Big Data MDX with Mondrian and Apache Kylin

Sébastien Jelsch
London, 7-11-2015
Agenda

- OLAP-on-Hadoop with Apache Kylin
- Features
- Apache Kylin & Mondrian
- Conclusion & Discussion
Agenda

- OLAP-on-Hadoop with Apache Kylin
- Features
- Apache Kylin & Mondrian
- Conclusion & Discussion
Situation

- More and more data becoming available on Hadoop
- Limitations in existing Business Intelligence Tools
  - Limited support for Hadoop
  - Data size growing exponentially
  - High latency of interactive queries
- Challenges to adapt Hadoop for interactive analysis
  - OLAP capability on Hadoop ecosystem not ready yet
OLAP and Big Data

Goals

▫ Full OLAP capability and advanced functionality
▫ Interactive analysis in subseconds
▫ ANSI SQL or MDX for analysts and engineers
▫ Seamless integration with BI Tools
▫ High concurrency with thousands of end users
▫ Distributed and scale out architecture for large data volume
What is Apache Kylin?

Solution: Apache Kylin

Extreme OLAP Engine for Big Data
- Distributed Analytics Engine from eBay
- OLAP-on-Hadoop
- Provides SQL interface for multidimensional analysis
- Based on Hadoop ecosystem

Open Source on: 1. October 2014
Accepted into incubation: 25. November 2014
Current version: 1.1 (25. October 2015)
Short introduction into OLAP

OLAP Cube

Beer

Wine

Water

Berlin

London

Paris

Big Data MDX with Mondrian and Apache Kylin

Sébastien Jelsch
Apache Kylin: Architecture

- 3rd Party App
- Web App
- BI Tools

- SQL
- SQL
- JDBC / ODBC

REST Server
Query Engine
Routing

- Metadata
- Cube Build Engine

Hive
HDFS

OLAP Cube (HBase)

Star Schema Data
Key Value Data

Mid Latency
Low Latency
Agenda

- OLAP-on-Hadoop with Apache Kylin
- Features
  - Apache Kylin & Mondrian
  - Conclusion & Discussion
### Cube Designer

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Table Name</th>
<th>Type</th>
<th>Column</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>quantity_dimension</td>
<td>DEFAULT.SJ.QB_FACTS</td>
<td>normal</td>
<td>[&quot;LO_QUANTITY&quot;]</td>
</tr>
<tr>
<td>2</td>
<td>customer_hierarchy</td>
<td>DEFAULT.LO_CUSTKEY</td>
<td>hierarchy</td>
<td>[&quot;LO_CUSTKEYREGIONLEVEL&quot;,&quot;LO_CUSTKEYNATIONLEVEL&quot;,&quot;LO_CUSTKEYCITYLEVEL&quot;]</td>
</tr>
<tr>
<td>3</td>
<td>part_hierarchy</td>
<td>DEFAULT.LO_PARTKEY</td>
<td>hierarchy</td>
<td>[&quot;LO_PARTKEYMFGLEVEL&quot;,&quot;LO_PARTKEYCATEGORYLEVEL&quot;,&quot;LO_PARTKEYBRANDLEVEL&quot;]</td>
</tr>
</tbody>
</table>

Note: inputs with light blue border are mandatory.

← Back to My Cubes
Cube Designer

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Expression</th>
<th>Param Type</th>
<th>Param Value</th>
<th>Return Type</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><em>COUNT</em></td>
<td>COUNT</td>
<td>constant</td>
<td>1</td>
<td>bigint</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>revenue_sum</td>
<td>SUM</td>
<td>column</td>
<td>LO_REVENUE</td>
<td>bigint</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>extended_price_sum</td>
<td>SUM</td>
<td>column</td>
<td>LO_EXTENDEDPRICE</td>
<td>decimal</td>
<td></td>
</tr>
</tbody>
</table>

Note: Inputs with light blue border are mandatory.

← Back to My Cubes
WHERE

product.category = "Shoes"

Tips

1. Where clause to filter data from source
2. Do not include date column which will be used for incremental refresh
3. Do not include "Where"
4. Please verify SQL when finish cube design from SQL view of cube

Note: inputs with light blue border are mandatory.

Back to My Cubes
Apache Kylin: Cube Designer

Cube Designer

Partition Date Column

DEFAULT.SJ_OB_FACTS.LO_ORDERD...
(data format in column should be 'YYYY-MM-DD')

Start Date

2015-09-01

Tips

1. Not required, leave as default if this cube always need full build
2. Partition column will select 'date' or 'string' type column from fact table
3. If column selected, please indicate start date to just pull certain data from source

Note: inputs with light blue border are mandatory.

Back to My Cubes
## Apache Kylin: Monitoring

### Jobs Table

<table>
<thead>
<tr>
<th>Job Name</th>
<th>Cube</th>
<th>Progress</th>
<th>Last Modified Time</th>
<th>Duration</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>complete_cube - 197001010000000_2922789940617071255 - BUILD - PDT 2015-09-13 03:32:19</td>
<td>complete_cube</td>
<td>64.00%</td>
<td>2015-09-13 03:07:16 PST</td>
<td>34.93 mins</td>
<td>Action</td>
</tr>
<tr>
<td>complete_cube - 197001010000000_2922789940617071255 - BUILD - PDT 2015-09-06 18:21:43</td>
<td>complete_cube</td>
<td>100%</td>
<td>2015-09-06 17:59:52 PST</td>
<td>37.63 mins</td>
<td>N/A</td>
</tr>
</tbody>
</table>

- **#16 Step Name**: Build N-Dimension Cuboid Data : 3-Dimension  
  **Data Size**: 69.27 KB  
  **Duration**: 0.53 mins

- **#17 Step Name**: Build N-Dimension Cuboid Data : 2-Dimension  
  **Duration**: 0 seconds

- **#21 Step Name**: Create HTable  
  **Duration**: 0 seconds

- **#22 Step Name**: Convert Cuboid Data to HFile  
  **Duration**: 0 seconds
Apache Kylin: SQL Interface
Agenda

- OLAP-on-Hadoop with Apache Kylin
- Features
- Apache Kylin & Mondrian
- Conclusion & Discussion
SQL returns 2-dimensional result set
For more dimensions SQL was not designed

Wish:
- Multidimensional result set
- Consider hierarchies and levels in the data

Query Language: MDX
Mondrian

- OLAP Engine
- Transforms MDX queries into SQL
- Multidimensional representation of data
- Integrated into Saiku / Pentahos Business Analytics Platform
- Expandable through SQL dialects
e.g. MySQL, Postgres, Hive, Impala, ...
Apache Kylin + Mondrian: Idea

![Diagram showing the integration of Apache Kylin and Mondrian.](image)
Apache Kylin + Mondrian: Implementation

Work done:
- Kylin dialect created
- Optimized Kylin’s JDBC driver
- Bugs fixed to get Mondrian working with Kylin

TBD:
- Integrate Kylin dialect into Mondrians official code*
- Make every MDX query executable

Successful tests**:
- Current Saiku and Mondrian 4.4
- Current Saiku and Mondrian 3.x (not tested very well)

* Pull Request: https://github.com/pentaho/mondrian/pull/480
** Github Project: https://github.com/mustangore/kylin-mondrian-interaction
Apache Kylin + Mondrian: Examples

User Management
- Add User
- admin
- smith

Data Source Management
- Data Sources
  - Add Data Source
  - foodmart
  - Kylin Data Source
- Schema
  - Add Schema
  - foodmart4.xml
  - MondrianSchema.xml

Maintenance
- Backup/Restore

Create Data Source
- Name: Kylin Data Source
- Connection Type: Mondrian
- URL: jdbc:kylin://srv-bds-01.inovex.de:7070/complete
- Schema: /datasources/MondrianSchema.xml
- Jdbc Driver: org.apache.kylin.jdbc.Driver
- Username: admin
- Password: ●●●●●

Advanced

[Buttons: Refresh Cache, Save, Remove]
```java
public class KylinMondrianOlap4J {
    public static void main(String[] args) throws ClassNotFoundException, SQLException {
        Class.forName("mondrian.olap4j.MondrianOlap4jDriver");

        Connection connection = DriverManager.getConnection(
            "jdbc:mondrian:"
            + "jdbc=kylin://YOUR_URL]:7070/" + YOUR_PROJECT_NAME + "=";
            + "JdbcDrivers=org.apache.kylin.jdbc.Driver;"
            + "JdbcUser=YOUR_USER;" // Default: admin
            + "JdbcPassword=YOUR_PASSWORD;" // Default: KYLIN
            + "Catalog=file:absolute/path/to/your/mondrianSchema.xml;";
        
        // We are dealing with an OLAP connection. we must unwrap it.
        final OlapConnection olapConnection = connection.unwrap(OlapConnection.class);

        // Prepare a statement.
        final OlapStatement olapStatement = olapConnection.createStatement();

        // We use the utility formatter.
        RectangularCellSetFormatter formatter = new RectangularCellSetFormatter(false);

        // Your MDX Statement
        String mdxStatement = "{YOUR_MDX_QUERY}";

        CellSet cellSet = olapStatement.executeOlapQuery(mdxStatement);

        // Print out.
        PrintWriter writer = new PrintWriter(System.out);
        formatter.format(cellSet, writer);
        writer.flush();
    }
}
Agenda

- OLAP-on-Hadoop with Apache Kylin
- Features
- Apache Kylin & Mondrian
- Conclusion & Discussion
Apache Kylin: Conclusion

- Extremely fast and scalable OLAP Engine
- OLAP-on-Hadoop
- Depends on Apache Hadoop infrastructure
- MOLAP Cube
- Incremental refresh of cubes
- Integration into existing BI Tools
- MDX queries with Mondrian possible (ongoing work)
Thank you for your attention

Contact

Sébastien Jelsch
Big Data Scientist

inovex GmbH
Office Karlsruhe
Ludwig-Erhard-Allee 6
76131 Karlsruhe

Tel: +49 176 - 45786280
E-Mail: sjelsch@inovex.de
Twitter: @inovexgmbh | @Mustangore
Introduction into OLAP

Cube: All combinations
Cuboid: One single combination

Number cuboids growing exponentially
**Problem:** Number of Cuboids grows exponentially

**Example:**
- Cube with 30 dimensions
  - Number of Cuboids: $2^{30} > 1$ billion

**Solution:** Partial Cube

Classificate the OLAP Cube in Aggregation Groups

**Example:**
- 30 dimensions splitted into 3 groups of 10 dimensions
  - Number of Cuboids: $2^{10} + 2^{10} + 2^{10} = 3072 < 1$ billion
Apache Kylin: Cube Build Process

1. Source Hive Tables
2. HiveQL
3. MapReduce

N-Cuboid
HDFS Sequence Files

HiveQL
Intermediate Hive Table

Dimension Dictionaries
Apache Kylin: Cube Build Process

1. **N-Cuboid**
   - HDFS Sequence Files
   - MapReduce

2. **N-1-Cuboid**
   - HDFS Sequence Files
   - MapReduce
   - MapReduce

3. **0-Cuboid**
   - HDFS Sequence Files
   - MapReduce

4. **HBase**
   - Bulk Import
   - HFiles