Mixed Workload CH-benCHmark

Michael Seibold #  Florian Funke #  Alfons Kemper #

# Technische Universität München (TUM)
Outline

Overview

Data Scaling

Benchmark Parameters

Performance Metrics

Conclusion
Outline

Overview

Data Scaling

Benchmark Parameters

Performance Metrics

Conclusion
Operational (Real-Time) Business Intelligence

- Process analytical queries directly on operational database
- Without impeding mission-critical transaction processing
- Enabled by advances in hardware architecture

Use cases: for low latency analytics

- BI dashboard: low-latency monitoring
- Interactive customer response (e.g. upsell, antifraud)

---

1 Curt Monash's Blog (April 10, 2011), Teradata
Mixed Workload CH-benCHmark

- Outcome of the Dagstuhl Seminar "Robust Query Processing" Fall 2010
  - Organized by Goetz Graefe et al.
  - Breakout Working Group: Workload Management (headed by: Harumi Kuno)

- Research project at Technische Universität München
  http://www-db.in.tum.de/research/projects/CH-benCHmark/

- Publications


- On-going work: integration into oltpbenchmark.com
  (open-source benchmark framework developed by Carlo Curino, Yahoo! Research)
Mixed Workload

Analytical + transactional loads are

- run in parallel
- on the same tables
- in a single database system
Single-workload benchmarks

- TPC-C: Transactional workload
- TPC-H:
  - Analytical load and refresh functions
  - BUT: data modification operations performed in bulk when no queries are running
    → No mixed workload
- TPC-C and TPC-H can be combined
  - Installed on a single database instance and run in parallel
  - BUT: different loads are run on separate data
    → No mixed workload
CH-benCHmark

- Derived from two standardized and widely accepted benchmarks: TPC-C and TPC-H

- Alternative: Composite Benchmark for Transaction processing and operational Reporting (CBTR)
  - Installed on a single database instance and run in parallel
  - BUT: different loads are run on separate data
  - uses the actual data of a real enterprise
Combine TPC-C and TPC-H: Schema and Initial Database Population

- Supplier (10k)
  - in Nation (62)
  - sup-by Warehouse (W)
  - sup-by Stock (W * 100k)
  - sup-by Item (100k)

- Warehouse (W)
  - sup-by Stock (W * 100k)
  - stored Stock (W * 100k)
  - history History (W * 30k+)

- Stock (W * 100k)
  - available Stock (W * 100k)

- New-Order (W * 9k+)
  - available New-Order (W * 9k+)

- Customer (W * 30k+)
  - pending Customer (W * 30k+)
  - issues Customer (W * 30k+)

- District (W * 10)
  - located-in District (W * 10)

- Region (5)
  - Region (5)

- Nation (62)
  - Nation (62)

- Item (100k)
  - Item (100k)

- Order-Line (W * 300k+)
  - Order-Line (W * 300k+)

- Order (W * 30k+)
  - Order (W * 30k+)

- Order (W * 30k+)
  - Order (W * 30k+)

- Warehouse (W)
  - Warehouse (W)

- Stock (W * 100k)
  - Stock (W * 100k)

- New-Order (W * 9k+)
  - New-Order (W * 9k+)

- Customer (W * 30k+)
  - Customer (W * 30k+)

- District (W * 10)
  - District (W * 10)

- Region (5)
  - Region (5)
**CH-benCHmark**

- **Transactional load**
  - Unchanged TPC-C business transactions
  - Processed on unchanged TPC-C tables
  - Initial database population according to TPC-C spec
  - But: CH-Benchmark does not simulate terminals
    → Transactional sessions (no think times or keying times)

- **Analytical load**
  - Read-only query suite modeled after TPC-H
  - All 22 TPC-H queries reformulated to match extended TPC-C schema
  - Goal: Preserve business semantics and syntactical structure

- Transactional load continuously updates (and expands) database
  → TPC-H refresh functions are omitted
Outline

Overview

Data Scaling

Benchmark Parameters

Performance Metrics

Conclusion
Data Scaling

• Scaling model: Maintains the ratio between
  • Transactional load presented to the system under test
  • Cardinality of the tables accessed by the transactions
  • Required space for storage
  • Number of terminals or sessions generating the system load
Data Scaling: TPC-C

Continuous scaling model

- Data volume has to be increased for higher transaction load
- Number of warehouses determines not only the cardinality of the other tables, but also the number of terminals
- Each terminal generates a limited load due to think times and keying times

*Increase transaction load*

→ *Increase number of terminals*
→ *Requires a higher number of warehouses*
→ *Results in a larger data volume*

PROBLEM:

- Size of initial database population depends on maximum transactional load

→ Mixed workload: Analytical queries on different data volumes
Data Scaling: TPC-H

Fixed scale factor model
- Database size is set by a scale factor
- Regardless of system performance
Data Scaling: CH-benCHmark

Fixed scale factor model (like TPC-H)

- Fixed number of warehouses, no wait times
- Not necessary to increase the number of warehouses for achieving higher throughput rates
  → Allow for high transaction rates on small database sizes

- Scenario: Emerging main-memory database systems (e.g. HyPer, VoltDB, SAP HANA)

Different transactional throughput rates

→ Difference in data volume growth

→ Normalization of response times for analytical queries
Outline

Overview

Data Scaling

Benchmark Parameters

Performance Metrics

Conclusion
Model different workload characteristics and application scenarios: e.g. data volumes that fit in main-memory or require secondary storage
Outline

Overview

Data Scaling

Benchmark Parameters

Performance Metrics

Conclusion
## Performance Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transactional Throughput (tpmCH)</td>
<td>Total number of New-Order transactions completed during the measurement interval divided by the elapsed time of the interval in minutes; New-Order transactions that rollback due to simulated user data entry errors must be included; Similar to the Maximum Qualified Throughput metric of TPC-C</td>
</tr>
<tr>
<td>Geometric Mean (ms)</td>
<td>For each query type the average response times of queries completed during the measurement interval is determined and the geometric mean of the average response times of all query types is reported.</td>
</tr>
<tr>
<td>Duration Per Query Set (s)</td>
<td>Query set consists of 22 queries, one query per query type; Sum of the average response times of all query types; Reported in seconds</td>
</tr>
</tbody>
</table>
| Queries Per Hour (QphCH)                 | Completed queries per hour; Can be deduced from Duration Per Query Set metric as follows: \[
\frac{60 \text{ minutes}}{\text{Duration Per Query Set (in seconds)}} \times 22
\] |
Performance Metrics

*Insert throughput metric of the transactional component interferes with the response-time metric of the analytic component of the mixed workload*

*Transactional Throughput* metric and the *Queries Per Hour* metric are not combined in a single metric

- Transactional load generated by transactional sessions is not limited by sleep times or keying times, but can only be throttled by the system under test
- Competing systems under test may prioritize transactions and analytical queries differently

→ Aspect would get lost if a single metric were used
<table>
<thead>
<tr>
<th>Q#</th>
<th>average response times (ms)</th>
<th>average normalized response times</th>
<th>average response times (ms)</th>
<th>average normalized response times</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>368</td>
<td>42</td>
<td>63632</td>
<td>6001</td>
</tr>
<tr>
<td>Q2</td>
<td>126</td>
<td>126</td>
<td>890</td>
<td>890</td>
</tr>
<tr>
<td>Q3</td>
<td>1012</td>
<td>121</td>
<td>2606</td>
<td>281</td>
</tr>
<tr>
<td>Q4</td>
<td>3368</td>
<td>319</td>
<td>15202</td>
<td>1403</td>
</tr>
<tr>
<td>Q5</td>
<td>5722</td>
<td>818</td>
<td>17104</td>
<td>1624</td>
</tr>
<tr>
<td>Q6</td>
<td>168</td>
<td>20</td>
<td>22549</td>
<td>2125</td>
</tr>
<tr>
<td>Q7</td>
<td>1782</td>
<td>260</td>
<td>177</td>
<td>65</td>
</tr>
<tr>
<td>Q8</td>
<td>729</td>
<td>131</td>
<td>4270</td>
<td>517</td>
</tr>
<tr>
<td>Q9</td>
<td>703</td>
<td>105</td>
<td>4829</td>
<td>551</td>
</tr>
<tr>
<td>Q10</td>
<td>6760</td>
<td>725</td>
<td>141634</td>
<td>7510</td>
</tr>
<tr>
<td>Q11</td>
<td>57</td>
<td>57</td>
<td>538</td>
<td>538</td>
</tr>
<tr>
<td>Q12</td>
<td>2497</td>
<td>254</td>
<td>39550</td>
<td>3619</td>
</tr>
<tr>
<td>Q13</td>
<td>120</td>
<td>27</td>
<td>1437</td>
<td>314</td>
</tr>
<tr>
<td>Q14</td>
<td>1430</td>
<td>163</td>
<td>42819</td>
<td>4165</td>
</tr>
<tr>
<td>Q15</td>
<td>2876</td>
<td>368</td>
<td>141051</td>
<td>14804</td>
</tr>
<tr>
<td>Q16</td>
<td>525</td>
<td>525</td>
<td>7767</td>
<td>7767</td>
</tr>
<tr>
<td>Q17</td>
<td>651</td>
<td>79</td>
<td>23405</td>
<td>1926</td>
</tr>
<tr>
<td>Q18</td>
<td>7828</td>
<td>587</td>
<td>332413</td>
<td>30598</td>
</tr>
<tr>
<td>Q19</td>
<td>614</td>
<td>70</td>
<td>27786</td>
<td>2683</td>
</tr>
<tr>
<td>Q20</td>
<td>517</td>
<td>73</td>
<td>6863</td>
<td>819</td>
</tr>
<tr>
<td>Q21</td>
<td>2552</td>
<td>280</td>
<td>31812</td>
<td>2710</td>
</tr>
<tr>
<td>Q22</td>
<td>245</td>
<td>53</td>
<td>1847</td>
<td>396</td>
</tr>
</tbody>
</table>

| Geometric mean (ms) | 146 | 1621 |
| Normalized geometric mean | 859 | 10814 |
| Duration per query set (s) | 41  | 931  |
| Normalized duration per query set | 6  | 92   |
| Queries per hour (QphH) | 1949 | 88   |
| Normalized queries per hour (QphH) | 15222 | 868 |
Outline

Overview

Data Scaling

Benchmark Parameters

Performance Metrics

Conclusion
Conclusion

- Introduced mixed workload CH-benCHmark
- Performance metrics for mixed workload benchmarks
  - Normalization
  - Alternative:
    - Not measure peak transactional and analytical performance
    - But measure how much analytical throughput can be achieved while a fixed transactional throughput rate is maintained

- Future Work: Measure resource usage or energy consumption while fixed transactional and analytical performance is maintained
Thank you for your attention