BigInsights 4.2 and BigSQL Overview

Les King
Director, Big Data, Analytics, Database & Cloud Data Services Solutions
November, 2016
lking@ca.ibm.com
calinkedin.com/pub/les-king/10/a68/426
Les King

Director, Big Data, Analytics, Database and Cloud Data Services Solutions
Analytics Division, IBM

Professor, Data Warehousing and DB2, Seneca College

lkling@ca.ibm.com
c.ca.linkedin.com/pub/les-king/10/a68/426

Professional Highlights
• 23 years of Information Management, Database and Analytics
• Technical sales (current)
• Technical customer support
• Software development teams
• Product management
• Taught mathematics at University of Toronto
• Teaching data warehousing, big data and DB2 at Seneca College

Personal Highlights
• English / Irish background
• Sports: squash, down hill skiing
• Certified Advanced Open Water diver
• Two sons: Philip and Richard
BigInsights 4.2 Highlights
IBM Open Platform and BigInsights v4.2
1H 2016 Roadmap Details

### Business Benefits
- Provide an [open and flexible](#) platform:
  - With enhanced tooling that enables data scientists to extract meaning and insight from large and often complex sets of data (Titan, Text Analytics on Spark)
  - With the resiliency and security platforms clients expect for their mission critical data and applications (Object store, Ranger)
  - That complies and is certified with the Open Data Platform Standards

- Shorten the [time to value](#) by reducing the complexity to access the data using SQL (Phoenix, Big SQL enhancements)

### Capabilities
- **Titan Inclusion**
  - Easily, map, and query relationships using Titan’s graph database. Graphs are offer a better way to represent complex relationships like social networks.

- **Text Analytics on Spark**
  - Derive value from text files by running the Text Analytics Information Extraction Engine (AQL) natively within Spark.

- **Ranger Inclusion**
  - Centralized security platform for managing authorization, access control, auditing, and data protection for data stored in Hadoop

- **Object Store Integration** (Cloud)
  - Object stores are particularly useful for users who need to store a large number of relatively smaller data objects

- **ODPi Certification**
  - Certify based on the ODPi requirements and standards

- **Phoenix Inclusion**
  - Eases access to Hbase with a SQL interface & allowing inputs, outputs using standard JDBC APIs instead of HBase’s Java client APIs.

- **Big SQL Enhancements**
  - 40% Performance gains for decision-support queries
  - Improve performance & usability of statistics via Auto Analyze
  - Enable Hive Impersonation – options to pass-through user credentials
## IBM Currency on Open Source Components

<table>
<thead>
<tr>
<th>Functional Areas</th>
<th>Component Name</th>
<th>Version 4.2</th>
<th>HDP 2.4</th>
<th>Cloudera 5.6</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Acquisition</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Flume</td>
<td>1.6.0</td>
<td>1.5.2</td>
<td>1.6.0</td>
</tr>
<tr>
<td></td>
<td>Kafka</td>
<td>0.9.0.0</td>
<td>0.9.0</td>
<td>0.9.0</td>
</tr>
<tr>
<td></td>
<td>Sqoop</td>
<td>1.4.6</td>
<td>1.4.6</td>
<td>1.4.6</td>
</tr>
<tr>
<td><strong>Security</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Knox</td>
<td>0.7.0</td>
<td>0.6.0</td>
<td>Proprietary</td>
</tr>
<tr>
<td></td>
<td>Ranger</td>
<td>0.5.0</td>
<td>0.5.0</td>
<td>Sentry</td>
</tr>
<tr>
<td><strong>Search</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Solr</td>
<td>5.4.1</td>
<td>5.2.1</td>
<td>4.10.3</td>
</tr>
<tr>
<td><strong>Format</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Parquet</td>
<td>2.2.0</td>
<td>2.2.0</td>
<td>2.1.0</td>
</tr>
<tr>
<td></td>
<td>Avro</td>
<td>1.7.7</td>
<td>1.7.7</td>
<td>1.7.6</td>
</tr>
<tr>
<td><strong>No SQL</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>HBase</td>
<td>1.2.0</td>
<td>1.1.2</td>
<td>1.0.0</td>
</tr>
<tr>
<td><strong>SQL</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hive</td>
<td>1.2.1</td>
<td>1.2.1</td>
<td>1.1.0</td>
</tr>
<tr>
<td></td>
<td>Spark</td>
<td>1.6.0</td>
<td>1.6.0</td>
<td>1.5.0</td>
</tr>
<tr>
<td><strong>Batch /Script</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Spark</td>
<td>1.6.0</td>
<td>1.6.0</td>
<td>1.5.0</td>
</tr>
<tr>
<td></td>
<td>Pig</td>
<td>0.15.0</td>
<td>0.15.0</td>
<td>0.12.0</td>
</tr>
<tr>
<td></td>
<td>MapReduce &amp; Yarn</td>
<td>2.7.2</td>
<td>2.7.1</td>
<td>2.6.0</td>
</tr>
<tr>
<td><strong>Scheduling</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Oozie</td>
<td>4.2.0</td>
<td>4.2.0</td>
<td>4.1.0</td>
</tr>
<tr>
<td></td>
<td>Slider</td>
<td>0.90.2</td>
<td>0.80.0</td>
<td></td>
</tr>
<tr>
<td><strong>Machine Learning</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Spark - MLlib</td>
<td>1.6.0</td>
<td>1.6.0</td>
<td>1.5.0</td>
</tr>
<tr>
<td><strong>Graph</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Spark – Graph X</td>
<td>1.6.0</td>
<td>1.6.0</td>
<td>1.5.0</td>
</tr>
<tr>
<td><strong>Mgmt/Monitoring</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ambari</td>
<td>2.2.0</td>
<td>2.2.1</td>
<td>Proprietary</td>
</tr>
<tr>
<td></td>
<td>Zookeeper</td>
<td>3.4.6</td>
<td>3.4.6</td>
<td>3.4.5</td>
</tr>
</tbody>
</table>

### 100% open source code
- Commitment to currency: days, not months
- Includes Spark
- Founding member of the Open Data Platform, [www.opendataplatform.com](http://www.opendataplatform.com)

### Free for production use
- Decoupled Apache Hadoop from IBM analytics and data science technologies
- Production support offering available

### Spark Leadership
- Latest currency on Spark (1.6 vs 1.5..x from competition)
- HortonWorks has no Spark committers
- Unique Spark support offerings, IBM for a clear differentiation

**NEW!** BigSQL support on HDP – has now GA’d!!
**IBM Open Platform & BigInsights v4.2**

**2016 Packaging Changes**

Objectives:
- Simple to understand and consume
- Adaptive to future market needs and growth
- Consistent cloud and on premise offerings

## Packaging Changes

<table>
<thead>
<tr>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enterprise Management Module Package removed</strong></td>
</tr>
<tr>
<td>- Available for customers from Systems Group</td>
</tr>
<tr>
<td><strong>Big R &amp; BigSheets deprecated</strong></td>
</tr>
<tr>
<td>- Functionality will not be removed until a suitable replacement is in place</td>
</tr>
<tr>
<td>- Functionality for Big R will be contributed to Spark R and SystemML</td>
</tr>
<tr>
<td><strong>BigInsights Premium replaces Data Analyst and Data Scientist packages</strong></td>
</tr>
<tr>
<td>- Simplified packaging includes all “value-adds” as BigInsights Premium</td>
</tr>
<tr>
<td><strong>Basic Plan (pay-as-you-go model) in Bluemix under BigInsights for Apache Hadoop</strong></td>
</tr>
<tr>
<td>- IOP clusters on an hourly model.</td>
</tr>
<tr>
<td>- Value adds to be included later in the year</td>
</tr>
<tr>
<td><strong>Migration Path for existing customers to new parts</strong></td>
</tr>
<tr>
<td><strong>Support for Linux on z System removed</strong></td>
</tr>
</tbody>
</table>
IBM Open Platform and BigInsights
2016 On-Premise Packaging Changes

IBM BigInsights Premium

- **Capability**
  - IBM Open Platform
  - Big SQL
  - Text Analytics
  - *IBM Streams*
  - *IBM Information Governance*

- **Support**
  - Standard IBM Support Options
  - Community Collaboration
  - Developer Assist for Spark

- **Services**
  - 2 week jump start program

High Availability & Disaster Recovery (Wandisco)

- **Analytics**
  - Watson Analytics

IBM Open Platform

- **Capability**
  - Apache Hadoop and Ecosystem
  - Apache Spark and Ecosystem

- **Support**
  - Community Collaboration
  - Elite Support Available

Data Preparation
BigIntegrate, BigQuality, Big Match

Data Security
Guardium

Analytics
SPSS, Analytic Server

Data Security
Guardium

Data Security
Guardium

High Availability & Disaster Recovery (Wandisco)

Future Integrated Offerings

Current Integrated Offerings
IBM Open Platform and BigInsights
2016 Cloud Packaging Changes

IB...
Example of text analytic tooling:
Graphical interface to describe structure of various textual formats – from log file data to natural language. Users do not need to know AQL.
WANdisco Fusion

Continuous Availability & Performance

- LAN-speed read/write access to the same data at every location
- Data is replicated as it’s ingested
- Delivers built-in continuous hot backup by default with automated failover and disaster recovery over LAN and WAN
  - DistCp solutions require scheduled backups outside of normal business hours due to resource contention.
  - DistCp solutions risk loss of data since last backup
- Install on top of live clusters without downtime
- Support for different distros and multiple versions of the same distro allows migration and upgrades across clusters and data centers without downtime.

100% Use of Compute Resources

- All clusters at all locations are fully readable and writeable
- No money wasted on read-only backup clusters
- Former backup clusters can be used to scale up deployments without spending more on hardware

Cluster Zoning

- Mix of hardware and storage to support a mix of applications requiring different SLAs
- Applications share data, not compute resources
- Isolates critical real-time applications from MapReduce and data ingest jobs
### IBM Open Platform and BigInsights

**Value-Add and Complementary Solutions**

<table>
<thead>
<tr>
<th>IBM BigInsights Premium</th>
</tr>
</thead>
<tbody>
<tr>
<td>- BigSQL – ANSI SQL and Spark Support</td>
</tr>
<tr>
<td>- Text Analytics - Information Extraction for unstructured data</td>
</tr>
<tr>
<td>- Premium Support</td>
</tr>
<tr>
<td>- Jump-Start Services</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IBM BigIntegrate</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Data Stage Engine on YARN</td>
</tr>
<tr>
<td>- Data Integration</td>
</tr>
<tr>
<td>- Data Transformation</td>
</tr>
<tr>
<td>- Self-service Integration</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IBM BigQuality</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Quality Stage Engine on YARN</td>
</tr>
<tr>
<td>- Data Cleansing</td>
</tr>
<tr>
<td>- Data Profiling</td>
</tr>
<tr>
<td>- Data Quality Monitoring</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IBM Big Match</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Big Match Engine on Spark</td>
</tr>
<tr>
<td>- Probabilistic Matching</td>
</tr>
<tr>
<td>- Unstructured Social Linking</td>
</tr>
<tr>
<td>- 360° Customer Analytics</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IBM SPSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Modeler with Spark Support</td>
</tr>
<tr>
<td>- Data Scientists Collaboration</td>
</tr>
<tr>
<td>- Analytic Server with Deployment Services for Hadoop and Spark</td>
</tr>
</tbody>
</table>

---

IBM Open Platform with Apache Hadoop and Spark
BigSQL 4.2 Overview
Common Analytics Engine

- Application compatibility: Write once, run anywhere
- Operational compatibility: Reuse operational and housekeeping procedures
- Licensing: Flexible entitlements for business agility & cost-optimization
- Integration: Common Fluid Query capabilities for query federation and data movement
- Standardized analytics: Common programming model for in-DB analytics
- Ecosystem: One ISV product certification for all platforms
Common Analytics Engine
will offer clients choice in selecting the best (combination of) data stores to satisfy their hybrid data warehouse solution needs

- Elastic capacity & pricing
- Fast time to deployment, load and go
- Built-in analytics

Managed Public Cloud Service

- On-prem or hosted
- Cost equivalent to Hadoop for structured data
- Utilizing existing Infrastructure
- Seamless scale-up and – out
- HW / OS-agnostic
- Combine compute / storage resources as needed

Software-defined

- Appliance
- FPGA-powered performance
- Built-in disaster recovery
- In-place expansion
- Elastic resource consumption

Appliance

- IBM or Client-managed
- Analytics on operational data or ODS
- Custom deployment on broad range of operating systems
- Data Discovery and exploration
- Highly available and disaster recovery needs

Custom Deployable Software

- IBM or Client-managed
- Data Lake or Day-0 archive
- Unstructured or hybrid data types
- Land data fast and apply instant analytics
- Data Discovery and exploration
- Data transformation

BigSQL on BigInsights (managed & on-prem)
SQL access for Hadoop: Why?

- Data warehouse modernization is a leading Hadoop use case
  - Off-load “cold” warehouse data into query-ready Hadoop platform
  - Explore / transform / analyze / aggregate social media data, log records, etc. and upload summary data to warehouse

- Limited availability of skills in MapReduce, Pig, etc.

- SQL opens the data to a much wider audience
  - Familiar, widely known syntax
  - Common catalog for identifying data and structure

---

Respondents with active big data efforts were asked which data sources they currently collect and analyze. Each data point was collected in a similar line/field; responses were on a scale from 1 to 5.07.

2012 Big Data @ Work Study surveying 1144 business and IT professionals in 95 countries
SQL-on-Hadoop landscape

- The SQL-on-Hadoop landscape changes constantly!

- Being relatively new to the SQL game, they’ve generally had to compromise in one or more of these areas:
  - Speed
  - Robust SQL
  - Enterprise features
  - Interoperability

- Big SQL based on decades of IBM R&D investment in relational technology that addresses these areas
What is Big SQL?

- **Comprehensive, standard SQL**
  - SELECT: joins, unions, aggregates, subqueries . . .
  - GRANT/REVOKE, INSERT ... INTO
  - SQL procedural logic (SQL PL)
  - Stored procs, user-defined functions
  - IBM data server JDBC and ODBC drivers

- **Optimization and performance**
  - IBM MPP engine (C++) replaces Java MapReduce layer
  - Continuous running daemons (no start up latency)
  - Message passing allow data to flow between nodes without persisting intermediate results
  - In-memory operations with ability to spill to disk (useful for aggregations, sorts that exceed available RAM)
  - Cost-based query optimization with 140+ rewrite rules

- **Various storage formats supported**
  - Text (delimited), Sequence, RCFile, ORC, Avro, Parquet
  - Data persisted in DFS, Hive, HBase
  - No IBM proprietary format required

- **Integration with RDBMSs via LOAD, query federation**
Big SQL architecture

- **Head (coordinator / management) 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
  - Optionally store user data in traditional RDBMS table (single node only). Useful for some reference data.

- **Big SQL worker processes reside on compute nodes (some or all)**
- **Worker nodes stream data between each other as needed**
- **Workers can spill large data sets to local disk if needed**
  - Allows Big SQL to work with data sets larger than available memory
Invocation options

- Command-line interface: Java SQL Shell (J Sqsh)

- Web tooling (Data Server Manager)

- Tools that support IBM JDBC/ODBC driver
Creating a Big SQL table

- **Standard CREATE TABLE DDL with extensions**

  ```sql
  create hadoop table users
  (id int not null primary key,
   office_id int null,
   fname varchar(30) not null,
   lname varchar(30) not null)
  row format delimited
  fields terminated by '|
  stored as textfile;
  ```

  **Worth noting:**
  • “Hadoop” keyword creates table in DFS
  • Row format delimited and textfile formats are default
  • Constraints not enforced (but useful for query optimization)

  • Examples in these charts focus on DFS storage, both within or external to Hive warehouse. HBase examples provided separately
Creating a View

- Standard SQL syntax

```sql
create view my_users as
select fname, lname from biadmin.users
where id > 100;
```
Populating tables via LOAD

- Typically best runtime performance
- Load data from local or remote file system

```sql
load hadoop using file url
'sftp://myID:myPassword@myServer.ibm.com:22/install
dir/bigsql/samples/data/GOSALES_DW.GO_REGION_DIM.txt' with SOURCE PROPERTIES
('field.delimiter'='\t') INTO TABLE gosalesdw.GO_REGION_DIM overwrite;
```

- Loads data from RDBMS (DB2, Netezza, Teradata, Oracle, MS-SQL, Informix) via JDBC connection

```sql
load hadoop using jdbc connection url 'jdbc:db2://some.host.com:portNum/sampledb'
with parameters (user='myID', password='myPassword')
from table MEDIA columns (ID, NAME)
where 'CONTACTDATE < ''2012-02-01'''
into table media_db2table_jan overwrite
with load properties ('num.map.tasks' = 10);
```
Populating tables via INSERT

- INSERT INTO ... SELECT FROM ...
  - Parallel read and write operations

CREATE HADOOP TABLE IF NOT EXISTS big_sales_parquet
    ( product_key INT NOT NULL, product_name VARCHAR(150),
      Quantity INT, order_method_en VARCHAR(90) )
STORED AS parquetfile;

-- source tables do not need to be in Parquet format
insert into big_sales_parquet
SELECT sales.product_key, pnumb.product_name, sales.quantity, meth.order_method_en
FROM sls_sales_fact sales, sls_product_dim prod,sls_product_lookup pnumb,
  sls_order_method_dim meth
WHERE
 pnumb.product_language='EN'
AND sales.product_key=prod.product_key
AND prod.product_number=pnumb.product_number
AND meth.order_method_key=sales.order_method_key
and sales.quantity > 5500;

- INSERT INTO ... VALUES(...) ...
  - Not parallelized. 1 file per INSERT. Not recommended except for quick tests

CREATE HADOOP TABLE foo col1 int, col2 varchar(10));
INSERT INTO foo VALUES (1, 'hello');
CREATE . . . TABLE . . . AS SELECT . . .

- Create a Big SQL table based on contents of other table(s)

- Source tables can be in different file formats or use different underlying storage mechanisms

-- source tables in this example are external (just DFS files)
CREATE HADOOP TABLE IF NOT EXISTS sls_product_flat
  ( product_key INT NOT NULL
    , product_line_code INT NOT NULL
    , product_type_key INT NOT NULL
    , product_type_code INT NOT NULL
    , product_line_en VARCHAR(90)
    , product_line_de VARCHAR(90)
  )
as select product_key, d.product_line_code, product_type_key,
    product_type_code, product_line_en, product_line_de
  from extern.sls_product_dim d, extern.sls_product_line_lookup l
  where d.product_line_code = l.product_line_code;
SQL capability highlights

- Query operations
  - Projections, restrictions
  - UNION, INTERSECT, EXCEPT
  - Wide range of built-in functions (e.g. OLAP)

- Full support for subqueries
  - In SELECT, FROM, WHERE and HAVING clauses
  - Correlated and uncorrelated
  - Equality, non-equality subqueries
  - EXISTS, NOT EXISTS, IN, ANY, SOME, etc.

- All standard join operations
  - Standard and ANSI join syntax
  - Inner, outer, and full outer joins
  - Equality, non-equality, cross join support
  - Multi-value join

- Stored procedures, UDFs
  - DB2 compatible SQL procedural logic
  - Cursors, flow of control (if/then/else, error handling, …), etc.

```sql
SELECT
  s_name,
  count(*) AS numwait
FROM
  supplier,
  lineitem l1,
  orders,
  nation
WHERE
  s_suppkey = l1.l_suppkey
  AND o_orderkey = l1.l_orderkey
  AND o_orderstatus = 'F'
  AND l1.l_receiptdate > l1.l_commitdate
  AND EXISTS (
    SELECT *
    FROM lineitem l2
    WHERE
      l2.l_orderkey = l1.l_orderkey
      AND l2.l_suppkey <> l1.l_suppkey
  )
  AND NOT EXISTS ( 
    SELECT *
    FROM lineitem l3
    WHERE
      l3.l_orderkey = l1.l_orderkey
      AND l3.l_suppkey <> l1.l_suppkey
      AND l3.l_receiptdate > l3.l_commitdate
  )
  AND s_nationkey = n_nationkey
  AND n_name = ':1'
GROUP BY s_name
ORDER BY numwait desc, s_name;
```
Power of standard SQL

- Everyone loves performance numbers, but that's not the whole story
  - How much work do you have to do to achieve those numbers?
- A portion of our internal performance numbers are based upon industry standard benchmarks
- Big SQL is capable of executing
  - All 22 TPC-H queries without modification
  - All 99 TPC-DS queries without modification
A word about . . . performance

- **TPC (Transaction Processing Performance Council)**
  - Formed August 1988
  - Widely recognized as most credible, vendor-independent SQL benchmarks
  - TPC-H and TPC-DS are the most relevant to SQL over Hadoop
    - R/W nature of workload not suitable for HDFS

- **Hadoop-DS benchmark: BigInsights, Hive, Cloudera**
  - Run by IBM & reviewed by TPC certified auditor
  - Based on TPC-DS. Key deviations
    - No data maintenance or persistence phases (not supported across all vendors)
  - Common set of queries across all solutions
    - Subset that all vendors can successfully execute at scale factor
    - Queries are not cherry picked
  - Most complete TPC-DS like benchmark executed so far
  - Analogous to porting a relational workload to SQL on Hadoop
Hadoop-DS benchmark topology

Big SQL 3.0 on BigInsights 3.0.0.1
- Parquet storage format

Impala 1.4 on CDH 5
- Parquet storage format

Hive 0.13 (Tez) on
HortonWorks HDP 2.1
- ORC storage format

3 identical 17 node clusters deployed

Hardware spec (per node):
- RHEL 6.4
- IBM x3650 M4 BD: Intel e5-2680@2.8GHz v2, 2 sockets, 10 cores each w/HT = 40 logical CPUs
- 128GB RAM, 1866MHz
- 10x3.5” HDD@2TB
- Dual port 10GbE

Workload:
- Hadoop-DS
- 10TB scale factor
  1) Load data
  2) Power run (single stream)
  3) Throughput run (4 streams)

# Big SQL - IBM Runs 100% of SQL Queries

TPC-DS is an industry standard analytic SQL query benchmark. These are the typical types of queries business analytic tools would generate. IBM has spent over 2 decades building an optimized SQL engine for MPP environments.

<table>
<thead>
<tr>
<th>Query</th>
<th>V1.4</th>
<th>V2.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Query 1</td>
<td>Query 1</td>
</tr>
<tr>
<td>2</td>
<td>Query 2</td>
<td>Query 2</td>
</tr>
<tr>
<td>3</td>
<td>Query 3</td>
<td>Query 3</td>
</tr>
<tr>
<td>4</td>
<td>Query 4</td>
<td>Query 4</td>
</tr>
<tr>
<td>5</td>
<td>Query 5</td>
<td>Query 5</td>
</tr>
<tr>
<td>6</td>
<td>Query 6</td>
<td>Query 6</td>
</tr>
<tr>
<td>7</td>
<td>Query 7</td>
<td>Query 7</td>
</tr>
<tr>
<td>8</td>
<td>Query 8</td>
<td>Query 8</td>
</tr>
<tr>
<td>9</td>
<td>Query 9</td>
<td>Query 9</td>
</tr>
<tr>
<td>10</td>
<td>Query 10</td>
<td>Query 10</td>
</tr>
<tr>
<td>11</td>
<td>Query 11</td>
<td>Query 11</td>
</tr>
<tr>
<td>12</td>
<td>Query 12</td>
<td>Query 12</td>
</tr>
<tr>
<td>13</td>
<td>Query 13</td>
<td>Query 13</td>
</tr>
<tr>
<td>14</td>
<td>Query 14</td>
<td>Query 14</td>
</tr>
<tr>
<td>15</td>
<td>Query 15</td>
<td>Query 15</td>
</tr>
<tr>
<td>16</td>
<td>Query 16</td>
<td>Query 16</td>
</tr>
<tr>
<td>17</td>
<td>Query 17</td>
<td>Query 17</td>
</tr>
<tr>
<td>18</td>
<td>Query 18</td>
<td>Query 18</td>
</tr>
<tr>
<td>19</td>
<td>Query 19</td>
<td>Query 19</td>
</tr>
<tr>
<td>20</td>
<td>Query 20</td>
<td>Query 20</td>
</tr>
<tr>
<td>21</td>
<td>Query 21</td>
<td>Query 21</td>
</tr>
<tr>
<td>22</td>
<td>Query 22</td>
<td>Query 22</td>
</tr>
<tr>
<td>23</td>
<td>Query 23</td>
<td>Query 23</td>
</tr>
<tr>
<td>24</td>
<td>Query 24</td>
<td>Query 24</td>
</tr>
<tr>
<td>25</td>
<td>Query 25</td>
<td>Query 25</td>
</tr>
<tr>
<td>26</td>
<td>Query 26</td>
<td>Query 26</td>
</tr>
<tr>
<td>27</td>
<td>Query 27</td>
<td>Query 27</td>
</tr>
<tr>
<td>28</td>
<td>Query 28</td>
<td>Query 28</td>
</tr>
<tr>
<td>29</td>
<td>Query 29</td>
<td>Query 29</td>
</tr>
<tr>
<td>30</td>
<td>Query 30</td>
<td>Query 30</td>
</tr>
<tr>
<td>31</td>
<td>Query 31</td>
<td>Query 31</td>
</tr>
<tr>
<td>32</td>
<td>Query 32</td>
<td>Query 32</td>
</tr>
<tr>
<td>33</td>
<td>Query 33</td>
<td>Query 33</td>
</tr>
<tr>
<td>34</td>
<td>Query 34</td>
<td>Query 34</td>
</tr>
<tr>
<td>35</td>
<td>Query 35</td>
<td>Query 35</td>
</tr>
<tr>
<td>36</td>
<td>Query 36</td>
<td>Query 36</td>
</tr>
<tr>
<td>37</td>
<td>Query 37</td>
<td>Query 37</td>
</tr>
<tr>
<td>38</td>
<td>Query 38</td>
<td>Query 38</td>
</tr>
<tr>
<td>39</td>
<td>Query 39</td>
<td>Query 39</td>
</tr>
<tr>
<td>40</td>
<td>Query 40</td>
<td>Query 40</td>
</tr>
<tr>
<td>41</td>
<td>Query 41</td>
<td>Query 41</td>
</tr>
<tr>
<td>42</td>
<td>Query 42</td>
<td>Query 42</td>
</tr>
<tr>
<td>43</td>
<td>Query 43</td>
<td>Query 43</td>
</tr>
<tr>
<td>44</td>
<td>Query 44</td>
<td>Query 44</td>
</tr>
<tr>
<td>45</td>
<td>Query 45</td>
<td>Query 45</td>
</tr>
<tr>
<td>46</td>
<td>Query 46</td>
<td>Query 46</td>
</tr>
<tr>
<td>47</td>
<td>Query 47</td>
<td>Query 47</td>
</tr>
<tr>
<td>48</td>
<td>Query 48</td>
<td>Query 48</td>
</tr>
<tr>
<td>49</td>
<td>Query 49</td>
<td>Query 49</td>
</tr>
<tr>
<td>50</td>
<td>Query 50</td>
<td>Query 50</td>
</tr>
<tr>
<td>51</td>
<td>Query 51</td>
<td>Query 51</td>
</tr>
<tr>
<td>52</td>
<td>Query 52</td>
<td>Query 52</td>
</tr>
<tr>
<td>53</td>
<td>Query 53</td>
<td>Query 53</td>
</tr>
<tr>
<td>54</td>
<td>Query 54</td>
<td>Query 54</td>
</tr>
<tr>
<td>55</td>
<td>Query 55</td>
<td>Query 55</td>
</tr>
<tr>
<td>56</td>
<td>Query 56</td>
<td>Query 56</td>
</tr>
<tr>
<td>57</td>
<td>Query 57</td>
<td>Query 57</td>
</tr>
<tr>
<td>58</td>
<td>Query 58</td>
<td>Query 58</td>
</tr>
<tr>
<td>59</td>
<td>Query 59</td>
<td>Query 59</td>
</tr>
<tr>
<td>60</td>
<td>Query 60</td>
<td>Query 60</td>
</tr>
<tr>
<td>61</td>
<td>Query 61</td>
<td>Query 61</td>
</tr>
<tr>
<td>62</td>
<td>Query 62</td>
<td>Query 62</td>
</tr>
<tr>
<td>63</td>
<td>Query 63</td>
<td>Query 63</td>
</tr>
<tr>
<td>64</td>
<td>Query 64</td>
<td>Query 64</td>
</tr>
<tr>
<td>65</td>
<td>Query 65</td>
<td>Query 65</td>
</tr>
<tr>
<td>66</td>
<td>Query 66</td>
<td>Query 66</td>
</tr>
<tr>
<td>67</td>
<td>Query 67</td>
<td>Query 67</td>
</tr>
<tr>
<td>68</td>
<td>Query 68</td>
<td>Query 68</td>
</tr>
<tr>
<td>69</td>
<td>Query 69</td>
<td>Query 69</td>
</tr>
<tr>
<td>70</td>
<td>Query 70</td>
<td>Query 70</td>
</tr>
<tr>
<td>71</td>
<td>Query 71</td>
<td>Query 71</td>
</tr>
<tr>
<td>72</td>
<td>Query 72</td>
<td>Query 72</td>
</tr>
<tr>
<td>73</td>
<td>Query 73</td>
<td>Query 73</td>
</tr>
<tr>
<td>74</td>
<td>Query 74</td>
<td>Query 74</td>
</tr>
<tr>
<td>75</td>
<td>Query 75</td>
<td>Query 75</td>
</tr>
<tr>
<td>76</td>
<td>Query 76</td>
<td>Query 76</td>
</tr>
<tr>
<td>77</td>
<td>Query 77</td>
<td>Query 77</td>
</tr>
<tr>
<td>78</td>
<td>Query 78</td>
<td>Query 78</td>
</tr>
<tr>
<td>79</td>
<td>Query 79</td>
<td>Query 79</td>
</tr>
<tr>
<td>80</td>
<td>Query 80</td>
<td>Query 80</td>
</tr>
<tr>
<td>81</td>
<td>Query 81</td>
<td>Query 81</td>
</tr>
<tr>
<td>82</td>
<td>Query 82</td>
<td>Query 82</td>
</tr>
<tr>
<td>83</td>
<td>Query 83</td>
<td>Query 83</td>
</tr>
<tr>
<td>84</td>
<td>Query 84</td>
<td>Query 84</td>
</tr>
<tr>
<td>85</td>
<td>Query 85</td>
<td>Query 85</td>
</tr>
<tr>
<td>86</td>
<td>Query 86</td>
<td>Query 86</td>
</tr>
<tr>
<td>87</td>
<td>Query 87</td>
<td>Query 87</td>
</tr>
<tr>
<td>88</td>
<td>Query 88</td>
<td>Query 88</td>
</tr>
<tr>
<td>89</td>
<td>Query 89</td>
<td>Query 89</td>
</tr>
<tr>
<td>90</td>
<td>Query 90</td>
<td>Query 90</td>
</tr>
<tr>
<td>91</td>
<td>Query 91</td>
<td>Query 91</td>
</tr>
<tr>
<td>92</td>
<td>Query 92</td>
<td>Query 92</td>
</tr>
<tr>
<td>93</td>
<td>Query 93</td>
<td>Query 93</td>
</tr>
<tr>
<td>94</td>
<td>Query 94</td>
<td>Query 94</td>
</tr>
<tr>
<td>95</td>
<td>Query 95</td>
<td>Query 95</td>
</tr>
<tr>
<td>96</td>
<td>Query 96</td>
<td>Query 96</td>
</tr>
<tr>
<td>97</td>
<td>Query 97</td>
<td>Query 97</td>
</tr>
<tr>
<td>98</td>
<td>Query 98</td>
<td>Query 98</td>
</tr>
<tr>
<td>99</td>
<td>Query 99</td>
<td>Query 99</td>
</tr>
</tbody>
</table>

**Regressions:**
- Query 4
- Query 18
- Query 23
- Query 77
- Query 85

>5% of analytic queries regressed from V1.4 to V2.1.

How will this impact your BA applications?
**Hadoop-DS benchmark single user performance 10TB**

**Big SQL is 3.6x faster than Impala and 5.4x faster than Hive 0.13**

for single query stream using 46 common queries

---

Based on IBM internal tests comparing BigInsights Big SQL, Cloudera Impala and Hortonworks Hive (current versions available as of 9/01/2014) running on identical hardware. The test workload was based on the latest revision of the TPC-DS benchmark specification at 10TB data size. Successful executions measure the ability to execute queries a) directly from the specification without modification, b) after simple modifications, c) after extensive query rewrites. All minor modifications are either permitted by the TPC-DS benchmark specification or are of a similar nature. All queries were reviewed and attested by a TPC certified auditor. Development effort measured time required by a skilled SQL developer familiar with each system to modify queries so they will execute correctly. Performance test measured scaled query throughput per hour of 4 concurrent users executing a common subset of 46 queries across all 3 systems at 10TB data size. Results may not be typical and will vary based on actual workload, configuration, applications, queries and other variables in a production environment.

Cloudera, the Cloudera logo, Cloudera Impala are trademarks of Cloudera. Hortonworks, the Hortonworks logo and other Hortonworks trademarks are trademarks of Hortonworks Inc. in the United States and other countries.
Hadoop-DS benchmark multi-user performance 10TB

With 4 streams, Big SQL is 2.1x faster than Impala and 8.5x faster than Hive 0.13

for 4 query streams using 46 common queries

Based on IBM internal tests comparing BigInsights Big SQL, Cloudera Impala and Hortonworks Hive (current versions available as of 9/01/2014) running on identical hardware. The test workload was based on the latest revision of the TPC-DS benchmark specification at 10TB data size. Successful executions measure the ability to execute queries a) directly from the specification without modification, b) after simple modifications, c) after extensive query rewrites. All minor modifications are either permitted by the TPC-DS benchmark specification or are of a similar nature. All queries were reviewed and attested by a TPC certified auditor. Development effort measured time required by a skilled SQL developer familiar with each system to modify queries so they will execute correctly. Performance test measured scaled query throughput per hour of 4 concurrent users executing a common subset of 46 queries across all 3 systems at 10TB data size. Results may not be typical and will vary based on actual workload, configuration, applications, queries and other variables in a production environment.

Cloudera, the Cloudera logo, Cloudera Impala are trademarks of Cloudera. Hortonworks, the Hortonworks logo and other Hortonworks trademarks are trademarks of Hortonworks Inc. in the United States and other countries.
Audited results

- Letters of attestation are available for both Hadoop-DS benchmarks at 10TB and 30TB scale
- InfoSizing, Transaction Processing Performance Council Certified Auditors verified both IBM results as well as results on Cloudera Impala and HortonWorks HIVE.
- These results are for a non-TPC benchmark. A subset of the TPC-DS Benchmark standard requirements was implemented.
A word about column masking

1) Create and grant access and roles *

CREATE ROLE MANAGER
CREATE ROLE EMPLOYEE

GRANT SELECT ON SAL_TBL TO USER socrates
GRANT SELECT ON SAL_TBL TO USER newton

GRANT ROLE MANAGER TO USER socrates
GRANT ROLE EMPLOYEE TO USER newton

2) Create permissions *

CREATE MASK SALARY_MASK ON SAL_TBL FOR COLUMN SALARY RETURN
CASE WHEN VERIFY_ROLE_FOR_USER(SESSION_USER,'MANAGER') = 1 THEN SALARY ELSE 0.00 END
ENABLE

3) Enable access control *

ALTER TABLE SAL_TBL ACTIVATE COLUMN ACCESS CONTROL

4a) Select as an EMPLOYEE

CONNECT TO TESTDB USER newton
SELECT "*" FROM SAL_TBL

EMP_NO  FIRST_NAME   SALARY
-------  ------------   -------
 1 Steve  250000
 2 Chris  200000
 3 Paula  100000

3 record(s) selected.

4b) Select as a MANAGER

CONNECT TO TESTDB USER socrates
SELECT "*" FROM SAL_TBL

EMP_NO  FIRST_NAME   SALARY
-------  ------------   -------
 1 Steve    0
 2 Chris    0
 3 Paula    0

3 record(s) selected.

* Note: Steps 1, 2, and 3 are done by a user with SECADM authority.
A word about ... row-based access control

1) Create and grant access and roles *

CREATE ROLE BRANCH_A_ROLE
GRANT ROLE BRANCH_A_ROLE TO USER newton
GRANT SELECT ON BRANCH_TBL TO USER newton

2) Create permissions *

CREATE PERMISSION BRANCH_A_ACCESS ON BRANCH_TBL
FOR ROWS WHERE(VERIFY_ROLE_FOR_USER(SESSION_USER,'BRANCH_A_ROLE') = 1
AND
BRANCH_TBL.BRANCH_NAME = 'Branch_A')
enforced for all access
enable

3) Enable access control *

ALTER TABLE BRANCH_TBL ACTIVATE ROW ACCESS CONTROL

4) Select as Branch_A user

CONNECT TO TESTDB USER newton
SELECT "*" FROM BRANCH_TBL

EMP_NO   FIRST_NAME   BRANCH_NAME
-------   -----------   -----------
 2 Chris    Branch_A
 3 Paula    Branch_A
 5 Pete     Branch_A
 8 Chrissie Branch_A

4 record(s) selected.

* Note: Steps 1, 2, and 3 are done by a user with SECADM authority.
A word about ... data types

- Variety of primitives supported
  - TINYINT, INT, DECIMAL(p,s), FLOAT, REAL, CHAR, VARCHAR, TIMESTAMP, DATE, VARBINARY, BINARY, ...
  - Maximum 32K

- Complex types
  - ARRAY: ordered collection of elements of same type
  - Associative ARRAY (equivalent to Hive MAP type): unordered collection of key/value pairs. Keys must be primitive types (consistent with Hive)
  - ROW (equivalent to Hive STRUCT type): collection of elements of different types
  - Nesting supported for ARRAY of ROW (STRUCT) types
  - Query predicates for ARRAY or ROW columns must specify elements of a primitive type

CREATE HADOOP TABLE mytable (id INT, info INT ARRAY[10]);
SELECT * FROM mytable WHERE info[8]=12;
A word about . . . SerDes

- **Custom serializers / deserializers (SerDes)**
  - Read / write complex or “unusual” data formats (e.g., JSON)
  - Commonly used by Hadoop community
  - Developed by user or available publicly

- **Users add SerDes to appropriate directory and reference SerDe when creating table**

- **Example**

```sql
-- Create table for JSON data using open source hive-json-serde-0.2.jar SerDe
-- Location clause points to DFS dir containing JSON data
-- External clause means DFS dir & data won’t be drop after DROP TABLE command
create external hadoop table socialmedia-json (Country varchar(20), FeedInfo varchar(300), . . . )
row format serde 'org.apache.hadoop.hive.contrib.serde2.JsonSerde'
location '</hdfs_path>/myJSON';

select * from socialmedia-json;
```
A word about . . . query federation

- Data rarely lives in isolation

- **Big SQL transparently queries heterogeneous systems**
  - Join Hadoop to RDBMSs
  - Query optimizer understands capabilities of external system
    - Including available statistics
  - As much work as possible is pushed to each system to process
A word about ... resource management

- Big SQL doesn't run in isolation

- Nodes tend to be shared with a variety of Hadoop services
  - HBase region servers
  - MapReduce jobs
  - ...

- Big SQL can be constrained to limit its footprint on the cluster
  - CPU utilization
  - Memory utilization

- Resources are automatically adjusted based upon workload
  - Always fitting within constraints
  - Self-tuning memory manager that re-distributes resources across components dynamically
  - default WLM concurrency control for heavy queries
A word about... high availability

- **Big SQL master node high availability**
  - Scheduler automatically restarted upon failure
  - Catalog changes replicated to warm standby instance
  - Warm standby automatically takes over if the primary fails

- **Worker node failure leads to black listing / auto resubmission**
A word about . . . application portability

- **Big SQL** adopts **IBM's standard Data Server Client Drivers**
  - Robust, standards compliant ODBC, JDBC, and .NET drivers
  - Same driver used for DB2 LUW, DB2/z and Informix

- **Putting the story together....**
  - Big SQL shares a common SQL dialect with DB2
  - Big SQL shares the same client drivers with DB2

- Data warehouse augmentation just got easier

- Integration with popular third party tools just got easier
A word about . . . HBase support

- **Big SQL with HBase - basic operations**
  - Create tables and views
  - LOAD / INSERT data
  - Query data with full SQL breadth
  - . . .

- **HBase-specific design points**
  - Column mapping
  - Dense / composite columns
  - FORCE KEY UNIQUE option
  - Salting option
  - Secondary indexes
  - . . .

- **Details covered in separate presentation**
A word about . . . Hive compatibility

- Big SQL can directly define and execute Hive User Defined Functions (UDF’s)

```sql
CREATE FUNCTION MyHiveUDF (ARG1 VARCHAR(20), ARG2 INT)
RETURNS INT
SPECIFIC MYHIVEUDF
PARAMETER STYLE HIVE
EXTERNAL NAME ‘com.myco.MyHiveUDF’
DETERMINISTIC
LANGUAGE JAVA
```

- Native implementations of key Hive built-in functions
  - get_json_object() – Installable native UDF
  - json_tuple() – Installable native UDF
  - from_utc_timestamp() – Provided as a Big SQL built-in
  - to_utc_timestamp() – Provided as a Big SQL built-in
A word about . . . YARN Integration

- Hadoop 2.0 introduced YARN for comprehensive resource management
  - Dynamically manages cluster resources
  - Works across heterogeneous cluster services
  - Allocates CPU and memory resources based upon a given application’s current needs

- Big SQL integrates with YARN via the Slider project
  - YARN chooses suitable hosts for Big SQL worker nodes
  - Big SQL resources are accounted for by YARN
  - Size of the Big SQL cluster may dynamically grow or shrink as needed
  - Configured by user (not by installation default)
SparkSQL

- Provide for relational queries expressed in SQL, HiveQL and Scala
- Seamlessly mix SQL queries with Spark programs
- DataFrame/Dataset provide a single interface for efficiently working with structured data including Apache Hive, Parquet and JSON files
- Leverages Hive frontend and metastore
  - Compatibility with Hive data, queries, and UDFs
  - HiveQL limitations may apply
  - Not ANSI SQL compliant
  - Little to no query rewrite optimization, automatic memory management or sophisticated workload management
- Graduated from alpha status with Spark 1.3
  - DataFrames API marked as experimental
- Standard connectivity through JDBC/ODBC
Big SQL and Spark

- **What is Apache Spark?**
  - Fast, general-purpose engine for working with Big Data
  - Part of IBM Open Platform for Apache Hadoop
  - Popular built-in libraries for machine learning, query, streaming, etc.

- **Data in Big SQL accessible through Spark**
  - Big SQL meta data in HCatalog
  - Big SQL tables use common Hadoop file formats
  - Spark SQL provides access to structured data

- **Approach (BigInsights 4.x)**
  - From Big SQL: CREATE HADOOP TABLE . . . in Hive warehouse or over DFS directory
  - From Spark:
    - Create HiveContext
    - Issue queries (Hive syntax) directly against Big SQL tables
    - Invoke Spark transformations and actions as desired, including those specific to other Spark libraries (e.g., MLlib)
Demo – SPSS, Cognos, DB2 BLU, BigInsights, BigSQL

https://www.youtube.com/watch?v=zIRWt4XdTcM

More Information

https://bigdatauniversity.com/
BigInsights 4.2 and BigSQL Overview

Les King
Director, Big Data, Analytics, Database & Cloud Data Services Solutions
November, 2016
lking@ca.ibm.com
c.ca.linkedin.com/pub/les-king/10/a68/426