TPC Benchmark™ H Full Disclosure Report

IBM Power 595 Model 9119-FHA

Using

Sybase IQ Single Application Server Edition v.15.1 ESD #1.2

First Edition
November 24, 2009
The following terms used in this publication are trademarks of their respective companies as follows:

Trademark of the Transaction Processing Performance Council:
TPC Benchmark
TPC-H
QppH
QthH
QphH

Trademark of International Business Machines Corporation:
IBM
pSeries
POWER6
TotalStorage
AIX
AIX VERSION 6.1

Trademark of Sybase Inc:
Sybase IQ

First Edition November 24, 2009
The information contained in this document has not been submitted to any formal test and is distributed on an AS IS basis without any warranty either expressed or implied. The use of this information or the implementation of any of these techniques is a customer's responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item has been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used; any functionally equivalent program may be used.

It is possible that this material may contain references to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such products, programming, or services in your country.

All performance data contained in this publication was obtained in a controlled environment, and therefore the results which may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable date in their specific environment.

© Copyright International Business Machines 2007 All rights reserved.

Permission is hereby granted to reproduce this document in whole or in part, provided the copyright notice printed above is set forth in full text on the title page of each item reproduced.
IBM Power 595 Model 9119-FHA

Total System Cost: 3,224,000 USD
Composite Query per Hour Rating: 156,537.3 QphH@3000GB
Price/Performance: 20.60 USD Price/QphH@3000GB

Database Size: 3000GB

Database Manager: Sybase IQ Single Application Server Edition v.15.1 ESD #1.2
Operating System: AIX V6.1
Other Software: None
Availability Date: Now

Database Load Time: 10:02:25
Load included backup: N
Total Data Storage/Database Size: 6.58
RAID (base tables): Y
RAID (Base Tables and Auxiliary Data Structures): Y
RAID (All): Y

System Configurations
1 IBM Power 595 Model 9119-FHA
Processors: 64 5.0GHz POWER6 Cores with 32MB L3 Cache per chip
Memory: 512GB
Disk Controllers: 24 x 4Gigabit Fibre Channel PCI-E Adapters & 12 IBM TotalStorage DS4800 dual controllers
Disk Drives: 288 x73GB 15K 4Gb FC drives; 5 x 146.8GB SAS drives
Total Disk Storage: 20,226.27 GB (GB is defined as 1024 * 1024 * 1024 bytes)
## IBM Power 595 Model 9119-FHA

**Report Date:** November 24, 2009

<table>
<thead>
<tr>
<th>Description</th>
<th>Part No.</th>
<th>Source</th>
<th>Unit Price</th>
<th>Qty</th>
<th>Ext Price</th>
<th>Maint Price</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server Hardware</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Server 1.9119 Model 595</td>
<td>9119-FHA</td>
<td>1</td>
<td>91,000</td>
<td>1</td>
<td>91,000</td>
<td>63,192</td>
</tr>
<tr>
<td>GX Dual-port 12x HCA</td>
<td>1818</td>
<td>1</td>
<td>4,000</td>
<td>6</td>
<td>24,000</td>
<td></td>
</tr>
<tr>
<td>2.5M Enhanced 12X Cable to connect to a gc card in</td>
<td>1831</td>
<td>1</td>
<td>650</td>
<td>12</td>
<td>7,800</td>
<td></td>
</tr>
<tr>
<td>145.8 GB Ultra320 15K RPM</td>
<td>3279</td>
<td>1</td>
<td>1,299</td>
<td>5</td>
<td>6,495</td>
<td></td>
</tr>
<tr>
<td>0/8 - core POWER6 5.0GHz CoD, 0-core Active</td>
<td>4695</td>
<td>1</td>
<td>67,700</td>
<td>8</td>
<td>541,600</td>
<td>99,840</td>
</tr>
<tr>
<td>Single Processor Activation for FC 4696 (5.0 GHz)</td>
<td>4755</td>
<td>1</td>
<td>33,900</td>
<td>64</td>
<td>2,169,600</td>
<td>430,080</td>
</tr>
<tr>
<td>255GB Bundle DDR2 Activations</td>
<td>5681</td>
<td>1</td>
<td>83,200</td>
<td>2</td>
<td>166,400</td>
<td></td>
</tr>
<tr>
<td>0/8GB DDR2 Memory (4x2GB) DIMMS - 667 MHz</td>
<td>5694</td>
<td>1</td>
<td>650</td>
<td>64</td>
<td>41,600</td>
<td></td>
</tr>
<tr>
<td>2-PORT BASETX ETHT,PCI-X A</td>
<td>5767</td>
<td>1</td>
<td>699</td>
<td>1</td>
<td>699</td>
<td></td>
</tr>
<tr>
<td>47 GB IDE Slimline DVD-RAM Drive</td>
<td>5757</td>
<td>1</td>
<td>660</td>
<td>1</td>
<td>660</td>
<td></td>
</tr>
<tr>
<td>4 Gigabit Dual-port Fibre Channel PCI-E Adapter</td>
<td>5774</td>
<td>1</td>
<td>3,308</td>
<td>24</td>
<td>79,392</td>
<td></td>
</tr>
<tr>
<td>PCI-X DDR Dual - x4 SAS Adapter</td>
<td>5912</td>
<td>1</td>
<td>1,092</td>
<td>1</td>
<td>1,092</td>
<td></td>
</tr>
<tr>
<td>12l/0 Drawer PCI-E, SFF disk</td>
<td>5803</td>
<td>1</td>
<td>32,000</td>
<td>3</td>
<td>95,000</td>
<td>26,712</td>
</tr>
<tr>
<td><strong>Bulk Power Regulator</strong></td>
<td>0333</td>
<td>1</td>
<td>4,200</td>
<td>6</td>
<td>33,600</td>
<td></td>
</tr>
<tr>
<td><strong>Bulk Power Distribution Assembly</strong></td>
<td>0334</td>
<td>1</td>
<td>2,500</td>
<td>4</td>
<td>10,000</td>
<td></td>
</tr>
<tr>
<td><strong>Slimline Doors (F&amp;R), H CEC AND I/O Expansion Radi</strong></td>
<td>0589</td>
<td>1</td>
<td>8,000</td>
<td>1</td>
<td>8,000</td>
<td></td>
</tr>
<tr>
<td>UPIC Cable Group, BPD1 to I/O Drawer at A01</td>
<td>8941</td>
<td>1</td>
<td>500</td>
<td>1</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td>UPIC Cable Group, BPD1 to I/O Drawer A05</td>
<td>8942</td>
<td>1</td>
<td>500</td>
<td>1</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td>UPIC Cable Group, BPD1 to I/O Drawer A09</td>
<td>8943</td>
<td>1</td>
<td>500</td>
<td>1</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td>3rd &amp; 4th AMDs + UPICY-Cable Group (BPC to Fans)</td>
<td>8952</td>
<td>1</td>
<td>5,000</td>
<td>1</td>
<td>5,000</td>
<td></td>
</tr>
<tr>
<td>UCIF Y-Cable Group BPD1 to 1st Processor Node</td>
<td>8961</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UCIF Y-Cable Group BPD1 to 2nd Processor Node</td>
<td>8962</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UCIF Y-Cable Group BPD1 to 3rd Processor Node</td>
<td>8963</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UCIF Y-Cable Group BPD1 to 4th Processor Node</td>
<td>8964</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UCIF Y-Cable Group BPD1 to 5th Processor Node</td>
<td>8965</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UCIF Y-Cable Group BPD1 to 6th Processor Node</td>
<td>8966</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UCIF Y-Cable Group BPD1 to 7th Processor Node</td>
<td>8967</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UCIF Y-Cable Group BPD1 to 8th Processor Node</td>
<td>8968</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>Ethernet Cable, 15M, Hardware Management</td>
<td>7802</td>
<td>1</td>
<td>34</td>
<td>2</td>
<td>68</td>
<td></td>
</tr>
<tr>
<td><strong>Line Cord, 5AWG/Type W, 14ft, IEC309 100A Plug</strong></td>
<td>8688</td>
<td>1</td>
<td>2,500</td>
<td>4</td>
<td>10,000</td>
<td></td>
</tr>
<tr>
<td>HMC 17042-C07 Desktop Hardw.Mgmt.Console</td>
<td>7042-C07</td>
<td>1</td>
<td>1,830</td>
<td></td>
<td>1,830</td>
<td>1,344</td>
</tr>
<tr>
<td>IBM T117 TFT 17 inch Color Display</td>
<td>3646</td>
<td>1</td>
<td>875</td>
<td>1</td>
<td>875</td>
<td></td>
</tr>
<tr>
<td>Quiet Touch Keyboard - USB, Business Black</td>
<td>5951</td>
<td>1</td>
<td>104</td>
<td></td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Power Cord (6-foot), To Wall (125V, 15A), Plugs Type #</td>
<td>8470</td>
<td>1</td>
<td>18</td>
<td>2</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>Ethernet Cable, 6M, Hardware Management</td>
<td>8701</td>
<td>1</td>
<td>15</td>
<td></td>
<td>15</td>
<td></td>
</tr>
<tr>
<td><strong>Mouse - Business Black with Keyboard Attachment</strong></td>
<td>8881</td>
<td>1</td>
<td>78</td>
<td>1</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>PCI-e 1Gb Ethernet UTP 2Port</td>
<td>5767</td>
<td>1</td>
<td>699</td>
<td>1</td>
<td>699</td>
<td></td>
</tr>
<tr>
<td><strong>Subtotal</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3,303,343</td>
<td>621,168</td>
</tr>
</tbody>
</table>

| Storage                                                                    |                |        |            |     |           |             |
| DS4800 Disk System Model 82 (4 GB Cache)                                   | 1815-82A       | 1      | 53,955     | 12  | 647,940   |             |
| DS4800 8-Storage Partitions                                                | 3870           | 1      | 10,000     | 12  | 120,000   |             |
| (22R4255) DS4800 AIX Host Kit                                              | 7711           | 1      | 7,000      | 12  | 84,000    |             |
| DS4000 EXP810 Enclosure                                                   | 1812-81A       | 1      | 6,000      | 48  | 288,000   |             |
| **ShortWave SFP**                                                          | 2410           | 1      | 998        | 96  | 95,806    |             |
| **16-PAK 4 GBPS 73GB GB/15K**                                              | 5433           | 1      | 14,816     | 18  | 266,688   |             |
| **Fiber Cable 25m**                                                       | 5625           | 1      | 189        | 48  | 9,072     |             |
| **Fiber Cable 1m**                                                        | 5601           | 1      | 79         | 96  | 7,584     |             |
| **3 Year Warranty Service Upgrade 1812-81A 24x7x4**                       | 41L2760        | 1      | 300        | 5   | 0         | 46,080      |
| **3 Year Warranty Service Upgrade 1815-82A 24x7x4**                       | 41L2760        | 1      | 300        | 12  | 0         | 38,400      |
| **Service Pac for 3-Year 24x7x4 Support (Rack)**                          | 41L2760        | 1      | 300        | 5   | 0         | 1,500       |
| IBM S2 42U Standard Rack                                                  | 93074RX        | 1      | 1,489      | 5   | 7,445     |             |
| **Subtotal**                                                               |                |        |            |     | 1,526,537 | 85,980      |
**Server Software**

<table>
<thead>
<tr>
<th>Description</th>
<th>Model</th>
<th>Quantity</th>
<th>Price</th>
<th>Discount</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIX 6.1 (media only)</td>
<td>5992-A6P</td>
<td>1</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>AIX Software per Processor</td>
<td>5765-G62</td>
<td>1</td>
<td>2,495</td>
<td>64</td>
</tr>
<tr>
<td>AIX IBM Management Edition for AIX</td>
<td>5765-AME</td>
<td>1</td>
<td>490</td>
<td>64</td>
</tr>
<tr>
<td>AIX Software Maintenance (3Y)</td>
<td>73-SM3-474</td>
<td>1</td>
<td>2,836</td>
<td>64</td>
</tr>
<tr>
<td>AIX Software Maintenance 24x7 Upgrade (3Y)</td>
<td>73-SM3-475</td>
<td>1</td>
<td>732</td>
<td>64</td>
</tr>
<tr>
<td>AIX IBM Mgmt Edition of AIX Software Maintenance (373-AME-969)</td>
<td>1</td>
<td>196</td>
<td>64</td>
<td>12,544</td>
</tr>
<tr>
<td>HMC Software SUB (3Y)</td>
<td>5773-0570</td>
<td>1</td>
<td>236</td>
<td>1</td>
</tr>
<tr>
<td>HMC Software Support (3Y)</td>
<td>5773-0569</td>
<td>1</td>
<td>675</td>
<td>1</td>
</tr>
<tr>
<td>C for AIX user Lic+SW maint 12 MO</td>
<td>D5A1DLL</td>
<td>1</td>
<td>975</td>
<td>1</td>
</tr>
<tr>
<td>C for AIX user annual SW maint renewal</td>
<td>E1A1FLL</td>
<td>1</td>
<td>195</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Subtotal</strong></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>192,065</td>
<td></td>
<td>242,197</td>
</tr>
</tbody>
</table>

**Third Party Hardware/Software**

<table>
<thead>
<tr>
<th>Description</th>
<th>Quantity</th>
<th>Price</th>
<th>Discount</th>
</tr>
</thead>
<tbody>
<tr>
<td>SybaseIQ15 (64 processors)</td>
<td>2</td>
<td>2,595</td>
<td>64</td>
</tr>
<tr>
<td>Sybase IQ15 (64 processors, 3 years)</td>
<td>2</td>
<td>1,713</td>
<td>64</td>
</tr>
</tbody>
</table>

| **Subtotal** |       | 166,080 |       | 109,632 |
| **Total**    |       | 5,188,025 |       | 1,058,977 |

| **Total IBM Discounts** |       | -2,981,645 |
| **Sybase Discount (15%)** |       | -41,357 |

**Three-Year Cost of Ownership** | 3,224,000 |
| QphH@3000GB | 156,537 |
| $/QphH@3000GB | 20.60 |

**Notes:**
For pricing details and contact information please see appendix E

**Pricing Sources:** 1) IBM 2) Sybase

*Discounts are based on US list prices for similar quantities & configurations including pre-payment for maintenance. The discount applies to the totality of all items with price source of *1*."

---

**Audited by:** Francois Raab of Infosizing (www.sizing.com)

The system as configured for the test is available Now.

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

Measurement Results

- Database Scaling (SF/Size) = 3000
- Total Data Storage/Database Size = 6.58
- End of Database Load = 10/27/09 23:51:04
- Database Load Time = 10:02:25
- Query Streams for Throughput Test = 9
- TPC-H Power Metric (QppH@3000GB) = 142,790.7
- TPC-H Throughput Metric (QthH@3000GB) = 171,607.4
- Composite Query-per-Hour Rating (QphH@3000GB) = 156,537.3
- Total System Price over 3 years = 3,224,000 USD
- TPC-H Price/Performance Metric ($/QphH@3000GB) = 20.60 USD

Measurement Intervals

Measurement Interval in Throughput Test (Ts) = 12,461 seconds

Duration of Stream Execution

<table>
<thead>
<tr>
<th>Stream ID</th>
<th>Seed Used</th>
<th>Query Start</th>
<th>RF1 Start</th>
<th>RF2 Start</th>
<th>Queries</th>
<th>Duration</th>
</tr>
</thead>
</table>
## Timing Interval (in seconds)

<table>
<thead>
<tr>
<th>Stream ID</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
<th>Q9</th>
<th>Q10</th>
<th>Q11</th>
<th>Q12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream 0</td>
<td>282.7</td>
<td>27.5</td>
<td>122.1</td>
<td>46.4</td>
<td>149.3</td>
<td>7.0</td>
<td>107.3</td>
<td>112.5</td>
<td>230.5</td>
<td>136.7</td>
<td>41.9</td>
<td>50.3</td>
</tr>
<tr>
<td>Stream 1</td>
<td>1,550.7</td>
<td>73.3</td>
<td>580.3</td>
<td>170.1</td>
<td>681.3</td>
<td>61.2</td>
<td>354.4</td>
<td>359.6</td>
<td>1,721.6</td>
<td>471.4</td>
<td>91.9</td>
<td>272.1</td>
</tr>
<tr>
<td>Stream 2</td>
<td>1,841.3</td>
<td>93.5</td>
<td>733.3</td>
<td>369.6</td>
<td>846.6</td>
<td>82.8</td>
<td>321.7</td>
<td>494.0</td>
<td>1,326.9</td>
<td>571.5</td>
<td>277.6</td>
<td>235.1</td>
</tr>
<tr>
<td>Stream 3</td>
<td>1,182.3</td>
<td>93.9</td>
<td>584.1</td>
<td>293.5</td>
<td>833.6</td>
<td>39.1</td>
<td>343.3</td>
<td>539.1</td>
<td>1,203.1</td>
<td>550.8</td>
<td>222.4</td>
<td>233.6</td>
</tr>
<tr>
<td>Stream 4</td>
<td>1,749.5</td>
<td>91.9</td>
<td>521.2</td>
<td>165.7</td>
<td>888.1</td>
<td>32.7</td>
<td>300.4</td>
<td>396.3</td>
<td>1,467.3</td>
<td>638.9</td>
<td>266.6</td>
<td>173.9</td>
</tr>
<tr>
<td>Stream 5</td>
<td>1,541.6</td>
<td>82.3</td>
<td>562.0</td>
<td>314.3</td>
<td>984.1</td>
<td>32.5</td>
<td>267.3</td>
<td>564.7</td>
<td>1,156.9</td>
<td>591.9</td>
<td>119.5</td>
<td>175.9</td>
</tr>
<tr>
<td>Stream 6</td>
<td>1,809.1</td>
<td>123.1</td>
<td>596.7</td>
<td>343.0</td>
<td>896.1</td>
<td>29.8</td>
<td>291.1</td>
<td>316.6</td>
<td>1,603.9</td>
<td>776.4</td>
<td>183.5</td>
<td>169.8</td>
</tr>
<tr>
<td>Stream 7</td>
<td>1,757.9</td>
<td>111.5</td>
<td>596.7</td>
<td>291.6</td>
<td>774.4</td>
<td>37.7</td>
<td>262.1</td>
<td>501.3</td>
<td>1,473.0</td>
<td>668.6</td>
<td>148.6</td>
<td>67.7</td>
</tr>
<tr>
<td>Stream 8</td>
<td>1,582.6</td>
<td>100.2</td>
<td>405.9</td>
<td>384.5</td>
<td>849.1</td>
<td>54.6</td>
<td>317.0</td>
<td>417.2</td>
<td>1,210.7</td>
<td>606.3</td>
<td>106.9</td>
<td>235.3</td>
</tr>
<tr>
<td>Stream 9</td>
<td>1,814.0</td>
<td>177.8</td>
<td>528.6</td>
<td>295.8</td>
<td>346.2</td>
<td>43.8</td>
<td>347.6</td>
<td>530.9</td>
<td>1,135.6</td>
<td>746.0</td>
<td>172.8</td>
<td>249.8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Minimum</th>
<th>Avgerage</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream ID</td>
<td>1,182.3</td>
<td>1,647.7</td>
<td>1,841.3</td>
</tr>
<tr>
<td>Q13</td>
<td>73.3</td>
<td>105.3</td>
<td>177.8</td>
</tr>
<tr>
<td>Q14</td>
<td>405.9</td>
<td>567.6</td>
<td>733.3</td>
</tr>
<tr>
<td>Q15</td>
<td>165.7</td>
<td>292.0</td>
<td>733.3</td>
</tr>
<tr>
<td>Q16</td>
<td>346.2</td>
<td>788.8</td>
<td>342.4</td>
</tr>
<tr>
<td>Q17</td>
<td>29.8</td>
<td>46.0</td>
<td>29.8</td>
</tr>
<tr>
<td>Q18</td>
<td>636.6</td>
<td>311.7</td>
<td>636.6</td>
</tr>
<tr>
<td>Q19</td>
<td>1,135.6</td>
<td>457.7</td>
<td>1,135.6</td>
</tr>
<tr>
<td>Q20</td>
<td>471.4</td>
<td>1,366.6</td>
<td>471.4</td>
</tr>
<tr>
<td>Q21</td>
<td>91.9</td>
<td>624.6</td>
<td>91.9</td>
</tr>
<tr>
<td>Q22</td>
<td>67.7</td>
<td>626.4</td>
<td>67.7</td>
</tr>
<tr>
<td>RF1</td>
<td>321.8</td>
<td>176.6</td>
<td>321.8</td>
</tr>
<tr>
<td>RF2</td>
<td>201.5</td>
<td>272.1</td>
<td>201.5</td>
</tr>
</tbody>
</table>

- **Minimum**
- **Avgerage**
- **Maximum**

---

**Model 9119**

- **Q13**
- **Q14**
- **Q15**
- **Q16**
- **Q17**
- **Q18**
- **Q19**
- **Q20**
- **Q21**
- **Q22**
- **RF1**
- **RF2**

---

**FHA**

- **Q13**
- **Q14**
- **Q15**
- **Q16**
- **Q17**
- **Q18**
- **Q19**
- **Q20**
- **Q21**
- **Q22**
- **RF1**
- **RF2**
Benchmark Sponsor: William Bostic  
IBM Power Systems Performance  
11501 Burnet Road  
Austin, TX 78758  

November 23, 2009  

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

Platform: IBM Power 595 Model 9119-FHA  
Database Manager: Sybase IQ Single Application Server Edition v.15.1 ESD#1.2  
Operating System: AIX V6.1  

The results were:  

<table>
<thead>
<tr>
<th>CPU (Speed)</th>
<th>Memory</th>
<th>Disks</th>
<th>QphH@3000GB</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 x POWER6 Dual-Core (5.0GHz)</td>
<td>32 MB L3/chip 512 GB Main</td>
<td>288 x 73GB ext. 5 x 146 GB int.</td>
<td>156,537.3</td>
</tr>
</tbody>
</table>

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 3,000GB and populated accordingly  
- The compliance of the database auxiliary data structures was verified  
- The database load time was correctly measured and reported
• The required ACID properties were verified and met
• The query input variables were generated by QGEN
• The query text was produced using minor modifications and no query variant
• The execution of the queries against the SF1 database produced compliant answers
• A compliant implementation specific layer was used to drive the tests
• The throughput tests involved 9 query streams
• The ratio between the longest and the shortest query was such that no query timings were 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
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>1.0</td>
<td>General Items</td>
<td>4</td>
</tr>
<tr>
<td>1.1</td>
<td>Benchmark Sponsor</td>
<td>4</td>
</tr>
<tr>
<td>1.2</td>
<td>Parameter Settings</td>
<td>4</td>
</tr>
<tr>
<td>1.3</td>
<td>Configuration Diagrams</td>
<td>4</td>
</tr>
<tr>
<td>2.0</td>
<td>Clause 2: Logical Database Design Related Items</td>
<td>6</td>
</tr>
<tr>
<td>2.1</td>
<td>Table Definitions</td>
<td>6</td>
</tr>
<tr>
<td>2.2</td>
<td>Database Organization</td>
<td>6</td>
</tr>
<tr>
<td>2.3</td>
<td>Horizontal Partitioning</td>
<td>6</td>
</tr>
<tr>
<td>2.4</td>
<td>Replication</td>
<td>6</td>
</tr>
<tr>
<td>3.0</td>
<td>Clause 3: Queries and Refresh Functions</td>
<td>7</td>
</tr>
<tr>
<td>3.1</td>
<td>Query Language</td>
<td>7</td>
</tr>
<tr>
<td>3.2</td>
<td>Verification for the Random Number Generator</td>
<td>7</td>
</tr>
<tr>
<td>3.3</td>
<td>Substitution Parameters</td>
<td>7</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Method of Generation</td>
<td>7</td>
</tr>
<tr>
<td>3.4</td>
<td>Query Text</td>
<td>7</td>
</tr>
<tr>
<td>3.5</td>
<td>Query Substitution Parameters and Seeds</td>
<td>8</td>
</tr>
<tr>
<td>3.6</td>
<td>Isolation Level</td>
<td>8</td>
</tr>
<tr>
<td>3.7</td>
<td>Refresh Functions</td>
<td>8</td>
</tr>
<tr>
<td>4.0</td>
<td>Clause 4: Database System Properties</td>
<td>9</td>
</tr>
<tr>
<td>4.1</td>
<td>Atomicity Requirements</td>
<td>9</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Atomicity of Completed Transaction</td>
<td>9</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Atomicity of Aborted Transactions</td>
<td>9</td>
</tr>
<tr>
<td>4.2</td>
<td>Consistency Requirements</td>
<td>10</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Consistency Condition</td>
<td>10</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Consistency Tests</td>
<td>10</td>
</tr>
<tr>
<td>4.3</td>
<td>Isolation Requirements</td>
<td>10</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Isolation Test 1</td>
<td>10</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Isolation Test 2</td>
<td>11</td>
</tr>
<tr>
<td>4.3.3</td>
<td>Isolation Test 3</td>
<td>11</td>
</tr>
<tr>
<td>4.3.4</td>
<td>Isolation Test 4</td>
<td>11</td>
</tr>
<tr>
<td>4.3.5</td>
<td>Isolation Test 5</td>
<td>12</td>
</tr>
<tr>
<td>4.3.6</td>
<td>Isolation Test 6</td>
<td>12</td>
</tr>
<tr>
<td>4.4</td>
<td>Durability Requirements</td>
<td>12</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Permanent Failure of Single Durable Medium and Loss of System Power</td>
<td>12</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Failure of Storage Controller, and Loss of System Power</td>
<td>13</td>
</tr>
<tr>
<td>5.0</td>
<td>Clause 5: Scaling and Database Population Related Items</td>
<td>14</td>
</tr>
<tr>
<td>5.1</td>
<td>Cardinality of Tables</td>
<td>14</td>
</tr>
<tr>
<td>5.2</td>
<td>Distribution of Tables and Logs</td>
<td>14</td>
</tr>
<tr>
<td>5.3</td>
<td>Mapping of Database Partitions/Replications</td>
<td>14</td>
</tr>
<tr>
<td>5.4</td>
<td>Implementation of RAID</td>
<td>15</td>
</tr>
<tr>
<td>5.5</td>
<td>DBGEN Modifications</td>
<td>15</td>
</tr>
</tbody>
</table>
5.6. Database Loading
5.7. Data Storage Ratio
5.8. Details of Database Loading

6.0 Clause 6: Performance Metrics and Execution-Rules Related Items
6.1. System Activity between Load and Performance Tests
6.2. Steps in the Power Test
6.3. Timing Intervals for Each Query and Refresh Function
6.4. Number of Streams for the Throughput Test
6.5. Start and End Date/Times for Each Query Stream
6.6. Total Elapsed Time for the Measurement Interval
6.7. Refresh Function Start Date/Time and Finish Date/Time
6.8. Timing Intervals for Each Query and Each Refresh Function for Each Stream
6.9. Performance Metrics
6.10. The Performance Metric and Numerical Quantities from Both Runs
6.11. System Activity between Tests

7.0 Clause 7: SUT and Driver Implementation
7.1. Driver
7.2. Implementation Specific Layer
7.3. Profile-Directed Optimization

8.0 Clause 8: Pricing-Related Items
8.1. Hardware and Software Used
8.2. Three Year Cost of System Configuration
8.3. System Availability Date

9.0 Clause 9: Audit Items

Appendix - A Tunable Parameters
A.1 Sybase IQ Database Configuration
A.2 AIX Parameters

Appendix - B Programs and Scripts
B.1 create_database.sql
B.2 create_dbspaces_15_0.sh
B.3 create_tables.sql
B.4 create_refresh_functions.sql
B.5 load_region.sql
B.6 load_nation.sql
B.7 load_customer.sql
B.8 load_part.sql
B.9 load_supplier.sql
B.10 load_partsupp.sql
B.11 load_orders.sql
| E.29 | Acid_isolation_test5_transaction1.tst | 69 |
| E.30 | Acid_isolation_test6_query.tst      | 70 |
| E.31 | Acid_isolation_test6_transaction1.tst | 71 |

*Appendix - F  Pricing Information  72*
Abstract

This report documents the full disclosure information required by the TPC Benchmark™ H Standard Specification Revision 2.8.0 dated September 11, 2008 for measurements on the IBM System Power 595 Model 9119-FHA.

The software used includes AIX V6.1 operating system with Sybase IQ Single Application Server Edition v.15.1 ESD #1.2.
Preface

TPC Benchmark™ H Standard Specification was developed by the Transaction Processing Performance Council (TPC). It was released on February 26, 1999, and most recently revised (Revision 2.8.0) on September 11, 2008. This is the full disclosure report for benchmark testing of the IBM Power 595 Model 9119-FHA according to the TPC Benchmark™ H Standard Specification.

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

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

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

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

The TPC-H operations are modeled as follows:

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

The minimum database required to run the benchmark holds business data from 10,000 suppliers. It contains almost ten million rows representing a raw storage capacity of about 1 gigabyte. Compliant benchmark implementations may also use one of the larger permissible database populations (e.g., 100 gigabytes), as defined in Clause 4.1.3.

The performance metric reported by TPC-H is called the TPC-H Composite Query-per-Hour Performance Metric (QphH@Size), and reflects multiple aspects of the capability of the system to process queries. These aspects include the selected database size against which the queries are executed, the query processing power when queries are submitted by a single stream, and the query throughput when queries are submitted by multiple concurrent users. The TPC-H Price/Performance metric is expressed as $/QphH@Size. To be compliant with the TPC-H standard, all references to TPC-H results for a given configuration must include all required reporting components (see Clause 5.4.6). The TPC believes that comparisons of TPC-H results measured against different database sizes are misleading and discourages such comparisons.

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

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

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

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

1.1. Benchmark Sponsor

A statement identifying the benchmark sponsor(s) and other participating companies must be provided. This benchmark was sponsored by International Business Machines Corporation.

1.2. Parameter Settings

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

- Data Base tuning options;
- Optimizer/Query execution options;
- Query Processing tool/language configuration parameters;
- Recovery/commit options;
- Consistency/locking options;
- Operating system and configuration parameters;
- Configuration parameters and options for any other software component incorporated into the pricing structure;
- Compiler optimization options.

Appendix A "Tunable Parameters" contains a list of all SYBASE parameters and operating system parameters. Session initialization parameters can be set during or immediately after establishing the connection to the database within the tpcdbatch program documented in Appendix D "Driver Source Code". This result uses the default session initialization parameters established during preprocessing/binding of the tpcdbatch program.

1.3. Configuration Diagrams

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

- Number and type of processors
- Size of allocated memory, and any specific mapping/partitioning of memory unique to the test and type of disk units (and controllers, if applicable)
- Number and type of disk units (and controllers, if applicable).
- Number of channels or bus connections to disk units, including the protocol type
- Number of LAN (e.g. Ethernet) connections, including routers, work stations, terminals, etc., that were physically used in the test or are incorporated into the pricing structure
- Type and run-time execution location of software components (e.g. DBMS, query processing tools/languages, middle-ware components, software drivers, etc.)
IBM Power 595 Model 9119-FHA Benchmark Configuration

The system was an IBM Power 595 Model 9119-FHA with

- 64 5.0GHz POWER6 Cores with 32MB L3 cache per chip
- 512 GB of memory
- 24 IBM 4Gb Dual-port Fibre Channel PCI-E adapters
- 5 146.8GB 15K RPM Internal SAS disk drives
- 288 73.4GB 15K RPM external FC disk drives
- 48 EXP810 disk enclosures
- 12 IBM TotalStorage DS4800 dual controllers

The tested and priced configurations differ in disk capacity. For full details of the priced configuration, see the pricing spreadsheet in the Executive Summary.
2.0 Clause 2: Logical Database Design Related Items

Appendix B "Programs and Scripts" contains the programs and input files used to load the test database. The test and qualification databases use the same table definitions, indices and partitioning methods. Thus, the buildtpcd script documented in Appendix B was used for both the qualification and test databases except that different input files were used to define the tablespace devices and sizes.

2.1. Table Definitions

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

Appendix B "Programs and Scripts" contains the table definitions and the programs to load the database.

2.2. Database Organization

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

Appendix B "Programs and Scripts" contains the DDL for the table and index definitions.

2.3. Horizontal Partitioning

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

Horizontal partitioning was used for lineitem and orders tables see Appendix B "Programs and Scripts".

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.
3.0 Clause 3: Queries and Refresh Functions

3.1. Query Language

The query language used to implement the queries must be identified (e.g., “RALF/SQL-Plus”). SQL was the query language used.

3.2. Verification for the Random Number Generator

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

The supplied QGEN version 2.8.0 and DBGEN 2.8.0 were used.

3.3. Substitution Parameters

3.3.1. Method of Generation

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 2.8.0 was used to generate the substitution parameters.

3.4. Query Text

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.1 "Qualification Query Output" contains the output for each of the queries.

The functional query definitions and variants used in this disclosure use the following minor query modifications.

1. In Q1, Q4, Q5, Q6, Q10, Q12, Q14, Q15 and Q20, the “dateadd” function is used to perform date arithmetic
2. In Q7, Q8 and Q9, the “datepart” function is used to extract part of a date (e.g., “year”).
3. In Q2, Q3, Q10, Q18 and Q21, the “top” function is used to restrict the number of output rows.
4. The semicolon ‘;’ is used as a command delimiter
3.5. **Query Substitution Parameters and Seeds**

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

Appendix C2, "Query Substitution Parameters" contains the query substitution parameters used in the performance tests.

3.6. **Isolation Level**

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

The isolation level used to run the queries was repeatable read.

3.7. **Refresh Functions**

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

The refresh functions are part of the implementation specific layer/driver code included in Appendix B.
4.0 Clause 4: Database System Properties

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

All ACID tests were conducted according to specification. The Atomicity, Isolation, Consistency and Durability tests were performed on the IBM Power 595 Model 9119-FHA. Appendix E. "Acid Transaction Source Code" contains the source code for the ACID transaction and query.

4.1. Atomicity Requirements

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.

4.1.1. Atomicity of Completed Transaction

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

The following steps were performed to verify the atomicity of completed transactions:

1. The total price from the ORDER table and the extended price from the LINEITEM table were retrieved for a random Orderkey. The number of records in the HISTORY table was also retrieved.
2. The ACID transaction T1 was executed for the Orderkey used in Step 1.
3. The ACID transaction committed.
4. The total price and the extended price were retrieved for the same orderkey used in step 1 and step 2. It was verified that: 
   
   \[
   \text{T1.EXTENDEDPRICE} = \text{OLD.EXTENDEDPRICE} + (\text{T1.DELTA} \times (\text{OLD.EXTENDEDPRICE}/\text{OLD.QUANTITY})),
   \text{T1.TOTALPRICE} = \text{OLD.TOTALPRICE} + ((\text{T1.EXTENDEDPRICE}-\text{OLD.EXTENDEDPRICE}) \times (1-\text{DISCOUNT}) \times (1+\text{TAX})),
   \]

   and that the number of records in the history table had increased by 1.

4.1.2. Atomicity of Aborted Transactions

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

The following steps were performed to verify the atomicity of the aborted ACID transaction:

1. The total price from the ORDER table and the extended price from the LINEITEM table were retrieved for a random Orderkey. The number of records in the HISTORY table was also retrieved.
2. The ACID transaction was executed for the Orderkey used in step 1.
3. The transaction was rolled back.
4. The total price and the extended price were retrieved for the same orderkey used in step 1 and step 2. It was verified that the extended price and the total price were the same as in step 1. The
The number of records in the HISTORY table was retrieved again and verified to be the same as in step 1.

4.2. Consistency Requirements

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

4.2.1. Consistency Condition

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

\[
O_{\text{TOTALPRICE}} = \text{SUM}(L_{\text{EXTENDEDPRICE}} \cdot (1 - L_{\text{DISCOUNT}}) \cdot (1 + L_{\text{TAX}})
\]

*for each ORDER and LINEITEM defined by \((O_{\text{ORDERKEY}} = L_{\text{ORDERKEY}})\)*

The following queries were executed before and after a measurement to show that the database was always in a consistent state both initially and after a measurement. Check the implementation if

\[
\text{SELECT sum ("truncate" ("truncate" (round(cast(l\_extendedprice as numeric(26,16)),2) * \((1 - round(cast(l\_discount as numeric(26,16)),2)),2) \cdot (1 + round(cast(l\_tax as numeric(26,16)),2)) , 2)) FROM lineitem WHERE l\_orderkey = o\_key;}
\]

is equal to o_total of order table.

4.2.2. Consistency Tests

*Verify that the ORDER and LINEITEM tables are initially consistent as defined in Clause 4.3.2.1, based on a random sample of at least 10 distinct values of O\_ORDERKEY.***

The queries defined in 4.3.2, "Consistency Condition" were run after initial database build and prior to executing the ACID transaction. The queries showed that the database is in a consistent state.

After executing 22 streams of 100 ACID transactions each, the queries defined in 4.3.2, "Consistency Condition" were run again. The queries showed that the database was still in a consistent state.

4.3. Isolation Requirements

4.3.1. Isolation Test 1

*This test demonstrates isolation for the read-write conflict of a read-write transaction and a read-only transaction when the read-write transaction is committed.*

The following steps were performed to satisfy the test of isolation for a read-only and a read-write committed transaction:

1. 1st session: Started an ACID transaction with a randomly selected O\_KEY, L\_KEY and DELTA. The transaction was delayed for 10 seconds just prior to the Commit.
2. 2nd session: Started an ACID query for the same O\_KEY as in the ACID transaction. The query completed without blocking and did not see any of the uncommitted changes made by the ACID transaction.
3. 1\textsuperscript{st} session: the ACID transaction resumed and successfully completed the Commit.
4.3.2. Isolation Test 2

*This test demonstrates isolation for the read-write conflict of read-write transaction and read-only transaction when the read-write transaction is rolled back.*

The following steps were performed to satisfy the test of isolation for read-only and a rolled back read-write transaction:

1. 1st session: Performed the ACID transaction for a random O_KEY, L_KEY and DELTA. The transaction was delayed for 10 seconds just prior to the Rollback.

2. 2nd session: Started an ACID query for the same O_KEY as in the ACID transaction. The query completed without blocking and did not see any of the uncommitted changes made by the ACID transaction.

3. 1st session: the ACID transaction resumed and successfully completed the Rollback.

4.3.3. Isolation Test 3

*This test demonstrates isolation for the write-write conflict of two refresh transactions when the first transaction is committed.*

The following steps were performed to verify isolation of two refresh transactions:

1. 1st session: Started an ACID transaction T1 for a randomly selected O_KEY, L_KEY and DELTA. The transaction was delayed for 30 seconds just prior to the COMMIT.

2. 2nd session: Started a second ACID transaction T2 for the same O_KEY, L_KEY, and for a randomly selected DELTA2. This transaction was forced to wait.

3. 1st session: The ACID transaction T1 was released and the Commit was executed, releasing the record. With the LINEITEM record now released, the ACID transaction T2 completed.

4. Verified that:

   \[ T2.L\_EXTENDEDPRICE = T1.L\_EXTENDEDPRICE + \]
   \[ (DELTA*(T1.L\_EXTENDEDPRICE)/T1.L\_QUANTITY) \]

4.3.4. Isolation Test 4

*This test demonstrates isolation for write-write conflict of two ACID transactions when the first transaction is rolled back.*

The following steps were performed to verify the isolation of two ACID transactions after the first one is rolled back:

1. 1st session: Started an ACID transaction T1 for a randomly selected O_KEY, L_KEY, and DELTA. The transaction was delayed for 30 seconds just prior to the rollback.

2. 2nd session: Started a second ACID transaction T2 for the same O_KEY, L_KEY used by the 1st session. This transaction was forced to wait.

3. 1st session: Rollback the ACID transaction T1. With the LINEITEM record now released, the ACID transaction T2 completed.

4. Verified that \[ T2.L\_EXTENDEDPRICE = T1.L\_EXTENDEDPRICE \]
4.3.5. Isolation Test 5

This test demonstrates the ability of read and writes transactions affecting different database tables to make progress concurrently.

1. 1st session: Started an ACID transaction, T1, for a randomly selected O_KEY, L_KEY and DELTA. The ACID transaction was suspended prior to COMMIT.
2. 2nd session: Started a second ACID transaction, T2, which selected random values of PS_PARTKEY and PS_SUPPKEY and returned all columns of the PARTSUPP table for which PS_PARTKEY and PS_SUPPKEY were equal to the selected values.
3. T2 completed.
4. T1 was allowed to complete.
5. It was verified that the appropriate rows in the ORDERS, LINEITEM and HISTORY tables have been changed.

4.3.6. Isolation Test 6

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

1. 1st session: A transaction T1, which executed TPC-H query 1 with a randomly selected DELTA was started.
2. 2nd session: Before T1 completed, an ACID transaction T2, with randomly selected values of O_KEY, L_KEY and DELTA, was started.
3. T2 completed and appropriate rows in the ORDERS, LINEITEM and HISTORY tables had been changed.
4. T1 completed executing query 1.

4.4. Durability Requirements

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

4.4.1. Permanent Failure of Single Durable Medium and Loss of System Power

These tests were combined and conducted on the qualification database. The following steps were performed:

1. The consistency test described in section 4.2.1 was verified.
2. The current count of the total number of records in the HISTORY table was determined giving hist1.
3. A test to run 200 ACID transactions on each of 10 execution streams was started such that each stream executes a different set of transactions.
4. One of the disks containing the SYBASE transaction log recovery data, database table data, and database index was removed from the enclosure after at least 100 ACID transactions had completed from each of the execution streams.

5. Because the disks were in RAID 1 configuration the applications continued running the ACID transactions.

6. A controller which was on the preferred path of database log data, database table data, and database index was removed, the preferred path automatically switched to another controller and the applications continued running the ACID transactions.

7. The system was shutdown by switching off the power for all system components.

8. The system was powered back on and rebooted, and the database was restarted.

9. Step 2 was performed giving hist2. It was verified that hist2 - hist1 was greater than or equal to the number of records in the success file.

10. Consistency condition described in 4.3.2 was verified.

4.4.2. Failure of Storage Controller, and Loss of System Power

The test was combined with 4.4.1.
5.0 Clause 5: Scaling and Database Population Related Items

5.1. Cardinality of Tables

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

The following table contains the TPC Benchmark™ H defined tables and the number of rows for each table as they existed upon build completion:

<table>
<thead>
<tr>
<th>Table</th>
<th>Rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lineitem</td>
<td>18,000,048,306</td>
</tr>
<tr>
<td>Orders</td>
<td>4,500,000,000</td>
</tr>
<tr>
<td>Customer</td>
<td>450,000,000</td>
</tr>
<tr>
<td>Supplier</td>
<td>30,000,000</td>
</tr>
<tr>
<td>Part</td>
<td>600,000,000</td>
</tr>
<tr>
<td>Partsupp</td>
<td>2,400,000,000</td>
</tr>
<tr>
<td>Nation</td>
<td>25</td>
</tr>
<tr>
<td>Region</td>
<td>5</td>
</tr>
</tbody>
</table>

5.2. Distribution of Tables and Logs

*The distribution of tables and logs across all media must be explicitly depicted for the tested and priced systems.*

Sybase IQ was configured on an IBM Power 595 Model 9119-FHA server. The system had

- 24 IBM 4Gb Fibre Channel PCI-E adapters
- 288 73.4GB external disk drives
- 5 146.8GB internal drives

The IBM Power 595 Model 9119-FHA had 12 IBM TotalStorage DS4800 dual controllers. 12 RAID-1 arrays were configured in each DS4800. Each of the controllers was accessed through two FC adapters. Permanent tables, their auxiliary data structures, database logs and temp dbspaces resided in tablespaces created on the RAID-1 arrays. See Appendix B "Programs and Scripts".

The Operating System resided on an internal disk. The database software resided on an internal disk.

5.3. Mapping of Database Partitions/Replications

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

The database was not replicated. Lineitem and orders are partitioned by year.
5.4. Implementation of RAID

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 used must be disclosed for each device. RAID level 1 was used for database tables, indexes, and recovery logs.

5.5. DBGEN Modifications

The version number, release number, modification number, and patch level of DBGEN must be disclosed. 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 standard distribution of DBGEN version 2.8.0 was used.

5.6. Database Loading

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

The database load time was 10:02:25.

5.7. Data Storage Ratio

The data storage ratio must be disclosed. It is computed by dividing the total data storage of the priced configuration (expressed in GB) by the size chosen for the test database as defined in clause 4.1.3.1. The ratio must be reported to the nearest 1/100th, rounded up.

The calculation of the data storage ratio is shown in the following table:

<table>
<thead>
<tr>
<th>Disk Type</th>
<th>Number of Disks</th>
<th>Space per Disk</th>
<th>Sub-Total Disk Space</th>
<th>Database Size</th>
<th>Data Storage Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>146GB</td>
<td>5</td>
<td>136.23 GB</td>
<td>681.15GB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4Gb FC 73.4GB</td>
<td>288</td>
<td>67.865 GB</td>
<td>19,545.12 GB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td>20,226.27 GB</td>
<td>3,000 GB</td>
<td>6.58</td>
</tr>
</tbody>
</table>
5.8. Details of Database Loading

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

Flat files for each of the tables were created using DBGEN. Appendix B "Programs and Scripts" contains the programs and input files used to load the database.

**Database Load Procedure:**

```
Create database &
tablesspaces

Create Tables

Load Tables

Create Indices

Set Configuration parameters
```
6.0 Clause 6: Performance Metrics and Execution-Rules Related Items

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

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

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. Start database server
2. RF1 Refresh Transaction
3. Stream 00 Execution
4. RF2 Refresh Transaction

6.3. Timing Intervals for Each Query and Refresh Function

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

See Numerical Quantities Summary in the Executive Summary.

6.4. Number of Streams for the Throughput Test

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

See Numerical Quantities Summary in the Executive Summary.

6.5. Start and End Date/Times for Each Query Stream

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

See Numerical Quantities Summary in the Executive Summary.

6.6. Total Elapsed Time for the Measurement Interval

The total elapsed time of the measurement interval (see Clause 5.3.6) must be reported for the throughput test.

See Numerical Quantities Summary in the Executive Summary.
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.

See Numerical Quantities Summary in the Executive Summary.

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

The timing intervals (see Clause 5.3.7) for each query of each stream and for each refresh function must be reported for the throughput test.

See Numerical Quantities Summary in the Executive Summary.

6.9. **Performance Metrics**

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

See Numerical Quantities Summary in the Executive Summary.

6.10. **The Performance Metric and Numerical Quantities from Both Runs**

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

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

<table>
<thead>
<tr>
<th>Metric Point</th>
<th>Run 1</th>
<th>Run 2</th>
<th>% Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>QppH@3000GB</td>
<td>142,790.7</td>
<td>142,600.2</td>
<td>0.13%</td>
</tr>
<tr>
<td>QthH@3000GB</td>
<td>171,607.4</td>
<td>172,479.4</td>
<td>0.51%</td>
</tr>
<tr>
<td>QphH@3000GB</td>
<td>156,537.3</td>
<td>156,829.8</td>
<td>0.19%</td>
</tr>
</tbody>
</table>

6.11. **System Activity between Tests**

Any activity on the SUT that takes place between the conclusion of Run 1 and the beginning of Run 2 must be disclosed.

The database server was restarted between runs.
7.0 Clause 7: SUT and Driver Implementation

7.1. Driver

A detailed textual 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.

Appendix D "Driver Source Code" contains the source code used for the driver and all scripts used in connection with it.

The power test and throughput test are invoked by calling run_tpch with the scale factor, the scope, and the total number of query streams specified. For the power test the power stream 0 SQL script will be executed along with the refresh functions SQL scripts. Then the throughput test followed with the specified query streams to be run, along with the refresh function streams.

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 performance tests are performed using dbisqlc and iqisql. dbisqlc and iqisql are Sybase-provided utilities which allow SQL statements to be executed against a Sybase IQ database. Both dbisqlc and iqisql utilities are invoked from the command line on the SUT. They read input from files containing SQL statements and sends results to stdout. dbisqlc uses information in the .odbc.ini file to connect to the database while iqisql uses information in interfaces file for the same.

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.

7.3. Profile-Directed Optimization

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

Profile-directed optimization was not used.
8.0 Clause 8: Pricing-Related Items

8.1. Hardware and Software Used

A detailed list of hardware and software used in the priced system must be reported. Each item must have a vendor part number, description, and release/revision level, and indicate General Availability (see Clause 7.2.2.1) either implicitly or explicitly (omitted Availability Dates default to the System Availability 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.

The detailed list of all hardware and software for the priced configuration is listed in the Executive Summary.

8.2. Three Year Cost of System Configuration

The total 3-year price of the entire configuration must be reported, including: hardware, software, hardware maintenance, and software support charges. Separate component pricing is required (see Clause 7.3.1. Pricing Spreadsheet.) Hardware maintenance and software support must be reported separately. The software support level must be disclosed separately from that of hardware, with separate pricing and discounts.

The price sheet for this disclosure is contained in the executive summary pages.

The pricing spreadsheet includes maintenance costs for 3 years. This service provides 7 days per week, 24 hours per day coverage.

Discounts are based on US list prices and for similar quantities and configurations. A discount of 49.95% has been applied to all IBM hardware, software, and services based on the total value and quantities of the components of the configuration, including full payment of all components and maintenance.

The prices listed for the IBM software products includes software support that provides the items identified in paragraph 7.1.5.6 of the TPC-H Benchmark Specification.

For assistance with any of these prices or their applicability to any customer’s requirements, please contact one of the following individuals:

Dan Hebrank, IBM Sales & Distribution, Systems & Technology Sales
email: dhebrank@us.ibm.com phone 1-314-283-4674

Das Joydeep, Sybase Inc.
email: Joydeep.Das@sybase.com phone 1-925-236-5214

8.3. System Availability Date

The System Availability Date (see Clause 7.2.2.1) must be the single availability date reported on the first page of the executive summary. The full disclosure report must report Availability Dates individually for at least each of the categories for which a pricing subtotal must be provided (see Clause
7.3.1.4). All Availability Dates required to be reported must be disclosed to a precision of 1 day, but the precise format is left to the test sponsor.

The System Availability Date is Now.
9.0 Clause 9: Audit Items

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

The auditor's attestation letter is included at the front of this report.
Appendix - A  Tunable Parameters

A.1  Sybase IQ Database Configuration

tpch.cfg
# tpch.cfg
-n tpch_15_0_sybase
-x tcpip(port=5788)

# The following parameters are also found in the configuration file
# $ASDIR/scripts/default.cfg. Any parameters not specified below
# and not in the start up parameter list, will be added by start_asiq
# using default.cfg as a guide.

-c 64M
-gd all
-gm 25
-gc 5000
-gr 5000
-gp 4096
-il 0
-iqmt 2000
-iqmc 190000
-iqtc 120000
-idtss 240
-iqnumbercpus 100
-iqpartition 64
-iqgovern 10

options.sql
SET OPTION "PUBLIC".STRING_RTRUNCATION='Off';
SET OPTION "PUBLIC".Allow_Nulls_By_Default='Off';
SET OPTION "PUBLIC".Append_Load='On';
SET OPTION "PUBLIC".Force_No_Scroll_Cursors='On';
SET OPTION "PUBLIC".Garray_Fill_Factor_Percent=3;
SET OPTION "PUBLIC".Load_Memory_Mb=0;
SET OPTION "PUBLIC".Max_IQ_Threads_Per_Connection=500;
SET OPTION "PUBLIC".Minimize_Storage='On';
SET OPTION "PUBLIC".Notify_Modulus=10000000;
SET OPTION "PUBLIC".Row_Counts='On';
SET OPTION "PUBLIC".Sweeper_Threads_Percent=8;
SET OPTION "PUBLIC".Wash_Area_Buffers_Percent = '20';
SET OPTION "PUBLIC".Prefetch_Threads_Percent = 15;
SET OPTION "PUBLIC".Max_Hash_Rows=18000000;
SET OPTION "PUBLIC".Default_Having_Selectivity_PPM = 40;
SET OPTION "PUBLIC".Hash_Thrashing_Percent=100;
SET OPTION "PUBLIC".QUERY_TEMP_SPACE_LIMIT=0;
SET OPTION "PUBLIC".subquery_flattening_preference=3;
SET OPTION "PUBLIC".Max_Query_Parallelism=100;
SET OPTION "PUBLIC".FP_PREDICATE_WORKUNIT_PAGES=50;
SET OPTION "PUBLIC".ROW_PREFETCH_SIZE=40;

.odbc.ini
[ODBC Data Source]
SybaseIQ=Sybase IQ Driver

[tpch_15_0]
Userid=DBA
Password=sql
EngineName=tpch_15_0_sybase
DatabaseName=tpch
DatabaseFile=tpch.db
CommLinks=tcpiip(host=tpch2:port=5788)
AutoStop=no

A.2  AIX Parameters

chdev -l sys0 -a maxuproc=8000
ioo -r -o lvm_bufcnt=16
LDR_CNTRL=TEXTPSIZE=64K@DATAPSIZE=64K@STACKPSIZE=64K
Appendix - B  Programs and Scripts

B.1  create_database.sql

CREATE DATABASE '/testdb/tpch.db'
TRANSACTION LOG ON
COLLATION 'ISO_BINENG'
CASE RESPECT
PAGE SIZE 4096
BLANK PADDING ON
JAVA ON
JCONNECT ON
IQ PATH '/testdb/M001'
IQ PAGE SIZE 524288
TEMPORARY PATH '/testdb/T001';

B.2  create_dbspaces_15_0.sh

-- Notation:
-- IQ MAIN DBSPACES will be named as iq[0-9]+
-- and
-- IQ TEMP DBSPACES will be named as iqtmp[0-9]+
--
-- This notation is required to help in reporting the main
-- and temp dbspaces in the kit.
--
B.2 create_dbspaces_15_0.sh

alter dbspace IQ_SYSTEM_MAIN ADD
FILE iq2 '/testdb/M002',
FILE iq3 '/testdb/M003',
FILE iq4 '/testdb/M004',
FILE iq5 '/testdb/M005',
FILE iq6 '/testdb/M006',
FILE iq7 '/testdb/M007',
FILE iq8 '/testdb/M008',
FILE iq9 '/testdb/M009',
FILE iq10 '/testdb/M010',
FILE iq11 '/testdb/M011',
FILE iq12 '/testdb/M012',
FILE iq13 '/testdb/M013',
FILE iq14 '/testdb/M014',
FILE iq15 '/testdb/M015',
FILE iq16 '/testdb/M016',
FILE iq17 '/testdb/M017',
FILE iq18 '/testdb/M018',
FILE iq19 '/testdb/M019',
FILE iq20 '/testdb/M020',
FILE iq21 '/testdb/M021',
FILE iq22 '/testdb/M022',
FILE iq23 '/testdb/M023',
FILE iq24 '/testdb/M024',
FILE iq25 '/testdb/M025',
FILE iq26 '/testdb/M026',
FILE iq27 '/testdb/M027',
FILE iq28 '/testdb/M028',
FILE iq29 '/testdb/M029',
FILE iq30 '/testdb/M030',
FILE iq31 '/testdb/M031',
FILE iq32 '/testdb/M032',
FILE iq33 '/testdb/M033',
FILE iq34 '/testdb/M034',
FILE iq35 '/testdb/M035',
FILE iq36 '/testdb/M036',
FILE iq37 '/testdb/M037',
FILE iq38 '/testdb/M038',
FILE iq39 '/testdb/M039',
FILE iq40 '/testdb/M040',
FILE iq41 '/testdb/M041',
FILE iq42 '/testdb/M042',
FILE iq43 '/testdb/M043',
FILE iq44 '/testdb/M044',
FILE iq45 '/testdb/M045',
FILE iq46 '/testdb/M046',
FILE iq47 '/testdb/M047',
FILE iq48 '/testdb/M048',
FILE iq49 '/testdb/M049',
FILE iq50 '/testdb/M050',
FILE iq51 '/testdb/M051',
FILE iq52 '/testdb/M052',
FILE iq53 '/testdb/M053',
FILE iq54 '/testdb/M054',
FILE iq55 '/testdb/M055',
FILE iq56 '/testdb/M056',
FILE iq57 '/testdb/M057',
FILE iq58 '/testdb/M058',
FILE iq59 '/testdb/M059',
FILE iq60 '/testdb/M060',
FILE iq61 '/testdb/M061',
FILE iq62 '/testdb/M062',
FILE iq63 '/testdb/M063',
FILE iq64 '/testdb/M064',
FILE iq65 '/testdb/M065',
FILE iq66 '/testdb/M066',
FILE iq67 '/testdb/M067',
FILE iq68 '/testdb/M068',
FILE iq69 '/testdb/M069',
FILE iq70 '/testdb/M070',
FILE iq71 '/testdb/M071',
FILE iq72 '/testdb/M072',
FILE iq73 '/testdb/M073',
FILE iq74 '/testdb/M074',
FILE iq75 '/testdb/M075',
FILE iq76 '/testdb/M076',
FILE iq77 '/testdb/M077',
FILE iq78 '/testdb/M078',
FILE iq79 '/testdb/M079',
FILE iq80 '/testdb/M080',
FILE iq81 '/testdb/M081',
FILE iq82 '/testdb/M082',
FILE iq83 '/testdb/M083',
FILE iq84 '/testdb/M084',
FILE iq85 '/testdb/M085',
FILE iq86 '/testdb/M086',
FILE iq87 '/testdb/M087',
FILE iq88 '/testdb/M088',
FILE iq89 '/testdb/M089',
FILE iq90 '/testdb/M090',
FILE iq91 '/testdb/M091',
FILE iq92 '/testdb/M092',
FILE iq93 '/testdb/M093',
FILE iq94 '/testdb/M094',
FILE iq95 '/testdb/M095',
FILE iq96 '/testdb/M096',
FILE iq97 '/testdb/M097',
FILE iq98 '/testdb/M098',
FILE iq99 '/testdb/M099',
FILE iq100 '/testdb/M100',
FILE iq101 '/testdb/M101',
FILE iq102 '/testdb/M102',
FILE iq103 '/testdb/M103',

---
FILE iq104 '/testdb/M104',
FILE iq105 '/testdb/M105',
FILE iq106 '/testdb/M106',
FILE iq107 '/testdb/M107',
FILE iq108 '/testdb/M108',
FILE iq109 '/testdb/M109',
FILE iq110 '/testdb/M110',
FILE iq111 '/testdb/M111',
FILE iq112 '/testdb/M112',
FILE iq113 '/testdb/M113',
FILE iq114 '/testdb/M114',
FILE iq115 '/testdb/M115',
FILE iq116 '/testdb/M116',
FILE iq117 '/testdb/M117',
FILE iq118 '/testdb/M118',
FILE iq119 '/testdb/M119',
FILE iq120 '/testdb/M120',
FILE iq121 '/testdb/M121',
FILE iq122 '/testdb/M122',
FILE iq123 '/testdb/M123',
FILE iq124 '/testdb/M124',
FILE iq125 '/testdb/M125',
FILE iq126 '/testdb/M126',
FILE iq127 '/testdb/M127',
FILE iq128 '/testdb/M128',
FILE iq129 '/testdb/M129',
FILE iq130 '/testdb/M130',
FILE iq131 '/testdb/M131',
FILE iq132 '/testdb/M132',
FILE iq133 '/testdb/M133',
FILE iq134 '/testdb/M134',
FILE iq135 '/testdb/M135',
FILE iq136 '/testdb/M136',
FILE iq137 '/testdb/M137',
FILE iq138 '/testdb/M138',
FILE iq139 '/testdb/M139',
FILE iq140 '/testdb/M140',
FILE iq141 '/testdb/M141',
FILE iq142 '/testdb/M142',
FILE iq143 '/testdb/M143',
FILE iq144 '/testdb/M144',

alter dbspace IQ_SYSTEM_TEMP ADD
FILE iqtmp28 '/testdb/T028',
FILE iqtmp29 '/testdb/T029',
FILE iqtmp30 '/testdb/T030',
FILE iqtmp31 '/testdb/T031',
FILE iqtmp32 '/testdb/T032',
FILE iqtmp33 '/testdb/T033',
FILE iqtmp34 '/testdb/T034',
FILE iqtmp35 '/testdb/T035',
FILE iqtmp36 '/testdb/T036',
FILE iqtmp37 '/testdb/T037',
FILE iqtmp38 '/testdb/T038',
FILE iqtmp39 '/testdb/T039',
FILE iqtmp40 '/testdb/T040',
FILE iqtmp41 '/testdb/T041',
FILE iqtmp42 '/testdb/T042',
FILE iqtmp43 '/testdb/T043',
FILE iqtmp44 '/testdb/T044',
FILE iqtmp45 '/testdb/T045',
FILE iqtmp46 '/testdb/T046',
FILE iqtmp47 '/testdb/T047',
FILE iqtmp48 '/testdb/T048',
FILE iqtmp49 '/testdb/T049',
FILE iqtmp50 '/testdb/T050',
FILE iqtmp51 '/testdb/T051',
FILE iqtmp52 '/testdb/T052',
FILE iqtmp53 '/testdb/T053',
FILE iqtmp54 '/testdb/T054',
FILE iqtmp55 '/testdb/T055',
FILE iqtmp56 '/testdb/T056',
FILE iqtmp57 '/testdb/T057',
FILE iqtmp58 '/testdb/T058',
FILE iqtmp59 '/testdb/T059',
FILE iqtmp60 '/testdb/T060',
FILE iqtmp61 '/testdb/T061',
FILE iqtmp62 '/testdb/T062',
FILE iqtmp63 '/testdb/T063',
FILE iqtmp64 '/testdb/T064',
FILE iqtmp65 '/testdb/T065',
FILE iqtmp66 '/testdb/T066',
FILE iqtmp67 '/testdb/T067',
FILE iqtmp68 '/testdb/T068',
FILE iqtmp69 '/testdb/T069',
FILE iqtmp70 '/testdb/T070',
FILE iqtmp71 '/testdb/T071',
FILE iqtmp72 '/testdb/T072',
FILE iqtmp73 '/testdb/T073',
FILE iqtmp74 '/testdb/T074',
FILE iqtmp75 '/testdb/T075',
FILE iqtmp76 '/testdb/T076',
FILE iqtmp77 '/testdb/T077',
FILE iqtmp78 '/testdb/T078',
FILE iqtmp79 '/testdb/T079',
FILE iqtmp80 '/testdb/T080',
FILE iqtmp81 '/testdb/T081',
FILE iqtmp82 '/testdb/T082',
FILE iqtmp83 '/testdb/T083',
FILE iqtmp84 '/testdb/T084',
FILE iqtmp85 '/testdb/T085',
FILE iqtmp86 '/testdb/T086',
FILE iqtmp87 '/testdb/T087',
FILE iqtmp88 '/testdb/T088',
FILE iqtmp89 '/testdb/T089',
FILE iqtmp90 '/testdb/T090',
FILE iqtmp91 '/testdb/T091',
FILE iqtmp92 '/testdb/T092',
FILE iqtmp93 '/testdb/T093',
FILE iqtmp94 '/testdb/T094',
FILE iqtmp95 '/testdb/T095',
FILE iqtmp96 '/testdb/T096',

TPC Benchmark™ H FDR – IBM System Power 595 with Sybase IQ Single Application Server Edition v.15.1 ESD #1.2- Page 25
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 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 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 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 TABLE orders
(
    o_orderkey unsigned bigint,
    o_custkey unsigned int,
    o_orderstatus char(1),
    PRIMARY KEY (o_orderkey)
);
CREATE TABLE orders
(
  o_orderkey          unsigned bigint,
  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)
)
partition by range (o_orderdate)
(p1 values <= ('1992-12-31'),
p2 values <= ('1993-12-31'),
p3 values <= ('1994-12-31'),
p4 values <= ('1995-12-31'),
p5 values <= ('1996-12-31'),
p6 values <= ('1997-12-31'),
p7 values <= ('1998-12-31'),
p8 values <= (MAX));

CREATE TABLE lineitem
(
  l_orderkey             unsigned bigint,
  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)
)
partition by range(l_shipdate)
(p1 values <= ('1992-12-31'),
p2 values <= ('1993-12-31'),
p3 values <= ('1994-12-31'),
p4 values <= ('1995-12-31'),
p5 values <= ('1996-12-31'),
p6 values <= ('1997-12-31'),
p7 values <= ('1998-12-31'),
p8 values <= (MAX));

CREATE INDEX o_orderdate_date ON orders(o_orderdate) ;
CREATE INDEX l_shipdate_date ON lineitem(l_shipdate) ;
CREATE INDEX l_receiptdate_date ON lineitem(l_receiptdate);
CREATE HG index n_regionkey_hg on nation ( n_regionkey );
CREATE HG index s_nationkey_hg on supplier ( s_nationkey );
CREATE HG index c_nationkey_hg on customer ( c_nationkey );
CREATE HG index o_custkey_hg on orders ( o_custkey );
CREATE HG index p_partkey_hg on partsupp ( p_partkey );
CREATE HG index l_partkey_hg on lineitem ( l_partkey );
CREATE HG index ps_suppkey_hg on partsupp ( ps_suppkey );
CREATE HG index l_suppkey_hg on lineitem ( l_suppkey );
CREATE HG index l_orderkey_hg on lineitem ( l_orderkey );

B.4 create_refresh_functions.sql

--------Create RF1--------

CREATE PROCEDURE DBA.tpch_rf1 (IN data_directory varchar(128),
  IN stream_number varchar(3))
ON EXCEPTION RESUME
BEGIN
  DECLARE sql_cmd long varchar;
  DECLARE c_lf varchar(2);
  DECLARE rf1_start timestamp;
  DECLARE rf1_stop timestamp;
  DECLARE n_seconds numeric(16,5);
  DECLARE cur_sqlstate CHAR(5);
  SET c_lf=char(10);
  SET rf1_start = dateformat(now() ,'yyyy-Mm-Dd hh:nn:ss.sss');
  SELECT 'Stream'||stream_number||' RF1 START TIME --,
    dateformat(rf1_start,'yyyy-mm-dd hh:nn:ss.sss');
  SET sql_cmd='load table orders ( '+c_lf;
  SET sql_cmd=sql_cmd+' o_orderkey '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' o_custkey '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' o_orderstatus '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' o_totalprice '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' o_orderdate date('+char(39)+'YYYY-MM- DD'+char(39)+'), filler(1), '+c_lf;
  SET sql_cmd=sql_cmd+' o_orderpriority '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' o_clerk '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' o_shippriority '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' o_comment '+char(39)+'|'+char(39)+' ) '+c_lf;
  SET sql_cmd=sql_cmd+'from
    '+char(39)+data_directory+'orders.tbl.u'+stream_number+char(39)+c_lf;
  SET sql_cmd=sql_cmd+'row delimited by
    '+char(39)+'\x0a'+char(39)+' quotes off escapes off preview on;';
  EXECUTE IMMEDIATE with quotes on sql_cmd;
  SELECT SQLSTATE INTO cur_sqlstate;
  IF cur_sqlstate != '00000' THEN
    ROLLBACK;
    RAISERROR 23002 'RF1 failed while inserting into orders with
      SQLSTATE: ', cur_sqlstate;
    RETURN(1);
  END IF;
  -----------insert into lineitem---------
  SELECT * INTO lineitem
  FROM orders
  WHERE
    l_orderkey = o_orderkey;
  SET sql_cmd='load table lineitem ( '+c_lf;
  SET sql_cmd=sql_cmd+' l_orderkey '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_partkey '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_suppkey '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_linenumber '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_quantity '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_extendedprice '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_discount '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_tax '+char(39)+'|'+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_returnflag '+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_linestatus '+char(39)+', '+c_lf;
  SET sql_cmd=sql_cmd+' l_shipdate date('+char(39)+'YYYY-MM- DD'+char(39)+'), filler(1), '+c_lf;
  SET sql_cmd=sql_cmd+' l_commitdate date, '+c_lf;
  SET sql_cmd=sql_cmd+' l_receiptdate date, '+c_lf;
  SET sql_cmd=sql_cmd+' l_shipinstruct char(25), '+c_lf;
  SET sql_cmd=sql_cmd+' l_shipmode char(10), '+c_lf;
  SET sql_cmd=sql_cmd+' l_comment varchar(44) ) '+c_lf;
  SET sql_cmd=sql_cmd+'from
    '+char(39)+data_directory+'orders.tbl.u'+stream_number+char(39)+c_lf;
  SET sql_cmd=sql_cmd+'row delimited by
    '+char(39)+'\x0a'+char(39)+' quotes off escapes off preview on;';
  EXECUTE IMMEDIATE with quotes on sql_cmd;
  SELECT SQLSTATE INTO cur_sqlstate;
  IF cur_sqlstate != '00000' THEN
    ROLLBACK;
    RAISERROR 23002 'RF1 failed while inserting into orders with
      SQLSTATE: ', cur_sqlstate;
    RETURN(1);
  END IF;
END IF;

TPC Benchmark™ H FDR – IBM System Power 595 with Sybase IQ Single Application Server Edition v.15.1 ESD #1.2- Page 27
SET sql_cmd=sql_cmd+' l_commitdate date('+char(39)+'YYYY-MM-DD'+char(39)+'), filler(1), '+c_lf;
SET sql_cmd=sql_cmd+' l_receiptdate date('+char(39)+'YYYY-MM-DD'+char(39)+'), filler(1), '+c_lf;
SET sql_cmd=sql_cmd+' l_shipinstruct '+char(39)+'|'+char(39)+', '+c_lf;
SET sql_cmd=sql_cmd+' l_shipmode '+char(39)+'|'+char(39)+', '+c_lf;
SET sql_cmd=sql_cmd+' l_comment '+char(39)+'|'+char(39)+' )'+c_lf;
SET sql_cmd=sql_cmd+'from
'+char(39)+data_directory+'lineitem.tbl.u'+stream_number+char(39)+c_lf;
SET sql_cmd=sql_cmd+'row delimited by
'+char(39)+'\x0a'+char(39)+c_lf+'quotes off escapes off preview on;';
EXECUTE IMMEDIATE with quotes on sql_cmd;

SELECT SQLSTATE INTO cur_sqlstate;
IF cur_sqlstate != '00000' THEN
ROLLBACK;
RAISERROR 23002 'RF1 failed while inserting into lineitem with
SQLSTATE: ', cur_sqlstate;
RETURN(1);
END IF;
COMMIT;

---------Find the execution time of rf1.----------
SET n_seconds=cast(datediff(millisecond,rf1_start,rf1_stop) AS numeric(16,5))/1000;
SET sql_cmd='Stream_'+stream_number+' RF1 Elapsed time --,
'+cast(n_seconds AS varchar(20))+ ' seconds' ;
SELECT sql_cmd;
RETURN(0);
END;

--------- Create RF2-------------------------
CREATE PROCEDURE DBA.tpch_rf2 (in data_directory varchar(128),
in stream_number var char(3))
ON exception resume
BEGIN
DECLARE sql_cmd long varchar;
DECLARE c_lf varchar(2);
DECLARE rf2_start timestamp;
DECLARE rf2_stop timestamp;
DECLARE n_seconds numeric(16,5);
DECLARE cur_sqlstate CHAR(5);
SET c_lf=char(10);
SET rf2_start = dateformat(now(*) ,'yyyy-Mm-Dd hh:nn:ss.sss');
SELECT 'Stream'||stream_number||' RF2 START TIME --,
'[dateformat(rf2_start,'yyyy-mm-dd hh:nn:ss.sss')];
CREATE TABLE #delete_table ( d_orderkey UNSIGNED BIGINT );
SET sql_cmd=sql_cmd+'
load table #delete_table ( d_orderkey
'+char(39)+')'+c_lf;
SET sql_cmd=sql_cmd+'from
'+char(39)+data_directory+'delete.'+stream_number+char(39)+c_lf;
SET sql_cmd=sql_cmd+'row delimited by
'+char(39)+'\x0a'+char(39)+c_lf+'quotes off escapes off preview on;';
EXECUTE IMMEDIATE with quotes on sql_cmd;

DELETE lineitem FROM lineitem
WHERE l_orderkey in (select d_orderkey from  #delete_table);
SELECT SQLSTATE INTO cur_sqlstate;
IF cur_sqlstate != '00000' THEN
ROLLBACK;
SET sql_cmd='RF2 failed at Step 2 with SQLSTATE: '+cur_sqlstate;
RAISERROR 23002 sql_cmd;
RETURN(1);
END IF;

DELETE FROM orders
WHERE o_orderkey in (select d_orderkey from  #delete_table);
SELECT SQLSTATE INTO cur_sqlstate;
IF cur_sqlstate != '00000' THEN
ROLLBACK;
SET sql_cmd='RF2 failed at Step 3 with SQLSTATE: '+cur_sqlstate;
RAISERROR 23002 sql_cmd;
RETURN(1);
END IF;

-----Delete from Orders-----------
DROP TABLE #delete_table;
SET rf2_stop = dateformat(now(*) ,'yyyy-Mm-Dd hh:nn:ss.sss');
SELECT 'Stream'||stream_number||' RF2 STOP TIME --,
'[dateformat(rf2_stop,'yyyy-mm-dd hh:nn:ss.sss')];
--------- Calculate execution time for rf2.-------
SET n_seconds=cast(datediff(millisecond,rf2_start,rf2_stop) as numeric(16,5))/1000;
SET sql_cmd='Stream_'+stream_number+' RF2 Elapsed time --,
'+cast(n_seconds as varchar(20))+ ' seconds' ;
SELECT sql_cmd;
RETURN(0);
END;

B.5  **load_region.sql**

LOAD TABLE REGION ( R_REGIONKEY             | ,
R_NAME                         | ,
R_COMMENT                      | )
FROM '/rawdata/region.tbl'

B.6  **load_nation.sql**

LOAD TABLE NATION ( N_NATIONKEY                    | ,
N_NAME                         | ,
N_REGIONKEY                    | ,
N_COMMENT                      | )
FROM '/rawdata/nation.tbl'
### B.7 load_customer.sql

```sql
LOAD TABLE CUSTOMER (  
    C_CUSTKEY                  '|',  
    C_NAME                    '|',  
    C_ADDRESS                  '|',  
    C_NATIONKEY                '|',  
    C_PHONE                    '|',  
    C_ACCTBAL                  '|',  
    C_MKTSEGMENT               '|',  
    C_COMMENT                  '|',  
)  
FROM '/rawdata/customer.tbl.1','/rawdata/customer.tbl.2',  
'/rawdata/customer.tbl.3','/rawdata/customer.tbl.4',  
'/rawdata/customer.tbl.5','/rawdata/customer.tbl.6',  
'/rawdata/customer.tbl.7','/rawdata/customer.tbl.8',  
'/rawdata/customer.tbl.9','/rawdata/customer.tbl.10',  
'/rawdata/customer.tbl.11','/rawdata/customer.tbl.12'  
WITH CHECKPOINT ON;  
commit;
```

### B.8 load_part.sql

```sql
LOAD TABLE PART (  
    P_PARTKEY                 '|',  
    P_NAME                    '|',  
    P_MFGR                    '|',  
    P_BRAND                   '|',  
    P_TYPE                    '|',  
    P_SIZE                    '|',  
    P_CONTAINER                '|',  
    P_RETAILPRICE              '|',  
    P_COMMENT                  '|'  
)  
FROM '/rawdata/part.tbl.1','/rawdata/part.tbl.2',  
'/rawdata/part.tbl.3','/rawdata/part.tbl.4',  
'/rawdata/part.tbl.5','/rawdata/part.tbl.6',  
'/rawdata/part.tbl.7','/rawdata/part.tbl.8',  
'/rawdata/part.tbl.9','/rawdata/part.tbl.10',  
'/rawdata/part.tbl.11','/rawdata/part.tbl.12'  
WITH CHECKPOINT ON;  
commit;
```

### B.9 load_supplier.sql

```sql
LOAD TABLE SUPPLIER (  
    S_SUPPKEY                  '|',  
    S_NAME                    '|',  
    S_ADDRESS                  '|',  
    S_NATIONKEY                '|',  
    S_PHONE                    '|',  
    S_ACCTBAL                  '|',  
    S_MKTSEGMENT               '|',  
    S_COMMENT                  '|'  
)  
FROM '/rawdata/supplier.tbl.1','/rawdata/supplier.tbl.2',  
'/rawdata/supplier.tbl.3','/rawdata/supplier.tbl.4',  
'/rawdata/supplier.tbl.5','/rawdata/supplier.tbl.6',  
'/rawdata/supplier.tbl.7','/rawdata/supplier.tbl.8',  
'/rawdata/supplier.tbl.9','/rawdata/supplier.tbl.10',  
'/rawdata/supplier.tbl.11','/rawdata/supplier.tbl.12'  
WITH CHECKPOINT ON;  
commit;
```

### B.10 load_partsupp.sql

```sql
LOAD TABLE PARTSUPP (  
    PS_PARTKEY                 '|',  
    PS_SUPPKEY                 '|',  
    PS_AVAILQTY                '|',  
    PS_SUPPLYCOST              '|',  
    PS_COMMENT                  '|'  
)  
FROM '/rawdata/partsupp.tbl.1','/rawdata/partsupp.tbl.2',  
'/rawdata/partsupp.tbl.3','/rawdata/partsupp.tbl.4',  
'/rawdata/partsupp.tbl.5','/rawdata/partsupp.tbl.6',  
'/rawdata/partsupp.tbl.7','/rawdata/partsupp.tbl.8',  
'/rawdata/partsupp.tbl.9','/rawdata/partsupp.tbl.10',  
'/rawdata/partsupp.tbl.11','/rawdata/partsupp.tbl.12',  
'/rawdata/partsupp.tbl.13','/rawdata/partsupp.tbl.14',  
'/rawdata/partsupp.tbl.15','/rawdata/partsupp.tbl.16',  
'/rawdata/partsupp.tbl.17','/rawdata/partsupp.tbl.18',  
'/rawdata/partsupp.tbl.19','/rawdata/partsupp.tbl.20',  
'/rawdata/partsupp.tbl.21','/rawdata/partsupp.tbl.22',  
'/rawdata/partsupp.tbl.23','/rawdata/partsupp.tbl.24',  
'/rawdata/partsupp.tbl.25','/rawdata/partsupp.tbl.26',  
'/rawdata/partsupp.tbl.27','/rawdata/partsupp.tbl.28',  
'/rawdata/partsupp.tbl.29','/rawdata/partsupp.tbl.30',  
'/rawdata/partsupp.tbl.31','/rawdata/partsupp.tbl.32',  
'/rawdata/partsupp.tbl.33','/rawdata/partsupp.tbl.34',  
'/rawdata/partsupp.tbl.35','/rawdata/partsupp.tbl.36'  
WITH CHECKPOINT ON;  
commit;
```

### B.11 load_orders.sql

```sql
LOAD TABLE ORDERS (  
    O_ORDERKEY               '|',  
    O_CUSTKEY                 '|',  
    O_ORDERSTATUS             '|',  
    O_TOTALPRICE              '|',  
    O_ORDERDATE                '|',  
    O_ORDERPRIORITY            '|',  
    O_CLERK                     '|'  
)  
FROM '/rawdata/orders.tbl.1','/rawdata/orders.tbl.2',  
'/rawdata/orders.tbl.3','/rawdata/orders.tbl.4',  
'/rawdata/orders.tbl.5','/rawdata/orders.tbl.6',  
'/rawdata/orders.tbl.7','/rawdata/orders.tbl.8',  
'/rawdata/orders.tbl.9','/rawdata/orders.tbl.10',  
'/rawdata/orders.tbl.11','/rawdata/orders.tbl.12'  
WITH CHECKPOINT ON;  
commit;
```
B.12 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_COMMITDATE,  
  L_RECEIPTDATE,  
  L_SHIPMODE,  
  L_SHIPDATE,  
  L_SHIPINSTRUCT,  
  L_SHIPMODE,  
  L_COMMENT)
FROM '/rawdata/lineitem.tbl.1','/rawdata/lineitem.tbl.2',  
'/rawdata/lineitem.tbl.3','/rawdata/lineitem.tbl.4',  
'/rawdata/lineitem.tbl.5','/rawdata/lineitem.tbl.6',  
'/rawdata/lineitem.tbl.7','/rawdata/lineitem.tbl.8',  
'/rawdata/lineitem.tbl.9','/rawdata/lineitem.tbl.10',  
'/rawdata/lineitem.tbl.11','/rawdata/lineitem.tbl.12',  
'/rawdata/lineitem.tbl.13','/rawdata/lineitem.tbl.14',  
'/rawdata/lineitem.tbl.15','/rawdata/lineitem.tbl.16',  
'/rawdata/lineitem.tbl.17','/rawdata/lineitem.tbl.18',  
'/rawdata/lineitem.tbl.19','/rawdata/lineitem.tbl.20',  
'/rawdata/lineitem.tbl.21','/rawdata/lineitem.tbl.22',  
'/rawdata/lineitem.tbl.23','/rawdata/lineitem.tbl.24',  
'/rawdata/lineitem.tbl.25','/rawdata/lineitem.tbl.26',  
'/rawdata/lineitem.tbl.27','/rawdata/lineitem.tbl.28',  
'/rawdata/lineitem.tbl.29','/rawdata/lineitem.tbl.30',  
'/rawdata/lineitem.tbl.31','/rawdata/lineitem.tbl.32',  
'/rawdata/lineitem.tbl.33','/rawdata/lineitem.tbl.34',  
'/rawdata/lineitem.tbl.35','/rawdata/lineitem.tbl.36'

escapes off  
quotes off  
row delimited by '\x0a'
WITH CHECKPOINT ON;
commit;

B.13 update_power.sql

-- THIS IS A GENERATED FILE.  
-- PLEASE CHANGE gen_update_power.sh FOR MODIFICATIONS
create variable c_path varchar(128);
create variable lineitem_start bigint;
create variable lineitem_end bigint;
create variable lineitem_diff bigint;
create variable orders_start bigint;
create variable orders_end bigint;
create variable orders_diff bigint;
create variable lineitem_start bigint;
create variable lineitem_end bigint;
create variable lineitem_diff bigint;
create variable orders_start bigint;
create variable orders_end bigint;
create variable orders_diff bigint;
set c_path="/testdb/data/
set lineitem_start=(select count(*) from lineitem);
set lineitem_end=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
set orders_end=(select count(*) from orders);

set lineitem_start=(select count(*) from lineitem);
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_end - lineitem_start;
set orders_diff = orders_end - orders_start;
select 'In lineitem RF1 Inserted ',lineitem_diff;
select 'In lineitem RF2 Deleted ',lineitem_diff;
select 'Stream 0 RF WAITING -- ,', dateformat (now(), 'yyyy-Mm-Dd hh:mm:ss.sss')
!!check_stream.sh

-- Sleep Until the query stream completes
select 'Stream 0 RF WAITING -- ,'
-- Stream execution is completed.
set lineitem_start=(select count(*) from lineitem);
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_end - lineitem_start;
set orders_diff = orders_end - orders_start;
select 'In lineitem RF1 Inserted ',lineitem_diff;
select 'In lineitem RF2 Deleted ',lineitem_diff;

!!check_stream.sh
-- Stream execution is completed.
set lineitem_start=(select count(*) from lineitem);
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_end - lineitem_start;
set orders_diff = orders_end - orders_start;
select 'In lineitem RF1 Inserted ',lineitem_diff;
select 'In lineitem RF2 Deleted ',lineitem_diff;

!!check_stream.sh
-- Stream execution is completed.
set lineitem_start=(select count(*) from lineitem);
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_end - lineitem_start;
set orders_diff = orders_end - orders_start;
select 'In lineitem RF1 Inserted ',lineitem_diff;
select 'In lineitem RF2 Deleted ',lineitem_diff;

!!check_stream.sh
-- Stream execution is completed.
B.14 update_throughput.sql

-- THIS IS A GENERATED FILE.
-- PLEASE CHANGE gen_update_throughput.sh FOR MODIFICATIONS
create variable c_path varchar(128);
create variable qstart timestamp;
create variable qstop timestamp;
create variable n_seconds numeric(16,5);
create variable lineitem_start bigint;
create variable lineitem_end bigint;
create variable lineitem_diff bigint;
create variable orders_start bigint;
create variable orders_end bigint;
create variable orders_diff bigint;
go

set qstart = now(*)
set c_path='/testdb/data/'
select @@servername, db_name()
go

select 'Throughput Stream1 starts';
set lineitem_start=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
go
call tpch_rf1 (c_path,2)
go
set lineitem_end=(select count(*) from lineitem);
set orders_end=(select count(*) from orders);
set lineitem_diff = lineitem_end - lineitem_start;
select 'In lineitem Rf1 Inserted ',lineitem_diff;
go
call tpch_rf2 (c_path,2)
go
set lineitem_start=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
go
call tpch_rf1 (c_path,3)
go
set lineitem_end=(select count(*) from lineitem);
set orders_end=(select count(*) from orders);
set lineitem_diff = lineitem_end - lineitem_start;
select 'In lineitem Rf1 Inserted ',lineitem_diff;
go
call tpch_rf2 (c_path,3)
go
set lineitem_start=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
go
call tpch_rf1 (c_path,4)
go
set lineitem_end=(select count(*) from lineitem);
set orders_end=(select count(*) from orders);
set lineitem_diff = lineitem_end - lineitem_start;
select 'In lineitem Rf1 Inserted ',lineitem_diff;
go
call tpch_rf2 (c_path,4)
go
set lineitem_start=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
go
call tpch_rf1 (c_path,5)
go
set lineitem_end=(select count(*) from lineitem);
set orders_end=(select count(*) from orders);
set lineitem_diff = lineitem_end - lineitem_start;
select 'In lineitem Rf1 Inserted ',lineitem_diff;
select 'In orders Rf1 Inserted ',orders_diff;
go
commit
go
tpch_wait
go

set lineitem_start=(select count(*) from lineitem);
go
call tpch_rf2 (c_path,5)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_start - lineitem_end;
select 'In lineitem Rf2 Deleted ',lineitem_diff;
go
tpch_wait
go
select 'Throughput Stream4 ends';
go

select 'Throughput Stream5 starts';

set lineitem_start=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
go
call tpch_rf1 (c_path,6)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set orders_end=(select count(*) from orders);
set lineitem_diff = orders_end - orders_start;
set orders_diff = orders_end - orders_start;
select 'In lineitem Rf1 Inserted ',lineitem_diff;
select 'In orders Rf1 Inserted ',orders_diff;
go
set lineitem_start=(select count(*) from lineitem);
go
call tpch_rf2 (c_path,6)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_start - lineitem_end;
select 'In lineitem Rf2 Deleted ',lineitem_diff;
go
tpch_wait
go
select 'Throughput Stream6 ends';
go

select 'Throughput Stream7 starts';

set lineitem_start=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
go
call tpch_rf1 (c_path,7)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_start - lineitem_end;
select 'In lineitem Rf1 Inserted ',lineitem_diff;
select 'In orders Rf1 Inserted ',orders_diff;
go
commit
go
tpch_wait
go

set lineitem_start=(select count(*) from lineitem);
go
call tpch_rf2 (c_path,7)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_start - lineitem_end;
select 'In lineitem Rf2 Deleted ',lineitem_diff;
go
tpch_wait
go
select 'Throughput Stream7 ends';
go

select 'Throughput Stream8 starts';

set lineitem_start=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
go
call tpch_rf1 (c_path,8)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set orders_end=(select count(*) from orders);
set lineitem_diff = lineitem_end - lineitem_start;
set orders_diff = orders_end - orders_start;
select 'In lineitem Rf1 Inserted ',lineitem_diff;
select 'In orders Rf1 Inserted ',orders_diff;
go
commit
go
tpch_wait
go

set lineitem_start=(select count(*) from lineitem);
go
call tpch_rf2 (c_path,8)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_start - lineitem_end;
select 'In lineitem Rf2 Deleted ',lineitem_diff;
go
tpch_wait
go
select 'Throughput Stream8 ends';
go

select 'Throughput Stream9 starts';

set lineitem_start=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
go
call tpch_rf1 (c_path,9)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_start - lineitem_end;
select 'In lineitem Rf1 Inserted ',lineitem_diff;
select 'In orders Rf1 Inserted ',orders_diff;
go

TPC Benchmark™ H FDR – IBM System Power 595 with Sybase IQ Single Application Server Edition v.15.1 ESD #1.2- Page 32
tpch_wait

go

set lineitem_start=(select count(*) from lineitem);
go
call tpch_rf2 (c_path,9)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_start - lineitem_end;
select 'In lineitem Rf2 Deleted',lineitem_diff;
go
tpch_wait
go
select 'Throughput Stream8 ends';
go

select 'Throughput Stream9 starts';
set lineitem_start=(select count(*) from lineitem);
set orders_start=(select count(*) from orders);
go
call tpch_rf1 (c_path,10)
go
set lineitem_end=(select count(*) from lineitem);
set orders_end=(select count(*) from orders);
set lineitem_diff = lineitem_end - lineitem_start;
set orders_diff = orders_end - orders_start;
select 'In lineitem Rf1 Inserted',lineitem_diff;
select 'In orders Rf1 Inserted',orders_diff;
go
commit
go
tpch_wait
go

set lineitem_start=(select count(*) from lineitem);
go
call tpch_rf2 (c_path,10)
go
commit
go
set lineitem_end=(select count(*) from lineitem);
set lineitem_diff = lineitem_start - lineitem_end;
select 'In lineitem Rf2 Deleted',lineitem_diff;
go
tpch_wait
go
select 'Throughput Stream9 ends';
go

set qstop = now(*)
select 'Refresh Stream START -- ,', dateformat (qstart, 'yyyy-mm-dd
hh:nn:ss.sss')
select 'Refresh Stream STOP -- ,', dateformat (qstop, 'yyyy-mm-dd
hh:nn:ss.sss')
SET n_seconds=cast(datediff(millisecond,qstart,qstop) AS numeric(16,5))/1000
select 'Refresh Stream Elapsed time -- ',+cast(n_seconds AS varchar(20))+ ' seconds'
go

B.15 update_refresh.sql
-- THIS IS A GENERATED FILE.

B.16 tpch_wait.sql
-- THIS IS A GENERATED FILE.
-- PLEASE CHANGE gen_tpch_wait.sh FOR MODIFICATIONS

if exists (select 1
    from SYS.SYSPROCEDURE
    where proc_name = 'tpch_wait') then
    DROP procedure tpch_wait;
end if
;
-- Script to put a delay between TPCH updates.
-- Normally we just want to sleep a bit to spread updates out
-- through the entire throughput test. Sometimes we run out of
-- space; if so, just wait some more...
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;

declaraintotal unsigned bigint;
decclaremainused unsigned bigint;
decclaretemptotal unsigned bigint;
decclaretempused unsigned bigint;
decclaremainfree unsigned bigint;
decclarecommand 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 > 372000000 )
            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 - C  Query Text and Output

C.1  Qualification Query Output

Query Text 1

Query Text
---------
9> select
10>  l_returnflag,
11>  l_linestatus,
12>  sum(l_quantity) as sum_qty,
13>  sum(l_extendedprice) as sum_base_price,
14>  sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
15>  sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
16>  avg(l_quantity) as avg_qty,
17>  avg(l_extendedprice) as avg_price,
18>  avg(l_discount) as avg_disc,
19>  count(*) as count_order
20> from
21>  lineitem
22> where
23>  l_shipdate <= dateadd(day, -90, '1998-12-01')
24> group by
25>  l_returnflag,
26>  l_linestatus
27> order by
28>  l_returnflag,
29>  l_linestatus

Query Result
------------

<table>
<thead>
<tr>
<th>l_returnflag</th>
<th>l_linestatus</th>
<th>sum_qty</th>
<th>sum_base_price</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>F</td>
<td>37734107.0000</td>
<td>56586554400.729141</td>
</tr>
<tr>
<td></td>
<td></td>
<td>53758257134.869675</td>
<td>55909065222.828072</td>
</tr>
<tr>
<td></td>
<td></td>
<td>25.522006</td>
<td>38273.129735</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.049985</td>
<td>38854</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>991417.0000</td>
<td>55686554400.729141</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1487504710.379999</td>
<td>1413082168.054105</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1469649223.194362</td>
<td>25.516472</td>
</tr>
<tr>
<td></td>
<td></td>
<td>38284.467761</td>
<td>0.050093</td>
</tr>
</tbody>
</table>

Query Text 2
---------
3> select top 100
4>  s_acctbal,
5>  s_name,
6>  n_name,
7>  p_partkey,
8>  p_mfgr,
9>  s_address,
10>  s_phone,
11>  s_comment
12> from
13>  part,
14>  supplier,
15>  partsupp,
16>  nation,
17>  region
18> where
19>  p_partkey = ps_partkey
20>  and s_suppkey = ps_suppkey
21>  and p_size = 15
22>  and p_type like '%BRASS'
23>  and s_nationkey = n_nationkey
24>  and n_regionkey = r_regionkey
25>  and r_name = 'EUROPE'
26>  and ps_supplycost = ( select
27>    min(ps_supplycost)
28>   from
29>    partsupp,
30>    supplier,
31>    nation,
32>    region
33>   where
34>    p_partkey = ps_partkey
35>    and s_suppkey = ps_suppkey
36>    and s_nationkey = n_nationkey
37>    and n_regionkey = r_regionkey
38>    and r_name = 'EUROPE'
39>  )
40> order by
41>  s_acctbal desc,
42>  n_name,
43>  s_name,
44>  p_partkey

Query Result
------------

<table>
<thead>
<tr>
<th>s_acctbal</th>
<th>s_name</th>
<th>n_name</th>
<th>p_partkey</th>
<th>p_mfgr</th>
<th>s_address</th>
<th>s_phone</th>
<th>s_comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>9871.220000 Supplier#000006373</td>
<td>GERMANY</td>
<td>43868 Manufacturer#5</td>
<td>J8fcXWstqM</td>
<td>320-228-2957 etect about the furiously final accounts. Slyly ico</td>
<td>9936.220000 Supplier#000005250</td>
<td>UNITED KINGDOM</td>
<td>33-429-790-6131</td>
</tr>
</tbody>
</table>

TPC Benchmark™ H FDR – IBM System Power 595 with Sybase IQ Single Application Server Edition v.15.1 ESD #1.2- Page 35
813-485-8637 etect blithely bold asymptotes. fluffily ironic platelets wake furiously; blit
9870.780000 Supplier#000001286    GERMANY
181285 Manufacturer#2
YKA,E2fjiVd7eUrzp2Ef8j1OxGo2DFnosaTEH 17-516-924-4574
regular accounts. furiously unusual courts above the f
9852.520000 Supplier#000008973    RUSSIA                         18972
Manufacturer#2            t5L67YdBYYH6o,Vz24jpDyQ9                 32-
188-594-7038 rns wake final foxes. carefully unusual depende
9847.570000 Supplier#000006345        FRANCE
VSt3rzk3qG698u6ld8HhOByvrTcSTSvQIDQdag   16-886-766-7945
ges. slyly regular requests are. ruthless, express excuses cajo
(10 rows affected)

**Query 3**

**Query Text**

`---`  
3> 4> select top 10  
5>    l_orderkey,  
6>    sum(l_extendedprice * (1 - l_discount)) as revenue,  
7>    dateformat(o_orderdate,'yyyy-mm-dd'),  
8>    o_shippriority  
9> from  
10>    customer,  
11>    orders,  
12>    lineitem  
13> where  
14>    c_mktsegment = 'BUILDING'  
15>    and c_custkey = o_custkey  
16>    and l_orderkey = o_orderkey  
17>    and o_orderdate < '1995-03-15'  
18>    and l_shipdate > '1995-03-15'  
19> group by  
20>    l_orderkey,  
21>    o_orderdate,  
22>    o_shippriority  
23> order by  
24>    revenue desc,  
25>    o_orderdate  
26>`  

**Query Result**

`---`  
<table>
<thead>
<tr>
<th>l_orderkey</th>
<th>revenue</th>
<th>dateformat(o_orderdate,'yyyy-mm-dd')</th>
<th>o_shippriority</th>
</tr>
</thead>
<tbody>
<tr>
<td>2456423</td>
<td>406181</td>
<td>011100 1995-03-05</td>
<td></td>
</tr>
<tr>
<td>3459808</td>
<td>405838</td>
<td>698900 1995-03-04</td>
<td></td>
</tr>
<tr>
<td>492164</td>
<td>390324</td>
<td>061000 1995-02-19</td>
<td></td>
</tr>
<tr>
<td>1188320</td>
<td>384537</td>
<td>939590 1995-03-09</td>
<td></td>
</tr>
</tbody>
</table>

(10 rows affected)

**Query 4**

**Query Text**

`---`  
2> 3> select  
4>    o_orderpriority,  
5>    count(*) as order_count  
6> from  
7>    orders  
8> where  
9>    o_orderdate >= '1993-07-01'  
10>    and o_orderdate < dateadd(month, 3, '1993-07-01')  
11>    and exists (  
12>       select *  
13>       from  
14>       lineitem  
15>       where  
16>         l_orderkey = o_orderkey  
17>       and l_commitdate < l_receiptdate  
18>   )  
19> group by  
20>    o_orderpriority  
21> order by  
22>    o_orderpriority  

**Query Result**

`---`  
<table>
<thead>
<tr>
<th>o_orderpriority</th>
<th>order_count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-URGENT</td>
<td>10594</td>
</tr>
<tr>
<td>2-HIGH</td>
<td>10476</td>
</tr>
<tr>
<td>3-MEDIUM</td>
<td>10410</td>
</tr>
<tr>
<td>4-NOT SPECIFIED</td>
<td>10556</td>
</tr>
<tr>
<td>5-LOW</td>
<td>10487</td>
</tr>
</tbody>
</table>

(5 rows affected)

**Query 5**

**Query Text**

`---`  
2> 3> select  
4>    n_name,  
5>    sum(l_extendedprice * (1 - l_discount)) as revenue  
6> from  
7>    customer,  
8>    orders,  
9>    lineitem,  
10>    supplier,  
11>    nation,  
12>    region  
13> where  

2435712 | 378673.055800 | 1995-02-26  
4878020 | 378376.795200 | 1995-03-12  
5521732 | 375153.921500 | 1995-03-13  
2628192 | 373133.309400 | 1995-02-22  
993600  | 371407.459500 | 1995-03-05  
2300070 | 367371.145200 | 1995-03-13  

(10 rows affected)
14>  c_custkey = o_custkey
15>  and l_orderkey = o_orderkey
16>  and l_suppkey = s_suppkey
17>  and c_nationkey = s_nationkey
18>  and s_nationkey = n_nationkey
19>  and n_regionkey = r_regionkey
20>  and r_name = 'ASIA'
21>  and o_orderdate >= '1994-01-01'
22>  and o_orderdate < dateadd(year, 1, '1994-01-01')
23> group by
24>  n_name
25> order by
26>  revenue desc
27>
28> Query Result
29> ------------
30> n_name                    revenue
31> ------------------------- ---------------------------
32> INDONESIA                             55502041.169700
33> VIETNAM                               55295086.996700
34> CHINA                                 53724494.256600
35> INDIA                                 52035512.0002 00
36> JAPAN                                 45410175.6954 00
37> (5 rows affected)

Query 6
Query Text
--------

2>
3> select
4>  sum(l_extendedprice * l_discount) as revenue
5> from
6>  lineitem
7> where
8>  l_shipdate >= '1994-01-01'
9>  and l_shipdate < dateadd(year, 1, '1994-01-01')
10>  and l_discount between .06 - 0.01 and .06 + 0.01
11>  and l_quantity < 24
12>
13> Query Result
14> ------------
15> revenue
16> ---------------------------
17> 123141078.228300
18> (1 row affected)

Query 7
Query Text
--------

3>
4> select
5>  supp_nation,
6>  cust_nation,
7>  l_year,
8>  sum(volume) as revenue
9> from
10> (
11>   select
12>    n1.n_name as supp_nation,
13>    n2.n_name as cust_nation,
14>    year(l_shipdate) as l_year,
15>    l_extendedprice * (1 - l_discount) as volume
16> from
17>   supplier,
18>   lineitem,
19>   customer,
20>   nation n1,
21>   nation n2
22>   where
23>    s_suppkey = l_suppkey
24>    and o_orderkey = l_orderkey
25>    and c_custkey = o_custkey
26>    and s_nationkey = n1.n_nationkey
27>    and c_nationkey = n2.n_nationkey
28>    and (n1.n_name = 'FRANCE' and
29>      n2.n_name = 'GERMANY')
30> or (n1.n_name = 'GERMANY'
31>   and n2.n_name = 'FRANCE')
32> ) as shipping
33> group by
34>  supp_nation,
35>  cust_nation,
36>  l_year
37> order by
38>  supp_nation,
39>  cust_nation,
40>  l_year
41>
42> Query Result
43> ------------
44> supp_nation               cust_nation               l_year      revenue
45> ------------------------- ------------------------- ----------- ---------------------------
46> FRANCE                    GERMANY                          1995         54639732.733600
47> FRANCE                    GERMANY                          1996         54633083.307600
48> GERMANY                   FRANCE                           1995         52531746.669700
49> GERMANY                   FRANCE                           1996         52520549.022400
50> (4 rows affected)

Query 8
Query Text
--------

5>
6> select
7>  o_year,
8>  sum(case
9>   when nation = 'BRAZIL' then volume
10>   else 0
11> end) / sum(volume) as mkt_share
12> from
13> (
14>   select
15>    year(o_orderdate) as o_year,
16>    l_extendedprice * (1 - l_discount) as volume,
17>    n2.n_name as nation
18> from
19>   part,
20>   supplier,
21>   lineitem,
22>   orders,
23>   customer,
30>  and o_custkey = c_custkey
31>  and c_nationkey = n1.n_nationkey
32>  and n1.n_regionkey = r.regionkey
33>  and r.name = 'AMERICA'
34>  and s_nationkey = n2.n_nationkey
35>  and o_orderdate between '1995-01-01'
36>  and '1996-12-31'
37>  ) as all_nations
38> group by
39>  o_year
40> order by
41>  o_year
42>
43> Query Result
44> ------------
45> o_year    mkt_share
46> ---------- ---------------
47> 1995      0.034436
48> 1996      0.041486
49>
50> (2 rows affected)

Query 9

Query Text
--------

3>
4> select
5>  nation,
6>  o_year,
7>  sum(amount) as sum_profit
8> from
9>  (select
10>    n_name as nation,
11>    year(o_orderdate) as o_year,
12>    l_extendedprice * (1 - l_discount) -
13>    ps_supplycost * l_quantity as amount
14>  from
15>    part,
16>    supplier,
17>    lineitem,
18>    partsupp,
19>    orders,
20>    nation
21>  where
22>    s_suppkey = l_suppkey
23>    and ps_suppkey = l_suppkey
24>    and p_partkey = l_partkey
25>    and p_partkey = l_partkey
26>    and o_orderkey = l_orderkey
27>    and s_nationkey = n.nationkey
28>    and p_name like '%green%'
29>  ) as profit
30> group by
31>  nation,
32>  o_year
33> order by
34>  revenue desc
35> Query Result
36> ------------
37> c_custkey c_name                    revenue                     c_acctbal
38> n_name                    c_address                                c_phone
39> c_comment
40> ------------- ---------------------------------------- --------------------------- -----
41> -----------------------------------------------
42> -------

(175 rows affected)

Query 10

Query Text
--------

3>
4> select top 20
5>  c_custkey,
6>  c_name,
7>  sum(l_extendedprice * (1 - l_discount)) as revenue,
8>  c_acctbal,
9>  n_name,
10>  c_address,
11>  c_phone,
12>  c_comment
13> from
14>  customer,
15>  orders,
16>  lineitem,
17>  nation
18> where
19>  c_custkey = o_custkey
20>  and l_orderkey = o_orderkey
21>  and o_orderdate >= '1993-10-01'
22>  and o_orderdate < dateadd(month, 3, '1993-10-01')
23>  and l_returnflag = 'R'
24>  and c_nationkey = n.nationkey
25> group by
26>  c_custkey,
27>  c_name,
28>  c_acctbal,
29>  n_name,
30>  c_address,
31>  c_phone,
32>  c_comment
33> order by
34>  revenue desc
35> Query Result
36> ------------
37> c_custkey c_name                    revenue                     c_acctbal
38> n_name                    c_address                                c_phone
39> c_comment
40> ------------- ---------------------------------------- --------------------------- -----
41> -----------------------------------------------
(1048 rows affected)

**Query 11**

**Query Text**

```
CREATE TABLE #COUNTRY AS
SELECT n_nationkey, n_name FROM nation WHERE n_name = 'GERMANY';
```

**Query Result**

```
ps_partkey   value
---------- -----------------------------
129760       17538456.860000
166726       16503353.920000
191287       16474801.970000
161758       16101755.540000
34452        15983844.720000
139035       15907078.340000
9403         15451755.620000
154358       15212937.880000
38823        15064802.860000
85606        15053957.150000
```

(104 rows affected)

**Query 12**

**Query Text**

```
CREATE TABLE #COUNTRY AS
SELECT n_nationkey, n_name FROM nation WHERE n_name = 'GERMANY';
```

**Query Result**

```
l_shipmode   high_line_count      low_line_count
---------- -------------------- ---------------------
MAIL        1                   0
SHIP        0                   1
```

(104 rows affected)
MAIL 6202 9324
SHIP 6200 9262

(2 rows affected)

**Query 13**

Query Text
--------
3>
4> select
5>  c_count,
6>  count(*) as custdist
7> from
8> (  
9>   select
10>    c_custkey,
11>    count(o_orderkey)
12>   from
13>     customer left outer join orders on
14>     c_custkey = o_custkey
15>     and o_comment not like '%special%requests%'
16>   group by
17>     c_custkey
18> ) as c_orders (c_custkey, c_count)
19> group by
20>  c_count
21> order by
22>  custdist desc,
23>  c_count desc
24>

Query Result
-----------
c_count              custdist
-------------------- --------------------
0                50005
9                 6641
10                 6532
11                 6014
8                 5937
12                 5639
13                 4793
7                 4687
17                 4587

(42 rows affected)

**Query 14**

Query Text
--------
3>
4> select
5>  100.00 * sum(case
6>   when p_type like 'PROMO\%' 
7>     then l_extendedprice * (1 - l_discount)
8>   else 0
9> end) / sum(l_extendedprice * (1 - l_discount)) as
10> promo_revenue
11> from
12> lineitem,
13> part
14> where
15> l_partkey = p_partkey
16> and l_shipdate < dateadd(month, 1, '1995-09-01')
17>

Query Result
----------
promo_revenue
----------------
16.380779

(1 row affected)

**Query 15**

Query Text
--------
3>
4> create view revenue0 (supplier_no, total_revenue) as
5> select
6>  l_suppkey,
7>  sum(l_extendedprice * (1 - l_discount))
8> from
9>  lineitem
10> where
11>  l_shipdate >= '1996-01-01'
12>  and l_shipdate < dateadd(month,3,'1996-01-01')
13> group by
14>  l_suppkey
15>

Query Result
----------
s_suppkey  s_name                    s_address
s_phone         total_revenue
---------- ------------------------- ---------------------------------------- --------------- --
8449 Supplier#000008449        Wp34zim9qYFbVctdW
20-469-856-8873              1772627.208700

(1 row affected)

2> drop view revenue0

**Query 16**

Query Text
--------
3>
4> select
5>  s_suppkey, s_name, s_address
6> from
7> supplier
8> where
9>  s_suppkey = supplier_no
10>  and total_revenue = ( 
11>   select
12>    max(total_revenue)
13>   from
14>    revenue0
15> )
16> order by
17>  s_suppkey
18>

Query Result
----------
8449 Supplier#000008449        Wp34zim9qYFbVctdW
20-469-856-8873              1772627.208700

(1 row affected)

2> drop view revenue0

TPC Benchmark™ H FDR – IBM System Power 595 with Sybase IQ Single Application Server Edition v.15.1 ESD #1.2- Page 40
```sql
6>  p_brand,
7>  p_type,
8>  p_size,
9>  count(distinct ps_suppkey) as supplier_cnt
10> from
11>  partsupp,
12>  part
13> where
14>  p_partkey = ps_partkey
15>  and p_brand <> 'Brand#45'
16>  and p_type not like 'MEDIUM POLISHED%'
17>  and p_size in (49, 14, 23, 45, 19, 3, 36, 9)
18>  and ps_suppkey not in ( select
19>     s_suppkey
20>   from
21>     supplier
22>   where
23>     s_comment like '%Customer%Complaints%'
24>   )
25> group by
26>  p_brand,
27>  p_type,
28>  p_size
29> order by
30>  supplier_cnt desc,
31>  p_brand,
32>  p_type,
33>  p_size
34> Query Result
35> ------------
p_brand  p_type                    p_size      supplier_cnt
---------- ------------------------- ----------- -------------------
Brand#41  MEDIUM BRUSHED TIN                  3                   28  
Brand#54  STANDARD BRUSHED COPPER            14                   27
Brand#11  STANDARD BURNISHED BRASS           23                   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
(18314 rows affected)

Query 17
--------

Query Text
----------

3>
4>
5> select top 100
6>  c_name,
7>  c_custkey,
8>  o_orderkey,
9>  dateformat(o_orderdate,'yyyy-mm-dd'),
10> o_totalprice,
11> sum(l_quantity)
12> from
13>  customer,
14>  orders,
15>  lineitem
16> where
17>  o_orderkey in ( select
18>    l_orderkey
19>  from
20>    lineitem
21>  group by
22>    l_orderkey having
23>     sum(l_quantity) > 300
24>   )
25>  and c_custkey = o_custkey
26>  and o_orderkey = l_orderkey
27> group by
28>  c_name,
29>  c_custkey,
30>  o_orderkey,
31>  o_orderdate,
32>  o_totalprice
33> order by
34>  o_totalprice desc,
35>  o_orderdate
36> Query Result
37> ------------
c_name                  c_custkey  o_orderkey               datefor-
mat(o_orderdate,'yyyy-mm-dd') o_totalprice
sum(lineitem.l_quantity)
------------------------- ---------- --------------------------- -------------------------------------
Customer#000128120      128120    472220 1994-04-07 544089.090000 323.000000
Customer#000144617      144617    3043270 1997-02-12 530604.440000 317.000000
Customer#000013940      13940     2199712 1996-09-30 522720.610000 327.000000
Customer#000066790      66790     2199712 1996-09-30 515531.820000 327.000000
(18314 rows affected)
```
Query 19

Query Text
----------

3>
4>
5> select
6>  sum(l_extendedprice*(1 - l_discount)) as revenue
7> from
8>  lineitem,
9>  part
10> where
11>  ( p_partkey = l_partkey
12>    and p_brand = 'Brand#12'
13>    and p_container in ('SM CASE', 'SM BOX', 'SM PACK', 'SM PKG')
14>    and l_quantity >= 1 and l_quantity <= 1 + 10
15>    and l_shipmode in ('AIR', 'AIR REG')
16>    and l_shipinstruct = 'DELIVER IN PERSON'
17>  )
18>  or
19>  ( p_partkey = l_partkey
20>    and p_brand = 'Brand#23'
21>    and p_container in ('MED BAG', 'MED BOX', 'MED PKG', 'MED PACK')
22>    and l_quantity >= 10 and l_quantity <= 10 + 10
23>    and l_shipmode in ('AIR', 'AIR REG')
24>    and l_shipinstruct = 'DELIVER IN PERSON'
25>  )
26>  or
27>  ( p_partkey = l_partkey
28>    and p_brand = 'Brand#34'
29>    and p_container in ('LG CASE', 'LG BOX', 'LG PKG')
30>    and l_quantity >= 20 and l_quantity <= 20 + 10
31>    and l_shipmode in ('AIR', 'AIR REG')
32>    and l_shipinstruct = 'DELIVER IN PERSON'
33>  )
34>  or
35>  ( p_partkey = l_partkey
36>    and p_brand = 'Brand#12'
37>    and p_container in ('SM CASE', 'SM BOX', 'SM PACK', 'SM PKG')
38>    and l_quantity >= 1 and l_quantity <= 1 + 10
39>    and l_shipmode in ('AIR', 'AIR REG')
40>    and l_shipinstruct = 'DELIVER IN PERSON'
41> )

Query Result
------------

revenue
---------------------------
3083843.057800

(57 rows affected)

Query 20

Query Text
----------

3>
4> select
5>  s_name,
6>  s_address
7> from
8>  supplier,
9>  nation
10> where
11>  s_suppkey in ( select
12>    ps_suppkey
13>  from
14>    partsupp
15>  where
16>    ps_partkey in ( select
17>      p_partkey
18>    from
19>      part
20>    where
21>      p_brand like 'for-
22>        est%'
23>    )
24>    and ps_availqty > ( select
25>      0.5 * sum(l_quantity)
26>    from
27>      lineitem
28>    where
29>      l_partkey = ps_partkey
30>      and l_suppkey = ps_suppkey
31>      and l_shipdate >= dateadd(year,1,'1994-01-01')
32>      and l_shipdate < dateadd(year,1,'1994-01-01')
33>    )
34>  )
35>  and s_nationkey = n_nationkey
36>  and n_name = 'CANADA'
37> order by
38>  s_name
39>  s_address
40>  s_name
41>

Query Result
------------

s_name                         s_address
-----------------------------------------------------------------------
Supplier#00000020    iybAE,RmTymrZVYaFZva2SH.j
Supplier#00000091    YV45D7TkIdQanO0Zq9QxxyGUapU1oOWU6q3
Supplier#00000197    YC2Acon6kjY3z3Fbx3k4Vdf7X0cd2F
Supplier#00000226    83qOnu2EYEdqAO0hEtEn GRZEd
Supplier#00000285    Br7e1ntt1xyw6lmgu7YdhFDjuBf
Supplier#00000378    FtbyhxOxWvCPr08itp9
Supplier#00000402    i9Sw4DoyMhzhKXCH9by,AYsgmD
Supplier#00000539    0qwCMwoBKJ YcmLYrxlaagA8ukENJv,
Supplier#00000688    D fw5ocppmZyYBBIPF178hCihLDZ5KlKX
Supplier#00000710    119YPvOyb QoWwjkC,oPycpGieBAcwKjo

(204 rows affected)
Query 21

Query Text
----------

4>
5> select
6>     top 100 s_name,
7>     count(*) as numwait
8> from
9>     supplier,
10>    lineitem l1,
11>   orders,
12>  nation
13> where
14>     s_suppkey = l1.l_suppkey
15>     and o_orderkey = l1.l_orderkey
16>     and o_orderstatus = 'F'
17>     and l1.l_receiptdate > l1.l_commitdate
18>     and exists (  
19>         select
20>             *
21>         from
22>             lineitem l2
23>     where
24>         l2.l_orderkey = l1.l_orderkey
25>         and l2.l_suppkey <> l1.l_suppkey
26>     )
27>     and not exists (  
28>         select
29>             *
30>         from
31>             lineitem l3
32>     where
33>         l3.l_orderkey = l1.l_orderkey
34>         and l3.l_suppkey <> l1.l_suppkey
35>         and l3.l_receiptdate > l3.l_commitdate
36>     )
37>     and s_nationkey = n_nationkey
38>     and n_name = 'SAUDI ARABIA'
39> group by
40>    s_name
41> order by
42>     numwait desc,
43>    s_name
44>

Query Result
----------
s_name numwait
--------------------
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

(100 rows affected)

Query 22

Query Text
----------

--------
3>
4> select
cntrycode,
count(*) as numcust,
sum(c_acctbal) as totacctbal
from
custsale
where
substr(c_phone,1,2) in
('13', '31', '23', '29', '30', '18', '17')
and c_acctbal >  
select avg(c_acctbal)
from
customer
where
c_acctbal > 0.00
and substr(c_phone,1,2) in
('13', '31', '23', '29', '30', '18', '17')
and not exists (  
select *  
from
orders
where
o_custkey = c_custkey
)  
and c_acctkey = c_custkey
)  
group by
cntrycode
order by
cntrycode

Query Result
----------

cntrycode numcust totacctbal
------------------------------
13 888 6737713.990000
17 861 6460573.720000
18 964 7236687.400000
23 892 6701457.950000
29 948 715866.630000
30 909 6808436.130000
31 922 6806670.180000

(7 rows affected)

C.2  Query Substitution Parameters

Opts.0

<table>
<thead>
<tr>
<th>Opts.</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>1994-03-01</td>
</tr>
<tr>
<td>2</td>
<td>37 COPPER ASIA</td>
</tr>
<tr>
<td>9</td>
<td>chocolate</td>
</tr>
<tr>
<td>20</td>
<td>honeydew 1996-01-01 UNITED STATES</td>
</tr>
<tr>
<td>6</td>
<td>1994-01-01 0.02 24</td>
</tr>
<tr>
<td>17</td>
<td>Brand#15 WRAP CASE</td>
</tr>
<tr>
<td>18</td>
<td>315</td>
</tr>
<tr>
<td>8</td>
<td>BRAZIL AMERICA SMALL ANODIZED TIN</td>
</tr>
<tr>
<td>Options</td>
<td>Codes</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td><strong>Opts.1</strong></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>GERMANY</td>
</tr>
<tr>
<td>3</td>
<td>FURNITURE</td>
</tr>
<tr>
<td>18</td>
<td>313</td>
</tr>
<tr>
<td>5</td>
<td>EUROPE 1994-01-01</td>
</tr>
<tr>
<td>11</td>
<td>JAPAN 0.0000000333</td>
</tr>
<tr>
<td>7</td>
<td>PERU ROMANIA</td>
</tr>
<tr>
<td>6</td>
<td>1994-01-01</td>
</tr>
<tr>
<td>6</td>
<td>salmon 1994-01-01</td>
</tr>
<tr>
<td>17</td>
<td>Brand#12 WRAP JAR</td>
</tr>
<tr>
<td>12</td>
<td>TRUCK MAIL 1994-01-01</td>
</tr>
<tr>
<td>16</td>
<td>Brand#11 SMALL PLATED</td>
</tr>
<tr>
<td>29</td>
<td>18</td>
</tr>
<tr>
<td>15</td>
<td>1996-06-01</td>
</tr>
<tr>
<td>13</td>
<td>unusual requests</td>
</tr>
<tr>
<td>10</td>
<td>25 STEEL AFRICA</td>
</tr>
<tr>
<td>8</td>
<td>ROMANIA EUROPE STANDARD POLISHED TIN</td>
</tr>
<tr>
<td>14</td>
<td>1994-06-01</td>
</tr>
<tr>
<td>19</td>
<td>Brand#55 Brand#23 Brand#24</td>
</tr>
<tr>
<td>9</td>
<td>blush</td>
</tr>
<tr>
<td>22</td>
<td>20</td>
</tr>
<tr>
<td>31</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1993-05-01</td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Options</th>
<th>Codes</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Opts.2</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1994-01-01</td>
<td>0.05 24</td>
</tr>
<tr>
<td>17</td>
<td>Brand#14</td>
<td>WRAP PKG</td>
</tr>
<tr>
<td>14</td>
<td>1994-09-01</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Brand#1 LARGE BRUSHED</td>
<td>29 12</td>
</tr>
<tr>
<td>49</td>
<td>17</td>
<td>1 24 45 21</td>
</tr>
<tr>
<td>19</td>
<td>Brand#52 Brand#51 Brand#23</td>
<td>7 13 26</td>
</tr>
<tr>
<td>10</td>
<td>1993-07-01</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>azure</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>12 BRASS ASIA</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1994-03-01</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>IRAQ MIDDLE EAST STANDARD BURNISHED TIN</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>MIDDLE EAST 1994-01-01</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>31 15 19 14 32 23 20</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>RAIL MAIL 1994-01-01</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>INDONESIA IRAQ</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>unusual requests</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>314</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>105</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1995-12-01</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>cyan 1997-01-01</td>
<td>CANADA</td>
</tr>
<tr>
<td>3</td>
<td>MACHINERY 1995-03-19</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ALGERIA 0.0000000333</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>UNITED STATES</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Options</th>
<th>Codes</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Opts.3</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>CANADA AMERICA</td>
<td>PROMO BRUSHED NICKEL</td>
</tr>
<tr>
<td>5</td>
<td>AFRICA 1994-01-01</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1993-09-01</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1994-01-01</td>
<td>0.02 24</td>
</tr>
<tr>
<td>17</td>
<td>Brand#11 SM CASE</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>ARGENTINA CANADA</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1994-01-01</td>
<td>22 31</td>
</tr>
<tr>
<td>14</td>
<td>1994-12-01</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>wheat</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1994-04-01</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1996-10-01</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>JORDAN 0.0000000333</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>orchid 1996-01-01</td>
<td>CHINA</td>
</tr>
<tr>
<td>2</td>
<td>50 NICKEL AFRICA</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>MOZAMBIQUE</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Brand#14 Brand#44 Brand#23</td>
<td>14 22</td>
</tr>
<tr>
<td>13</td>
<td>unusual requests</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Brand#21 STANDARD ANODIZED</td>
<td>1 14</td>
</tr>
<tr>
<td>34</td>
<td>38 22 37 12 20</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>AIR MAIL 1994-01-01</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>BUILDING 1995-03-05</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Options</th>
<th>Codes</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Opts.4</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ASIA 1995-01-01</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>INDIA</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>1995-03-01</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Brand#11 Brand#22 Brand#11</td>
<td>8 15 29</td>
</tr>
<tr>
<td>15</td>
<td>1994-06-01</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Brand#13 SM JAR</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>REG AIR MAIL 1995-01-01</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1995-01-01</td>
<td>0.07 25</td>
</tr>
<tr>
<td>4</td>
<td>1996-04-01</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>steel</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>SAUDI ARABIA MIDDLE EAST PROMO PLATED NICKEL</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Brand#11 MEDIUM PLATED</td>
<td>27 28 34</td>
</tr>
<tr>
<td>49</td>
<td>8</td>
<td>5 38 25</td>
</tr>
<tr>
<td>11</td>
<td>ARGENTINA 0.0000000333</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>38 TIN EUROPE</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1995-01-01</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>313</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>unusual requests</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>CHINA SAUDI ARABIA</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>17 13 11 28 22 24 16</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MACHINERY 1995-03-21</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>bisque 1994-01-01</td>
<td>INDIA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Options</th>
<th>Codes</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Opts.5</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>ALGERIA</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1997-01-01</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1994-01-01</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1995-01-01</td>
<td>0.05 24</td>
</tr>
<tr>
<td>7</td>
<td>IRAN JAPAN</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Brand#41 ECONOMY POLISHED</td>
<td>6 3</td>
</tr>
<tr>
<td>17</td>
<td>20 2 12 8 14 9</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Brand#13 Brand#15 Brand#11</td>
<td>16 25 315</td>
</tr>
<tr>
<td>14</td>
<td>1995-07-01</td>
<td></td>
</tr>
</tbody>
</table>
Appendix - D  
Driver Source Code

D.1  run_tpch

#!/bin/ksh

if (( $# < 1 ))
then
    echo "Usage: run_tpch scope scale_factor (plus additional args depending upon scope)"
    echo "scope values:"
    echo "  - load   load the tpch test database"
    echo "  - power  do a single power run "
    echo "     (with or without refresh streams)"
    echo "  - throughput run":
    echo "     (with or without refresh streams)"
    echo "     - refresh run refresh pair(s)"
    echo "     - perf   run a single power-throughput cycle (without a load)"
    echo "     - all do a full-audit run"
    echo "     (load + 2 power-throughput cycles)"
    echo "     -audit_only Runs only audit_scripts."
    echo "To get usage help on individual scope options, type" run_tpch scope help"
else
    if [ $SCOPE = audit_only ]
then
    if [ $# -lt 2 ] || [ $2 = help ]
then
    echo "Usage: run_tpch $SCOPE scale_factor"
    echo "  -- Default value for enable_monitoring:"
    echo "  stream_count: will be set to minimum required"
    echo "  $stream_count number of query streams"
    echo "  This generates"
    echo "Scale Factor $SCALE_FACTOR"
    fi
    if [ -z "$3" ]
then
    Enable_Monitoring=0; export Enable_Monitoring
    else
    Enable_Monitoring=$3; export Enable_Monitoring
    fi
    fi
else
    SCALE_FACTOR=$2
    . /bm_setup $SCALE_FACTOR
    echo "run_tpch SCOPE=$SCOPE"
fi
fi
unset STREAM_COUNT
nParams=$#

if [ ! -z "$2" ] && [ $2 != help ]
then
    while [ $i -le $nParams ]
    do
        eval i=$i
        junk=`expr $i + 1` 2>/dev/null
        if [ $? != 0 ] || [ -z "$junk" ]
        then
            echo "input parameter $i supplied is not correct"
            echo " Please run /run_tpch help to get usage information."
            exit
        fi
        i=`expr $i + 1`
        done
    fi
SCOPE=$1
elif \[ $SCOPE = power \] 
then
  if \[ \$# = 2 \] || \[ \$2 = help \]
  then
    echo "Usage: run_tpch $SCOPE scale_factor
    [with_rf] [enable_monitoring]"
    echo "  -- Default value for *
    a) with_rf option: 1
    b) enable_monitoring: 0 (no monitoring)"
    echo "  
    (run with refresh streams)"
    echo "  
    SCALE_FACTOR=$2
    "
    fi
  else
    SCALE_FACTOR=$2
    fi
  fi
elif \[ $SCOPE = throughput \] || \[ $SCOPE = all \]
then
  if \[ \$# = 2 \] || \[ \$2 = help \]
  then
    echo "Usage: run_tpch $SCOPE scale_factor
    [with_rf] [enable_monitoring]"
    echo "  -- Default value for *
    a) with_rf option: 1
    b) enable_monitoring: 0 (no monitoring)"
    echo "  
    (run with refresh streams)"
    echo "  
    SCALE_FACTOR=$SCALE_FACTOR
    with_rf=$WITH_RF
    enable_monitoring=$ENABLE_MONITORING
    "
    fi
  else
    fi
  fi
elif \[ $SCOPE = perf \]
then
  if \[ \$# = 3 \] || \[ \$2 = help \]
  then
    echo "Usage: run_tpch $SCOPE scale_factor
    run_number [with_rf] [enable_monitoring]"
    echo "  -- Default value for *
    a) with_rf option: 1
    b) enable_monitoring: 0 (no monitoring)"
    echo "  
    (run with refresh streams)"
    echo "  
    SCALE_FACTOR=$SCALE_FACTOR
    with_rf=$WITH_RF
    enable_monitoring=$ENABLE_MONITORING
    RUN_NUMBER=$RUN_NUMBER
    "
    fi
  else
    fi
  fi
else
  echo "run_tpch SCOPE=$SCOPE
  SCALE_FACTOR=$SCALE_FACTOR
  WITH_RF=$WITH_RF
  ENABLE_MONITORING=$ENABLE_MONITORING
  STREAM_COUNT=$STREAM_COUNT"
  echo "  
  "
  fi
then Enable_Monitoring=0; export Enable_Monitoring
else Enable_Monitoring=$5; export Enable_Monitoring
fi
if [ ! -z "$6" ]
then
STREAM_COUNT=$6; export STREAM_COUNT
fi
./bm_setup $SCALE_FACTOR
echo "Please supply valid name for scope"
exit 1
fi
if [ ! "$RUN_NUMBER" ]
then
RUN_NUMBER=0
fi

# Set Results directory with seq_number
SEQ_NUMBER=`cat $(DEV_DIR)/data/seq_number`
RESULTS=$(RESULTS)/run_${SEQ_NUMBER}; export RESULTS
mkdir -p $RESULTS > /dev/null 2>&1
SEQ_NUMBER=$((SEQ_NUMBER + 1))

os=`uname`
if [ $os = AIX ]
then
rm -f $RESULTS/sysunused
echo "The following disks are assigned to the indicated volume groups">
lspv >> $RESULTS/sysunused
else
$STOP_SERVER -stop all
rm $TPCH_ROOT/plans/* > /dev/null 2>&1
os=`uname`
if [ $os = AIX ]
then
rm -f $RESULTS/sysunused
else
endif
fi

sleep 10
echo "The following volume groups are currently online">
echo `date` >> $RESULTS/sysunused
lsvg -o >> $RESULTS/sysunused
fi

$STOP_SERVER -stop all
rm $TPCH_ROOT/plans/* > /dev/null 2>&1
os=`uname`
if [ $os = AIX ]
then
rm -f $RESULTS/sysunused
else
endif
fi

if [ $2 = help ]
then
echo "Usage: run_tpch $SCOPE $SCALE_FACTOR [stream_count] [enable_monitoring]"
echo "  -- Default value for "
echo "   a) stream_count: 1  (runs one refresh pair)"
echo "   b) enable_monitoring: 0 (no monitoring)"
echo "exit"
else
SCALE_FACTOR=$2
if [ ! "$3" ];
then
STREAM_COUNT=1; export STREAM_COUNT
else
STREAM_COUNT=3; export STREAM_COUNT
fi
if [ ! "$4" ];
then
Enable_Monitoring=0; export Enable_Monitoring
else
Enable_Monitoring=$4; export Enable_Monitoring
fi
./bm_setup $SCALE_FACTOR
echo "run_tpch SCOPE=$SCOPE SCALE_FACTOR=$SCALE_FACTOR STREAM_COUNT=$STREAM_COUNT ENABLE_MONITORING=$Enable_Monitoring"
fi

if [ $SCOPE = 'refresh' ]
then
run_refresh.sh $STREAM_COUNT $Enable_Monitoring
fi

# After the load completes run the Audit SQL if SCOPE=all
if [ $SCOPE = 'all' -o $SCOPE = 'load' ]
then
# Load the tpch test database
load_test $Enable_Monitoring
fi

if [ $SCOPE = 'all' ]
then
start_log_details=`ls -l $DEV_DIR/tpch.log`
echo "$start_log_details" > $RESULTS/mlog
fi

# Generate streams & set the Debug mode.
cwd=`pwd`
cd $QUERYGEN
SEED=`cat $DEV_DIR/data/seed`
./querygen.sh $SEED $SCALE_FACTOR $STREAM_COUNT
echo "cd $cwd"
cd $cwd
if [ $SCOPE = 'refresh' ]
then
run_refresh.sh $STREAM_COUNT $Enable_Monitoring
fi

echo echo # After the load completes run the Audit SQL if SCOPE=all
if [ $SCOPE = 'all' -o $SCOPE = 'audit_only' -o $SCOPE = 'perf' -o $RUN_NUMBER = '1' ]
then
  echo "Running the Audit Script `date`"
  dbisqlc -c "DSN=tpch_$VERSION" -q $SQL/dbtables-
syb.sql > $RESULTS/dbatables
  dbisqlc -c "DSN=tpch_$VERSION" -q $SQL/dew_cat1.sql >
  $RESULTS/dew_cat1_start.out
  dbisqlc -c "DSN=tpch_$VERSION" -q $SQL/dew_cat2.sql >
  $RESULTS/dew_cat2_start.out
  dbisqlc -c "DSN=tpch_$VERSION" -q $SQL/dew_cat3.sql >
  $RESULTS/dew_cat3_start.out
fi

if [ $MODE = 'all' -o $MODE = 'power' -o $MODE = 'perf' -o
  $MODE = 'throughput' ]
then
  SEED=`cat $DEV_DIR/data/seed`;export SEED
fi

if [ $MODE = 'all' -o $MODE = 'perf' ]
then
  ######################## FIRST RUN ########################
  #Run power test.
  powertest.sh Run $SCALE_FACTOR $WITH_RF $Enable_Monitoring
  #Run throughput test.
  throughputtest.sh Run $SCALE_FACTOR $WITH_RF $Enable_Monitoring
  $STREAM_COUNT
  echo " "
  tpch_report.sh $MODE $SCALE_FACTOR
  $WITH_RF 0 0 1 $SEED $STREAM_COUNT 0 0 0 > $RESULTS/mrun1_report.out
fi

if [ $MODE = 'all' -o $MODE = 'perf' ]
then
  ######################## SECOND RUN
  ########################
  #Run power test.
  powertest.sh Run $SCALE_FACTOR $WITH_RF $Enable_Monitoring
  #Run throughput test.
  throughputtest.sh Run $SCALE_FACTOR $WITH_RF $Enable_Monitoring
  $STREAM_COUNT
  echo " "
  tpch_report.sh $MODE $SCALE_FACTOR
  $WITH_RF 0 0 2 $SEED $STREAM_COUNT 0 0 0 > $RESULTS/mrun2_report.out
  if $MODE = 'all' -o $MODE = 'load'
  then
    cat $RESULTS/deventlog | tr -s ' ' | tr -s '"' | tr -s "='>
    $RESULTS/deventlog."date +%y%m%d_%H%M%S"
    rm $RESULTS/deventlog
    if [ $MODE = 'all' ]
    then
      move_out_files 2 $WITH_RF > /dev/null 2>&1
    fi
  fi
fi
echo "================================" >>
$RESULTS/syshw
cat $RESULTS/sysconfig/Volume_Detail* >>
$RESULTS/syshw
echo "Volume_Group_List info" >>
$RESULTS/syshw
echo "================================" >>
$RESULTS/syshw
cat $RESULTS/sysconfig/Volume_Group_List* >>
$RESULTS/syshw
echo "Volume_Group_Member_list info" >>
$RESULTS/syshw
echo "================================" >>
$RESULTS/syshw
cat $RESULTS/sysconfig/Volume_Group_Member_list* >>
$RESULTS/syshw
echo "Volume_Group_Pdisk_list info" >>
$RESULTS/syshw
echo "================================" >>
$RESULTS/syshw
cat $RESULTS/sysconfig/Volume_Group_Pdisk_list* >>
$RESULTS/syshw

mv $RESULTS/syshw $RESULTS/syshw.`date
+"%y%m%d_%H%M%S"`.machinename
fi

if [ $SCOPE = 'power' ]
then
  #Run power test.
  powertest.sh All $SCALE_FACTOR $WITH_RF $Enable_Monitoring
fi

if [ $SCOPE = 'throughput' ]
then
  #Run throughput test.
  throughputtest.sh All $SCALE_FACTOR $WITH_RF $Enable_Monitoring $STREAM_COUNT
fi

#Clean up
rm $TEMPFILES/* > /dev/null 2>&1

#Move the $TPCH_ROOT/plans to $RESULTS/plans.
plan_count=$(ls -1 $TPCH_ROOT/plans/|wc -l)
if [ $plan_count -ne 0 ]
then
  mkdir $RESULTS/plans > /dev/null 2>&1
  mv $TPCH_ROOT/plans/*.html $RESULTS/plans/ >
  /dev/null 2>&1
fi
if [ $os = AIX ]
then
  mv $RESULTS/sysunused $RESULTS/sysunused.`date
  +"%y%m%d_%H%M%S"`
fi

#unset STREAM_COUNT. Otherwise same value will be remain in consecutive runs.
unset STREAM_COUNT
Appendix - E  ACID Transaction Source Code

E.1 Acid_atomic_main.tst

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%
%
%
% Purpose of this test is to run and verify the pass of the ACID Atomicity test.
%
% Atomicity test with rollback
%
% Atomicity test with commit
%
Test  "tpcd_acid_atomic_main.tst"
Description  "To run the ACID atomicity test"
stringconnect "dsn=qual_15_0;"
execute (select now()) into times
print 'Atomicity test start = ', times
include 'acid_functions.tst'
commit
%
% Atomicity test with rollback
%
print 'Starting atomicity test with rollback'
%
%include 'acid_atomic_setup.tst'
run test 'acid_atomic_setup.tst'

stringconnect "dsn=qual_15_0;"
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, delta
    close cur2
    execute (select count(*)
        from history
        where h_o_key =^ and h_l_key =^)
    substitute ordr, line, 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, delta
    execute (select o_totalprice, l_quantity, l_extendedprice from orders, lineitem
        where h_o_key =^ and h_l_key =^)
    substitute ordr, line, delta
    execute (select count(*)
        from history
        where h_o_key =^ and h_l_key =^)
    substitute ordr, line, delta
} ENDLOOP
commit
%
% Atomicity test with commit
%
stringconnect "dsn=qual_15_0;"
print 'Starting atomicity test with commit '
%
%include 'acid_atomic_setup.tst'
run test 'acid_atomic_setup.tst'

stringconnect "dsn=qual_15_0;"
open cur1 {select ordr, line, delta from aa_whattodo}

print 'Initial values:
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, delta
print 'O_totalprice = ',o_total,' l_quantity = ',l_quan,' l_extendedprice = ',l_price
execute (call acid_transaction(^, ^, ^, rprice, quantity, 
    tax, disc, extrprice, ototal)
) substitute ordr, line, delta
close cur2
execute (select count(*)
    from history
    where h_o_key =^ and h_l_key =^)
    substitute ordr, line, 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, delta
execute (select o_totalprice, l_quantity, l_extendedprice from orders, lineitem
    where h_o_key =^ and h_l_key =^)
    substitute ordr, line, delta
execute (select count(*)
    from history
    where h_o_key =^ and h_l_key =^)
    substitute ordr, line, delta
execute (select count(*)
    from history
    where h_o_key =^ and h_l_key =^)
    substitute ordr, line, delta
} ENDLOOP
commit
End Test

E.2 Acid_atomic_setup.tst

Test "acid_setup.tst"
Description "Creates aa_whattodo table"

string connect "dsn=qual_15_0;"
% Drop Table if found
print 'aa_whattodo!!'
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

E.3 Acid_functions.tst

% Created By: David Walrath
% Create Date: 7/15/1999
% This script creates various functions used by the Acid tests.

TPC Benchmark™ H FDR – IBM System Power 595 with Sybase IQ Single Application Server Edition v.15.1 ESD #1.2- Page 52
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)
--in SYSTEM
}
commit
execute {checkpoint}
print 'history table created'
print ''
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);
        DECLARE num         INT ;
        LOOP1: LOOP
            COMMIT;
            acid1:
            BEGIN ATOMIC
                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 IMPRECISE 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_quantity = quantity + delta;
                --
                -- Update LinItem
                --
                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
                            then message 'Completed ',o_key,' ......';
                        END IF;
                        END IF;
                        END acid1;
                        -- if c_sqlstate = '00000'
                        -- then commit;
                        -- else rollback;
                        -- end if;
            END IF;
    END LOOP1;
}
if c_sqlstate = '00000'
then LEAVE LOOP1;
end if;

select cast( rand() * 4.5 as int) into num;
message 'rollback sleep='; num, 'sqlstate='; c_sqlstate;
call dbo.sleep(num);
END LOOP LOOP1;
-- commit;
RETURN(0);
END;

print 'Acid transaction created'
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
    sum ("truncate"("truncate"("truncate"("truncate"("truncate="/l_extendedprice as numeric(26,16)),2) * (1 - round(cast(l_discount as numeric(26,16)),2)) * (1 + round(cast(l_tax as numeric(26,16)),2)),2) ,2)) into o_total
FROM lineitem WHERE l_orderkey = o_key;
END )

print 'Acid query created'
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

ORDERKEY INT
LINENUMBER INT
DELTA INT

BEGIN

declare seed     bigint;
declare rand_dbl  double precision;
declare rand_int  int;
declare out_key   int;
declare times cursor for select date-
diff(millisecond,convert(char(10),getdate(), 116),now(*));
declare random1 cursor for select rand(seed);
declare random cursor for select rand(dbl);
declare get_order cursor for
    select o_orderkey from orders where o_orderkey = orderkey;
declare get_linenumber cursor for
    select o_linenumber from lineitem
    where o_orderkey = orderkey;

open times;
fetch next times into seed;
open random1;
fresh 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;
fresh 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

print 'Generate_acid_values function created'
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

PARTKEY INT
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 date-
diff(millisecond,convert(char(10),getdate(), 116),now(*));
declare random1 cursor for select rand(seed);
declare random cursor for select rand(dbl);
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;
fresh next random1 into rand_dbl;
set out_key = 0;
loop1:
    while out_key = 0 LOOP
        open random;
        open get_supp;
        fetch next random into rand_dbl;
        set rand_int = rand_dbl * 6001215 + 1;
        close random;
        close get_supp;
    end loop loop1;
set orderkey = out_key;
open get_supp;
fetch next get_supp into suppkey;
close get_supp;

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

END
)
commit

print 'Generate_Ps_Values function created'
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
print 'Generate_Ps_Values function created'
print ''

print 'Creating Generate_acid_values2 function'
allow error -265
execute { DROP PROCEDURE generate_acid_values2 }
allow no error
execute{
create procedure generate_acid_values2(
in  streams  int,
in  txns int )
BEGIN
  declare seed    int;
  declare rand_dbl  double precision;
  declare i int;
  declare j int;
  declare random cursor for select rand();

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

  set i=1;
  set j=1;
loop1:
  while i < streams LOOP
    loop2:
      while j < txns LOOP
        insert into acid_table (stream,seqnum) values (i,j);
      end loop loop2;
end loop loop1;
  end loop loop1;

  update acid_table
    set line=cast(rand(rowid(acid_table)+seed)*1500000+1 as int);
  commit;
  update acid_table
    set okey=o_orderkey
      from orders where o_orderkey=line;
  update acid_table
    set delta=cast(rand(line)*100+1 as int);
  update acid_table
    set line=max(l_linenumber)
      from lineitem where l_orderkey=ordr;
  commit;
END
}

commit
print 'Generate_acid_values function2 created'
print ''

E.4  Run_atomicity
#!/bin/ksh
cd $ACID_ROOT/atomicity
dbtest $ACID_ROOT/atomicity/acid_atomic_main.tst >
$ACID_RESULTS/acid_atomic_main.out
roll_back=1
rm -i $ACID_RESULTS/atomc $ACID_RESULTS/atomr
while read line
  do
    if [ roll_back -eq 1 ]
      then
        commit_started=`echo $line |grep "Starting atom-
icity test with commit"`
        if [ ! -z "$commit_started" ]
          then
            roll_back=0
  fi
  fi
 done
E.5 Acid_consistency_main.tst

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% acid_consistency_main.tst
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Test  "tpch_acid_consistency_main.tst"  
Description  "To run the ACID consistency test"

stringconnect "dsn=qual_15_0;"
execute (select now(1)) into times
print 'Consistency test start = ', times
print ' ' 

include 'acid_functions.tst'
include 'acid_consistency_setup.tst'

%run test 'acid_consistency_setup.tst'
execute (select now(1)) into times
print 'Consistency test time = ', times
print ' ' 

run test '-o' 'acid_consist_user', i, '.ot'
'acid_consistency_query.tst'
disconnect

let i = 1
LOOP {
  if i > 22 then { BREAK LOOP } endif 
  let ot_file = "acid_consist_user", i, ".ot"
  let my_str = "stream=", i
  print ot_file, my_str
  start test '-o' ot_file my_str 'acid_consistency_txn.tst' sleep 900
  let i = i + 1
}

ENDLOOP

synchronize 23
%
% let the log flush...
sleep 900
stringconnect "dsn=qual_15_0;"
%include 'acid_consistency_query.tst'
run test '-o' 'acid_consistency_q1.ot' 'acid_consistency_query.tst'

execute (select now(1)) into times
print 'Consistency test end = ', times
print ' ' 

End Test

E.6 Acid_consistency_query.lst

Test 'tpch_acid_query'
Description 'perform the acid query.'

stringconnect "dsn=qual_15_0;"
open cur1 (select stream, seqnum, ordr, line, delta from acid_table
where seqnum > 10 order by seqnum)

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
  execute (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.lstTest
"acid_consistency_setup.tst"
Description "Creates acid_table table"

stringconnect "dsn=qual_15_0;"
execute { set option public.isolation_level=3 }
execute {set option public.query_plan='off'}
execute {set temporary option chained='on'}
execute {set option public.auto_commit=off}

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

execute {
create table acid_table (stream int null, seqnum int null, ordr int null, line int null, delta int null)
on SYSTEM
}

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

print 'acid_table created'
execute {create table latest(stream int, last int null) on SYSTEM}

LOOP ({let j = 1}; {j <= 22}; {let j = j + 1})
{
execute {insert into latest(stream, last) values (^, 0)}
substitute j
}
endloop

commit

print 'latest created'

LOOP ({let i = 1}; {i <= 22}; {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 = 2200
print 'Number of rows after load: ',ROWS

End Test

---------------------------------------------

E.7 Acid_consistency_setup.tst
---------------------------------------------

Test "acid_consistency_setup.tst"
Description "Creates acid_table table"

stringconnect "dsn=qual_15_0;"

execute {set option public.isolation_level=3}
execute {set option public.query_plan=off}
execute {set temporary option chained='on'}
execute {set option public.auto_commit=off}

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

execute {
create table acid_table (stream int null, seqnum int null, ordr int null, line int null, delta int null)
on SYSTEM
}

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

print 'acid_table created'
execute {create table latest(stream int, last int null) on SYSTEM}

LOOP ({let j = 1}; {j <= 22}; {let j = j + 1})
{
execute {insert into latest(stream, last) values (^, 0)}
substitute j
}
endloop

commit

print 'latest created'

LOOP ({let i = 1}; {i <= 22}; {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 = 2200
print 'Number of rows after load: ',ROWS

End Test

---------------------------------------------

E.8 Acid_durability_main.tst
---------------------------------------------

Test "tpch_acid_durability_main.tst"
Description "To run the ACID durability test"

stringconnect "dsn=qual_15_0;"

execute {set option public.isolation_level=3}
execute {set option public.query_plan=off}
execute {set temporary option chained='on'}
execute {set option public.auto_commit=off}

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

execute {
create table acid_table (stream int null, seqnum int null, ordr int null, line int null, delta int null)
on SYSTEM
}

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

print 'acid_table created'
execute (create table latest(stream int, last int null) on SYSTEM)

LOOP ({let j = 1}; {j <= 22}; {let j = j + 1})
{
execute {insert into latest(stream, last) values (^, 0)}
substitute j
}
endloop

commit

print 'latest created'

LOOP ({let i = 1}; {i <= 22}; {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 = 2200
print 'Number of rows after load: ',ROWS

End Test
execute {select now(*)} into times
print 'Durability test time = ', times
print '
run test -o' 'acid_duration_q1.ot' 'acid_duration_query.tst'
%start the fault to occur after 100 + transactions.
%start test -o' 'kill.out' 'acid_duration_kill_and_continue.tst'
LOOP( { let i = 1 }; { i <= 10 }; { let i = i + 1 } )
{
  let ot_file = "acid_dura_user", i, ".ot"
  let my_str = "stream=", i
  start test -o ot_file my_str 'acid_duration_txn.tst'
  sleep 950
}
ENDLOOP
print 'Out of loop. Parent waiting for synch'
synchronize 11
execute {select now(*)} into times
print 'Durability test end = ', times
print '
End Test

E.9 Acid_duration_query.tst

Test 'tpch_acid_query'
Description 'perform the acid query.'
stringconnect "dsn=qual_15_0;"
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
  execute { call acid_single_query (^) }
  substitute ord into o_total
  fetch {select cast(^ as numeric(12,2)) } substitute o_price into
  o_price
  fetch {select cast(^ as numeric(12,2)) } substitute o_total into l_total
  execute {call acid_single_query(^) substitute l_total into o_price
  substitute l_total into o_total}
  if o_price = o_total then { print 'Did not compare correctly' } ENDASSERT
  let n=n+1
}
ENDLOOP
print 'orderkey = ', ord, ' o_totalprice = ', o_price,
  '  acid query = ', l_total
assert o_price = l_total
print 'Did not compare correctly'
let n=n+1
} ENDLOOP
disconnect

E.10 Acid_duration_setup.tst

Test "acid_duration_setup.tst"
Description "Creates acid_table table"
stringconnect "dsn=qual_15_0;"
execute {set option public.query_plan='off'}
execute {set temporary option chained='on'}
execute {set option public.auto_commit=off}
execute { set option public.isolation_level=3 }
% 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 null,
    line int null,
    delta int null)
  on SYSTEM
}
fetch {select count(*) from acid_table } into ROWS
assert ROWS = 0
print 'Number of rows before load: ',ROWS
commit
print 'acid_table created'
allow error -141
execute { drop table latest }
allow no error
execute {create table latest(stream int ,last int null) on SYSTEM }
LOOP ({let j = 1}; {j <= 10}; {let j = j + 1 })
{
  execute { insert into latest(stream,last) values (^,0) }
}
commit
print 'latest created'
LOOP {let i = 1; { let i = i + 1}}
{
  LOOP {let j = 1; { let j = j + 1}}
  {
    execute { call generate_acid_values()} into ordr, line, delta
    execute { insert into acid_table values (^,^,^,^ ,^) }
    substitute i, ordr, line, delta
  } endloop
  print (j-1)*i
} endloop
print 'Number of rows after load: ', ROWS
End Test

E.11 Acid_durability_txn.tst

Test "tpcd_transaction1.tst"
Description "Run Acid Multiple Transactions"
string connect "dsn=qual_15_0;"
execute {select now(*)} into times
print 'Durability test start = ', times
print 'stream trans. o_key l_key p_key s_key delta date_t'
commit

let commit_delay=0
LOOP {let i = 1; { let i = i + 1}}
{
  fetch {select ordr, line, delta from acid_table
    where stream=^ and seqnum=^ }
  substitute ordr, line, delta
  commit

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

if i=101 then {
  let smallest=0
  allow error -210
  commit
  execute { set temporary option isolation_level=1 } 
  execute { select min(last) from latest } into smallest
  execute { set temporary option isolation_level=3 } 
  commit
  LOOP{ 
  if smallest >= 100 then {
    print 'Stream ', stream,
    ' Entering the Second phase with delays'
    'break loop'
  } endif
  let sleep_time =10
  sleep sleep_time
  commit
  execute { set temporary option isolation_level=1 } 
  execute { select min(last) from latest } into smallest
  execute { set temporary option isolation_level=3 } 
  }
}

execute { call set_acid_transaction( ^, ^, ^)
substitute commit_delay
commit

print 'transaction=',TxnId
print 'before commiting'
print 'stream,'
print 'txn ',i,' ',
print 'ordr, ' 
print 'line, '
print 'p_key, ' 
print 's_key, '
print 'delta '
print 'times, '
commit
execute { update latest set last=^ where stream=^ }
substitute 1,stream
execute { set temporary option isolation_level=1 }
execute { select max(last) from latest } into biggest
execute { select min(last) from latest } into smallest
commit
execute { set temporary option isolation_level=3 }

let num=120*(i-smallest)
if i+4>=biggest
then {let num=num+800}
endif

--print 'user',stream,' = ',num
sleep num

ENDLOOP

print 'Out of loop. Child waiting for synch'
synchronize 11

End Test

E.12  Acid_functions.tst

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); 
    DECLARE num            INT ; 
    LOOP1: LOOP
    COMMIT;
    acid1:
    BEGIN ATOMIC
    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_quantity = quantity + delta ;
    -- Update Lineitem
    -- UPDATE lineitem
    SET l_quantity = new_quantity,
    l_extendedprice = new_extprice
    WHERE l_orderkey=o_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 message 'Completed ',o_key,' .......';
END IF;
END IF;
END acid1;
-- if c_sqlstate = '00000'
-- then commit;
-- else rollback;
-- end if;
if c_sqlstate = '00000'
then LEAVE LOOP1;
end if;

-- if c_sqlstate = '00000'
-- then commit;
-- then rollback;
-- end if;

select cast( rand()*4.5 as int) into num;
message 'rollback sleep=', num,' sqlstate=',c_sqlstate;
call dbo.sleep(num);
END LOOP LOOP1;
-- commit ;
RETURN(0);
END;

print 'Acid transaction created'
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
   sum ( "truncate" ("truncate"
   round(cast(l_extendedprice as numeric(26,16)),2) * 
   (1 - round(cast(l_discount as numeric(26,16)),2)) )
   * (1 + round(cast(l_tax as numeric(26,16)),2)) )
2) into o_total
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 date-
diff(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
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 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 date-
diff(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
print 'Generate_Ps_Values function created'
print ' '}

TPC Benchmark™ H FDR – IBM System Power 595 with Sybase IQ Single Application Server Edition v.15.1 ESD #1.2- Page 61
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 date_diff(millisecond, convert(char(10), getdate(), 116), now("("));
declare random 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;
fetch next random1 into rand_dbl;
end loop;
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;
end loop;
set suppkey = out_key;
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
}

print 'Generate_Ps_Values function created'
print ''

print 'Creating Generate_acid_values2 function'
allow error -265
execute { DROP PROCEDURE generate_acid_values2 } allow no error
execute{
create procedure generate_acid_values2(
in streams int,
in txns int
)
BEGIN

declare seed int;
declare rand_dbl double precision;
declare rand_int int;
declare i int;
declare j int;

declare times cursor for select date_diff(millisecond, convert(char(10), getdate(), 116), now("("));
declare random1 cursor for select rand(seed);
declare random cursor for select rand();
open times;
fetch next times into seed;
fetch next random1 into rand_dbl;
set i=1;
set j=1;
loop1:
    while i < streams LOOP
        loop2:
            while j < txns LOOP
                insert into acid_table (stream,seqnum) values (i,j);
            end loop loop2;
        end loop loop1;
set i=i+1;
set j=j+1;
end loop;
update acid_table
set line=cast(rand(rowid(acid_table)+seed)*1500000+1 as int);
update acid_table
set okey=o_orderkey
from orders where o_orderkey=line;
update acid_table
set delta=cast(rand(line)*100+1 as int);
update acid_table
set line=max(l_linenumber)
from lineitem where l_orderkey=ordr;

END
}

print 'Generate_acid_values function2 created'
print ''

TPC Benchmark™ H FDR – IBM System Power 595 with Sybase IQ Single Application Server Edition v.15.1 ESD #1.2- Page 62
E.13 Acid_isolation_main1.tst

Test            "tpch_acid_isolation_main1.tst"
Description     "To run the ACID isolation test1"

stringconnect "dsn=qual_15_0;"
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'

E.14 Acid_isolation_main2.tst

Test            "tpcd_acid_isolation_main2.tst"
Description     "To run the ACID isolation test2"

stringconnect "dsn=qual_15_0;"
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'

E.15 Acid_isolation_main3.tst

Test            "tpcd_acid_isolation_main3.tst"
Description     "To run the ACID isolation test3"

stringconnect "dsn=qual_15_0;"
execute {select now(*)} into times
print ' '
print ' '
print 'Isolation test start = ', times
print ' '
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

E.16 Acid_isolation_main4.txt

Test "tpcd_acid_isolation_main4.tst"
Description "To run the ACID isolation test4"
stringconnect "dsn=qual_15_0;"
execute {select now(*)} into times
define 'start' as times
execute {select now(*)} into times
define 'start2' as 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

E.17 Acid_isolation_main5.tst

Test "tpcd_acid_isolation_main5.tst"
Description "To run the ACID isolation test5."
stringconnect "dsn=qual_15_0;"
execute {select now(*)} into times
define 'start' as times
execute {select now(*)} into times
define 'start2' as times
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

E.18 Acid_isolation_main6.tst

Test "tpcd_acid_isolation_main6.tst"
Description "To run the ACID isolation test6."
stringconnect "dsn=qual_15_0;"
execute {select now(*)} into times
define 'start' as times
execute {select now(*)} into times
define 'start2' as times
include 'acid_functions.tst'
include 'acid_isolation_setup.tst'
E.19 Acid_isolation_setup.tst

Test       "acid_isolation_setup.tst"
Description "Creates acid_isolation_table"

stringconnect "dsn=qual_15_0;"

% 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 {commit}
print 'acid_isolation_table CREATED!!'
execute {select ordr, line, delta from acid_isolation_table} into ordr, line, delta
execute { select round(cast(o_totalprice as numeric(18,2)),2) from orders where o_orderkey = ^} substitute ordr into o_total
print 'User 1 old values:  ' print 'user 1 ordr= ', ordr
print 'user 1 o_total= ', o_total

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( ^, ^,  ^) substitute ordr, line, delta into rprice, quantity, tax, disc, extprice, ototal}
execute {select now(*)} into times
print 'User 1 waiting to commit = ', times
synchronize 2
sleep 10000
execute {select now(*)} into times
print 'User 1 about to commit = ', times
commit
declare
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

End Test

E.20 Acid_isolation_test1.tst

Test       "tpch_acid_isolation_test1.tst"
Description "Run Acid isolation test 1"

stringconnect "dsn=qual_15_0;"
execute (select ordr, line, delta from acid_isolation_table) into ordr, line, delta
execute ( select round(cast(o_totalprice as numeric(18,2)),2) from orders where o_orderkey = ^) substitute ordr into o_total
print 'User 1 old values:  ' print 'user 1 ordr= ', ordr
print 'user 1 o_total= ', o_total

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( ^, ^,  ^) substitute ordr, line, delta into rprice, quantity, tax, disc, extprice, ototal}
execute (select now()) into times
print 'User 1 waiting to commit = ', times
sleep 10000
execute (select now()) into times
print 'User 1 about to commit = ', times
commit
declare
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

End Test

E.21 Acid_isolation_test1_query.tst

Test 'tpch_acid_query_isolation_test1'
Description 'perform the acid query for user2.'
stringconnect "dsn=qual_15_0;"

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
execute { 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

---

E.23 Acid_isolation_test2_query.tst

stringconnect "dsn=qual_15_0;"
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
execute { 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

---

E.24 Acid_isolation_test3_transaction1.tst

stringconnect "dsn=qual_15_0;"
substitute ordr into o_total
print 'User 1 new values:'
print 'user 1 ordr= ', ordr
print 'user 1 o_total= ', o_total
print ' '
% Script name: tpcd_acid_isolation_test3_transaction1.tst
% ---------------------------------------------------------------------  %
% This test could be run by itself, but it is recommended to run it as  %
% part of tpcd_acid_isolation_main3.tst file.                          %
% Test  "acid_isolation_test3_transaction1.tst"
Description "Run Acid Transaction 1 for isolation test 3"
stringconnect "dsn=qual_15_0;"
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 -- About to commit: ', times
commit
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 exprice, delta, exprice, quantity
into result1
% make it format nicely...
execute { select cast(^ as numeric(18,2)) } substitute result1 into result2
print ''
print 'User 1 -- result1 = '}

E.25 Acid_isolation_test3_transaction2.tst
% Test  "acid_isolation_test3_transaction2.tst"
Description "Run Acid Transaction 2 for isolation test 3"
stringconnect "dsn=qual_15_0;"
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
% generate a new set of values; we only use delta2
execute { call generate_acid_values() } into ordr2, line2, delta2
print ''
execute (select now()) into times
print 'User 2 -- The input data values for the Acid Transaction.'
print 'User 2 -- o_key = ', ordr2
print 'User 2 -- l_key= ', line2
print 'User 2 -- delta2 = ', delta2
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
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 ' ' 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 2 -- result1 = ' print '    txn2_extendedprice + (delta2 * (txn2_extendedprice/txn2_quantity))' print 'User 2 -- result1= ', result2 print ' ' End Test

E.26 Acid_isolation_test4_transaction1.tst

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Created by: Masood Dirin       %
%  Created Date: 5/25/1999       %
%  Script name: tpcd_acid_isolation_test3_transaction1.tst %
%  ---------------------------------------------------------------------  %
%                                                                         %
%  This test could be run by itself, but it is recommended to run it as %
%  part of tpcd_acid_isolation_main3.tst file.                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Top level commands:                                                   %
%                                                                         %
%  Test  "acid_isolation_test4_transaction1.tst"                        %
%  Description "Transaction 1 for isolation test 4"                    %
%                                                                         %
%  stringconnect "dsn=qual_15_0;"                                      %
%  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 'Isolation test 3 test start = ', times                       %
%  execute {select l_extendedprice from lineitem where l_linenumber=^ %
%    and l_orderkey=^}                                                  %
%    substitute line, ordr into extprice3                              %
%  execute {select round(cast(^ as numeric(20,6)),2) }                 %
%    substitute extprice3 into extprice4                               %
%  print ' ' print 'USER 1 -- extendedprice before acid transaction = ', extprice4 %
%  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'                                      %
%  SYNCRONIZE                                                           %
%  sleep 30000                                                          %
%  execute {select l_extendedprice from lineitem where l_linenumber=^ %
%    and l_orderkey=^}                                                  %
%    substitute line, ordr into extprice3                              %
%  execute {select round(cast(^ as numeric(20,6)),2) }                 %
%    substitute extprice3 into extprice4                               %
%  print ' ' print 'USER 1 -- extendedprice before rooling back = ', extprice4 %
%  execute {select now(*)} into times                                   %
%  print 'User 1 -- starting rollback: ', times                        %
%  rollback                                                             %
%  print ' ' execute {select now(*)} into times                       %
%  print 'USER 1 -- original extendedprice = ', extprice2               %
%  print 'USER 1 -- original quantity = ', quantity                    %
%  print ' '                                                            %
%  disconnect                                                           %
%  End Test                                                            %

E.27 Acid_isolation_test4_transaction2.tst

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Created by: Masood Dirin                                              %
%  Created Date: 5/25/1999                                               %
%  Script name: tpcd_acid_isolation_test3_transaction2.tst               %
%  --------------------------------------------------------------------  %
%                                                                         %
%  This test could be run by itself, but it is recommended to run it as %
%  part of tpcd_acid_isolation_main3.tst  file.                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Top level commands:                                                   %
%                                                                         %
%  Test  "acid_isolation_test4_transaction2.tst"                        %
%  Description "Transaction 2 for isolation test 4"                    %
%                                                                         %
%  stringconnect "dsn=qual_15_0;"                                      %
%  execute {select ordr, line, delta from acid_isolation_table}         %
%    into ordr, line, delta                                             %
%  print ' ' execute {select l_extendedprice from lineitem where l_linenumber=^ %
%    and l_orderkey=^}                                                  %
%    substitute line, ordr into extprice3                              %
%  execute {select round(cast(^ as numeric(20,6)),2) }                 %
%    substitute extprice3 into extprice4                               %
%  print ' '
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
sleep 5000

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

execute { call generate_ps_values() } into ps_ptky, ps_spky

print ''
execute {select now(*)} into times
print 'user 2 ps_partkey = ', ps_ptky
print 'user 2 ps_suppkey = ', ps_spky
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
execute {select now(*)} into times
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 ''
End Test

---

E.28 Acid_isolation_test5_query.tst

Test "tpcd_acid_isolation_query_test5.tst"
Description "Run Acid isolation query for test 5"

stringconnect "dsn=qual_15_0;"
synchronize 2
execute { call generate_ps_values() } into ps_ptky, ps_spky

print ''
execute {select now(*)} into times
print 'User 2 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 ''
End Test

---

E.29 Acid_isolation_test5_transaction1.tst

Test "tpcd_acid_isolation_test5_transaction1.tst"
Description "Run Acid isolation test 5 transaction 1"

stringconnect "dsn=qual_15_0;"
synchronize 2
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
sleep 5000

print ''
execute {select now(*)} into times
print 'User 2 beginning query = ', 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 query complete = ', times
print 'User 2 about to commit = ', times
commit
execute {select now(*)} into times
print 'User 2 transaction commit complete = ', times
print ''
End Test
**E.30 Acid_isolation_test6_query.tst**

---

**Test**  
"tpcd_acid_isolation_query_test6.tst"

**Description**  
"Run Acid isolation query for test 6"

`stringconnect "dsn=qual_15_0;"`

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_quantity, l_pk, l_sk

print 'User1 o_totalprice =', o_tprice
print 'User1 l_extendedprice =', l_price,' l_quantity =', l_quantity
print 'User1 l_partkey = ', l_pk,' l_suppkey = ', l_sk

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
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_quantity

print 'User1 o_totalprice =', o_tprice
print 'User1 l_extendedprice =', l_price,' l_quantity =', l_quantity

execute (select * from history where h_o_key=^ and h_date_t=(select max(h_date_t) from history where h_o_key=^))
substitute h_pk, h_sk into h_pk, h_sk, h_key, h_date, h_t

eexecute (select now(*) ) into times
print 'Stop time for User1 Q1 =', qstop

---

**Print User 1 history entry:**

print ' h_p_key = ', h_pk
print ' h_s_key = ', h_sk
print ' h_o_key = ', h_o_key
print ' h_l_key = ', h_lk
print ' h_delta = ', hda
print ' h_date_t = ', hdt

execute (select now(*) ) into times
print 'User 1 isolation test time = ', times
print ''
Acid_isolation_test6_transaction1.tst

stringconnect "dsn=qual_15_0;"

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_parkey, 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, exprice, 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 l_extendedprice = ', l_price,'  l_quantity = ', l_quant
print 'User 2 l_parkey      = ', l_ptky,'  l_suppkey = ', l_spky
print ' ' execute (select now(*) into times
print 'User 2 completed = ', times
print ''
print ''
execute {select * 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
print ''
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute {select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) into times
print 'User 2 completed = ', times
print ''
execute (select now(*) int
Dear Lotus,

Here is the requested quote for the System IBM Power 595 Server for the TPC-H benchmark using IBM System Storage DS4800.

<table>
<thead>
<tr>
<th>Description</th>
<th>Part No.</th>
<th>Source</th>
<th>Unit Price</th>
<th>Qty</th>
<th>Ext Price</th>
<th>Maint Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server 1:9119 Model 595</td>
<td>9119-FHA</td>
<td>1</td>
<td>91,000</td>
<td>1</td>
<td>91,000</td>
<td>63,192</td>
</tr>
<tr>
<td>GX Dual-port 12x HCA</td>
<td>1816</td>
<td>1</td>
<td>4,000</td>
<td>6</td>
<td>24,000</td>
<td></td>
</tr>
<tr>
<td>2.5M Enhanced 12x Cable (to connect to a gx card in)</td>
<td>1831</td>
<td>1</td>
<td>650</td>
<td>12</td>
<td>7,800</td>
<td></td>
</tr>
<tr>
<td>146.8 GB Ultra320 15K RPM</td>
<td>3279</td>
<td>1</td>
<td>1,299</td>
<td>5</td>
<td>6,495</td>
<td></td>
</tr>
<tr>
<td>0/8 - core POWER6 5.0GHz CoD, 0-core Active</td>
<td>4695</td>
<td>1</td>
<td>67,700</td>
<td>8</td>
<td>541,600</td>
<td>99,840</td>
</tr>
<tr>
<td>Single Processor Activation for FC 4695 (5.0 GHz)</td>
<td>4755</td>
<td>1</td>
<td>33,900</td>
<td>64</td>
<td>2,169,600</td>
<td>430,080</td>
</tr>
<tr>
<td>256GB Bundle DDR2 Activation</td>
<td>5681</td>
<td>1</td>
<td>83,200</td>
<td>2</td>
<td>166,400</td>
<td></td>
</tr>
<tr>
<td>0/8GB DDR2 Memory (4x2GB) DIMMS - 667 MHz</td>
<td>5694</td>
<td>1</td>
<td>650</td>
<td>64</td>
<td>41,600</td>
<td></td>
</tr>
<tr>
<td>2-PORT BASETX ETHT,PCI-X A</td>
<td>5767</td>
<td>1</td>
<td>699</td>
<td>1</td>
<td>699</td>
<td></td>
</tr>
<tr>
<td>4.7 GB IDE Slimline DVD-ROM Drive</td>
<td>5757</td>
<td>1</td>
<td>680</td>
<td>1</td>
<td>680</td>
<td></td>
</tr>
<tr>
<td>4 Gigabit Dual-port Fibre Channel PCI-E Adapter</td>
<td>5774</td>
<td>1</td>
<td>3,308</td>
<td>24</td>
<td>79,392</td>
<td></td>
</tr>
<tr>
<td>PCI-X DDR Dual - x4 SAS Adapter</td>
<td>5912</td>
<td>1</td>
<td>1,092</td>
<td>1</td>
<td>1,092</td>
<td></td>
</tr>
<tr>
<td>12x I/O Drawer PCI-E, SFF disk</td>
<td>5803</td>
<td>1</td>
<td>32,000</td>
<td>3</td>
<td>96,000</td>
<td>26,712</td>
</tr>
<tr>
<td>Bulk Power Regulator</td>
<td>6333</td>
<td>1</td>
<td>4,200</td>
<td>3</td>
<td>33,000</td>
<td></td>
</tr>
<tr>
<td>Bulk Power Distribution Assembly</td>
<td>6334</td>
<td>1</td>
<td>2,500</td>
<td>4</td>
<td>10,000</td>
<td></td>
</tr>
<tr>
<td>Slimline Doors (F&amp;R), H CEC AND I/O Expansion Rack</td>
<td>6869</td>
<td>1</td>
<td>8,000</td>
<td>1</td>
<td>8,000</td>
<td></td>
</tr>
<tr>
<td>UPC Cable Group, BDPI to I/O Drawer at A01</td>
<td>6941</td>
<td>1</td>
<td>500</td>
<td>1</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td>UPC Cable Group, BDPI to I/O Drawer A05</td>
<td>6942</td>
<td>1</td>
<td>500</td>
<td>1</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td>UPC Cable Group, BDPI to I/O Drawer A09</td>
<td>6943</td>
<td>1</td>
<td>500</td>
<td>1</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td>3rd &amp; 4th AMDs + UPIC Y-Cable Group (BPC to Fans)</td>
<td>6952</td>
<td>1</td>
<td>5,000</td>
<td>1</td>
<td>5,000</td>
<td></td>
</tr>
<tr>
<td>UPC Y-Cable Group BDPI to 1st Processor Node</td>
<td>6961</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UPC Y-Cable Group BDPI to 2nd Processor Node</td>
<td>6962</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UPC Y-Cable Group BDPI to 3rd Processor Node</td>
<td>6963</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UPC Y-Cable Group BDPI to 4th Processor Node</td>
<td>6964</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UPC Y-Cable Group BDPI to 5th Processor Node</td>
<td>6965</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UPC Y-Cable Group BDPI to 6th Processor Node</td>
<td>6966</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UPC Y-Cable Group BDPI to 7th Processor Node</td>
<td>6967</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>UPC Y-Cable Group BDPI to 8th Processor Node</td>
<td>6968</td>
<td>1</td>
<td>650</td>
<td>1</td>
<td>650</td>
<td></td>
</tr>
<tr>
<td>Ethernet Cable, 15M, Hardware Management</td>
<td>7802</td>
<td>1</td>
<td>34</td>
<td>2</td>
<td>68</td>
<td></td>
</tr>
<tr>
<td>Line Cord, 6AWG/Type W, 14ft, IEC309 100A Plug</td>
<td>8686</td>
<td>1</td>
<td>2,500</td>
<td>4</td>
<td>10,000</td>
<td></td>
</tr>
<tr>
<td>HMC 1:7042-C07 Desktop Hardw.Mgmt.Console</td>
<td>7042-C07</td>
<td>1</td>
<td>1,830</td>
<td>1</td>
<td>1,830</td>
<td>1,344</td>
</tr>
<tr>
<td>IBM T117 TFT 17 inch Color Display</td>
<td>3645</td>
<td>1</td>
<td>875</td>
<td>1</td>
<td>875</td>
<td></td>
</tr>
<tr>
<td>Quiet Touch Keyboard - USB, Business Black,</td>
<td>5951</td>
<td>1</td>
<td>104</td>
<td>1</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Power Cord (6-foot), To Wall (125V, 15A), Plug Type #</td>
<td>6470</td>
<td>1</td>
<td>18</td>
<td>2</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>Ethernet Cable, 6M, Hardware Management</td>
<td>7801</td>
<td>1</td>
<td>15</td>
<td>1</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Mouse - Business Black with Keyboard Attachment</td>
<td>8841</td>
<td>1</td>
<td>78</td>
<td>1</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>PCI-e 1Gb Ethernet UTP 2Port</td>
<td>5767</td>
<td>1</td>
<td>699</td>
<td>1</td>
<td>699</td>
<td></td>
</tr>
</tbody>
</table>

Subtotal 3,303,343 621,168
Please note that the discount used is based on US list prices for similar quantities & configurations including pre-payment for maintenance. The discount applies to the totality of this quote and it does not apply to individual items listed.

For more information on:
For System p products:  [http://www-03.ibm.com/systems/p](http://www-03.ibm.com/systems/p)

For additional information, please contact me directly:
Dan Hebrank
IBM Sales & Distribution STG Sales
1-314-283-4674
<table>
<thead>
<tr>
<th>Catalogue Number</th>
<th>Product Description</th>
<th>License Type</th>
<th>Machine</th>
<th>PS</th>
<th>List Price Per Unit</th>
<th>Quantity</th>
<th>Price</th>
<th>Discount</th>
<th>Extended Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sybase IQ Single App Svr, per cpu core</td>
<td>CP</td>
<td>IBM AIX</td>
<td>P</td>
<td>2,595</td>
<td>64</td>
<td>166,080</td>
<td>24,912</td>
<td>141,168,00</td>
</tr>
<tr>
<td>2</td>
<td>Sybase IQ Single App Svr, per cpu core</td>
<td>CP</td>
<td>IBM AIX</td>
<td>P</td>
<td>1,713</td>
<td>64</td>
<td>169,832</td>
<td>15,445</td>
<td>93,387,20</td>
</tr>
</tbody>
</table>

**Quote Date:** 11/23/09

**Valid thru:** 11/23/09

**Total:** $254,555.30

**Payment terms:** Net 30 Days