IBM Big SQL 3.0: An Introduction

October 29, 2014

John Poelman
BigInsights performance team, IBM Silicon Valley Lab (San Jose)
poelman@us.ibm.com

Special thanks to…

- Cindy Saracco
- Scott Gray
- Hebert Pereyra
- Bert Van der Linden
- Adriana Zubiri
- Simon Harris
- Klaus Roder

Please note

- This presentation is provided as-is.
- The content is accurate on a “best effort” basis.
- IBM’s plans can change at any time.
- Do not make decisions or rely on forward looking information stated or implied in this presentation. (Example: BigInsights beta content listed on slide 7).

Agenda

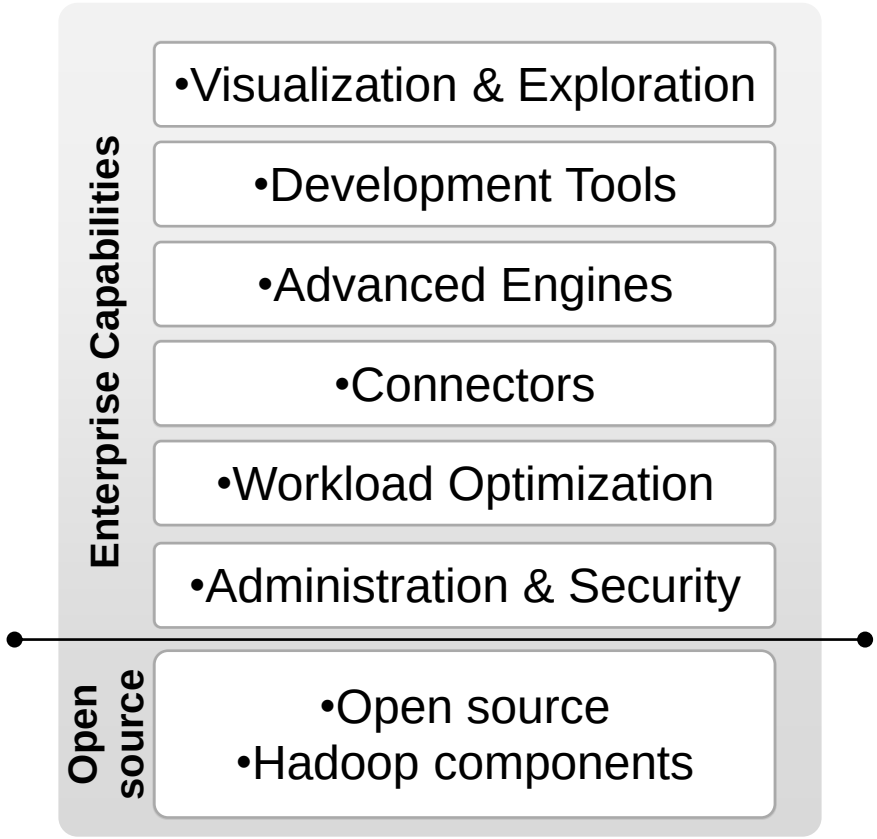
- **Why SQL access for Hadoop?**
- **Overview of**
  - IBM BigInsights
  - IBM Big SQL 3.0

Why SQL Access for Hadoop?

- SQL opens the data to a much wider audience
- Familiar, widely known syntax
- Lower cost data warehouse
- We see many open source and proprietary offerings in this space

What is BigInsights?

- **IBM's Hadoop distribution**
- **Builds on open source Hadoop capabilities for enterprise class deployments**



BigInsights Overview

Value-Added Capabilities, Optionally Deployable

- **SQL on Hadoop**
  Big SQL – optimized ANSI compliant SQL
- **Application Tooling**
  Toolkits and accelerators
- **Search**
  BigIndex and Data Explorer
- **Data Exploration**
  BigSheets "schema-on-read"
- **Predictive Modeling**
  Big R – scalable data mining
- **Text Analytics**
  Advanced text processing with AQL
- **Real-time Analytics**
  InfoSphere Streams
- **Data Governance and Security**
  Data Click, LDAP, Secure cluster
- **Storage Integration**
  GPFS - POSIX Distributed Filesystem
- **Enterprise Features**
  Adaptive MapReduce, Recoverable jobs

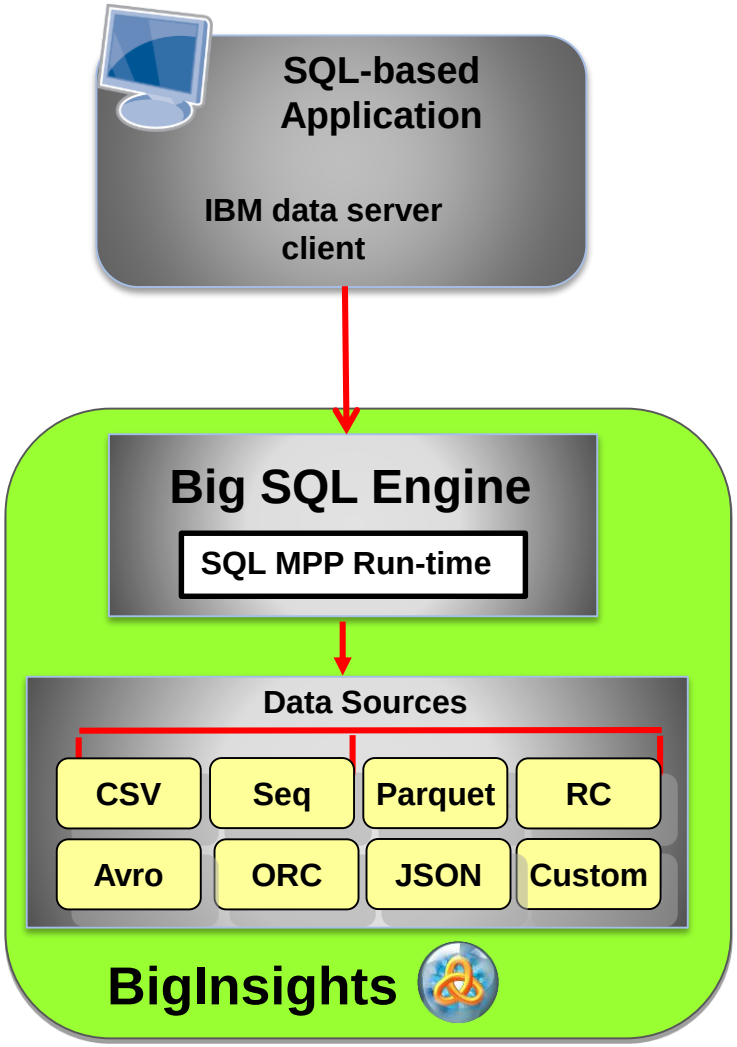
100% Standard Apache Open-Source Components

•Oozie	•Jaql	•Zookeeper	•Hive	•HCatalog
•HDFS	•MapReduce	•HBase	•Flume	•Sqoop
•YARN*	•Spark*	•Avro	•Pig	•Solr/Lucene

*** In current BigInsights beta**

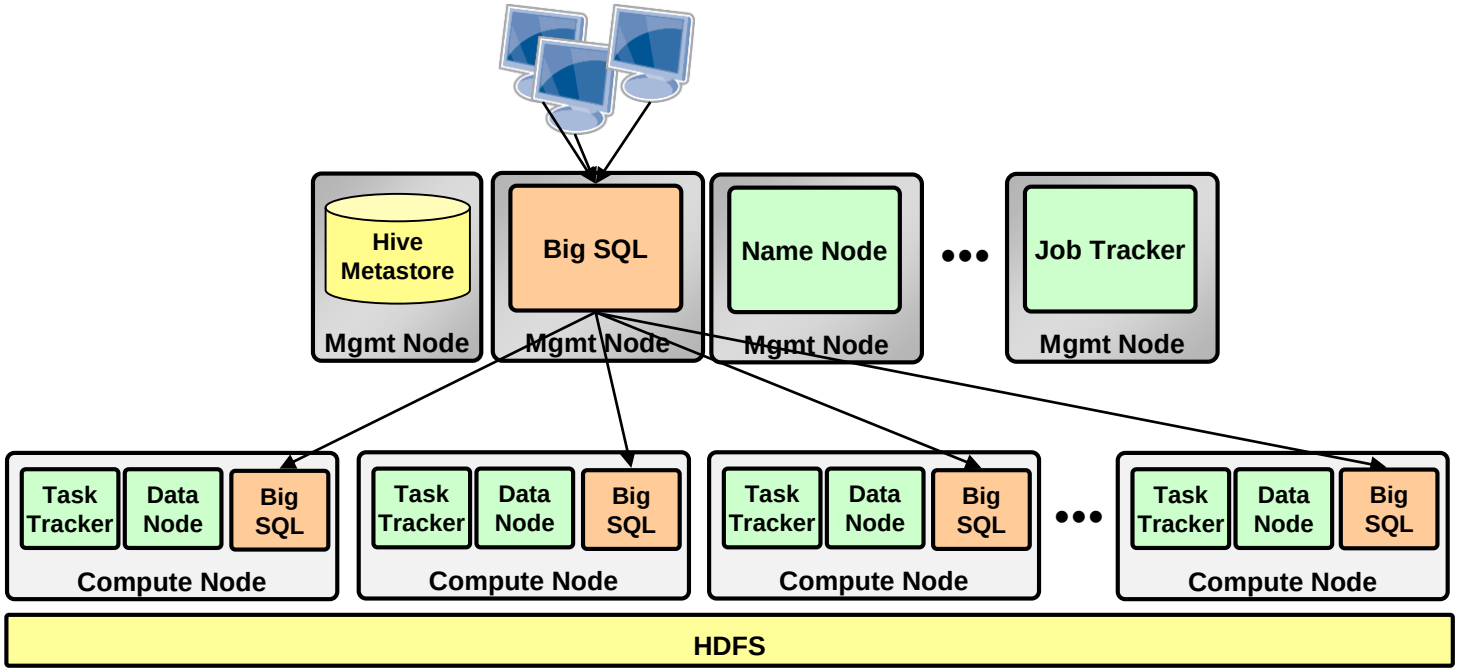
What is Big SQL 3.0?

- **SQL engine included with BigInsights**
- **Rich, ANSI compliant SQL support**
- **High performance access to Hadoop data**
  - Various storage formats supported (no IBM proprietary format required)
- **Integrates with RDBMSs via LOAD, query federation**
- **Big SQL 3.0 co-exists with older Big SQL 1.0**
  - Big SQL 1.0 is based on MapReduce



Big SQL 3.0 – Architecture

- **Big SQL head node**
  - Listens to the JDBC/ODBC connections
  - Compiles and optimizes the query
  - Coordinates the execution of the query . . . . Analogous to Job Tracker for Big SQL
- **Big SQL worker process resides on one or more compute nodes**
- **Compute nodes stream data between each other as needed**
- **Compute nodes can spill large data sets to local disk if needed**
  - Allows Big SQL to work with data sets larger than available memory



Architecture notes

- **Big SQL 3.0 does *not* own the data**
  - The traditional RDBMs storage layer has been replaced with data residing on HDFS
  - Therefore, no data caching (except temporary data) and no indexes
  - Data can be in many different formats and accessible by other Hadoop components

- **Big SQL still gains many great features from the RDBMs world including the query optimizer, self tuning memory, and advanced workload management**

Big SQL 3.0 Query Optimizer

- **Big SQL query performance depends heavily on efficient plan selection by the query optimizer**
  - Statistics and heuristics driven query optimization
    - Up to date runtime statistics critical for the query optimizer to choose good plans
    - Use ANALYZE TABLE … to collect statistics
  - Leverages additional metadata such as PK-FK constraints, primary key constraints, and column nullability
  - Exhaustive query rewrite capabilities

- **Tools and metrics**
  - Highly detailed explain plans and query diagnostic tools

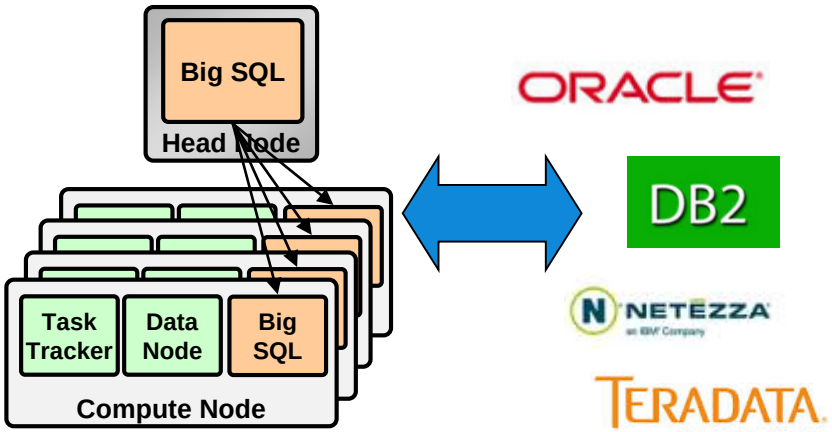
Big SQL 3.0 Query Processing Pushdown

- **Pushdown is important because it reduces the volume of data flowing from HDFS into Big SQL**
- **Pushdown moves processing down as close to the data as possible**
  - Projection pushdown – retrieve only necessary columns
  - Selection pushdown – push search criteria
- **Big SQL understands the capabilities of HDFS readers and storage formats involved**
  - As much as possible is pushed down
  - Residual processing done in the server
  - Optimizer costs queries based upon how much can be pushed down
- **Parquet format provides a good combination of efficient storage format and pushdown for Big SQL**

Big SQL 3.0 Query Federation

- **Data rarely lives in isolation**

- **Big SQL transparently queries heterogeneous systems**
  - Join Hadoop to other relational databases
  - Query optimizer understands capabilities of external system
    - Including available statistics
  - As much work as possible is pushed to each system to process

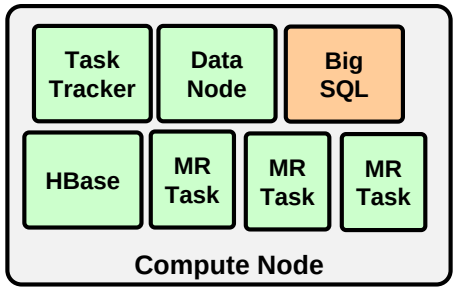


Data source	Supported versions
DB2 LUW	9.7, 9.8, 10.1, 10.5
Oracle	11g, 11gR1, 11gR2
Teradata	12,13
Netezza	4.6, 5.0, 6.0, 7.2

Hadoop cluster resource management

- **Big SQL 3.0 doesn't run in isolation**

- **Nodes tend to be shared with a variety of Hadoop services**
  - JobTracker, TaskTracker, and MapReduce tasks
  - HDFS Namenode and Data nodes
  - HBase region servers
  - etc…

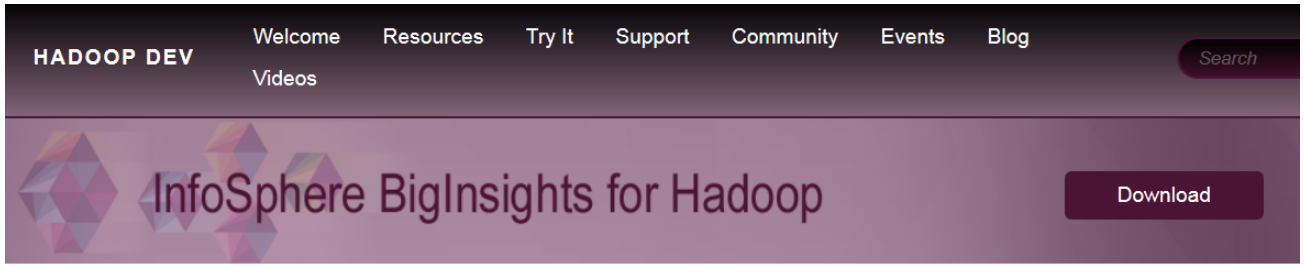


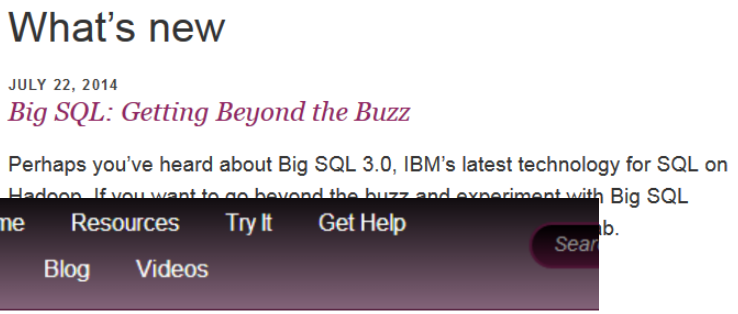
- **Big SQL can be constrained to limit its footprint on the cluster**
  - % of CPU utilization
  - % of memory utilization

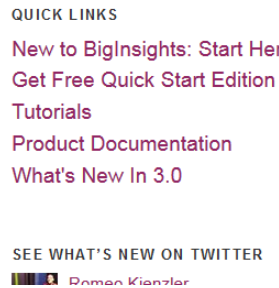
Get started with Big SQL: External resources

- **Hadoop Dev: links to videos, white paper, lab, ...**

https://developer.ibm.com/hadoop/







Questions?

John Poelman
BigInsights performance team, IBM Silicon Valley Lab (San Jose)
poelman@us.ibm.com