Sun microsystems

THE NETWORK IS THE COMPUTER™

TPC Benchmark™ H
Full Disclosure Report

Sun Microsystems Sun Fire™ V880 Server
Using Sybase IQ 12.5 Single Application Server

Submitted for Review
Report Date: report date: may 9, 2005

TPC Benchmark H Full Disclosure Report

First Printing

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

Sun Fire™ V880 Server with Sybase IQ 12.5 Single Application Server

TPC-H Rev. 2.0

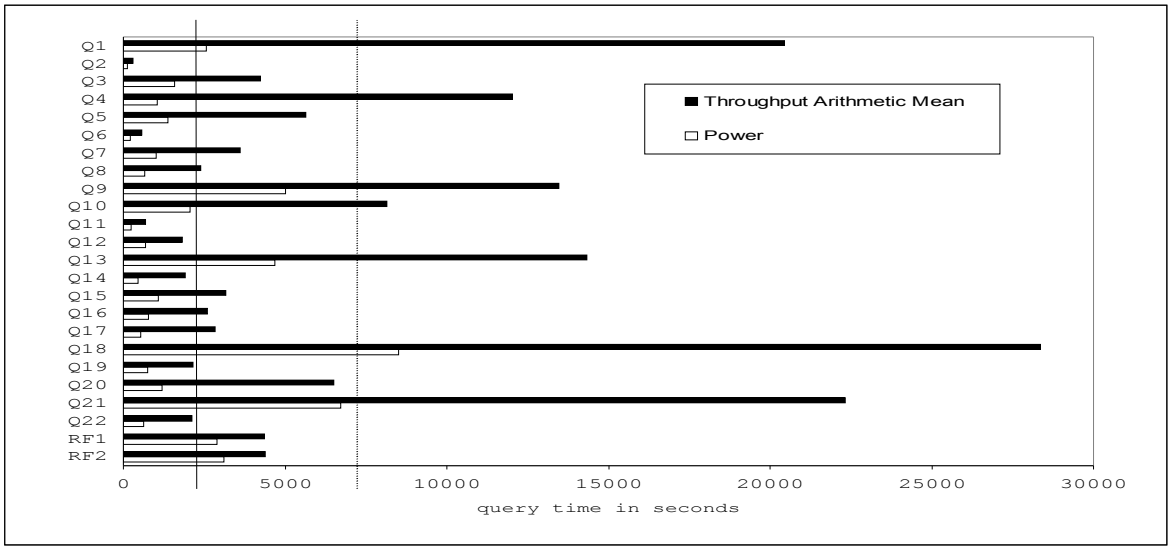
report date: may 9, 2005
Revision Date: july 25, 2005

Total System Cost	Composite Query per Hour Metric	Price/Performance
$ 171,455.20	**3247.4** QphH@1000GB	**$53** per QphH@1000GB

Database Size	Database Manager	Operating System	Other Software	Availability Date
1000GB	**Sybase IQ 12.5 Single Application Server**	**Solaris10**	**Solaris Volume Manager**	**may 9, 2005**

1163.9 = power geometric mean

7254.5 = throughput arithmetic mean



Database Load Time = 24 hrs 49 mins Load Includes Backup: N Total Storage/Database Size=2.04

RAID (Base tables): RAID 5 RAID (Base tables and auxiliary data structures): RAID 5 RAID (All): N

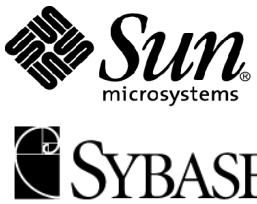
System Configuration:

SunFire V880 Server with
- 8 UltraSPARC III Cu 1.2GHz processors
- 32 GB memory
- 6 x 73GB internal disks

2 x SS3310 SCSI disk array with
- 12 x 73GB disks

Total Storage: 2039.6 GB

(in this calculation 1 GB is defined as 1024 * 1024 * 1024)



Sun Fire™ V880 Server with Sybase IQ 12.5 Single Application Server

TPC-H Rev. 2.0

report date: may 9, 2005
Revision Date: july 25, 2005

3 Year Gold Warranty Upgrade – V880 Server (includes internal disks)		1	15,507.40	1		15,507.40
Eakins Open Systsem Discount					-15,800.00	
Server HardwareSubtotal					94,195.00	15,507.40
Storage						
SS3310	XTA3310R01A0R876	1	14,995.00	2	29,990.00	
3 Year Gold Warranty Upgrade – ss3310 (includes all disks)	W9D-SE3310-3G	1	3,703.00	2		7,406.00
SCSI cable	x1138A	1	95.00	4	380.00	
SCSI HBA			800.00	2	1,600.00	
Eakins Open Systsem Discount					-9,178.40	
Storage Subtotal					22,791.60	7,406.00
Server Software						
Sybase IQ-M Single App Svr per cpr	11467	2	2,595.00	8	20,760.00	
Sybase IQ 3 Years Extended Support 24 x 7 per cpu	98477	2	1,557	8		12,456.00
Sybase Discount					-1,660.80	
Server Software Subtotal					19,099.20	12,456.00
			Total		136,085.80	35,369.40
			3 Yr. Cost		171,455.20	
			QphH@ 1000GB		3,247.40	
			$/QphH@ 100GB		$53	

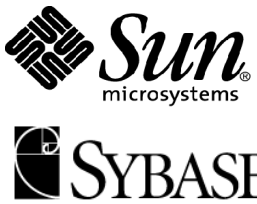
Service for all Sun products is from Sun Microsystems, Inc.

Service for Sybase products is from Sybase Inc.

Notes (Source):

1. Eakins Open Systems Inc.
2. Sybase Inc.
3. PriceQuotes provided in Appendix G
4. The pricing of the Sybase IQ Single Application Server was found to be an "insignificant deviation" of Clause 0.2 by the General Council. This was corrected by increasing the visibility of Sybase IQ Single Application Server on the Web.

Audited by: Brad Askins, InfoSizing, Inc. (www.sizing.com)



Sun Fire™ V880 Server with Sybase IQ 12.5 Single Application Server

TPC-H Rev. 2.0

report date: may 9, 2005
Revision Date: july 25, 2005

Numerical Quantities

Measurement Results:

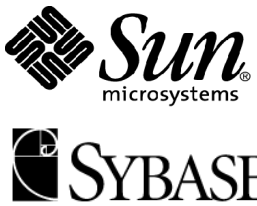
Database Scale Factor	= 1000GB
Total Data Storage / Database Size	= 2.04
Start of database load time	= 2004-07-28 22:14:48
End of database load time	= 2004-07-29 23:03:26
Database Load Time	= 24 hrs 49 mins
Query Streams for Throughput Test	= 7
TPC-H Power	= 3,093.1
TPC-H Throughput	= 3409.3
TPC-H Composite Query-per-Hour Rating (QphH@1000GB)	= 3247.4
Total System Price Over 3 Years	= $ 171,455.20
TPC-H Price/Performance Metric ($/QphH@1000GB)	= $53

Measurement Intervals:

Measurement Interval in Throughput Test (Ts)	= 162,616 seconds

Duration of Stream Execution:

Stream ID	Seed	Start Date	Start Time	End Date	End Time	Duration
Stream 00	729230326	Aug 2, 2004	10:37:57	Aug 2, 2004	23:55:29	13:17:32
Stream 01	729230327	Aug 2, 2004	23:55:31	Aug 4, 2004	20:27:28	44:31:57
Stream 02	729230328	Aug 2, 2004	23:55:31	Aug 4, 2004	21:05:47	45:10:16
Stream 03	729230329	Aug 2, 2004	23:55:31	Aug 4, 2004	20:17:18	44:21:47
Stream 04	729230330	Aug 2, 2004	23:55:31	Aug 4, 2004	19:53:32	43:58:01
Stream 05	729230331	Aug 2, 2004	23:55:31	Aug 4, 2004	19:35:23	43:39:52
Stream 06	729230332	Aug 2, 2004	23:55:31	Aug 4, 2004	19:35:42	43:40:11
Stream 07	729230333	Aug 2, 2004	23:55:31	Aug 4, 2004	20:53:39	44:58:08
Refresh		Aug 2, 2004	23:55:30	Aug 4, 2004	2:11:43	26:16:13



Sun Fire™ V880 Server with Sybase IQ 12.5 Single Application Server

TPC-H Rev. 2.0

report date: may 9, 2005
Revision Date: july 25, 2005

TPC-H Timing Intervals (in seconds)

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
Stream 00	2571.16	109.92	1574.18	1045.26	1378.06	209.47	1003.56	648.22	5015.99	2066.05	239.20	682.75
Stream 01	20269.43	238.48	3251.65	9034.93	4063.31	492.01	2762.06	1792.44	16199.54	5843.27	482.73	2250.61
Stream 02	20849.93	371.37	4437.10	10909.44	4870.30	1127.25	3975.00	2315.17	14542.63	5195.12	513.16	1959.34
Stream 03	23326.82	314.59	2481.27	12380.89	9103.69	359.13	2821.32	4971.04	13133.33	6204.62	438.21	1514.14
Stream 04	21743.07	252.16	5040.74	13876.17	10579.94	619.05	3868.14	1959.48	11028.23	8308.28	723.83	2450.34
Stream 05	21112.36	358.95	4032.56	12787.57	3672.91	600.14	4250.79	1604.31	11726.59	6766.38	957.67	2529.41
Stream 06	21310.09	240.26	6087.90	12061.49	4104.83	468.73	3976.93	1911.92	14852.72	16891.43	760.82	1340.00
Stream 07	14519.53	213.00	4395.74	13175.51	3175.56	261.42	3551.24	2060.97	12780.15	7811.37	997.93	725.99
Minimum	14519.53	213.00	2481.27	9034.93	3175.56	261.42	2762.06	1604.31	11028.23	5195.12	438.21	725.99
Average	20447.32	284.11	20447.32	12032.28	5652.93	561.10	3600.78	2373.62	13466.17	8145.78	696.34	1824.26
Maximum	23326.82	371.37	6087.90	13876.17	10579.94	1127.25	4250.79	4971.04	16199.54	16891.43	997.93	2529.41

	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	RF1	Rf2
Stream 00	4687.22	442.17	1070.52	767.73	530.92	8513.95	748.98	1200.31	6720.66	616.91	2894.50	3109.83
Stream 01	11630.39	1641.11	5191.37	3021.57	960.89	32189.56	1818.78	5760.43	29739.63	1677.93	4838.27	4789.78
Stream 02	15589.03	4002.08	4998.98	4410.54	8906.08	26876.40	2523.17	9452.94	12481.56	2306.50	4600.81	4328.83
Stream 03	15872.31	1851.35	2434.72	2550.17	982.81	27172.48	2373.12	7064.05	19924.35	2430.74	4083.51	4856.38
Stream 04	14769.21	1154.51	3999.28	1775.62	3293.78	24529.27	2072.42	2161.73	22319.23	1752.48	4907.78	4191.57
Stream 05	13160.05	2426.26	2418.38	2658.77	1381.10	26737.83	2657.43	3709.46	29524.08	2116.63	3452.02	4380.25
Stream 06	13850.72	1699.95	1816.22	2328.60	1588.60	26297.85	1812.26	1620.52	20792.22	1394.97	4355.22	3808.25
Stream 07	15333.03	563.94	1252.17	1460.05	2779.04	34718.15	1693.48	15828.79	21532.87	3057.26	4202.45	4444.48
Minimum	11630.39	563.94	1252.17	1460.05	960.89	24529.27	1693.48	1620.52	12481.56	1394.97	3452.02	3808.25
Average	14314.96	1905.60	3158.73	2600.76	2841.76	28360.22	2135.81	6513.99	22330.56	2105.21	4348.58	4399.93
Maximum	15872.31	4002.08	5191.37	4410.54	8906.08	34718.15	2657.43	15828.79	29739.63	3057.26	4907.78	4856.38

Table of Contents

INFO SIZING

Benchmark Sponsors:

Brad Carlile
Director, Enterprise Benchmarking
Sun Microsystems, Inc.
8305 S. W. Creekside Place
Beaverton, OR 97008

Paul Kreneta
Chief Technologist, Sybase IQ
Sybase Inc.
One Sybase Drive
Dublin, CA 94568

August 18, 2004

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

Platform: **Sun Fire V880 Server**

Database Manager: **Sybase IQ 12.5**

Operating System: **Solaris 10**

The results were:

CPU (Speed)	Memory	Disks	**QphH@1000GB**
One (1) Sun Fire 880			
8 x UltraSPARC III Cu (1.2 GHz)	32 GB Main	30 x 73 GB	**3247.4**

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:

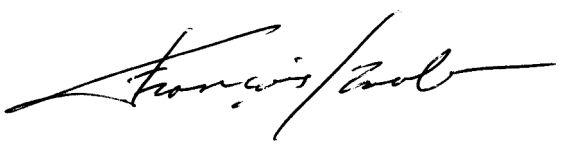
- The database records were defined with the proper layout and size
- The database population was generated using DBGEN
- The database was properly scaled to 1000GB and populated accordingly
- The compliance of the database auxiliary data structures was verified

- 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
- The execution of the queries against the SF1 database produced compliant answers
- A compliant implementation specific layer was used to drive the tests
- The throughput tests involved 7 query streams
- The ratio between the longest and the shortest query was such that no query timing was adjusted
- The execution times for queries and refresh functions were correctly measured and reported
- The repeatability of the measured results was verified
- The required amount of database log was configured
- The system pricing was verified for major components and maintenance
- The major pages from the FDR were verified for accuracy

Additional Audit Notes:

None.

Respectfully Yours,



François Raab, President

Bradley J. Askins, Auditor

TPC Benchmark H Overview

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

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

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

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

1. General Items

1.1 Benchmark Sponsor

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

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

1.2 Parameter Settings

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

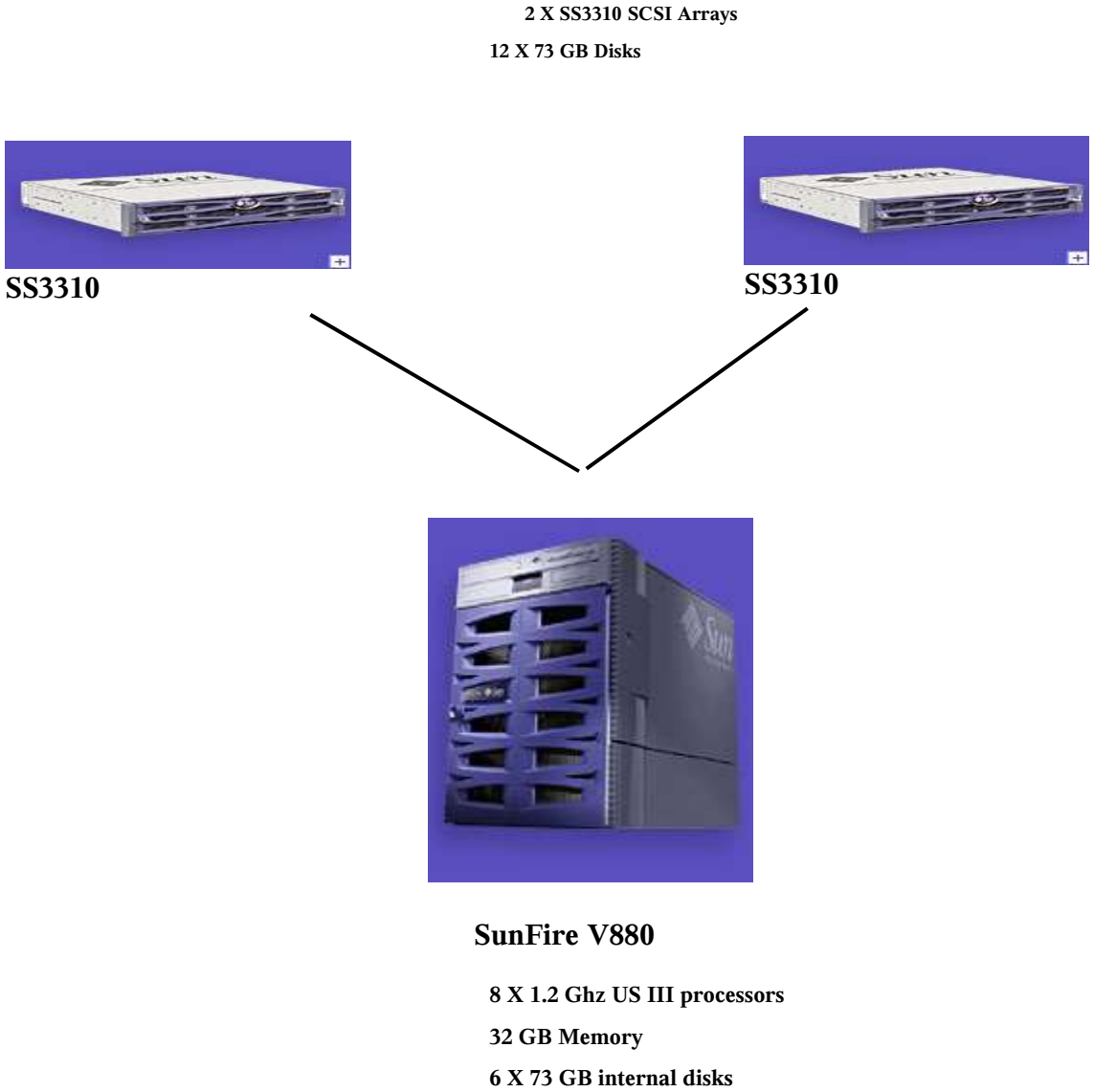
- *Database Tuning Options*
- *Optimizer/Query execution options*
- *Query processing tool/language configuration parameters*
- *Recovery/commit options*
- *Consistency/locking options*
- *Operating system and configuration parameters*
- *Configuration parameters and options for any other software component incorporated into the pricing structure*
- *Compiler optimization options*

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

1.3 Configuration Diagram

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

The priced and measured configurations were identical:



2. Clause 1 Logical Database Design

2.1 Database Definition Statements

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

Appendix B contains the programs and scripts that create and analyze the tables and indexes for the TPC-H database.

2.2 Physical Organization

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

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

2.3 Horizontal Partitioning

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

Horizontal partitioning was not used for any of the tables.

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.

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

3.4 Query Text and Output Data from Qualification Database

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

Appendix C contains the query text and query output. The standard queries were used throughout with the following modifications:

- In Q1, Q4, Q5, Q6, Q10, Q12, Q14, Q15 and Q20, the "dateadd" function is used to perform date arithmetic.
- In Q7, Q8 and Q9, the "datepart" function is used to extract part of a date (e.g., "year").
- In Q2, Q3, Q10, Q18 and Q21, the "top" function is used to restrict the number of output rows.
- The semicolon (;) is used as a command delimiter.

3.5 Query Substitution Parameters and Seeds Used

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

Appendix D contains the seed and query substitution parameters.

3.6 Query Isolation Level

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

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

3.7 Source Code of Refresh Functions

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

Appendix B contains the source code for the refresh functions.

4. Clause 3 Database System Properties

4.1 ACID Properties

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

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

4.2 Atomicity

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

4.2.1 Completed Transaction

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

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

4.2.2 Aborted Transaction

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

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

4.3 Consistency

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

4.3.1 Consistency Test

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

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

4.4 Isolation

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

4.4.1 Read-Write Conflict with Commit

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

1. An ACID Transaction was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID Transaction was suspended prior to COMMIT.
2. An ACID Query was started for the same O_KEY used in step 1.
3. The ACID Transaction was resumed and COMMITTED.
4. The ACID Query completed. It returned the data as committed by the ACID Transaction.

4.4.2 Read-Write Conflict with Rollback

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

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

4.4.3 Write-Write Conflict with Commit

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

1. An ACID Transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. T1 was suspended prior to COMMIT.
2. Another ACID Transaction, T2, was started using the same O_KEY and L_KEY and a randomly selected DELTA.
3. T2 waited.
4. T1 was allowed to COMMIT and T2 completed.

5. It was verified that T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE + (DELTA1* (T1.L_EXTENDEDPRICE/T1.L_QUANTITY))

4.4.4 Write-Write Conflict with Rollback

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

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

4.4.5 Concurrent Progress of Read and Write Transactions

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

1. An ACID Transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. T1 was suspended prior to ROLLBACK.
2. Another Transaction, T2, was started which did the following:

   For random values of PS_PARTKEY and PS_SUPPKEY, all columns of the PARTSUPP table for which PS_PARTKEY and PS_SUPPKEY are equal, are returned.
3. T2 completed.
4. T1 was allowed to COMMIT.
5. It was verified that appropriate rows in ORDERS, LINEITEM and HISTORY tables were changed.

4.4.6 Read-Only Query Conflict with Update Transaction

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

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

4.5 Durability

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

4.5.1 Failure of a Durable Medium

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

All disks containing TPC-H tables, indexes and the catalog file are on RAID5 volumes. In addition the catalog log file is on a mirrored partition. When one of these disks was removed from the shared array during the durability tests, processing continued without interruption. After a span of about 60 seconds, power to the server was cut off. The outcome is described in section 3.5.2.

4.5.2 System Crash

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

The system crash and memory failure tests were combined by cutting off power to server during the durability test. When power was restored, the system rebooted and the database was restarted. The durability success file and the HISTORY table were compared successfully.

4.5.3 Memory Failure

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

See section 3.5.2.

5. Clause 4 Scaling and Database Population

5.1 Ending Cardinality of Tables

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

Table	***Rows***
Lineitem	*6000016166*
orders	*1,500,000,000*
Partsupp	*800,000,000*
Part	*200,000,000*
Customer	*150,000,000*
Supplier	*10,000,000*
Nation	*25*
Region	*5*

5.2 Distribution of Tables and Logs Across Media

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

- All tables and indexes were stored on two RAID 5 volumes. Each volume was constructed from eleven raw partitions using the Solaris Volume Manager.
- The Temp database for each of the two Sybase IQ Multiplex servers was configured using fifteen raw partitions, two on each of two internal disks, and eleven on the six internal disks. The Temp databases were not mirrored.

The following table shows all the non-SVM raw partitions used for the benchmark.

Partition	Use	Size (in GB)
c1t9d0s0	root partition	17.58
c19d0s3	swap	14.77
c1t9d0s1	reader temp	36
c1t1d0s0	reader temp	36
c1t1d0s1	writer temp	32
c1t2d0s0	reader temp	36
c1t2d0s1	writer temp	32
c1t8d0s0	reader temp	36
c1t8d0s1	writer temp	32
c1t10d0s0	reader temp	34
c1t10d0s1	writer temp	32
c1t11d0s0	reader temp	34
c1t11d0s1	writer temp	32
c3t13d0s0	reader temp	36
c3t13d0s1	writer temp	32

Partition	Use	Size (in GB)
c9t13d0s0	reader temp	36
c9t13d0s1	writer temp	32
c1t1d0s7	SVM state replica	.0015
c1t2d0s7	SVM state replica	.0015
c1t8d0s7	SVM state replica	.0015
c1t10d0s7	SVM state replica	.0015
c1t11d0s7	SVM state replica	.0015

The next table shows the partitions used for the SVM devices. The /sybase2 file system is used to store the IQ catalog file and catalog log.

SVM Device Details:

Raw Partition Name	SVM Device Name	SVM Containing Device	Symbolic Link	Database Usage	Database Device Size GB	RAID
c2t5d0s7	d21	d20	none	/sybase2	2	RAID 1
c2t12d0s7	d22	d20	none	/sybase2)	2	RAID 1
c2t8d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c2t9d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c2t10d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c2t11d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c2t12d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c2t13d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c3t8d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c3t9d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c3t10d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c3t11d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c3t12d0s1		d5	/sybase2/M1	IQ Main	68.35	RAID5
c8t8d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c8t9d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c8t10d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c8t11d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c8t12d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c8t13d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c9t8d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c9t9d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c9t10d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c9t11d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5
c9t12d0s1		d6	/sybase2/M1	IQ Main	68.35	RAID5

Additional details can be found in the disk configuration section in Appendix B.

5.3 Database partition/replication mapping

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

Database partitioning/replication was not used.

5.4 RAID Feature

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

RAID 5 was used for all base tables and auxiliary data structures. In addition, the Sybase IQ root .db file and log file were protected by RAID 1.

5.5 Modifications to the DBGEN

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

The supplied DBGEN version 1.3.0 was used to generate the database population for this benchmark.

5.6 Database Load Time

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

The database load time was =24 hrs 49 mins

5.7 Data Storage Ratio

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

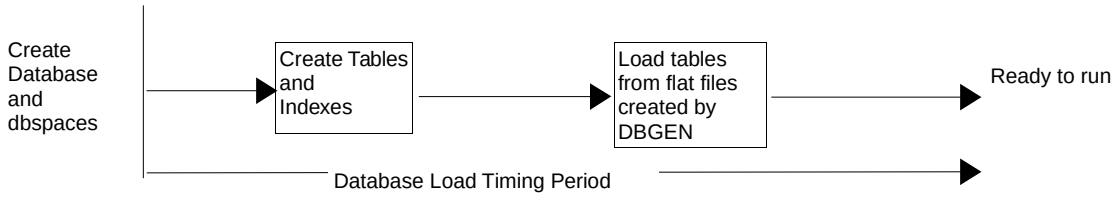
The data storage ratio is computed from the following information:

Disk Type	# Of Disks	Space Per Disk*	Sub-Total Disk Space**
internal	6	73 GB	407.92 GB
external	24	73 GB	1631.68 GB
		Total Space	**2039.6 GB**
		Data Storage Ratio	**2.04**

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

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 test database was loaded using flat files. All load scripts are included in Appendix B.

5.9 Qualification Database Configuration

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

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

6. Clause 5 Performance Metrics and Execution Rules

6.1 System Activity Between Load and Performance Tests

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

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

All scripts and queries used are included in Appendix F

6.2 Steps in the Power Test

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

The following steps were used to implement the power test:

1. RF1 Refresh Transaction
2. Stream 00 Execution
3. RF2 Refresh Transaction

6.3 Timing Intervals for Each Query and Refresh Functions

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

The timing intervals for each query and both update functions are reported on the page titled “Numerical Quantities”, contained in the beginning of this document and replicated in the Executive Summary document.

6.4 Number of Streams for the Throughput Test

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

Seven streams were used for the throughput test.

6.5 Start and End Date/Times for Each Query Stream

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

The start times and finish times for each query stream in the throughput test are reported on the page titled “Numerical Quantities”, contained in the beginning of this document and replicated in the Executive Summary 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 reported on the page titled “Numerical Quantities”, contained in the beginning of this document and replicated in the Executive Summary document.

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

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

The start and finish times for each refresh function:

Stream ID	Refresh Function	Start Date	Start Time	End Date	End Time
Stream 01	RF1	Aug 2, 2004	23:55:30	Aug 3, 2004	1: 6:08
Stream 01	RF2	Aug 3, 2004	1:21:09	Aug 3, 2004	2:40:59
Stream 02	RF1	Aug 3, 2004	2:46:00	Aug 3, 2004	4:02:41
Stream 02	RF2	Aug 3, 2004	4:07:41	Aug 23 2004	5:19:50
Stream 03	RF1	Aug 3, 2004	9:44:59	Aug 3, 2004	10:53:03
Stream 03	RF2	Aug 3, 2004	10:58:03	Aug 3, 2004	12:19:00
Stream 04	RF1	Aug 3, 2004	12:24:00	Aug 3, 2004	13:45:48
Stream 04	RF2	Aug 3, 2004	13:50:49	Aug 3, 2004	15:00:40
Stream 05	RF1	Aug 3, 2004	18:10:47	Aug 3, 2004	19:8:19
Stream 05	RF2	Aug 3, 2004	19:13:20	Aug 3, 2004	20:26:20
Stream 06	RF1	Aug 3, 2004	20:31:20	Aug 3, 2004	21:43:56
Stream 06	RF2	Aug 3, 2004	22:34:06	Aug 3, 2004	23:37:34
Stream 07	RF1	Aug 3, 2004	23:42:35	Aug 4, 2004	0:52:38
Stream 07	RF2	Aug 4, 2004	0:57:38	Aug 4, 2004	2:11:43

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

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

The timing intervals for each query and each refresh function for the throughput test are reported on the page titled “Numerical Quantities”, contained in the beginning of this document and replicated in the Executive Summary document.

6.9 Performance Metrics

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

The performance metrics, and the numbers on which they are based, are reported on the page titled “Numerical Quantities”, contained in the beginning of this document and replicated in the Executive Summary 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 three metrics:

Run ID	QppH@1000GB	QthH@1000GB	QphH@1000GB
Run 1	3,101.6	3,444.7	3268.7
Run 2	3,093.9	3,409.3	3,247.4
% Difference	-0.20%	-1.02%	-0.07%

6.11 System Activity Between Performance Tests

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

The database was not restarted after it was loaded or between the two runs.

7. Clause 6 SUT and Driver Implementation

7.1 Driver

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

The entire test is run by executing the shellscript newtest..

The text of newtest is provided in Appendix E and the scripts that newtest executes are provided in Appendix B.

The Power Test and Throughput Test are performed by the do_test.bash script. The query streams are generated by a script called gen_streams.ksh which uses QGEN to generate the query stream files.

The Power Test is invoked within the do_test.bash script by executing the update_power.sql which runs the refresh functions and the stream0.sql which contains the power stream queries.

The Throughput Test is invoked within the do_test.bash script by executing the five query stream scripts stream[1-7].sql files along with simultaneously executing the update_throughput.sql script. The refresh functions execute in parallel with the ad-hoc queries streams trickle feeding the refresh transactions.

7.2 Implementation-Specific Layer

If an implementation-specific layer is used, then a detailed description of how it performs its functions must be supplied, including any related source code or scripts. This description should allow an independent reconstruction of the implementation-specific layer.

The configuration of the IQ Multiplex instances was done through Sybase Central, a graphical management tool that is supplied as part of the Sybase IQ software. The test sponsor started the Sybase Central software, connecting to the initial database. The second Multiplex instance was created through a Create Multiplex dialog. The parameters used to configure the multiplex were the node name (same as the initial database) and port number of the second Multiplex database. All other database configuration was done through scripts disclosed in Appendix B.

The performance tests are performed using dbisqlc. dbisqlc is a Sybase-provided utility that allows SQL statements to be executed against an Sybase IQ database. The dbisqlc utility is invoked from the command-line on the SUT. It reads input from files containing SQL statements and sends results to stdout. dbisqlc uses information in the .odbc.ini file to connect to the database. Performance test scripts utilizing dbisqlc can be found in Appendix E.

The ACID tests are performed using dbtest. dbtest is a Sybase-provided utility, similar to dbisqlc, but providing additional scripting capabilities. It is invoked from the command-line on the SUT and uses information in the .odbc.ini file to connect to the database. ACID test scripts utilizing dbtest can be found in Appendix B.

7.3 Profile-Directed Optimization

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

Profile-directed optimization was not used.

8. Clause 7 Pricing

8.1 Hardware and Software Used

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

Refer to the Executive Summary.

8.2 Total Three Year Price

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

The total 3-year price of the configuration is $ 171,455.20. For details of pricing, see the second page of the Executive Summary.

Discounts were taken from actual price quotes,available to any buyer with like conditions, provided by Eakins Open Systems Inc. and Sybase Inc. The respective price quotes are included in Appendix G of this document.

8.3 Availability Date

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

All the components used in the measured configuration will be generally available as of may 9, 2005.

9. Auditor's Information and Attestation Letter

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

The auditor's attestation letter follows the table of contents.

Appendix A. Solaris 9 and Sybase IQ 12.5 Parameters

This Appendix contains Solaris kernel parameters and environment variables and Sybase IQ system parameters.

```
================================================
```

Sybase IQ Server Configuration Parameters

```
================================================
```

params.cfg

```
#
# these are the default suggested startup parameters
for the asiq
#
# please note that you will still need to provide the
server
# with the server name on startup "-n", and possibly a
# different system port number
#
-n tpch880_2
-x tcpip{port=3002}
-c  8m
-gd all
-gl all
-gm 18
-gp 4096
-ti 4400
-tl 300
-iqmc 8900
-iqtc 17800
-iqmt 1000
-iqgovern 16
-iqpartition 4
```

writer.cfg

```
#
# these are the default suggested startup parameters
for the asiq
#
# please note that you will still need to provide the
server
# with the server name on startup "-n", and possibly a
# different system port number
#
-n tpch880_1
-x tcpip{port=3001}
-c  8m
-gd all
-gl all
-gm 12
-gp 4096
-ti 4400
-tl 300
-iqmc 1250
-iqtc 1450
-iqmt 400
-iqgovern 5
-iqpartition 1
```

```
================================================
```

Sybase IQ Database Options

(altered from default)

```
================================================
```

options.sql

```
================================================
SET OPTION PUBLIC.Row_Counts='On';
SET OPTION PUBLIC.Main_Reserved_DBSpace_MB=300;

SET OPTION PUBLIC.Force_No_Scroll_Cursors='On';
SET OPTION PUBLIC.Max_IQ_Threads_Per_Connection=100;
SET OPTION PUBLIC.Query_Temp_Space_Limit=0;
SET OPTION PUBLIC.Hash_Thrashing_Percent=100;
SET OPTION
PUBLIC.SignificantDigitsForDoubleEquality=15;
SET OPTION PUBLIC.Flatten_Subqueries = 'On';
SET OPTION PUBLIC.Default_Like_Range_Selectivity = 1;
SET OPTION PUBLIC.Default_Having_Selectivity = 1;
SET OPTION PUBLIC.Max_Hash_Rows = 25000000;
SET OPTION PUBLIC.Sort_Phase1_Helpers=3;

SET OPTION PUBLIC.Sweeper_Threads_Percent=12;
SET OPTION PUBLIC.Prefetch_Threads_Percent=14;
SET OPTION PUBLIC.Wash_Area_Buffers_Percent = '20';

SET OPTION PUBLIC.Load_Memory_Mb=0;
SET OPTION PUBLIC.Append_Load='On';
SET OPTION PUBLIC.Garray_Fill_Factor_Percent=3;
SET OPTION PUBLIC.Minimize_Storage='On';
SET OPTION PUBLIC.Notify_Modulus=10000000;
SET OPTION PUBLIC.Allow_Nulls_By_Default='Off';
```

```
================================================
```

options_reader.sql

```
================================================
SET OPTION PUBLIC.Main_Reserved_DBSpace_MB=300;
SET OPTION PUBLIC.Max_IQ_Threads_Per_Connection=70;
SET OPTION PUBLIC.Sweeper_Threads_Percent=6;
SET OPTION PUBLIC.Prefetch_Threads_Percent=15;
SET OPTION PUBLIC.Wash_Area_Buffers_Percent = '20';
SET OPTION PUBLIC.Sort_Phase1_Helpers=3;
```

```
================================================
```

Sybase IQ Environment Variables

```
================================================

SYBASE="/export/home/sybase"
export SYBASE
SYBASE_OCS="OCS-12_5"
export SYBASE_OCS
SYBASE_JRE="${SYBASE}/shared/jre-1_22"
export SYBASE_JRE
ASDIR="${SYBASE}/ASIQ-12_5"
export ASDIR
PATH="${ASDIR}/bin:${SYBASE}/${SYBASE_OCS}/bin:$
{PATH}:/etc:."
export PATH
IQLIB="${ASDIR}/usrlib:${ASDIR}/lib:${SYBASE}/
${SYBASE_OCS}/lib"
LD_LIBRARY_PATH_64="${IQLIB}:$LD_LIBRARY_PATH_64"
export LD_LIBRARY_PATH_64
LD_LIBRARY_PATH="${IQLIB}:$LD_LIBRARY_PATH"
export LD_LIBRARY_PATH
unset IQLIB
LD_PRELOAD=mpss.so.1
export LD_PRELOAD
MPSSHEAP=512K
MPSSSTACK=64K
export MPSSHEAP
export MPSSSTACK
```

```
================================================
```

.odbc.ini

```
================================================
[ODBC Data Sources]
tpch=ASIQ Driver
utility_db=ASIQ Driver

[tpch1]
Driver=/export/home/sybase/asiq12/lib/dbodbc7_r.so.1
EngineName=tpch880_1
CommLinks=tcpip{host=10.8.5.64;Port=3001}
DatabaseName=tpch880
UserID=DBA
Password=SQL
DBG=yes
LOG=/tmp/tpch_odbc.log

[tpch2]
Driver=/export/home/sybase/asiq12/lib/dbodbc7_r.so.1
EngineName=tpch880_2
CommLinks=tcpip{host=10.8.5.64;Port=3002}
DatabaseName=tpch880
```

```
UserID=DBA
Password=SQL
DBG=yes
LOG=/tmp/tpch_odbc.log

[utility_db]
Driver=/export/home/sybase/asiq12/lib/dbodbc7_r.so.1
EngineName=tpch440
CommLinks=tcpip{host=10.8.5.64;Port=2638}
DatabaseName=utility_db
UserID=DBA
Password=SQL
DBG=yes
LOG=/tmp/utility_db_odbc.log
```

Solaris Parameters

(altered from default)

```
================================================
```

/etc/system

```
================================================

set tune_t_fsflushr=600
set autoup=36000000
```

Appendix B. Programs and Scripts

==========

check_query1.bash

==========

```
#!/bin/bash
#
# First remove the rf1.lock so that the Query Stream
will start
#
rm -f /export/home/sybase/run/scripts/rf1.lock
#
# Sleep while the rf2.lock file exists
# when the query stream completes it will remove the
rf2.lock
#
while [ -f /export/home/sybase/run/scripts/rf2.lock ]
do
 # Wait for the Query Steam to complete
 # check every 10 seconds
 # echo "Lock File Exists"
 sleep 10
done
# Return Control to the RF stream
```

==========

create_database.sql

```
CREATE DATABASE '/sybase2/tpch880.db'
TRANSACTION LOG ON
COLLATION 'ISO_BINENG'
CASE RESPECT
PAGE SIZE 4096
BLANK PADDING ON
JAVA ON
JCONNECT ON
IQ PATH '/sybase2/M01'
IQ PAGE SIZE 524288
TEMPORARY PATH '/sybase2/WT01'
```

==========

create_dbspaces.sql (writer instance only)

==========

```
create dbspace main2     as '/sybase2/M02' iq store;

create dbspace iqtemp2   as '/sybase2/WT02' iq
temporary store;
create dbspace iqtemp3   as '/sybase2/WT03' iq
temporary store;
create dbspace iqtemp4   as '/sybase2/WT04' iq
temporary store;
create dbspace iqtemp5   as '/sybase2/WT05' iq
temporary store;
create dbspace iqtemp6   as '/sybase2/WT06' iq
temporary store;
create dbspace iqtemp8   as '/sybase2/WT07' iq
temporary store;
```

==========

create_dbspaces2.sql (reader instance only)

```
create dbspace iqtmp02 as '/sybase2/T02' iq temporary
store;
create dbspace iqtmp03 as '/sybase2/T03' iq temporary
store;
create dbspace iqtmp04 as '/sybase2/T04' iq temporary
store;
create dbspace iqtmp05 as '/sybase2/T05' iq temporary
store;
create dbspace iqtmp06 as '/sybase2/T06' iq temporary
store;
create dbspace iqtmp07 as '/sybase2/T07' iq temporary
store;
```

==========

create_tables.sql

==========

```
CREATE TABLE region
  (
    r_regionkey            unsigned int,
    r_name                 char(25),
    r_comment              varchar(152),
    PRIMARY KEY (r_regionkey)
  );


CREATE TABLE nation
  (
    n_nationkey            unsigned int,
    n_name                 char(25),
    n_regionkey            unsigned int,
    n_comment              varchar(152),
    PRIMARY KEY (n_nationkey)
  );
CREATE HG INDEX n_regionkey_hg ON nation
(n_regionkey) ;


CREATE TABLE supplier
  (
    s_suppkey              unsigned int,
    s_name                 char(25),
    s_address              varchar(40),
    s_nationkey            unsigned int,
    s_phone                char(15),
    s_acctbal              double precision,
    s_comment              varchar(101),
    PRIMARY KEY (s_suppkey)
  );
CREATE HG INDEX s_nationkey_hg ON supplier
(s_nationkey) ;


CREATE TABLE part
  (
    p_partkey              unsigned int,
    p_name                 varchar(55),
    p_mfgr                 char(25),
    p_brand                char(10),
    p_type                 varchar(25),
    p_size                 int,
    p_container            char(10),
    p_retailprice          double precision,
    p_comment              varchar(23),
    PRIMARY KEY(p_partkey)
  );

CREATE TABLE partsupp
  (
    ps_partkey             unsigned int,
    ps_suppkey             unsigned int,
    ps_availqty            integer,
    ps_supplycost          double precision,
    ps_comment             varchar(199),
    PRIMARY KEY (ps_partkey, ps_suppkey)
  );
CREATE HG INDEX ps_partkey_hg ON partsupp
(ps_partkey) ;
CREATE HG INDEX ps_suppkey_hg ON partsupp
(ps_suppkey) ;


CREATE TABLE customer
  (
    c_custkey              unsigned int,
    c_name                 varchar(25),
    c_address              varchar(40),
    c_nationkey            unsigned int,
    c_phone                char(15),
    c_acctbal              double precision,
    c_mktsegment           char(10),
    c_comment              varchar(117),
    PRIMARY KEY(c_custkey)
  );
CREATE HG INDEX c_nationkey_hg ON customer
(c_nationkey) ;


CREATE TABLE orders
```

```
  (
    o_orderkey              unsigned int,
    o_custkey               unsigned int,
    o_orderstatus           char(1),
    o_totalprice            double precision,
    o_orderdate             date,
    o_orderpriority         char(15),
    o_clerk                 char(15),
    o_shippriority          int,
    o_comment               varchar(79),
    PRIMARY KEY (o_orderkey)
  );
CREATE HG INDEX o_custkey_hg ON orders(o_custkey) ;
CREATE DATE INDEX o_orderdate_date ON orders
(o_orderdate) ;


CREATE TABLE lineitem
  (
    l_orderkey              unsigned int,
    l_partkey               unsigned int,
    l_suppkey               unsigned int,
    l_linenumber            int,
    l_quantity              double precision,
    l_extendedprice         double precision,
    l_discount              double precision,
    l_tax                   double precision,
    l_returnflag            char(1),
    l_linestatus            char(1),
    l_shipdate              date,
    l_commitdate            date,
    l_receiptdate           date,
    l_shipinstruct          char(25),
    l_shipmode              char(10),
    l_comment               varchar(44)
  );

CREATE HG INDEX l_partsupp_hg ON lineitem
(l_partkey,l_suppkey) ;
CREATE HG INDEX l_orderkey_hg ON lineitem
(l_orderkey) ;
CREATE HG INDEX l_partkey_hg ON lineitem(l_partkey) ;
CREATE HG INDEX l_suppkey_hg ON lineitem(l_suppkey) ;
CREATE DATE INDEX l_shipdate_date ON lineitem
(l_shipdate) ;
CREATE DATE INDEX l_receiptdate_date ON lineitem
(l_receiptdate);
```

==========

tpch_rf.sql

==========

```
create table refresh_control ( rf1_data_set int not
null, rf2_data_set int not null);
insert into refresh_control values (0,0);
commit;
CREATE PROCEDURE DBA.tpch_rf1 (IN c_directory varchar
(128),
                               IN c_stream varchar(3))
ON EXCEPTION RESUME
BEGIN
  DECLARE delim_asci integer;
  DECLARE c_data_set varchar(3);
  DECLARE i_data_set integer;
  DECLARE c_cmd long varchar;
  DECLARE outfilename varchar(128); -- Debug
  DECLARE outfilename2 varchar(128); -- Debug
  DECLARE c_lf varchar(2);
  DECLARE t_qstart timestamp;
  DECLARE t_qstop timestamp;
  DECLARE n_seconds numeric(12,5);
  DECLARE c_sqlstate CHAR(5);
  SET t_qstart = now(*);
  SET c_lf=char(10);
  SELECT rf1_data_set INTO i_data_set FROM
refresh_control;
  SET c_data_set=CAST(i_data_set+1 AS varchar(3));
  SET c_cmd='load table orders ( '+c_lf;
  SET c_cmd=c_cmd+' o_orderkey '+char(39)+'|'+char(39)
+', '+c_lf;
  SET c_cmd=c_cmd+' o_custkey '+char(39)+'|'+char(39)
+', '+c_lf;
  SET c_cmd=c_cmd+' o_orderstatus '+char(39)+'|'+char
(39)+', '+c_lf;
  SET c_cmd=c_cmd+' o_totalprice '+char(39)+'|'+char
(39)+', '+c_lf;
  SET c_cmd=c_cmd+' o_orderdate date('+char(39)+'YYYY-
MM-DD'+char(39)+'), filler(1), '+c_lf;
  SET c_cmd=c_cmd+' o_orderpriority '+char(39)+'
|'+char(39)+', '+c_lf;
  SET c_cmd=c_cmd+' o_clerk '+char(39)+'|'+char(39)+',
'+c_lf;
  SET c_cmd=c_cmd+' o_shippriority '+char(39)+'|'+char
(39)+','+c_lf;
  SET c_cmd=c_cmd+' o_comment '+char(39)+'|'+char(39)
+' ) '+c_lf;
  SET c_cmd=c_cmd+'from '+char(39)
+c_directory+'orders.tbl.u'+c_data_set+char(39)+c_lf;
  SET c_cmd=c_cmd+'row delimited by '+char(39)
+'\\x0a'+char(39)+' quotes off escapes off preview
on;';
  EXECUTE IMMEDIATE c_cmd;
  SELECT SQLSTATE INTO c_sqlstate;
  IF c_sqlstate != '00000' THEN
    ROLLBACK;
    RAISERROR 23002 'RF1 failed at Step 1 with
SQLSTATE: ', c_sqlstate;
    RETURN(1);
  END IF;
  SET c_cmd='load table lineitem ( '+c_lf;
  SET c_cmd=c_cmd+' l_orderkey '+char(39)+'|'+char(39)
+', '+c_lf;
  SET c_cmd=c_cmd+' l_partkey '+char(39)+'|'+char(39)
+', '+c_lf;
  SET c_cmd=c_cmd+' l_suppkey '+char(39)+'|'+char(39)
+', '+c_lf;
  SET c_cmd=c_cmd+' l_linenumber '+char(39)+'|'+char
(39)+', '+c_lf;
  SET c_cmd=c_cmd+' l_quantity '+char(39)+'|'+char(39)
+', '+c_lf;
  SET c_cmd=c_cmd+' l_extendedprice '+char(39)+'
|'+char(39)+', '+c_lf;
  SET c_cmd=c_cmd+' l_discount '+char(39)+'|'+char(39)
+', '+c_lf;
  SET c_cmd=c_cmd+' l_tax '+char(39)+'|'+char(39)+',
'+c_lf;
  SET c_cmd=c_cmd+' l_returnflag '+char(39)+'|'+char
(39)+', '+c_lf;
  SET c_cmd=c_cmd+' l_linestatus '+char(39)+'|'+char
(39)+', '+c_lf;
  SET c_cmd=c_cmd+' l_shipdate date('+char(39)+'YYYY-
MM-DD'+char(39)+'), filler(1), '+c_lf;
  SET c_cmd=c_cmd+' l_commitdate date('+char(39)
+'YYYY-MM-DD'+char(39)+'), filler(1), '+c_lf;
  SET c_cmd=c_cmd+' l_receiptdate date('+char(39)
+'YYYY-MM-DD'+char(39)+'), filler(1), '+c_lf;
  SET c_cmd=c_cmd+' l_shipinstruct '+char(39)+'|'+char
(39)+', '+c_lf;
  SET c_cmd=c_cmd+' l_shipmode '+char(39)+'|'+char(39)
+', '+c_lf;
  SET c_cmd=c_cmd+' l_comment '+char(39)+'|'+char(39)
+' )'+c_lf;
  SET c_cmd=c_cmd+'from '+char(39)
+c_directory+'lineitem.tbl.u'+c_data_set+char(39)
+c_lf;
  SET c_cmd=c_cmd+'row delimited by '+char(39)
+'\\x0a'+char(39)+c_lf+'quotes off escapes off preview
on;';
  EXECUTE IMMEDIATE c_cmd;
  SELECT SQLSTATE INTO c_sqlstate;
  IF c_sqlstate != '00000' THEN
    rollback;
    RAISERROR 23002 'RF1 failed at Step 2 with
SQLSTATE: ', c_sqlstate;
    RETURN(1);
  END IF;
  UPDATE refresh_control SET rf1_data_set=cast
(c_data_set AS integer);
  COMMIT;
  SET t_qstop = now(*);
  SET n_seconds=cast(datediff
(millisecond,t_qstart,t_qstop) AS numeric(12,5))/1000;
  SET c_cmd='Stream updates Update
update_'+c_stream+'_RF1 LENGTH -- '+cast(n_seconds AS
varchar(20))+ ' seconds' ;
  SELECT c_cmd;
  RETURN(0);
END;
CREATE PROCEDURE DBA.tpch_rf2 (in c_directory varchar
```

```
(128),
                                in c_stream varchar(3))
ON exception resume
BEGIN
  DECLARE delim_asci integer;
  DECLARE c_data_set varchar(3);
  DECLARE i_data_set integer;
  DECLARE c_cmd long varchar;
  DECLARE outfilename varchar(128); -- Debug
  DECLARE c_lf varchar(2);
  DECLARE t_qstart timestamp;
  DECLARE t_qstop timestamp;
  DECLARE n_seconds numeric(12,5);
  DECLARE c_sqlstate CHAR(5);
  SET t_qstart = now(*);
  SET c_lf=char(10);
  SELECT rf2_data_set INTO i_data_set FROM
refresh_control;
  SET c_data_set=CAST(i_data_set+1 AS varchar(3));
  CREATE TABLE #delete_table ( d_orderkey UNSIGNED
INT, PRIMARY KEY (d_orderkey) );
  SET c_cmd='load table #delete_table  (d_orderkey
'+char(39)+'\\x0a'+char(39)+') '+c_lf;
  SET c_cmd=c_cmd+'from '+char(39)
+c_directory+'delete.'+c_data_set+char(39)+c_lf;
  SET c_cmd=c_cmd+'quotes off '+c_lf;
  SET c_cmd=c_cmd+'escapes off; '+c_lf;
  EXECUTE IMMEDIATE c_cmd;
  SELECT SQLSTATE INTO c_sqlstate;
  IF c_sqlstate != '00000' THEN
    ROLLBACK;
    SET c_cmd='RF2 failed at Step 1 with SQLSTATE:
'+c_sqlstate;
    RAISERROR 23002 c_cmd;
    RETURN(1);
  END IF;
  DELETE lineitem FROM lineitem, #delete_table WHERE
l_orderkey = d_orderkey;
  SELECT SQLSTATE INTO c_sqlstate;
  IF c_sqlstate != '00000' THEN
    ROLLBACK;
    SET c_cmd='RF2 failed at Step 2 with SQLSTATE:
'+c_sqlstate;
    RAISERROR 23002 c_cmd;
    RETURN(1);
  END IF;
  DELETE orders FROM orders, #delete_table WHERE
o_orderkey = d_orderkey;
  SELECT SQLSTATE INTO c_sqlstate;
  IF c_sqlstate != '00000' THEN
    ROLLBACK;
    SET c_cmd='RF2 failed at Step 3 with SQLSTATE:
'+c_sqlstate;
    RAISERROR 23002 c_cmd;
    RETURN(1);
  END IF;
  UPDATE refresh_control SET rf2_data_set=CAST
(c_data_set AS integer);
  COMMIT;
  DROP TABLE #delete_table;
  SET t_qstop = now(*);
  SET n_seconds=cast(datediff
(millisecond,t_qstart,t_qstop) as numeric(12,5))/1000;
  SET c_cmd='Stream updates Update
update_'+c_stream+'_RF2 LENGTH -- '+cast(n_seconds as
varchar(20))+ ' seconds' ;
  SELECT c_cmd;
  RETURN(0);
END;
```

==========

load_region.sql

==========

```
LOAD TABLE REGION (
R_REGIONKEY                                    '|',
R_NAME                                         '|',
R_COMMENT                                      '|'
)
FROM '/sybase_stage/region.tbl'
escapes off
quotes off
row delimited by '\x0a'
WITH CHECKPOINT ON;
commit;
```

==========

load_nation.sql

==========

```
LOAD TABLE NATION (
N_NATIONKEY                                    '|',
N_NAME                                         '|',
N_REGIONKEY                                    '|',
N_COMMENT                                      '|'
)
FROM '/sybase_stage/nation.tbl'
escapes off
quotes off
row delimited by '\x0a'
WITH CHECKPOINT ON;
commit;
```

==========

load_customer.sql

==========

```
LOAD TABLE CUSTOMER (
C_CUSTKEY                                      '|',
C_NAME                                         '|',
C_ADDRESS                                      '|',
C_NATIONKEY                                    '|',
C_PHONE                                        '|',
C_ACCTBAL                                      '|',
C_MKTSEGMENT                                   '|',
C_COMMENT                                      '|'
)
FROM '/sybase_stage/customer.tbl'
escapes off
quotes off
row delimited by '\x0a'
WITH CHECKPOINT ON;
commit;
```

==========

load_part.sql

==========

```
LOAD TABLE PART (
P_PARTKEY                                      '|',
P_NAME                                         '|',
P_MFGR                                         '|',
P_BRAND                                        '|',
P_TYPE                                         '|',
P_SIZE                                         '|',
P_CONTAINER                                    '|',
P_RETAILPRICE                                  '|',
P_COMMENT                                      '|'
)
FROM '/sybase_stage/part.tbl'
escapes off
quotes off
row delimited by '\x0a'
WITH CHECKPOINT ON;
commit;
```

==========

load_supplier.sql

==========

```
LOAD TABLE SUPPLIER (
S_SUPPKEY                                      '|',
S_NAME                                         '|',
S_ADDRESS                                      '|',
S_NATIONKEY                                    '|',
S_PHONE                                        '|',
S_ACCTBAL                                      '|',
S_COMMENT                                      '|'
)
FROM '/sybase_stage/supplier.tbl'
escapes off
quotes off
row delimited by '\x0a'
WITH CHECKPOINT ON;
commit;
```

==========

load_partsupp.sql

```
================================================

LOAD TABLE PARTSUPP (
PS_PARTKEY                                  '|',
PS_SUPPKEY                                  '|',
PS_AVAILQTY                                 '|',
PS_SUPPLYCOST                               '|',
PS_COMMENT                                  '|'
)
FROM '/sybase_stage/partsupp.tbl'
escapes off
quotes off
row delimited by '\x0a'
WITH CHECKPOINT ON;
commit;

================================================
```

load_orders.sql

```
================================================

LOAD TABLE ORDERS (
O_ORDERKEY                                      '|',
O_CUSTKEY                                       '|',
O_ORDERSTATUS                                   '|',
O_TOTALPRICE                                    '|',
O_ORDERDATE                                     '|',
O_ORDERPRIORITY                                 '|',
O_CLERK                                         '|',
O_SHIPPRIORITY                                  '|',
O_COMMENT                                       '|'
)
FROM
'/sybase_stage/orders.tbl'
escapes off
quotes off
row delimited by '\x0a'
WITH CHECKPOINT ON;
commit;

================================================
```

load_lineitem.sql

```
================================================

LOAD TABLE LINEITEM (
L_ORDERKEY                                  '|',
L_PARTKEY                                   '|',
L_SUPPKEY                                   '|',
L_LINENUMBER                                '|',
L_QUANTITY                                  '|',
L_EXTENDEDPRICE                             '|',
L_DISCOUNT                                  '|',
L_TAX                                       '|',
L_RETURNFLAG                                '|',
L_LINESTATUS                                '|',
L_SHIPDATE                                  '|',
L_COMMITDATE                                '|',
L_RECEIPTDATE                               '|',
L_SHIPINSTRUCT                              '|',
L_SHIPMODE                                  '|',
L_COMMENT                                   '|'
)
FROM
     '/sybase_stage/lineitem.tbl'
escapes off
quotes off
row delimited by '\x0a'
WITH CHECKPOINT ON;
commit;
checkpoint;
commit;

================================================
```

update_power.sql

```
================================================

create variable qstart timestamp;
create variable qstop timestamp;
create variable c_sqlstate CHAR(5);
create variable c_path varchar(128);
set c_path='/sybase_stage/';
set qstart=now(*);
select 'Stream 0 RF1 START -- ', qstart ;
call tpch_rf1 (c_path,'0');
set qstop=now(*);
select 'Stream 0 Update RF1 LENGTH -- ',cast(datediff
(millisecond,qstart,qstop) as numeric)/1000, '
seconds';
select 'Stream 0 RF1 FINISH -- ', qstop ;
-- Sleep Until the query stream completes
set qstart = now(*);
select 'Stream 0 RF WAITING -- ', qstart;
xp_cmdshell
('/export/home/sybase/run/scripts/check_query1.bash');
set qstart = now(*);
select 'Stream 0 RF CONTINUING -- ', qstart;
set qstart = now(*);
select 'Stream 0 RF2 START -- ', qstart ;
call tpch_rf2 (c_path,'0');
set qstop=now(*);
select 'Stream 0 Update RF2 LENGTH -- ',cast(datediff
(millisecond,qstart,qstop) as numeric)/1000, '
seconds';
select 'Stream 0 RF2 FINISH -- ', qstop ;

================================================
```

update_throughput.sql

```
create variable qstart timestamp;
create variable qstop timestamp;
create variable c_sqlstate CHAR(5);
create variable c_path varchar(128);
set qstart = now(*);
set c_path='/sybase_stage/';
select 'Stream updates START -- ', qstart ;
select @@servername, db_name();
call tpch_rf1 (c_path,'1');
commit;
tpch_wait;
call tpch_rf2 (c_path,'1');
commit;
tpch_wait;
call tpch_rf1 (c_path,'2');
commit;
tpch_wait;
call tpch_rf2 (c_path,'2');
commit;
tpch_wait;
call tpch_rf1 (c_path,'3');
commit;
tpch_wait;
call tpch_rf2 (c_path,'3');
commit;
tpch_wait;
call tpch_rf1 (c_path,'4');
commit;
tpch_wait;
call tpch_rf2 (c_path,'4');
commit;
tpch_wait;
call tpch_rf1 (c_path,'5');
commit;
tpch_wait;
call tpch_rf2 (c_path,'5');
commit;
tpch_wait;
call tpch_rf1 (c_path,'6');
commit;
tpch_wait;
call tpch_rf2 (c_path,'6');
commit;
tpch_wait;
call tpch_rf1 (c_path,'7');
commit;
tpch_wait;
call tpch_rf2 (c_path,'7');
commit;
set qstop = now(*);
select 'Stream updates STOP -- ', qstop ;

================================================
```

my_start_asiq

```
================================================
#!/bin/bash

export LD_PRELOAD=mpss.so.1
```

```
export MPSSHEAP=$1
export MPSSSTACK=$2

start_asiq  $3  $4
================================================
```

start_asiq

```
================================================
#!/bin/sh
#
------------------------------------------------------
---------------
# @(#)start_asiq              01/20/2004
Sybase, Inc.
#
------------------------------------------------------
---------------
# Start Adaptive Server IQ 12.5
#
------------------------------------------------------
---------------
# Copyright (c) 1995 - 2003 Sybase, Inc.  All rights
reserved.
# Unpublished rights reserved under U.S. copyright
laws. This software
# contains confidential and trade secret information
of Sybase, Inc.
# Use, duplication or disclosure of the software and
documentation by
# the U.S. Government is subject to restrictions set
forth in a license
# agreement between the Government and Sybase, Inc. or
other written
# agreement specifying the Government's rights to use
the software and
# any applicable FAR provisions, for example, FAR
52.227-19.
#
------------------------------------------------------
---------------


if [ -z "$LANG" ]       # ... If not english, ignore
(for now) ....
then    TMP_LANG=""
else    TMP_LANG="$LANG"; unset LANG
fi

PATH=/bin:/usr/bin:/etc:/usr/sbin:/usr/bsd:.:$PATH

export PATH TMP_LANG

#
------------------------------------------------------
---------------
# Write an error message to the error log
#
------------------------------------------------------
---------------

Write2ErrorLog()
{
if [ -z "$STDERR_LOG" ]
then
        if [ -d "$ASDIR/logfiles" ]
        then
STDERR_LOG="$ASDIR/logfiles/start_asiq.log"
        else    STDERR_LOG="/tmp/start_asiq.log"
        fi
fi

NOW=`date "+%m/%d %H:%M:%S"`

echo "
-----------
start_asiq:\n
E. $NOW. $1
" >> $STDERR_LOG

echo "Error written to: $STDERR_LOG\n"
}

#
------------------------------------------------------
---------------
# Display an error message and exit
#
------------------------------------------------------
---------------

Error()
{
case "$1" in

    ERR_ASDIR)         MSG="Unable to find valid \$ASIQ
directory"
                       ;;

    ERR_BAD_START)     MSG="Parameter '-STARTDIR'
cannot be used with '-ud'"
                       ;;

    ERR_CREATEFILE)    MSG="Cannot create or write to
the file:\n\t$2"
                       ;;

    ERR_ENVDIR)                MSG="$2 is not set to a
valid directory ($3)"
                       ;;

    ERR_MISS_FILE)     MSG="Cannot find required file
'$2'"
                       ;;

    ERR_SHMEM)         MSG="Shared memory requested
(-iqsmem) exceeds o/s maximum ($2)"
                       ;;

    ERR_STARTDIR)      MSG="Invalid directory/parameter
-STARTDIR ($2)"
                       ;;

    ERR_SYBASE)                MSG="Unable to find
valid \$SYBASE directory"
                       ;;

    ERR_SYBASE_OCS)    MSG="\$SYBASE_OCS is set
incorrectly ($SYBASE_OCS)"
                       ;;

    ERR_WRITEDIR)      MSG="Cannot write to $2
directory ($3)."
                       ;;

    ERR_CHDIR)         MSG="Cannot CD to the directory
($2)."
                       ;;

    *)                 MSG="Failed to print error. $1
is missing from error list"
                       ;;
esac

echo "\007 \nError:\t${MSG}\n\tUnable to continue.\n"

Write2ErrorLog "$MSG"

rm -rf $TMP_FILE
exit 1
}

#
------------------------------------------------------
---------------
# Display a warning to the stderr log (if exists)
#
------------------------------------------------------
---------------

Warning()
{
case "$1" in

    WARN_MEMORY)       MSG="
        Ambiguous memory cache sizes detected.
        A server should be started using either the
'-c' parameter
        or the ( '-cl' and '-ch' ) parameters.
Combining conflicting
        memory parameters in the same startup line
could cause
        unexpected results."
```

```
                        ;;

    WARN_LOGS)          MSG="
        log directory '$ASLOGDIR' is filling.
        This server alone has $TOTAL_NUM_LOGS log
files.
        Please clean out soon!"
                        ;;

    *)                  MSG="Failed to print warning. $1
is missing from warning list"
                        ;;
esac

if [ -f "$STDERR_LOG" ]
then    echo "\nWarning:  $MSG" >> "$STDERR_LOG"
else    echo "\nWarning:  $MSG"
fi

sleep 2
}

#
-----------------------------------------------------
----------------
# Return 0 if a directory is wriable
#
-----------------------------------------------------
----------------

IsDirWriteable()
{
if [ -d "$1" ]
then
        if touch      "$1/.wri_tst.$$" 2>/dev/null
        then
                rm -f "$1/.wri_tst.$$" && return 0
        fi
fi

return 1
}

#
-----------------------------------------------------
----------------
# Extract all parameter from the cmd line, @file and
@env_var and
# highlight parameters that we are looking for:  xSVR,
xDB, xPORT
# to get -x tcpip{;}/(;) to work, set all non-
alphanumerics to "="
#
-----------------------------------------------------
----------------

Show_All_Parms()
{
eval set -- `echo . $* |sed 's/[{}();]/=/g'` # Remove
"quotes", clean comms param
shift

while [ -n "$1" ]
do
        case "$1" in
                NULL)   NULL=""
                        ;;

                -n)     echo xSVR $2
                        shift
                        ;;

                -o)     echo xLOG $2
                        shift
                        ;;

                -x*)    FOUND_PORT=0;

                        for PORT_PARM in `echo $1 $2|
sed 's/=/ /g'`
                        do
                                if [ $FOUND_PORT = 1 ]
                                then
                                        echo xPORT
$PORT_PARM
                                        break
                                fi

                                [ "$PORT_PARM" = port ]
&& FOUND_PORT=1;
                        done

                        [ ".$1" = ".-x" ] && shift    #
only shift if it's value follows.
                        ;;

                -*)     echo "$1"

                        case "$2" in
                                all)
echo "$2"; shift;;     # Follow on "all"
                                [1-9])
echo "$2"; shift;;     # Follow on 0-9
                                [1-9][0-9])
echo "$2"; shift;;     # Follow on 10-99
                                [1-9][0-9][0-9])
echo "$2"; shift;;     # Follow on 100-999
                                [1-9][0-9][0-9][0-9])
echo "$2"; shift;;     # Follow on 1000-9999
                        esac
                        ;;

                @*)     AT_PARM=`echo $1 | cut -c2-160`

                        if [ -f "$AT_PARM" ]
                        then
                                Show_All_Parms `grep -v
'^#' $AT_PARM`
                        else
                                eval VARS=$`echo
${AT_PARM}`

                                Show_All_Parms $VARS
                        fi
                        ;;

                *)      if [ -s "$1" -o -s "$1.db" ]
                        then
                                DB_FILE=`echo $1 | sed
's/\.db$//g'`
                                echo xDB `basename
"$DB_FILE"`
                        else
                                echo "$1";
                        fi
                        ;;
        esac

        [ $# -ge 1 ] && shift
done
}

#
-----------------------------------------------------
----------------
# Extract from the file default.cfg all parameters not
listed by user
#
-----------------------------------------------------
----------------

Get_Default_Parms()
{
DEFAULT_FILE="$ASDIR/scripts/default.cfg"

[ ! -f $DEFAULT_FILE ] && Error "ERR_MISS_FILE"
"$DEFAULT_FILE"

# ... Write an awk script out to a file for extracting
params ...

echo '
BEGIN {
        default_file="'$DEFAULT_FILE'";
        def_name[0]=""; def_value[0]=""; ndef=0;
def_pos=0;
        usr_name[0]=""; usr_value[0]=""; nusr=0;
usr_pos=0;
      }
#
-----------------------------------------------------
------
```

```
# Find Parameter in either array

function ParmExists( which_arry, search_parm )
{
if ( which_arry == "usr" )
        {
        for (usr_pos=0; usr_pos<nusr; usr_pos++)
                {
                if ( usr_name[usr_pos] == search_parm )
return(1);
                }
        }
else    {
        for (def_pos=0; def_pos<ndef; def_pos++)
                {
                if ( def_name[def_pos] == search_parm )
return(1);
                }
        }

return(0);
}
#
------------------------------------------------------
------
# If parameter does not exist, add to default arry

function AddDefaultParm( parm_n, parm_v )
{
if ( ! ParmExists( "usr", parm_n ) )
        {
        if ( ! ParmExists( "def", parm_n ) )
                {
                def_name[ ndef] = parm_n;
                def_value[ndef] = parm_v;
                ndef++;
                }
        }
}
#
------------------------------------------------------
------
# Put the users parameters into name/value arrays

function GetUsrParm()
{
tot_usr_parm = split( $0, usr_parms );

for (i=1; i<tot_usr_parm; )
        {
        if ( substr( usr_parms[i], 1, 1 ) == "-" )
                {
                usr_name[nusr] = usr_parms[i++];
                }

        if ( substr( usr_parms[i], 1, 1 ) != "-" )
                {
                usr_value[nusr] = usr_parms[i++];
                }

        nusr++;
        }
}
#
------------------------------------------------------
------
# Put paramers in default.cfg into the name/value
arrays

function GetDefParm()
{
while ( getline < default_file )
        {
        if ( substr( $1, 1, 1 ) != "-" ) continue

        def_name[ndef]  = $1;

        if ( index( $2, "\"" ) < 1 )
                {
                def_value[ndef] = $2
                }
        else    {
                split( $0, inp, "\"" );
                def_value[ndef] = sprintf( "\"%s\"",
inp[2] );
                }
        ndef++;
        }
}
#
------------------------------------------------------
------
# Add OS, Calculated, and Missing paramteters

function FindMissingParm()
{
# ... Remove -c if -cl or -ch found ..............

if ( ParmExists( "usr", "-cl" ) || ParmExists( "usr",
"-ch" ))
        {
        if ( ParmExists( "def", "-c" ) )
                {
                def_name[ def_pos] = "";
                def_value[def_pos] = "";
                }
        }

# ... Resolve -gm & -gn settings ..................

if ( ParmExists( "usr", "-gm" ) )
        gm_val = usr_value[usr_pos];
else    gm_val = 10;

gn_val = gm_val + 5;

if ( gn_val < 25 ) gn_val = 25;

AddDefaultParm( "-gm", gm_val );
AddDefaultParm( "-gn", gn_val );

# ... Add parms for Special OS .....................

if ( "'$OS_TYPE'" == "AIX" )
        {
        AddDefaultParm( "-iqsmem", "512" );
        AddDefaultParm( "-iqmt",   "450" );
        }

if ( "'$OS_TYPE'" == "HP-UX" )
        {
        AddDefaultParm( "-iqsmem", "500" );
        }
}
#
------------------------------------------------------
------
# Print parameters in default, not in users list

function PrintMissingParm()
{
for (d=0; d<ndef; d++)
        {
        if ( ! ParmExists( "usr", def_name[d] ) )
                {
                if ( def_name[ d] > " " )     printf
( "%s ", def_name[ d] );
                if ( def_value[d] > " " )     printf
( "%s ", def_value[d] );
                }
        }
}
#
------------------------------------------------------
------
# AWK MAIN

{
GetUsrParm();
GetDefParm();
FindMissingParm();
PrintMissingParm();
print ""
}
' > $TMP_FILE

case "$OS_TYPE" in
        Linux*|HP-UX*)
                AWK=awk ;;      # HP-UX & Linux don't
have nawk
        *)      AWK=nawk;;
```

```
esac

# ... Pasing in the users parms, return missing
default parms ...

DEF_PARM_LIST=`echo $* | $AWK -f $TMP_FILE`
export DEF_PARM_LIST
}


#
==================================================
================
#                                MAIN
#
==================================================
================

if [ $# = 0 ]
then
        echo '
Usage: start_asiq [server switches] {database
[switches]}*
                [ '-?' for a full list of parameters
] \n'
        exit 1
fi

# ... tmp file for
starting ..................................

TMP_FILE=/tmp/.start_asiq.$$.tmp      ; export TMP_FILE

# ... Set server parameters to
null ..........................

SERVER_NAME="";
SERVER_LOG="";
STDERR_LOG="";
DB_NAME="";
PORT_NUM="(default)";          # default port number
2638.

export SERVER_NAME SERVER_LOG STDERR_LOG DB_NAME
PORT_NUM

# ... Where do I start the
server ........................

if [ ".$1" = ".-STARTDIR" ]
then
        [ ! -d "$2" ] && Error "ERR_STARTDIR" "$2"

        cd "$2" || Error "ERR_STARTDIR" "$2"

        shift 2

        if expr "$* " : '.*-ud ' >/dev/null
        then
                Error "ERR_BAD_START"
        fi
fi

[ "`pwd`" = "$ASDIR/bin" ] && cd ..   # Don't start in
bin dir

START_DIR=`pwd`

# --- Make sure $SYBASE and $SYBASE_OCS are set
---------------

[ -z "$SYBASE" ]  &&  Error "ERR_ENVDIR" "\$SYBASE"
"$SYBASE"

[ ! -d "$SYBASE/locales" ]  &&  Error "ERR_SYBASE"

for TRY_DIR in "$SYBASE_OCS" "OCS-12_5" "OCS-12_0"
Not_Found
do
        if [ -d "$SYBASE/$TRY_DIR/lib" ]
        then
                SYBASE_OCS="$TRY_DIR" ; export
SYBASE_OCS
                break
        fi
done

[ "$TRY_DIR" = "Not_Found" ] && Error "ERR_SYBASE_OCS"

# --- Find the ASIQ directory
--------------------------------

for TRY_DIR in "$ASDIR" `dirname $0`/.. "$SYBASE/ASIQ-
12_5" . Not_Found
do
        [ -d "$TRY_DIR/lib" -a \
          -d "$TRY_DIR/res" -a \
          -x "$TRY_DIR/bin/asiqsrv12" ] && break;
done

[ "$TRY_DIR" = "Not_Found" ] && Error ERR_ASDIR
"\$ASDIR"

cd "$TRY_DIR"  || Error ERR_CHDIR "$TRY_DIR"

ASDIR=`pwd`    ; export ASDIR

cd "$START_DIR"        || Error ERR_CHDIR "$START_DIR"

# --- Clean out temp log file if exists
---------------------------

rm -rf $ASDIR/logfiles/start_asiq.log

# --- Check the current environment
-------------------------------

if [ -s $XWAY_ENV/switch_cc -a -s $XWAY_ENV/CC_setup ]
then    IN_HOUSE="Y"
else    IN_HOUSE=""
fi

if [ -n "$ASTMP" ]
then
        if [ ! -d "$ASTMP" ]
        then
                Error ERR_ENVDIR "\$ASTMP" "$ASTMP"
        fi

        IsDirWriteable "$ASTMP" || Error ERR_WRITEDIR
"\$ASTMP" "$ASTMP"
fi

if [ -z "$ASLOGDIR" ]
then
        ASLOGDIR=$ASDIR/logfiles      ; export ASLOGDIR

        mkdir  -p $ASLOGDIR 2>&-
        chmod 777 $ASLOGDIR 2>&-
fi

[ ! -d "$ASLOGDIR" ] && Error ERR_ENVDIR "\$ASLOGDIR"
"$ASLOGDIR"

IsDirWriteable "$ASLOGDIR" || Error ERR_WRITEDIR
"\$ASLOGDIR" "$ASLOGDIR"

# --- For what O/S is this software built
-------------------

OS_TYPE=`uname -s`
OS_NAME=`uname -n | cut -f1 -d'.'`
OS_REV=`uname -r`

case "$OS_TYPE" in
        AIX*)   if file $ASDIR/lib/libiq*1 |grep 64-bit
>/dev/null
                then
                        OS_TYPE=AIX64
                fi
                ;;

        HP-UX)  if file $ASDIR/lib/libiq*1 |grep ELF-64
>/dev/null
                then
                        OS_TYPE=HP-UX64
                fi
                ;;

        SunOS)  if file $ASDIR/lib/libiq*1 |grep
SPARCV9 >/dev/null
                then
```

```
                OS_TYPE=SunOS64
        fi
        ;;
esac

# ... Setup LIB
Paths .....................................

IQ_LIB="$ASDIR/usrlib:$ASDIR/lib:$SYBASE/$SYBASE_OCS/l
ib"

case "$OS_TYPE" in

    AIX)        LIBPATH="$IQ_LIB:$LIBPATH"
                YIELDLOOPTIME=40
                SPINLOOPTIME=40

                export LIBPATH YIELDLOOPTIME
SPINLOOPTIME
                ;;

    AIX64)      LIBPATH="$IQ_LIB:$LIBPATH"
                YIELDLOOPTIME=40
                SPINLOOPTIME=40

                AIXTHREAD_SCOPE="S"
                AIXTHREAD_MNRATIO="4:1"

                export LIBPATH YIELDLOOPTIME
SPINLOOPTIME
                export AIXTHREAD_SCOPE
AIXTHREAD_MNRATIO
                ;;

    HP-UX)      SHLIB_PATH="$IQ_LIB:$SHLIB_PATH"

                export SHLIB_PATH
                ;;

    HP-UX64)    SHLIB_PATH="$IQ_LIB:$SHLIB_PATH"

                if [ "$OS_REV" = "B.11.11" ]  #
CR273296 - enable thread patch
                then
                        export
PTHREAD_DISABLE_HANDOFF="ON"
                fi

                export SHLIB_PATH
PTHREAD_DISABLE_HANDOFF
                ;;

    Linux*)     LD_ASSUME_KERNEL=2.4.1        # Server
uses 2.4 threads
                export LD_ASSUME_KERNEL
                ;;

    SunOS)      if [ "$OS_REV" = "5.8" ]      # .. Fix
for Solaris 2.8. connect
                then
                        IQ_LIB="$IQ_LIB:/usr/lib/lwp/32"
                fi
                ;;

    SunOS64)    if [ "$OS_REV" = "5.8" ]      # .. Fix
for Solaris 2.8. connect
                then
                        IQ_LIB="$IQ_LIB:/usr/lib/lwp/64"
                fi


SYS_LIB="/usr/lib/sparcv9:/usr/ucblib/sparcv9"

                LD_LIBRARY_PATH_64="${IQ_LIB}:
${SYS_LIB}:$LD_LIBRARY_PATH_64"

                export LD_LIBRARY_PATH_64
                ;;
esac

LD_LIBRARY_PATH="$IQ_LIB:$LD_LIBRARY_PATH"

PATH="$ASDIR/bin:$PATH"

export LD_LIBRARY_PATH PATH

unset IQ_LIB

# ... Show only help or
version? ..........................

for RUN_IT in $* "no_run_parm"
do
        case "$RUN_IT" in
                -?|-help)       continue;;
                -v*)            continue;;
                 *)             break;;
        esac
done

if [ "no_run_parm" = "$RUN_IT" ]
then
        if [ -n "$TMP_LANG" ]  # retore orignal
language
        then
                LANG="$TMP_LANG"        ; export TMP_LANG
        fi

        echo ""
        $ASDIR/bin/asiqsrv12 $* 2>&1 | sed
's/asiqsrv12/start_asiq/g'
        echo ""
        exit 1
fi

sync &

# ... Expand user's parameter to include all hidden
parameters .....

USR_PARM_LIST=""
USR_ORIG_LIST="$*"

while [ -n "$1" ]
do
        USR_PARM_LIST="$USR_PARM_LIST '$1'"
        shift
done

set -- `Show_All_Parms $USR_PARM_LIST`

Get_Default_Parms $*

# ... Verify parameters being passed
in ......................

case "$OS_TYPE" in
        AIX)            MAX_IQSMEM=512          ;;
        HP-UX)          MAX_IQSMEM=2750
        ;;
        *)              MAX_IQSMEM=9999999      ;;
esac

while [ -n "$1" ]       # Get Startup Info, Check for
Errors
do
        case "$1" in

            -iqsmem)    if test "$2" -le $MAX_IQSMEM
2>&-
                        then
                                shift 2; continue;
                        fi

                        Error ERR_SHMEM $MAX_IQSMEM
                        ;;

                        # ... Extract Special
Info .............

                xDB)    [ -z "$DB_NAME"  ] &&
DB_NAME="$2"
                        shift
                        ;;

                xLOG)   SERVER_LOG="$2"
                        shift
                        ;;

                xSVR)   [ -z "$SERVER_NAME" ] &&
SERVER_NAME="$2"
                        shift
```

```
                        ;;

                xPORT)  PORT_NUM="$2"
                        shift
                        ;;
        esac

        [ $# -ge 1 ] && shift
done

# ... Ok, put it all
together ..................................

# If the environmental variable "IQ_NO_DEFAULTS" is
set to "YES"
# then the default paramerers will not be added,
explicit parm only.

if [ "$IQ_NO_DEFAULTS" = "YES" ]
then
        DEF_PARM_LIST="(Removed by user)"
        FULL_PARM_LIST="$USR_PARM_LIST"
        # Final List - Defaults Removed
else
        FULL_PARM_LIST="$USR_PARM_LIST $DEF_PARM_LIST"
        # Final list of all parameters
fi

[ -z "$DB_NAME"               ] && DB_NAME="none"
[ -z "$SERVER_NAME"    ] && SERVER_NAME="$DB_NAME"

# --- Setup the log file (increment since
last)-------------------

TOTAL_NUM_LOGS=0

if [ -z "$SERVER_LOG" ]               #CR271539 fix "."
in $ASLOGDIR
then
        set -- `cd $ASLOGDIR ; ls -1t $SERVER_NAME.[0-
9]*[0-9].srvlog 2>/dev/null |
                cut -f2 '-d.'` 0

        TOTAL_NUM_LOGS=$#

        NUM=${1:-1}

        while :
        do
                NUM=`expr \( $NUM + 1 \) % 1000`
                NUM=`printf "%03d" $NUM`


SERVER_LOG="$ASLOGDIR/$SERVER_NAME.$NUM.srvlog"

STDERR_LOG="$ASLOGDIR/$SERVER_NAME.$NUM.stderr"

                [ ! -f $SERVER_LOG ] && break
        done

        LOG_PARM="-o $SERVER_LOG"
else
        # ... Make a stderr file based on the logfile
name ..............

        LOG_PARM=""

        STDERR_LOG=`echo $SERVER_LOG |
                    sed
's?\....$??g;s?\.svrlog$??g;s?\.stdout$??g'`

        STDERR_LOG=$STDERR_LOG.stderr
fi

export SERVER_NAME SERVER_LOG STDERR_LOG LOG_PARM

# --- Start tails on stdout and stderr files
----------------------

if touch $SERVER_LOG
then
        echo "" >>     $SERVER_LOG    # loggings from
the server
        tail -1f       $SERVER_LOG &

        LOG_PID=$!
else
        Error ERR_CREATEFILE "$SERVER_LOG"
fi

if touch $STDERR_LOG
then
        echo "" >>     $STDERR_LOG    # stdout and
stderr from the server
        tail -1f       $STDERR_LOG &  # required for HP

        ERR_PID=$!
else
        kill $LOG_PID
        Error ERR_CREATEFILE "$STDERR_LOG"
fi

RET_CODE="1"   ; export RET_CODE

trap 'rm -rf $TMP_FILE; kill $LOG_PID $ERR_PID 2>&- ;
exit $RET_CODE' 0 1 2 3

# --- Display Startup Message
-------------------------------------

cat >> $STDERR_LOG <<-!
        Starting server $SERVER_NAME on $OS_NAME at
port $PORT_NUM
        !

# --- Find IQ & OC versions (can take a few seconds)
-------------

OC_VER=`$SYBASE/$SYBASE_OCS/bin/bcp -v 2>/dev/null |
grep '/1' |cut '-f2,3' '-d/'`

IQ_VER=`strings -a $ASDIR/lib/libiq*1 2>/dev/null |
grep "IDENT=" |cut -f2,5 -d'/'`

# --- Display rest of startup info
------------------------------

cat >> $STDERR_LOG <<-!

        Run Directory      : $START_DIR
        Server Exe         : $ASDIR/bin/asiqsrv12
        Server Output Log  : $SERVER_LOG
        Server Version     : ${IQ_VER:-N/A}
        OpenClient Version : ${OC_VER:-N/A}
        User Parameters    : $USR_ORIG_LIST
        Default Parameters : $DEF_PARM_LIST

        !

# ... Issue any warnings to stderr
file ......................

if [ "$TOTAL_NUM_LOGS" -gt 500 ]
then
        Warning "WARN_LOGS"
fi

# ... If -cl used in conjunction with -c, issue
warning ..........

if expr "$USR_PARM_LIST $DEF_PARM_LIST " : ".*-c " > /
dev/null
then
        if  expr "$USR_PARM_LIST $DEF_PARM_LIST " :
".*-cl " > /dev/null
        then
                Warning "WARN_MEMORY"
        fi
fi

# --- Create Scripts to start server
-----------------------------

PRINT_ARG2="awk '{print \$2}'"        ; export
PRINT_ARG2

echo    '
        # This will remove/set soft limits before
starting server
        # This c shell allows the usage of the
"unlimit" command
```

```
        set OS_HW=`uname -sm`    # Itanium and unlimit
do not mix

        if ( "$OS_HW" != "HP-UX ia64" ) then
                ( unlimit >&! /dev/null ) && unlimit
        endif

        # ... limit
discriptors ............................

        if ( '$OS_TYPE' != AIX && '$OS_TYPE' != AIX64 )
then

                set NFILE=`limit descriptors |
'${PRINT_ARG2}'`
                set NFILE_MAX=4096

                if ( $NFILE == unlimited )  set
NFILE=2100100
                        if ( $NFILE > $NFILE_MAX )
limit descriptors $NFILE_MAX
        endif

        # ... limit coredump to 64 blocks
(CR263753) .......

        # Any core file size is limited to 32k unless
end-user sets
        # env var IQ_UNLIMIT_CORE to YES before
starting server

        if ( "'$IQ_UNLIMIT_CORE'" == "YES" ) then

                echo    "NOTICE: Limits on core file
size have been removed"
                echo    ""
        else
                set CDUMP=`limit coredumpsize |
'${PRINT_ARG2}'`
                set CDUMP_MAX=32

                if ( $CDUMP == unlimited )  set
CDUMP=2100100
                if ( $CDUMP > $CDUMP_MAX )  limit
coredump $CDUMP_MAX
        endif

        # ... limit stack
size ............................

        set STACK=`limit stacksize | '${PRINT_ARG2}'`
        set STACK_MAX=8192

        if ( $STACK == unlimited )  set STACK=2100100
        if ( $STACK > $STACK_MAX )  limit stacksize
$STACK_MAX

        # ... Reset the language and start
server .........

        if ( "$TMP_LANG" != "" )      setenv LANG
"$TMP_LANG"

        dbspawn -v -f -p asiqsrv12 '$FULL_PARM_LIST
$LOG_PARM' >>&! '$STDERR_LOG'

        exit $status
        ' > $TMP_FILE

sleep 1         # Need a sec to display stdout & stderr

csh -f $TMP_FILE  ;  RET_CODE=$?

sleep 1         # Need a sec to display stdout & stderr

# ... How did we do ................

if [ $RET_CODE = 0 ]
then
        # started okay!
        # Get the process id and other server info, and
put them
        # in the server info for for use with stop_asiq

        START_ASIQ_FILE=/tmp/.start_asiq.log

        if [ ! -f $START_ASIQ_FILE ]
        then
                touch     $START_ASIQ_FILE
                chmod 666 $START_ASIQ_FILE
        fi

        set -- `grep -i "process id" $STDERR_LOG`
        ## kt: Language Watch

        if [ $# -gt 1 ]
        then
                shift `expr $# - 1`

                echo    $1                              \
                        STARTED:`date "+%y/%m/%d.%T"`   \
                        SVR:$SERVER_NAME                \
                        DB:$DB_NAME                     \
                        PORT:$PORT_NUM
>> $START_ASIQ_FILE
        fi

        MSG="Server started successfully"
else
        MSG="Server failed to start."
        ERR="Possible cause:"

        set -- `grep " ERROR:" $STDERR_LOG` ""

        case "$3" in
                  -82)   MSG="$MSG $ERR\n\t- Server name
is already in use on network."
                         ;;
                  -85)   MSG="$MSG $ERR\n\t- Port number
is invalid or already in use."
                         ;;
                 -300)   MSG="$MSG $ERR
        - Invalid or conflicting parameter or database
name
        - Port number already in use
        - Server name already in use on network"
                         ;;
                 *)      ERR=""
                         ;;
        esac
fi

echo "\n$MSG\n"

rm -rf $TMP_FILE

exit $RET_CODE

# --- eof ---
```

==========

gen_streams_new.ksh

==========

```
#!/bin/ksh

if (( $# < 2 ))
then
     echo "usage:  $0    seed  scale_factor
num_streams"
     exit
fi


PATH=/export/home/sybase/ASIQ-
12_5/bin:/export/home/sybase/OCS-
12_5/bin:/usr/openwin/bin:/bin:.:/usr/dist/pkgs/forte_
dev/SUNWspro/bin:/usr/ccs/bin:/usr/dt/bin:/usr/dist/pk
gs/devpro,v4.0/5.x-
sparc/bin:/usr/dist/local/exe:/usr/dist/exe:/usr/ucb:/
usr/sbin:/net/josie/export/home18/rgostan/bin:/export/
home/sybase/run/scripts:/etc:.:/export/home/sybase/run
/tpch/appendix/dbgen
export PATH
export DSS_PATH=/export/home/sybase/run/scripts;
export
DSS_CONFIG=/export/home/sybase/run/tpch/appendix/dbgen
;
export DSS_DIST=dists.dss;
export
DSS_QUERY=/export/home/sybase/run/tpch/appendix/templa
```

```
tes/queries;
#export
DSS_QUERY=/export/home/sybase/run/tpch/appendix/templa
tes/queries.debug;

seed=$1;
sf=$2;
ns=$3

i=0

while ((i<=ns))
do
   qgen -c -p $i -l qparm${i}.txt -i
$DSS_QUERY/init.sql -t $DSS_QUERY/complete.sql -r
$seed -s $sf  \
      1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
20 21 22 > stream${i}.sql
   ((seed=seed+1))
   ((i=i+1))
done

((last_seed=seed-1))
echo $last_seed
```

==========

ACID Test Execution Code

==========

==========

atomicity test

==========

```
dbtest acid_atomic_main.tst > acid_atomic_main.out
```

==========

consistency test

==========

```
dbtest acid_consistency_main.tst >
acid_consistency_main.out
```

==========

isolation tests

==========

```
dbtest acid_isolation_main1.tst >
acid_isolation_main1.out
dbtest acid_isolation_main2.tst >
acid_isolation_main2.out
dbtest acid_isolation_main3.tst >
acid_isolation_main3.out
dbtest acid_isolation_main4.tst >
acid_isolation_main4.out
dbtest acid_isolation_main5.tst >
acid_isolation_main5.out
dbtest acid_isolation_main6.tst >
acid_isolation_main6.out
```

==========

durability test

==========

```
dbtest acid_durability_main.tst >
acid_durability_main.out
```

==========

ACID Test Source Code

==========

==========

acid_atomic_main.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print 'Atomicity test start = ', times
print ' '
include 'acid_functions.tst'
commit

%
% Atomicity test with rollback
%
print ' '
print 'Starting atomicity test with rollback'
print ' '

run test 'acid_atomic_setup.tst'

stringconnect "dsn=tpch;"
let counter=0

LOOP {
open cur2 {select ordr, line, delta from aa_whattodo
where seqnum=^}
       substitute counter
print 'counter = ',counter
fetch cur2 into ordr, line, delta
if ROWSTATUS != FOUND then { BREAK LOOP } endif
print 'Acid transaction for: o_key-',ordr,'  l_key-',
line,'  delta-',delta

execute {select o_totalprice, l_quantity,
l_extendedprice
       from orders, lineitem
       where o_orderkey = l_orderkey and o_orderkey =^
and l_linenumber = ^}
       substitute ordr, line
       into o_total, l_quan, l_price
print   'o_totalprice = ',o_total,'    l_quantity =
',l_quan,
         '       l_extendedprice = ',l_price

execute {call acid_transaction(^, ^,  ^, rprice,
quantity,
                                tax, disc, extprice,
ototal)
       } substitute ordr, line, delta
close cur2
let counter = counter+1

rollback
execute {select now(*)} into times
print 'rollback : ', times

execute {select o_totalprice, l_quantity,
l_extendedprice
       from orders, lineitem
       where o_orderkey = l_orderkey and o_orderkey =^
and l_linenumber = ^}
       substitute ordr, line
       into o_total, l_quan, l_price
print   'o_totalprice = ',o_total,'    l_quantity =
',l_quan,
         '       l_extendedprice = ',l_price
print ' '

} ENDLOOP

commit

%
% Atomicity test with commit
%
stringconnect "dsn=tpch;"
print ' '
print 'Starting atomicity test with commit '
print ' '
run test 'acid_atomic_setup.tst'

stringconnect "dsn=tpch;"

open cur1 {select ordr, line, delta from aa_whattodo}
LOOP {
fetch cur1 into ordr, line, delta
if ROWSTATUS != FOUND then { BREAK LOOP } endif
print 'Acid transaction for: o_key-',ordr,'  l_key-',
line,'  delta-',delta
execute {select o_totalprice, l_quantity,
l_extendedprice
       from orders, lineitem
       where o_orderkey = l_orderkey and o_orderkey =^
and l_linenumber = ^}
```

```
        substitute ordr, line
        into o_total, l_quan, l_price
print   'o_totalprice = ',o_total,'    l_quantity =
',l_quan,
        '       l_extendedprice = ',l_price

execute {call acid_transaction(^, ^,  ^, rprice,
quantity,
                               tax, disc, extprice,
ototal)
        } substitute ordr, line, delta
commit
execute {select now(*)} into times
print 'commit : ', times

execute {select o_totalprice, l_quantity,
l_extendedprice
        from orders, lineitem
        where o_orderkey = l_orderkey and o_orderkey =^
and l_linenumber = ^}
        substitute ordr, line
        into o_total, l_quan, l_price
print   'o_totalprice = ',o_total,'    l_quantity =
',l_quan,
        '       l_extendedprice = ',l_price
print ' '

} ENDLOOP

close cur1
commit

execute {select now(*)} into times
print 'Atomicity test end = ', times

End Test
```

==========

acid_atomic_setup.tst

==========

```
Description    "Creates aa_whattodo table"

stringconnect "dsn=tpch;"

% Drop Table if found

allow error -141
execute { commit }
execute { drop table aa_whattodo }
allow no error

execute {
create table aa_whattodo (
                seqnum       int      not null,
                ordr         int        not null,
                line         int        null,
                delta        int         null)
}

print 'aa_whattodo CREATED!!'
execute {select now(*)} into times
print 'time = ', times

fetch {select count(*) from aa_whattodo } into ROWS
assert ROWS = 0

print 'Number of rows before load: ',ROWS

LOOP ({let counter = 0}; {counter < 5}; {let counter =
counter + 1})
{
        execute {call generate_acid_values()}
                into orderkey, linenumber,delta
        execute {insert into aa_whattodo values ( ^ , ^
, ^ , ^ ) }
                substitute counter, orderkey,
linenumber, delta
 print  counter, ' ',orderkey, ' ',linenumber,' ',
delta
}
ENDLOOP

commit

fetch {select count(*) from aa_whattodo } into ROWS
assert ROWS = 5

print 'Number of rows after load: ',ROWS

disconnect

End Test
```

==========

acid_consistency_main.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print 'Consistency test start = ', times
print ' '

include 'acid_functions.tst'

run test 'acid_consistency_setup.tst'

execute {select now(*)} into times
print 'Consistency test time = ', times
print ' '

run test '-o' 'acid_consistency_q1.ot'
'acid_consistency_query.tst'
disconnect

start test '-o' 'acid_consist_user1.ot' 'stream=1'
'acid_consistency_txn.tst'
sleep 1000
start test '-o' 'acid_consist_user2.ot' 'stream=2'
'acid_consistency_txn.tst'
sleep 1000
start test '-o' 'acid_consist_user3.ot' 'stream=3'
'acid_consistency_txn.tst'
sleep 1000
start test '-o' 'acid_consist_user4.ot' 'stream=4'
'acid_consistency_txn.tst'
sleep 1000
start test '-o' 'acid_consist_user5.ot' 'stream=5'
'acid_consistency_txn.tst'
sleep 1000
start test '-o' 'acid_consist_user6.ot' 'stream=6'
'acid_consistency_txn.tst'
sleep 1000
start test '-o' 'acid_consist_user7.ot' 'stream=7'
'acid_consistency_txn.tst'
sleep 1000
start test '-o' 'acid_consist_user8.ot' 'stream=8'
'acid_consistency_txn.tst'
sleep 1000
start test '-o' 'acid_consist_user9.ot' 'stream=9'
'acid_consistency_txn.tst'

synchronize 10
% let the log flush...
sleep 10000

stringconnect "dsn=tpch;"
run test '-o' 'acid_consistency_q2.ot'
'acid_consistency_query.tst'

execute {select now(*)} into times
print 'Consistency test end = ', times
print ' '

End Test
```

==========

acid_consistency_query.tst

==========

```
stringconnect "dsn=tpch;"

open cur1 {select stream, seqnum, ordr, line, delta
from acid_table
                where seqnum > 10 order by seqnum}
print ' '

let n=1
```

```
LOOP {
   fetch cur1 into str, seq, ord, lin, delta

   fetch {select round(cast(o_totalprice as numeric
(26,16)),2)
                from orders where o_orderkey=^ }
         substitute ord into o_price

   if ROWSTATUS != FOUND then { BREAK LOOP } endif
   if n > 25 then { BREAK LOOP } endif

   fetch { call acid_single_query (^) } substitute ord
into l_total

   fetch {select cast(^ as numeric(12,2)) } substitute
o_price into o_price
   fetch {select cast(^ as numeric(12,2)) } substitute
l_total into l_total

   print 'orderkey = ', ord, '        o_totalprice = ',
o_price,
         '       acid query = ' , l_total

   ASSERT (o_price = l_total)
        then { print 'Did not compare correctly' }
ENDASSERT
   let n=n+1
 } ENDLOOP

disconnect

END Test
```

acid_consistency_setup.tst

```
stringconnect "dsn=tpch;"

% Drop Table if found
allow error -141
execute { drop table acid_table }
allow no error

execute {
create table acid_table (
                stream  int     not null,
                 seqnum  int    not null,
                 ordr    int    not null,
                 line    int    null,
                 delta   int    null)
}

execute {checkpoint}

print 'acid_table CREATED!!'

fetch {select count(*) from acid_table } into ROWS
assert ROWS = 0
print 'Number of rows before load: ',ROWS
commit

LOOP ({let i = 1}; {i <= 9}; { let i = i + 1})
{
  LOOP ({let j = 1}; {j <= 100}; {let j = j + 1})
    {
       execute { call generate_acid_values()} into
ordr, line, delta
       execute { insert into acid_table values
(^,^,^,^,^) }
                substitute i,j,ordr,line,delta
    } endloop
  print (j-1)*i
} endloop

commit

fetch {select count(*) from acid_table } into ROWS
assert ROWS = 900
print 'Number of rows after load: ',ROWS

End Test
```

acid_consistency_txn.tst

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print 'Consistancy test start = ', times
print ' '

LOOP ({let i = 1}; {i <= 100}; { let i = i + 1})
{
 fetch {select ordr, line, delta from acid_table
                where stream=^ and seqnum=^ }
        substitute stream, i
 if ROWSTATUS != FOUND then { print 'not enough rows'
                BREAK LOOP }
 endif

 print 'Acid Trenasaction ',i,
        ':  o_key-', ordr , '  l_key-', line , '
delta-' ,delta

 execute {call acid_transaction( ^, ^,  ^, rprice,
quantity,
                                tax, disc, extprice,
ototal)
 } substitute ordr, line, delta
 commit
 print 'committed'
 sleep 1000

}
ENDLOOP

synchronize 10

End Test
```

acid_durability_main.tst

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print 'Durability test start = ', times
print ' '

include 'acid_functions.tst'
run test 'acid_durability_setup.tst'

execute {select now(*)} into times
print 'Durability test time = ', times
print ' '

run test  '-o' 'acid_durability_q1.ot'
'acid_durability_query.tst'

start test '-o' 'acid_dura_user1.ot' 'stream=1'
'acid_durability_txn.tst'
sleep 1000
start test '-o' 'acid_dura_user2.ot' 'stream=2'
'acid_durability_txn.tst'
sleep 1000
start test '-o' 'acid_dura_user3.ot' 'stream=3'
'acid_durability_txn.tst'
sleep 1000
start test '-o' 'acid_dura_user4.ot' 'stream=4'
'acid_durability_txn.tst'
sleep 1000
start test '-o' 'acid_dura_user5.ot' 'stream=5'
'acid_durability_txn.tst'
sleep 1000
start test '-o' 'acid_dura_user6.ot' 'stream=6'
'acid_durability_txn.tst'
sleep 1000
start test '-o' 'acid_dura_user7.ot' 'stream=7'
'acid_durability_txn.tst'
sleep 1000
start test '-o' 'acid_dura_user8.ot' 'stream=8'
'acid_durability_txn.tst'
sleep 1000
start test '-o' 'acid_dura_user9.ot' 'stream=9'
'acid_durability_txn.tst'
```

```
synchronize 10

execute {select now(*)} into times
print 'Durability test time = ', times
print ' '


run test  '-o' 'acid_durability_q2.ot'
'acid_durability_query.tst'

execute {select now(*)} into times
print 'Durability test end = ', times
print ' '

End Test
```

==========

acid_durability_query.tst

==========

```
stringconnect "dsn=tpch;"

open cur1 {select stream, seqnum, ordr, line, delta
from acid_table
                where seqnum > 5 order by seqnum}
print ' '

let n=1
LOOP {
   fetch cur1 into str, seq, ord, lin, delta

   fetch {select round(cast(o_totalprice as numeric
(26,16)),2)
                from orders where o_orderkey=^ }
         substitute ord into o_price

   if ROWSTATUS != FOUND then { BREAK LOOP } endif
   if n > 50 then { BREAK LOOP } endif

   fetch { call acid_single_query (^) } substitute ord
into l_total

   fetch {select cast(^ as numeric(12,2)) } substitute
o_price into o_price
   fetch {select cast(^ as numeric(12,2)) } substitute
l_total into l_total

   print 'orderkey = ', ord, '        o_totalprice = ',
o_price,
         '       acid query = ' , l_total

   ASSERT (o_price = l_total)
        then { print 'Did not compare correctly' }
ENDASSERT
   let n=n+1

 } ENDLOOP

disconnect

END Test
```

==========

acid_durability_setup.tst

==========

```
stringconnect "dsn=tpch;"

% Drop Table if found
allow error -141
execute { drop table acid_table }
allow no error

execute {
create table acid_table (
                stream  int     not null,
                 seqnum  int    not null,
                 ordr    int    not null,
                 line    int    null,
                 delta   int    null)
}

execute {checkpoint}

print 'acid_table CREATED!!'

fetch {select count(*) from acid_table } into ROWS
assert ROWS = 0
print 'Number of rows before load: ',ROWS
commit

LOOP ({let i = 1}; {i <= 9}; { let i = i + 1})
{
  LOOP ({let j = 1}; {j <= 200}; { let j = j + 1})
     {
        execute { call generate_acid_values()} into
ordr, line, delta
        execute { insert into acid_table values
(^,^,^,^,^) }
                substitute i,j,ordr,line,delta
     } endloop
  print (j-1)*i
} endloop

commit
execute {checkpoint}

fetch {select count(*) from acid_table } into ROWS

print 'Number of rows after load: ',ROWS

End Test
```

==========

acid_durability_txn.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print 'Durability test start = ', times
print ' '
print 'stream  trans.  o_key   l_key   p_key   s_key
delta   date_t '

LOOP ({let i = 1}; {i <= 200}; { let i = i + 1})
{
 fetch {select ordr, line, delta from acid_table
        where stream=^ and seqnum=^ }
        substitute stream, i

 if ROWSTATUS != FOUND then { print 'not enough rows'
BREAK LOOP }
 endif

 execute {select l_partkey, l_suppkey from lineitem
                where l_orderkey=^ and l_linenumber=^}
        substitute ordr, line
        into p_key, s_key

 execute {call acid_transaction( ^, ^,  ^)
         } substitute ordr, line, delta
        into  rprice, quantity, tax, disc, extprice,
ototal

 assert SQLCODE=0 then { DIE } endassert
 commit

 execute {select now(*)} into times
 print  stream,'    ',
        'txn ',i, '     ',
        ordr,  '       ',
        line,  '       ',
        p_key, '       ',
        s_key, '       ',
        delta, '       ',
        times, '       '
 sleep 1000
}
ENDLOOP

synchronize 10

End Test
```

==========

acid_functions.tst

```
=====================================================

print 'creating the sleep procedure'

allow error -265
execute { DROP PROCEDURE dbo.sleep}
allow no error

execute{ create procedure dbo.sleep(in sleep_time
integer default null)
begin
  declare command varchar(255);
  select 'xp_cmdshell ''sleep '+str(sleep_time)+''''
into command;
  execute immediate command
end;
}

print 'creating the Acid Transaction'

allow error -265
execute { DROP PROCEDURE acid_transaction }
allow no error

execute{ CREATE PROCEDURE acid_transaction(
                IN  o_key      INT,
                IN  l_key      INT,
                IN  delta      INT,
                OUT rprice     Numeric(18,8),
                OUT quantity   INT,
                OUT tax        Numeric(18,8),
                OUT disc       Numeric(18,8),
                OUT extprice   Numeric(18,8),
                OUT ototal     Numeric(18,8)
                )
ON EXCEPTION RESUME
BEGIN
 DECLARE pkey           INT ;
 DECLARE skey           INT ;
 DECLARE cost           NUMERIC(18,8) ;
 DECLARE new_extprice   NUMERIC(18,8) ;
 DECLARE new_ototal     NUMERIC(18,8) ;
 DECLARE new_quantity   INT ;
 DECLARE c_sqlstate     char(5);
 LOOP1: LOOP
    COMMIT;
    SELECT o_totalprice
      INTO ototal
      FROM orders
     WHERE o_orderkey = o_key ;
    SELECT l_quantity,
           l_extendedprice,
           l_partkey,
           l_suppkey,
           l_tax,
           l_discount
      INTO quantity,
           extprice,
           pkey,
           skey,
           tax,
           disc
      FROM lineitem
     WHERE l_orderkey   = o_key
       AND l_linenumber = l_key;
    -- CLEAN UP IMPRECICE NUMBERS
    SET ototal = ototal -
"TRUNCATE"("truncate"(extprice*(1-disc),2)*(1+tax),2);
    SET rprice = "TRUNCATE"((extprice  / quantity),2);
    SET cost = "TRUNCATE"((rprice * delta),2);
    SET new_extprice = extprice + cost;
    SET new_ototal = "TRUNCATE"(new_extprice * (1.0 -
disc),2);
    SET new_ototal = "TRUNCATE"(new_ototal * (1.0 +
tax),2);
    SET new_ototal = ototal + new_ototal ;
    SET new_quantity = quantity + delta ;
    --
    -- Update LineItem
    --
    UPDATE lineitem
       SET l_quantity      = new_quantity,
           l_extendedprice = new_extprice
     WHERE l_orderkey=o_key
       AND l_linenumber=l_key;
    SELECT SQLSTATE INTO c_sqlstate;
    IF c_sqlstate = '00000' THEN
        --
        -- Update Orders
        --
        UPDATE orders
           SET o_totalprice = new_ototal
         WHERE o_orderkey = o_key;
        SELECT SQLSTATE INTO c_sqlstate;
        IF c_sqlstate = '00000'  THEN
            INSERT INTO history VALUES ( pkey, skey,
o_key, l_key, delta, now(*)) ;
            SELECT SQLSTATE INTO c_sqlstate;
            IF c_sqlstate = '00000'  THEN
                LEAVE LOOP1;
            END IF;
        END IF;
    END IF;
 END LOOP LOOP1;
 RETURN(0);
END;
}

print 'Acid transaction created'
print ' '

print 'Creating Acid query'

allow error -265
execute { DROP PROCEDURE acid_single_query }
allow no error

execute{
CREATE PROCEDURE acid_single_query( IN o_key INT, OUT
o_total NUMERIC(26,16) )
BEGIN
    SELECT o_total =
        sum ("truncate" ("truncate"(
                          round(cast(l_extendedprice as
numeric(26,16)),2) *
                          (1 - round(cast(l_discount as
numeric(26,16)),2)),2)
                    * (1 + round(cast(l_tax as numeric
(26,16)),2)) , 2))
    FROM lineitem WHERE l_orderkey = o_key;
 END
}

print 'Acid query created'
print ' '

print 'Creating Generate_acid_values function'

allow error -265
execute { DROP PROCEDURE generate_acid_values }
allow no error

execute{
create procedure generate_acid_values(
out orderkey   int,
out linenumber int,
out delta      int)

BEGIN

 declare seed          bigint;
 declare rand_dbl      double precision;
 declare rand_int      int;
 declare out_key       int;

 declare times cursor for select datediff
(millisecond,convert(char(10),getdate(), 116),now(*));
 declare random1 cursor for select rand(seed);
 declare random cursor for select rand();
 declare get_order cursor for
       select o_orderkey from orders where o_orderkey
= rand_int;
 declare get_linenumber cursor for
       select max(l_linenumber) from lineitem
       where l_orderkey = orderkey;

 open times;
 fetch next times into seed;
 open random1;
 fetch next random1 into rand_dbl;
```

```
 set out_key = 0;
 loop1:
 while out_key = 0 LOOP
        open random;
        open get_order;

        fetch next random into rand_dbl;
        set rand_int = rand_dbl * 6001215 +1;
        fetch next get_order into out_key;

        close random;
        close get_order;
 end loop loop1;

 set orderkey = out_key;

 open get_linenumber;
 fetch next get_linenumber into linenumber;
 close get_linenumber;

 open random;
 fetch next random into rand_dbl;
 set delta = rand_dbl * 100 + 1;
 close random;

END
}
commit
execute {checkpoint}
print 'Generate_acid_values function created'
print ' '

print 'Creating Generate_Ps_Values function'

allow error -265
execute { DROP PROCEDURE generate_ps_values }
allow no error

execute{
create procedure generate_ps_values(
out partkey    int,
out suppkey    int)

BEGIN

 declare seed          bigint;
 declare rand_dbl      double precision;
 declare rand_int      int;
 declare out_key       int;
 declare counter       int;

 declare times cursor for select datediff
(millisecond,convert(char(10),getdate(), 116),now(*));
 declare random1 cursor for select rand(seed);
 declare random cursor for select rand();
 declare get_supp cursor for
        select ps_suppkey from partsupp
        where ps_suppkey = rand_int;
 declare get_part cursor for
        select ps_partkey from partsupp
        where ps_suppkey = suppkey;

 open times;
 fetch next times into seed;
 open random1;
 fetch next random1 into rand_dbl;
 close random1;

 set out_key = 0;
 while out_key = 0 LOOP
        open random;
        open get_supp ;

        fetch next random into rand_dbl;
        set rand_int = rand_dbl * 10000 +1;
        fetch next get_supp into out_key;

        close random;
        close get_supp ;
 end loop;
 set suppkey = out_key;

 set out_key = 0;
 set counter  = 0;
 open random;
 open get_part;
 fetch next random into rand_dbl;
 set rand_int = rand_dbl * 10 +1;

 loop1:
 while counter < rand_int LOOP
        set counter = counter+1;
        fetch next get_part into out_key;
 end loop loop1;

 set partkey = out_key;
 close random;
 close get_part;

END
}
commit
execute {checkpoint}
print 'Generate_Ps_Values function created'
print ' '

print 'Creating history table'

allow error -141
execute { drop table history }
allow no error

execute {
create table history (
   h_p_key    unsigned INT NOT NULL ,
   h_s_key    unsigned INT NOT NULL ,
   h_o_key    unsigned INT NOT NULL ,
   h_l_key    INT NOT NULL,
   h_delta    INT NOT NULL,
   h_date_t   TIMESTAMP NOT NULL)
}

commit
execute {checkpoint}
print 'history table created'
print ' '
```

==========

acid_isolation_main1.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print ' '
print ' '
print 'Isolation test 1'
print 'start = ', times
print ' '

include 'acid_functions.tst'
include 'acid_isolation_setup.tst'

start test  'acid_isolation_test1.tst'
start test  'acid_isolation_test1_query.tst'

End Test
```

==========

acid_isolation_main2.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print ' '
print ' '
print 'Isolation test 2'
print 'start = ', times
print ' '


include 'acid_functions.tst'
include 'acid_isolation_setup.tst'

start test  'acid_isolation_test2.tst'
start test  'acid_isolation_test2_query.tst'

End Test
```

==========

acid_isolation_main3.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print ' '
print ' '
print 'Isolation test 3'
print 'start = ', times
print ' '
print 'Isolation test start = ', times

include "acid_functions.tst"
include 'acid_isolation_setup.tst'

start test  'acid_isolation_test3_transaction1.tst'
start test  'acid_isolation_test3_transaction2.tst'

End Test
```

==========

acid_isolation_main4.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print ' '
print ' '
print 'Isolation test 4'
print 'start = ', times
print ' '
print 'Isolation test start = ', times

include 'acid_functions.tst'
include 'acid_isolation_setup.tst'

start test  'acid_isolation_test4_transaction1.tst'
start test  'acid_isolation_test4_transaction2.tst'

End Test
```

==========

acid_isolation_main5.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print ' '
print ' '
print 'Isolation test 5'
print 'start = ', times
print ' '

include 'acid_functions.tst'
include 'acid_isolation_setup.tst'

start test 'acid_isolation_test5_transaction1.tst'
start test 'acid_isolation_test5_query.tst'

End Test
```

==========

acid_isolation_main6.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print ' '
print ' '
print 'Isolation test 6'
print 'start = ', times
print ' '

include 'acid_functions.tst'
include 'acid_isolation_setup.tst'

start test '-u' 'acid_isolation_test6_query.tst'
start test  'acid_isolation_test6_transaction1.tst'

End Test
```

==========

acid_isolation_setup.tst

==========

```
stringconnect "dsn=tpch;"

% Drop Table if found

allow error -141
execute { commit }
execute { drop table acid_isolation_table }
allow no error

execute {
create table acid_isolation_table (
                ordr         int    not null,
                line         int    null,
                delta        int    null)
}

execute {checkpoint}

print 'acid_isolation_table CREATED!!'
execute {select now(*)} into times
print 'time = ', times

fetch {select count(*) from acid_isolation_table }
into ROWS
assert ROWS = 0

print 'Number of rows before load: ',ROWS

execute {call generate_acid_values()} into orderkey,
linenumber,delta
execute {insert into acid_isolation_table values ( ^ ,
^ , ^ ) }
                substitute orderkey, linenumber, delta
print  orderkey, ' ',linenumber,' ', delta

commit

fetch {select count(*) from acid_isolation_table }
into ROWS
assert ROWS = 1

print 'Number of rows after load: ',ROWS

disconnect

End Test
```

==========

acid_isolation_test1.tst

==========

```
stringconnect "dsn=tpch;"

execute {select ordr, line, delta from
acid_isolation_table}
        into ordr, line, delta

 print 'The following are the data input values for
the ACID Transaction.'
 print '(user 1) o_key-',ordr, '  l_key-', line, '
delta-',delta

 execute {call acid_transaction( ^, ^,  ^,
       rprice, quantity, tax, disc, extprice, ototal)
       } substitute ordr, line, delta

execute {select now(*)} into times
print 'User 1 waiting to commit = ', times
print ' '
synchronize 2
sleep 10000
execute {select now(*)} into times
print 'User 1 about to commit = ', times
commit

execute { select round(cast(o_totalprice as numeric
```

```
(18,2)),2)
       from orders where o_orderkey = ^}
       substitute ordr into o_total
print 'User 1 new values: '
print 'user 1 ordr= ', ordr
print 'user 1 o_total= ', o_total
print ' '

End Test
```

==========

acid_isolation_test1_query.tst

==========

```
stringconnect "dsn=tpch;"

 synchronize 2
 print ' '
 execute {select now(*)} into times
 print 'User 2 start query = ', times

 execute {select ordr from acid_isolation_table}
        into ordr

 print 'user 2 ordr = ', ordr
 fetch { call acid_single_query (^) } substitute ordr
into o_total
 print 'user 2 o_total=' , o_total
 print ' '

execute {select now(*)} into times
print 'User 2 completed query = ', times

disconnect

END Test
```

==========

acid_isolation_test2.tst

==========

```
stringconnect "dsn=tpch;"

execute {select ordr, line, delta from
acid_isolation_table}
        into ordr, line, delta

 print 'The following are the data input values for
the ACID Transaction.'
 print '(user 1) o_key-',ordr, '  l_key-', line, '
delta-',delta

 execute {call acid_transaction( ^, ^,  ^,
       rprice, quantity, tax, disc, extprice, ototal)
       } substitute ordr, line, delta

execute {select now(*)} into times
print 'User 1 waiting to roll back = ', times
print ' '
synchronize 2
sleep 10000
execute {select now(*)} into times
print 'User 1 about to roll back = ', times
rollback

execute { select round(cast(o_totalprice as numeric
(18,2)),2)
       from orders where o_orderkey = ^}
       substitute ordr into o_total
print 'User 1 new values: '
print 'user 1 ordr= ', ordr
print 'user 1 o_total= ', o_total
print ' '

End Test
```

==========

acid_isolation_test2_query.tst

==========

```
stringconnect "dsn=tpch;"

 synchronize 2
 print ' '
 execute {select now(*)} into times
 print 'User 2 start query = ', times

 execute {select ordr from acid_isolation_table}
        into ordr

 print 'user 2 ordr = ', ordr
 fetch { call acid_single_query (^) } substitute ordr
into o_total
 print 'user 2 o_total=' , o_total
 print ' '

execute {select now(*)} into times
print 'User 2 completed query = ', times

disconnect

END Test
```

==========

acid_isolation_test3_transaction1.tst

==========

```
stringconnect "dsn=tpch;"

execute {select now(*)} into times
print 'Isolation test 3 test start = ', times
print ' '

 execute {select ordr, line, delta from
acid_isolation_table}
        into ordr, line, delta

 print 'User 1 --  The input data values for User 1
Acid Transaction.'
 print 'User 1 -- o_key = ',ordr
 print 'User 1 -- l_key = ',line
 print 'User 1 -- delta1 = ',delta

 print ' '
 execute {select now(*)} into times
 print 'User 1 -- Starting the Acid Transaction: ',
times

 execute {call acid_transaction( ^, ^,  ^ )}
        substitute ordr, line, delta
        into rprice, quantity, tax, disc, extprice,
ototal

 print ' '
 execute {select now(*)} into times
 print 'User 1 -- Acid Transaction complete: ', times
 print '30 second timer started'
SYNCHRONIZE 2
 sleep 30000

 print ' '
 execute {select now(*)} into times
 print 'User 1 -- starting commit: ', times

 commit
 print ' '
 execute {select now(*)} into times
 print 'User 1 -- transaction commit complete: ',
times

 print ' '
 print 'USER 1 -- original extendedprice = ', extprice
 print 'USER 1 -- original quantity = ', quantity

 fetch { select cast(^ as numeric(18,6))
              + (cast(^ as numeric(18,6))*(cast (^ as
numeric(18,6))
              /cast (^ as numeric(18,6)))) }
        substitute extprice, delta, extprice, quantity
        into result1
 % make it format nicely...
 execute { select cast(^ as numeric(18,2)) }
substitute result1 into result2

 print ' '
 print 'User 1 -- result1 = '
 print '    txn1_extendedprice + (delta1 *
(txn1_extendedprice/txn1_quantity))'
 print 'User 1 -- result1= ', result2
```

```
 print ' '

 disconnect
 End Test
```

```
==================================================
```

acid_isolation_test3_transaction2.tst

```
==================================================

stringconnect "dsn=tpch;"

execute {select ordr, line, delta from
acid_isolation_table}
        into ordr, line, delta
% generate a new set of values; we only use delta2
execute { call generate_acid_values()} into ordr2,
line2, delta2

 print ' '
 print 'User 2 - The input data values for the Acid
Transaction.'
 print 'User 2 -- o_key = ',ordr
 print 'User 2 -- l_key= ',line
 print 'User 2 -- delta2 = ',delta2

SYNCHRONIZE 2
sleep 5000

 print ' '
 execute {select now(*)} into times
 print 'User 2 -- Starting the Acid Transaction: ',
times

execute {call acid_transaction( ^, ^,  ^ ) }
       substitute ordr, line, delta2
       into rprice, quantity, tax, disc, extprice,
ototal
execute {select round(cast(^ as numeric(20,6)),2) }
       substitute extprice into extprice2

sleep 5000
print ' '
execute {select now(*)} into times
print 'User 2 -- About to commit: ', times
commit

execute {select now(*)} into times
print 'User 2 -- transaction commit complete: ', times

print ' '

print 'USER 2 -- original extendedprice = ', extprice2
print 'USER 2 -- original quantity = ', quantity
print ' '

End Test
```

```
==================================================
```

acid_isolation_test4_transaction1.tst

```
==================================================

stringconnect "dsn=tpch;"

execute {select now(*)} into times
print 'Isolation test 3 test start = ', times
print ' '

 execute {select ordr, line, delta from
acid_isolation_table}
        into ordr, line, delta

 print 'User 1 --  The input data values for User 1
Acid Transaction.'
 print 'User 1 -- o_key = ',ordr
 print 'User 1 -- l_key = ',line
 print 'User 1 -- delta1 = ',delta

 print ' '
 execute {select now(*)} into times
 print 'User 1 -- Starting the Acid Transaction: ',
times

 execute {call acid_transaction( ^, ^,  ^ )}
       substitute ordr, line, delta
       into rprice, quantity, tax, disc, extprice,
ototal

 print ' '
 execute {select now(*)} into times
 print 'User 1 -- Acid Transaction complete: ', times
 print '30 second timer started'
SYNCHRONIZE 2
 sleep 30000

 print ' '
 execute {select now(*)} into times
 print 'User 1 -- starting rollback: ', times

 rollback
 print ' '
 execute {select now(*)} into times
 print 'User 1 -- transaction rollback complete: ',
times

 execute {select round(cast(^ as numeric(20,6)),2) }
       substitute extprice into extprice2
 print ' '
 print 'USER 1 -- original extendedprice = ',
extprice2
 print 'USER 1 -- original quantity = ', quantity
 print ' '

 disconnect
 End Test
```

```
==================================================
```

acid_isolation_test4_transaction2.tst

```
==================================================

stringconnect "dsn=tpch;"

execute {select ordr, line, delta from
acid_isolation_table}
        into ordr, line, delta
% generate a new set of values; we only use delta2
execute { call generate_acid_values()} into ordr2,
line2, delta2

 print ' '
 print 'User 2 - The input data values for the Acid
Transaction.'
 print 'User 2 -- o_key = ',ordr
 print 'User 2 -- l_key= ',line
 print 'User 2 -- delta2 = ',delta2

SYNCHRONIZE 2
sleep 5000

 print ' '
 execute {select now(*)} into times
 print 'User 2 -- Starting the Acid Transaction: ',
times

execute {call acid_transaction( ^, ^,  ^ ) }
       substitute ordr, line, delta2
       into rprice, quantity, tax, disc, extprice,
ototal
execute {select round(cast(^ as numeric(20,6)),2) }
       substitute extprice into extprice2

sleep 5000
print ' '
execute {select now(*)} into times
print 'User 2 -- About to commit: ', times
commit

execute {select now(*)} into times
print 'User 2 -- transaction commit complete: ', times
print ' '
print 'USER 2 -- original extendedprice = ', extprice2
print 'USER 2 -- original quantity = ', quantity
print ' '

End Test
```

```
==================================================
```

acid_isolation_test5_query.tst

```
==================================================
```

```
stringconnect "dsn=tpch;"

synchronize 2

execute { call generate_ps_values() } into ps_ptky,
ps_spky
print ' '
print 'user 2 ps_partkey = ', ps_ptky
print 'user 2 ps_suppkey = ', ps_spky
print ' '

execute {select now(*)} into times
print 'User 2 beginning query = ', times
execute {select * from partsupp where ps_partkey=^ and
ps_suppkey=^}
       substitute ps_ptky, ps_spky
       into ps_ptky, ps_spky, ps_aly, ps_spct, ps_ct

print ' '
print 'User2 gets all columns of the PARTSUPP table '
print ' for selected ps_partkey and ps_suppkey doing a
query.'
print ' '
print 'ps_partkey = ', ps_ptky, '     ps_suppkey = ',
ps_spky
print 'ps_availqty = ', ps_aly, '     ps_supplycost =
',ps_spct
print 'ps_comment = ', ps_ct
execute {select now(*)} into times
print 'User 2 query complete = ', times
print ' '

execute {select now(*)} into times
print 'User 2 about to commit = ', times
commit
execute {select now(*)} into times
print 'User 2 transaction commit complete = ', times

print ' '

End Test
```

==========

acid_isolation_test5_transaction1.tst

==========

```
stringconnect "dsn=tpch;"

execute {select ordr, line, delta from
acid_isolation_table}
       into ordr, line, delta

 print ' '
 print 'The following are the input values for the
users1 ACID Transaction.'
 print 'o_key = ',ordr,'      l_key = ',line,'
delta = ',delta
 print ''
execute {select now(*)} into times
print 'User 1 isolation test time = ', times
print ' '
print ' '
execute {select o_totalprice from orders where
o_orderkey=^ }
       substitute ordr into o_tprice
execute {select l_extendedprice, l_quantity,l_partkey,
l_suppkey
       from lineitem
       where l_orderkey=^ and l_linenumber=^}
       substitute ordr, line
       into l_price, l_quant, l_ptky, l_spky
print 'User 1 o_totalprice = ', o_tprice
print 'User 1 l_extendedprice = ', l_price,'
l_quantity = ', l_quant
print 'User 1 l_partkey       = ', l_ptky,'   l_suppkey
= ', l_spky
print ' '

execute {select now(*)} into times
print 'User 1 starting acid transaction = ', times

execute {call acid_transaction( ^, ^,  ^, rprice,
quantity, tax, disc,
       extprice, ototal) } substitute ordr, line,
delta

execute {select now(*)} into times
print 'User 1 waiting to commit = ', times
print ' '
synchronize 2
sleep 10000
execute {select now(*)} into times
print 'User 1 about to commit = ', times
commit
execute {select now(*)} into times
print 'User 1 transaction commit complete = ', times

execute {select o_totalprice from orders where
o_orderkey=^ }
       substitute ordr into o_tprice
execute {select l_extendedprice, l_quantity
       from lineitem where l_orderkey=^ and
l_linenumber=^}
       substitute ordr, line
       into l_price, l_quant
print 'User 1 o_totalprice = ', o_tprice
print 'User 1 l_extendedprice = ', l_price,'
l_quantity = ', l_quant
print 'User 1 l_partkey       = ', l_ptky,'   l_suppkey
= ', l_spky

print ' '
execute {select * from history where h_o_key=^
       and h_date_t=(select max(h_date_t) from history
where h_o_key=^)}
       substitute ordr, ordr
       into hpk, hsk, hok, hlk, hda, hdt

print 'User 1 history entry:'
print '  h_p_key = ', hpk
print '  h_s_key = ', hsk
print '  h_o_key = ', hok
print '  h_l_key = ', hlk
print '  h_delta = ', hda
print '  h_date_t = ', hdt

execute {select now(*)} into times
print 'User 1 isolation test time = ', times
print ''

End Test
```

==========

acid_isolation_test6_query.tst

==========

```
stringconnect "dsn=tpch;"

print 'User1 Query: '
print ' '
print 'User1 starts its query (Q1) here.'
execute {select now(*)} into qstart
print 'Start time for User1 Q1 =', qstart
print ' '
compare fetchall {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 <= dateadd(day, -1, '1998-12-01')
group by l_returnflag,l_linestatus
order by l_returnflag,l_linestatus
 } in 'queryresult'

execute {select now(*)} into qstop
print 'Stop time for User1 Q1 =', qstop
print ' '

End Test
```

==========

acid_isolation_test6_transaction1.tst

==========

```
stringconnect "dsn=tpch;"

execute {select ordr, line, delta from
acid_isolation_table}
       into ordr, line, delta

 execute {select now(*)} into qstart2
 print 'User2 acid Transaction = ', qstart2
 print 'o_key = ',ordr, '      l_key = ',line, '
delta = ',delta
 print ' '
execute {select o_totalprice from orders where
o_orderkey=^ }
       substitute ordr into o_tprice
execute {select l_extendedprice, l_quantity,l_partkey,
l_suppkey
       from lineitem where l_orderkey=^ and
l_linenumber=^}
       substitute ordr, line
       into l_price, l_quant, l_ptky, l_spky
print 'User 2 o_totalprice = ', o_tprice
print 'User 2 l_extendedprice = ', l_price,'
l_quantity = ', l_quant
print 'User 2 l_partkey       = ', l_ptky,'
l_suppkey = ', l_spky
print ' '

execute {select now(*)} into qstart2
print 'Start Time for User2 Transaction = ', qstart2
print ' '
execute {call acid_transaction( ^, ^,  ^, rprice,
quantity,
                               tax, disc, extprice,
ototal) }
       substitute ordr, line, delta

execute {select now(*)} into qstop2
print 'User 2 about to commit = ', qstop2
commit
execute {select now(*)} into qstop2
print 'User 2 transaction commit complete = ', qstop2
print ' '

execute {select o_totalprice from orders where
o_orderkey=^ }
       substitute ordr
       into o_tprice
execute {select l_extendedprice, l_quantity
       from lineitem where l_orderkey=^ and
l_linenumber=^}
       substitute ordr, line
       into l_price, l_quant
print 'User 2 o_totalprice = ', o_tprice
print 'User 2 l_extendedprice = ', l_price,'
l_quantity = ', l_quant
print 'User 2 l_partkey       = ', l_ptky,'
l_suppkey = ', l_spky
print ' '

print ' '
execute {select * from history
       where h_o_key=^
       and h_date_t=(select max(h_date_t) from history
where h_o_key=^)}
       substitute ordr, ordr
       into hpk, hsk, hok, hlk, hda, hdt

print 'User 2 history entry:'
print '  h_p_key = ', hpk
print '  h_s_key = ', hsk
print '  h_o_key = ', hok
print '  h_l_key = ', hlk
print '  h_delta = ', hda
print '  h_date_t = ', hdt

print ' '
execute {select now(*)} into times
print 'User 2 completed = ', times

End Test
```

==========

Disk Configuration Details

==========

Solaris Volume Manager Setup

Perform the following steps to create the RAID devices:

1) create the following 15 MB slices
   - c1t1d0s7
   - c1t2d0s7
   - c1t8d0s7
   - c1t10d0s7
   - c1t11d0s7

using the Solaris format command

2) create the SVM state replicas via

```
metadb -c 2 c1t1d0s7  c1t2d0s7 c1t8d0s7 \
            c1t10d0s7 c1t11d0s7
```

3) create 68.35 GB s0 slices on each of the following disks:
   - c2t8d0
   - c2t9d0
   - c2t10d0
   - c2t11d0
   - c2t12d0
   - c2t13d0
   - c3t8d0
   - c3t9d0
   - c3t10d0
   - c3t11d0
   - c3t12d0
   - c8t8d0
   - c8t9d0
   - c8t10d0
   - c8t11d0
   - c8t12d0
   - c8t13d0
   - c9t8d0
   - c9t9d0
   - c9t10d0
   - c9t11d0
   - c9t12d0

using the Solaris format command

4) create 2 raid5 volumes as follows

```
metainit d5 -r  c2t8d0   c2t9d0    \
                c2t10d0  c2t11d0   \
                c2t12d0  c2t13d0   \
                c3t8d0   c3t9d0    \
                c3t10d0  c3t11d0   \
                c3t12d0

metainit d5 -r  c8t8d0   c8t9d0    \
                c8t10d0  c8t11d0   \
                c8t12d0  c8t13d0   \
                c9t8d0   c9t9d0    \
                c9t10d0  c9t11d0   \
                c9t12d0
```

5) create the followin 2 GB slices:
   - c1t10d0s6
   - c1t11d0s6

using the Solaris format command

6) create the mirrored volume d20 via the following:

```
metainit d21 1 1 c1t10d0s6

metainit d22 1 1 c1t11d0s6

metainit d20 -m d21

metattach d20 d22
```

File System Setup

7) create a filesystem on /dev/md/rdsk/d20

```
8) mount the filesystem on /sybase2
```

Database Device Links

```
# Create the symbolic links:

ln -s /dev/md/rdsk/d5     /sybase2/M01
ln -s /dev/md/rdsk/d6     /sybase2/M02

ln -s /dev/rdsk/c1t9d0s0  /sybase2/RT01
ln -s /dev/rdsk/c1t1d0s0  /sybase2/RT02
ln -s /dev/rdsk/c1t2d0s0  /sybase2/RT03
ln -s /dev/rdsk/c1t8d0s0  /sybase2/RT04
ln -s /dev/rdsk/c1t10d0s0 /sybase2/RT05
ln -s /dev/rdsk/c1t11d0s0 /sybase2/RT06
ln -s /dev/rdsk/c3t13d0s0 /sybase2/RT07
ln -s /dev/rdsk/c9t13d0s0 /sybase2/RT08

ln -s /dev/rdsk/c1t1d0s1  /sybase2/WT01
ln -s /dev/rdsk/c1t2d0s1  /sybase2/WT02
ln -s /dev/rdsk/c1t8d0s1  /sybase2/WT03
ln -s /dev/rdsk/c1t10d0s1 /sybase2/WT04
ln -s /dev/rdsk/c1t11d0s1 /sybase2/WT05
ln -s /dev/rdsk/c3t13d0s1 /sybase2/WT06
ln -s /dev/rdsk/c9t13d0s1 /sybase2/WT07
```

Appendix C. Query Text and Query Output

```
====================================================
```

qualification query 1

```
====================================================

% 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 <= dateadd(day,-90,'1998-12-01')
% group by
% l_returnflag,
% l_linestatus
% order by
% l_returnflag,
% l_linestatus;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.32000 seconds - current
time 16:40:13
'A','F',37734107,56586554400.7292032,53758257134.86945
63,55909065222.8284717,25.5220058532573342,38273.12973
46211374,.0499852958383577168,1478493
'N','F',991417,1487504710.38000107,1413082168.05409968
,1469649223.19436967,25.5164719205229819,38284.4677608
483374,.0500934266742134809,38854
'N','O',74476040,111701729697.737336,106118230307.6073
83,110367043872.495174,25.5022267695849895,38249.11798
8907361,.049996586053555131,2920374
'R','F',37719753,56568041380.8983326,53741292684.60453
75,55889619119.8339581,25.5057936126907617,38250.85462
60985255,.0500094058300870121,1478870
% total of 4 rows written

====================================================
```

qualification query 2

```
====================================================

% select top 100
% 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;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.76000 seconds - current
time 16:40:25
9938.53,'Supplier#000005359       ','UNITED KINGDOM
',185358,'Manufacturer#4
','QKuHYh,vZGiwu2FWEJoLDx04','33-429-790-
6131','blithely silent pinto beans are furiously.
slyly final deposits acros'
9937.84,'Supplier#000005969       ','ROMANIA
',108438,'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,'Manufacturer#4           ','B3rqp0xbSEim4Mpy2RH
J','33-320-228-2957','blithely special packages are.
stealthily express deposits across the closely final
instructi'
9923.77000000000119,'Supplier#000002324
','GERMANY                 ',29821,'Manufacturer#4
','y3OD9UywSTOk','17-779-299-1839','quickly express
packages breach quiet pinto beans. requ'
9871.22,'Supplier#000006373       ','GERMANY
',43868,'Manufacturer#5           ','J8fcXWsTqM','17-
813-485-8637','never silent deposits integrate
furiously blit'
9870.78,'Supplier#000001286       ','GERMANY
',81285,'Manufacturer#2
','YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH','17-516-924-
4574','final theodolites cajole slyly special,'
9870.78,'Supplier#000001286       ','GERMANY
',181285,'Manufacturer#4
','YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH','17-516-924-
4574','final theodolites cajole slyly special,'
9852.52000000000119,'Supplier#000008973
','RUSSIA                  ',18972,'Manufacturer#2
','t5L67YdBYYH6o,Vz24jpDyQ9','32-188-594-
7038','quickly regular instructions wake-- carefully
unusual braids into the expres'
9847.83,'Supplier#000008097       ','RUSSIA
',130557,'Manufacturer#2
','xMe97bpE69NzdwLoX','32-375-640-3593','slyly regular
dependencies sleep slyly furiously express dep'
9847.57,'Supplier#000006345       ','FRANCE
',86344,'Manufacturer#1
','VSt3rzk3qG698u6ld8HhOByvrTcSTSvQlDQDag','16-886-
766-7945','silent pinto beans should have to snooze
carefully along the final reques'
% total of 100 rows written

====================================================
```

qualification query 3

```
====================================================

% select top 10
% 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 < '1995-03-15'
% and l_shipdate > '1995-03-15'
% group by
% l_orderkey,
% o_orderdate,
% o_shippriority
% order by
% revenue desc,
% o_orderdate;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.36000 seconds - current
time 16:40:27
2456423,406181.011100000024,'1995-03-05',0
3459808,405838.698899999917,'1995-03-04',0
492164,390324.061,'1995-02-19',0
1188320,384537.935899999976,'1995-03-09',0
2435712,378673.055799999952,'1995-02-26',0
4878020,378376.795200000048,'1995-03-12',0
5521732,375153.9215,'1995-03-13',0
2628192,373133.309399999976,'1995-02-22',0
993600,371407.45949999994,'1995-03-05',0
2300070,367371.145200000107,'1995-03-13',0
% total of 10 rows written
```

==========

qualification query 4

==========

```
% select
% o_orderpriority,
% count(*) as order_count
% from
% orders
% where
% o_orderdate >= '1993-07-01'
% and o_orderdate < dateadd(month,3,'1993-07-01')
% and exists (
% select
% *
% from
% lineitem
% where
% l_orderkey = o_orderkey
% and l_commitdate < l_receiptdate
% )
% group by
% o_orderpriority
% order by
% o_orderpriority;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.25000 seconds - current
time 16:40:31
'1-URGENT       ',10594
'2-HIGH         ',10476
'3-MEDIUM       ',10410
'4-NOT SPECIFIED',10556
'5-LOW          ',10487
% total of 5 rows written
```

==========

qualification query 5

==========

```
% 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 >= '1994-01-01'
% and o_orderdate < dateadd(year,1,'1994-01-01')
% group by
% n_name
% order by
% revenue desc;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.65000 seconds - current
time 16:40:36
'INDONESIA                ',55502041.1696999431
'VIETNAM                  ',55295086.9966999531
'CHINA                    ',53724494.2565999746
'INDIA                    ',52035512.000200057
'JAPAN                    ',45410175.6954000235
% total of 5 rows written
```

==========

qualification query 6

==========

```
% select
% sum(l_extendedprice * l_discount) as revenue
% from
% lineitem
% where
% l_shipdate >= '1994-01-01'
% and l_shipdate < dateadd(year,1,'1994-01-01')
% and l_discount between .06 - 0.01 and .06 + 0.01
% and l_quantity < 24;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.15000 seconds - current
time 16:40:41
123141078.228299007
% total of 1 rows written
```

==========

qualification query 7

==========

```
% 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,
% datepart(year, l_shipdate) 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 '1995-01-01' and '1996-12-31'
% ) as shipping
```

```
% group by
% supp_nation,
% cust_nation,
% l_year
% order by
% supp_nation,
% cust_nation,
% l_year;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.67000 seconds - current
time 16:40:43
'FRANCE                    ','GERMANY
',1995,54639732.7335999489
'FRANCE                    ','GERMANY
',1996,54633083.3075999737
'GERMANY                   ','FRANCE
',1995,52531746.6696999669
'GERMANY                   ','FRANCE
',1996,52520549.0223998487
% total of 4 rows written
```

==========

qualification query 8

==========

```
% select
% o_year,
% sum(case
% when nation = 'BRAZIL' then volume
% else 0
% end) / sum(volume) as mkt_share
% from
% (
% select
% datepart(year, o_orderdate) 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 '1995-01-01' and '1996-12-
31'
% and p_type = 'ECONOMY ANODIZED STEEL'
% ) as all_nations
% group by
% o_year
% order by
% o_year;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.86000 seconds - current
time 16:40:47
1995,.0344358904066548347
1996,.041485521293530345
% total of 2 rows written
```

==========

qualification query 9

==========

```
% select
% nation,
% o_year,
% sum(amount) as sum_profit
% from
% (
% select
% n_name as nation,
% datepart(year, o_orderdate) 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'
% ) as profit
% group by
% nation,
% o_year
% order by
% nation,
% o_year desc;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.69000 seconds - current
time 16:40:48
'ALGERIA                   ',1998,31342867.2345000029
'ALGERIA                   ',1997,57138193.0233001232
'ALGERIA                   ',1996,56140140.1330001235
'ALGERIA                   ',1995,53051469.6533999741
'ALGERIA                   ',1994,53867582.128600049
'ALGERIA                   ',1993,54942718.132400012
'ALGERIA                   ',1992,54628034.7126999021
'ARGENTINA                 ',1998,30211185.708099997
'ARGENTINA                 ',1997,50805741.75230003
'ARGENTINA                 ',1996,51923746.5754999459
% total of 175 rows written
```

==========

qualification query 10

==========

```
% select top 20
% 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 >= '1993-10-01'
% and o_orderdate < dateadd(month,3,'1993-10-01')
% 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;
```

```
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.50000 seconds - current
time 16:40:55
57040,'Customer#000057040',734235.2455,632.87,'JAPAN
','Eioyzjf4pp','22-895-641-3466','requests sleep
blithely about the furiously i'
143347,'Customer#000143347',721002.694799999952,2557.4
700000000003,'EGYPT
','1aReFYv,Kw4','14-742-935-3718','fluffily bold
excuses haggle finally after the u'
60838,'Customer#000060838',679127.307700000048,2454.77
,'BRAZIL
','64EaJ5vMAHWJlBOxJklpNc2RJiWE','12-913-494-
9813','furiously even pinto beans integrate under the
ruthless foxes; ironic, even dolphins across the slyl'
101998,'Customer#000101998',637029.566699999809,3790.8
9,'UNITED KINGDOM           ','01c9CILnNtfOQYmZj','33-
593-865-6378','accounts doze blithely! enticing, final
deposits sleep blithely special accounts. slyly
express accounts pla'
125341,'Customer#000125341',633508.086,4983.5100000000
006,'GERMANY
','S29ODD6bceU8QSuuEJznkNaK','17-582-695-
5962','quickly express requests wake quickly blithely'
25501,'Customer#000025501',620269.784899999976,7725.04
,'ETHIOPIA                  ','
W556MXuoiaYCCZamJI,Rn0B4ACUGdkQ8DZ','15-874-808-
6793','quickly special requests sleep evenly among the
special deposits. special deposi'
115831,'Customer#000115831',596423.867200000167,5098.1
,'FRANCE                    ','rFeBbEEyk dl
ne7zV5fDrmiq1oK09wV7pxqCgIc','16-715-386-
3788','carefully bold excuses sleep alongside of the
thinly idle'
84223,'Customer#000084223',594998.023899999976,528.65,
'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,'Customer#000054289',585603.391799999952,5583.02
,'IRAN                      ','vXCxoCsU0Bad5JQI ,
oobkZ','20-834-292-4707','express requests sublate
blithely regular requests. regular, even ideas solve.'
39922,'Customer#000039922',584878.113399999976,7321.10
999999999881,'GERMANY
','Zgy4s50l2GKN4pLDPBU8m342gIw6R','17-147-757-
8036','even pinto beans haggle. slyly bold accounts
inte'
6226,'Customer#000006226',576783.760599999905,2230.09,
'UNITED KINGDOM
','8gPu8,NPGkfyQQ0hcIYUGPIBWc,ybP5g,','33-657-701-
3391','quickly final requests against the regular
instructions wake blithely final instructions. pa'
922,'Customer#000000922',576767.533299999833,3869.25,'
GERMANY
','Az9RFaut7NkPnc5zSD2PwHgVwr4jRzq','17-945-916-
9648','boldly final requests cajole blith'
147946,'Customer#000147946',576455.132,2030.1300000000
003,'ALGERIA
','iANyZHjqhyy7Ajah0pTrYyhJ','10-886-956-
3143','furiously even accounts are blithely above the
furiousl'
115640,'Customer#000115640',569341.193299999952,6436.1
,'ARGENTINA                 ','Vtgfia9qI
7EpHgecU1X','11-411-543-4901','final instructions are
slyly according to the'
73606,'Customer#000073606',568656.857799999952,1785.67
,'JAPAN
','xuR0Tro5yChDfOCrjkd2ol','22-437-653-
6966','furiously bold orbits about the furiously busy
requests wake across the furiously quiet theodolites.
d'
110246,'Customer#000110246',566842.981499999881,7763.3
5,'VIETNAM                  ','7KzflgX MDOq7sOkI','31-
943-426-9837','dolphins sleep blithely among the slyly
final'
142549,'Customer#000142549',563537.236799999952,5085.9
899999999994,'INDONESIA
','ChqEoK43OysjdHbtKCp6dKqjNyvvi9','19-955-562-
2398','regular, unusual dependencies boost slyly;
ironic attainments nag fluffily into the unusual
packages?'
146149,'Customer#000146149',557254.9865,1791.55,'ROMAN
IA                  ','s87fvzFQpU','29-744-164-
6487','silent, unusual requests detect quickly slyly
regul'
52528,'Customer#000052528',556397.350899999976,551.79,
'ARGENTINA                  ','NFztyTOR10UOJ','11-208-
192-3205','unusual requests detect. slyly dogged
theodolites use slyly. deposit'
23431,'Customer#000023431',554269.536000000119,3381.86
,'ROMANIA
','HgiV0phqhaIa9aydNoIlb','29-915-458-
2654','instructions nag quickly. furiously bold
accounts cajol'
% total of 20 rows written
```

==========

qualification query 11

==========

```
% 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;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.49000 seconds - current
time 16:41:01
129760,17538456.8599999994
166726,16503353.9199999988
191287,16474801.9699999988
161758,16101755.5399999976
34452,15983844.7200000018
139035,15907078.3400000006
9403,15451755.6199999988
154358,15212937.8799999982
38823,15064802.8599999994
85606,15053957.150000003
% total of 1048 rows written
```

==========

qualification query 12

==========

```
% 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 >= '1994-01-01'
% and l_receiptdate < dateadd(year,1,'1994-01-01')
% group by
% l_shipmode
% order by
% l_shipmode;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.26000 seconds - current
time 16:41:03
'MAIL      ',6202,9324
'SHIP      ',6200,9262
% total of 2 rows written
```

qualification query 13

```
% select
% c_count,
% count(*) as custdist
% from
% (
% select
% c_custkey,
% count(o_orderkey)
% from
% customer left outer join orders on
% c_custkey = o_custkey
% and o_comment not like 'specialrequests'
% group by
% c_custkey
% ) as c_orders (c_custkey, c_count)
% group by
% c_count
% order by
% custdist desc,
% c_count desc;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.17000 seconds - current
time 16:41:06
0,50004
9,6641
10,6566
11,6058
8,5949
12,5553
13,4989
19,4748
7,4707
18,4625
% total of 42 rows written
```

qualification query 14

```
% 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 >= '1995-09-01'
% and l_shipdate < dateadd(month,1,'1995-09-01');
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.24000 seconds - current
time 16:41:19
16.3807786263955563
% total of 1 rows written
```

qualification query 15

```
 Executing command:
% create view revenue0 (supplier_no, total_revenue) as
% select
% l_suppkey,
% sum(l_extendedprice * (1 - l_discount))
% from
% lineitem
% where
% l_shipdate >= '1996-01-01'
% and l_shipdate < dateadd(month,3,'1996-01-01')
% group by
% l_suppkey;
% execution time 0.81000 seconds - current time
16:41:21

 Executing command:
%
% select
% s_suppkey,
% s_name,
% s_address,
% s_phone,
% total_revenue
% from
% supplier,
% revenue0
% where
% s_suppkey = supplier_no
% and total_revenue = (
% select
% max(total_revenue)
% from
% revenue0
% )
% order by
% s_suppkey;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.27000 seconds - current
time 16:41:21
8449,'Supplier#000008449
','Wp34zim9qYFbVctdW','20-469-856-
8873',1772627.20870000005
% total of 1 rows written
```

qualification query 16

```
% 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 'CustomerComplaints'
% )
% group by
% p_brand,
% p_type,
% p_size
% order by
% supplier_cnt desc,
% p_brand,
% p_type,
% p_size;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.27000 seconds - current
time 16:41:22
'Brand#41  ','MEDIUM BRUSHED TIN',3,28
'Brand#54  ','STANDARD BRUSHED COPPER',14,27
'Brand#11  ','STANDARD BRUSHED TIN',23,24
'Brand#11  ','STANDARD BURNISHED BRASS',36,24
'Brand#15  ','MEDIUM ANODIZED NICKEL',3,24
'Brand#15  ','SMALL ANODIZED BRASS',45,24
'Brand#15  ','SMALL BURNISHED NICKEL',19,24
'Brand#21  ','MEDIUM ANODIZED COPPER',3,24
'Brand#22  ','SMALL BRUSHED NICKEL',3,24
'Brand#22  ','SMALL BURNISHED BRASS',19,24

% total of 18314 rows written
```

==========

qualification query 17

==========

```
% 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
% );
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.22000 seconds - current
time 16:41:28
348406.054285713732
% total of 1 rows written
```

==========

qualification query 18

==========

```
% select top 100
% 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;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.34000 seconds - current
time 16:41:29
'Customer#000128120',128120,4722021,'1994-04-
07',544089.089999999881,323
'Customer#000144617',144617,3043270,'1997-02-
12',530604.43999999994,317
'Customer#000013940',13940,2232932,'1997-04-
13',522720.61,304
'Customer#000066790',66790,2199712,'1996-09-
30',515531.82,327
'Customer#000046435',46435,4745607,'1997-07-
03',508047.99,309
'Customer#000015272',15272,3883783,'1993-07-
28',500241.33,302
'Customer#000146608',146608,3342468,'1994-06-
12',499794.58,303
'Customer#000096103',96103,5984582,'1992-03-
16',494398.78999999994,312
'Customer#000024341',24341,1474818,'1992-11-
15',491348.26,302
'Customer#000137446',137446,5489475,'1997-05-
23',487763.25,311
% total of 57 rows written
```

==========

qualification query 19

==========

```
% 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'
% );
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.35000 seconds - current
time 16:41:46
3083843.05780000031
% total of 1 rows written
```

qualification query 20

```
% 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 >= '1994-01-01'
% and l_shipdate < dateadd(year,1,'1994-01-01')
% )
% )
% and s_nationkey = n_nationkey
% and n_name = 'CANADA'
% order by
% s_name;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.37000 seconds - current
time 16:41:51
'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'
% total of 204 rows written
```

qualification query 21

```
% select top 100
% s_name,
% count(*) as 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;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.54000 seconds - current
time 16:41:53
'Supplier#000002829       ',20
'Supplier#000005808       ',18
'Supplier#000000262       ',17
'Supplier#000000496       ',17
'Supplier#000002160       ',17
'Supplier#000002301       ',17
'Supplier#000002540       ',17
'Supplier#000003063       ',17
'Supplier#000005178       ',17
'Supplier#000008331       ',17
% total of 100 rows written
```

qualification query 22

```
% select
% cntrycode,
% count(*) as numcust,
% sum(c_acctbal) as totacctbal
% from
% (
% select
% substring(c_phone,1,2) as cntrycode,
% c_acctbal
% from
% customer
% where
% substring(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 substring(c_phone,1,2) in
```

```
% ('13', '31', '23', '29', '30', '18', '17')
% )
% and not exists (
% select
% *
% from
% orders
% where
% o_custkey = c_custkey
% )
% ) as custsale
% group by
% cntrycode
% order by
% cntrycode;
% Estimated 1 rows in query (I/O estimate 1010)
% PLAN> vt_1 (seq)
%
%
% 1 record(s) selected -- actual I/O 0
% select time including I/O 0.18000 seconds - current
time 16:42:07
'13',888,6737713.98999999881
'17',861,6460573.72
'18',964,7236687.40000001431
'23',892,6701457.95000000954
'29',948,7158866.62999999642
'30',909,6808436.13000000119
'31',922,6806670.17999998569
% total of 7 rows written
Appendix D. Seed and Query Substitution Parameters
```

This Appendix contains Seed values and substitution parameters for each stream

Seed Values

```
stream0 729230326
stream1 729230327
stream2 729230328
stream3 729230329
stream4 729230330
stream5 729230331
stream6 729230332
stream7 729230333
```

Query Parameters

```
==================================================
```

stream0: 729230326

```
14      1994-08-01
2       21      NICKEL  ASIA
9       green
20      black   1994-01-01      ARGENTINA
6       1993-01-01      0.06    25
17      Brand#45        LG BOX
18      315
8       SAUDI ARABIA    MIDDLE EAST     MEDIUM
ANODIZED COPPER
21      ETHIOPIA
13      unusual requests
3       BUILDING        1995-03-14
22      11      29      19      20      23      32
26
16      Brand#24        ECONOMY BRUSHED 12      4
30      11      13      38      6       3
4       1995-02-01
11      RUSSIA  0.0000001000
15      1997-01-01
1       94
10      1993-03-01
19      Brand#21        Brand#21        Brand#42
5       12      30
5       AFRICA  1993-01-01
7       KENYA   SAUDI ARABIA
12      SHIP    MAIL    1994-01-01
```

```
==================================================
```

stream1: 729230327

```
21      RUSSIA
3       MACHINERY       1995-03-30
18      312
5       AMERICA 1993-01-01
11      IRAN    0.0000001000
7       FRANCE  JAPAN
6       1993-01-01      0.04    24
20      lime    1997-01-01      MOROCCO
17      Brand#41        LG JAR
12      FOB     MAIL    1994-01-01
16      Brand#14        SMALL ANODIZED  21      25
22      43      35      27      49      1
15      1994-10-01
13      unusual requests
10      1993-12-01
2       9       TIN     MIDDLE EAST
8       JAPAN   ASIA    SMALL BRUSHED TIN
14      1994-11-01
19      Brand#23        Brand#54        Brand#42
10      13      26
9       floral
22      20      28      15      21      19      24
27
1       102
4       1997-09-01
```

```
==================================================
```

stream2: 729230328

```
==================================================
6       1994-01-01      0.09    24
17      Brand#43        LG CAN
14      1995-03-01
16      Brand#44        LARGE PLATED    19      27
30      37      32      17      18      10
19      Brand#25        Brand#32        Brand#41
5       14      22
10      1994-09-01
9       dark
2       47      COPPER  ASIA
15      1997-05-01
8       EGYPT   MIDDLE EAST     SMALL PLATED TIN
5       ASIA    1994-01-01
22      24      12      11      21      19      16
13
12      MAIL    RAIL    1997-01-01
7       UNITED KINGDOM  EGYPT
13      unusual requests
18      314
1       110
4       1995-06-01
20      steel   1995-01-01      ETHIOPIA
3       BUILDING        1995-03-16
11      UNITED KINGDOM  0.0000001000
21      KENYA
```

```
==================================================
```

stream3: 729230329

```
==================================================
8       VIETNAM ASIA    SMALL ANODIZED TIN
5       EUROPE  1994-01-01
4       1993-03-01
6       1994-01-01      0.07    25
17      Brand#45        MED BOX
7       MOROCCO VIETNAM
1       118
18      315
22      13      23      18      30      22      20
34
14      1995-06-01
9       chocolate
10      1993-06-01
15      1995-02-01
11      IRAQ    0.0000001000
20      gainsboro       1994-01-01      SAUDI ARABIA
2       35      BRASS   AFRICA
21      FRANCE
```

```
19      Brand#32        Brand#25        Brand#35
1       15      29
13      unusual requests
16      Brand#34        PROMO POLISHED  7       17
14      41      43      46      22      25
12      RAIL    MAIL    1995-01-01
3       HOUSEHOLD       1995-03-01
```

```
==================================================
```

stream4: 729230330

```
==================================================
5       MIDDLE EAST     1994-01-01
21      UNITED KINGDOM
14      1995-09-01
19      Brand#34        Brand#53        Brand#34
6       16      25
15      1997-08-01
17      Brand#42        MED JAR
12      AIR     FOB     1995-01-01
6       1994-01-01      0.04    25
4       1995-10-01
9       blush
8       JORDAN  MIDDLE EAST     STANDARD POLISHED TIN
16      Brand#14        MEDIUM ANODIZED 11      10
6       24      37      14      28      8
11      UNITED STATES   0.0000001000
2       23      NICKEL  ASIA
10      1994-03-01
18      313
1       65
13      unusual accounts
7       GERMANY JORDAN
22      15      30      17      23      21      13
18
3       AUTOMOBILE      1995-03-18
20      purple  1997-01-01      IRAN
```

```
==================================================
```

stream5: 729230331

```
==================================================
21      MOZAMBIQUE
15      1995-05-01
4       1993-07-01
6       1994-01-01      0.09    24
7       UNITED STATES   ETHIOPIA
16      Brand#44        ECONOMY BURNISHED       29
9       22      7       43      31      11
    6
19      Brand#31        Brand#45        Brand#34
1       17      22
18      314
14      1995-12-01
22      34      16      25      17      15      26
33
11      JAPAN   0.0000001000
13      express accounts
3       HOUSEHOLD       1995-03-03
1       73
2       10      TIN     AFRICA
5       AMERICA 1994-01-01
8       ETHIOPIA        AFRICA  STANDARD BURNISHED TIN
20      chiffon 1996-01-01      UNITED STATES
12      REG AIR FOB     1995-01-01
17      Brand#44        MED CAN
10      1995-01-01
9       azure
==================================================
```

stream6: 729230332

```
==================================================
10      1993-10-01
3       AUTOMOBILE      1995-03-20
15      1993-02-01
13      express accounts
6       1994-01-01      0.07    25
8       RUSSIA  EUROPE  PROMO BRUSHED NICKEL
9       wheat
7       MOZAMBIQUE      RUSSIA
4       1996-02-01
11      ALGERIA 0.0000001000
22      20      12      21      34      31      30
10
18      312
12      SHIP    FOB     1996-01-01
1       81
5       ASIA    1994-01-01
16      Brand#34        STANDARD POLISHED       13
29      15      8       18      23      4
    34
2       48      COPPER  EUROPE
14      1996-03-01
19      Brand#43        Brand#23        Brand#23
6       18      29
20      misty   1994-01-01      KENYA
17      Brand#41        JUMBO BOX
21      INDIA
==================================================
```

stream7: 729230333

```
==================================================
18      313
8       KENYA   AFRICA  PROMO PLATED NICKEL
20      yellow  1993-01-01      EGYPT
21      ALGERIA
2       36      STEEL   AFRICA
4       1993-11-01
22      12      26      21      13      31      11
24
17      Brand#43        JUMBO JAR
1       89
11      JORDAN  0.0000001000
9       steel
19      Brand#45        Brand#11        Brand#22
2       19      25
3       FURNITURE       1995-03-05
13      express accounts
5       EUROPE  1995-01-01
7       INDIA   KENYA
10      1994-07-01
16      Brand#14        MEDIUM BRUSHED  17      31
35      26      5       8       34      43
6       1995-01-01      0.04    25
14      1996-07-01
15      1995-08-01
12      FOB     MAIL    1995-01-01
```

Appendix E. Implementation-Specific Layer/Driver Code

==========

newtest

```
#!/bin/bash

scope=all
sf=1000
mpssheap=4m
mpssstack=64k
nrfs=7

big=

asiq_running=`ps -eaf | grep asiq | grep -v grep | wc
-l`
if ((asiq_running > 0))
then
   dbstop -c "DSN=tpch" -y
fi

start_asiq @utility.cfg
dbisqlc -c "DSN=utility_db" -q create_database.sql
dbstop -c "DSN=utility_db" -y
start_asiq @tpch.cfg /sybase2/tpch.db
dbisqlc -c "DSN=tpch" -q create_dbspaces.sql
dbisqlc -c "DSN=tpch" -q tpch_rf_int.sql
dbisqlc -c "DSN=tpch" -q tpch_wait.sql
dbisqlc -c "DSN=tpch" -q options.sql         >o.ooo
dbstop -c "DSN=tpch" -y
my_start_asiq $mpssheap  $mpssstack  @tpch.cfg  /
sybase2/tpch.db

dbisqlc -c "DSN=tpch" -q create_tables_${big}int.sql
dbisqlc -c "DSN=tpch" -q load_lineitem.sql &
loadlpid=$!
dbisqlc -c "DSN=tpch" -q load_region.sql
dbisqlc -c "DSN=tpch" -q load_nation.sql
dbisqlc -c "DSN=tpch" -q load_customer.sql
dbisqlc -c "DSN=tpch" -q load_part.sql
dbisqlc -c "DSN=tpch" -q load_supplier.sql
dbisqlc -c "DSN=tpch" -q load_partsupp.sql
dbisqlc -c "DSN=tpch" -q load_orders.sql
loadopid=$!
wait $loadopid
wait $loadlpid
seed=`date '+%m%d%H%M%S'`
gen_streams_new.ksh    $seed  $sf  $nrfs

dbisqlc -c "DSN=tpch" -q dbtables-syb.sql >
rdbtablest_start.out
dbisqlc -c "DSN=tpch" -q dew_cat1.sql >
dew_cat1_start.out
dbisqlc -c "DSN=tpch" -q dew_cat2.sql >
dew_cat2_start.out
dbisqlc -c "DSN=tpch" -q dew_cat3.sql >
dew_cat3_start.out

dbisqlc -q -c "DSN=tpch" check_options.sql >
check_options_start.out

touch /export/home/sybase/run/scripts/rf1.lock
touch /export/home/sybase/run/scripts/rf2.lock

dbisqlc -c "DSN=tpch" -q update_power.sql >
update_power.out &
rfspid=$!

while [ -f /export/home/sybase/run/scripts/rf1.lock ]
do
   sleep 10
done

dbisqlc -c "DSN=tpch" -q stream0.sql > stream0.out
rm -f /export/home/sybase/run/scripts/rf2.lock
wait $rfspid

((i=1))
while ((i<=nrfs))
do
   dbisqlc -c "DSN=tpch" -q stream${i}.sql > stream$
{i}.out &
   ((i=i+1))
done

dbisqlc -c "DSN=tpch" -q update_throughput${nrfs}.sql
> update_throughput.out &
rf0pid=$!
wait $rf0pid

wait

((i=0))
while ((i<=nrfs))
do
   cp stream${i}.out m1s0${i}q.out
   ((i=i+1))
done

cp update_power.out m1s00rf.out
cp update_throughput.out m1s01rf.out

rm *.lock

touch /export/home/sybase/run/scripts/rf1.lock
touch /export/home/sybase/run/scripts/rf2.lock

dbisqlc -c "DSN=tpch" -q update_power.sql >
update_power.out &
rfspid=$!

while [ -f /export/home/sybase/run/scripts/rf1.lock ]
do
   sleep 10
done

dbisqlc -c "DSN=tpch" -q stream0.sql > stream0.out
rm -f /export/home/sybase/run/scripts/rf2.lock
wait $rfspid

((i=1))
while ((i<=nrfs))
do
   dbisqlc -c "DSN=tpch" -q stream${i}.sql > stream$
{i}.out &
   ((i=i+1))
done

dbisqlc -c "DSN=tpch" -q update_throughput${nrfs}.sql
> update_throughput.out &
rf0pid=$!

wait $rf0pid
wait

((i=0))
while ((i<=nrfs))
do
   cp stream${i}.out m2s0${i}q.out
   ((i=i+1))
done

cp update_power.out m2s00rf.out
cp update_throughput.out m2s01rf.out

dbisqlc -c "DSN=tpch" -q dbtables-syb.sql >
rdbtablest_end.out
dbisqlc -c "DSN=tpch" -q dew_cat1.sql >
dew_cat1_end.out
dbisqlc -c "DSN=tpch" -q dew_cat2.sql >
dew_cat2_end.out
dbisqlc -c "DSN=tpch" -q dew_cat3.sql >
dew_cat3_end.out

dbisqlc -q -c "DSN=tpch" check_options.sql >
check_options_end.out
```

==========

tpch_wait.sql

```
if exists (select 1
           from SYS.SYSPROCEDURE
           where proc_name = 'tpch_wait') then
    DROP procedure tpch_wait;
end if
```

```
;

create procedure tpch_wait()
begin

  declare local temporary table t_iq_spaceused(
    mainKB     unsigned bigint,
    mainKBUsed unsigned bigInt,
    tempKB     unsigned bigInt,
    tempKBUsed unsigned bigInt,
    )
    in SYSTEM on commit preserve rows;

  declare maintotal unsigned bigint;
  declare mainused unsigned bigint;
  declare temptotal unsigned bigint;
  declare tempused unsigned bigint;
  declare mainfree unsigned bigint;
  declare command varchar(255);

  select 'xp_cmdshell ''sleep 120''' into command;

  waitloop:
  LOOP
        truncate table t_iq_spaceused;
        execute immediate
             'iq utilities main into t_iq_spaceused
command statistics 30000' ;

        select          mainKB,
                        mainKBUsed,
                        tempKB,
                        tempKBUsed
          into maintotal, mainused, temptotal,
tempused
          from t_iq_spaceused;

        message 'TPCH main total: ',maintotal,'  main
used : ',mainused;
        message 'TPCH temp total: ',temptotal,'  temp
used : ',tempused;
        set mainfree = maintotal-mainused;
        message 'TPCH main free : ',mainfree;

        if ( mainfree > 130000000 )
                then leave waitloop;
                end if;

        select 'xp_cmdshell ''sleep 300''' into
command;
        execute immediate command;

  END LOOP waitloop;

  drop table t_iq_spaceused;
  commit;
end
;
```

Appendix F. Misc database scripts

The dbtables-syb.sql script was run to validate the correctness of the database after the database load. Three other scripts were used to extract basic information about tables and indexes from the database dew_cat1.sql, dew_cat2.sql, dew_cat3.sql.

Auditor Scripts

==========

dbtables-syb.sql

==========

```
-- ================================================
--  FILENAME
--       DBTABLES.SQL
--  DESCRIPTION
--       CHECK ROW COUNT AND ROW STRUCTURE/CONTENT FOR
EACH TABLE
--       IN THE TPC-H DATABASE.
--
-- ================================================
--
-- GET TIMESTAMP
SELECT 'START TIME', CONVERT(CHAR(30), GETDATE(),
120);
go
-- ================================================
--           TABLE: LINEITEM
-- ================================================
SELECT COUNT(*) FROM LINEITEM;
go
SELECT * FROM LINEITEM
WHERE L_ORDERKEY IN
      ( 4, 26598,  148577,  387431, 56704, 517442,
600000)
      AND L_LINENUMBER = 1
ORDER BY L_ORDERKEY;
go
-- ================================================
--           TABLE: ORDERS
-- ================================================
-- GET TIMESTAMP
SELECT 'TIME', CONVERT(CHAR(30), GETDATE(), 120);
go
SELECT COUNT(*) FROM ORDERS;
go
SELECT * FROM ORDERS
WHERE O_ORDERKEY IN ( 7, 44065, 287590, 411111,
483876, 599942 )
ORDER BY O_ORDERKEY;
go
-- ================================================
--           TABLE: PART
-- ================================================
-- GET TIMESTAMP
SELECT 'TIME', CONVERT(CHAR(30), GETDATE(), 120);
go
SELECT COUNT(*) FROM PART;
go
SELECT * FROM PART
WHERE P_PARTKEY IN (1,984,8743,9028,13876,17899,20000)
ORDER BY P_PARTKEY;
go
-- ================================================
--           TABLE: PARTSUPP
-- ================================================
-- GET TIMESTAMP
SELECT 'TIME', CONVERT(CHAR(30), GETDATE(), 120);
go
SELECT COUNT(*) FROM PARTSUPP;
go
SELECT* FROM PARTSUPP
   WHERE PS_PARTKEY = 3398
   AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
      FROM PARTSUPP WHERE PS_PARTKEY = 3398);
go
SELECT* FROM PARTSUPP
   WHERE PS_PARTKEY =15873
   AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
      FROM PARTSUPP WHERE PS_PARTKEY = 15873);
go
SELECT* FROM PARTSUPP
   WHERE PS_PARTKEY = 11394
   AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
      FROM PARTSUPP WHERE PS_PARTKEY = 11394);
go
SELECT* FROM PARTSUPP
   WHERE PS_PARTKEY = 6743
   AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)
      FROM PARTSUPP WHERE PS_PARTKEY = 6743);
go
SELECT* FROM PARTSUPP
   WHERE PS_PARTKEY = 19763
   AND PS_SUPPKEY = (SELECT MIN(PS_SUPPKEY)      FROM
PARTSUPP WHERE PS_PARTKEY =19763);
go
-- ================================================
--           TABLE: SUPPLIER
-- ================================================
-- GET TIMESTAMP
SELECT 'TIME', CONVERT(CHAR(30), GETDATE(), 120);
go
SELECT COUNT(*) FROM SUPPLIER;
go
SELECT * FROM SUPPLIER
WHERE  S_SUPPKEY IN (83,265,492,784,901,1000)
ORDER BY S_SUPPKEY;
go
-- ================================================
--           TABLE: CUSTOMER
-- ================================================
-- GET TIMESTAMP
SELECT 'TIME', CONVERT(CHAR(30), GETDATE(), 120);
go
SELECT COUNT(*) FROM CUSTOMER;
go
SELECT * FROM CUSTOMER
WHERE  C_CUSTKEY IN (832,2653,4924,7845,92016,108070)
ORDER BY C_CUSTKEY;
go
-- ================================================
--           TABLE: NATION & REGION
-- ================================================
-- GET TIMESTAMP
SELECT 'TIME', CONVERT(CHAR(30), GETDATE(), 120);
go
SELECT * FROM REGION;
go
SELECT COUNT(*) FROM NATION;
go
SELECT * FROM NATION
WHERE N_NATIONKEY IN (3,10,14,20)
ORDER BY N_NATIONKEY;
go
-- ================================================
--           CHECK KEY VALUES
-- ================================================
-- GET TIMESTAMP
SELECT 'TIME', CONVERT(CHAR(30), GETDATE(), 120);
go
if exists (select name from sysobjects where
name='MINMAX')
  drop table MINMAX
go
CREATE TABLE MINMAX
(TNAME CHAR(15),
KEYMIN INTEGER,
KEYMAX INTEGER);
go
INSERT INTO MINMAX
SELECT 'LINEITEM_ORD',MIN(L_ORDERKEY),MAX(L_ORDERKEY)
FROM LINEITEM;
go
INSERT INTO MINMAX
SELECT 'LINEITEM_NBR',MIN(L_LINENUMBER),MAX
(L_LINENUMBER)
FROM LINEITEM;
go
INSERT INTO MINMAX
SELECT 'ORDERS',MIN(O_ORDERKEY),MAX(O_ORDERKEY)
FROM ORDERS;
go
INSERT INTO MINMAX
SELECT 'CUSTOMER',MIN(C_CUSTKEY),MAX(C_CUSTKEY)
FROM CUSTOMER;
```

```
go
INSERT INTO MINMAX
SELECT 'PART',MIN(P_PARTKEY),MAX(P_PARTKEY)
FROM PART;
go
INSERT INTO MINMAX
SELECT 'SUPPLIER',MIN(S_SUPPKEY),MAX(S_SUPPKEY)
FROM SUPPLIER;
go
INSERT INTO MINMAX
SELECT 'PARTSUPP_PART',MIN(PS_PARTKEY),MAX(PS_PARTKEY)
FROM PARTSUPP;
go
INSERT INTO MINMAX
SELECT 'PARTSUPP_SUPP',MIN(PS_SUPPKEY),MAX(PS_SUPPKEY)
FROM PARTSUPP;
go
INSERT INTO MINMAX
SELECT 'NATION',MIN(N_NATIONKEY),MAX(N_NATIONKEY)
FROM NATION;
go
INSERT INTO MINMAX
SELECT 'REGION',MIN(R_REGIONKEY),MAX(R_REGIONKEY)
FROM REGION;
go
SELECT * FROM MINMAX;
go
if exists (select name from sysobjects where
name='MINMAX')
  drop table MINMAX
go
SELECT 'END TIME', CONVERT(CHAR(30), GETDATE(), 120);
go
```

==========

dew_cat1.sql

==========

```
SELECT  st.table_name,
                st.table_type,
                su.user_name,
                st.server_type
          from SYS.SYSTABLE st, SYS.SYSUSERPERMS su
          where creator = user_id
order by 4,1,3;
```

==========

dew_cat2.sql

==========

```
 select T.table_name    ,
        T.table_type    ,
        C.column_name   ,
        C.column_id
 From   SYS.SYSTABLE T,
        SYS.SYSCOLUMN C,
        SYS.SYSDOMAIN D,
        SYS.SYSUSERPERMS SU
 where T.creator = SU.user_id
   and T.table_id = C.table_id
   and C.domain_id = D.domain_id
order by 1,2;
```

==========

dew_cat3.sql

==========

```
SELECT  index_name,T.table_name  ,
        column_name   ,
        index_type
 from   SYS.SYSTABLE T,
        SYS.SYSCOLUMN C,
        SYS.SYSINDEX I,
        SYS.SYSUSERPERMS UP,
        SYS.SYSFILE F,
        SYS.SYSIXCOL IC
 where T.table_id = C.table_id
   and C.table_id = I.table_id
   and T.file_id = F.file_id
   and I.table_id = IC.table_id
   AND I.index_id = IC.index_id
   AND IC.column_id = C.column_id
   and T.creator = UP.user_id;
```

Appendix G. Pricing information

Company Sun Microsystems
Contact Richard Gostanian
Phone 781-442-3063
Fax
Address 1 Network Drive, Burlington MA 01803

Quotation for Software and Support

SYBASE Sales lHollie Nash
Phone: 972-687-6412
Fax: 972-687-6409

CBSS#

	Catalogue Number	Product Description	License Type	Machine	P/S	List Price Per Unit	Quantity	Discount	Extended Price	Extended Support Fees
										3 YEARS
1	11467	Sybase IQ Single App Svr, per cpu core	CP	Sun64	P	2,595	8		20,760.00	
3	98477	3 yr support Single App Svr, per cpu core				1,557	8			12,456.00
4		Discount (5% if total license + support dollar value > 25000)						-1660.80		
5										
6										
7										
8										
9										
10										
11										
12										

Quote Date: ***5/5/05***
Valid thru:

Total 31,555.20
Licence + 3 year support

Payment terms : Net 30 Days

5400 LBJ Freeway, Suite 1500, Dallas, TX 75240

Eakins Open Systems
SYSTEMS INTEGRATOR

Quotation

Sun Microsystems

Friday, August 13, 2004

,
, CA

Quote No.: 15581
ATTN: Mr. Richard Gostanian
By Laurent Catabelle
Phone: (781) 442-3063 Ext. **Fax:**
Ext. 2119
E-mail: Richard.Gostanian@Sun.com

LEASE OPTION

Monthly lease payment for below items : for: months
Please call for additional details

Item	Qty	Description	List Price	Unit Price	Total Price
A	1	A30-WUF8-32GRF; Sun Fire V880 Server, 8 1200-MHz UltraSPARC III Cu Processors w/ 8-MB External Cache Each, 32-GB Memory, 6 73-GB 10000 RPM FC-AL Disks, DVD-ROM, 3 1500-Watt Power Supplies & Redundant Cooling Fan Trays, Solaris 9 Operating System	$109,995.00	$94,195.00	$94,195.00
B	1	W9D-A30-8P-3G; SunSpectrum Gold Support Agreement, 3 Years, for Sun 8-Way V880.		$15,507.40	$15,507.40
C	2	XTA3310R01A0R876; Sun StorEdge 3310 SCSI Array,RR,876GB(12x73GB 10K RPM disks), Ultra160 SCSI-JBOD, and 2 ACpower supplies.	$14,995.00	$10,696.00	$21,392.00
D	2	W9D-SE3310-3G; SunSpectrum Gold Support Agreement, 3 Years, for Sun StorEdge 3310.		$3,703.00	$7,406.00
E	2	X6758A; Sun PCI dual channel Ultra 3 differential SCSI host adapter	$800.00	$544.00	$1,088.00
F	4	X1138A; Sun Cable SCSI VHDCI 2m	$95.00	$77.90	$311.60

Eakins Open Systems
SYSTEMS INTEGRATOR

Quotation

Sun Microsystems

Friday, August 13, 2004

,
, CA

Quote No.: 15581
By Laurent Catabelle
Ext. 2119

ATTN: Mr. Richard Gostanian
Phone: (781) 442-3063 Ext. **Fax:**
E-mail: Richard.Gostanian@Sun.com

Total **$139,900.00**

Eakins has a fully qualified technical support staff and offers annual maintenance agreements on most popular Sun equipment, as well as telephone and e-mail support. Call for prices.

To order, telephone your salesperson at (650) 969-5109 referring to the above quote number, or FAX your purchase order at (650) 961-2130.

This quote is valid for 60 days, unless otherwise stated, or manufacturer change in pricing. The products quoted above will be supplied with manufacturer's warranty and without installation.

Installation can be provided either on-site or in our shop for a fee. To receive installation, the purchase order must state the request.

Payment options are as follows: C.O.D., credit card (MC & VISA), company check, lease, or other terms available on request with approved credit. Please allow 5-7 business days for credit review. Sales tax must be added unless documentation of exemption is provided.

All shipments are F.O.B. shipping point.

650-969-5109, FAX 650-961-2130, info@eos.com