Autonomous Data Ingestion Tuning in Data Warehouse Accelerators

Knut Stolze, Felix Beier, Jens Müller
IBM Germany Research & Development
2017-03-08
Disclaimer

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY.

WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.

IN ADDITION, THIS INFORMATION IS BASED ON IBM’S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION.

NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, OR SHALL HAVE THE EFFECT OF:
- CREATING ANY WARRANTY OR REPRESENTATION FROM IBM (OR ITS AFFILIATES OR ITS OR THEIR SUPPLIERS AND/OR LICENSORS); OR
- ALTERING THE TERMS AND CONDITIONS OF THE APPLICABLE LICENSE AGREEMENT GOVERNING THE USE OF IBM SOFTWARE.
Agenda

- IDAA Overview
- Load Processing
- Autonomous Load Performance Tuning / Smart Load
- Preliminary Performance Evaluation
- Summary & Outlook
IBM DB2 Analytics Accelerator
Loading Data from DB2 z/OS into IDAA

**Full Table Reload**

**Partial Reload**

**Incremental Update**
Incremental Update

• Use replication to replay changes in accelerator

• Low latency

• Concurrent to queries; commit scope on transaction boundaries
Batch Update Scenarios

• Reload many tables fully or partially, e.g. in batch window

• Fully concurrent to queries; commit scope is LOAD operation

• Excellent throughput

• Tables processed sequentially

• Partitions of partitioned tables loaded concurrently
Loading Data from DB2 z/OS into IDAA
Challenges

• Load optimized for throughput of large tables only

• Many small tables handled not very efficiently

• Manual management of parallelism by starting multiple, independent load processes (by customer)

• Lack of control of resources / workload balancing amongst load operations and between load and other operations (e.g. query)
Smart Load

• Automatically manage scale concurrent load operations up and down

• Considers involved resources
  - CPU consumption in Netezza backend
  - CPU consumption on System z
  - Network saturation
  - I/O utilization in Netezza backend
Characteristics

• Oscillating behavior for scaling up/down possible, but not a problem
  • Increment step is only 1
  • Already running load streams are not terminated prematurely

• Expected resource consumption for new load stream is estimated by averaging over history of prior load streams
Preliminary Performance Evaluation

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Sequential Load</th>
<th>Smart Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 tables with 1 row each</td>
<td>4min 16s</td>
<td>13s</td>
</tr>
<tr>
<td>TPC-H (30MB)</td>
<td>42s</td>
<td>7s</td>
</tr>
</tbody>
</table>

**Improvements:**

- Keeping load streams open for multiple tables or table partitions
- Inter-table parallelism very beneficial for non-partitioned tables
Summary & Outlook

• Introduction of Load Scheduler to consolidate management of concurrent load operations

• Noticeably improved throughput

• Significantly easier to use

Next Steps:

• Product integration (currently ongoing)

• Deeper integration with workload management in Netezza backend