql;\n";
    &dump1("*sql");
    &dump0("*wait");
    &time0("Done database creation");

# prepare for multi-user
#^^sd
    $cmd .= "shutdown;\n";
    &dump1("*sql");
    &dump0("*wait");
    &startdb("dbcre");
}                               # end dbcre

sub shutd
{

&dump0("#######################################
#################################");
    &dump0("# Shutdown Database - All
Instances");
    &shutdb();
    &time0("Done database shutdown - all
instances");
}

sub start
{

&dump0("#######################################
#################################");
    &dump0("# Startup Database - All
Instances");
    &startdb($nextphase);
    &time0("Done database startup - all
instance");
}

sub sdgen
{

&dump0("#######################################
#################################");
    &dump0("# Data (Seed File) Generation
Phase");
    &time0("Begin creating seed files");
    @load_tablelist = split(/,/,
$params{'load_tables'});
    foreach $table (@load_tablelist)
    {
        $bgopt = "";
        $dbgprc  =
$params{'load_dbgen_'.$table.'_option_C'};
        if (! $seen{$dbgprc}++) {
                $dbgopt = $verbose ? "-v " : "";
                $dbgopt  .= "-O s -s " .
$params{'scale_factor'};
                $cmd .= $dbgopt . " -C " .
$dbgprc;
                $cmd .= sprintf(" 2>>
dbgen_seed_%d.log", $dbgprc) if $log_output;
                $cmd .= "\n";
                &dump1("*dbgen");
                &time0("Done creating seed files
for degree $dbgprc");
        }
    }
    &dump0("*wait");
    &time0("Done creating seed files");
    &dump0("*wait");
} # end of sdgen

sub dbgen
{

&dump0("#######################################
#################################");
    &dump0("# Data (Flat File) Generation
Phase");
    &time0("Begin creating flat files");

# change DSS_PATH
```

```
    $cmd = "setenv DSS_PATH
$params{'load_flatfile_area'}";
    &dump1("*sh");

    $cur_inst = 1 if $multi;  # for possible
locality on SP2
    ($params{'load_type'} =~ /delim/) ? ($ud =
1) : ($ud = 0);
    $params{'load_tables'} =
$params{'tab_tables'} if !defined
$params{'load_tables'};
    @load_tablelist = split(/,/,
$params{'load_tables'});

    foreach $table (@load_tablelist)
    {
        @load_flatfile_area_list = split(/,/,
$params{'load_flatfile_area_'.$table}) if
(defined
$params{'load_flatfile_area_'.$table});
        $curts = $params{'tab_'.$table.'_ts'};
        $usels = 0;
        $lfexist = 0;
        $bfexist = 0;
        $dafexist = 0;
        $difexist = 0;
        $ffexist = 0;

       # see if we are using pipes
       $up = $params{'load_use_pipes'} =~
/$table/ ? 1 : 0;

        $dp =
$params{'tab_'.$table.'_load_degpar'};
       $dgp = ($params{'load_dbgen_partition'}
=~ /$table/) ? 1 : 0;
       $dbgpar =
$params{'load_dbgen_'.$table.'_option_C'};

# create all of the "executable" load sections
# These variables will hold for all loaders for
a given table
        $upwd = "";
        $parbool = "";
        $dirbool = "";
        $silent = "";
        $dismax = "";
       $file = "";
        $errors = "";
        $rows = "";
        $bsize = "";
        $load = "";
        $skip = "";
       $dbgopt = "";
       $dbgtab = "";

       if ($table =~ /lineitem/)
       {
           $dbgtab = "L";
       }
       elsif ($table =~ /orders/)
       {
           $dbgtab = "O";
       }
       elsif ($table =~ /partsupp/)
       {
           $dbgtab = "S";
       }
       elsif ($table =~ /part/)
       {
           $dbgtab = "P";
       }
       elsif ($table =~ /customer/)
       {
           $dbgtab = "c";
       }
       elsif ($table =~ /supplier/)
       {
           $dbgtab = "s";
       }
       elsif ($table =~ /nation/)
       {
           $dbgtab = "n";
       }
       elsif ($table =~ /region/)
       {
           $dbgtab = "r";
       }

       $dbgopt .= "-T ".$dbgtab." ";
       $dbgopt .= "-s ". $params{'scale_factor'}
. " ";
       $dbgopt .= "-C ";
       $dbgopt .=
$params{'load_dbgen_'.$table.'_option_C'};
       # if using pipes, create the pipes

       $nperset = 1;

       if ($dgp && $dp > 1)
       {
           $nset =
($params{'tab_'.$table.'_#part'} > $dp) ? 1
               : int($dp /
$params{'tab_'.$table.'_#part'});
           $nperset = int($dbgpar / $nset);
       }
       if (defined @load_flatfile_area_list) {
         $number_of_flatfile_areas =
@load_flatfile_area_list;
         if ($number_of_flatfile_areas <
$dbgpar) {
             print "load_flatfile_area_list has
to few entries\n";
             print "set it to at least ".
$dbgpar." elements\n";
             print "baling out ...\n";
          }
       }
       for ($i=0;$i<$dbgpar;$i++)
       {
           # create script to generate flat
files for each dbgpar
           $cmd = sprintf("%s -S %d", $dbgopt,
($i + 1));
           if (defined
$params{'load_flatfile_area_'.$table}) {
               $fpre =
@load_flatfile_area_list[$i].$params{'load_dbgen
_'.$table.'_output_prefix'};
           }
           else {
              $fpre =
$params{'load_flatfile_area'}.$params{'load_dbge
n_'.$table.'_output_prefix'};
            }
           if ($dgp)
           {
              # create script to generate
partitioned flat files

              # if the output prefix name is
specified with #
```

```
            # divide output file names into
$dp/#numpart sets
            if ($fpre =~ /\#/)
            {
                $snum = 1;
                $snum = (int($i/$nperset) +
1) if ($dp > 1);
                $fpre =~ s/\#/$snum/g;
            }
            # I commented this out to enable
support for fixed dbgen (fixed to 84 partitions)
            $cmd .= " -p -I ";
            $cmd .=
$params{'load_dbgen_'.$table.'_input_params'};
            $cmd .= sprintf(" -o %s_%d ",
$fpre,($i + 1));
            # the folowing is the new stuff
            #$cmd = "-D ".$cmd;

          } else {
            # $cmd .= sprintf(" -o $fpre");
this is only temporary
          }
          $cmd = $verbose ? "-v -f " . $cmd :
"-f " . $cmd;
#          $cmd .= " & ";
          $cmd .= "\n";
          &dump1("*dbgen");
       }
       &dump0("*wait");
       &time0("Done creating flat files for
table $table");
    }
    &dump0("*wait");
    &time0("Done creating flat files");
} # end of dbgen


sub dbgen_old
{

&dump0("#######################################
####################################");
    &dump0("# Data (Flat File) Generation
Phase");
    &time0("Begin creating flat files");

    $cur_inst = 1 if $multi;  # for possible
locality on SP2
    ($params{'load_type'} =~ /delim/) ? ($ud =
1) : ($ud = 0);
    $params{'load_tables'} =
$params{'tab_tables'} if !defined
$params{'load_tables'};
    @load_tablelist = split(/,/,
$params{'load_tables'});
    foreach $table (@load_tablelist)
    {
       $dbgprc =
$params{'load_dbgen_'.$table.'_option_C'};
        $curts = $params{'tab_'.$table.'_ts'};
        $dp =
$params{'tab_'.$table.'_load_degpar'};
        $usels = 0;
        $lfexist = 0;
        $bfexist = 0;
        $dafexist = 0;
        $difexist = 0;
        $ffexist = 0;

       # see if we are using pipes
       $up = $params{'load_use_pipes'} =~
/$table/ ? 1 : 0;

       $dgp = ($params{'load_dbgen_partition'}
=~ /$table/) ? 1 : 0;
       $dbgpar =
$params{'load_dbgen_'.$table.'_option_C'};

# create all of the "executable" load sections
# These variables will hold for all loaders for
a given table
        $upwd = "";
        $parbool = "";
        $dirbool = "";
        $silent = "";
        $dismax = "";
       $file = "";
        $errors = "";
        $rows = "";
        $bsize = "";
        $load = "";
        $skip = "";
       $dbgopt = "";
       $dbgtab = "";

       if ($table =~ /lineitem/)
       {
           $dbgtab = "L";
       }
       elsif ($table =~ /orders/)
       {
           $dbgtab = "O";
       }
       elsif ($table =~ /partsupp/)
       {
           $dbgtab = "S";
       }
       elsif ($table =~ /part/)
       {
           $dbgtab = "P";
       }
       elsif ($table =~ /customer/)
       {
           $dbgtab = "c";
       }
       elsif ($table =~ /supplier/)
       {
           $dbgtab = "s";
       }
       elsif ($table =~ /nation/)
       {
           $dbgtab = "n";
       }
       elsif ($table =~ /region/)
       {
           $dbgtab = "r";
       }

       $dbgopt .= "-T ".$dbgtab." ";
       $dbgopt .= "-s ". $params{'scale_factor'}
. " ";
       $dbgopt .= "-C ";
       $dbgopt .= ((defined
$params{'load_dbgen_'.$table.'_option_C'}) ?

$params{'load_dbgen_'.$table.'_option_C'} :
           $params{'load_dbgen_def_option_C'});
       if ($dgp)
       {
           $dbgopt .= " -p -i ";
           $dbgopt .= (defined
$params{'load_dbgen_'.$table.'_input_params'}) ?
```

```
$params{'load_dbgen_'.$table.'_input_params'} :
$params{'load_dbgen_def_input_params'};
        }

        # Start - Barry N. Perkins 8/27/1998
#
        # included the following lines to copy
flat files#
        # from default location to backup
location       #
        # Defined a new variable for backup
location     #
        $fbkp = (defined
$params{'load_dbgen_flatfile_backup'}) ?
            $params{'load_dbgen_flatfile_backup'}
:
            0;
        $bpre = (defined
$params{'load_dbgen_backup_'.$table.'_prefix'})
?

$params{'load_dbgen_backup_'.$table.'_prefix'} :
            $params{'load_dbgen_backup_area'} .
$table;
        # End   - Barry N. Perkins 8/27/1998
#
        # if using pipes, create the pipes
        if($dgp)
        {
            # create script to generate
partitioned flat files for each dbgpar

            $nperset = 1;

            if ($dgp && $dp > 1)
            {
                $nset =
($params{'tab_'.$table.'_#part'} > $dp) ? 1 :
int($dp / $params{'tab_'.$table.'_#part'});
                $nperset = int($dbgpar / $nset);
            }

            for ($i=0;$i<$dbgpar;$i++)
            {
                $cmd = sprintf("%s -S %d",
$dbgopt, ($i + 1));
                if ($dgp)
                {
                    # if the output prefix name
is specified with #
                    # divide output file names
into $dp/#numpart sets
                    $fpre = (defined
$params{'load_dbgen_'.$table.'_output_prefix'})
?

        $params{'load_dbgen_'.$table.'_output_pre
fix'} :

        $params{'load_flatfile_area'} . $table;
                    if ($fpre =~ /\#/)
                    {
                        $snum = 1;
                        $snum = (int($i/$nperset)
+ 1) if ($dp > 1);
                        $fpre =~ s/\#/$snum/g;
                    }
                    $cmd .= sprintf(" -o %s_%d ",
$fpre,($i + 1));
                }
                $cmd = $verbose ? "-v -f " . $cmd
: "-f " . $cmd;
                $cmd .= sprintf(" 2>> %s_%d.log",
$table, ($i + 1)) if $log_output;
#               $cmd .= " & ";
                $cmd .= "\n";
                &dump1("*dbgen");
            }
#           &dump0("*wait");
        }
        else
        {
            $cmd = $verbose ? "-v " : "";
            $cmd .= "-f " . $dbgopt;
            $cmd .= sprintf(" 2>> %s.log",
$table) if $verbose;
            $cmd .= "\n";
            &dump1("*dbgen");
        }
        # Start - Barry N. Perkins 8/27/1998
#
        # Included the following lines to copy
flat files#
        # from default location to backup
location       #
        # Copy from default location to backup
location  #
        if ($fbkp) {
            $cmd = sprintf("copy %s%s*.*
%s*.*\n", $fpre, $table, $bpre);
            &dump1("*sh");
            &dump0("*wait");
        }
        # End   - Barry N. Perkins 8/27/1998
#
        &time0("Done creating flat files for
table $table");
    }
    &dump0("*wait");
    &time0("Done creating flat files");
} # end of dbgen

sub dbgen_my_split_version # I will delete this
and dbgen_nt and dbgen_unix as soon as new dbgen
is fully tested
{
    if (($os cmp "unix") ==0)
    {
        &dbgen_unix();
    }
    else
    {
        &dbgen_nt();
    }
}

sub dbgen_nt
{

&dump0("#######################################
#####################################");
    &dump0("# Data (Flat File) Generation
Phase");
    &time0("Begin creating flat files");

    $cur_inst = 1 if $multi;  # for possible
locality on SP2
    ($params{'load_type'} =~ /delim/) ? ($ud =
1) : ($ud = 0);
    $params{'load_tables'} =
$params{'tab_tables'} if !defined
$params{'load_tables'};
    @load_tablelist = split(/,/,
$params{'load_tables'});
```

```
    foreach $table (@load_tablelist)
    {
        $curts = $params{'tab_'.$table.'_ts'};
        $dp =
$params{'tab_'.$table.'_load_degpar'};
        $usels = 0;
        $lfexist = 0;
        $bfexist = 0;
        $dafexist = 0;
        $difexist = 0;
        $ffexist = 0;
       $up = 0;
       $dgp = 0;
       $dbgpar = 1;

       # see if we are using pipes

       $up = 1 if ($params{'load_use_pipes'} =~
/$table/);
       $dgp = 1 if
($params{'load_dbgen_partition'} =~ /$table/);
       $dbgpar =
$params{'load_dbgen_def_option_C'} if defined
$params{'load_dbgen_def_option_C'};
       $dbgpar =
$params{'load_dbgen_'.$table.'_option_C'} if
defined
$params{'load_dbgen_'.$table.'_option_C'};


# create all of the "executable" load sections
# These variables will hold for all loaders for
a given table
        $upwd = "";
        $parbool = "";
        $dirbool = "";
        $silent = "";
        $dismax = "";
       $file = "";
        $errors = "";
        $rows = "";
        $bsize = "";
        $load = "";
        $skip = "";
       $dbgopt = "";
       $dbgtab = "";


           if ($table =~ /lineitem/)
           {
               $dbgtab = "L";
           }
           elsif ($table =~ /orders/)
           {
               $dbgtab = "O";
           }
           elsif ($table =~ /partsupp/)
           {
               $dbgtab = "S";
           }
           elsif ($table =~ /part/)
           {
               $dbgtab = "P";
           }
           elsif ($table =~ /customer/)
           {
               $dbgtab = "c";
           }
           elsif ($table =~ /supplier/)
           {
               $dbgtab = "s";
           }
           elsif ($table =~ /nation/)
           {
               $dbgtab = "n";
           }
           elsif ($table =~ /region/)
           {
               $dbgtab = "r";
           }

       if ($up) {
           $dbgopt .= "-T ".$dbgtab." ";
           $dbgopt .= "-s ".
$params{'scale_factor'} . " ";
           $dbgopt .= "-C ";
           $dbgopt .= ((defined
$params{'load_dbgen_'.$table.'_option_C'}) ?

       $params{'load_dbgen_'.$table.'_option_C'}
:

       $params{'load_dbgen_def_option_C'});
           if ($dgp) {
               $dbgopt .= " -p -i ";
               $dbgopt .= (defined
$params{'load_dbgen_'.$table.'_input_params'}) ?
$params{'load_dbgen_'.$table.'_input_params'} :
$params{'load_dbgen_def_input_params'};
           }
       }
       else
       {
           $dbgopt .= "-T ".$dbgtab." ";
           $dbgopt .= "-s ".
$params{'scale_factor'} . " ";
           $dbgopt .= "-C ";
           $dbgopt .= ((defined
$params{'load_dbgen_'.$table.'_option_C'}) ?

       $params{'load_dbgen_'.$table.'_option_C'}
:

       $params{'load_dbgen_def_option_C'});
       }

       # if using pipes, create the pipes

       if($up)
       {
           # create script to generate
partitioned flat files for each dbgpar

           $nperset = 1;

           if ($dgp && $dp > 1) {
               $nset = int($dp /
$params{'tab_'.$table.'_#part'});
               $nperset = int($dbgpar / $nset);
           }

           for ($i=0;$i<$dbgpar;$i++) {
               $cmd = sprintf("%s -S %d",
$dbgopt, ($i + 1));
               if ($dgp) {
                   # if the output prefix name
is specified with #
                   # divide output file names
into $dp/#numpart sets
                   $fpre = (defined
$params{'load_dbgen_'. $table.'_output_prefix'})
? $params{'load_dbgen_'.$table.'_output_prefix'}
: $params{'load_dbgen_def_output_prefix'};
                   if ($fpre =~ /\#/) {
```

```
                    $snum = 1;
                    $snum = (int($i/$nperset)
+ 1) if ($dp > 1);
                    $fpre =~ s/\#/$snum/g;
                }
                $cmd .= sprintf(" -o %s_%d ",
$fpre,($i + 1));
            }

            $cmd = " -f " . $cmd;
#           $cmd .= " & ";

            $cmd .= "\n";
            &dump1("*dbgen");
        }

        &dump0("*wait");

         # create script to concatenate
partitioned flat files for each dbgpar into
partitioned flat files by partitions and number
of partitioned files in each partition

         if ($dgp) {
            $nptn =
$params{'tab_'.$table.'_#part'};
            for ($i=0;$i<$dbgpar;$i++) {
                # if the output prefix name
is specified with #
                # divide output file names
into $dp/#numpart sets
                $fpre = (defined
$params{'load_dbgen_'. $table.'_output_prefix'})
? $params{'load_dbgen_'.$table.'_output_prefix'}
: $params{'load_dbgen_def_output_prefix'};
                if ($fpre =~ /\#/) {
                    $snum = 1;
                    $snum = (int($i/$nperset)
+ 1) if ($dp > 1);
                    $fpre =~ s/\#/$snum/g;
                }
                $pnum = ($i % $nptn) + 1;
                $cmd = sprintf("copy %s_*.%d
%s.%d\n", $fpre, $pnum, $fpre, $pnum);
                &dump1("*sh");
                &dump0("*wait");
                $cmd = sprintf("del
%s_*.%d\n", $fpre, $pnum);
                &dump1("*sh");
                &dump0("*wait");
            }
        }
       }
       else
        {
            $cmd = "-f " . $dbgopt;
            $cmd .= "\n";
            &dump1("*dbgen");
        }
       &dump0("*wait");
       &time0("Done creating flat files for
table $table");
    }
    &dump0("*wait");
    &time0("Done creating flat files");
} # dbgen_nt


sub dbgen_unix
{
&dump0("#####################################
##################################");
    &dump0("# Database Population Phase");

    $cur_inst = 1 if $multi;  # for possible
locality on SP2
    ($params{'load_type'} =~ /delim/) ? ($ud =
1) : ($ud = 0);
    $params{'load_tables'} =
$params{'tab_tables'} if !defined
$params{'load_tables'};
    @load_tablelist = split(/,/,
$params{'load_tables'});
    foreach $table (@load_tablelist)
    {
        $curts = $params{'tab_'.$table.'_ts'};
        $dp =
$params{'tab_'.$table.'_load_degpar'};
        $usels = 0;
        $lfexist = 0;
        $bfexist = 0;
        $dafexist = 0;
        $difexist = 0;
        $ffexist = 0;
        $up = 0;
        $dgp = 0;
        $dbgpar = 1;

        # see if we are using pipes

        $up = 1 if ($params{'load_use_pipes'} =~
/$table/);
# <bug> parts/partsupp can be confused
        $dgp = 1 if
($params{'load_dbgen_partition'} =~ /$table/);
        $dbgpar =
$params{'load_dbgen_def_option_C'} if defined
$params{'load_dbgen_def_option_C'};
        $dbgpar =
$params{'load_dbgen_'.$table.'_option_C'} if
defined
$params{'load_dbgen_'.$table.'_option_C'};

        $params{'tab_'.$table.'_load_ctlf'} =
$table.'.ctl' if !defined
$params{'tab_'.$table.'_load_ctlf'};
        @ctlf = split(/,/,
$params{'tab_'.$table.'_load_ctlf'});
        $err = "ctl" if ((@ctlf > 1) && (@ctlf
!= $dp));
        @logf = split(/,/,
$params{'tab_'.$table.'_load_logf'});
        $err = "log" if ((@logf > 1) && (@logf
!= $dp));
        $lfexist = 1 if @logf > 0;
        @badf = split(/,/,
$params{'tab_'.$table.'_load_badf'});
        $err = "bad" if ((@badf > 1) && (@badf
!= $dp));
        $bfexist = 1 if @badf > 0;
        @datf = split(/,/,
$params{'tab_'.$table.'_load_datf'});
        $err = "dat" if ((@datf > 1) && (@datf
!= $dp));
        $dafexist = 1 if @datf > 0;
        @disf = split(/,/,
$params{'tab_'.$table.'_load_disf'});
        $err = "dis" if ((@disf > 1) && (@disf
!= $dp));
        $difexist = 1 if @diff > 0;
        @filf = ();
```

```
#        $params{'tab_'.$table.'_load_filf'} =
$params{$curts.'_datafiles'} if
($params{'tab_'.$table.'_load_filf'} =~
/alltsdatafiles/);
        if ($params{'tab_'.$table.'_load_filf'}
=~ /alltsdatafiles/)
        {
          $params{'tab_'.$table.'_load_filf'} =
$params{$curts.'_datafiles'}
        }
        @filf = split(/,/,
$params{'tab_'.$table.'_load_filf'}) if defined
$params{'tab_'.$table.'_load_filf'};
       # relax the requirement a little bit by
allowing round robining
       $numfil = @filf;
#      if (($numfil > 1) && ($numfil < $dp) &&
(($dp % $numfil) == 0)) {
       if (($numfil > 1) && ($numfil < $dp)) {
#           $p = $dp / $numfil;
            $fil = "";
            for ($i = 0; $i < $dp; $i++)
            {
                if ($i == 0) {
                    $fil =
$params{'tab_'.$table.'_load_filf'};
                } else {
                    $fil =
join(',',$fil,$params{'tab_'.$table.'_load_filf'
});
                }
            }
            @filf = split(/,/, $fil);
       }
        $err = "fil" if ((@filf > 1) && ($dp !=
@filf));
        $ffexist = 1 if @filf > 0;
        if ($err)
        {
            print "Error with load parameters
for table $table\n";
            print "  degree is $dp, $err list
has bad number of elements\n";
            return;
        }
        $usels = 1 if ((($dp > 1) && (@datf ==
1) && $params{'tab_'.$table.'_load_datf'} !~
/\#/));

# expand all of the xxxf arrays to arrays of
size $dp
        @ff = ();
        for ($i=0; $i < $dp; $i++)
        {
            $iplus = $i + 1;
            if (@ctlf > 1)
            {
                @cf[$i] = @ctlf[$i];
            }
            else
            {
                @cf[$i] = @ctlf[0];
                @cf[$i] =~ s/\#/$iplus/g;
            }
            if (@logf > 1)
            {
                @lf[$i] = @logf[$i];
            }
            else
            {
                @lf[$i] = @logf[0];
                @lf[$i] =~ s/\#/$iplus/g;
            }
            if (@badf > 1)
            {
                @bf[$i] = @badf[$i];
            }
            else
            {
                @bf[$i] = @badf[0];
                @bf[$i] =~ s/\#/$iplus/g;
            }
            if (@datf > 1)
            {
                @daf[$i] = @datf[$i];
            }
            else
            {
               # if load_degpar is larger than
the number partitions
               # and we are using pipes,
               # then load_degpar has to be a
multiple of the number
               # of partition and
load_degpar/#partitions sets of
               # pipes will be setup.
               # The goal is to maintain only
one pipe/flatfile per
               # sqlldr.

               if
(($params{'tab_'.$table.'_#part'} > 0) &&
                    ($dp >
$params{'tab_'.$table.'_#part'})) {
                    if ($dp %
$params{'tab_'.$table.'_#part'} == 0) {
                        if (@datf[0] =~ /\#.*\#/)
{
                            # set number

$snum=int($i/$params{'tab_'.$table.'_#part'}) +
1;
                            # partiton number

$pnum=$i%$params{'tab_'.$table.'_#part'} + 1;
                            @daf[$i] = @datf[0];
                            @daf[$i] =~
s/\#/$snum/;
                            @daf[$i] =~
s/\#/$pnum/;
                        } else {
                            @daf[$i] = @datf[0];
                            @daf[$i] =~
s/\#/$iplus/g;
                        }
                    }
                    else
                    {
                        print "Error: #partitons
for table $table does not divide load_degpar for
the table.\n";
                        print "degree is $dp,
number of partitions is
$params{'tab_'.$table.'_#part'}.\n";
                        return;
                    }
                }
                else
                {
                    # assign
                    @daf[$i] = @datf[0];
                    @daf[$i] =~ s/\#/$iplus/g;
                }
            }
```

```
            if (@disf > 1)
            {
                @dif[$i] = @disf[$i];
            }
            else
            {
                @dif[$i] = @disf[0];
                @dif[$i] =~ s/\#/$iplus/g;
            }
            if (@filf > 1)
            {
                @ff[$i] = @filf[$i];
            }
            else
            {
                # round robining - temp fix
                @ff[$i] = @filf[0] if (@filf);
                if (defined
$params{$params{'tab_'.$table.'_ts'}.'_#files'})
{
                   $numfil =
$params{$params{'tab_'.$table.'_ts'}.'_#files'};
                   $filnum = $i % $numfil + 1;

                   @ff[$i] =~ s/\#/$filnum/g;
                } else {
                   @ff[$i] =~ s/\#/$iplus/g;
                }
            }
        }

        $control = sprintf ("%s%s",
$params{'load_controlfile_area'}, @cf[0]);
        $control =~ s/\?/$ENV{'ORACLE_HOME'}/g;

        $ff_path =
$params{'load_dbgen_'.$table.'_output_prefix'};
        $numchild =
$params{'load_dbgen_'.$table.'_option_C'};
        $pname =
$params{'tab_'.$table.'_partnames'};
        @pname = split(/,/, $pname);
# create the controlfiles (fixed-length fields
or delimited records)
#       if ($params{'skip_mk_ldctlfs'} !~
/$table/)
#        {
          $npart = (defined
$params{'tab_'.$table.'_#part'}) ?
$params{'tab_'.$table.'_#part'} : 1;
          for ($ictl=1; $ictl<=$npart; $ictl++)
{
            open (CTLFILE, ">$control");
            print CTLFILE "---\n";
            print CTLFILE "---  $table.ctl for
delimited records\n" if ($ud == 1);
            print CTLFILE "---  $table.ctl for
fixed-length fields\n" if ($ud == 0);
            print CTLFILE "---\n\n";
            if (!($pre72) && defined
$params{'tab_'.$table.'_loadextent'})
            {
              print CTLFILE "options\n";
              print CTLFILE "(\n";
              print CTLFILE "storage = (initial
$params{'tab_'.$table.'_loadextent'} next
$params{'tab_'.$table.'_loadextent'})\n";
              print CTLFILE ")\n";
            }
           print CTLFILE "unrecoverable\n" if
($params{'load_unrecoverable'} =~
/[tT][rR][uU][eE]/);
            print CTLFILE "load\n";
            print CTLFILE "-- This is where
INFILE should go.\n";
            if ((!($up)) && ($dbgpar)) {
              for
($jpart=1;$jpart<=$numchild;$jpart++) {
                 print CTLFILE "INFILE
'".$ff_path.$jpart.".".$ictl."'\n";
              }
            }
            print CTLFILE
"$params{'load_insert_type'}\n";
            print CTLFILE "into table
$table\n";
            if ($dgp) {
              print CTLFILE "partition
($pname[$ictl-1])\n";
            }
            print CTLFILE
"$params{'load_insert_type'}\n";
            print CTLFILE "fields terminated by
$params{'load_field_terminator'}\n" if ($ud ==
1);
            print CTLFILE "(\n";
            @tab_collist = split(/,/,
$params{'tab_'.$table.'_columns'});
            while ($col = shift(@tab_collist))
            {
              (@tab_collist == 0) ? ($addcomma
= "") : ($addcomma = ",");
              $pos = "";
              $pos = sprintf ("position
(%s)",$params{'tab_'.$table.'_'.$col.'_pos'}) if
($ud == 0);
              $ctlline = sprintf ("  %-20s %s
%s$addcomma", $col, $pos,
$params{'tab_'.$table.'_'.$col.'_loadcolx'});
              print CTLFILE "$ctlline\n";
            }
            print CTLFILE ")\n";
            close (CTLFILE);
            if ($npart > 1) {
              $newi=$ictl+1;
              $control =~
s/$table$ictl/$table$newi/;
            }
          }
#       }

# create all of the "executable" load sections
# These variables will hold for all loaders for
a given table
        $upwd = "";
        $parbool = "";
        $dirbool = "";
        $silent = "";
        $dismax = "";
        $file = "";
        $errors = "";
        $rows = "";
        $bsize = "";
        $load = "";
        $skip = "";
        $dbgopt = "";
        $dbgtab = "";

        if ($up) {
            if ($table =~ /lineitem/)
            {
                $dbgtab = "L";
            }
            elsif ($table =~ /orders/)
```

```
            {
                $dbgtab = "O";
            }
            elsif ($table =~ /partsupp/)
            {
                $dbgtab = "S";
            }
            elsif ($table =~ /part/)
            {
                $dbgtab = "P";
            }
            elsif ($table =~ /customer/)
            {
                $dbgtab = "c";
            }
            elsif ($table =~ /supplier/)
            {
                $dbgtab = "s";
            }
            elsif ($table =~ /nation/)
            {
                $dbgtab = "n";
            }
            elsif ($table =~ /region/)
            {
                $dbgtab = "r";
            }
            $dbgopt .= "-T ".$dbgtab." ";
            $dbgopt .= "-s ".
$params{'scale_factor'} . " ";
            $dbgopt .= "-C ";
            $dbgopt .= ((defined
$params{'load_dbgen_'.$table.'_option_C'}) ?

        $params{'load_dbgen_'.$table.'_option_C'}
:

        $params{'load_dbgen_def_option_C'});
            if ($dgp) {
                $dbgopt .= " -p -i ";
                $dbgopt .= (defined
$params{'load_dbgen_'.$table.'_input_params'}) ?
$params{'load_dbgen_'.$table.'_input_params'} :
$params{'load_dbgen_def_input_params'};
            }
        }

        $upwd =
"$params{'user'}/$params{'passwd'}";
        $parbool = " parallel=true" if
(($params{'tab_'.$table.'_load_parallel'} =~
/true/) || ($dp > 1));
        $dirbool = " direct=true" if
(($params{'tab_'.$table.'_load_direct'} =~
/[Tt][Rr][Uu][Ee]/) || $parbool);
        $silent = "
silent=$params{'tab_'.$table.'_load_silent'}" if
defined $params{'tab_'.$table.'_load_silent'};
        $dismax = "
discardmax=$params{'tab_'.$table.'_load_dismax'}
" if defined
$params{'tab_'.$table.'_load_dismax'};
        $errors = "
errors=$params{'tab_'.$table.'_load_errors'}" if
defined $params{'tab_'.$table.'_load_errors'};
        $rows = "
rows=$params{'tab_'.$table.'_load_rows'}" if
defined $params{'tab_'.$table.'_load_rows'};
        $bsize = "
bindsize=$params{'tab_'.$table.'_load_bsize'}"
if defined $params{'tab_'.$table.'_load_bsize'};

        if ($usels)
        {
            $split =
int($params{'tab_'.$table.'_#rows'} / $dp);
            $extra =
int($params{'tab_'.$table.'_#rows'} % $dp);
            $loadval = $split;
            $skipval = 0;
        }

       # if using pipes, create the pipes

       if($up)
       {
           for ($i=0;$i<$dp;$i++) {
               $cmd = sprintf("rm -f
%s%s\nmknod %s%s
p\n",$params{'load_flatfile_area'},@daf[$i],$par
ams{'load_flatfile_area'},@daf[$i]);
#              $cmd = sprintf("%s%s
p\n",$params{'load_flatfile_area'},@daf[$i]);
               &dump1("*sh");
           }
           &dump0("*wait");

           # start the dbgens

           if ($dgp) {
               $nset = int($dp /
$params{'tab_'.$table.'_#part'});
               $nperset = int($dbgpar / $nset);
           }

           for ($i=0;$i<$dbgpar;$i++) {
               $cmd =
$params{'load_flatfile_area'} . "\n";
               $cmd2 = sprintf("-f %s -S %d",
$dbgopt, ($i + 1));
               if ($dgp) {
                   # if the output prefix name
is specified with #
                   # divide output file names
into $dp/#numpart sets
                   $fpre = (defined
$params{'load_dbgen_'. $table.'_output_prefix'})
? $params{'load_dbgen_'.$table.'_output_prefix'}
: $params{'load_dbgen_def_output_prefix'};
                   if ($fpre =~ /\#/) {
                       $snum = int($i/$nperset) +
1;
                       $fpre =~ s/\#/$snum/g;
                   }
                   $cmd2 .= sprintf(" -o %s ",
$fpre);
               }
               $cmd2 .= "\n";
               &dump2("*dbgen");
           }
       }

       $control = "";
        $log = "";
        $bad = "";
        $data = "";
        $discard = "";
        $file = "";
        $load = "";
        $loadval = "";
        $skip = "";

        for ($i=0; $i < $dp; $i++)
        {
```

```
            &advmulti();

            $control = sprintf (" control=%s%s",
$params{'load_controlfile_area'}, @cf[$i]);
            $control = sprintf ("
control=/host%s%s",
$params{'load_controlfile_area'}, @cf[$i]) if
($params{'special_machine'} eq 'ncube');
            $log = sprintf (" log=%s%s",
$params{'load_otherfile_area'}, @lf[$i]) if
$lfexist;
            $bad = sprintf (" bad=%s%s",
$params{'load_otherfile_area'}, @bf[$i]) if
$bfexist;
            $data = sprintf (" data=%s%s",
$params{'load_flatfile_area'}, @daf[$i]) if
$dafexist;
            $discard = sprintf (" discard=%s%s",
$params{'load_otherfile_area'}, @dif[$i]) if
$difexist;
            $file = sprintf (" file=%s%s",
$params{$curts.'_area'}, @ff[$i]) if $ffexist;

            if ($usels)
            {
                $loadval = $loadval + 1 if
($extra > $i);
                $load = sprintf (" load=%d",
$loadval);
                $skip = sprintf (" skip=%d",
$skipval);
                $skipval = $skipval + $loadval;
                $loadval = $split;
            }
            if ($up)
            {
                $load = " load=99999999";
            }

            $cmd = sprintf
("%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n",$upwd,$con
trol,$log,$bad,$data,$discard,$dismax,$skip,$loa
d,$errors,$rows,$bsize,$silent,$dirbool,$parbool
,$file);
            &dump1("*load");
            &dump0("*wait") if ((!$parbool) &&
(!defined $params{'load_no_waits'}));
        }
        &dump0("*wait") if !defined
$params{'load_no_waits'};

       if($up)
       {
           for ($i=0;$i<$dp;$i++) {
               $cmd = sprintf("rm -f
%s%s\n",$params{'load_flatfile_area'},@daf[$i]);
               &dump1("*sh");
           }
           &dump0("*wait");
       }
   }
}


sub plcre
{

&dump0("#######################################
####################################");
    &dump0("# NT Raw Partition and Setlink
Creation Phase");
    &time0("Begin NT Raw Partition and Setlink
creation");

# create NT Raw Partition and Setlink input
files
    $lnksfile =
$params{'plcre_setlinks_input_file'};
    open (LNKSFILE, ">$lnksfile");
    $drivenum = $params{'plcre_drivenums'};
    $md = $drivenum =~ tr/,/,/;
    @drivenum =
split(/,/,$params{'plcre_drivenums'});
if ($params{'plcre_recreate_extended_partitions'
=~ /[tT][rR][uU][eE]/})
    {
        foreach $drivenum (@drivenum)
        {
            &recreate_drive_extended_part;
        }
        if (defined
$params{'plcre_log_drivenum'})
        {
            &recreate_drive_extended_part;
        }
    }
    $io_control_files =
$params{'io_control_files'};
    $io_control_files =~ s/[\(\"\\\.\)]*//g;
    @io_control_files =
split(/,/,$io_control_files);
    $d = 0;
    foreach $file (@io_control_files)
    {
        $size =
$params{'plcre_control_file_size'};
        $size =~ s/[Mm]*//g;
        &create_drive_part_file;
    }
    foreach $ts_entry (@ts_all)
    {
        &create_drive_part("$ts_entry") if
($ts_entry =~ /ts_sys/ || $ts_entry =~
/ts_log/);
    }
    $d = 0;
    foreach $ts_entry (@ts_most)
    {
        &create_drive_part("$ts_entry");
    }
    close (LNKSFILE);
    &dump0("*wait");
    &time0("Done NT Raw Partition creation");
    $cmd = "setlinks /F:" . $lnksfile . "\n";
    &dump1("*sh");
    &dump0("*wait");
    &time0("Done Setlink links creation");
    &time0("Done NT Raw Partition and Setlink
creation");
} # end of plcre



sub sccre
{

&dump0("#######################################
####################################");
    &dump0("# Schema Creation Phase");
    &time0("Begin schema creation");
```

```
# Create user
    if ($params{'skip_create_user'} !~ /true/)
    {
        &dump0("# creating $params{'user'}
user");
        $cmd .= "drop user $params{'user'}
cascade;\n";
        $cmd .= "grant $params{'privileges'}\n";
        $cmd .= "  to $params{'user'} identified
by $params{'passwd'};\n";
        $cmd .= "@?/rdbms/admin/utlxplan;\n" if
(($os cmp "unix") == 0);
       &dump1("*sql");
    }
    &dump0("*wait");

# Create data tablespaces (including datafiles
and tables)
    &dump0("# creating data tablespaces,
datafiles, and tables");

# create tablespaces with initial datafile
    foreach $ts_entry (@ts_data)
    {
        &create_ts("$ts_entry");
    }
    foreach $ts_entry (@ts_index)
    {
        &create_ts("$ts_entry");
    }
    foreach $ts_entry (@ts_temp)
    {
        $ts_temporary = "temporary" if !$pre73;
        &create_ts("$ts_entry");
    }
    &dump0("*wait");

# create remaining datafiles for the tablespaces
    foreach $ts_entry (@ts_data)
    {
        &add_dfs("$ts_entry");
    }
    foreach $ts_entry (@ts_index)
    {
        &add_dfs("$ts_entry");
    }
    if ($params{'skip_ts'} !~ /temp/)
    {
        foreach $ts_entry (@ts_temp)
        {
            &add_dfs("$ts_entry");
        }
    }
    &dump0("*wait");

    if ($params{'skip_create_tables'} !~ /true/)
    {
        &create_clusters();
        &dump0("*wait");
        &create_objects('tab',@tab_list) if
defined(@tab_list);
        &dump0("*wait");
    }

    if ($params{'skip_ts'} !~ /temp/)
    {
# Alter user's temporary tablespace
        &dump0("# altering $params{'user'}'s
temporary tablespace");
        $ts_entry = shift (@ts_temp);
        $cmd = "alter user $params{'user'}
temporary tablespace $ts_entry;\n";
        &dump1("*sql");
        &dump0("*wait");
    }
    if ($params{'skip_ts'} !~ /data/)
    {
# Alter user's default tablespace
        &dump0("# altering $params{'user'}'s
default tablespace");
        $ts_entry = shift (@ts_data);
        $cmd = "alter user $params{'user'}
default tablespace $ts_entry;\n";
        &dump1("*sql");
        &dump0("*wait");
    }
    &time0("Done schema creation");
}


sub sctso
{

    if ($phaselist !~ /sccre/)
    {
        &dump0("##################################
##########################################");
        &dump0("# Schema Creation Phase -
datafiles only (no tables or users)");

# Create data tablespaces (including datafiles)
        &dump0("# creating data tablespaces,
datafiles");

# create tablespaces with initial datafile
        foreach $ts_entry (@ts_data)
        {
            &create_ts("$ts_entry");
        }
        foreach $ts_entry (@ts_index)
        {
            &create_ts("$ts_entry");
        }
        foreach $ts_entry (@ts_temp)
        {
            $ts_temporary = "temporary" if
!$pre73;
            &create_ts("$ts_entry");
        }
        &dump0("*wait");

# create remaining datafiles for the tablespaces
        foreach $ts_entry (@ts_data)
        {
            &add_dfs("$ts_entry");
        }
        foreach $ts_entry (@ts_index)
        {
            &add_dfs("$ts_entry");
        }
        if ($params{'skip_ts'} !~ /temp/)
        {
            foreach $ts_entry (@ts_temp)
            {
                &add_dfs("$ts_entry");
            }
        }
        &dump0("*wait");

        if ($params{'skip_ts'} !~ /temp/)
        {
# Alter user's temporary tablespace
#           &dump0("# altering $params{'user'}'s
temporary tablespace");
```

```
#           $ts_entry = shift (@ts_temp);
#           $cmd = "alter user $params{'user'}
temporary tablespace $ts_entry;\n";
#           &dump1("*sql");
#           &dump0("*wait");
        }
    }
}


sub scuto
{
    if ($phaselist !~ /sccre/)
    {

&dump0("#######################################
##################################");
        &dump0("# Schema Creation Phase - User
and Tables ONLY (no datafiles)");
       &time0("Begin creating user and tables
(no datafiles)");

        if ($params{'skip_create_user'} !~
/true/)
        {
            &dump0("# creating $params{'user'}
user");
            $cmd .= "drop user $params{'user'}
cascade;\n";
            $cmd .= "grant
$params{'privileges'}\n";
            $cmd .= "  to $params{'user'}
identified by $params{'passwd'};\n";
            &dump1("*sql");
        }
        &dump0("*wait");

        &create_clusters();
        &dump0("*wait");
        &create_objects('tab',@tab_list) if
defined(@tab_list);
        &dump0("*wait");

        if ($params{'skip_ts'} !~ /temp/)
        {
# Alter user's temporary tablespace
            &dump0("# altering $params{'user'}'s
temp tablespace");
            $ts_entry = shift (@ts_temp);
            $cmd = "alter user $params{'user'}
temporary tablespace $ts_entry;\n";
            &dump1("*sql");
            &dump0("*wait");
        }
# Alter user's default tablespace
        if ($params{'skip_ts'} !~ /data/)
        {
            &dump0("# altering $params{'user'}'s
default tablespace");
            $ts_entry = shift (@ts_data);
            $cmd = "alter user $params{'user'}
default tablespace $ts_entry;\n";
            &dump1("*sql");
            &dump0("*wait");
        }
       $cmd .= "connect tpcd/tpcd\n";
       $cmd .= '@' . "$params{'dd_sql_area'}" .
"utlxplan.sql;\n";
       &dump1("*sql");
       &dump0("*wait");
       &time0("Done creating user and tables (no
datafiles)");
    }
}


sub scuvo
{
    if ($phaselist =~ /scuvo/)
    {

&dump0("#######################################
##################################");
        &dump0("# Schema Creation Phase - Views
ONLY (no datafiles)");

       if (defined(@tablelog_list))
       {

&dump0("#######################################
##################################");
           &dump0("# First I will create the
materialized log for the base tables");

&create_objects('viewlog',@tablelog_list);
           &dump0("*wait");
       }

&dump0("#######################################
##################################");
        &dump0("# Now I can create the views
with the corresponding materialized logs");

       $cmd .= "connect
$params{'user'}/$params{'passwd'};\n";

       foreach $view (@view_list)
       {
           @viewlogs =
split(/,/,$params{'view_'.$view.'_viewlogs'});
           foreach $viewlog (@viewlogs)
           {
               $ob_type = 'viewlog';
               $object = $viewlog;
               if (!(defined
$params{'created_'.$object}) )
               {
                   $params{'created_'.$object} =
$object;
                   &create_object;
               }
           }
           $ob_type = 'view';
           $object = $view;
           &create_object;
       }
       &dump1("*sql");




#       &create_objects('view',@view_list) if
defined(@view_list);
#      &create_objects('viewlog',@viewlog_list)
if defined(@tablelog_list);

        &dump0("*wait")
       }
}

sub dsffs
{
```

```
&dump0("########################################
####################################");
    &dump0("# Flatfile Distribution Phase");
    &time0("Begin data population");

    # &recycle_db() if
($params{'startup_db_dapop'});

    $cur_inst = 1 if $multi;  # for possible
locality on SP2
    ($params{'load_type'} =~ /delim/) ? ($ud =
1) : ($ud = 0);

}
sub dapop # Meikel's version
{

&dump0("########################################
####################################");
    &dump0("# Database Population Phase");
    &time0("Begin data population");

    # &recycle_db() if
($params{'startup_db_dapop'});


    # find out how many nodes we have
    if (!(defined(@ops_nodes))) {
        $nnodes=1
    } else {
        $nnodes= ($#ops_nodes + 1);
    }

    $cur_inst = 1 if $multi;  # for possible
locality on SP2
    ($params{'load_type'} =~ /delim/) ? ($ud =
1) : ($ud = 0);
    $params{'load_tables'} =
$params{'tab_tables'} if !defined
$params{'load_tables'};
    @load_tablelist = split(/,/,
$params{'load_tables'});
    foreach $table (@load_tablelist)
    {
        &time0("Begin $table load");
        $curts = $params{'tab_'.$table.'_ts'};

# see if we are using pipes

        $up = $params{'load_use_pipes'} =~
/$table/ ? 1 : 0;

        $dp =
$params{'tab_'.$table.'_load_degpar'};
        $dgp = ($params{'load_dbgen_partition'}
=~ /$table/) ? 1 : 0;
        $dbgpar =
$params{'load_dbgen_'.$table.'_option_C'};

# These variables will hold for all loaders for
a given table
        $upwd = "";
        $parbool = "";
        $dirbool = "";
        $silent = "";
        $dismax = "";
        $errors = "";
        $rows = "";
        $bsize = "";
        $load = "";
        $skip = "";
        $dbgopt = "";
        $dbgtab = "";

        $upwd =
"$params{'user'}/$params{'passwd'}";
        $parbool = " parallel=true" if
(($params{'tab_'.$table.'_load_parallel'} =~
/true/) || ($dp > 1));
        $dirbool = " direct=true" if
(($params{'tab_'.$table.'_load_direct'} =~
/[Tt][Rr][Uu][Ee]/) || $parbool);
        $silent = "
silent=$params{'tab_'.$table.'_load_silent'}" if
defined $params{'tab_'.$table.'_load_silent'};
        $dismax = "
discardmax=$params{'tab_'.$table.'_load_dismax'}
" if defined
$params{'tab_'.$table.'_load_dismax'};
        $errors = "
errors=$params{'tab_'.$table.'_load_errors'}" if
defined $params{'tab_'.$table.'_load_errors'};
        $rows = "
rows=$params{'tab_'.$table.'_load_rows'}" if
defined $params{'tab_'.$table.'_load_rows'};
        $bsize = "
bindsize=$params{'tab_'.$table.'_load_bsize'}"
if defined $params{'tab_'.$table.'_load_bsize'};

        $ff_path =
$params{'load_flatfile_area'}.$params{'load_dbge
n_'.$table.'_output_prefix'};
        $pname =
$params{'tab_'.$table.'_partnames'};
        $pname =~ s/\#//g;
        print $pname."--".@pname;
        @pname = split(/\,/,$pname);
        $npart = ((defined
$params{'tab_'.$table.'_#part'}) && $dgp) ?
            $params{'tab_'.$table.'_#part'} : 1;
        $ctlarea =
$params{'load_controlfile_area'};
        $logarea =
$params{'load_otherfile_area'};

        if ($dbgpar*$npart <= $dp)
        {
            $dpmax = $dbgpar*$npart;
            $numchild = 1;
            $remchild = 0;
        }
        else
        {
            $dpmax = $dp;
            $numchild =
int(($dbgpar*$npart)/$dp); # numchild is the
number of infiles of one loader
            $remchild = ($dbgpar*$npart) -
($numchild * $dp);
            #print "dp numchild remchild ".$dp."
".$numchild." ".$remchild."\n";
            $loaders_p_part=$dp/$npart;
            #print "dp npart loaders per
partition ".$loaders_p_part." ".$dp."
".$npart."\n";

$ninfiles=int($dbgpar/$loaders_p_part);
            $ninfiles_last=$ninfiles+$dbgpar-
($ninfiles * $loaders_p_part);
            #print "infiles lastloader
".$ninfiles." ".$ninfiles_last."\n";
        }
```

```
       # if we load on an ops we want to load
each partition only on one node

       $npartitions_per_node = int($npart /
$nnodes); # #partition/#nodes
       $rempartitions = $npart -
($npartitions_per_node * $nnodes); #reminder of
the partitions
       print "nnodes $nnodes npart $npart table
$table npartitions_per_node
$npartitions_per_node rempartitions
$rempartitions\n";

       $node = 0;
        $idbgpar = 0;
       $ipart=1;
       $iipart=1;
       if ($dp > $npart) {
           print "$dp $npart\n";
           $sqlldr_per_partition = int ($dp /
$npart); # load_degpar / #partitions
           print "** dp $dp npart $npart
sqlldr_per_partition $sqlldr_per_partition\n";
           $infiles_per_sqlldr = int ($dbgpar /
$sqlldr_per_partition ); #_C /
sqlldr_per_partition
           $sqlldr_per_partition_rest = $dbgpar
- $sqlldr_per_partition * $infiles_per_sqlldr;
           print "** dbgpar $dbgpar
sqlldr_per_partition $sqlldr_per_partition\n";
           print "** sqlldr_per_partition
$sqlldr_per_partition sqlldr_per_partition_rest
$sqlldr_per_partition_rest infiles_per_sqlldr
$infiles_per_sqlldr \n";
       }
       else{
            $sqlldr_per_partition = 1;
            $infiles_per_sqlldr = int ($dbgpar /
$sqlldr_per_partition ); #_C /
sqlldr_per_partition
        }

       $partcount = 1;
       if (defined(@ops_nodes)){
           &sqlldr_ctl_ops; # this is a special
controlfile generation for OPS
       }
       else {
       for ($idp=1; $idp<=$dpmax; $idp++)
       {
           if ($partcount >
$sqlldr_per_partition) {
               $partcount = 1;
           }
           if (defined(@ops_nodes)){
               $node = $node + 1;
               if ($node > $#ops_nodes+1) {
                   $node = 1;
               }
           }
           else {
               $node = "";
           }
           #$idbgpar = 0;
# create all of the "executable" load sections
           #$ipart= int((($idp-
1)*$numchild)/$dbgpar)+1; # this is the first
partition of the new series
           &advmulti();
           $ctlfile =
$params{'tab_'.$table.'_load_ctlf'};
           $ctlfile =~ s/\#/$idp/g;
           $control = $ctlarea.$ctlfile;
           if ($dgp) # load this table in
partition mode
           {
               #$control = sprintf ("
%s%s_%d_%d.ctl", $ctlarea, $table, $idp,
$ipart);
               $log = sprintf ("
%s%s_%d_%d.log", $logarea, $table, $idp,
$ipart);
           }
           else
           {
               if ($dbgpar == 1) #dbgpar is
option _C for this table
               {
                   #$control = sprintf ("
%s%s.ctl", $ctlarea, $table);
                   $log = sprintf (" %s%s.log",
$logarea, $table);
               }
               else
               {
                   #$control = sprintf ("
%s%s_%d.ctl", $ctlarea, $table, $idp);
                   $log = sprintf ("
%s%s_%d.log", $logarea, $table, $idp);
               }
           }
           $cmd = sprintf ("%s control=%s%s%s
log=%s%s%s%s%s%s%s\n",

$upwd,$control,$skip,$load,$log,$errors,$rows,

$bsize,$silent,$dirbool,$parbool);
           &dump1("*load$node");
           &dump0("*wait") if ((!$parbool) &&
(!defined $params{'load_no_waits'}));

# create the controlfiles (fixed-length fields
or delimited records)
           if ($params{'skip_mk_ldctlfs'} !~
/$table/)
           {
               &load_ctl_head;
               $infiles = 0;
               $badfiles = 0;
               $discardfiles = 0;
               $intopartitions = "BEGINNING"; #
this will collect the partition numbers that
will be served.
               if ($dp < $npart) {
                   if ($idp <= $remchild) {
                       $nchild = $numchild+1;
                   }else{
                       $nchild = $numchild;
                   }
               }else{
                   if ($partcount <=
$sqlldr_per_partition_rest) {
                       $nchild =
$infiles_per_sqlldr+1;
                   }else{
                       $nchild =
$infiles_per_sqlldr;
                   }
               }
               for ($jpart=1; $jpart<=$nchild;
$jpart++)
               {
                   #print "ipart $ipart\n";
```

```
                $idbgpar = 0 if $idbgpar >=
$dbgpar;
                #$ipart = int((($idp-
1)*$nchild+$jpart)/$dbgpar);
                #$ipart = $ipart+1 if
((((($idp-1)*$nchild+$jpart)-$ipart*$dbgpar)>0);
                if ($dgp)
                {
                    $pn=@ipart[$node];
                    $partitionname =
$pname.$pn;
                    if (!($intopartitions =~
/$partitionname/)) {
                        if ($intopartitions =~
/BEGINNING/)
                        {
                            $intopartitions =
$partitionname;
                        }
                        else {
                            $intopartitions .=
",".$partitionname;}
                    }
                    @infile[$infiles++] = "'"
. $ff_path . "_" . ++$idbgpar .
                        "." . $ipart . "'";
                    @badfile[$badfiles++] =
"'" . $ff_path . "_" . $idbgpar .
                        "_" . $ipart .
".bad'";

       @discardfile[$discardfiles++] = "'" .
$ff_path . "_" . $idbgpar .
                        "_" . $ipart .
".dsc'";
                    #if ($remchild-- > 0)
                    #{
                    #    @infile[$infiles++] =
"'" . $ff_path . "_" . ++$idbgpar .
                    #       "." . $ipart .
"'";
                    #    @badfile[$badfiles++]
= "'" . $ff_path . "_" . $idbgpar .
                    #       "_" . $ipart .
".bad'";
                    #
@discardfile[$discardfiles++] = "'" . $ff_path .
"_" . $idbgpar .
                    #       "_" . $ipart .
".dsc'";
                    #}
                }
                elsif ($dbgpar == 1)
                {
                    @infile[$infiles++] = "'"
. $ff_path . ".tbl'";
                    @badfile[$badfiles++] =
"'" . $ff_path . ".bad'";

       @discardfile[$discardfiles++] = "'" .
$ff_path . ".dsc'";
                }
                else
                {
                    @infile[$infiles++] = "'"
. $ff_path . ".tbl." . ++$idbgpar . "'";
                    @badfile[$badfiles++] =
"'" . $ff_path . "_" . $idbgpar . ".bad'";

       @discardfile[$discardfiles++] = "'" .
$ff_path . "_" . $idbgpar .
                        ".dsc'";
                    if ($remchild-- > 0)
                    {
                        @infile[$infiles++] =
"'" . $ff_path . ".tbl." . ++$idbgpar . "'";
                        @badfile[$badfiles++]
= "'" . $ff_path . "_" . $idbgpar . ".bad'";

@discardfile[$discardfiles++] = "'" . $ff_path .
"_" . $idbgpar .
                            ".dsc'";
                    }
                }
                #print "vorher jpart $jpart
ipart $ipart iipart $iipart dbgpar $dbgpar\n";
                if ($iipart >= $dbgpar){
                    $iipart=0;
                    $ipart=$ipart+1;
                }
                $iipart=$iipart+1;
                #print "nacher jpart $jpart
ipart $ipart iipart $iipart dbgpar $dbgpar\n";
            }
            --$infiles;
            if ($up)
            {
                print CTLFILE "INFILE ";
                for ($fpart=0;
$fpart<=$infiles; $fpart++)
                {
                    print CTLFILE
@infile[$fpart] . ", " if $fpart < $infiles;
                    print CTLFILE
@infile[$fpart] . " " if $fpart == $infiles;
                }
                print CTLFILE "BADFILE " .
@badfile[0] . " ";
                print CTLFILE "DISCARDFILE "
. @discardfile[0] . "\n";
            }
            else
            {
                for ($fpart=0;
$fpart<=$infiles; $fpart++)
                {
                    print CTLFILE "INFILE " .
@infile[$fpart] . " ";
                    print CTLFILE "BADFILE " .
@badfile[$fpart] . " ";
                    print CTLFILE "DISCARDFILE
" . @discardfile[$fpart] . "\n";
                }
            }
            print CTLFILE
"$params{'load_insert_type'}\n";
            print CTLFILE "into table
$table\n";
            if ($dgp)
            {
                #print CTLFILE "partition
($pname$ipart)\n" if
!($params{'tab_'.$table.'_parttype'} =~ /hash/);
                print CTLFILE "partition
($intopartitions)\n" if
!($params{'tab_'.$table.'_parttype'} =~ /hash/);
            }
            &load_ctl_tail;
        }
        $partcount = $partcount + 1;
    }
    &dump0("*wait") if !defined
$params{'load_no_waits'} && $dgp;
```

```
       &time0("End of load for Partition: $ipart
for Table: $table") if ($dgp);

        &dump0("*wait") if !defined
$params{'load_no_waits'};
    }   &time0("End $table load");
    }
    &dump0("*wait");
    &time0("Done data population");
} # end of dapop


sub sqlldr_ctl_ops
{
    $controlfiles_per_node = $dpmax / $nnodes;
    print "controlfiles_per_node
$controlfiles_per_node nnodes $nnodes\n";
    for ($i=1; $i<=$nnodes; $i++){
        @ipart[$i] = (($i-
1)*$npartitions_per_node)+1;
        @iipart[$i] = 1;
        print "Offset for node $i is:
@ipart[$i]\n";
    }
    for ($idpp=1; $idpp<=$controlfiles_per_node;
$idpp++) {
    for ($node=1; $node<=$nnodes; $node++)
    {
        print "idpp $idpp node $node
infiles_per_sqlldr $infiles_per_sqlldr\n";
        $idp = (($node-1)*$controlfiles_per_node)
+ $idpp;
        if ($partcount > $sqlldr_per_partition) {
            $partcount = 1;
        }
        #if (defined(@ops_nodes)){
        #    $node = $node + 1;
        #    if ($node > $#ops_nodes+1) {
        #       $node = 1;
        #    }
        #}
        #else {
        #    $node = "";
        #}
        #$idbgpar = 0;
# create all of the "executable" load sections
        #@ipart[$node]= int((($idp-
1)*$numchild)/$dbgpar)+1; # this is the first
partition of the new series
        &advmulti();
        $ctlfile =
$params{'tab_'.$table.'_load_ctlf'};
        $ctlfile =~ s/\#/$idp/g;
        $control = $ctlarea.$ctlfile;
        if ($dgp) # load this table in partition
mode
        {
            #$control = sprintf ("
%s%s_%d_%d.ctl", $ctlarea, $table, $idp,
@ipart[$node]);
            $log = sprintf (" %s%s_%d_%d.log",
$logarea, $table, $idp, @ipart[$node]);
        }
        else
        {
            if ($dbgpar == 1) #dbgpar is option
_C for this table
            {
                #$control = sprintf (" %s%s.ctl",
$ctlarea, $table);
                $log = sprintf (" %s%s.log",
$logarea, $table);
            }
            else
            {
                #$control = sprintf ("
%s%s_%d.ctl", $ctlarea, $table, $idp);
                $log = sprintf (" %s%s_%d.log",
$logarea, $table, $idp);
            }
        }
        $cmd = sprintf ("%s control=%s%s%s
log=%s%s%s%s%s%s%s%s\n",

        $upwd,$control,$skip,$load,$log,$errors,$
rows,

        $bsize,$silent,$dirbool,$parbool);
        &dump1("*load$node");
        &dump0("*wait") if ((!$parbool) &&
(!defined $params{'load_no_waits'}));

# create the controlfiles (fixed-length fields
or delimited records)
        if ($params{'skip_mk_ldctlfs'} !~
/$table/)
        {
            &load_ctl_head;
            $infiles = 0;
            $badfiles = 0;
            $discardfiles = 0;
            $intopartitions = "BEGINNING"; # this
will collect the partition numbers that will be
served.
            if ($dp < $npart) {
                if ($idp <= $remchild) {
                    $nchild = $numchild+1;
                }else{
                    $nchild = $numchild;
                }
            }else{
                if ($partcount <=
$sqlldr_per_partition_rest) {
                    $nchild =
$infiles_per_sqlldr+1;
                }else{
                    $nchild =
$infiles_per_sqlldr;
                }
            }
            for ($jpart=1; $jpart<=$nchild;
$jpart++)
            {
                #print "ipart @ipart[$node]\n";
                $idbgpar = 0 if $idbgpar >=
$dbgpar;
                #@ipart[$node] = int((($idp-
1)*$nchild+$jpart)/$dbgpar);
                #@ipart[$node] = @ipart[$node]+1
if ((($idp-1)*$nchild+$jpart)-
@ipart[$node]*$dbgpar)>0);
                if ($dgp) # load in partition
mode
                {
                    print @ipart."\n";
                    $pn=@ipart[$node];
                    $partitionname = $pname.$pn;
                    if (!($intopartitions =~
/$partitionname/)) {
                        if ($intopartitions =~
/BEGINNING/)
                        {
                            $intopartitions =
$partitionname;
```

```
                    }
                    else {
                        $intopartitions .=
","."$partitionname;}
                    }
                    @infile[$infiles++] = "'" .
$ff_path . "_" . ++$idbgpar .
                        "." . @ipart[$node] . "'";
                    @badfile[$badfiles++] = "'" .
$ff_path . "_" . $idbgpar .
                        "_" . @ipart[$node] .
".bad'";
                    @discardfile[$discardfiles++]
= "'" . $ff_path . "_" . $idbgpar .
                        "_" . @ipart[$node] .
".dsc'";
                    #if ($remchild-- > 0)
                    #{
                    #    @infile[$infiles++] =
"'" . $ff_path . "_" . ++$idbgpar .
                    #       "." .
@ipart[$node] . "'";
                    #    @badfile[$badfiles++] =
"'" . $ff_path . "_" . $idbgpar .
                    #   "_" . @ipart[$node] .
".bad'";
                    #
@discardfile[$discardfiles++] = "'" . $ff_path .
"_" . $idbgpar .
                    #   "_" . @ipart[$node] .
".dsc'";
                    #}
                }
                elsif ($dbgpar == 1)
                {
                    @infile[$infiles++] = "'" .
$ff_path . ".tbl'";
                    @badfile[$badfiles++] = "'" .
$ff_path . ".bad'";
                    @discardfile[$discardfiles++]
= "'" . $ff_path . ".dsc'";
                }
                else
                {
                    @infile[$infiles++] = "'" .
$ff_path . ".tbl." . ++$idbgpar . "'";
                    @badfile[$badfiles++] = "'" .
$ff_path . "_" . $idbgpar . ".bad'";
                    @discardfile[$discardfiles++]
= "'" . $ff_path . "_" . $idbgpar .
                        ".dsc'";
                    if ($remchild-- > 0)
                    {
                       @infile[$infiles++] = "'"
. $ff_path . ".tbl." . ++$idbgpar . "'";
                        @badfile[$badfiles++] =
"'" . $ff_path . "_" . $idbgpar . ".bad'";

       @discardfile[$discardfiles++] = "'" .
$ff_path . "_" . $idbgpar .
                            ".dsc'";
                    }
                }
                #print "vorher jpart $jpart ipart
@ipart[$node] iipart @iipart[$node] dbgpar
$dbgpar\n";
                if (@iipart[$node] >= $dbgpar){
                    @iipart[$node]=0;

@ipart[$node]=@ipart[$node]+1;
                }
                @iipart[$node]=@iipart[$node]+1;
                #print "nacher jpart $jpart ipart
@ipart[$node] iipart @iipart[$node] dbgpar
$dbgpar\n";
            }
            --$infiles;
            if ($up)
            {
                print CTLFILE "INFILE ";
                for ($fpart=0; $fpart<=$infiles;
$fpart++)
                {
                    print CTLFILE @infile[$fpart]
. ", " if $fpart < $infiles;
                    print CTLFILE @infile[$fpart]
. " " if $fpart == $infiles;
                }
                print CTLFILE "BADFILE " .
@badfile[0] . " ";
                print CTLFILE "DISCARDFILE " .
@discardfile[0] . "\n";
            }
            else
            {
                for ($fpart=0; $fpart<=$infiles;
$fpart++)
                {
                    print CTLFILE "INFILE " .
@infile[$fpart] . " ";
                    print CTLFILE "BADFILE " .
@badfile[$fpart] . " ";
                    print CTLFILE "DISCARDFILE "
. @discardfile[$fpart] . "\n";
                }
            }
            print CTLFILE
"$params{'load_insert_type'}\n";
            print CTLFILE "into table $table\n";
            if ($dgp)
            {
                #print CTLFILE "partition
($pname@ipart[$node])\n" if
!($params{'tab_'.$table.'_parttype'} =~ /hash/);
                print CTLFILE "partition
($intopartitions)\n" if
!($params{'tab_'.$table.'_parttype'} =~ /hash/);
            }
            &load_ctl_tail;
        }
        $partcount = $partcount + 1;
    }
}
    &dump0("*wait") if !defined
$params{'load_no_waits'} && $dgp;
    &time0("End of load for Partition:
@ipart[$node] for Table: $table") if ($dgp);

    &dump0("*wait") if !defined
$params{'load_no_waits'};
    &time0("End $table load");
}



sub dapop_frank ## copied from Frank's version
{

&dump0("#######################################
##################################");
    &dump0("# Database Population Phase");
    &time0("Begin data population");
```

```
    $cur_inst = 1 if $multi;  # for possible
locality on SP2
    ($params{'load_type'} =~ /delim/) ? ($ud =
1) : ($ud = 0);
    $params{'load_tables'} =
$params{'tab_tables'} if !defined
$params{'load_tables'};
    @load_tablelist = split(/,/,
$params{'load_tables'});
    foreach $table (@load_tablelist)
    {
        &time0("Begin $table load");

        $curts = $params{'tab_'.$table.'_ts'};
        $dp =
$params{'tab_'.$table.'_load_degpar'};
        $usels = 0;
        $lfexist = 0;
        $bfexist = 0;
        $dafexist = 0;
        $difexist = 0;
        $ffexist = 0;
       $up = 0;
       $dgp = 0;
       $dbgpar = 1;

       # see if we are using pipes

       $up = 1 if ($params{'load_use_pipes'} =~
/$table/);
       $dgp = 1 if
($params{'load_dbgen_partition'} =~ /$table/);
       $dbgpar =
$params{'load_dbgen_def_option_C'} if defined
$params{'load_dbgen_def_option_C'};
       $dbgpar =
$params{'load_dbgen_'.$table.'_option_C'} if
defined
$params{'load_dbgen_'.$table.'_option_C'};

        $params{'tab_'.$table.'_load_ctlf'} =
$table.'.ctl' if !defined
$params{'tab_'.$table.'_load_ctlf'};
        @ctlf = split(/,/,
$params{'tab_'.$table.'_load_ctlf'});
        $err = "ctl" if ((@ctlf > 1) && (@ctlf
!= $dp));
        @logf = split(/,/,
$params{'tab_'.$table.'_load_logf'});
        $err = "log" if ((@logf > 1) && (@logf
!= $dp));
        $lfexist = 1 if @logf > 0;
        @badf = split(/,/,
$params{'tab_'.$table.'_load_badf'});
        $err = "bad" if ((@badf > 1) && (@badf
!= $dp));
        $bfexist = 1 if @badf > 0;
        @datf = split(/,/,
$params{'tab_'.$table.'_load_datf'});
        $err = "dat" if ((@datf > 1) && (@datf
!= $dp));
        $dafexist = 1 if @datf > 0;
        @disf = split(/,/,
$params{'tab_'.$table.'_load_disf'});
        $err = "dis" if ((@disf > 1) && (@disf
!= $dp));
        $difexist = 1 if @diff > 0;
       @filf = ();

## NT Port
## exapnd all files for all partitions
       if ($params{'tab_'.$table.'_load_filf'}
=~ /alltsdatafiles/ )
       {   if ($params{'tab_'.$table.'_part_ts'}
=~ /^(.*)\#$/ && $dgp == 1)
           {
               $curts = "";
               $params{'tab_'.$table.'_part_ts'}
=~ /^(.*)\#$/;
               $tsnam = $1;

       $params{'tab_'.$table.'_load_filf'} = "";
               $numts =
$params{'tab_'.$table.'_#part'};
               $numfil =
$params{$params{'tab_'.$table.'_ts'}.'_#files'};

               for ($j = 1; $j <= $numts; $j++)
               {
                   $tspnam =
sprintf("%s%d",$tsnam,$j);
                   if (defined
$params{$tspnam.'_areas'})
                   {
                       $tsareas =
$params{$tspnam.'_area'} . "," .
$params{$tspnam.'_areas'};
                   }
                   else
                   {
                       $tsareas =
$params{$tspnam.'_area'};
                       for ($i = 1; $i < $numfil;
$i++)
                       {
                           $tsareas = $tsareas .
"," . $params{$tspnam.'_area'};
                       }
                    }
                    $tsarea[$j] = [ split(/,/,
$tsareas) ];
               }

               for ($i = 1; $i <= $numfil; $i++)
               {
                   for ($j = 1; $j <= $numts;
$j++)
                   {
                       $tspnam =
sprintf("%s%d",$tsnam,$j);
                       (($i == $numfil) && ($j ==
$numts)) ?
                           ($addcomma = "") :
($addcomma = ",");
                       if ($tsarea[$j][$i-1] !~
/\\\\.\\/)
                       {
                          $nextfile =
sprintf("%s%s_%d.dbf%s",$tsarea[$j][$i-
1],$tspnam,$i,$addcomma);
                       }
                       else
                       {
# Temporay walkaround for 804 & SQLLDR bugs
#                          $nextfile =
sprintf("%s%s_%d%s",$tsarea[$j][$i-
1],$tspnam,$i,$addcomma);
                          if (defined
$params{'tab_'.$table.'_part_lfn'})
                          {
                             $nextfile =
sprintf("\\\\\\\\.\\\\%s%d_%d%s",$params{'tab_'.
$table.'_part_lfn'},$j,$i,$addcomma);
```

```
                        }
                        else
                        {
                          $nextfile =
sprintf("\\\\\\\\\.\\\\%s_%d%s",$tspnam,$i,$addco
mma);
                        }
                      }

$params{'tab_'.$table.'_load_filf'} .=
$nextfile;
                  }
                }
            }
            else
            {

$params{'tab_'.$table.'_load_filf'} =
$params{$curts.'_datafiles'};
            }
        }
        elsif
($params{'tab_'.$table.'_load_filf'} =~
/^(.*)\#$/ )
            {
                $curts = "";

        $params{'tab_'.$table.'_load_filf'} =~
/^(.*)\#$/;
                $tsnam = $1;

        $params{'tab_'.$table.'_load_filf'} = "";
                $tsareas = $params{$tsnam.'area'}
. "," . $params{$tsnam.'areas'};
                @tsarea = split(/,/, $tsareas);
                $numfil = @tsarea;
                @dfile = split(/,/,
$params{$tsnam.'datafiles'});
                $numfil = @dfile if ($numfil <
@dfile);
                for ($i = 0; $i < $numfil; $i++)
                {
                    ($i == $numfil - 1) ?
($addcomma = "") : ($addcomma = ",");
                    if (@tsarea[0] !~ /\\\\.\\/)
                    {
                      $nextfile =
sprintf("%s%s.dbf%s",@tsarea[$i],$dfile[$i],$add
comma);
                    }
                    else
                    {
# Temporay workaround of 804 and SQLLDR bugs
#                     $nextfile =
sprintf("%s%s%s",@tsarea[$i],$dfile[$i],$addcomm
a);
                      $nextfile =
sprintf("\\\\\\\\\.\\\\%s%s",$dfile[$i],$addcomma
);

                    }

$params{'tab_'.$table.'_load_filf'} .=
$nextfile;
                }
            }

## NT End

        @filf = split(/,/,
$params{'tab_'.$table.'_load_filf'}) if defined
$params{'tab_'.$table.'_load_filf'};

       # relax the requirement a little bit by
allowing round robining
       $numfil = @filf;
## NT Port (fix the case for partition tables
which has only one file in each partition
##
##--
##      if (($numfil > 1) && ($numfil < $dp) &&
(($dp % $numfil) == 0)) {
##          $p = $dp / $numfil;
##--
        if ((($numfil > 1) && ($numfil < $dp) &&
(($dp % $numfil) == 0)) ||
            (($dgp == 1) && ($dp == 1)))
        {
            $p = 1;
            $p = ($dp / $numfil) if ($dp > 1);
## NT End
            $fil = "";
            for ($i = 0; $i < $p; $i++)
            {
                if ($i == 0) {
                    $fil =
$params{'tab_'.$table.'_load_filf'};
                } else {
                    $fil =
join(',',$fil,$params{'tab_'.$table.'_load_filf'
});
                }
            }
            @filf = split(/,/, $fil);
        }

        $err = "fil" if ((@filf > 1) && ($dp !=
@filf) && ($dp != 1));
        $ffexist = 1 if @filf > 0;
        if ($err)
        {
            print "Error with load parameters
for table $table\n";
            print "  degree is $dp, $err list
has bad number of elements\n";
            return;
        }
        $usels = 1 if ((($dp > 1) && (@datf ==
1) && $params{'tab_'.$table.'_load_datf'} !~
/\#/));

# expand all of the xxxf arrays to arrays of
size $dp

        @ff = ();

        $dp = @filf if (($dp == 1) && ($dgp ==
1));

        for ($i=0; $i < $dp; $i++)
        {
            $iplus = $i + 1;
            if (@ctlf > 1)
            {
                @cf[$i] = @ctlf[$i];
            }
            else
            {
                @cf[$i] = @ctlf[0];
                @cf[$i] =~ s/\#/$iplus/g;
            }
            if (@logf > 1)
            {
                @lf[$i] = @logf[$i];
            }
```

```
            else
            {
                @lf[$i] = @logf[0];
                @lf[$i] =~ s/\#/$iplus/g;
            }
            if (@badf > 1)
            {
                @bf[$i] = @badf[$i];
            }
            else
            {
                @bf[$i] = @badf[0];
                @bf[$i] =~ s/\#/$iplus/g;
            }
            if (@datf > 1)
            {
                @daf[$i] = @datf[$i];
            }
            else
            {
               # if load_degpar is larger than
the number partitions
               # and we are using pipes,
               # then load_degpar has to be a
multiple of the number
               # of partition and
load_degpar/#partitions sets of
               # pipes will be setup.
               # The goal is to maintain only
one pipe/flatfile per
               # sqlldr.

               if
(($params{'tab_'.$table.'_#part'} > 0) &&
                   ($dp >
$params{'tab_'.$table.'_#part'})) {
                  if ($dp %
$params{'tab_'.$table.'_#part'} == 0) {
                     if (@datf[0] =~ /\#.*\#/)
{
                        # set number

$snum=int($i/$params{'tab_'.$table.'_#part'}) +
1;
                        # partiton number

$pnum=$i%$params{'tab_'.$table.'_#part'} + 1;
                        @daf[$i] = @datf[0];
                        @daf[$i] =~
s/\#/$snum/;
                        @daf[$i] =~
s/\#/$pnum/;
                     } else {
                        @daf[$i] = @datf[0];
                        @daf[$i] =~
s/\#/$iplus/g;
                     }
                  }
                  else
                  {
                     print "Error: #partitons
for table $table does not divide load_degpar for
the table.\n";
                     print "degree is $dp,
number of partitions is
$params{'tab_'.$table.'_#part'}.\n";
                     return;
                  }
               }
               else
               {
                   # assign
                        @daf[$i] = @datf[0];
                        @daf[$i] =~ s/\#/$iplus/g;
                   }
               }
               if (@disf > 1)
               {
                   @dif[$i] = @disf[$i];
               }
               else
               {
                   @dif[$i] = @disf[0];
                   @dif[$i] =~ s/\#/$iplus/g;
               }
               if (@filf > 1)
               {
                   @ff[$i] = @filf[$i];
               }
               else
               {
                   # round robining - temp fix
                   @ff[$i] = @filf[0] if (@filf);
                   if (defined
$params{$params{'tab_'.$table.'_ts'}.'_#files'})
{
                      $numfil =
$params{$params{'tab_'.$table.'_ts'}.'_#files'};
#MP                   print "$table \n";
#MP                   print
"$params{$params{'tab_'.$table.'_ts'}.'_#files'}
\n";
                      $filnum = $i % $numfil + 1;

                      @ff[$i] =~ s/\#/$filnum/g;
                   } else {
                      @ff[$i] =~ s/\#/$iplus/g;
                   }
               }
           }

           $control = sprintf ("%s%s",
$params{'load_controlfile_area'}, @cf[0]);
           $control =~ s/\?/$ENV{'ORACLE_HOME'}/g;

           # add code to create head and tail of
control files

           $controlh = sprintf ("%s%s.head",
$params{'load_controlfile_area'}, $table);
           $controlh =~ s/\?/$ENV{'ORACLE_HOME'}/g;

           $controlt = sprintf ("%s%s.tail",
$params{'load_controlfile_area'}, $table);
           $controlt =~ s/\?/$ENV{'ORACLE_HOME'}/g;

           open (CTLFILEH, ">$controlh");
           open (CTLFILET, ">$controlt");
           print CTLFILEH "---\n";
           if (!($pre72) && defined
$params{'tab_'.$table.'_loadextent'})
               {
                  print CTLFILEH "options\n";
                   print CTLFILEH "(\n";
                   print CTLFILEH "storage =
(initial $params{'tab_'.$table.'_loadextent'}
next $params{'tab_'.$table.'_loadextent'})\n";
                   print CTLFILEH ")\n";
               }

#          print CTLFILE "unrecoverable\n" if
($params{'load_unrecoverable'} =~
/[tT][rR][uU][eE]/);
```

```
       print CTLFILEH "unrecoverable\n" if
($params{'load_unrecoverable'} =~
/[tT][rR][uU][eE]/);

       print CTLFILEH "load\n";
       print CTLFILEH "-- This is where INFILE
should go.\n";
#       print CTLFILEH
"$params{'load_insert_type'}\n";
       print CTLFILEH "into table $table\n";


       print CTLFILET "-- This is where
PARTITION is specified.\n";
       print CTLFILET
"$params{'load_insert_type'}\n";
       print CTLFILET "fields terminated by
$params{'load_field_terminator'}\n" if ($ud ==
1);
       print CTLFILET "(\n";

       @tab_collist = split(/,/,
$params{'tab_'.$table.'_columns'});
       while ($col = shift(@tab_collist))
       {
           (@tab_collist == 0) ? ($addcomma =
"") : ($addcomma = ",");
           $pos = "";
           $pos = sprintf ("position
(%s)",$params{'tab_'.$table.'_'.$col.'_pos'}) if
($ud == 0);
           $ctlline = sprintf ("  %-20s %s
%s$addcomma", $col, $pos,
$params{'tab_'.$table.'_'.$col.'_loadcolx'});
           print CTLFILET "$ctlline\n";
       }
       print CTLFILET ")\n";
       close (CTLFILEH);
       close (CTLFILET);

# create the controlfiles (fixed-length fields
or delimited records)
       if ($params{'skip_mk_ldctlfs'} !~
/$table/)
       {
           #print "control $control\n";
           open (CTLFILE, ">$control");
           print CTLFILE "---\n";
           print CTLFILE "---  $table.ctl for
delimited records\n" if ($ud == 1);
           print CTLFILE "---  $table.ctl for
fixed-length fields\n" if ($ud == 0);
           print CTLFILE "---\n\n";
           if (!($pre72) && defined
$params{'tab_'.$table.'_loadextent'})
           {
               print CTLFILE "options\n";
               print CTLFILE "(\n";
               print CTLFILE "storage =
(initial $params{'tab_'.$table.'_loadextent'}
next $params{'tab_'.$table.'_loadextent'})\n";
               print CTLFILE ")\n";
           }
       print CTLFILE "unrecoverable\n" if
($params{'load_unrecoverable'} =~
/[tT][rR][uU][eE]/);
           print CTLFILE "load\n";
#          print CTLFILE
"$params{'load_insert_type'}\n";
           print CTLFILE "into table $table\n";
           print CTLFILE
"$params{'load_insert_type'}\n";
           print CTLFILE "fields terminated by
$params{'load_field_terminator'}\n" if ($ud ==
1);
           print CTLFILE "(\n";
           @tab_collist = split(/,/,
$params{'tab_'.$table.'_columns'});
           while ($col = shift(@tab_collist))
           {
               (@tab_collist == 0) ? ($addcomma
= "") : ($addcomma = ",");
               $pos = "";
               $pos = sprintf ("position
(%s)",$params{'tab_'.$table.'_'.$col.'_pos'}) if
($ud == 0);
               $ctlline = sprintf ("  %-20s %s
%s$addcomma", $col, $pos,
$params{'tab_'.$table.'_'.$col.'_loadcolx'});
               print CTLFILE "$ctlline\n";
           }
           print CTLFILE ")\n";
           close (CTLFILE);
       }

# create all of the "executable" load sections
# These variables will hold for all loaders for
a given table
       $upwd = "";
       $parbool = "";
       $dirbool = "";
       $silent = "";
       $dismax = "";
       $file = "";
       $errors = "";
       $rows = "";
       $bsize = "";
       $load = "";
       $skip = "";
       $dbgopt = "";
       $dbgtab = "";

       if ($up) {
           if ($table =~ /lineitem/)
           {
              $dbgtab = "L";
           }
           elsif ($table =~ /orders/)
           {
              $dbgtab = "O";
           }
           elsif ($table =~ /partsupp/)
           {
              $dbgtab = "S";
           }
           elsif ($table =~ /part/)
           {
              $dbgtab = "P";
           }
           elsif ($table =~ /customer/)
           {
              $dbgtab = "c";
           }
           elsif ($table =~ /supplier/)
           {
              $dbgtab = "s";
           }
           elsif ($table =~ /nation/)
           {
              $dbgtab = "n";
           }
           elsif ($table =~ /region/)
```

```
            {
                $dbgtab = "r";
            }
            $dbgopt .= "-T ".$dbgtab." ";
            $dbgopt .= "-s ".
$params{'scale_factor'} . " ";
            $dbgopt .= "-C ";
            $dbgopt .= ((defined
$params{'load_dbgen_'.$table.'_option_C'}) ?

        $params{'load_dbgen_'.$table.'_option_C'}
:

        $params{'load_dbgen_def_option_C'});
            if ($dgp) {
                $dbgopt .= " -p -i ";
                $dbgopt .= (defined
$params{'load_dbgen_'.$table.'_input_params'}) ?
$params{'load_dbgen_'.$table.'_input_params'} :
$params{'load_dbgen_def_input_params'};
            }
        }

        $upwd =
"$params{'user'}/$params{'passwd'}";
        $parbool = " parallel=true" if
(($params{'tab_'.$table.'_load_parallel'} =~
/true/) || ($dp > 1));
        $dirbool = " direct=true" if
(($params{'tab_'.$table.'_load_direct'} =~
/[Tt][Rr][Uu][Ee]/) || $parbool);
        $silent = "
silent=$params{'tab_'.$table.'_load_silent'}" if
defined $params{'tab_'.$table.'_load_silent'};
        $dismax = "
discardmax=$params{'tab_'.$table.'_load_dismax'}
" if defined
$params{'tab_'.$table.'_load_dismax'};
        $errors = "
errors=$params{'tab_'.$table.'_load_errors'}" if
defined $params{'tab_'.$table.'_load_errors'};
        $rows = "
rows=$params{'tab_'.$table.'_load_rows'}" if
defined $params{'tab_'.$table.'_load_rows'};
        $bsize = "
bindsize=$params{'tab_'.$table.'_load_bsize'}"
if defined $params{'tab_'.$table.'_load_bsize'};

        if ($usels)
        {
            $split =
int($params{'tab_'.$table.'_#rows'} / $dp);
            $extra =
int($params{'tab_'.$table.'_#rows'} % $dp);
            $loadval = $split;
            $skipval = 0;
        }

## NT Port (use dbgen to create flat files)
##
##      # if using pipes, create the pipes
##
##      if($up)
##      {
##          for ($i=0;$i<$dp;$i++) {
##              $cmd = sprintf("%s%s
p\n",$params{'load_flatfile_area'},@daf[$i]);
##              &dump1("*mknod");
##          }
##          &dump0("*wait");
##
##          # start the dbgens
##
##          if ($dgp) {
##              $nset = int($dp /
$params{'tab_'.$table.'_#part'});
##              $nperset = int($dbgpar / $nset);
##          }
##
##          for ($i=0;$i<$dbgpar;$i++) {
##              $cmd =
$params{'load_flatfile_area'} . "\n";
##              $cmd2 = sprintf("%s -S %d",
$dbgopt, ($i + 1));
##              if ($dgp) {
##                  # if the output prefix name
is specified with #
##                  # divide output file names
into $dp/#numpart sets
##                  $fpre = (defined
$params{'load_dbgen_'. $table.'_output_prefix'})
? $params{'load_dbgen_'.$table.'_output_prefix'}
: $params{'load_dbgen_def_output_prefix'};
##                  if ($fpre =~ /\#/) {
##                      $snum = int($i/$nperset) +
1;
##                      $fpre =~ s/\#/$snum/g;
##                  }
##                  $cmd2 .= sprintf(" -o %s ",
$fpre);
##              }
##
### Added 4/28?  - Ari
### need to force in case pipe still there, and
need to kick
### off in the backgroud from here
##              $cmd2 = " -f " . $cmd2;
##              $cmd2 .= " & ";
##
##              $cmd2 .= "\n";
##              &dump2("*dbgen");
##          }
##      }
##
##      &dump0("*wait");
## NT End

        for ($i=0; $i < $dp; $i++)
        {
            &advmulti();
            $control = "";
            $log = "";
            $bad = "";
            $data="";
            $discard = "";
            $control = sprintf (" control=%s%s",
$params{'load_controlfile_area'}, @cf[$i]);
            $control = sprintf ("
control=/host%s%s",
$params{'load_controlfile_area'}, @cf[$i]) if
($params{'special_machine'} eq 'ncube');
            $log = sprintf (" log=%s%s",
$params{'load_otherfile_area'}, @lf[$i]) if
($lfexist == 1);
            $bad = sprintf (" bad=%s%s",
$params{'load_otherfile_area'}, @bf[$i]) if
($bfexist == 1);
            $data = sprintf (" data=%s%s",
$params{'load_flatfile_area'}, @daf[$i]) if
($dafexist== 1);
            $discard = sprintf (" discard=%s%s",
$params{'load_otherfile_area'}, @dif[$i]) if
($difexist== 1);
            if ($ffexist ==1)
```

```
            {
                if ($curts == "")
                 {
                    $file = sprintf (" file=%s",
@ff[$i]);
                }
                else
                {
                    $file = sprintf ("
file=%s%s", $params{$curts.'_area'}, @ff[$i]);
                }
            }
## NT Port
## Change to UPPER case string as a workaround
for SQLLDR
            $file =~ tr/a-z/A-Z/;
## NT End
            if ($usels)
            {
                 $loadval = $loadval + 1 if
($extra > $i);
                 $load = sprintf (" load=%d",
$loadval);
                 $skip = sprintf (" skip=%d",
$skipval);
                 $skipval = $skipval + $loadval;
                 $loadval = $split;
            }
##          if ($up)
##          {
##              $load = " load=99999999";
##          }

            $cmd = sprintf
("%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n",$upwd,$con
trol,$log,$bad,$data,$discard,$dismax,$skip,$loa
d,$errors,$rows,$bsize,$silent,$dirbool,$parbool
,$file);
            &dump1("*load");
            &dump0("*wait") if ((!$parbool) &&
(!defined $params{'load_no_waits'}));
        }
        &dump0("*wait") if !defined
$params{'load_no_waits'};

##--    if($up)
##      {
##          for ($i=0;$i<$dp;$i++) {
##              $cmd = sprintf("rm -f
%s%s\n",$params{'load_flatfile_area'},@daf[$i]);
##              &dump1("*sh");
##          }
##          &dump0("*wait");
##--    }
        &time0("End $table load");
    }

    $cmd = "shutdown\n"; #^^sd
    &dump1("*sql");
    &dump0("*wait");
    &time0("Done data population");
}



sub ixcre
{

&dump0("#######################################
####################################");
    &dump0("# Index Creation Phase");
    &time0("Begin index creation");

    foreach $index (@indexlist)
    {
        if (defined
$params{'ind_'.$index.'_table'})
        {
            $ob_type = 'table';
            $ob_stype = 'tab';
        }
        else
        {
            $ob_type = 'view';
            $ob_stype = 'view';
        }

        &time0("Begin creating index $index");
        &advmulti();
         $cmd .= "connect
$params{'user'}/$params{'passwd'};\n";
         $cmd .= "drop index $index;\n";
         $cmd .= "create ";
        if ($params{'ind_'.$index.'_unique'} =~
/[tT][rR][uU][eE]/)
        {
            $cmd .= "unique ";
        } elsif ($params{'ind_'.$index.'_bitmap'}
=~ /[tT][rR][uU][eE]/) {
            $cmd .= "bitmap ";
        }
        $cmd .= "index $index\n";
         if (defined
$params{'ind_'.$index.'_'.$ob_type})
         {
             $cmd .= "on
$params{'ind_'.$index.'_'.$ob_type}
($params{'ind_'.$index.'_'.$ob_stype.'cols'})\n"
;
            $indtab =
$params{'ind_'.$index.'_'.$ob_type};
         }
         else
         {
             $cmd .= "on cluster
$params{'ind_'.$index.'_cluster'}\n";
         }
         $cmd .= "pctfree
$params{'ind_'.$index.'_%f'}\n" if defined
$params{'ind_'.$index.'_%f'};
         $cmd .= "initrans
$params{'ind_'.$index.'_it'}\n" if defined
$params{'ind_'.$index.'_it'};
         $cmd .= "maxtrans
$params{'ind_'.$index.'_mt'}\n" if defined
$params{'ind_'.$index.'_mt'};
        if ($params{'compatible'} =~ /7\./)
        {
            $cmd .= "unrecoverable\n" if
($params{'ind_'.$index.'_unrecoverable'} =~
/[tT][rR][uU][eE]/);
        } elsif ($params{'compatible'} =~ /8\./)
{
            $cmd .= "nologging\n" if
($params{'ind_'.$index.'_nolg'} =~
/[tT][rR][uU][eE]/);
        }
        $cmd .= "compute statistics\n" if
($params{'ind_'.$index.'_compstats'} =~
/[tT][rR][uU][eE]/);
         $cmd .= "tablespace
$params{'ind_'.$index.'_ts'}\n" if ((defined
$params{'ind_'.$index.'_ts'}) &&
($params{'skip_ts'} !~ /index/));
```

```
        $cmd .= "storage
$params{'ind_'.$index.'_storage'}\n" if defined
$params{'ind_'.$index.'_storage'};
       $cmd .= "reverse\n" if
($params{'ind_'.$index.'_reverse'} =~
/[tT][rR][uU][eE]/);
        $cmd .= "nosort\n" if
($params{'ind_'.$index.'_nosort'} =~
/[tT][rR][uU][eE]/);

       if ((defined
$params{'ind_'.$index.'_pardeg'}) || (defined
$params{'ind-'.$index.'_parinst'}))
        {
            $cmd .= "parallel";
           if ($params{'ind_'.$index.'_pardeg'}
=~/[dD][eE][fF][aA][uU][lL]/)
           {
               $cmd .= "\n";
           }
           else
           {
               $cmd .= " (degree
$params{'ind_'.$index.'_pardeg'} "
                   if defined
$params{'ind_'.$index.'_pardeg'};
               $cmd .= "instances
$params{'ind_'.$index.'_parinst'}"
                   if defined
$params{'ind_'.$index.'_parinst'};
               $cmd = $cmd . ")\n";
           }
       }

       # deal with partitioning

       if ($params{'ind_'.$index.'_partition'}
=~ /[lL][oO][cC][aA][lL]/)
       {
           $numpart =
$params{$ob_stype.'_'.$indtab.'_#part'};
           $cmd .= "local";
           if (defined
$params{'ind_'.$indtab.'_partnames'})
           {
               &exp_ind_part_l;
           }
           else
           {
               $cmd .= "\n";
           }
       } elsif
($params{'ind_'.$index.'_partition'} =~
/[gG][lL][oO][bB][aA][lL]/) {
           $numpart =
$params{'ind_'.$index.'_#part'};
           $cmd .= "global partition by range
(";
           &exp_ind_part_g;
       }
        $cmd .= ";\n";
        &dump1("*sql");
    }


# Constraints which are enabled after the load
    foreach $const (@constlist)
    {
        &time0("Begin creating constraint
$const");
        &advmulti();
        $cmd .= "connect
$params{'user'}/$params{'passwd'};\n";
        if ($params{'con_'.$const.'_constraint'}
=~ /null/)
        {
            @collist = split(/,/,
$params{'con_'.$const.'_columns'});
            $cmd .= "alter table
$params{'con_'.$const.'_table'} ";
            $cmd .= "modify (";
            while ($col = shift(@collist))
            {
                (@collist == 0) ? ($addcomma =
"") : ($addcomma = ",");
                $cmd .= "$col
$params{'con_'.$const.'_constraint'}$addcomma";
            }
            $cmd .= ");\n";
        }
        else
        {
            $cmd .= "alter table
$params{'con_'.$const.'_table'} drop constraint
$const;\n";
            $cmd .= "alter table
$params{'con_'.$const.'_table'} ";
            $cmd .= "add constraint $const\n";
            $cmd .= "
$params{'con_'.$const.'_constraint'} key
($params{'con_'.$const.'_columns'}) ";
            if
($params{'con_'.$const.'_has_index'} =~
/[Tt][Rr][Uu][Ee]/) {
                $cmd .= "using index";
            }
            if ($params{'con_'.$const.'_disable'}
=~ /[Tt][Rr][Uu][Ee]/) {
                $cmd .= "disable;\n";
            } else {
                $cmd .= ";\n";
            }
            if
($params{'con_'.$const.'_constraint'} =~
/foreign/) {
                $cmd .= "alter table
$params{'con_'.$const.'_table'} ";
                $cmd .= "enable novalidate
primary key;\n";
             }
            if
($params{'con_'.$const.'_constraint'} =~
/foreign/) {
                $cmd .= "alter table
$params{'con_'.$const.'_table'} ";
                $cmd .= "enable primary key";
            }
            if
($params{'con_'.$const.'_has_index'} =~
/[Tt][Rr][Uu][Ee]/) {
                $cmd .= "";
            } else {
                $cmd .= "using index\n";
                $cmd .= "pctfree
$params{'con_'.$const.'_%f'}\n" if defined
$params{'con_'.$const.'_%f'};
                $cmd .= "initrans
$params{'con_'.$const.'_it'}\n" if defined
$params{'con_'.$const.'_it'};
                $cmd .= "maxtrans
$params{'con_'.$const.'_mt'}\n" if defined
$params{'con_'.$const.'_mt'};
```

```
            $cmd .= "tablespace
$params{'con_'.$const.'_ts'}\n" if ((defined
$params{'con_'.$const.'_ts'}) &&
($params{'skip_ts'} !~ /index/));
            $cmd .= "storage
$params{'con_'.$const.'_storage'}\n" if defined
$params{'con_'.$const.'_storage'};
            $cmd .= "nosort\n" if
($params{'con_'.$const.'_nosort'} =~ /true/);
            $cmd .= ";\n";
        }
       }
       &dump1("*sql");
       &dump0("*wait") if (($os cmp "nt") ==0);
       &time0("Done creating constraint
$const");
    }
# Alter DOP of indexes (NT support)
    if (($os cmp "nt") ==0)
    {
        &dump0("*wait");
        &time0("Alter DOP of indexes");
        $cmd  = "connect
$params{'user'}/$params{'passwd'};\n";

        foreach $index (@indexlist)
        {
            &advmulti();

            if (defined
$params{'ind_'.$index.'_pardeg_alter'})
            {
                $cmd .= "alter index $index
parallel";
                $cmd .= " (degree
$params{'ind_'.$index.'_pardeg_alter'}";
                $cmd .= " instances
$params{'ind_'.$index.'_parinst'});\n";
            }
        }
        &dump1("*sql");
        &dump0("*wait");
        &time0("Done altering DOP of indexes");
        &time0("Done index creation");
    }
} # end of ixcre


sub exp_ind_part_l
{
    @ind_partnames = ();
    $params{'ind_'.$index.'_#part'} =
        $params{$ob_stype.'_'.$params{'ind_'.$ind
ex.'_'.$ob_type}.'_#part'};

    if (!defined
$params{'ind_'.$index.'_partnames'})
    {
        # if no partnames are specified, use a
default part name
        for ($i = 1; $i <=
$params{'ind_'.$index.'_#part'}; $i++)
        {
            ($i ==
$params{'ind_'.$index.'_#part'}) ? ($addcomma =
"") :
                ($addcomma = ",");
            $nextfile = sprintf("%s%s%d%s",
$index,"_p", $i, $addcomma );
            $params{'ind_'.$index.'_partnames'}
=
$params{'ind_'.$index.'_partnames'} . $nextfile;
        }
    }

    @ind_partnames =
split(/,/,$params{'ind_'.$index.'_partnames'});
    if ($ind_partnames[0] =~ /\#/)
    {
        $filenm = shift(@ind_partnames);
        $savename = $filenm;
        $params{'ind_'.$index.'_partnames'} = "";
        # indtab defined in ixcre
        for ($i = 1; $i <= $numpart; $i++)
        {
            $filenm =~ s/\#/$i/g;
            ($i == $numpart) ? ($addcomma = "") :
($addcomma = ",");
            $params{'ind_'.$index.'_partnames'} =

        $params{'ind_'.$index.'_partnames'} .
$filenm . $addcomma;
            $filenm = $savename;
        }
        @ind_partnames =
split(/,/,$params{'ind_'.$index.'_partnames'});
        print "Expanded $savename
to:\n$params{'ind_'.$index.'_partnames'}\n\n" if
$verbose;
    } else {
        if (@ind_partnames != $numpart)
        {
            print "number of partitions $numpart
from the base table $indtab\ndoes not equal to
the number of partition names defined in
ind_".$index."_partnames.\n";
            exit(-1);
        }
    }

    &process_ind_part_ts;

    # complete the local partition index
statement
    if ($numpart >0)
    {
        $cmd .= "(\n";
        for ($i=0; $i < $numpart; $i++)
        {
            $cmd .= "partition ";
            $cmd .= $ind_partnames[$i] . "\n" if
@ind_partnames;

&process_part_params('ind','%f','pctfree',$index
,
                                    (@ind_partnames)
? $ind_partnames[$i] : "");

&process_part_params('ind','%u','pctused',$index
,
                                    (@ind_partnames)
? $ind_partnames[$i] : "");

&process_part_params('ind','it','initrans',$inde
x,
                                    (@ind_partnames)
? $ind_partnames[$i] : "");

&process_part_params('ind','mt','maxtrans',$inde
x,
                                    (@ind_partnames)
? $ind_partnames[$i] : "");
```

```
            if ((@ind_partnames) &&
                (defined
$params{'ind_'.$index.'_'.$ind_partnames[$i].'_t
s'}))
            {
                $cmd .= 'tablespace '.

$params{'ind_'.$index.'_'.$ind_partnames[$i].'_t
s'} . "\n";

        &process_part_storage('ind',$index,$ind_p
artnames[$i]);
            } elsif ((@ind_partnames) &&
                     defined
$params{'ind_'.$index.'_part_ts'})
            {
                $cmd .= 'tablespace ' .
$ind_part_ts[$i] . "\n";

        &process_part_storage('ind',$index,$ind_p
artnames[$i]);
            }
            if (@ind_partnames &&
                defined
$params{'ind_'.$index.'_'.$ind_partnames[$i].'_n
olg'})
            {
                $cmd .= "nologging\n"
                    if
($params{'ind_'.$index.'_'.$ind_partnames[$i].'_
nolg'}
                        =~ /[tT][rR][uU][eE]/);
            } elsif (defined
$params{'ind_'.$table.'_part_def_nolg'}) {
                $cmd .= "nologging\n"
                    if
($params{'ind_'.$index.'_part_def_nolg'} =~
                        /[tT][rR][uU][eE]/);
            }
            $cmd .= (($i+1) == $numpart) ? "\n" :
",\n";
        }
        $cmd .= ")\n";
    }
} # end of exp_ind_part_l


sub exp_ind_part_g
{
    @ind_partnames = ();
    @pcollist = split(/,/,
$params{'ind_'.$index.'_partcol'});
    $cmd .=
"$params{'ind_'.$index.'_partcol'})\n(\n";

    if (!defined
$params{'ind_'.$index.'_partnames'})
    {
        # if no partnames are specified, use a
default part name
        for ($i = 1; $i <=
$params{'ind_'.$index.'_#part'}; $i++)
        {
            ($i ==
$params{'ind_'.$index.'_#part'}) ? ($addcomma =
"") :
                ($addcomma = ",");
            $nextfile = sprintf("%s%s%d%s",
$index,"_p", $i, $addcomma );
            $params{'ind_'.$index.'_partnames'}
=
$params{'ind_'.$index.'_partnames'} . $nextfile;
        }
    }

    @ind_partnames =
split(/,/,$params{'ind_'.$index.'_partnames'});

    if ($ind_partnames[0] =~ /\#/)
    {
        $filenm = $ind_partnames[0];
        $savename = $filenm;
        $params{'ind_'.$index.'_partnames'} = "";
        # indtab defined in ixcre
        for ($i = 1; $i <= $numpart; $i++)
        {
            $filenm =~ s/\#/$i/g;
            ($i == $numpart) ? ($addcomma = "") :
($addcomma = ",");
            $params{'ind_'.$index.'_partnames'} =

        $params{'ind_'.$index.'_partnames'} .
$filenm . $addcomma;
            $filenm = $savename;
        }
        @ind_partnames =
split(/,/,$params{'ind_'.$index.'_partnames'});
        print "Expanded $savename
to:\n$params{'ind_'.$index.'_partnames'}\n\n" if
$verbose;
    } else {
        if (@ind_partnames != $numpart)
        {
            print "number of partitions $numpart
from the base table $indtab\ndoes not equal to
the number of partition names defined in
ind_$index_partnames.\n";
            exit(-1);
        }
    }

    # process boundaries and tablespaces

    &process_ind_part_brys;
    &process_ind_part_ts;

    # complete the local partition index
statement

    for ($i=0; $i < $numpart; $i++)
    {
        $cmd .= "partition " . $ind_partnames[$i]
. " values less than ";
         if ($i==$params{'ind_'.$table.'_#part'}-
1) {
           $cmd .= "(MAXVALUE)\n";
         }
         else {
           $cmd .= "(" . $ind_part_vals[$i] .
")\n";
         }

        &process_part_params('ind','%f','pctfree'
,$index,$ind_partnames[$i]);
        &process_part_params('ind','%u','pctused'
,$index,$ind_partnames[$i]);

&process_part_params('ind','it','initrans',$inde
x,$ind_partnames[$i]);

&process_part_params('ind','mt','maxtrans',$inde
x,$ind_partnames[$i]);
```

```
       if (defined
$params{'ind_'.$index.'_'.$ind_partnames[$i].'_t
s'})
       {
           $cmd .= 'tablespace '.

       $params{'ind_'.$index.'_'.$ind_partnames[
$i].'_ts'} . "\n";

&process_part_storage('ind',$index,$ind_partname
s[$i]);
       } elsif (defined
$params{'ind_'.$index.'_part_ts'})
       {
           $cmd .= 'tablespace ' .
$ind_part_ts[$i] . "\n";

&process_part_storage('ind',$index,$ind_partname
s[$i]);
       }
       if (defined
$params{'ind_'.$index.'_'.$ind_partnames[$i].'_n
olg'})
       {
           $cmd .= "nologging\n"
               if
($params{'ind_'.$index.'_'.$ind_partnames[$i].'_
nolg'}
                   =~ /[tT][rR][uU][eE]/);
       } elsif (defined
$params{'ind_'.$table.'_part_def_nolg'}) {
           $cmd .= "nologging\n"
               if
($params{'ind_'.$index.'_part_def_nolg'} =~
                   /[tT][rR][uU][eE]/);
       }
       $cmd .= (($i+1) == $numpart) ? "" :
",\n";
    }
    $cmd .= ")\n";

} # end of exp_ind_part_g


sub process_ind_part_ts
{
    # is ind_<name>_part_ts is in the form of
XXXX#, expand it
    # else treat it as a comma separated list of
ts names
    if (defined
$params{'ind_'.$index.'_part_ts'})
    {
        @ind_part_ts =
split(/,/,$params{'ind_'.$index.'_part_ts'});
        if ($ind_part_ts[0] =~ /\#/)
        {
            $filenm = shift(@ind_part_ts);
            $savename = $filenm;
            $params{'ind_'.$index.'_part_ts'} =
"";
            for ($i = 1; $i <= $numpart; $i++)
            {
                $filenm =~ s/\#/$i/g;
                ($i == $numpart) ? ($addcomma =
"") : ($addcomma = ",");

$params{'ind_'.$index.'_part_ts'} =

$params{'ind_'.$index.'_part_ts'} . $filenm .
$addcomma;
                $filenm = $savename;
            }
            @ind_part_ts =
split(/,/,$params{'ind_'.$index.'_part_ts'});
        } else {
            if (@ind_part_ts != $numpart)
            {
               $numpart = @ind_part_ts;
               if (($numpart % $numfil) == 0) {
                   $p = $numpart / $numfil;
                   $fil = "";
                   for ($i = 0; $i < $p; $i++)
                   {
                       if ($i == 0) {
                           $fil =
$params{'ind_'.$index.'_part_ts'};
                       } else {
                           $fil =
join(',',$fil,$params{'ind_'.$index.'_part_ts'})
;
                       }
                   }
                   @ind_part_ts = split(/,/,
$fil);
               } else {
                   print "Number of partitions
$numpart for table $index\n doesn't match
ind_$index_part_ts parameter.\n";
                   exit (-1);
               }
            }
        }
    }
} # end of process_ind_part_ts

sub process_ind_part_brys
{
    $cnt = 0;
    @ind_part_vals = ();

    foreach $col (@pcollist)
    {
        # add quotes for character strings and
dates

        if
(($params{$ob_stype.'_'.$indtab.'_'.$col.'_type'
} =~ /[cC][hH][aA][rR]/) ||

($params{$ob_stype.'_'.$indtab.'_'.$col.'_type'}
=~ /[dD][aA][tT][eE]/))
        {
            $addquote = "'";
        }

        @ind_part_col = ();

        # calculate the values for l_orderkey

        if ($col =~ /orderkey$/)
        {
           $high_l_orderkey =
$params{'scale_factor'} * 1500000 * 4 if ($col
=~ /^l_orderkey$/);
           $high_l_orderkey =
$params{'scale_factor'} * 1500000 * 4 if ($col
=~ /^o_orderkey$/);
           $pval = 1;
         if (defined
$params{$ob_stype.'_'.$table.'_#part'})
         {
```

```
            $numpart =
$params{$ob_stype.'_'.$table.'_#part'};
          }
          else
          {
            $numpart =
$params{'ind_'.$index.'_#part'};
          }
         $rest = $high_l_orderkey % $numpart;
          $incre = ($high_l_orderkey-$rest) /
$numpart;
          for ($i=0; $i < $numpart-1; $i++) {
             $pval = $pval + $incre;
             push(@ind_part_col, $pval);
          }
          push(@ind_part_col, 'MAXVALUE');
        }
        else {
          if ($col =~ /^l_partkey$/) {
            $high_l_partkey =
$params{'scale_factor'} * 200000;
            $pval = 1;
            if (defined
$params{$ob_stype.'_'.$table.'_#part'})
            {
              $numpart =
$params{$ob_stype.'_'.$table.'_#part'};
            }
            else
            {
              $numpart =
$params{'ind_'.$index.'_#part'};
            }
            $incre = $high_l_partkey / $numpart;
            for ($i=0; $i < $numpart-1; $i++) {
               $pval = $pval + $incre;
               push(@ind_part_col, $pval);
            }
           push(@ind_part_col, 'MAXVALUE');
          }
          else
          {
            @ind_part_col =
split(/,/,$params{'ind_'.$index.'_'.$col.'_partv
als'});
          }
        }

        if (@ind_part_col !=
$params{'ind_'.$index.'_#part'})
        {
            printf "Number of partition boundary
values %d for column $col in global index $index
doesn't match the number of partitions
($params{'ind_'.$index.'_#part'}) of the
index\n", ($#ind_part_col + 1);
            exit(-1);
        }
        for ($i=0; $i <
$params{'ind_'.$index.'_#part'}; $i++)
        {
            ($cnt == $#pcollist) ? ($addcomma =
"") : ($addcomma = ",");
           if ($ind_part_col[$i] =~
/[Mm][Aa][Xx][Vv][Aa][Ll][Uu][Ee]/) {
               $addquote = "";
            }
           if
($params{$ob_stype.'_'.$indtab.'_'.$col.'_type'}
=~ /[dD][aA][tT][eE]/)
            {
                if ($i ==
$params{'ind_'.$index.'_#part'} - 1)
                {
                    $ind_part_vals[$i] .=
"MAXVALUE";
                } else {
                    $nls_format = (defined
$params{$ob_stype.'_'.$indtab.'_'.$col.'_date_fo
rmat'}) ?
$params{$ob_stype.'_'.$indtab.'_'.$col.'_date_fo
rmat'} : "YYYY-MM-DD";
                    $ind_part_vals[$i] .=
"to_date('".$ind_part_col[$i]."','";
                    $ind_part_vals[$i] .=
$nls_format."')".$addcomma;
                }
            }
              elsif
(!(($params{$ob_stype.'_'.$indtab.'_'.$col.'_typ
e'} =~ /[iI][nN][tT][eE][gG][eE][rR]/) ||
($params{$ob_stype.'_'.$indtab.'_'.$col.'_type'}
=~ /[nN][uU][mM][bB][eE][rR]/)))
                {
                if ($index=~/l_ored/) {
                  print
"Before:$ind_part_vals[$i]";
                }
                $ind_part_vals[$i] =
$ind_part_vals[$i] . $addquote.
                    $ind_part_col[$i] .
$addquote . $addcomma ;
                if ($index=~/l_ored/) {
                  print
"After:$ind_part_vals[$i]\n";
                }

            } else {
              $ind_part_vals[$i] =
$ind_part_col[$i]. $addcomma ;
            }
        }
        $cnt++;
    }
} # end of process_ind_part_brys


sub anlyz
{

&dump0("#######################################
####################################");
    &dump0("# Analyze Phase");
    &time0("Begin anlyze");

    if ($params{'analyze_type'} =~ /via package
dbms.stats/)
    {
       &time0("Begin via package dbms.stats
analyzing");
        foreach $object (@anlyzlist)
        {
            $cmd = "connect
$params{'user'}/$params{'passwd'};\n";
            $phead="anl_".$object;
            $type =
$params{$phead.'_object_type'};
            $type = "table" if
($params{$phead.'_object_type'} =~ /view/);
            $type = "table" if
($params{$phead.'_object_type'} =~ /viewlog/);
            $objectname = $object;
```

```
            $objectname = "MLOG\\\$_".$object if
($params{$phead.'_object_type'} =~ /viewlog/);
            $cmd .= "execute
dbms_stats.$params{$phead.'_anl_type'}_".$type."
_stats('$params{'user'}'";
            $cmd .= " , estimate_percent =>
$params{$phead.'_percent'}" if (defined
$params{$phead.'_percent'});
            $cmd .= " , degree =>
$params{$phead.'_degree'}" if (defined
$params{$phead.'_degree'});
            $cmd .= " , granularity =>
'$params{$phead.'_granularity'}'" if (defined
$params{$phead.'_granularity'});
            $cmd .= " , block_sample =>
$params{$phead.'_block_sample'}" if (defined
$params{$phead.'_block_sample'});
            $cmd .= " , '".$objectname."'" if
!($object =~ /schema/);
            $cmd .= ");\n";
            &dump1("*sql");
             &dump0("*wait") if ($object =~
/schema/);
        }
        &time0("End per schema analyzing");
    }
    else
    {
        foreach $object (@anlyzlist)
        {
             &time0("Begin analyzing $object");
            &advmulti();
            if (($params{'anl_'.$object.'_type'}
=~ /table/) &&
                ($params{'tab_'.$object.'_#part'}
> 1)) {
                $tab_entry = 'tab_'.$object;
                if (!defined
$params{$tab_entry.'_partnames'})
                {
                    # if no partnames are
specified, use a default part name
                    for ($i = 1; $i <=
$params{$tab_entry.'_#part'}; $i++)
                    {
                       ($i ==
$params{$tab_entry.'_#part'}) ? ($addcomma = "")
:
                            ($addcomma = ",");
                      $nextfile =
sprintf("%s%s%d%s", $table,"_p",
                                           $i,
$addcomma );

        $params{$tab_entry.'_partnames'} =

$params{$tab_entry.'_partnames'} . $nextfile;
                    }
                }
                @tab_partnames =
split(/,/,$params{$tab_entry.'_partnames'});
                # if partnames is specified as
XXXX#, then expand
                if ($tab_partnames[0] =~ /\#/)
                {
                    $filenm =
shift(@tab_partnames);
                    $savename = $filenm;

$params{$tab_entry.'_partnames'} = "";
                    for ($i = 1; $i <=
$params{$tab_entry.'_#part'}; $i++)
                    {
                        $filenm =~ s/\#/$i/g;
                        ($i ==
$params{$tab_entry.'_#part'}) ? ($addcomma = "")
:
                            ($addcomma = ",");

        $params{$tab_entry.'_partnames'} =

$params{$tab_entry.'_partnames'} . $filenm .
$addcomma;
                        $filenm = $savename;
                    }
                    @tab_partnames =
split(/,/,$params{$tab_entry.'_partnames'});
                    printf("Expanded $savename
to:\n$params{$tab_entry.'_partnames'}\n\n") if
$verbose;
                } else {
                    if (@tab_partnames !=
$params{$tab_entry.'_#part'})
                    {
                        print "Number of
partitions $params{$tab_entry.'_#part'} for
$table doesn't match\n _partnames parameter for
$params{$tab_entry.'_partnames'}\n";
                        exit(-1);
                    }
                }
                @tab_partnames =
split(/,/,$params{'tab_'.$object.'_partnames'});
                foreach $partname
(@tab_partnames)
                {
                    $cmd = "connect
$params{'user'}/$params{'passwd'};\n";
                    $cmd .= "analyze
$params{'anl_'.$object.'_type'} $object ";
                    $cmd .= "partition
($partname) ";
                    if (defined
$params{'anl_'.$object.'_estimate'})
                    {
                        $cmd .= "estimate
statistics
$params{'anl_'.$object.'_estimate'};\n";
                    }
                    else
                    {
                        $cmd .= "compute
statistics;\n";
                    }
                    &dump1("*sql");
                }
            } else {
                if
(($params{'anl_'.$object.'_type'} =~ /index/) &&
                    (defined
$params{'ind_'.$object.'_partition'})) {
                    $ind_entry = 'ind_'.$object;
                    if
($params{'ind_'.$object.'_partition'} =~
/[lL][oO][cC][aA][lL]/) {

        $params{$ind_entry.'_#part'} =

$params{'tab_'.$params{$ind_entry.'_table'}.'_#p
art'};
                    }
                    if (!defined
$params{$ind_entry.'_partnames'})
                    {
```

```
                    # if no partnames are
specified, use a default part name
                    for ($i = 1; $i <=
$params{$ind_entry.'_#part'}; $i++)
                    {
                        ($i ==
$params{$ind_entry.'_#part'}) ?
                            ($addcomma = "") :
($addcomma = ",");
                        $nextfile =
sprintf("%s%s%d%s",$object,"_p",
                                                $i,
$addcomma );

$params{$ind_entry.'_partnames'} =

       $params{$ind_entry.'_partnames'} .
$nextfile;
                    }
                }

                @ind_partnames =
split(/,/,$params{$ind_entry.'_partnames'});

                # if partnames is specified
as XXXX#, then expand
                if ($ind_partnames[0] =~
/\#/)
                {
                    $filenm =
shift(@ind_partnames);
                    $savename = $filenm;

       $params{$ind_entry.'_partnames'} = "";

                    for ($i = 1; $i <=
$params{$ind_entry.'_#part'}; $i++)
                    {
                        $filenm =~ s/\#/$i/g;

($i==$params{$ind_entry.'_#part'}) ? ($addcomma
= "") :
                            ($addcomma = ",");

$params{$ind_entry.'_partnames'} =

       $params{$ind_entry.'_partnames'} .
$filenm .
                                $addcomma;
                        $filenm = $savename;
                    }

       @tab_partnames=split(/,/,$params{$ind_ent
ry.'_partnames'});
                    printf("Expanded $savename
to:\n$params{$ind_entry.'_partnames'}\n\n") if
$verbose;
                } else {
                    if (@ind_partnames !=
$params{$ind_entry.'_#part'})
                    {
                        print "Number of
partitions $params{$ind_entry.'_#part'} for
$table doesn't match\n _partnames parameter for
$params{$ind_entry.'_partnames'}\n";
                        exit(-1);
                    }
                }
                @ind_partnames =
split(/,/,$params{'ind_'.$object.'_partnames'});
                foreach $partname
(@ind_partnames)
                {
                    $cmd = "connect
$params{'user'}/$params{'passwd'};\n";
                    $cmd .= "analyze
$params{'anl_'.$object.'_type'} $object ";
                    $cmd .= "partition
($partname) ";
                    if (defined
$params{'anl_'.$object.'_estimate'})
                    {
                        $cmd .= "estimate
statistics
$params{'anl_'.$object.'_estimate'};\n";
                    }
                    else
                    {
                        $cmd .= "compute
statistics;\n";
                    }
                    &dump1("*sql");
                }
            } else {
                $cmd = "connect
$params{'user'}/$params{'passwd'};\n";
#               $cmd .= "analyze
$params{'anl_'.$object.'_type'} $object ";
#               $cmd .= "delete statistics;\n";
                $cmd .= "analyze
$params{'anl_'.$object.'_type'} $object ";
                if (defined
$params{'anl_'.$object.'_estimate'})
                {
                    $cmd .= "estimate
statistics
$params{'anl_'.$object.'_estimate'};\n";
                }
                else
                {
                    $cmd .= "compute
statistics;\n";
                }
                &dump1("*sql");
            }
        }
        &time0("Done analyzing $object");
    }
    &time0("End anlyze");
}

sub chdop_old
{

&dump0("#######################################
####################################");
    &dump0("# Make DOP Phase");

    if (@chdoplist) {
        $cmd = "connect
$params{'user'}/$params{'passwd'};\n";
    }
    foreach $object (@chdoplist)
    {
       if ($params{'chdop_'.$object.'_type'} =~
/table/) {
         $cmd .= "alter table $object parallel
(degree $chdoplist{$object} instances 1);\n";
       }
       else {
         $cmd .= "alter index $object parallel
(degree $chdoplist{$object} instances 1);\n";
       }
```

```
    }
    foreach $object (@chstorlist)
    {
       if ($params{'chstor_'.$object.'_type'} =~
/table/) {
         $cmd .= "alter table $object storage
(next $chstorlist{$object} pctincrease 0);\n";
       }
       else {
         $cmd .= "alter index $object storage
(next $chstorlist{$object} pctincrease 0);\n";
       }
    }
    &dump1("*sql");
}

sub chob
{

&dump0("#################################b######
#####################################");
    &dump0("# Make CHOB Phase");

    if (@choblist) {
        $cmd = "connect
$params{'user'}/$params{'passwd'};\n";
    }
  FLOOP:
    while (@choblist)
    {
        $object = shift(@choblist);
        if ((defined
$params{'chob_tab_'.$object.'_degpar'}) ||
(defined
$params{'chob_tab'.$object.'_storage'}))
        { $objecttype = 'table';  $objecttypes =
'tab';$objectn=$object;}
        elsif ((defined
$params{'chob_ind_'.$object.'_degpar'}) ||
(defined
$params{'chob_ind'.$object.'_storage'}))
        {$objecttype = 'index'; $objecttypes =
'ind';$objectn=$object;}
        elsif ((defined
$params{'chob_viewlog_'.$object.'_degpar'}) ||
(defined
$params{'chob_viewlog'.$object.'_storage'}))
        {$objecttype = 'table'; $objecttypes =
'viewlog';$objectn="MLOG\\\$_".$object}
        else
        {next FLOOP;}
        if (defined
$params{'chob_'.$objecttypes.'_'.$object.'_degpa
r'})
        {
            $cmd .= "alter $objecttype $objectn";
            if
($params{'chob_'.$objecttypes.'_'.$object.'_degp
ar'} =~ /[dD][eE][fF][aA][uU][lL][tT]/)
            {
               $cmd .= " parallel";
            }
            elsif
($params{'chob_'.$objecttypes.'_'.$object.'_degp
ar'} =~
/[nN][oO][pP][aA][rR][aA][lL][lL][eE][lL]/)
            {
               $cmd .= " noparallel";
            }
            else
            {
               $cmd .= " parallel
$params{'chob_'.$objecttypes.'_'.$object.'_degpa
r'}"
                  if (defined
$params{'chob_'.$objecttypes.'_'.$object.'_degpa
r'});
            }
            $cmd .= ";\n";
        }
        if (defined
$params{'chob_'.$objecttypes.'_'.$object.'_stora
ge'})
        {
            $cmd .= "alter $objecttype $objectn";
            if (defined
$params{'chob_'.$objecttypes.'_'.$object.'_stora
ge'})
            {
               $cmd .= " storage
($params{'chob_'.$objecttypes.'_'.$object.'_stor
age'});\n";
            }
        }
        if ($objecttypes eq 'tab')
        {
            delete
$params{'chob_tab_'.$object.'_degpar'};
            delete
$params{'chob_tab'.$object.'_storage'};
            push (@choblist,$object);
        }
    }
    &dump1("*sql");
}




sub expln
{

&dump0("#########################################
#####################################");
    &dump0("# Explain Plan Phase");

    $params{'qry_explain_queries'} =
$params{'qry_all_queries'} if !defined
$params{'qry_explain_queries'};
    if ($params{'qry_explain_queries'})
    {
        $astr = "2_1";
        $astr = "2_".$params{'db_maxinstances'}
if defined $params{'db_maxinstances'};
        &alttabs($astr);
        @qrynames = split(/,/,
$params{'qry_explain_queries'});
        foreach $qname (@qrynames)
        {
            $cmd = "";
            $cmd .= "$params{'qry_area'}expl." .
$qname . "\n";
            $cmd2 = "";
            $cmd2 = $cmd2 . "connect
$params{'user'}/$params{'passwd'};\n";
            $cmd2 = $cmd2 . "set longwidth
360\n";
            if (defined
$params{'qry_'.$qname.'_text'})
            {
               $cmd2 = $cmd2 . "drop table
plan_table;\n";
```

```
                $cmd2 = $cmd2 . '@' .
"$params{'dd_sql_area'}" . "utlxplan.sql\n";
                $cmd2 = $cmd2 . "explain plan
for\n";
                $cmd2 = $cmd2 .
"$params{'qry_'.$qname.'_text'}\n";
                $cmd2 = $cmd2 . "select level,
rpad(lpad(' ',2*(level-1)) || operation || ' '
|| object_name || ' ' || options || ' (' ||
object_node || ',' || cost || ',' || cardinality
|| ')',68)\n";
                $cmd2 = $cmd2 . "from
plan_table\n";
                $cmd2 = $cmd2 . "connect by
parent_id=prior id start with id=0;\n";
                $cmd2 = $cmd2 . "select level,
lpad(' ',2*(level-1)) || operation || ' ' ||
object_name || ' ' || options || ' (' ||
object_node || ')',\n";
                $cmd2 = $cmd2 . "other_tag,
other\n";
                $cmd2 = $cmd2 . "from
plan_table\n";
                $cmd2 = $cmd2 . "connect by
parent_id=prior id start with id=0;\n";
            }
            else
            {
                if (defined
$params{'qry_'.$qname.'_texte'})
                {
                    @expln_texts = split(/,/,
$params{'qry_'.$qname.'_texte'});
                    $text_index = 1;
                    while ($expln_next = shift
(@expln_texts))
                    {
                      EXPLNLOOP:
                        while ($expln_next ne
('text'.$text_index))
                        {
                            if (!defined
$params{'qry_'.$qname.'_text'.$text_index})
                            {
                                print "\n**
Missing text for $qname: text$text_index\n";
                                last EXPLNLOOP;
                            }
                            $cmd2 = $cmd2 .
"$params{'qry_'.$qname.'_text'.$text_index}\n";
                            $text_index++;
                        }
                        $cmd2 = $cmd2 . "drop
table plan_table;\n";
                        $cmd2 = $cmd2 . '@' .
"$params{'dd_sql_area'}" . "utlxplan.sql\n";
                        $cmd2 = $cmd2 . "explain
plan for\n";
                        $sqltext =
'qry_'.$qname.'_'.$expln_next;
                        $cmd2 = $cmd2 .
"$params{$sqltext}\n";
                        $cmd2 = $cmd2 . "select
level, rpad(lpad(' ',2*(level-1)) || operation
|| ' ' || object_name || ' ' || options || ' ('
|| object_node || ',' || cost || ',' ||
cardinality || ')',68)\n";
                        $cmd2 = $cmd2 . "from
plan_table\n";
                        $cmd2 = $cmd2 . "connect
by parent_id=prior id start with id=0;\n";
                        $cmd2 = $cmd2 . "select
level, lpad(' ',2*(level-1)) || operation || ' '
|| object_name || ' ' || options || ' (' ||
object_node || ')',\n";
                        $cmd2 = $cmd2 .
"other_tag, other\n";
                        $cmd2 = $cmd2 . "from
plan_table\n";
                        $cmd2 = $cmd2 . "connect
by parent_id=prior id start with id=0;\n";
                        $text_index++;
                    }
                    while (defined
$params{'qry_'.$qname.'_text'.$text_index})
                    {
                        $cmd2 = $cmd2 .
"$params{'qry_'.$qname.'_text'.$text_index}\n";
                        $text_index++;
                    }
                 }
                else
                {
                    print "\n** No sql text or
texte corresponding to $qname\n";
                }
            }
            &dump2("*sql2");
        }
    }
}

sub query
{

&dump0("########################################
####################################");
    &dump0("# Query Phase");

    $params{'qry_run_queries'} =
$params{'qry_all_queries'} if !defined
$params{'qry_run_queries'};
    if ($params{'qry_run_queries'})
    {
        $params{'qry_config'} =  "1_1" if
!defined $params{'qry_config'};
        @qconfs = split(/,/,
$params{'qry_config'});
        foreach $qconf (@qconfs)
        {
            &alttabs($qconf);
            if ($params{'qry_separate_queries'}
!~ /true/)
            {
                $cmd = "";
                $cmd .=
"$params{'qry_area'}qry.all." . $qconf . "\n";
                $cmd2 = "";
                $cmd2 = $cmd2 . "set timing
on\n";
                $cmd2 = $cmd2 . "connect
$params{'user'}/$params{'passwd'};\n";
            }
            @qrynames = split(/,/,
$params{'qry_run_queries'});
            foreach $qname (@qrynames)
            {
                if
($params{'qry_separate_queries'} =~ /true/)
                {
                    $cmd = "";
```

```
                    $cmd .=
"$params{'qry_area'}qry." . $qname . "." .
$qconf . "\n";
                    $cmd2 = "";
                    $cmd2 = $cmd2 . "set timing
on\n";
                    $cmd2 = $cmd2 . "connect
$params{'user'}/$params{'passwd'};\n";
                }
                for ($i = 0; $i <
$params{'qry_number_trials'}; $i++)
                {
                    if (defined
$params{'qry_'.$qname.'_text'})
                    {
                        $cmd2 = $cmd2 .
"$params{'qry_'.$qname.'_text'}\n";
                    }
                    else
                    {
                        $text_index = 1;
                      WTLOOP:
                        while(1)
                        {
                            if (defined
$params{'qry_'.$qname.'_text'.$text_index})
                            {
                                $cmd2 = $cmd2 .
"$params{'qry_'.$qname.'_text'.$text_index}\n";
                                $text_index++;
                            }
                            else
                            {
                                print "No sql
text corresponding to $qname\n" if ($text_index
== 1);
                                last WTLOOP;
                            }
                        }
                    }
                }
                if
($params{'qry_separate_queries'} =~ /true/)
                {
                    &dump2("*sql2");
                }
            }
            if ($params{'qry_separate_queries'}
!~ /true/)
            {
                &dump2("*sql2");
            }
        }
    }
}

#*********************************************
*****************************
#*********************************************
*****************************
# Utility Functions
#*********************************************
*****************************
#*********************************************
*****************************

sub prepmulti
{
# Some multi-instance stuff
    $num_inst = $params{$phase.'_num_inst'};
    if ($num_inst > 1)
    {
        $multi = 1;
        $cur_inst = 1;
        &advmulti(); # Make sure it's set to
some appropriate value
    }
    else
    {
        $multi = 0;
    }
}

sub advmulti
{
    if ($multi)
    {
        &setinst($cur_inst);
        $cur_inst = ($cur_inst % $num_inst) + 1;
    }
}

sub startdb #does not work for OPS systems!!
            #if you have an OPS system use
startdb_old
{
    $initora=$params{'startupfile_'.$_[0]};
    $mounttype=$_[1];
    $cmd = "startup pfile=$initora
$mounttype\n";
    &dump1("*sql");
}
sub startdb_old
{
# Get init.oras for startup/shutdown
    $checkios = 0;
    $ifile = sprintf ("%s%s",
$params{'dbs_area'},
"init_$params{'oracle_sid'}.ora");
    $checkios = 1 if (!-e $ifile);
    for ($j = 1; $j <= $num_inst; $j++)
    {
        $iofname[$j] = sprintf ("%s%s",
$params{'dbs_area'},
"init".$j."_$params{'oracle_sid'}.ora");
        $iofname[$j] = sprintf ("/host%s",
$iofname[$j])
            if ($params{'special_machine'} eq
'ncube');
        $iofname[$j] =~
s/\?/$ENV{'ORACLE_HOME'}/g;
        $checkios = 1 if (!-e $iofname[$j]);
    }
    &createios() if $checkios;

# startup exclusive/shared single-
instance/multi-instance
    if ($multi)
    {
        for ($loop = 1; $loop <= $num_inst;
$loop++)
        {
            &setinst($loop);
            $cmd = "startup shared
pfile=$iofname[$loop];\n";
            &dump1("*sql");
            &dump0("*wait") if
($params{'sync_startup'} =~ /true/);
        }
    }
    else
    {
        $cmd = "startup pfile=$iofname[1];\n";
        &dump1("*sql");
```

```
    }
    &dump0("*wait");
}

sub shutdb
{
# shutdown whatever might be running
    if ($multi)
    {
        for ($loop = $num_inst; $loop >= 1;
$loop--)
        {
            &setinst($loop);
            $cmd = "shutdown
$params{'shutdown_level'};\n";
            &dump1("*sql");
        }
    }
    else
    {
        $cmd = "shutdown
$params{'shutdown_level'};\n";
        &dump1("*sql");
    }
    &dump0("*wait");
}

sub create_ts
{
    $tsname = shift(@_);      # 1. parameter is
the list of tablespaces

    &dump0("# creating $params{'user'}'s $tsname
tablespace");
    &advmulti();
    @ts_datafiles = split(/,/,
$params{$tsname.'_datafiles'});
    $ts_datafile = shift (@ts_datafiles);

    if ($params{$tsname.'_managed'} =~
/[uU][sS][eE][rR]/)
    {                         # user managed
temporary tablespace
       if ($params{$tsname.'_temporary'} =~
/[tT][rR][uU][eE]/)
           {                  # temporary
tablespace
               $cmd .= "drop tablespace $tsname
including contents;\n";
               $cmd .= "create tablespace
$tsname temporary\n";
           }
           else               # user managed
permanent tablespace
           {
               $cmd .= "drop tablespace $tsname
including contents;\n";
               $cmd .= "create tablespace
$tsname\n";
           }
       $cmd .= "datafile
'$params{$tsname.'_area'}$ts_datafile' size
$params{$tsname.'_first_size'}
$params{$tsname.'_options'}\n";
       $cmd .= "default storage
$params{$tsname.'_storage'}\n" if defined
$params{$tsname.'_storage'};
    }
    else                        # system managed
temporary tablespace
    {
       if ($params{$tsname.'_temporary'} =~
/[tT][rR][uU][eE]/)
           {                  # temporary
tablespace
               $cmd .= "drop tablespace $tsname
including contents;\n";
               $cmd .= "create temporary
tablespace $tsname\n";
               $cmd .= "tempfile
'$params{$tsname.'_area'}$ts_datafile' size
$params{$tsname.'_first_size'}
$params{$tsname.'_options'}\n";
               $cmd .= "extent management
local\n";
               if (defined
($params{$tsname.'_extent_size'}) &&
($params{$tsname.'_extent_size'} =~
/[aA][uU][Tt][oO][Aa][lL][lL][oO][cC][aA][tT][eE
]/))
               {
                   $cmd .= "autoallocate\n";
               }
               elsif (defined
($params{$tsname.'_extent_size'}) &&
!($params{$tsname.'_extent_size'} =~
/[aA][uU][Tt][oO][Aa][lL][lL][oO][cC][aA][tT][eE
]/))
               {
                   $cmd .= "uniform size
".$params{$tsname.'_extent_size'}."\n";
               }
               else
               {
                   $cmd .= "uniform";
               }
           }
       else           # system managed permanent
tablespace
       {
           $cmd .= "drop tablespace $tsname
including contents;\n";
           $cmd .= "create tablespace
$tsname\n";
           $cmd .= "datafile
'$params{$tsname.'_area'}$ts_datafile' size
$params{$tsname.'_first_size'}
$params{$tsname.'_options'}\n";
           $cmd .= "extent management local\n";
           if (defined
($params{$tsname.'_extent_size'}) &&
($params{$tsname.'_extent_size'} =~
/[aA][uU][Tt][oO][Aa][lL][lL][oO][cC][aA][tT][eE
]/))
           {
               $cmd .= "autoallocate\n";
           }
           elsif (defined
($params{$tsname.'_extent_size'}) &&
!($params{$tsname.'_extent_size'} =~
/[aA][uU][Tt][oO][Aa][lL][lL][oO][cC][aA][tT][eE
]/))
           {
               $cmd .= "uniform size
".$params{$tsname.'_extent_size'}."\n";
           }
           else
           {
               $cmd .= "uniform";
           }
       }
   }
```

```
    $cmd .= " nologging\n" if
($params{$tsname.'_nolg'} =~
/[tT][rR][uU][eE]/);
    $cmd .= ";\n";
    &dump1("*sql");
}

sub add_ts_rollb
{
    $ts_undo_l = $_[0];
    @ts_datafiles = split(/,/,
$params{'ts_'.$ts_undo_l.'_datafiles'});
    $ts_datafile = shift (@ts_datafiles);
    $cmd .= "create tablespace
ts_".$ts_undo_l."\n";
    $cmd .= "  datafile
'$params{'ts_undo_area'}$ts_datafile' size
$params{'ts_'.$ts_undo_l.'_first_size'}
$params{'ts_'.$ts_undo_l.'_options'};\n";
    &dump1("*sql");
    &dump0("*wait");
    &add_dfs("ts_$ts_undo_l");

# currently set up for the foreground - maybe
should be changed
    &dump0("# creating extra rollback
segments");
    for ($i = 1; $i <=
$params{'ts_'.$ts_undo_l.'_#rs'}; $i++)
    {
        $cmd .= "create public rollback segment
$params{'ts_'.$ts_undo_l.'_rs_prefix'}$i";
        $cmd .= " storage
$params{'ts_undo_rs_storage'}" if defined
$params{'ts_'.$ts_undo_l.'_rs_storage'};
        $cmd .= " tablespace ts_$ts_undo_l" if
($params{'skip_ts'} !~ /$ts_undo_l/);
        $cmd .= ";\n";
    }
    &dump1("*sql");
    &dump0("*wait");

}

sub add_dfs
{
    $tsname = shift(@_);
    &dump0("# adding $params{'user'}'s $tsname
datafiles");
    @ts_datafiles = split(/,/,
$params{$tsname.'_datafiles'});
    $ts_datafile = shift (@ts_datafiles); # drop
first one
    $cur_inst = 2 if $multi;  # for possible
locality on SP2
    for ($i = 1; $i <
$params{$tsname.'_#files'}; $i++)
    {
        $ts_datafile = shift (@ts_datafiles);
        &advmulti();
       $cmd .= "alter tablespace $tsname\n";
       if ($params{$tsname.'_managed'} =~
/[uU][sS][eE][rR]/)
       {  # user managed temporary tablespace
           $cmd .= "  add datafile ";
       }
       else
       {  # system managed temporary tablespace
           if ($params{$tsname.'_temporary'} =~
/true/)
           {
               $cmd .= "  add tempfile ";
           }
           else
           {
               $cmd .= "  add datafile ";
           }
       }
       $cmd .=
"'$params{$tsname.'_area'}$ts_datafile' size
$params{$tsname.'_size'}
$params{$tsname.'_options'};\n";
        &dump1("*sql");
    }
}

sub createios
{
    push (@iokv,
"db_name=$params{'oracle_sid'}");
    push (@iokv,
"control_files=$params{'io_control_files'}");
    push (@iokv, "db_block_buffers=1000");
    push (@iokv, "shared_pool_size=35000000");
    push (@iokv, "parallel_max_servers=144");
    push (@iokv, "parallel_min_servers=0");
    push (@iokv, "max_dump_file_size=5000");
    push (@iokv, "audit_trail=FALSE");
    push (@iokv, "global_names=FALSE");
#    push (@iokv, "commit_point_strength=1");
#    push (@iokv, "dblink_encrypt_login=true");
#    push (@iokv, "db_block_size=16384");
    push (@iokv, "db_block_size=8192");
    push (@iokv,
"db_file_multiblock_read_count=32");
    push (@iokv, "processes=256");
    push (@iokv, "sessions=256");
    push (@iokv, "transactions=512");
    push (@iokv,
"transactions_per_rollback_segment=20");
    push (@iokv, "max_rollback_segments=256");
    push (@iokv, "distributed_transactions=0");
    push (@iokv, "nls_date_format=YYYY-MM-DD");
    push (@iokv, "db_files=1023");
    push (@iokv, "open_cursors=1024");
    push (@iokv, "optimizer_mode=CHOOSE");
    push (@iokv,
"optimizer_percent_parallel=100");
    push (@iokv, "sort_area_size=3000000");
    push(@iokv, "always_semi_join=HASH");
    push(@iokv,
"parallel_broadcast_enabled=TRUE");
    push(@iokv,
"optimizer_features_enable=8.0.4");
    push(@iokv, "compatible=8.0.4");
    push(@iokv, "hash_multiblock_io_count=64");
    push(@iokv, "always_anti_join=HASH");


    $ifile = sprintf ("%s%s",
$params{'dbs_area'},
"init_$params{'oracle_sid'}.ora");
    $ifile =~ s/\?/$ENV{'ORACLE_HOME'}/g;
    if ($params{'skip_mk_initoras'} !~
/include/)
    {
        open (IOFILE, ">$ifile");
        print IOFILE "# Init.ora include file
created by bumpx.pl\n\n";
        while ($kv = shift(@iokv))
        {
            $kv =~ /(.*)=(.*)/;
            $entry = sprintf ("%-25s = %s\n",
$1, $2);
```

```
            print IOFILE $entry;
        }
        if (defined $params{'io_ifile'})
        {
            $entry = sprintf ("%-25s = %s\n",
"ifile", $params{'io_ifile'});
            print IOFILE $entry;
        }
        close (IOFILE);
    }

    for ($j = 1; $j <= $num_inst; $j++)
    {
        $iofname[$j] = sprintf ("%s%s",
$params{'dbs_area'},
"init".$j."_$params{'oracle_sid'}.ora");
        $iofname_opn = $iofname[$j];
        $iofname_opn =~
s/\?/$ENV{'ORACLE_HOME'}/g;
        if ($params{'skip_mk_initoras'} !~
/instance/)
        {
            open (IOFILE, ">$iofname_opn");
            print IOFILE "# Init.ora for
instance $j created by bumpx.pl\n\n";
            print IOFILE "thread = $j\n"
                if ($params{'special_machine'}
ne 'ncube');
            print IOFILE "ifile = $ifile\n";
            close (IOFILE);
        }
        $iofname[$j] = sprintf ("/host%s",
$iofname[$j])
            if ($params{'special_machine'} eq
'ncube');
    }
}


sub create_objects

{
    $ob_type = shift(@_);
    @ob_objectlist = @_;
    $cmd .= "connect
$params{'user'}/$params{'passwd'};\n";
    foreach $object (@ob_objectlist)
    {
       &create_object;
    }
    &dump1("*sql");
}


sub create_object

# this routine only creates one object
# need it because of the creation of
materialized views

{

    $ob_entry = $ob_type.'_'.$object;
    if (($ob_type cmp 'tab') == 0 )  # we are
creating a table
    {
        $cmd .= "drop table $object;\n";
        $cmd .= "create table $object";
    }
    elsif (($ob_type cmp 'view') == 0) # we are
creating a view
    {
        if ($params{$ob_entry.'_creation_type'}
=~ /[dD][iI][rR][eE][cC][tT]/) # direct
        {
            $cmd .= "drop materialized view
$object;\n";
            $cmd .= "create materialized view
$object";
        }
        else           # with table
        {
            $cmd .= "drop table $object;\n";
            $cmd .= "create table $object";
        }
    }
    if (($ob_type cmp 'viewlog') == 0)       #
we have a viewlog
    {
        $cmd .= "drop materialized view log on
$object;\n";
        $cmd .= "create materialized view log ";
        $cmd .= "on $object\n";
    }
    elsif (!($ob_type eq 'view') ||
           (($ob_type eq 'view') &&
($params{$ob_entry.'_creation_type'} =~
/[wW][iI][tT][hH]\s[tT][aA][bB][lL][eE]/)))
    {
        $cmd .= "(\n";
        $cmd .= "$params{$ob_entry.'_cons'}\n" if
defined $params{$ob_entry.'_cons'};
        @ob_collist = split(/,/,
$params{$ob_entry.'_columns'});
        &add_columns;  # the list of columns are
in @ob_collist
        $cmd .= ")\n";
    }
    else
    {
        $cmd .= "\n";
    }
    if (defined $params{$ob_entry.'_cluster'})
    {
        $cmd .= "cluster
$params{$ob_entry.'_cluster'}
($params{$ob_entry.'_clucols'})\n";
    }
    else
    {
        # add organization and pctthreshold for
index only tables
        $cmd .= "organization
$params{$ob_entry.'_org'}\n" if defined
$params{$ob_entry.'_org'};
        $cmd .= "pctthreshold
$params{$ob_entry.'_%t'}\n" if defined
$params{$ob_entry.'_%t'};
        $cmd .= "pctfree
$params{$ob_entry.'_%f'}\n" if defined
$params{$ob_entry.'_%f'};
        $cmd .= "pctused
$params{$ob_entry.'_%u'}\n" if defined
$params{$ob_entry.'_%u'};
        $cmd .= "initrans
$params{$ob_entry.'_it'}\n" if defined
$params{$ob_entry.'_it'};
        $cmd .= "maxtrans
$params{$ob_entry.'_mt'}\n" if defined
$params{$ob_entry.'_mt'};
        $cmd .= "tablespace
$params{$ob_entry.'_ts'}\n" if defined
$params{$ob_entry.'_ts'};
```

```
        $cmd .= "storage
$params{$ob_entry.'_storage'}\n" if defined
$params{$ob_entry.'_storage'};
        if (defined $params{$ob_entry.'_of_ts'})
        {
            # overflow tablespace specs
            $cmd .= "overflow ";
            $cmd .= "pctfree
$params{$ob_entry.'_of_%f'} " if defined
$params{$ob_entry.'_of_%f'};
            $cmd .= "pctused
$params{$ob_entry.'_of_%u'} " if defined
$params{$ob_entry.'_of_%u'};
            $cmd .= "initrans
$params{$ob_entry.'_of_it'} " if defined
$params{$ob_entry.'_of_it'};
            $cmd .= "maxtrans
$params{$ob_entry.'_of_mt'} " if defined
$params{$ob_entry.'_of_mt'};
            $cmd .= "tablespace
$params{$ob_entry.'_of_ts'} " if defined
$params{$ob_entry.'_of_ts'};
            $cmd .= "storage
$params{$ob_entry.'_of_storage'}\n" if defined
$params{$ob_entry.'_of_storage'};
        }
    }
    if ((defined $params{$ob_entry.'_pardeg'})
|| (defined $params{$ob_entry.'_parinst'}))
    {
        $cmd .= "parallel";
        if ((defined
$params{$ob_entry.'_pardeg'}) &&
            ($params{$ob_entry.'_pardeg'} =~
/[dD][eE][fF][aA][uU][lL][tT]/))
        {
            $cmd .= "\n";
        }
        else
        {
            $cmd .= "(degree
$params{$ob_entry.'_pardeg'} " if defined
$params{$ob_entry.'_pardeg'};
            $cmd .= "instances
$params{$ob_entry.'_parinst'}" if defined
$params{$ob_entry.'_parinst'};
            $cmd .= ")\n";
        }
    }
    $cmd .= "nologging\n" if
($params{$ob_entry.'_nolg'} =~
/[tT][rR][uU][eE]/);
    $cmd .= "cache\n" if
($params{$ob_entry.'_cache'} =~
/[tT][rR][uU][eE]/);

    # add partition support
    if (defined $params{$ob_entry.'_#part'} &&
($params{$ob_entry.'_#part'} > 1))
    {
        &expand_partitions
($ob_entry,$object,$ob_type);
    }                 # no partitioning


    if (($ob_type cmp 'view') == 0)
    {
        $cmd .= "enable row movement\n" if
defined($params{$ob_entry.'_parttype'});
        if ($params{$ob_entry.'_creation_type'}
=~ /[dD][iI][rR][eE][cC][tT]/)
        {
            $cmd .= "BUILD
".$params{$ob_entry.'_build_when'}."\n" if
(defined $params{$ob_entry.'_build_when'});
            &add_refresh_clause($ob_entry);
        }

        if ($params{$ob_entry.'_rewrite'} =~
/[tT][rR][uU][eE]/)
        {
            $cmd .= "enable query rewrite\n";
        }
        elsif ($params{$ob_entry.'_rewrite'} =~
/[fF][aA][lL][sS][eE]/)
        {
            $cmd .= "disable query rewrite\n";
        }
    }

    if (defined $params{$ob_entry.'_as_select'})
# as select (here I only copy a variable called
<viewname>_define_as_select
    {
        &print_as_select_stm($params{$ob_entry.'_
as_select'});
        $cmd .= "\n";
    }

    if (($ob_type cmp 'viewlog') == 0)
    {
        $cmd .= "with rowid\n";
        if (defined
$params{$ob_entry.'_columns'})
        {
            $cmd .= "(\n";
            @ob_collist = split(/,/,
$params{$ob_entry.'_columns'});
            &add_columns;     # the list of
columns are in @ob_collist
            $cmd .= ")\n";
        }
        $cmd .= "including new values\n";
    }

    if ($params{$ob_entry.'_creation_type'} =~
/[wW][iI][tT][hH]\s[tT][aA][bB][lL][eE]/)
    {
        $cmd .= ";\ndrop materialized view
$object;\n";
        $cmd .= "create materialized view $object
\n";
        $cmd .= "BUILD
".$params{$ob_entry.'_build_when'}."\n" if
(defined $params{$ob_entry.'_build_when'});
        $cmd .= "on prebuilt table \n";
#       if (defined
$params{$ob_entry.'_precision'})
#       {
#           ($params{$ob_entry.'_precision'} =~
/[rR][eE][dD][uU][cC][eE][dD]/) ?
#              $cmd .= "WITH REDUCED PRECISION
\n"
#            : $cmd .= "WITHOUT REDUCED
PRECISION \n";
#       }

        &add_refresh_clause($ob_entry);

        if ((defined
$params{$ob_entry.'_rewrite'}) &&
($params{$ob_entry.'_rewrite'} =~
/[tT][rR][uU][eE]/))
        {
```

```
            $cmd .= "enable query rewrite\n";
        }
        elsif ($params{$ob_entry.'_rewrite'} =~
/[fF][aA][lL][sS][eE]/)
        {
            $cmd .= "disable query rewrite\n";
        }

        $cmd .= "refresh " .
$params{$ob_entry.'_refresh'} ."\n" if (defined
$params{$ob_entry.'_refresh'});
        $cmd .= $params{$ob_entry.'_queryrw'}
."\n" if (defined
$params{$ob_entry.'_queryrw'});
        &print_as_select_stm($params{$ob_entry.'_
as_select'});
    }
    $cmd .= ";\n";
}

sub add_columns
{
    while ($col = shift(@ob_collist))
    {
        (@ob_collist == 0) ? ($addcomma = "") :
($addcomma = ",");
        if (($ob_type cmp 'tab') == 0 )
        {
            $columntype =
$params{$ob_entry.'_'.$col.'_type'};
        }
        else
        {
            $columntype = "";
        }
        $nextline = sprintf ("    %-20s %s
%s$addcomma\n", $col, $columntype,
$params{$ob_entry.'_'.$col.'_cons'});
        $cmd .= $nextline;
    }
}


sub add_refresh_clause

{
    $_ob_entry = $_[0];
    if (defined
$params{$_ob_entry.'_refresh_how'})
    {
        $cmd .= "REFRESH
".$params{$_ob_entry.'_refresh_how'}."\n";
        $cmd .= "ON
".$params{$_ob_entry.'_refresh_when'}."\n" if
(defined $params{$_ob_entry.'_refresh_when'});
    }
}

sub print_as_select_stm
{
    if ($_[0] =~ /\/\*/)
    {
        ($beforehint,$rest)=split(/\/\*/,$_[0]);
        ($hint,$as_select) =split(/\*\//,$rest);
    }
    else
    {
        $as_select = $_[0];
        $hint="^";
    }
    $as_select =~ s/(, )|(,\t)/,\n /g;
    $as_select =~ s/from|FROM/\nFROM\n/g;
    $as_select =~ s/select|SELECT/SELECT\n/g;
    $as_select =~ s/where|WHERE/\nWHERE\n/g;
    $as_select =~ s/group by|GROUP BY/\nGROUP
BY\n/g;
    $as_select =~ s/ and | AND /\n AND \n/g;
    $as_select =~ s/ or | OR /\n OR \n/g;
    $cmd .= "as select\n";
    $cmd .= "/*".$hint."*/"."\n" if ($hint !~
/\^/);
    $cmd .= $as_select;
}
sub create_clusters
{
    $cmd .= "connect
$params{'user'}/$params{'passwd'};\n";
    foreach $cluster (@clulist)
    {
        $cmd .= "drop cluster $cluster including
tables;\n";
        $cmd .= "create cluster $cluster (\n";
        @clu_collist = split(/,/,
$params{'clu_'.$cluster.'_columns'});
        while ($col = shift(@clu_collist))
        {
            (@clu_collist == 0) ? ($addcomma =
"") : ($addcomma = ",");
            $nextline = sprintf ("    %-20s
%s$addcomma\n", $col,
$params{'clu_'.$cluster.'_'.$col.'_type'});
            $cmd .= $nextline;
        }
        $cmd .= ")\n";
        $cmd .= "pctfree
$params{'clu_'.$cluster.'_%f'}\n" if defined
$params{'clu_'.$cluster.'_%f'};
        $cmd .= "pctused
$params{'clu_'.$cluster.'_%u'}\n" if defined
$params{'clu_'.$cluster.'_%u'};
        $cmd .= "initrans
$params{'clu_'.$cluster.'_it'}\n" if defined
$params{'clu_'.$cluster.'_it'};
        $cmd .= "maxtrans
$params{'clu_'.$cluster.'_mt'}\n" if defined
$params{'clu_'.$cluster.'_mt'};
        $cmd .= "size
$params{'clu_'.$cluster.'_size'}\n" if defined
$params{'clu_'.$cluster.'_size'};
        $cmd .= "tablespace
$params{'clu_'.$cluster.'_ts'}\n" if defined
$params{'clu_'.$cluster.'_ts'};
        $cmd .= "storage
$params{'clu_'.$cluster.'_storage'}\n" if
defined $params{'clu_'.$cluster.'_storage'};
        $cmd .= "index\n" if
($params{'clu_'.$cluster.'_index'} =~ true);
        if (defined
$params{'clu_'.$cluster.'_hashkeys'})
        {
            $cmd .= "hash is
$params{'clu_'.$cluster.'_hashcol'} " if defined
$params{'clu_'.$cluster.'_hashcol'};
            $cmd .= "hashkeys
$params{'clu_'.$cluster.'_hashkeys'}\n";
        }
        if ((defined
$params{'clu_'.$cluster.'_pardeg'}) || (defined
$params{'clu_'.$cluster.'_parinst'}))
        {
            $cmd .= "parallel ";
            $cmd .= "degree
$params{'clu_'.$cluster.'_pardeg'} " if defined
$params{'clu_'.$cluster.'_pardeg'};
```

```
            $cmd .= "instances
$params{'clu_'.$cluster.'_parinst'}" if defined
$params{'clu_'.$cluster.'_parinst'};
            $cmd .= "\n";
        }
        $cmd .= "cache\n" if
$params{'clu_'.$cluster.'_cache'} =~ /true/;
        $cmd .= ";\n";
# Now, create the cluster index, if necessary
        if ($params{'clu_'.$cluster.'_index'} =~
true)
        {
            $cluindex = "ind_".$cluster;
            $cmd .= "drop index $cluindex;\n";
            $cmd .= "create index $cluindex on
cluster $cluster;\n";
        }
    }
    &dump1("*sql");
}


sub expand_partitions

# *MP* parameterize procedure: changed
$tab_entry into $ob_entry 1. parameter $_[0]
#                                       $table
into $ob           2. parameter $_[1]
#                              introduced
$ob_type               3. parameter $_[2]

{
    $ob_entry = $_[0];
    $ob = $_[1];
    $ob_type = $_[2];


    if ($params{$ob_entry.'_parttype'} =~
/[rR][aA][nN][gG][eE]/)
    {
        $cmd .= "partition by range (" .
$params{$ob_entry.'_partcol'} . ")\n(\n";
        &expand_partitions_doit($ob_entry,$ob,$ob
_type);
    }
    elsif ($params{$ob_entry.'_parttype'} =~
/[hH][aA][sS][hH]/)
    {
        $cmd .= "partition by hash (" .
$params{$ob_entry.'_partcol'} . ")\n";
        $cmd .= "partitions ".
$params{$ob_entry.'_#part'} . "\n";
    }
    elsif ($params{$ob_entry.'_parttype'} =~
/[cC][oO][mM][pP][oO][sS][iI][tT][eE]/)
    {
        $cmd .= "partition by range (" .
$params{$ob_entry.'_partcol'} . ")\n";
        $cmd .= "subpartition by hash(".
$params{$ob_entry.'_subpartcol'} . ")\n";
        $cmd .= "subpartitions ".
$params{$ob_entry.'_#subpart'} . "\n(\n";
        &expand_partitions_doit($ob_entry,$ob,$ob
_type);
    }
}                    # no partitioning


sub expand_partitions_doit

# *MP* parameterize procedure: changed
$tab_entry into $ob_entry 1. parameter $_[0]
#                                       $table
into $ob           2. parameter $_[1]
#                              introduced
$ob_type               3. parameter $_[2]

{
    $ob_entry = $_[0];
    $ob = $_[1];
    $ob_type = $_[2];

    @pcollist = split(/,/,
$params{$ob_entry.'_partcol'});
    if (!defined
$params{$ob_entry.'_partnames'})
    {
        # if no partnames are specified, use a
default part name
        for ($i = 1; $i <=
$params{$ob_entry.'_#part'}; $i++)
        {
            ($i == $params{$ob_entry.'_#part'}) ?
($addcomma = "") :
                ($addcomma = ",");
            $nextfile = sprintf("%s%s%d%s",
$ob,"_p", $i, $addcomma );
            $params{$ob_entry.'_partnames'} =
                $params{$ob_entry.'_partnames'} .
$nextfile;
        }
    }

    @ob_partnames =
split(/,/,$params{$ob_entry.'_partnames'});

    # if partnames is specified as XXXX#, then
expand
    if ($ob_partnames[0] =~ /\#/)
    {
        $filenm = shift(@ob_partnames);
        $savename = $filenm;
        $params{$ob_entry.'_partnames'} = "";
        for ($i = 1; $i <=
$params{$ob_entry.'_#part'}; $i++)
        {
            $filenm =~ s/\#/$i/g;
            ($i == $params{$ob_entry.'_#part'}) ?
($addcomma = "") :
                ($addcomma = ",");
            $params{$ob_entry.'_partnames'} =
                $params{$ob_entry.'_partnames'} .
$filenm . $addcomma;
            $filenm = $savename;
        }
        @ob_partnames =
split(/,/,$params{$ob_entry.'_partnames'});
        printf("Expanded $savename
to:\n$params{$ob_entry.'_partnames'}\n\n") if
$verbose;
    } else {
        if (@ob_partnames !=
$params{$ob_entry.'_#part'})
        {
            print "Number of partitions
$params{$ob_entry.'_#part'} for $ob doesn't
match\n _partnames parameter for
$params{$ob_entry.'_partnames'}\n";
            exit(-1);
        }
    }
    # now process the partition boundary

    &process_part_brys ($ob_entry,$ob);
```

```
    &process_part_ts ($ob_entry,$ob);

    # complete the partition statement

    for ($i=0; $i < $params{$ob_entry.'_#part'};
$i++)
    {
        $cmd .= "partition " . $ob_partnames[$i]
. " values less than ";
        if ($i==$params{$ob_entry.'_#part'}-1) {
          $cmd .= "(MAXVALUE)\n";
        }
        else {
          $cmd .= "(" . $ob_part_vals[$i] .
")\n";
        }
       &process_part_params($ob_type,'%f','pctfr
ee',$ob,$ob_partnames[$i]);
       &process_part_params($ob_type,'it','initr
ans',$ob,$ob_partnames[$i]);
       &process_part_params($ob_type,'mt','maxtr
ans',$ob,$ob_partnames[$i]);

       if (defined
$params{$ob_entry.'_'.$ob_partnames[$i].'_ts'})
       {
            $cmd .= 'tablespace ' .
$params{$ob_entry.'_'.$ob_partnames[$i].'_ts'} .
"\n";

&process_part_storage($ob_type,$ob,$ob_partnames
[$i]);
       } elsif (defined
$params{$ob_entry.'_part_ts'}) {
            $cmd .= 'tablespace ' .
$ob_part_ts[$i] . "\n";

&process_part_storage($ob_type,$ob,$ob_partnames
[$i]);
       }
       if (defined
$params{$ob_entry.'_'.$ob_partnames[$i].'_nolg'}
)
       {
            $cmd .= "nologging\n" if
($params{$ob_entry.'_'.$ob_partnames[$i].'_nolg'
} =~ /[tT][rR][uU][eE]/);
       } elsif (defined
$params{$ob_entry.'_part_def_nolg'}) {
            $cmd .= "nologging\n" if
($params{$ob_entry.'_part_def_nolg'} =~
/[tT][rR][uU][eE]/);
       }
       $cmd .= (($i+1) ==
$params{$ob_entry.'_#part'}) ? "" : ",\n";
    }
    $cmd .= ")\n";
} #end expand_partitions_doit


sub process_part_storage
{
    local($type,$tname,$pname) = @_;
    if (defined
$params{$type.'_'.$tname.'_'.$pname.'_storage'})
    {
        $cmd .= 'storage
'.$params{$type.'_'.$tname.'_'.$pname.'_storage'
}."\n";
    } elsif (defined
$params{$type.'_'.$tname.'_part_def_storage'}) {
        $cmd .= 'storage
'.$params{$type.'_'.$tname.'_part_def_storage'}.
"\n";
    }
}


sub process_part_params
{
    local($type,$p1,$p2,$tname,$pname) = @_;
    if (defined
$params{$type.'_'.$tname.'_'.$pname.'_'.$p1}) {
        $cmd .= $p2 . " " .
$params{$type.'_'.$tname.'_'.$pname.'_'.$p1}."\n
";
    } elsif (defined
$params{$type.'_'.$tname.'_part_def_'.$p1}) {
        $cmd .= $p2 . " " .
$params{$type.'_'.$tname.'_part_def_'.$p1} .
"\n";
    }
}


sub process_part_ts

# *MP* parameterize procedure: changed
'tab'.$table into $ob__entry 1. parameter $_[0]
(same as $_entry in expand_partition)
#                                   $table
into $ob_             2. parameter $_[1] (same
as $ob in expand_partition)

{
    $ob__entry = $_[0];
    $ob_ = $_[1];
    # is objecttype_<name>_part_ts is in the
form of XXXX#, expand it
    # else treat it as a comma separated list of
ts names
    if (defined $params{$ob__entry.'_part_ts'})
    {
        $nts=$params{$ob__entry.'_part_ts'};
        $nts=~s/#//;
        $nts=$params{$nts."_instances"};
        @ob_part_ts =
split(/,/,$params{$ob__entry.'_part_ts'});
        if ($ob_part_ts[0] =~ /\#/)
        {
            $filenm = shift(@ob_part_ts);
            $savename = $filenm;
            $params{$ob__entry.'_part_ts'} = "";
            $its = 0;
            for ($i = 1; $i <=
$params{$ob__entry.'_#part'}; $i++)
            {
                $its = $its + 1;
                if ($its > $nts) {
                    $its=1;
                }
                $filenm =~ s/\#/$its/g;
                ($i ==
$params{$ob__entry.'_#part'}) ? ($addcomma = "")
:
                    ($addcomma = ",");
                $params{$ob__entry.'_part_ts'} =

$params{$ob__entry.'_part_ts'} . $filenm .
$addcomma;
                $filenm = $savename;
            }
```

```
           @ob_part_ts =
split(/,/,$params{$ob__entry.'_part_ts'});
        } else {
           if (@ob_part_ts !=
$params{$ob__entry.'_#part'})
           {
              if (($params{$ob__entry.'_#part'}
% @ob_part_ts) == 0) {
                 $numfil = @ob_part_ts;
                 $p =
$params{$ob__entry.'_#part'} / $numfil;
                 $fil = "";
                 for ($i = 0; $i < $p; $i++)
                 {
                    if ($i == 0) {
                       $fil =
$params{$ob__entry.'_part_ts'};
                    } else {
                       $fil =
join(',',$fil,$params{$ob__entry.'_part_ts'});
                    }
                 }
                 @ob_part_ts =
split(/,/,$fil);
              } else {
                 print "Number of partitions
$params{$ob__entry.'_#part'} for object $ob_\n
doesn't match $table_part_ts parameter.\n";
                 exit (-1);
              }
           }
        }
    }
} # end of process_part_ts


sub process_part_brys


# *MP* parameterize procedure: changed
'tab'.$table into $ob__entry 1. parameter $_[0]
(same as $ob_entry in expand_partition)
#                                      $table
into $ob_            2. parameter $_[1] (same
as $ob in expand_partition)

{
    $ob__entry = $_[0];
    $ob_ = $_[1];

    $cnt = 0;
    @ob_part_vals = ();

    foreach $col (@pcollist)
    {
        # add quotes for character strings and
dates



        if (($params{$ob__entry.'_'.$col.'_type'}
=~ /[cC][hH][aA][rR]/) ||
           ($params{$ob__entry.'_'.$col.'_type'}
=~ /[dD][aA][tT][eE]/))
        {
            $addquote = "'";
        } else {
            $addquote = "";
        }

        @ob_part_col =
split(/,/,$params{$ob__entry.'_'.$col.'_partvals
'});
        if (@ob_part_col !=
$params{$ob__entry.'_#part'})
        {
            printf "Number of partition boundary
values %d for column $col in object $ob_ doesn't
match the number of partitions
($params{$ob__entry.'_#part'}) of the object\n",
($#ob_part_col + 1);
            exit(-1);
        }
        for ($i=0; $i <
$params{$ob__entry.'_#part'}; $i++)
        {
            ($cnt == $#pcollist) ? ($addcomma =
"") : ($addcomma = ",");
            if ($ob_part_col[$i] =~
/[Mm][Aa][Xx][Vv][Aa][Ll][Uu][Ee]/) {
                $addquote = "";
            }
            if
($params{$ob__entry.'_'.$col.'_type'} =~
/[dD][aA][tT][eE]/)
            {
                if ($i ==
$params{$ob_entry.'_#part'} - 1)
                {
                   $ob_part_vals[$i] .=
"MAXVALUE";
                } else {
                  $nls_format = (defined
$params{$ob__entry.'_'.$col.'_date_format'}) ?
$params{$ob__entry.'_'.$col.'_date_format'} :
"YYYY-MM-DD";
                   $ob_part_vals[$i] .=
"to_date('".$ob_part_col[$i]."','";
                   $ob_part_vals[$i] .=
$nls_format."')".$addcomma;
                }
            } else {
                $ob_part_vals[$i] =
$ob_part_vals[$i] . $addquote.
                   $ob_part_col[$i] . $addquote
. $addcomma ;
            }
        }
        $cnt++;
    }
} # end of process_part_brys

sub alttabs
{
    $di = shift(@_);
    $di =~ /(\d*)_(\d*)/;
    $cmd = "";
    $cmd .= "connect
$params{'user'}/$params{'passwd'};\n";
    $params{'alter_tables'} =
$params{'tab_tables'} if !defined
$params{'alter_tables'};
    @atabs = split(/,/,
$params{'alter_tables'});
    foreach $atab (@atabs)
    {
        $cmd .= "alter table $atab parallel
(degree $1 instances $2);\n";
    }
    &dump1("*sql");
    &dump0("*wait");
}

sub setinst
{
```

```
    $inum = shift(@_);
    &dump0("*inst");
    &dump0("{$inum}");
}

sub dump2
{
    $value = shift(@_);
    &dump0($value);
    &dump0("{");
    print OUTFILE $cmd;
    &dump0("}");
    &dump0("{");
    print OUTFILE $cmd2;
    &dump0("}");
    $cmd = "";
    $cmd2 = "";
    print "." if $verbose;
}

sub dump1
{
    $value = shift(@_);
    &dump0($value);
    &dump0("{");
    print OUTFILE $cmd;
    &dump0("}");
    $cmd = "";
    print "." if $verbose;
}

sub dump0
{
    $value = shift (@_);
    $value = $value . "\n" if ($value !~
/.*\n$/);
    print OUTFILE "$value";
}

sub recreate_drive_extended_part
{
    $cmd = "creapart -d PhysicalDrive" .
$drivenum . "\n";
    &dump1("*sh");
    $cmd = "Deleted partitions on PhysicalDrive"
. $drivenum;
    &time0($cmd);
    $cmd = "creapart -x PhysicalDrive" .
$drivenum . "\n";
    &dump1("*sh");
    $cmd = "Created extended partition on
PhysicalDrive" . $drivenum;
    &time0($cmd);
}

sub create_drive_part
{
    $numfiles = $params{$ts_entry.'_#files'}-1;
    if ($d == 0)
    {
        $size = (defined
$params{$ts_entry.'_first_size'}) ?
                $params{$ts_entry.'_first_size'}
:
                $params{$ts_entry.'_size'};
    }
    else
    {
        $size = $params{$ts_entry.'_size'};
    }
    $size =~ s/[Mm]*//g;
    @files =
split(/,/,$params{$ts_entry.'_datafiles'});
    foreach $file (@files)
    {
        &create_drive_part_file;
    }

}

sub create_drive_part_file
{
#   add 1MB to nt partition size to prevent
writing data to cyl 0
    $psize = $size+1;
    $drivenum = $file =~ /^log/ ?
$params{'plcre_log_drivenum'} :
        @drivenum[$d];
    $cmd = "creapart -l PhysicalDrive";
    $cmd .= $drivenum . " " . $psize . "\n";
    &dump1("*sh");
    $cmd = $file . "\tdevice\\PhysicalDrive" .
$drivenum;
    $cmd .=    "\\partition" .
++$partnum{$drivenum} . "\n";
    &time0($cmd);
    print LNKSFILE $cmd;
    if ($file =~ /^sys/ || $file =~ /^cntr/)
    {
        $d = $d < $md ? ++$d : 0;
    }
    elsif ($file =~ !/^log/)
    {
        $d = ($d < $md) && ($d < $numfiles) ?
++$d : 0;
    }
}

sub load_ctl_head
{
    print "control $control\n";
    open (CTLFILE, ">$control");
    print CTLFILE "---\n";
    print CTLFILE "---  $table.ctl for delimited
records\n" if ($ud == 1);
    print CTLFILE "---  $table.ctl for fixed-
length fields\n" if ($ud == 0);
    print CTLFILE "---\n\n";
    if (!($pre72) && defined
$params{'tab_'.$table.'_loadextent'})
    {
        print CTLFILE "options\n";
        print CTLFILE "(\n";
        print CTLFILE "storage = (initial
$params{'tab_'.$table.'_loadextent'} next
$params{'tab_'.$table.'_loadextent'})\n";
        print CTLFILE ")\n";
    }
    print CTLFILE "unrecoverable\n" if
($params{'load_unrecoverable'} =~
/[tT][rR][uU][eE]/);
    print CTLFILE "load\n";
    print CTLFILE "-- This is where INFILE
should go.\n";
}

sub load_ctl_tail
{
    print CTLFILE
"$params{'load_insert_type'}\n";
    print CTLFILE "fields terminated by
$params{'load_field_terminator'}\n" if ($ud ==
1);
```

```
    print CTLFILE "(\n";
    @tab_collist = split(/,/,
$params{'tab_'.$table.'_columns'});
    while ($col = shift(@tab_collist))
    {
        (@tab_collist == 0) ? ($addcomma = "") :
($addcomma = ",");
        $pos = "";
        $pos = sprintf ("position
(%s)",$params{'tab_'.$table.'_'.$col.'_pos'}) if
($ud == 0);
        $ctlline = sprintf ("  %-20s %s
%s$addcomma", $col, $pos,
$params{'tab_'.$table.'_'.$col.'_loadcolx'});
        print CTLFILE "$ctlline\n";
    }
    print CTLFILE ")\n";
    close (CTLFILE);
}

#*********************************************
*****************************
#*********************************************
*****************************
# Assigning Defaults
#*********************************************
*****************************
#*********************************************
*****************************

sub defaults
{
# defaults for the params associative array
    if (defined $bmtype)
    {
        eval(&$bmtype);
        &assigndefs;
    }
    &auxdefaults;
    &assigndefs;


$params{'dd_sql_area'}="$ENV{'ORACLE_HOME'}/rdbm
s/admin/" if !(defined $params{'dd_sql_area'});

$params{'dd_sqlplus_area'}="$ENV{'ORACLE_HOME'}/
sqlplus/admin/" if !(defined
$params{'dd_sqlplus_area'});

    if (defined $params{'compatible'})
    {
        $pre72 = 1 if $params{'compatible'} =~
/7\.(0|1)/;
        $pre73 = 1 if $params{'compatible'} =~
/7\.(0|1|2)/;
    }
    $params{'dbs_area'} =~
s/\?/$ENV{'ORACLE_HOME'}/g;
    $params{'load_tables'} =
$params{'tab_tables'} if !defined
$params{'load_tables'};
    foreach $phase (@phases)
    {
        $params{$phase.'_num_inst'} =
$params{'def_num_inst'} if !defined
$params{$phase.'_num_inst'};
        $params{$phase.'_max_bg'} =
$params{'max_bg'} if !defined
$params{$phase.'_max_bg'};
        $params{'startupfile_'.$phase} =
$params{'io_ifile'} if
!defined($params{'startupfile_'.$phase});
        $params{'startupfile_'.$phase} = sprintf
("/host%s", $params{'startupfile_'.$phase})
            if ($params{'special_machine'} eq
'ncube');
        #$params{'startupfile_'.$phase} =~
s/\?/$ENV{'ORACLE_HOME'}/g;
    }
    $params{'ops_nodes'} = 'undo' if
!defined($params{'ops_nodes'});
    @ops_nodes = split(/,/,$params{'ops_nodes'})
if defined $params{'ops_nodes'};
    $params{'db_maxinstances'} = @ops_nodes if
!defined $params{'db_maxinstances'};

    #----------------------- undo tablespace
definition ----------------------------------
    $params{'ts_undo_#rs'} =
$params{'def_num_inst'} if !defined
$params{'ts_undo_#rs'};
    $params{'ts_undo_area'} =
$params{'dbs_area'} if !defined
$params{'ts_undo_area'};
    $params{'ts_undo_rs_prefix'} = 'r' if
!defined $params{'ts_undo_rs_prefix'};
    $save_datafiles =
$params{'ts_undo_datafiles'};

&expand_filename('ts_undo_datafiles',$params{'ts
_undo_#files'});

    if (@ops_nodes > 1)
    {
        foreach $node (@ops_nodes)
        {
            $params{'ts_undo_'.$node.'_#files'} =
$params{'ts_undo_#files'}
            if
!defined($params{'ts_undo_'.$node.'_#files'});

            $df= $save_datafiles.'_'.$node;


$params{'ts_undo_'.$node.'_datafiles'} = $df
                if
!defined($params{'ts_undo_'.$node.'_datafiles'})
;

&expand_filename('ts_undo_'.$node.'_datafiles',$
params{'ts_undo_'.$node.'_#files'});


$params{'ts_undo_'.$node.'_first_size'} =
$params{'ts_undo_first_size'}
            if
!defined($params{'ts_undo_'.$node.'_first_size'}
);

            $params{'ts_undo_'.$node.'_#rs'} =
$params{'ts_undo_#rs'}
            if
!defined($params{'ts_undo_'.$node.'_#rs'});


$params{'ts_undo_'.$node.'_rs_storage'} =
$params{'ts_undo_rs_storage'}
            if
!defined($params{'ts_undo_'.$node.'_rs_storage'}
);

            $pr =
$params{'ts_undo_rs_prefix'}.$node;
```

```
$params{'ts_undo_'.$node.'_rs_prefix'} = $pr
            if
!defined($params{'ts_undo_'.$node.'_rs_prefix'})
;
        }
    }

    $params{'ts_log_#files'} =
$params{'ts_log_#threads'} *
$params{'ts_log_#files_pt'};
    $params{'ts_def_area'} = $params{'dbs_area'}
if !defined $params{'ts_def_area'};
    $params{'load_flatfile_area'} =
$params{'ts_def_area'} if !defined
$params{'load_flatfile_area'};
    $params{'load_controlfile_area'} =
$params{'ts_def_area'} if !defined
$params{'load_controlfile_area'};
    $params{'load_otherfile_area'} =
$params{'ts_def_area'} if !defined
$params{'load_otherfile_area'};

    $params{'ts_index_names'} = "ts_index" if
((!defined $params{'ts_index_names'}) &&
($params{'skip_ts'} !~ /index/));
    $params{'ts_temp_names'} = "ts_temp" if
((!defined $params{'ts_temp_names'}) &&
($params{'skip_ts'} !~ /temp/));
    $params{'ts_data_names'} = "ts_data" if
((!defined $params{'ts_data_names'}) &&
($params{'skip_ts'} !~ /data/));

    # expand ts_data groups

    if (defined $params{'ts_data_groups'})
    {
        @ts_data_group =
split(/,/,$params{'ts_data_groups'});
        foreach $gname (@ts_data_group)
        {
            $nts = $params{$gname.'_group_#ts'};
            for ($i=0;$i<$nts;$i++)
            {
                $tsn =
sprintf("%s%d",$gname,($i+1));
                $params{'ts_data_names'} .=
",".$tsn;
                $params{$tsn.'_datafiles'} =
$tsn."_#";
                $params{$tsn.'_#files'} =
$params{$gname.'_group_#files'} if defined
$params{$gname.'_group_#files'};
                $params{$tsn.'_option'} =
$params{$gname.'_group_option'} if defined
$params{$gname.'_group_option'};
                $params{$tsn.'_area'} =
$params{$gname.'_group_area'} if defined
$params{$gname.'_group_area'};
                $params{$tsn.'_storage'} =
$params{$gname.'_group_storage'} if defined
$params{$gname.'_group_storage'};
                $params{$tsn.'_first_size'} =
$params{$gname.'_group_first_size'} if defined
$params{$gname.'_group_first_size'};
                $params{$tsn.'_size'} =
$params{$gname.'_group_size'} if defined
$params{$gname.'_group_size'};
                $params{$tsn.'_nolg'} =
$params{$gname.'_nolg'} if defined
$params{$gname.'_nolg'};
            }
        }
    }

    @ts_data = split(/,/,
$params{'ts_data_names'});
    @ts_index = split(/,/,
$params{'ts_index_names'});
    @ts_temp = split(/,/,
$params{'ts_temp_names'});

# expand tablespaces
    @ts_data_new =();
    $ts_data_names="@@";
    foreach $ts_entry (@ts_data)
    {
        if
(defined($params{$ts_entry.'_instances'})){
            for ($i =
$params{$ts_entry.'_instances'}; $i > 0; $i --)
{
                $ts_new_entry = $ts_entry.$i;

        $params{$ts_new_entry.'_size'}=$params{$t
s_entry.'_size'};

        $params{$ts_new_entry.'_datafiles'}=$para
ms{$ts_entry.'_datafiles'};

        $params{$ts_new_entry.'_datafiles'}=~s/\#
/$i/;

        $params{$ts_new_entry.'_#files'}=$params{
$ts_entry.'_#files'};

        $params{$ts_new_entry.'_storage'}=$params
{$ts_entry.'_storage'};
                unshift (@ts_data_new,
$ts_new_entry);

        $ts_data_names=$ts_data_names.",".$ts_new
_entry;
            }
        }
        else {
            unshift (@ts_data_new, $ts_entry);

$ts_data_names=$ts_data_names.",".$ts_entry;
        }
    }
    $ts_data_names=~s/@@,//;
    #print "ts_data_names: $ts_data_names\n";
    @ts_data = @ts_data_new;
    #print "ts_all before substitution
@ts_all\n";
    #print "ts_all_new before substitution
@ts_all_new\n";
    #while (@ts_data) {
#       $ts = shift (@ts_data);
        #unshift (@ts_data_new,$ts);
    #}
    #foreach $ts_entry (@ts_data)
    #{
#       print "ts_entry: $ts_entry\n";
#       print "size:
$params{$ts_entry.'_size'}\n";
#       print "datafiles:
$params{$ts_entry.'_datafiles'}\n";
#       print "#files:
$params{$ts_entry.'_#files'}\n";
#       print "storage:
$params{$ts_entry.'_storage'}\n";
#   }
```

```
    #print "ts_all after substitution
@ts_all\n";
    #print "$ts_list\n";
    #exit(0);
    $mostts =
"$params{'ts_temp_names'},$ts_data_names,$params
{'ts_index_names'}";
    $allts =
"ts_sys,ts_log,$params{'ts_undo'},$params{'ts_te
mp_names'},$ts_data_names,$params{'ts_index_name
s'}";
    @ts_most = split(/,/, $mostts);
    @ts_all = split(/,/, $allts);

    foreach $ts_entry (@ts_most)
    {
        $params{$ts_entry.'_#files'} =
$params{'ts_def_#files'} if !defined
$params{$ts_entry.'_#files'};
        $params{$ts_entry.'_size'} =
$params{'ts_def_size'} if !defined
$params{$ts_entry.'_size'};
        $params{$ts_entry.'_storage'} =
$params{'ts_def_storage'} if !defined
$params{$ts_entry.'_storage'};
    }


    #rint "ts_all: @ts_all\n";
    foreach $ts_entry (@ts_all)
    {
       print "ts_entry $ts_entry\n";
        $params{$ts_entry.'_area'} =
$params{'ts_def_area'} if !defined
$params{$ts_entry.'_area'};
        $params{$ts_entry.'_options'} =
$params{'ts_def_options'} if !defined
$params{$ts_entry.'_options'};
        $params{$ts_entry.'_first_size'} =
$params{$ts_entry.'_size'} if !defined
$params{$ts_entry.'_first_size'};
       $params{$ts_entry.'_temporary'} = "false"
if !defined $params{$ts_entry.'_temporary'};
       $params{$ts_entry.'_managed'} = "user" if
!defined $params{$ts_entry.'_managed'};
        if (!defined
$params{$ts_entry.'_datafiles'})
        {
            for ($i = 0; $i <
$params{$ts_entry.'_#files'}; $i++)
            {
                ((i+1) ==
$params{$ts_entry.'_#files'}) ? ($addcomma = "")
: ($addcomma = ",");
                $nextfile = sprintf ("%s%d.f%s",
$ts_entry, $i+1, $addcomma);
                $params{$ts_entry.'_datafiles'}
= $params{$ts_entry.'_datafiles'} . $nextfile;
            }
        }
        @ts_datafiles = split(/,/,
$params{$ts_entry.'_datafiles'});
        $cntr = 1;
        if ($ts_datafiles[0] =~ /\#/) # we want
to replace all #'s
        {
            $filenm = shift(@ts_datafiles);
            $savename = $filenm;
            $params{$ts_entry.'_datafiles'} =
"";
            for ($i = 0; $i <
$params{$ts_entry.'_#files'}; $i++)
            {
                $filenm =~ s/\#/$cntr/g;
                $cntr++;
                (($i+1) ==
$params{$ts_entry.'_#files'}) ? ($addcomma = "")
: ($addcomma = ",");
                $params{$ts_entry.'_datafiles'}
= $params{$ts_entry.'_datafiles'} .
                    $filenm . $addcomma;
                $filenm = $savename;
            }
            printf ("Expanded $savename
to:\n$params{$ts_entry.'_datafiles'}\n\n") if
$verbose;
        }
        else
        {
            if (@ts_datafiles !=
$params{$ts_entry.'_#files'})
            {
                print "Number of files
($params{$ts_entry.'_#files'}) for $ts_entry
doesn't match\n _datafile parameter
$params{$ts_entry.'_datafiles'}\n";
                exit(-1);
            }
        }
    }
    $params{'io_control_files'} =
$params{'ts_sys_area'}.'t_cf1.f' if !defined
$params{'io_control_files'};

    @tab_list = split(/,/,
$params{'tab_tables'});
    foreach $table (@tab_list)
    {
        $params{'load_dbgen_'.$table.'_option_C'}
= $params{'load_dbgen_def_option_C'} if !defined
$params{'load_dbgen_'.$table.'_option_C'};

        $params{'tab_'.$table.'_ts'} = "ts_data"
if !defined $params{'tab_'.$table.'_ts'};
        if
(defined($params{'load_deg_parallel'})) {
            $params{'tab_'.$table.'_load_degpar'}
= $params{'load_deg_parallel'} if !defined
$params{'tab_'.$table.'_load_degpar'};
        } else {
            $params{'tab_'.$table.'_load_degpar'}
= 1 if !defined
$params{'tab_'.$table.'_load_degpar'};
        }

        #$params{'load_dbgen_'.$table.'_output_pr
efix'} = "$table.tbl" if !defined
$params{'load_dbgen_'.$table.'_output_prefix'};

        $params{'tab_'.$table.'_load_datf'} =
"$table.tbl" if !defined
$params{'tab_'.$table.'_load_datf'};

        $params{'tab_'.$table.'_load_badf'} =
"$table.badf" if !defined
$params{'tab_'.$table.'_load_badf'};

        $params{'tab_'.$table.'_load_ctlf'} =
"$table.ctl" if !defined
$params{'tab_'.$table.'_load_ctlf'};
```

```
       $params{'tab_'.$table.'_load_logf'} =
"$table.log" if !defined
$params{'tab_'.$table.'_load_logf'};

       $params{'tab_'.$table.'_load_direct'} =
"true" if !defined
$params{'tab_'.$table.'_load_direct'};

       $params{'tab_'.$table.'_#rows'} *=
$params{'scale_factor'};
    }

    @view_list = split(/,/,
$params{'view_views'});
    foreach $view (@view_list)
    {
        $params{'view_'.$view.'_ts'} = "ts_data"
if !defined $params{'view_'.$view.'_ts'};
       $params{'view_'.$view.'_load_degpar'} =
$params{'load_deg_parallel'} if !defined
$params{'view_'.$view.'_load_degpar'};
       $params{'view_'.$view.'_#rows'} *=
$params{'scale_factor'};
       $params{$ob_entry.'_rewrite'} = "" if
!defined $params{$ob_entry.'_rewrite'};
    }

    @tablelog_list = split(/,/,
$params{'tablelog_list'});

    @clulist = split(/,/,
$params{'clu_clusters'});
    foreach $cluster (@clulist)
    {
        $params{'clu_'.$cluster.'_ts'} =
"ts_data" if !defined
$params{'clu_'.$cluster.'_ts'};
    }

    @constlist = split(/,/,
$params{'con_constraints'});
    foreach $const (@constlist)
    {
        $params{'con_'.$const.'_ts'} =
"ts_index" if ((!defined
$params{'con_'.$const.'_ts'}) &&
($params{'skip_ts'} !~ /index/));
    }

    @indexlist = split(/,/,
$params{'ind_indices'});
    foreach $index (@indexlist)
    {
        $params{'ind_'.$index.'_ts'} =
"ts_index" if ((!defined
$params{'ind_'.$index.'_ts'}) &&
($params{'skip_ts'} !~ /index/));
    }
    $params{'analyze_type'} = "via package
dbms.stats" if
!defined($params{'analyze_type'});
    @anlyzlist = split(/,/,
$params{'anl_objects'});
    @anlyzlist = "schema" if (@anlyzlist == 0);
    foreach $object (@anlyzlist)
    {
       $params{'anl_'.$object.'_object_type'} =
"schema" if ($object =~ /schema/);
       $params{'anl_'.$object.'_object_type'} =
"table" if !defined
$params{'anl_'.$object.'_object_type'};
       $params{'anl_'.$object.'_anl_type'} =
"gather" if !defined
$params{'anl_'.$object.'_anl_type'};
    }
    @choblist = split(/,/,
$params{'chob_objects'});
#    foreach $object (@choblist)
#    {
#      $choblist{$object} =
$params{'chob_'.$object.'_pardeg'};
#    }
    @chstorlist = split(/,/,
$params{'chstor_objects'});
    foreach $object (@chstorlist)
    {
      $chstorlist{$object} =
$params{'chstor_'.$object.'_next'};
    }
    if ((defined $params{'verbose'}) &&
(($params{'verbose'} =~ /[tT][rR][uU][eE]/) ||
($params{'verbose'} =~ 1))){
       $verbose=1;}
    else {
       $verbose=0;}
    if ((defined $params{'log_output'}) &&
(($params{'log_output'} =~ /[tT][rR][uU][eE]/)
|| ($params{'log_output'} =~ 1))){
       $log_output=1;}
    else {
       $log_output=0;}

$params{'plcre_recreate_extended_partitions'} =
"false" if !defined
($params{'plcre_recreate_extended_partitions'});
}

sub assigndefs
{
    while ($onedef = shift(@defparams))
    {
        $onedef =~ /^(.*)=(.*)$/;
        $key = $1;
        $value = $2;
        $params{$key} = $value if !defined
$params{$key};
    }
}

sub auxdefaults
{
    @defparams = ();
    push (@defparams, 'scale_factor=0.1');
    push (@defparams, 'dbs_area=?/dbs/');
    push (@defparams, 'dbs_area=/tmp');
    push (@defparams,
'dd_sql_area=?/rdbms/admin/');
    push (@defparams, 'max_bg=-1');
    push (@defparams, 'user=bmuser');
    push (@defparams, 'passwd=bmpasswd');
    push (@defparams, 'load_type=delim');
    push (@defparams,
'load_field_terminator=whitespace');
    push (@defparams, 'ts_def_#files=1');
    push (@defparams, 'ts_def_options=reuse');
    push (@defparams, 'ts_def_first_size=1m');
    push (@defparams, 'ts_def_size=1m');
    push (@defparams, 'ts_sys_#files=1');
    push (@defparams, 'ts_log_#threads=1');
    push (@defparams, 'ts_log_#files_pt=2');
    push (@defparams, 'ts_undo_#files=1');
    push (@defparams, 'ts_sys_size=25m');
    push (@defparams, 'ts_log_size=2m');
```

```
    push (@defparams, 'ts_undo_size=10m');
#    push (@defparams, 'ts_data_names=ts_data');
    push (@defparams,
'load_insert_type=append');
    push (@defparams, 'load_deg_parallel=1');
    push (@defparams, 'special_machine=');
    push (@defparams, 'shutdown_level=abort');
    push (@defparams, 'def_num_inst=1');
#    push (@defparams,
'privileges=resource,unlimited
tablespace,connect');
    #push (@defparams, 'privileges=DBA,query
rewrite,global query rewrite');
    push (@defparams, 'privileges=DBA');
    push (@defparams, 'qry_number_trials=1');
}

sub tpcd
{
    @defparams = ();
    push (@defparams, 'user=tpcd');
    push (@defparams, 'passwd=tpcd');
    push (@defparams,
'tab_tables=lineitem,orders,partsupp,part,custom
er,supplier,nation,region');
    push (@defparams,
'view_views=mav_li,mv_locs');
    push (@defparams,
'viewlog_list=lineitem,orders');
    push (@defparams,
'mav_li_columns=count_p_group,l_shipdate,l_retur
nflag,l_linestatus,sum_qty,sum_base_price
,sum_disc_price ,count_disc_price,sum_charge,
count_charge,count_qty,count_price,sum_disc,coun
t_disc,count_order');
    push (@defparams,
'tab_lineitem_columns=l_orderkey,l_partkey,l_sup
pkey,l_linenumber,l_quantity,l_extendedprice,l_d
iscount,l_tax,l_returnflag,l_linestatus,l_shipda
te,l_commitdate,l_receiptdate,l_shipinstruct,l_s
hipmode,l_comment');
    push (@defparams,
'tab_orders_columns=o_orderkey,o_custkey,o_order
status,o_totalprice,o_orderdate,o_orderpriority,
o_clerk,o_shippriority,o_comment');
    push (@defparams,
'tab_partsupp_columns=ps_partkey,ps_suppkey,ps_a
vailqty,ps_supplycost,ps_comment');
    push (@defparams,
'tab_part_columns=p_partkey,p_name,p_mfgr,p_bran
d,p_type,p_size,p_container,p_retailprice,p_comm
ent');
    push (@defparams,
'tab_customer_columns=c_custkey,c_name,c_address
,c_nationkey,c_phone,c_acctbal,c_mktsegment,c_co
mment');
    push (@defparams,
'tab_supplier_columns=s_suppkey,s_name,s_address
,s_nationkey,s_phone,s_acctbal,s_comment');
    push (@defparams,
'tab_nation_columns=n_nationkey,n_name,n_regionk
ey,n_comment');
    push (@defparams,
'tab_region_columns=r_regionkey,r_name,r_comment
');
    push (@defparams,
'tab_lineitem_#rows=6100000'); # Somewhat of a
random value
    push (@defparams,
'tab_orders_#rows=1500000');
    push (@defparams,
'tab_partsupp_#rows=800000');
    push (@defparams, 'tab_part_#rows=200000');
    push (@defparams,
'tab_customer_#rows=150000');
    push (@defparams,
'tab_supplier_#rows=10000');
    push (@defparams, 'tab_nation_#rows=0');
    push (@defparams, 'tab_region_#rows=0');
    push (@defparams,
'tab_lineitem_parttype=range');
    push (@defparams,
'tab_lineitem_l_orderkey_type=number');
    push (@defparams,
'tab_lineitem_l_partkey_type=number');
    push (@defparams,
'tab_lineitem_l_suppkey_type=number');
    push (@defparams,
'tab_lineitem_l_linenumber_type=number');
    push (@defparams,
'tab_lineitem_l_quantity_type=number');
    push (@defparams,
'tab_lineitem_l_extendedprice_type=number');
    push (@defparams,
'tab_lineitem_l_discount_type=number');
    push (@defparams,
'tab_lineitem_l_tax_type=number');
    push (@defparams,
'tab_lineitem_l_returnflag_type=char(1)');
    push (@defparams,
'tab_lineitem_l_linestatus_type=char(1)');
    push (@defparams,
'tab_lineitem_l_shipdate_type=date');
    push (@defparams,
'tab_lineitem_l_commitdate_type=date');
    push (@defparams,
'tab_lineitem_l_receiptdate_type=date');
    push (@defparams,
'tab_lineitem_l_shipinstruct_type=char(25)');
    push (@defparams,
'tab_lineitem_l_shipmode_type=char(10)');
    push (@defparams,
'tab_lineitem_l_comment_type=varchar(44)');
    push (@defparams,
'tab_lineitem_l_orderkey_pos=13:24');
    push (@defparams,
'tab_lineitem_l_partkey_pos=25:36');
    push (@defparams,
'tab_lineitem_l_suppkey_pos=37:48');
    push (@defparams,
'tab_lineitem_l_linenumber_pos=49:60');
    push (@defparams,
'tab_lineitem_l_quantity_pos=61:72');
    push (@defparams,
'tab_lineitem_l_extendedprice_pos=73:87');
    push (@defparams,
'tab_lineitem_l_discount_pos=88:102');
    push (@defparams,
'tab_lineitem_l_tax_pos=103:117');
    push (@defparams,
'tab_lineitem_l_returnflag_pos=118:118');
    push (@defparams,
'tab_lineitem_l_linestatus_pos=120:120');
    push (@defparams,
'tab_lineitem_l_shipdate_pos=122:131');
    push (@defparams,
'tab_lineitem_l_commitdate_pos=135:144');
    push (@defparams,
'tab_lineitem_l_receiptdate_pos=148:157');
    push (@defparams,
'tab_lineitem_l_shipinstruct_pos=161:185');
    push (@defparams,
'tab_lineitem_l_shipmode_pos=186:195');
```

```
    push (@defparams,
'tab_lineitem_l_comment_pos=196:239');
    push (@defparams,
'tab_lineitem_l_shipdate_loadcolx=date "yyyy-mm-
dd"');
    push (@defparams,
'tab_lineitem_l_commitdate_loadcolx=date "yyyy-
mm-dd"');
    push (@defparams,
'tab_lineitem_l_receiptdate_loadcolx=date "yyyy-
mm-dd"');
    push (@defparams,
'tab_orders_parttype=range');
    push (@defparams,
'tab_orders_o_orderkey_type=number');
    push (@defparams,
'tab_orders_o_custkey_type=number');
    push (@defparams,
'tab_orders_o_orderstatus_type=char(1)');
    push (@defparams,
'tab_orders_o_totalprice_type=number');
    push (@defparams,
'tab_orders_o_orderdate_type=date');
    push (@defparams,
'tab_orders_o_orderpriority_type=char(15)');
    push (@defparams,
'tab_orders_o_clerk_type=char(15)');
    push (@defparams,
'tab_orders_o_shippriority_type=number');
    push (@defparams,
'tab_orders_o_comment_type=varchar(79)');
    push (@defparams,
'tab_orders_o_orderkey_pos=13:24');
    push (@defparams,
'tab_orders_o_custkey_pos=25:36');
    push (@defparams,
'tab_orders_o_orderstatus_pos=37:37');
    push (@defparams,
'tab_orders_o_totalprice_pos=39:53');
    push (@defparams,
'tab_orders_o_orderdate_pos=54:63');
    push (@defparams,
'tab_orders_o_orderpriority_pos=67:81');
    push (@defparams,
'tab_orders_o_clerk_pos=82:96');
    push (@defparams,
'tab_orders_o_shippriority_pos=97:108');
    push (@defparams,
'tab_orders_o_comment_pos=109:187');
    push (@defparams,
'tab_orders_o_orderdate_loadcolx=date "yyyy-mm-
dd"');
    push (@defparams,
'tab_partsupp_parttype=range');
    push (@defparams,
'tab_partsupp_ps_partkey_type=number');
    push (@defparams,
'tab_partsupp_ps_suppkey_type=number');
    push (@defparams,
'tab_partsupp_ps_availqty_type=number');
    push (@defparams,
'tab_partsupp_ps_supplycost_type=number');
    push (@defparams,
'tab_partsupp_ps_comment_type=varchar(199)');
    push (@defparams,
'tab_partsupp_ps_partkey_pos=1:12');
    push (@defparams,
'tab_partsupp_ps_suppkey_pos=13:24');
    push (@defparams,
'tab_partsupp_ps_availqty_pos=25:36');
    push (@defparams,
'tab_partsupp_ps_supplycost_pos=37:51');
    push (@defparams,
'tab_partsupp_ps_comment_pos=52:250');
    push (@defparams,
'tab_part_parttype=range');
    push (@defparams,
'tab_part_p_partkey_type=number');
    push (@defparams,
'tab_part_p_name_type=varchar(55)');
    push (@defparams,
'tab_part_p_mfgr_type=char(25)');
    push (@defparams,
'tab_part_p_brand_type=char(10)');
    push (@defparams,
'tab_part_p_type_type=varchar(25)');
    push (@defparams,
'tab_part_p_size_type=number');
    push (@defparams,
'tab_part_p_container_type=char(10)');
    push (@defparams,
'tab_part_p_retailprice_type=number');
    push (@defparams,
'tab_part_p_comment_type=varchar(23)');
    push (@defparams,
'tab_part_p_partkey_pos=1:12');
    push (@defparams,
'tab_part_p_name_pos=13:67');
    push (@defparams,
'tab_part_p_mfgr_pos=68:92');
    push (@defparams,
'tab_part_p_brand_pos=93:102');
    push (@defparams,
'tab_part_p_type_pos=103:127');
    push (@defparams,
'tab_part_p_size_pos=128:139');
    push (@defparams,
'tab_part_p_container_pos=140:149');
    push (@defparams,
'tab_part_p_retailprice_pos=150:164');
    push (@defparams,
'tab_part_p_comment_pos=165:187');
    push (@defparams,
'tab_customer_parttype=range');
    push (@defparams,
'tab_customer_c_custkey_type=number');
    push (@defparams,
'tab_customer_c_name_type=varchar(25)');
    push (@defparams,
'tab_customer_c_address_type=varchar(40)');
    push (@defparams,
'tab_customer_c_nationkey_type=number');
    push (@defparams,
'tab_customer_c_phone_type=char(15)');
    push (@defparams,
'tab_customer_c_acctbal_type=number');
    push (@defparams,
'tab_customer_c_mktsegment_type=char(10)');
    push (@defparams,
'tab_customer_c_comment_type=varchar(117)');
    push (@defparams,
'tab_customer_c_custkey_pos=1:12');
    push (@defparams,
'tab_customer_c_name_pos=13:30');
    push (@defparams,
'tab_customer_c_address_pos=31:70');
    push (@defparams,
'tab_customer_c_nationkey_pos=71:82');
    push (@defparams,
'tab_customer_c_phone_pos=83:97');
    push (@defparams,
'tab_customer_c_acctbal_pos=98:112');
    push (@defparams,
'tab_customer_c_mktsegment_pos=113:122');
```

```
    push (@defparams,
'tab_customer_c_comment_pos=123:239');
    push (@defparams,
'tab_supplier_parttype=range');
    push (@defparams,
'tab_supplier_s_suppkey_type=number');
    push (@defparams,
'tab_supplier_s_name_type=char(25)');
    push (@defparams,
'tab_supplier_s_address_type=varchar(40)');
    push (@defparams,
'tab_supplier_s_nationkey_type=number');
    push (@defparams,
'tab_supplier_s_phone_type=char(15)');
    push (@defparams,
'tab_supplier_s_acctbal_type=number');
    push (@defparams,
'tab_supplier_s_comment_type=varchar(101)');
    push (@defparams,
'tab_supplier_s_suppkey_pos=1:12');
    push (@defparams,
'tab_supplier_s_name_pos=13:37');
    push (@defparams,
'tab_supplier_s_address_pos=38:77');
    push (@defparams,
'tab_supplier_s_nationkey_pos=78:89');
    push (@defparams,
'tab_supplier_s_phone_pos=90:104');
    push (@defparams,
'tab_supplier_s_acctbal_pos=105:119');
    push (@defparams,
'tab_supplier_s_comment_pos=120:220');
    push (@defparams,
'tab_nation_parttype=range');
    push (@defparams,
'tab_nation_n_nationkey_type=number');
    push (@defparams,
'tab_nation_n_name_type=char(25)');
    push (@defparams,
'tab_nation_n_regionkey_type=number');
    push (@defparams,
'tab_nation_n_comment_type=varchar(152)');
    push (@defparams,
'tab_nation_n_nationkey_pos=1:12');
    push (@defparams,
'tab_nation_n_name_pos=13:37');
    push (@defparams,
'tab_nation_n_regionkey_pos=38:49');
    push (@defparams,
'tab_nation_n_comment_pos=50:164');
    push (@defparams,
'tab_region_r_regionkey_type=number');
    push (@defparams,
'tab_region_r_name_type=char(25)');
    push (@defparams,
'tab_region_r_comment_type=varchar(152)');
    push (@defparams,
'tab_region_r_regionkey_pos=1:12');
    push (@defparams,
'tab_region_r_name_pos=13:37');
    push (@defparams,
'tab_region_r_comment_pos=38:152');
    push (@defparams,
'load_dbgen_lineitem_output_prefix='.'lineitem')
;
    push (@defparams,
'load_dbgen_orders_output_prefix='.'orders');
    push (@defparams,
'load_dbgen_customer_output_prefix='.'customer')
;
    push (@defparams,
'load_dbgen_part_output_prefix='.'part');
    push (@defparams,
'load_dbgen_partsupp_output_prefix='.'partsupp')
;
    push (@defparams,
'load_dbgen_supplier_output_prefix='.'supplier')
;
    push (@defparams,
'load_dbgen_nation_output_prefix='.'nation');
    push (@defparams,
'load_dbgen_region_output_prefix='.'region');
    push (@defparams, 'load_type=fixed');
}


sub wisc
{
    @defparams = ();
    push (@defparams, 'user=wisc');
    push (@defparams, 'passwd=wisc');
    push (@defparams, 'tab_tables=wisc');
    push (@defparams,
'load_field_terminator=whitespace');
    push (@defparams,
'tab_wisc_columns=unique1,unique2,two,four,ten,t
wenty,hundred,thousand,fivethous,tenthous,odd100
,even100,stringu1,stringu2,string4');
    push (@defparams, 'tab_wisc_#rows=1000000');
    push (@defparams,
'tab_wisc_unique1_type=number');
    push (@defparams,
'tab_wisc_unique2_type=number');
    push (@defparams,
'tab_wisc_two_type=number');
    push (@defparams,
'tab_wisc_four_type=number');
    push (@defparams,
'tab_wisc_ten_type=number');
    push (@defparams,
'tab_wisc_twenty_type=number');
    push (@defparams,
'tab_wisc_hundred_type=number');
    push (@defparams,
'tab_wisc_thousand_type=number');
    push (@defparams,
'tab_wisc_fivethous_type=number');
    push (@defparams,
'tab_wisc_tenthous_type=number');
    push (@defparams,
'tab_wisc_odd100_type=number');
    push (@defparams,
'tab_wisc_even100_type=number');
    push (@defparams,
'tab_wisc_stringu1_type=varchar(52)');
    push (@defparams,
'tab_wisc_stringu2_type=varchar(52)');
    push (@defparams,
'tab_wisc_string4_type=varchar(52)');
    push (@defparams,
'tab_wisc_unique1_loadcolx=integer external');
    push (@defparams,
'tab_wisc_unique2_loadcolx=integer external');
    push (@defparams,
'tab_wisc_two_loadcolx=integer external');
    push (@defparams,
'tab_wisc_four_loadcolx=integer external');
    push (@defparams,
'tab_wisc_ten_loadcolx=integer external');
    push (@defparams,
'tab_wisc_twenty_loadcolx=integer external');
    push (@defparams,
'tab_wisc_hundred_loadcolx=integer external');
```

```
    push (@defparams,
'tab_wisc_thousand_loadcolx=integer external');
    push (@defparams,
'tab_wisc_fivethous_loadcolx=integer external');
    push (@defparams,
'tab_wisc_tenthous_loadcolx=integer external');
    push (@defparams,
'tab_wisc_odd100_loadcolx=integer external');
    push (@defparams,
'tab_wisc_even100_loadcolx=integer external');
    push (@defparams,
'tab_wisc_stringu1_loadcolx=char(52)');
    push (@defparams,
'tab_wisc_stringu2_loadcolx=char(52)');
    push (@defparams,
'tab_wisc_string4_loadcolx=char(52)');
}

sub expand_filename
{
    # 1. parameter: name of datafiles in
associative array (params)
    # 2. parameter: number of files the name
should be expanded to
    @tsdf = split(/,/, $params{$_[0]});
    $no_files = $_[1];
    $cntr = 1;
    if ($tsdf[0] =~ /\#/) # we want to replace
all #'s
    {
        $filenm = shift(@tsdf);
        $savename = $filenm;
        $params{$_[0]} = "";
        for ($i = 0; $i < $no_files; $i++)
        {
            $filenm =~ s/\#/$cntr/g;
            $cntr++;
            (($i+1) == $no_files) ? ($addcomma =
"") : ($addcomma = ",");
            $params{$_[0]} .= $filenm .
$addcomma;
            $filenm = $savename;
        }
        printf ("Expanded $savename
to:\n$params{$_[1]}\n\n") if $verbose;
    }
    else
    {
        if (@tsdf != $no_files)
        {
            print "Number of files
(@ts_datafiles) for $_[1] doesn't match\n
_datafile parameter $no_files}\n";
            exit(-1);
        }
    }
}

sub time0
{
    if (($os cmp "nt") == 0)
        {
            $value = shift (@_);
            $value = "*time=" . $value;
            $value = $value . "\n" if ($value !~
/.*\n$/);
            print OUTFILE "$value";
            $value = "";
        }
}

sub read_parameter_description
{
    @paramdesc = ();
    stat ("$pdfile");
    if (-e _)
    {
        open (PDFILE, "$pdfile");
        while (<PDFILE>)
        {
            ($key,$desc)=split (/::/,$_,2);
#           $key = shift(@l);
            $key =~
s/(\w*)<(\w*)>(\w*)/$1\\w\*$3/g;
            $paramdesc{$key} = $desc;
        }
    }
    else
    {
        print "Parameterfile: $pdfile does not
exist\n  will continue without ...\n";
    }
}

sub key_exists
{
    $result = 1;
    return if ($runsilent == 1);
    @keys = keys %paramdesc;
    $p = $_[0]."!";
    while ($#keys >=0)
    {
        $key = pop(@keys);
        $key =$key."!";
        return if ($p =~ /$key/);
    }
    $result = -1;
}
```

Appendix C Acid Scripts

a_query.sql

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

set serverout on;

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

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

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

exit;
```

a_query2.sql

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

set serverout on;

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

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

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

exit;
```

atom.sh

```
#!/bin/ksh
#
# $Header: atom.sh 08-aug-99.13:48:02 mpoess Exp
$
#
# atom.sh
#
#  Copyright (c) Oracle Corporation 1999. All
Rights Reserved.
#
#    NAME
#      atom.sh - <one-line expansion of the
name>
#
#  DESCRIPTION
#      Performs atomicity tests.
#      Usage: atom.sh [-n iter] [-p prog] [-u
usr/pswd] -h
```

```
#
#      Options: See usage below
#
#    NOTES
#      <other useful comments, qualifications,
etc.>
#
#    MODIFIED   (MM/DD/YY)
#    mpoess      08/08/99 - Creation
#    mpoess      08/08/99 - Creation
#

. $KIT_DIR/env

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

usage() {

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

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

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

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

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

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

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

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

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

atranspl.c

```
/* Copyright (c) 2001, 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      10/17/01 - add parameter in
ACIDinit
   mpoess      02/22/01 - enlarge timing array
   mpoess      01/04/01 - Creation

*/

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

#include "atranspl.h"

/* Declare error handling functions */

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

/* declarations for ORDERS */

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

/* declarations for LINEITEM */

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

int l_quan = 0;
int l_newquan = 0;
double l_eprice = 0.0;
double l_neweprice = 0.0;
```

```
double l_disc = 0.0;
double l_tax = 0.0;

sb2 l_npricei;

/* other declarations */

int delta = 0;
double rprice;
double cost;

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

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

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

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

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

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

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

time_t curr_time;         /* Current Time
*/

/* OCI handles */

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

/* OCI bind handles */

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

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

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

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

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

char sqlstmt[1024];

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

void usage()
{

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

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

```
  fprintf(stderr,"    t<trigger>
:Trigger Time - sleep <trigger> seconds before
start\n\n");
  fprintf(stderr,"    s<sleep>
:Sleep Time - sleep <sleep> seconds before
commit or rollback\n\n");
  exit(-1);

}


void ACIDexit() {

  OCILogoff(tpcsvc,errhp);
  OCIhfree(tpcenv,OCI_HTYPE_STMT);
  OCIhfree(tpcsvc,OCI_HTYPE_SVCCTX);
  OCIhfree(tpcsrv,OCI_HTYPE_SERVER);
  OCIhfree(tpcusr,OCI_HTYPE_SESSION);

}



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

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

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

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

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

  ACIDexit();

  exit(1);
}

#ifdef LINUX
int main(argc,argv)
#else
void main(argc,argv)
#endif
     int argc;
     char *argv[];
{

  int i;
  char line[64];
  ub4 errcode;
  char msg[2048];
  int need_commit = 0;

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

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

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

  proc_no = atoi(argv[1]);

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

  num_streams = atoi(argv[2]);

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

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

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

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

  /* Process  optional parameters */

  argc -= 4;
  argv += 4;

  while(--argc) {
    ++argv;
    switch(argv[0][0]) {
    case 'u':
      strncpy((char *) lname, ++(argv[0]),
UNAME_LEN);
      if (strchr((char *) lname, '/') == NULL) {
```

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

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

  /* Initialize the cursors etc. */

  (void) ACIDinit();

  /* sleep for some time (triggering) */

  sleep(trig);

  /* start doing the ACID transactions */

  tr_start = gettime();

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

  while (fgets(line, 64, infile) != NULL) {

#ifdef NOLKEY
    sscanf(line, "%d %d\n", &o_key, &delta);

    /* Obtain l_key from l_key query */

    OCIsexec(tpcsvc,curi,errhp,1);

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

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

    /* Generate delta if necessary */

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

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

    curr_time = time(NULL);

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

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

    OCIsexec(tpcsvc,curr,errhp,1);

    curr_time = time(NULL);

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

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

    /* Shall we commit? */

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

if((status=OCITransCommit(tpcsvc,errhp,OCI_DEFAU
LT)) != OCI_SUCCESS) {
              OCIrol(tpcsvc,errhp);
              OCIsexec(tpcsvc,curr,errhp,1);
        } else {
            need_commit = 0;
            curr_time = time(NULL);
```

```
            FPRTF2(outfile, "ACID Transaction
iteration %d COMMITED at %s\n",
                num_iter, ctime(&curr_time));
          }
       }
    } else {
      OCIrol(tpcsvc,errhp);
      curr_time = time(NULL);
      FPRTF2(outfile, "ACID Transaction
iteration %d ROLLBACK at %s\n",
              num_iter, ctime(&curr_time));
    }

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

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

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

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

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

      OCIsexec(tpcsvc,cure1,errhp,1);
      OCIsexec(tpcsvc,cure2,errhp,1);

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

  tr_end = gettime();

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

  /*  Disconnect from ORACLE. */

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

  ACIDexit();

  exit(0);
}


void ACIDinit()
{

  /* run random seed */

  srand48(getpid());

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

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

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

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

```
  /* get username and password */

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

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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcsrv,0,OCI_ATT
R_SERVER,errhp);

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

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(p
asswd),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_ATT
R_SESSION,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_k
ey),SIZ(l_key),SQLT_INT);

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

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

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

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

OCIbbname(curr,l_skey_bp,errhp,":l_skey",ADR(l_s
key),SIZ(l_skey),SQLT_INT);

OCIbbname(curr,l_quan_bp,errhp,":l_quan",ADR(l_q
uan),SIZ(l_quan),SQLT_INT);

OCIbbname(curr,l_newquan_bp,errhp,":l_newquan",A
DR(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_d
isc),SIZ(l_disc),SQLT_FLT);

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

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


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

```
OCIbbname(curr,o_newtprice_bp,errhp,":o_newtpric
e",ADR(o_newtprice),
            SIZ(o_newtprice), SQLT_FLT);

OCIbbname(curr,rprice_bp,errhp,":rprice",ADR(rpr
ice),SIZ(rprice), SQLT_FLT);

OCIbbname(curr,cost_bp,errhp,":cost",ADR(cost),S
IZ(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_newepr
ice",ADR(l_neweprice),
            SIZ(l_neweprice),SQLT_FLT);

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

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

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


OCIbbname(cure2,o_newtprice2_bp,errhp,":o_newtpr
ice",ADR(o_newtprice),
            SIZ(o_newtprice),SQLT_FLT);

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

}
```

atranspl.h

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

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

   DESCRIPTION

   MODIFIED   (MM/DD/YY)
   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_DEFAULT)) != OCI_SUCCESS) \
       sql_error(errh,status,1); \
    else \
       DISCARD 0

#define
OCIbbname(stmh,bindp,errh,sqlvar,progv,progvl,ft
ype) \

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,f
type,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_DEFAUL
T)) != OCI_SUCCESS) \
        sql_error(errh,status,1); \
     else \
       DISCARD 0

#define ISOTXT "alter session set
isolation_level = serializable"
#define PDMLTXT "alter session force parallel
dml parallel (degree 4)"
#define PDDLTXT "alter session force parallel
ddl parallel (degree 4)"
#define OICATXT "alter session set
optimizer_index_cost_adj=25"

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

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

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

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

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

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

#endif /* ATRANSPL_H */
```

ckpt.sh

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

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

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

cnt_hist.sql

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

consist.sh

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

. $KIT_DIR/env

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

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

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

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

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

usage() {

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

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

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


if [ $ITER -lt 100 ]
then
echo "Error: Must at least run 100 iterations!"
echo "Exiting..."
```

```
exit 1
fi

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

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

i=0

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

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

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

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

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

i=0

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

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

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

wait

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

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

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

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

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

consist.sql

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

set verify off
rem set termout on
rem set echo on


REM
```

```
REM Get today's date.
REM

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

set serverout on;

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


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

    diff := l_tprice - o_tprice;

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

END;
.
/


spool off
exit
```

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

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${1}_
USER=tpch/tpch
TRIG=1
HCNT=duracnta

# get history count

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

# perform the consistency

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

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

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

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

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

USER=$DATABASE_USER
PROG=atranspl


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

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

```
usage() {

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

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

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


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

# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

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

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

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

sleep 1

# start ACID transaction, Sleep for 60 second
before COMMIT

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

# let's sleep 10 seconds before starting ACID
query

sleep 10

# start ACID query with the same OKEY

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

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

iso2.sh

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

. $KIT_DIR/env

RSH=rsh

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

DURA_DIR=$ACID_DIR/dura

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

```
ISOFILE=$OUT_DIR/iso2

USER=$DATABASE_USER
PROG=atranspl

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

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

usage() {

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

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

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


# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

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

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

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

sleep 1

# start ACID transaction, Sleep for 30 second
before ROLLBACK

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

# let's sleep 10 seconds before starting ACID
query

sleep 10

# start ACID query with the same OKEY

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

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

iso3.sh

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

. $KIT_DIR/env

# May need to change the following:
RSH=rsh

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

```
DURA_DIR=$ACID_DIR/dura

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

USER=$DATABASE_USER
PROG=atranspl

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

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

usage() {

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

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

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


# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

sleep 1

# start ACID transaction, Sleep for 30 second
before COMMIT

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

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

sleep 10

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

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

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

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

iso4.sh

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

. $KIT_DIR/env

# May need to change the following:
RSH=rsh

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

DURA_DIR=$ACID_DIR/dura

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

USER=$DATABASE_USER
PROG=atranspl

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

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

usage() {

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

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

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


# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

sleep 1

# start ACID transaction, Sleep for 30 second
before ROLLBACK

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

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

sleep 10

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

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

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

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

iso5.sh

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

. $KIT_DIR/env

# May need to change the following:
RSH=rsh

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

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

USER=$DATABASE_USER
PROG=atranspl

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

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

usage() {

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

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

while :
do
   case "$1" in
   -u) shift; USER=$1;;
```

```
   -n) shift; HOST="$1";;
   -h) usage; exit 0;;
   --) break;;
   esac
   shift;
done


# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

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 0.1`

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

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

wait

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

cat $TXN1FILE $TXN2FILE >> $ISOFILE

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

iso6.sh

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

. $KIT_DIR/env

# May need to change the following:
RSH=rsh

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

DURA_DIR=$ACID_DIR/dura

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

USER=$DATABASE_USER
PROG=atranspl

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

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

usage() {
```

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

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

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


# generate key files

randkey 1 0.1 u"$USER" > $KEYFILE

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) Oracle Corporation 2001. All
Rights Reserved. */

/*

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

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

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

#define ORDERCNT 150000.0

/* MK_SPARSE adopted from dss.h */

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

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

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

typedef struct aciddef {
  long okey;
  long lkey;
  int delta;
```

```
} adef;

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

/* OCI handles */

OCIEnv *tpcenv;
OCIServer *tpcsrv;
OCIError *errhp;
OCISvcCtx *tpcsvc;
OCISession *tpcusr;
OCIStmt *curi;

OCIBind *l_key_bp;
OCIBind *o_key_bp;

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

char sqlstmt[1024];


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


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

void sql_error(errhp,status,type)
     OCIError *errhp;
     sword status;
     sword type;
{
  char msg[2048];
  sb4 errcode;
  ub4 msglen;
  int i,j;

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

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

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

  ACIDexit();

  exit(1);
}


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

  long count;
  long i;
  double sf;         /* need to accomodate sf
0.1 */
  double random;
  double ordcnt;
  adef *res;

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

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

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

  argc -= 2;
  argv += 2;

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

```
  ACIDinit();

  /* initialize array for random numbers */

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

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

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

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

    /* Obtain l_key from l_key query */

    OCIsexec(tpcsvc,curi,errhp,1);

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

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

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

  ACIDexit();
  free(res);

}


void usage() {

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

void ACIDinit()
{

  /* run random seed */

  srand48(getpid());

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

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

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

  /* get username and password */

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

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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcsrv,0,OCI_ATT
R_SERVER,errhp);

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

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(p
asswd),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_ATT
R_SESSION,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),SQLT_INT);

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

randpsup.c

```
/* Copyright (c) Oracle Corporation 2001. 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      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 tpch/tpch @sample $j
i=`expr $i + 1`
done
```

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

sample.sql

```
Rem
Rem $Header: sample.sql 08-aug-99.17:10:34
mpoess Exp $
Rem
Rem sample.sql
Rem
Rem  Copyright (c) Oracle Corporation 1999. All
Rights Reserved.
Rem
Rem    NAME
Rem      sample.sql - <one-line expansion of the
name>
Rem
Rem    DESCRIPTION
Rem      <short description of component this
file declares/defines>
Rem
Rem    NOTES
Rem      <other useful comments, qualifications,
etc.>
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$$_
USER=tpch/tpch
TRIG=1
HCNT=duracntb

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

# get all the options

while :
do
   case "$1" in
   -n) shift; ITER=$1;;
   -s) shift; STEM=$1;;
```

```
   -p) shift; PROG=$1;;
   -i) shift; IN=$1;;
   -o) shift; OUT=$1;;
   -d) shift; DURA=$1;;
   -u) shift; USER=$1;;
   -h) usage; exit 0;;
   -t) shift; TRIG=$1;;
   -f) shift; SF=$1;;
   --) break;;
   esac
   shift;
done


echo "Starting ACID run..."

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

# Get history count before the run

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

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

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

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

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

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

done

wait

echo "ACID run completed"
```

Appendix D Query text and Output

1.log

```
Begin Execution at Mon Jun 10 15:54:21 2002


-- using default substitutions

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

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


4 rows processed.
Statement Processed in 21.63 seconds.

Ended Executing this Stream at Mon Jun 10
15:54:42 2002


Stream Started at 1023749661.30
Stream Ended at 1023749682.93
Stream Processed in 21.63 seconds


SQL statements processed: 1
```

2.log

```
Begin Execution at Mon Jun 10 15:54:43 2002


-- using default substitutions

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

S_ACCTBAL              S_NAME
N_NAME
P_PARTKEY              P_MFGR
S_ADDRESS                                S_PHONE
S_COMMENT
```

```
9938.53                 Supplier#000005359
UNITED KINGDOM
185358.00               Manufacturer#4
QKuHYh,vZGiwu2FWEJoLDx04                33-429-
790-6131
blithely silent pinto beans are furiously. slyly
final deposits acros
9937.84                 Supplier#000005969
ROMANIA
108438.00               Manufacturer#1
ANDENSOSmk,miq23Xfb5RWt6dvUcvt6Qa       29-520-
692-3537
carefully slow deposits use furiously. slyly
ironic platelets above the ironic
9936.22                 Supplier#000005250
UNITED KINGDOM
249.00                  Manufacturer#4
B3rqp0xbSEim4Mpy2RH J                   33-320-
228-2957
blithely special packages are. stealthily
express deposits across the closely final
instructi
9923.77                 Supplier#000002324
GERMANY
29821.00                Manufacturer#4
y3OD9UywSTOk                            17-779-
299-1839
quickly express packages breach quiet pinto
beans. requ
9871.22                 Supplier#000006373
GERMANY
43868.00                Manufacturer#5
J8fcXWsTqM                              17-813-
485-8637
never silent deposits integrate furiously blit
9870.78                 Supplier#000001286
GERMANY
81285.00                Manufacturer#2
YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH    17-516-
924-4574
final theodolites cajole slyly special,
9870.78                 Supplier#000001286
GERMANY
181285.00               Manufacturer#4
YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH    17-516-
924-4574
final theodolites cajole slyly special,
9852.52                 Supplier#000008973
RUSSIA
18972.00                Manufacturer#2
t5L67YdBYYH6o,Vz24jpDyQ9                 32-188-
594-7038
quickly regular instructions wake-- carefully
unusual braids into the expres
9847.83                 Supplier#000008097
RUSSIA
130557.00               Manufacturer#2
xMe97bpE69NzdwLoX                       32-375-
640-3593
slyly regular dependencies sleep slyly furiously
express dep
9847.57                 Supplier#000006345
FRANCE
86344.00                Manufacturer#1
VSt3rzk3qG698u6ld8HhOByvrTcSTSvQlDQDag   16-886-
766-7945
silent pinto beans should have to snooze
carefully along the final reques
9847.57                 Supplier#000006345
FRANCE
(rows truncated>

100 rows processed.
Statement Processed in 22.96 seconds.

Ended Executing this Stream at Mon Jun 10
15:55:06 2002


Stream Started at 1023749683.21
Stream Ended at 1023749706.17
Stream Processed in 22.96 seconds


SQL statements processed: 1
```

3.log

```
Begin Execution at Mon Jun 10 15:55:06 2002


-- using default substitutions

select * from (
select
l_orderkey,
sum(l_extendedprice * (1 - l_discount)) as
revenue,
o_orderdate,
o_shippriority
from
customer,
orders,
lineitem
where
c_mktsegment = 'BUILDING'
and c_custkey = o_custkey
and l_orderkey = o_orderkey
and o_orderdate < to_date( '1995-03-15', 'YYYY-
MM-DD')
and l_shipdate > to_date( '1995-03-15', 'YYYY-
MM-DD')
group by
l_orderkey,
o_orderdate,
o_shippriority
order by
revenue desc,
o_orderdate)
where rownum <= 10

L_ORDERKEY              REVENUE
O_ORDERDATE O_SHIPPRIORITY
2456423.00              406181.01
1995-03-05  0.00
3459808.00              405838.70
1995-03-04  0.00
492164.00               390324.06
1995-02-19  0.00
1188320.00              384537.94
1995-03-09  0.00
2435712.00              378673.06
1995-02-26  0.00
4878020.00              378376.80
1995-03-12  0.00
5521732.00              375153.92
1995-03-13  0.00
2628192.00              373133.31
1995-02-22  0.00
993600.00               371407.46
1995-03-05  0.00
```

```
2300070.00              367371.15
1995-03-13  0.00


10 rows processed.
Statement Processed in 13.83 seconds.

Ended Executing this Stream at Mon Jun 10
15:55:20 2002


Stream Started at 1023749706.44
Stream Ended at 1023749720.27
Stream Processed in 13.83 seconds


SQL statements processed: 1
```

4.log

```
Begin Execution at Mon Jun 10 15:55:20 2002


-- using default substitutions

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.
Statement Processed in 12.87 seconds.

Ended Executing this Stream at Mon Jun 10
15:55:33 2002


Stream Started at 1023749720.56
Stream Ended at 1023749733.43
Stream Processed in 12.87 seconds

SQL statements processed: 1
```

5.log

```
Begin Execution at Mon Jun 10 15:55:33 2002


-- using default substitutions

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.
Statement Processed in 30.41 seconds.

Ended Executing this Stream at Mon Jun 10
15:56:04 2002


Stream Started at 1023749733.70
Stream Ended at 1023749764.13
Stream Processed in 30.43 seconds


SQL statements processed: 1
```

6.log

```
Begin Execution at Mon Jun 10 15:56:04 2002


-- using default substitutions

select
sum(l_extendedprice * l_discount) as revenue
```

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

REVENUE
123141078.23


1 row  processed.
Statement Processed in 2.16 seconds.

Ended Executing this Stream at Mon Jun 10
15:56:06 2002


Stream Started at 1023749764.40
Stream Ended at 1023749766.56
Stream Processed in 2.16 seconds


SQL statements processed: 1
```

7.log

```
Begin Execution at Mon Jun 10 15:56:06 2002


-- using default substitutions

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.
Statement Processed in 17.61 seconds.

Ended Executing this Stream at Mon Jun 10
15:56:24 2002


Stream Started at 1023749766.83
Stream Ended at 1023749784.44
Stream Processed in 17.61 seconds


SQL statements processed: 1
```

8.log

```
Begin Execution at Mon Jun 10 15:56:24 2002


-- using default substitutions

select
o_year,
sum(case when nation='BRAZIL' then volume else 0
end )/ sum(volume)
as mkt_share
from
(
select
to_number (to_char (o_orderdate, 'yyyy')) as
o_year,
l_extendedprice * (1 - l_discount) as volume,
n2.n_name as nation
from
part,
supplier,
lineitem,
orders,
customer,
nation n1,
nation n2,
region
where
p_partkey = l_partkey
```

```
and s_suppkey = l_suppkey
and l_orderkey = o_orderkey
and o_custkey = c_custkey
and c_nationkey = n1.n_nationkey
and n1.n_regionkey = r_regionkey
and r_name = 'AMERICA'
and s_nationkey = n2.n_nationkey
and o_orderdate between to_date ('1995-01-01',
'YYYY-MM-DD') and to_date ('1996-12-31', 'YYYY-
MM-DD')
and p_type = 'ECONOMY ANODIZED STEEL'
) all_nations
group by
o_year
order by
o_year

O_YEAR                  MKT_SHARE
1995.00                 0.03
1996.00                 0.04


2 rows processed.
Statement Processed in 17.89 seconds.

Ended Executing this Stream at Mon Jun 10
15:56:42 2002


Stream Started at 1023749784.73
Stream Ended at 1023749802.62
Stream Processed in 17.89 seconds


SQL statements processed: 1
```

9.log

```
Begin Execution at Mon Jun 10 15:56:42 2002


-- using default substitutions

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
(rows truncated>


175 rows processed.
Statement Processed in 24.87 seconds.

Ended Executing this Stream at Mon Jun 10
15:57:07 2002


Stream Started at 1023749802.91
Stream Ended at 1023749827.78
Stream Processed in 24.87 seconds


SQL statements processed: 1
```

10.log

```
Begin Execution at Mon Jun 10 15:57:08 2002


-- using default substitutions

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.
Statement Processed in 22.89 seconds.

Ended Executing this Stream at Mon Jun 10
15:57:30 2002


Stream Started at 1023749828.06
Stream Ended at 1023749850.97
Stream Processed in 22.91 seconds


SQL statements processed: 1
```

11.log

```
Begin Execution at Mon Jun 10 15:57:31 2002


-- using default substitutions

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
<rows truncated>


1048 rows processed.
Statement Processed in 9.34 seconds.

Ended Executing this Stream at Mon Jun 10
15:57:40 2002


Stream Started at 1023749851.27
Stream Ended at 1023749860.61
Stream Processed in 9.34 seconds


SQL statements processed: 1
```

12.log

```
Begin Execution at Mon Jun 10 15:57:40 2002


-- using default substitutions

select
       l_shipmode,
       sum(case
              when o_orderpriority = '1-URGENT'
```

```
                        or o_orderpriority = '2-
HIGH'
                        then 1
                else 0
        end) as high_line_count,
        sum(case
                when o_orderpriority <> '1-
URGENT'
                        and o_orderpriority <> '2-
HIGH'
                        then 1
                else 0
        end) as low_line_count
from
        orders,
        lineitem
where
        o_orderkey = l_orderkey
        and l_shipmode in ('MAIL', 'SHIP')
        and l_commitdate < l_receiptdate
        and l_shipdate < l_commitdate
and l_receiptdate >= to_date( '1994-01-01',
'YYYY-MM-DD')
and l_receiptdate < add_months(to_date ('1994-
01-01',  'YYYY-MM-DD'),  12)
group by
        l_shipmode
order by
        l_shipmode

L_SHIPMODE HIGH_LINE_COUNT        LOW_LINE_COUNT
MAIL       6202.00                9324.00
SHIP       6200.00                9262.00


2 rows processed.
Statement Processed in 8.03 seconds.

Ended Executing this Stream at Mon Jun 10
15:57:48 2002


Stream Started at 1023749860.93
Stream Ended at 1023749868.96
Stream Processed in 8.03 seconds


SQL statements processed: 1
```

13.log

```
Begin Execution at Mon Jun 10 15:57:49 2002


-- using default substitutions

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

C_COUNT                CUSTDIST
0.00                   50004.00
9.00                   6641.00
10.00                  6566.00
11.00                  6058.00
8.00                   5949.00
12.00                  5553.00
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.
Statement Processed in 3.40 seconds.

Ended Executing this Stream at Mon Jun 10
15:57:52 2002


Stream Started at 1023749869.24
Stream Ended at 1023749872.64
Stream Processed in 3.40 seconds


SQL statements processed: 1
```

14.log

```
Begin Execution at Mon Jun 10 15:57:52 2002
```

```
-- using default substitutions

select
       100.00 * sum(case
              when p_type like 'PROMO%'
                     then l_extendedprice * (1
- l_discount)
              else 0
       end) / sum(l_extendedprice * (1 -
l_discount)) as promo_revenue
from
       lineitem,
       part
where
       l_partkey = p_partkey
       and l_shipdate >= date '1995-09-01'
       and l_shipdate < date '1995-09-01' +
interval '1' month

PROMO_REVENUE
16.38


1 row  processed.
Statement Processed in 0.73 seconds.

Ended Executing this Stream at Mon Jun 10
15:57:53 2002


Stream Started at 1023749872.93
Stream Ended at 1023749873.67
Stream Processed in 0.73 seconds


SQL statements processed: 1
```

15.log

```
Begin Execution at Mon Jun 10 15:57:53 2002


-- using default substitutions

with   revenue
as (select
l_suppkey supplier_no,
sum(l_extendedprice * (1 - l_discount))
total_revenue
from
lineitem
where
l_shipdate >= to_date( '1996-01-01', 'YYYY-MM-
DD')
and l_shipdate < add_months( to_date ('1996-01-
01', 'YYYY-MM-DD'), 3)
group by
l_suppkey)
select
s_suppkey,
s_name,
s_address,
s_phone,
total_revenue
from
supplier,
revenue
where
s_suppkey = supplier_no
and total_revenue = (
select
max(total_revenue)
from
revenue  )
order by
s_suppkey

S_SUPPKEY              S_NAME
S_ADDRESS                                S_PHONE
TOTAL_REVENUE
8449.00                Supplier#000008449
Wp34zim9qYFbVctdW                        20-469-
856-8873 1772627.21


1 row  processed.
Statement Processed in 2.74 seconds.

Ended Executing this Stream at Mon Jun 10
15:57:56 2002


Stream Started at 1023749873.95
Stream Ended at 1023749876.68
Stream Processed in 2.74 seconds


SQL statements processed: 1
```

16.log

```
Begin Execution at Mon Jun 10 15:57:56 2002


-- using default substitutions

select
p_brand,
p_type,
p_size,
count(distinct ps_suppkey) as supplier_cnt
from
partsupp,
part
where
p_partkey = ps_partkey
and p_brand <> 'Brand#45'
and p_type not like 'MEDIUM POLISHED%'
and p_size in (49, 14, 23, 45, 19, 3, 36, 9)
and ps_suppkey not in (
select
s_suppkey
from
supplier
where
s_comment like '%Customer%Complaints%'
)
group by
p_brand,
p_type,
p_size
order by
supplier_cnt desc,
p_brand,
p_type,
p_size

P_BRAND    P_TYPE                    P_SIZE
SUPPLIER_CNT
```

```
Brand#41   MEDIUM BRUSHED TIN        3.00
28.00
Brand#54   STANDARD BRUSHED COPPER   14.00
27.00
Brand#11   STANDARD BRUSHED TIN      23.00
24.00
Brand#11   STANDARD BURNISHED BRASS  36.00
24.00
Brand#15   MEDIUM ANODIZED NICKEL    3.00
24.00
Brand#15   SMALL ANODIZED BRASS      45.00
24.00
Brand#15   SMALL BURNISHED NICKEL    19.00
24.00
Brand#21   MEDIUM ANODIZED COPPER    3.00
24.00
Brand#22   SMALL BRUSHED NICKEL      3.00
24.00
Brand#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
<rows truncated>

18314 rows processed.
Statement Processed in 2.83 seconds.

Ended Executing this Stream at Mon Jun 10
15:57:59 2002


Stream Started at 1023749876.96
Stream Ended at 1023749879.78
Stream Processed in 2.83 seconds


SQL statements processed: 1
```

17.log

```
Begin Execution at Mon Jun 10 15:58:00 2002


-- using default substitutions

select
sum(l_extendedprice) / 7.0 as avg_yearly
from
lineitem,
part
where
p_partkey = l_partkey
and p_brand = 'Brand#23'
and p_container = 'MED BOX'
and l_quantity < (
select
0.2 * avg(l_quantity)
from
lineitem
where
l_partkey = p_partkey
)

AVG_YEARLY
348406.05


1 row  processed.
Statement Processed in 12.40 seconds.

Ended Executing this Stream at Mon Jun 10
15:58:12 2002


Stream Started at 1023749880.06
Stream Ended at 1023749892.46
Stream Processed in 12.40 seconds


SQL statements processed: 1
```

18.log

```
Begin Execution at Mon Jun 10 15:58:12 2002


-- using default substitutions

select  *  from  (
select
c_name,
c_custkey,
o_orderkey,
o_orderdate,
o_totalprice,
sum(l_quantity)
from
customer,
orders,
lineitem
where
o_orderkey in (
select
l_orderkey
from
lineitem
group by
l_orderkey having
sum(l_quantity) > 300
)
and c_custkey = o_custkey
and o_orderkey = l_orderkey
group by
c_name,
c_custkey,
o_orderkey,
o_orderdate,
o_totalprice
order by
o_totalprice desc,
o_orderdate
)
where rownum <= 100
```

C_NAME	C_CUSTKEY	O_ORDERKEY	O_ORDERDATE	O_TOTALPRICE	SUM(L_QUANTITY)
Customer#000128120	128120.00	4722021.00	1994-04-07	544089.09	323.00
Customer#000144617	144617.00	3043270.00	1997-02-12	530604.44	317.00
Customer#000013940	13940.00	2232932.00	1997-04-13	522720.61	304.00
Customer#000066790	66790.00	2199712.00	1996-09-30	515531.82	327.00
Customer#000046435	46435.00	4745607.00	1997-07-03	508047.99	309.00
Customer#000015272	15272.00	3883783.00	1993-07-28	500241.33	302.00
Customer#000146608	146608.00	3342468.00	1994-06-12	499794.58	303.00
Customer#000096103	96103.00	5984582.00	1992-03-16	494398.79	312.00
Customer#000024341	24341.00	1474818.00	1992-11-15	491348.26	302.00
Customer#000137446	137446.00	5489475.00	1997-05-23	487763.25	311.00
Customer#000107590	107590.00	4267751.00	1994-11-04	485141.38	301.00
Customer#000050008	50008.00	2366755.00	1996-12-09	483891.26	302.00
Customer#000015619	15619.00	3767271.00	1996-08-07	480083.96	318.00
Customer#000077260	77260.00	1436544.00	1992-09-12	479499.43	307.00
Customer#000109379	109379.00	5746311.00	1996-10-10	478064.11	302.00
Customer#000054602	54602.00	5832321.00	1997-02-09	471220.08	307.00
Customer#000105995	105995.00	2096705.00	1994-07-03	469692.58	307.00
Customer#000148885	148885.00	2942469.00	1992-05-31	469630.44	313.00
Customer#000114586	114586.00	551136.00	1993-05-19	469605.59	308.00
Customer#000105260	105260.00	5296167.00	1996-09-06	469360.57	303.00
Customer#000147197	147197.00	1263015.00	1997-02-02	467149.67	320.00
Customer#000064483	64483.00	2745894.00	1996-07-04	466991.35	304.00
Customer#000136573	136573.00	2761378.00	1996-05-31	461282.73	301.00
Customer#000016384	16384.00	502886.00	1994-04-12	458378.92	312.00
Customer#000117919	117919.00	2869152.00	1996-06-20	456815.92	317.00
Customer#000012251	12251.00	735366.00	1993-11-24	455107.26	309.00
Customer#000120098	120098.00	1971680.00	1995-06-14	453451.23	308.00
Customer#000066098	66098.00	5007490.00	1992-08-07	453436.16	304.00
Customer#000117076	117076.00	4290656.00	1997-02-05	449545.85	301.00
Customer#000129379	129379.00	4720454.00	1997-06-07	448665.79	303.00
Customer#000126865	126865.00	4702759.00	1994-11-07	447606.65	320.00
Customer#000088876	88876.00	983201.00	1993-12-30	446717.46	304.00
Customer#000036619	36619.00	4806726.00	1995-01-17	446704.09	328.00
Customer#000141823	141823.00	2806245.00	1996-12-29	446269.12	310.00
Customer#000053029	53029.00	2662214.00	1993-08-13	446144.49	302.00
Customer#000018188	18188.00	3037414.00	1995-01-25	443807.22	308.00
Customer#000066533	66533.00	29158.00	1995-10-21	443576.50	305.00
Customer#000037729	37729.00	4134341.00	1995-06-29	441082.97	309.00
Customer#000003566	3566.00	2329187.00	1998-01-04	439803.36	304.00
Customer#000045538	45538.00	4527553.00	1994-05-22	436275.31	305.00
Customer#000081581	81581.00	4739650.00	1995-11-04	435405.90	305.00
Customer#000119989	119989.00	1544643.00	1997-09-20	434568.25	320.00
Customer#000003680	3680.00	3861123.00	1998-07-03	433525.97	301.00
Customer#000113131	113131.00	967334.00	1995-12-15	432957.75	301.00
Customer#000141098	141098.00	565574.00	1995-09-24	430986.69	301.00
Customer#000093392	93392.00	5200102.00	1997-01-22	425487.51	304.00
Customer#000015631	15631.00	1845057.00	1994-05-12	419879.59	302.00

```
Customer#000112987        112987.00
4439686.00              1996-09-17
418161.49               305.00
Customer#000012599        12599.00
4259524.00              1998-02-12
415200.61               304.00
Customer#000105410        105410.00
4478371.00              1996-03-05
412754.51               302.00
Customer#000149842        149842.00
5156581.00              1994-05-30
411329.35               302.00
Customer#000010129        10129.00
5849444.00              1994-03-21
409129.85               309.00
Customer#000069904        69904.00
1742403.00              1996-10-19
408513.00               305.00
Customer#000017746        17746.00
6882.00                 1997-04-09
408446.93               303.00
Customer#000013072        13072.00
1481925.00              1998-03-15
399195.47               301.00
Customer#000082441        82441.00
857959.00               1994-02-07
382579.74               305.00
Customer#000088703        88703.00
2995076.00              1994-01-30
363812.12               302.00


57 rows processed.
Statement Processed in 12.77 seconds.

Ended Executing this Stream at Mon Jun 10
15:58:25 2002


Stream Started at 1023749892.75
Stream Ended at 1023749905.54
Stream Processed in 12.79 seconds


SQL statements processed: 1
```

19.log

```
Begin Execution at Mon Jun 10 15:58:25 2002


-- using default substitutions

select
sum(l_extendedprice* (1 - l_discount)) as
revenue
from
lineitem,
part
where
(
p_partkey = l_partkey
and p_brand = 'Brand#12'
and p_container in ('SM CASE', 'SM BOX', 'SM
PACK', 'SM PKG')
and l_quantity >= 1 and l_quantity <= 1 + 10
and p_size between 1 and 5
and l_shipmode in ('AIR', 'AIR REG')
and l_shipinstruct = 'DELIVER IN PERSON'
)
or
(
p_partkey = l_partkey
and p_brand = 'Brand#23'
and p_container in ('MED BAG', 'MED BOX', 'MED
PKG', 'MED PACK')
and l_quantity >= 10 and l_quantity <= 10 + 10
and p_size between 1 and 10
and l_shipmode in ('AIR', 'AIR REG')
and l_shipinstruct = 'DELIVER IN PERSON'
)
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.
Statement Processed in 11.66 seconds.

Ended Executing this Stream at Mon Jun 10
15:58:37 2002


Stream Started at 1023749905.82
Stream Ended at 1023749917.48
Stream Processed in 11.66 seconds


SQL statements processed: 1
```

20.log

```
Begin Execution at Mon Jun 10 15:58:37 2002


-- using default substitutions

select
s_name,
s_address
from
supplier,
nation
where
s_suppkey in (
select
ps_suppkey
from
partsupp
where
ps_partkey in (
select
p_partkey
from
part
where
p_name like 'forest%'
)
and ps_availqty > (
```

```
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.
Statement Processed in 2.62 seconds.

Ended Executing this Stream at Mon Jun 10
15:58:40 2002


Stream Started at 1023749917.77
Stream Ended at 1023749920.39
Stream Processed in 2.62 seconds


SQL statements processed: 1
```

21.log

```
Begin Execution at Mon Jun 10 15:58:40 2002


-- using default substitutions

select * from (
select
s_name,
count(*)  numwait
from
supplier,
lineitem l1,
orders,
nation
where
s_suppkey = l1.l_suppkey
and o_orderkey = l1.l_orderkey
and o_orderstatus = 'F'
and l1.l_receiptdate > l1.l_commitdate
and exists (
select
*
from
lineitem l2
where
l2.l_orderkey = l1.l_orderkey
and l2.l_suppkey <> l1.l_suppkey
)
and not exists (
select
*
from
lineitem l3
where
l3.l_orderkey = l1.l_orderkey
and l3.l_suppkey <> l1.l_suppkey
and l3.l_receiptdate > l3.l_commitdate
)
and s_nationkey = n_nationkey
and n_name = 'SAUDI ARABIA'
group by
s_name
order by
numwait desc,
s_name)
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.
Statement Processed in 47.53 seconds.

Ended Executing this Stream at Mon Jun 10
15:59:28 2002


Stream Started at 1023749920.67
Stream Ended at 1023749968.21
Stream Processed in 47.55 seconds


SQL statements processed: 1
```

22.log

```
Begin Execution at Mon Jun 10 15:59:28 2002


-- using default substitutions

select
cntrycode,
count(*) as numcust,
sum(c_acctbal) as totacctbal
from
(
select
substr(c_phone, 1, 2) as cntrycode,
c_acctbal
from
customer
where
substr(c_phone,1, 2) in
('13', '31', '23', '29', '30', '18', '17')
and c_acctbal > (
select
avg(c_acctbal)
from
customer
where
c_acctbal > 0.00
and substr(c_phone, 1, 2) in
('13', '31', '23', '29', '30', '18', '17')
)
and not exists (
select
*
from
orders
where
o_custkey = c_custkey
)
) custsale
group by
cntrycode
order by
cntrycode

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.
Statement Processed in 3.67 seconds.
```

```
Ended Executing this Stream at Mon Jun 10
15:59:32 2002


Stream Started at 1023749968.52
Stream Ended at 1023749972.19
Stream Processed in 3.67 seconds


SQL statements processed: 1
```

Appendix E Seed and Input Parameters

seed

```
603234609
```

stream00

```
14      1995-08-01
2       5       COPPER  AFRICA
9       navy
20      midnight        1996-01-01      VIETNAM
6       1996-01-01      0.02    25
17      Brand#21        SM CASE
18      312
8       ROMANIA EUROPE  LARGE POLISHED BRASS
21      MOROCCO
13      unusual accounts
3       MACHINERY       1995-03-05
22      20      15      13      24      33      34
        11
16      Brand#22        SMALL ANODIZED 14       20
        17      11      5       7       6       43
4       1993-10-01
11      BRAZIL  0.0000001000
15      1993-02-01
1       72
10      1994-08-01
19      Brand#13        Brand#45        Brand#12
        2       20      25
5       AFRICA  1996-01-01
7       ARGENTINA       ROMANIA
12      MAIL    TRUCK   1993-01-01
```

stream01

```
21      GERMANY
3       FURNITURE       1995-03-22
18      314
5       AMERICA 1996-01-01
11      MOROCCO 0.0000001000
7       CHINA   IRAQ
6       1996-01-01      0.07    24
20      white   1994-01-01      IRAQ
17      Brand#23        SM JAR
12      TRUCK   MAIL    1995-01-01
16      Brand#52        LARGE BURNISHED         7
        24      15      23      31      37      33
        12
15      1995-09-01
13      unusual accounts
10      1993-05-01
2       43      STEEL   EUROPE
8       IRAQ    MIDDLE EAST     LARGE BURNISHED
BRASS
14      1995-11-01
19      Brand#15        Brand#23        Brand#11
        7       10      22
9       metallic
22      13      21      14      18      15      19
        33
1       80
4       1996-05-01
```

stream02

```
6       1996-01-01      0.05    24
17      Brand#25        SM PKG
14      1996-03-01
16      Brand#42        PROMO POLISHED 34       30
        22      25      48      1       9       45
19      Brand#13        Brand#51        Brand#51
        2       11      29
10      1994-02-01
9       light
2       31      BRASS   AMERICA
15      1993-06-01
8       CANADA  AMERICA MEDIUM BRUSHED STEEL
5       ASIA    1996-01-01
22      20      15      13      14      11      16
        25
12      RAIL    MAIL    1996-01-01
7       IRAN    CANADA
13      unusual accounts
18      315
1       88
4       1994-02-01
20      hot     1997-01-01      ARGENTINA
3       MACHINERY       1995-03-07
11      CANADA  0.0000001000
21      UNITED STATES
```

stream03

```
8       SAUDI ARABIA    MIDDLE EAST     MEDIUM
PLATED STEEL
5       MIDDLE EAST     1997-01-01
4       1996-09-01
6       1997-01-01      0.02    25
17      Brand#22        LG CASE
7       BRAZIL  SAUDI ARABIA
1       96
18      313
22      16      23      18      27      17      22
        10
14      1996-06-01
9       ivory
10      1994-11-01
15      1995-12-01
11      MOZAMBIQUE      0.0000001000
20      sandy   1996-01-01      MOROCCO
2       18      NICKEL  EUROPE
21      MOZAMBIQUE
19      Brand#25        Brand#44        Brand#55
        7       12      25
13      unusual accounts
16      Brand#22        SMALL BRUSHED  19       18
        17      32      28      5       35      42
12      AIR     MAIL    1996-01-01
3       BUILDING        1995-03-24
```

stream04

```
5       AFRICA  1997-01-01
21      INDIA
14      1996-09-01
19      Brand#22        Brand#22        Brand#54
        3       13      21
15      1993-09-01
```

```
17      Brand#24        LG JAR
12      SHIP    FOB     1996-01-01
6       1997-01-01      0.08    24
4       1994-06-01
9       goldenrod
8       JAPAN   ASIA    MEDIUM ANODIZED STEEL
16      Brand#52        ECONOMY BURNISHED       15
        10      42      14      41      37      2
        9
11      EGYPT   0.0000001000
2       6       COPPER  AMERICA
10      1993-09-01
18      314
1       104
13      unusual accounts
7       CHINA   JAPAN
22      16      20      28      13      11      10
        15
3       MACHINERY       1995-03-09
20      dark    1994-01-01      ETHIOPIA
```

stream05

```
21      ARGENTINA
15      1996-04-01
4       1997-01-01
6       1997-01-01      0.05    24
7       IRAN    EGYPT
16      Brand#42        STANDARD PLATED         21
        33      3       41      15      45      5
        19
19      Brand#24        Brand#15        Brand#43
        8       14      28
18      312
14      1996-12-01
22      22      28      24      11      19      10
        14
11      PERU    0.0000001000
13      unusual deposits
3       BUILDING        1995-03-26
1       112
2       44      STEEL   MIDDLE EAST
5       AMERICA 1997-01-01
8       EGYPT   MIDDLE EAST     SMALL POLISHED
STEEL
20      pale    1993-01-01      SAUDI ARABIA
12      FOB     MAIL    1996-01-01
17      Brand#21        LG PKG
10      1994-06-01
9       firebrick
```

stream06

```
10      1993-03-01
3       HOUSEHOLD       1995-03-11
15      1994-01-01
13      express deposits
6       1997-01-01      0.02    25
8       VIETNAM ASIA    SMALL BURNISHED STEEL
9       cyan
```

```
7       BRAZIL  VIETNAM
4       1994-10-01
11      ETHIOPIA        0.0000001000
22      23      13      20      16      21      14
        25
18      313
12      MAIL    FOB     1997-01-01
1       120
5       ASIA    1997-01-01
16      Brand#22        MEDIUM BRUSHED 21       17
        6       25      29      22      14      40
2       32      BRASS   AMERICA
14      1997-04-01
19      Brand#31        Brand#42        Brand#43
        3       15      25
20      black   1996-01-01      INDONESIA
17      Brand#23        MED CASE
21      CHINA
```

stream07

```
18      315
8       JORDAN  MIDDLE EAST     STANDARD BRUSHED
COPPER
20      light   1995-01-01      UNITED STATES
21      IRAN
2       20      NICKEL  MIDDLE EAST
4       1997-04-01
22      18      25      15      30      24      16
        32
17      Brand#35        MED JAR
1       67
11      CHINA   0.0000001000
9       chiffon
19      Brand#33        Brand#35        Brand#42
        9       16      21
3       BUILDING        1995-03-28
13      express deposits
5       EUROPE  1997-01-01
7       ROMANIA JORDAN
10      1993-12-01
16      Brand#53        PROMO ANODIZED 17       19
        13      50      30      37      41      29
6       1997-01-01      0.08    25
14      1997-07-01
15      1996-07-01
12      TRUCK   FOB     1997-01-01
```

Appendix F Benchmark Scripts

dbtables.sql

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

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



SELECT * FROM REGION;

SELECT COUNT(*) FROM NATION;

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



SELECT COUNT(*) FROM ORDERS;

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



SELECT COUNT(*) FROM PART;

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



SELECT COUNT(*) FROM PARTSUPP;

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

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

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

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

SELECT* FROM PARTSUPP
   WHERE PS_PARTKEY = 19763
   AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
      FROM PARTSUPP WHERE PS_PARTKEY =19763);



SELECT COUNT(*) FROM SUPPLIER;

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



DROP TABLE MINMAX;

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

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

INSERT INTO MINMAX
SELECT
'LINEITEM_NBR',MIN(L_LINENUMBER),MAX(L_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;
```

gen_seed.sh

```
#!/bin/ksh

SEED_FILE=$1

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

gtime.c

```
/* Copyright (c) Oracle Corporation 2001. All
Rights Reserved. */

/*

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

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

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

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

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

   NOTES
     <other useful comments, qualifications,
etc.>

   MODIFIED   (MM/DD/YY)
   mpoess      08/29/01 - Creation

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

# include <sys/time.h>

main ()
{

  struct timeval tv;

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

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

}

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

qexecpl.c

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

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

/*

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

   DESCRIPTION
     SQL Execution Engine, Oracle v8,  OCI
version

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

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

*/

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

#include "qexecpl.h"

/* Function Prototypes */

extern double gettime();

/* function prototypes from gen.c */

int get_statement();

/* Declare error handling functions */

void sql_error();

/* Other prototypes */

int define_output_variables();
```

```
void process_select_list();
void usage();
void SQLinit();
void SQLexec();
void SQLexit();
void *memalloc();
void print_header();
void print_rows();
int OFEN();
void remove_newline();

char logname[UNAME_LEN];  /* username/passwd
combo */
char *passwd;

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

double s_tr_start = 0.0;  /* statement start
time               */
double s_tr_end = 0.0;    /* statement end time
*/

/* For our purpose of timing, we will treat
comments as delimiters */
/* for queries.   Thus, we will collect query
timings whenever we  */
/* encounter a comment (of course not for the
first comment in a   */
/* file).
*/

int end_flag = 0;         /* flag to indicate
that we have reached */
                          /* the end of a query
*/

int stmt_cnt = 0;         /* Number of
statements processed.     */
int qry_cnt = 0;          /* Number of query
processed.          */

double product = 1.0;     /* cumulative product
of query times   */
int rows_ret = 0;         /* the number of rows
fetched          */
int num_sel_list = 0;     /* the number of
select list item      */

long num_to_fetch = -1;   /* Number of rows to
fetch. -1 means fetch all */

sltype slist[MAX_SEL_LIST];  /* Array for
describing Select List         */
dltype *dlist[MAX_SEL_LIST]; /* Array of ptrs
for Defining Select List   */

char stmt[SQL_LEN];       /* The SQL statement
or comment line.  */
char qn[3];               /* Number of the query
being executed  */
char qnp[3];              /* Number of the
previous query executed  */
char cmnt[5000];            /* Buffer to save
the comment.         */
#ifdef LINUX
FILE *qtemp;      /* fd for query template
*/
FILE *logfile;   /* log and report files
*/
FILE *rep;
#else
FILE *qtemp = stdin;      /* fd for query
template               */
FILE *logfile = stdout;   /* log and report
files                */
FILE *rep = stdout;
#endif
void *defbuf;             /* Buffer pointer for
ODEFIN           */
int deflen = 0;           /* Size of data type
for ODEFIN        */
int deftype = 1;          /* Oracle type number
for ODEFIN       */

int pfmem = PFMEMSIZE;    /* Memory to prefetch
rows             */

time_t tim;               /* To get wall clock
time              */

/* OCI handles */

OCIEnv *tpcenv = NULL;
OCIServer *tpcsrv = NULL;
OCIError *errhp = NULL;
OCISvcCtx *tpcsvc = NULL;
OCISession *tpcusr = NULL;
OCIStmt *curq = NULL;
OCIStmt *cur_dml = NULL;
OCIStmt *cur_ddl = NULL;
OCIParam *tpcpar = NULL;

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

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

void usage() {

  fprintf(stderr,"\nUsage: qexec
username/password [q<path name for query
template file>]\n");
  fprintf(stderr,"                 [l<path name
for log>] [r<path name for reports>]\n\n");
  fprintf(stderr,"Options:\n");
  fprintf(stderr,"q<path for query>         :
full path name for the query template file.\n");
  fprintf(stderr,"
(default is stdin)\n");
  fprintf(stderr,"l<path name for log>      :
full path name for log files\n");
  fprintf(stderr,"
(default is stdout)\n");
  fprintf(stderr,"r<path name for reports>  :
full path name for reports\n");
  fprintf(stderr,"
(default is stdout)\n");
  exit(-1);
}



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

void sql_error(errhp,status,type)
     OCIError *errhp;
     sword status;
     sword type;
{
  char msg[2048];
```

```
  ub4 errcode;
  ub4 msglen;
  int i,j;

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

  /* Rollback just in case */

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

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

  SQLexit();

  exit(1);
}

#ifdef LINUX
int main(argc,argv)
#else
void main(argc,argv)
#endif
     int argc;
     char *argv[];
{
  int i,pos,pos2;
  int retcode;      /* Return code for
get_statement */
#ifdef LINUX
  logfile=fopen("/dev/stdout","w");
  qtemp=fopen("/dev/stdin","rw");
  rep=fopen("/dev/stdout","w");
#endif
  /* Initialize some variables */

  if ((argc > 5) || (argc < 2)) {
    usage();
  }

  /* argv[1] -- User and Password for Database
*/

  strcpy(logname, argv[1]);

  /* Process  optional parameters */

  argc -= 1;
  argv += 1;

  while(--argc) {
    ++argv;
    switch(argv[0][0]) {
    case 'q':
      if ((qtemp = fopen(++(argv[0]),"r")) ==
NULL) {
       fprintf(stderr,"Unable to open file
'%s'\n", argv[0]);
       fprintf(stderr,"%s: %s\n", argv[0],
strerror(errno));
       exit(-1);
      }
      break;
    case 'r':
      if ((rep = fopen(++(argv[0]),"a")) ==
NULL) {
       fprintf(stderr,"Unable to open file
'%s'\n", argv[0]);
       fprintf(stderr,"%s: %s\n", argv[0],
strerror(errno));
       exit(-1);
      }
      break;
    case 'l':
      if ((logfile = fopen(++(argv[0]),"a")) ==
NULL) {
       fprintf(stderr,"Unable to open file
'%s'\n", argv[0]);
       fprintf(stderr,"%s: %s\n", argv[0],
strerror(errno));
       exit(-1);
      }
      break;
    default:
      fprintf(stderr,"Invalid Option: %c\n",
argv[0][0]);
      usage();
      break;
    }
  }

  /* Do some initialization and establish
connection with the database */

  SQLinit();

  /* May want to add some triggering mechanism
here */
```

```
  time(&tim);
  fprintf(logfile, "Begin Execution at %s\n\n",
ctime(&tim));
  fprintf(rep, "Begin Executing this Stream at
%s\n\n", ctime(&tim));
  /* Get the next statement and start processing
it */

  while ((retcode = get_statement()) > 0) {

    switch (retcode) {

      /* If this is a comment, skips it */
    case COMMENT:
      /*if (end_flag) {
        end_flag = 0;  */    /* reset query end
flag */
       /* save the comment so that we can print
it out later on  */
       /* strcpy(cmnt, stmt);
       break;
      } */
      if (stmt[3]== '@') {
         pos=4;
         strcpy(qnp,qn);
         while (stmt[pos] != ')') {
           pos++;
         }
         pos2=0;
         pos++;
         while (stmt[pos] != '.') {
           /*printf ("qn %d %c
\n",pos2,stmt[pos]);*/
           qn[pos2]=stmt[pos];
           pos2++;
           pos++;
         }
         qn[pos2] = 0;
         /* printf("found a new query:
%s\n",qn); */
      }
      /* save the comment so that we can print
it out later on  */
      strcat(cmnt, stmt);
      break;

      /* if this is a set_row_fetch command */
    case SET_FETCHROW:
      fprintf(logfile,"Setting the number of
rows to fetch to: %ld\n\n",
              num_to_fetch);
      break;

      /* if this is a SQL statement */
    case SQL_STMT:

      /* Executes the query */
      SQLexec();

      stmt_cnt++;
      qry_cnt++;
      fflush(rep);
      fflush(logfile);
      /*
      fprintf(logfile,"\nStatement Started at
%.2f\n", s_tr_start);
      fprintf(logfile,"Statement Ended at
%.2f\n", s_tr_end);

      fprintf(logfile,"Statement Processed in
%.2f seconds.\n",
                (s_tr_end - s_tr_start));
      fprintf(rep, "Query %s: Execution Time:
%.2f started %.2f ended %.2f\n",
                     qn,(s_tr_end -
s_tr_start)s_tr_start,s_tr_end);
      fflush(rep);
      fflush(logfile);*/
      break;

      /* Should never reach here */
    default:
      fprintf(stderr, "Invalid statement
type!!\n");
      SQLexit();
      break;
    }
  }

  /* Get Timing for the last query */

  tr_end = gettime();

  fprintf(logfile,"Query Processed in %.2f
seconds.\n\n",(tr_end - s_tr_start));

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

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

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

  time(&tim);
  fprintf(logfile,"\nEnded Executing this Stream
at %s\n", ctime(&tim));
  fprintf(logfile,"\nStream Started at %.2f\n",
tr_start);
  fprintf(logfile,"Stream Ended at %.2f\n",
tr_end);
  fprintf(logfile,"Stream Processed in %.2f
seconds\n\n",(tr_end - tr_start));

  fprintf(rep,"\nEnded Executing this Stream at
%s\n", ctime(&tim));
  fprintf(rep,"\nStream Started at %.2f\n",
tr_start);
  fprintf(rep,"Stream Ended at %.2f\n", tr_end);
  fprintf(rep,"Stream Processed in %.2f
seconds\n\n",
          (tr_end - tr_start));

  fprintf(logfile, "\nSQL statements processed:
%d\n", stmt_cnt);
  /*fprintf(logfile, "Queries processed: %d\n",
qry_cnt);*/

  fflush(rep);
  fflush(logfile);

  /* Close the query template file */

  fclose(qtemp);

  /*  Disconnect from ORACLE. */

  SQLexit();
  exit(0);
```

```
}


/* SQLinit(): Perform initialization tasks.
*/
/*            Logs on to Oracle, opens some
files and open a cursor for */
/*            later use.
*/

void SQLinit() {

  int i;

  /* preallocate MAX_PREALLOC members of the
dlist array                    */
  /* initializes others to NULL so that we can
determine who to free later */

  for (i=0; i<MAX_SEL_LIST; i++) {
    if (i < MAX_PREALLOC) {
      dlist[i] = (dltype *) memalloc
(sizeof(dltype));
      dlist[i]->defhdl = NULL;
/*      OCIhalloc(curq,&(dlist[i]-
>defhdl),OCI_HTYPE_DEFINE); */
    }
    else
      dlist[i] = NULL;
  }

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

  (void) OCIInitialize(OCI_DEFAULT,(dvoid
*)0,0,0,0);

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

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

  /* get username and password */

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

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


OCIaset(tpcsvc,OCI_HTYPE_SVCCTX,tpcsrv,0,OCI_ATT
R_SERVER,errhp);

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

OCIaset(tpcusr,OCI_HTYPE_SESSION,passwd,strlen(p
asswd),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_ATT
R_SESSION,errhp);

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

}



/* SQLexec()  Executes the SQL statement.
*/
/*            Parse the SQL statement.
*/
/*            If DDL or DML statements, execute
right away.             */
/*            Else describe and define select
list outputs,             */
/*                 execute and fetch results.
*/

void SQLexec()
{
  int i;
  ub2 stmttyp = OCI_STMT_SELECT;     /* default
is a SELECT statement */

  /* Clause 5.3.6.2: QI(i,s) is the time between
the first character */
  /*                 of this query text is
submitted and the first   */
  /*                 character of the next query
text is submitted.  */

  if (qry_cnt) {
     time(&tim);
     s_tr_end = gettime();
     fprintf(logfile,"Query Processed in %.2f
seconds.\n\n",
      (s_tr_end - s_tr_start));

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

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

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

```
                    qnp,(s_tr_end -
s_tr_start),s_tr_start,s_tr_end);

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

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

  s_tr_start = gettime();

  /* prepare the statement */

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

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

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

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


OCIaget(curq,OCI_HTYPE_STMT,&stmttyp,NULL,OCI_AT
TR_STMT_TYPE,errhp);

  if (stmttyp != OCI_STMT_SELECT) {
    OCIsexec(tpcsvc,curq,errhp,1);
    return;
  }

  /* otherwise, this is a select statement */
  /* Describe and define output variables  */

  /* first let's execute it to get the select-
list definition */

  OCIaset(curq, OCI_HTYPE_STMT, &pfmem, 0,
OCI_ATTR_PREFETCH_MEMORY, errhp);

  OCIsexec(tpcsvc,curq,errhp,0);

  num_sel_list = define_output_variables();

  /* Executes the query and fetches the rows */

  (void) process_select_list(num_sel_list);

  /* Need to get the number of rows fetched
first   */
  /* since the following statments will screw it
up */


OCIaget(curq,OCI_HTYPE_STMT,&rows_ret,NULL,OCI_A
TTR_ROW_COUNT,errhp);
```

```
  /* To control memory usage, let's free up the
extra dlist entries */
  /* that we have allocated.
*/

  i=MAX_PREALLOC;
  while(dlist[i] != NULL) {
    free(dlist[i]);
    dlist[i++] = NULL;
  }

  /* reset set_fetchrows */

  num_to_fetch = -1;

}


void SQLexit() {

  int i;

  OCILogoff(tpcsvc,errhp);
  OCIhfree(tpcenv,OCI_HTYPE_STMT);
  OCIhfree(tpcsvc,OCI_HTYPE_SVCCTX);
  OCIhfree(tpcsrv,OCI_HTYPE_SERVER);
  OCIhfree(tpcusr,OCI_HTYPE_SESSION);

  /* free all memory */

  for (i=0; i<MAX_SEL_LIST; i++) {
    if (dlist[i] != NULL) {
      free(dlist[i]);
    }
  }

  /* Flush all output */

  fflush(rep);
  fflush(logfile);

}


/* define_output_variables(): Describe and
define select-list items for */
/*                            a query statement.
*/
/*                            Returns the number
of select-list items   */
/*                            for this query.
*/

int define_output_variables()
{

  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_A
TTR_ROW_COUNT,errhp);

    print_rows(num,rows_ret);

    /* To avoid 1022 from OFEN */
    /* More rows to fetch...   */

    if (stats != OCI_NO_DATA) {
      if (num_to_fetch == -1) {
        while ((stats =
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,N
ULL,
                  OCI_ATTR_ROW_COUNT,errhp);
      print_rows(num,(num_so_far-rows_ret));
      rows_ret = num_so_far;
    } else {
      ntf -= MAX_ARRAY;

      while ((stats = 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,N
ULL,
                  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_A
TTR_ROW_COUNT,errhp);
    print_rows(num,rows_ret);
  }

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

}


int get_statement()
{

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

  /* Reset statement buffer */

  stmt[0] = '\0';

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

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

    /* remove blanks */

    str = line;

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

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

    strcat(stmt, str);

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

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

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


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

void *memalloc(size)
     int size;
{

  void *tmp;

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



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

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

  fprintf(logfile, "\n");

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

    /* extract the column name */

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

    /* format the output a little */
```

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

    /* do a little bit of formatting */

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

  fprintf(logfile,"\n");
}


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

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

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

    len = 0;

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

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

      /* do a little bit of formatting */

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

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


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

void remove_newline(str)
     char *str;
{

  char *p;

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

qexecpl.h

```
/*
 * $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_DEFAULT)) != OCI_SUCCESS) \
       sql_error(errh,status,1); \
    else \
       DISCARD 0


#define ISOTXT "alter session set
isolation_level = serializable"
#define PDMLTXT "alter session force parallel
dml parallel (degree 84)"
#define PDDLTXT "alter session force parallel
ddl parallel (degree 84)"

#endif /* QSTREAMPL_H */
```

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=/dbms/oracle9i/kit/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

mv
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log.p
reAudit.$RUN_ID
```

```
touch
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log

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

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

echo "Start: dbtables.sql and count.sql" >>
$SCRIPT_LOG_FILE
$sqlplus ${DATABASE_USER}
@$KIT_DIR/audit/dbtables > ${RDB_TABLES} 2>&1
$sqlplus ${DATABASE_USER}
@$KIT_DIR/audit/firstten > ${FIRST_TEN} 2>&1
echo "End: dbtables.sql and count.sql `date`" >>
$SCRIPT_LOG_FILE

tshut >>  $SCRIPT_LOG_FILE
tstart >>  $SCRIPT_LOG_FILE
ckpnt.sh
runTPCHpt ${SCALE_FACTOR} 1 ${RUN_ID}

tshut >>  $SCRIPT_LOG_FILE
tstart >>  $SCRIPT_LOG_FILE
ckpnt.sh
runTPCHpt ${SCALE_FACTOR} 2 ${RUN_ID}

sleep 600
tshut >> $SCRIPT_LOG_FILE

cp
$ORACLE_HOME/rdbms/log/alert_${ORACLE_SID}.log
$OUT_DIR

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

runTPCHpt

```
#!/bin/ksh
. $KIT_DIR/env
#set -x
#ECHO=/bin/echo
SCRIPT_DIR=${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_STREAMS-
1"

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

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

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

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

# Execute UF1

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

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

UF1_END=`${GTIME}`
E1DATE=`date`

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

echo "End UF1 $UF1_END, ${E1DATE}" >>
$SCRIPT_LOG_FILE
echo UF1: Execution Time: $UF1_TIME >>
$SCRIPT_LOG_FILE
echo >> $SCRIPT_LOG_FILE

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

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

# Execute UF2

UF2_START=`${GTIME}`
E2DATE=`date`

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

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

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


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

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

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

i=$START_SET
PSEED=`cat $SEED_FILE`

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

  PSEED=`expr $PSEED + 1`
```

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

  i=`expr $i + 1`
done

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

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

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

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

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

(${KIT_DIR}/audit/runTPCHus $RUN_ID
$START_SET_UPDATE $STOP_SET_UPDATE ${SF} $PARA
>> $SCRIPT_LOG_FILE 2>&1 &)

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

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

i=$START_SET
while [ $i -le $STOP_SET ]; do
        TPCD_LOG_FILE=${TPCD_LOG}/m${PARA}s${i}
  ${KIT_DIR}/audit/abridge.pl ${TPCD_LOG_FILE}
        i=`expr $i + 1`
done
PIDS=`ps -fu oracle | grep scnt.sh | grep -v
grep | awk '{print $2}'`
kill -9 $PIDS
#calculate the metric
#analyze_streams.pl -f p -n $RUN_ID >
${TPCD_RPT}/tpch_metric.${RUN_ID}.${HID}.rpt
```

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}

LOGPATH=.
PASSWD=${DATABASE_USER}

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

# perform the update function 1

START=`$GTIME`

# first create the temp tables

sqlplus /NOLOG << !

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

drop directory data_dir;
create directory data_dir as
'/dbms/oracle9i/kit/update/update_sets';

drop table temp_l_et;
create table temp_l_et(
    l_shipdate          date ,
    l_orderkey          number ,
    l_discount          number ,
    l_extendedprice     number ,
    l_suppkey           number ,
    l_quantity          number ,
    l_returnflag        char(1) ,
    l_partkey           number ,
    l_linestatus        char(1) ,
    l_tax               number ,
    l_commitdate        date ,
    l_receiptdate       date ,
    l_shipmode          char(10) ,
    l_linenumber        number ,
    l_shipinstruct      char(25) ,
    l_comment           varchar(44)
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
                records delimited by newline
                badfile 'l_et.${SETNUM}.bad'
                logfile 'l_et.${SETNUM}.log'
                fields terminated by '|'
                missing field values are null
        )
        location (
        'lineitem.tbl.u${SETNUM}'))
reject limit unlimited;

drop table temp_o_et;
create table temp_o_et(
    o_orderdate         date ,
    o_orderkey          number ,
    o_custkey           number ,
```

```
    o_orderpriority      char(15) ,
    o_shippriority       number ,
    o_clerk              char(15) ,
    o_orderstatus        char(1) ,
    o_totalprice         number ,
    o_comment            varchar(79)
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
                records delimited by newline
                badfile 'o_et.${SETNUM}.bad'
                logfile 'o_et.${SETNUM}.log'
                fields terminated by '|'
                missing field values are null
        )
        location (
        'orders.tbl.u${SETNUM}'))
reject limit unlimited;

alter table temp_l_et parallel  ${PAR_HINT};
alter table temp_o_et parallel  ${PAR_HINT};

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

commit;

insert into orders (select * from temp_o_et);

insert into lineitem (select * from temp_l_et);

commit;

drop table temp_l_et;
drop table temp_o_et;

exit;
!

END=`$GTIME`

# Done

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

runuf2.sh

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

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

SETNUM=$1

i=1
PID=""

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

sqlplus /NOLOG << !

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

drop directory data_dir;
create directory data_dir as
'/dbms/oracle9i/kit/update/update_sets';

drop table temp_okey_et;
drop table temp_okey;

create table temp_okey_et(
    t_orderkey           number
)
organization external (
type ORACLE_LOADER
default directory data_dir
access parameters
(
                records delimited by newline
                badfile 'okey.${SETNUM}.bad'
                logfile 'okey.${SETNUM}.log'
                fields terminated by '|'
                missing field values are null
        )
```

```
        location (
        'delete.${SETNUM}'))
reject limit unlimited;

alter table temp_okey_et parallel ${PAR_HINT};

create table temp_okey parallel  ${PAR_HINT}
nologging as select * from temp_okey_et;

create unique index i_temp_okey on temp_okey
(t_orderkey) parallel ${PAR_HINT} nologging
compute statistics;
alter index i_temp_okey parallel;
analyze table temp_okey estimate statistics
sample 2 percent;

alter session  force parallel dml parallel
${PAR_HINT};
alter session set isolation_level=serializable;
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 ""
```

audit_stream.sh

```
#!/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

RUN_ID_FILE=${KIT_DIR}/r_id

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

TPCD_BIN=${KIT_DIR}/audit/bin
TPCD_LOG=${KIT_DIR}/audit/log
TPCD_RPT=${KIT_DIR}/audit/rpt

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

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

# The defaults

USER="tpch/tpch"
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 "2" ]
then
  usage
  exit 1
fi
```

```
SF=$1
PARA=$2

if [ $PARA = first ]
then
  UF_SET=1
  START_SET=1
  STOP_SET=7
  START_SET_UPDATE=2
  STOP_SET_UPDATE=8
else
  UF_SET=9
  START_SET=1
  STOP_SET=7
  START_SET_UPDATE=10
  STOP_SET_UPDATE=16
fi

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

echo "TPC-H Test Run `date`" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt
echo "RUNID is $RUN_ID" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt
echo ""  >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

TPCD_LOG_FILE=${TPCD_LOG}/query.${RUN_ID}.log
TPCD_RPT_FILE=${TPCD_RPT}/query.${RUN_ID}.rpt
QRY_FILE=${QRY_DIR}/qtemp.${RUN_ID}.${SF}

echo "Generates query template file with given
seed for stream 0" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

${QGEN} -c -r `cat $SEED_FILE` -p 0 -s ${SF} -l
qparameter0 > ${QRY_FILE}

echo "Done generating query template file with
given seed for stream 0" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

START=`$GTIME`
echo "TPC-H Power Test Execution Starts $START,
`date`"
echo "TPC-H Power Test Execution Starts $START,
`date`" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt
echo "" >> ${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

# Execute UF1

SDATE=`date`
UF1_START=`$GTIME`
echo "Start UF1  ${SDATE}"
echo "Start UF1  ${SDATE}" >>
${OUT}/uf1.${RUN_ID}.${HID} 2>&1
echo "Start UF1  $UF1_START, `date`" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

${ECHO} ${UPD_SPT}/runuf1.sh ${UF_SET} >>
${OUT}/uf1.${RUN_ID}.${HID} 2>&1

# Execute Query Stream

UF1_END=`${GTIME}`
E1DATE=`date`

echo "End UF1, Start Query Stream ${E1DATE}"
echo "End UF1, Start Query Stream ${E1DATE}" >>
${OUT}/uf1.${RUN_ID}.${HID} 2>&1
echo "" >>  ${OUT}/uf1.${RUN_ID}.${HID} 2>&1
echo "End UF1, Start Query Stream $UF1_END,
${E1DATE}" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

${QPROG} tpch/tpch q${QRY_FILE}
l${TPCD_LOG_FILE} r${TPCD_RPT_FILE}  >>
${OUT}/qs.${RUN_ID}.${HID} 2>&1

# Execute UF2

UF2_START=`${GTIME}`
E2DATE=`date`

echo "End Query Stream, Start UF2 $UF2_START,
${E2DATE}" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt
echo "End Query Stream, Start UF2 ${E2DATE}"
echo "End Query Stream, Start UF2 ${E2DATE}" >>
${OUT}/uf2.${RUN_ID}.${HID} 2>&1

${ECHO} ${UPD_SPT}/runuf2.sh ${UF_SET} >>
${OUT}/uf2.${RUN_ID}.${HID} 2>&1

UF2_END=`${GTIME}`
END=`${GTIME}`
EDATE=`date`

echo "End UF2 $UF2_END, $EDATE" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt
echo "END UF2 ${EDATE}"
echo "End UF2\t${EDATE}" >>
${OUT}/uf2.${RUN_ID}.${HID} 2>&1

echo "TPC-H Power Test Execution Ends $EDATE" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt
echo "TPC-H Power Test Execution Ends $EDATE"
MEA_INT=`echo $END - $START | bc`
echo "Elapsed Time for TPC-H Power Test is
$MEA_INT"

UF1_TIME=`echo $UF1_END - $UF1_START | bc`
echo "Elapsed Time for Update Function is
$UF1_TIME"
echo "-- UF1" >> ${TPCD_RPT_FILE}
echo "$UF1_TIME" >> ${TPCD_RPT_FILE}

UF2_TIME=`echo $UF2_END - $UF2_START | bc`
echo "Elapsed Time for Update Function is
$UF2_TIME"
echo "-- UF2" >> ${TPCD_RPT_FILE}
echo "$UF2_TIME" >> ${TPCD_RPT_FILE}

echo ""
echo "----------------------------------------
-----"
echo ""

#${SRC_DIR}/metric ${SF} < ${TPCD_RPT_FILE} \
        >> ${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

#THRUPUT=`expr 22 \* 3600 \* ${SF}`
#THRUPUT=`echo "scale=2\n${THRUPUT}/${MEA_INT}"
| bc`
#echo "Real Throughput Metric is: $THRUPUT"
```

```
#echo "Real Throughput Metric is: $THRUPUT" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt


echo "TPC-H Throughput Test Execution Starts
`date`" >> ${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt
echo "TPC-H Throughput Test Execution Starts
`date`"


i=$START_SET
PSEED=`cat $SEED_FILE`

while [ $i -le $STOP_SET ]; do

TPCD_LOG_FILE=${TPCD_LOG}/query.${RUN_ID}_${i}.l
og

TPCD_RPT_FILE=${TPCD_RPT}/query.${RUN_ID}_${i}.r
pt
  QRY_FILE=${QRY_DIR}/qtemp.${RUN_ID}_${i}.${SF}

  echo "Generate query files for STREAM ${i}" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

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

  i=`expr $i + 1`
done

echo "Done creating multiple streams" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

echo
I1DATE=`date`
M_START=`${GTIME}`
echo "Start Throughput Test $I1DATE"
echo "Start Throughput Test $M_START, $I1DATE"
>> ${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

rm -f /tmp/th_pipe1
mknod /tmp/th_pipe1 p
rm -f /tmp/th_pipe2
mknod /tmp/th_pipe2 p

i=$START_SET

while [ $i -le $STOP_SET ]; do
M_SDATE=`date`
M_STIME=`${GTIME}`
  echo "Start Query Stream $i $M_STIME,
${M_SDATE}" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

TPCD_LOG_FILE=${TPCD_LOG}/query.${RUN_ID}_${i}.l
og

TPCD_RPT_FILE=${TPCD_RPT}/query.${RUN_ID}_${i}.r
pt
  QRY_FILE=${QRY_DIR}/qtemp.${RUN_ID}_${i}.${SF}
  ${QPROG} tpch/tpch q${QRY_FILE}
l${TPCD_LOG_FILE} >>
${OUT}/qs.${RUN_ID}_${i}.${HID} 2>&1 &
  i=`expr $i + 1`
done
sleep.sh &
(${KIT_DIR}/audit/update_stream.sh ${RUN_ID}
$START_SET_UPDATE $STOP_SET_UPDATE ${SF} >>
${OUT}/us.${RUN_ID}_1.${HID} 2>&1 &)
wait
print > /tmp/th_pipe1
read < /tmp/th_pipe2

M_EDATE=`date`
M_END=`$GTIME`
echo "End Throughput Test $M_END"
echo "End Throughput Test $M_END, $M_EDATE" >>
${TPCD_RPT}/tpch.${RUN_ID}.${HID}.rpt

#calculate the metric
analyze_streams.pl -f p -n $RUN_ID >
${TPCD_RPT}/tpch_metric.${RUN_ID}.${HID}.rpt
```

tstart

```
#!/bin/ksh

if [ $# -ne 0 ]; then
  INUM=$1
else
  INUM=1
fi

echo
echo "Starting instance $INUM"

if [ -f $ORACLE_HOME/dbs/init_"$ORACLE_SID".ora
]; then
  PFILE=$ORACLE_HOME/dbs/init_"$ORACLE_SID".ora
else
  if [ -f $ORACLE_HOME/work/t_init$INUM.ora ];
then
    if [ $INUM -ne 1 ]; then
      export ORACLE_SID="$ORACLE_SID"$INUM
    fi
    PFILE=$ORACLE_HOME/work/t_init$INUM.ora
  else
    if [ -f $ORACLE_HOME/dbs/tkinit.ora ]; then
      PFILE=$ORACLE_HOME/dbs/tkinit.ora
    else
      echo "What pfile should I use ???"
      exit 1
    fi
  fi
fi

sqlplus /NOLOG << !
connect / as sysdba
startup pfile=$PFILE
exit
!
/Lvm/set_queue
```

tshut

```
#!/bin/ksh

if [ "$2" != "" -a "$2" != "1" ]; then
  INUM=$2
  if [ -f $ORACLE_HOME/work/t_init$INUM.ora ];
then
    export ORACLE_SID="$ORACLE_SID"$INUM
  fi
fi
```

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

set_queue

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c33t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c33t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c33t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c33t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c33t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c33t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c33t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c33t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c25t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c25t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c25t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c25t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c25t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c25t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c25t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c25t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c35t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c35t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c35t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c35t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c35t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c35t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c35t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c35t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c51t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c51t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c51t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c51t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c51t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c51t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c51t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c51t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c39t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c39t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c39t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c39t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c39t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c39t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c39t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c39t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c29t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c29t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c29t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c29t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c29t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c29t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c29t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c29t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c43t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c43t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c43t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c43t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c43t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c43t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c43t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c43t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c59t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c59t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c59t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c59t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c59t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c59t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c59t0d6
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c59t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c34t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c34t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c34t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c34t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c34t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c34t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c34t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c34t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c27t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c27t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c27t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c27t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c27t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c27t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c27t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c27t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c36t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c36t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c36t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c36t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c36t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c36t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c36t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c36t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c52t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c52t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c52t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c52t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c52t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c52t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c52t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c52t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c21t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c21t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c21t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c21t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c21t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c21t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c21t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c21t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c44t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c44t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c44t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c44t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c44t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c44t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c44t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c44t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c60t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c60t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c60t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c60t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c60t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c60t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c60t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c60t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c48t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c48t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c48t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c48t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c48t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c48t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c48t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c48t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c17t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c17t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c17t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c17t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c17t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c17t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c17t0d6
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c17t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c37t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c37t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c37t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c37t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c37t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c37t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c37t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c37t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c53t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c53t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c53t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c53t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c53t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c53t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c53t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c53t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c56t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c56t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c56t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c56t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c56t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c56t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c56t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c56t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c45t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c45t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c45t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c45t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c45t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c45t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c45t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c45t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c61t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c61t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c61t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c61t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c61t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c61t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c61t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c61t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c19t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c19t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c19t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c19t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c19t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c19t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c19t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c19t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c66t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c66t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c66t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c66t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c66t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c66t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c66t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c66t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c1t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c1t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c1t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c1t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c1t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c1t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c1t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c1t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c13t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c13t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c13t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c13t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c13t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c13t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c13t0d6
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c13t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c40t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c40t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c40t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c40t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c40t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c40t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c40t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c40t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c57t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c57t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c57t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c57t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c57t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c57t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c57t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c57t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c5t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c5t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c5t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c5t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c5t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c5t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c5t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c5t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c64t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c64t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c64t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c64t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c64t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c64t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c64t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c64t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c49t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c49t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c49t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c49t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c49t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c49t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c49t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c49t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c3t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c3t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c3t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c3t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c3t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c3t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c3t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c3t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c8t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c8t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c8t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c8t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c8t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c8t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c8t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c8t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c42t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c42t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c42t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c42t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c42t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c42t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c42t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c42t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c58t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c58t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c58t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c58t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c58t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c58t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c58t0d6
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c58t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c6t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c6t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c6t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c6t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c6t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c6t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c6t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c6t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c117t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c117t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c117t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c117t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c117t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c117t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c117t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c117t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c150t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c150t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c150t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c150t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c150t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c150t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c150t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c150t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c154t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c154t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c154t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c154t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c154t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c154t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c154t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c154t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c159t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c159t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c159t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c159t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c159t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c159t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c159t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c159t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c129t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c129t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c129t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c129t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c129t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c129t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c129t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c129t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c144t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c144t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c144t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c144t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c144t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c144t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c144t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c144t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c137t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c137t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c137t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c137t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c137t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c137t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c137t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c137t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c31t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c31t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c31t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c31t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c31t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c31t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c31t0d6
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c31t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c178t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c178t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c178t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c178t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c178t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c178t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c178t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c178t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c151t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c151t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c151t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c151t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c151t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c151t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c151t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c151t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c155t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c155t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c155t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c155t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c155t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c155t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c155t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c155t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c161t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c161t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c161t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c161t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c161t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c161t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c161t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c161t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c163t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c163t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c163t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c163t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c163t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c163t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c163t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c163t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c167t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c167t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c167t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c167t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c167t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c167t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c167t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c167t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c169t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c169t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c169t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c169t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c169t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c169t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c169t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c169t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c173t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c173t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c173t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c173t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c173t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c173t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c173t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c173t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c180t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c180t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c180t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c180t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c180t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c180t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c180t0d6
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c180t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c152t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c152t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c152t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c152t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c152t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c152t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c152t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c152t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c157t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c157t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c157t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c157t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c157t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c157t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c157t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c157t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c86t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c86t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c86t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c86t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c86t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c86t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c86t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c86t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c165t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c165t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c165t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c165t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c165t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c165t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c165t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c165t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c111t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c111t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c111t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c111t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c111t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c111t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c111t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c111t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c148t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c148t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c148t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c148t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c148t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c148t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c148t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c148t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c175t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c175t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c175t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c175t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c175t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c175t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c175t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c175t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c10t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c10t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c10t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c10t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c10t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c10t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c10t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c10t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c81t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c81t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c81t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c81t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c81t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c81t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c81t0d6
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c81t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c125t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c125t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c125t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c125t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c125t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c125t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c125t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c125t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c87t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c87t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c87t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c87t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c87t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c87t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c87t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c87t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c90t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c90t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c90t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c90t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c90t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c90t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c90t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c90t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c168t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c168t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c168t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c168t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c168t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c168t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c168t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c168t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c171t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c171t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c171t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c171t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c171t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c171t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c171t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c171t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c177t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c177t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c177t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c177t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c177t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c177t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c177t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c177t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c11t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c11t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c11t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c11t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c11t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c11t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c11t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c11t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c98t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c98t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c98t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c98t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c98t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c98t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c98t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c98t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c99t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c99t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c99t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c99t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c99t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c99t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c99t0d6
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c99t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c84t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c84t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c84t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c84t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c84t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c84t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c84t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c84t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c101t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c101t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c101t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c101t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c101t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c101t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c101t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c101t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c83t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c83t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c83t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c83t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c83t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c83t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c83t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c83t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c102t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c102t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c102t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c102t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c102t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c102t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c102t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c102t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c82t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c82t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c82t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c82t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c82t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c82t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c82t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c82t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c142t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c142t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c142t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c142t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c142t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c142t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c142t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c142t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c91t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c91t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c91t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c91t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c91t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c91t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c91t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c91t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c94t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c94t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c94t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c94t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c94t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c94t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c94t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c94t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c95t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c95t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c95t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c95t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c95t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c95t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c95t0d6
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c95t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c146t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c146t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c146t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c146t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c146t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c146t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c146t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c146t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c96t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c96t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c96t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c96t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c96t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c96t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c96t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c96t0d7
```

```
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c23t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c23t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c23t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c23t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c23t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c23t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c23t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c23t0d7
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c131t0d0
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c131t0d1
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c131t0d2
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c131t0d3
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c131t0d4
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c131t0d5
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c131t0d6
/usr/sbin/scsictl -m queue_depth=32
/dev/rdsk/c131t0d7
```

Appendix G Price Quotes

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

Juergen Mueller
HP
Cupertino, CA 95014
October 28, 2002

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

HP 9000 Superdome Enterprise Server

TPC-H Rev 2

Report Date: October 28, 2002

Description	Part Number	Source	Reference Price	Qty	Extended Price	3 yr. Maint. Price
Server Hardware						
Super Dome left chassis (support includes hardware, servcies, and support)	A5201A, Opt. 429	1	205,840	1	205,840	268,626
Super Dome right chassis	A5202A, Opt. 429	1	218,435	1	218,435	
Memory module - 2 GB	A5198A, Opt. 0D1	1	14,000	64	896,000	
I/O enclosures	A4856A, Opt. 0D1	1	14,805	8	118,440	
PDCA Redundant Power Source	A5800A, Opt. 0D1	1	578	2	1,156	
Cell Board with 4 PA-8700 875MHz Processors	A6862A	1	10,080	16	161,280	
ICOD right to use processor	A6885A, Opt. 104	1	23,000	64	1,472,000	403,392
Super Dome PCI Core I/O card	A6855A, Opt. 0D1	1	1,045	1	1,045	
PCI Dual FWD SCSI-2 Card	A5159A, Opt 0D1	1	1,245	2	2,490	
PCI 1000BT Lan Adapter	A4926A, Opt. 0D1	1	2,135	1	2,135	
PCI Fibre Channel 2X	A5158A, Opt 0D1	1	2,240	84	188,160	
RackInstallation Kit	A5170A. Opt 0D1	1	410	1	410	
DVD-ROM (includes SCSI-2 card, cables, enclosures)	C7499A	1	3,503	1	3,503	
HP9000 A500 Support Management Station (includes memory,CPU,lan card,disk,OS, etc)	A5570B	1	11,780	1	11,780	
hp surestore disk system 2100	A5675A	1	700	1	700	
1-18GB FWD disk module	A6537A	1	525	3	1,575	
5m SCSI-2 Cable	C2978B	1	99	1	99	
				Subtotal	**3,285,048**	**672,018**
Server Software						
HP-UX 11i LTU	B9088AC, Opt AAF	1		1	0	
HP-UX 11i Media	B3920EA, Opt UM9	1		1	0	
HP-UX 11i	B3920EA, Opt OD1	1		1	0	
HP-UX 11.i Factory Integration	B3920EA, Opt AAF	1	520	1	520	
				Subtotal	**520**	**0**
Storage						
Surestore VA 7100 w/dual controllers 512MB cache	A6262A	1	44,250	84	3,717,000	308,616
18GB 15K RPM FC HDD	A6191A, Opt. 0D1	1	914	1260	1,151,640	
16 meter Fibre Optic Cable	A3531AR	1	170	84	14,280	
HP9000 Std. Rack System E41	A4902A	1	1,910	7	13,370	
Modular Power Dist.	A5137AZ	1	145	28	4,060	
200-240 Volts	A5137AZ, Opt AW4	1	94	28	2,632	
				Subtotal	**4,902,982**	**308,616**
				Total	**8,188,550**	**980,634**
Large Configuration Discount and Support Prepayment*					(4,186,820)	(377,997)
				Grand Total	**4,001,730**	**602,637**
				3-yr Cost of Ownership:	**$**	**4,604,367**

All the components in the price list are currently available. Maintenance support price is for 24 hours, 7 days with 4 hour response time.

-----Original Message-----

From: MaryBeth Pierantoni [mailto:mary.beth.pierantoni@oracle.com]

Sent: Monday, October 28, 2002 2:47 PM

To: lucille_boushey@hp.com

Cc: mary.beth.pierantoni@oracle.com

Subject: TPC-H Pricing

A.1 Product	Price	Quantity	Extended Price
Oracle9i Database Enterprise Edition Release 2, Named User Plus for 3 years	**$10,000**	**64**	**$640,000**
Partitioning, Named User Plus for 3 years	**$2,500**	**64**	**$160,000**
Oracle Database Server Support Package for 3 years	**$6,000**	**1**	**$6,000**
Oracle Mandatory E-Business Discount (license and Support)*			**<$161,200>**

Oracle pricing contact: MaryBeth Pierantoni, mary.beth.pierantoni@oracle.com, 650-506-2118