

cloudera®



Big Data Analytics with Tableau and Apache Impala

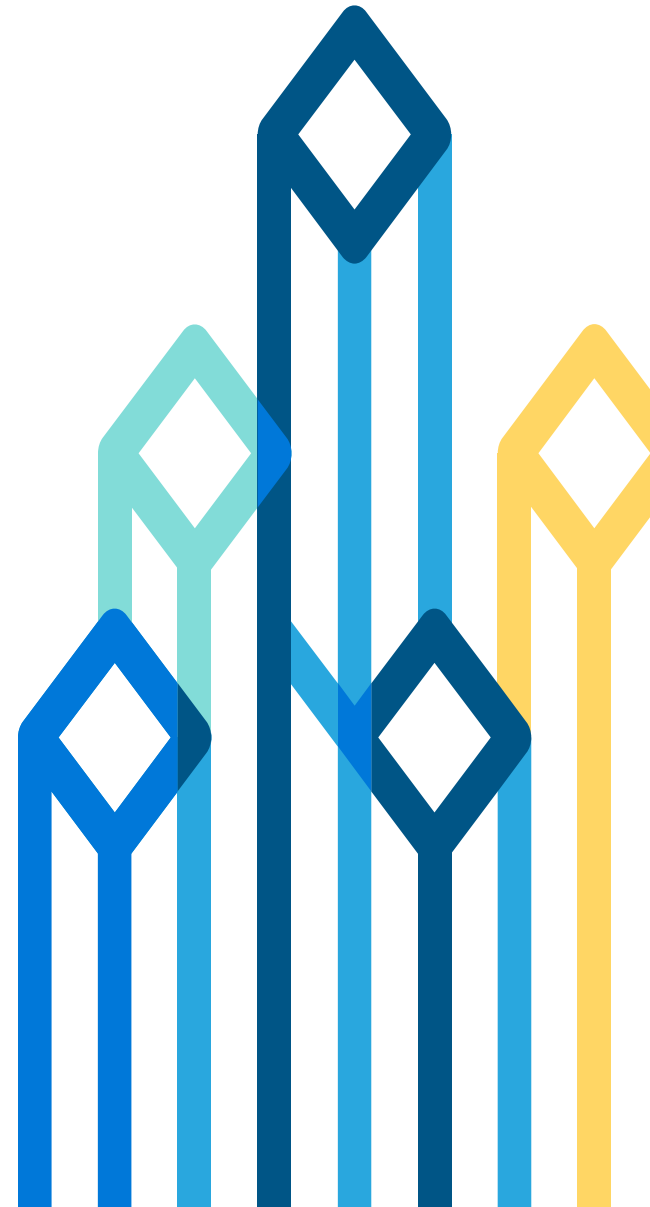
The Leader for Analytic SQL on Hadoop

Scott Armstrong, Director, Cloudera

Murali Krishna, CTO Colaberry

Madhu Ganta, Big Data Architect, Cloudera

Jatin Shah, Sr. Sales Engineer, Cloudera

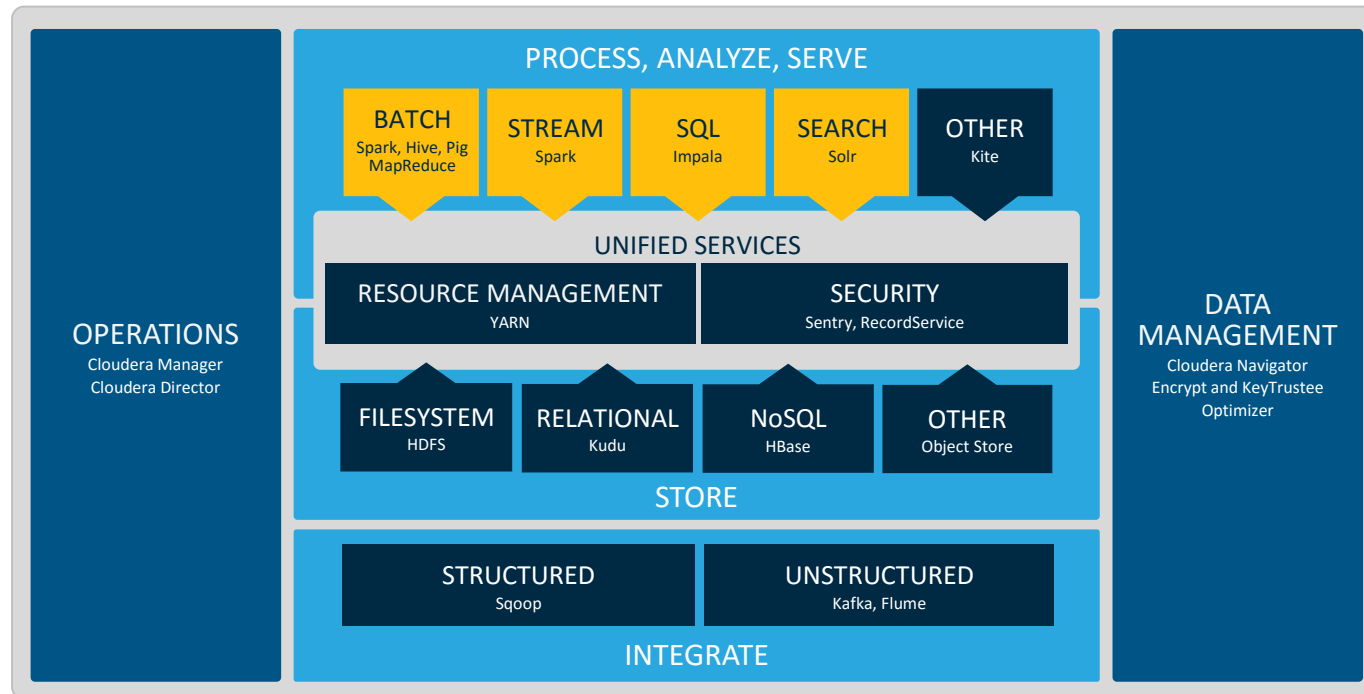


Solution: Hadoop & Ecosystem

- Hadoop is a software platform for storing, processing, and analyzing “big data”
 - Distributed
 - Scalable
 - Fault-tolerant
 - Open source



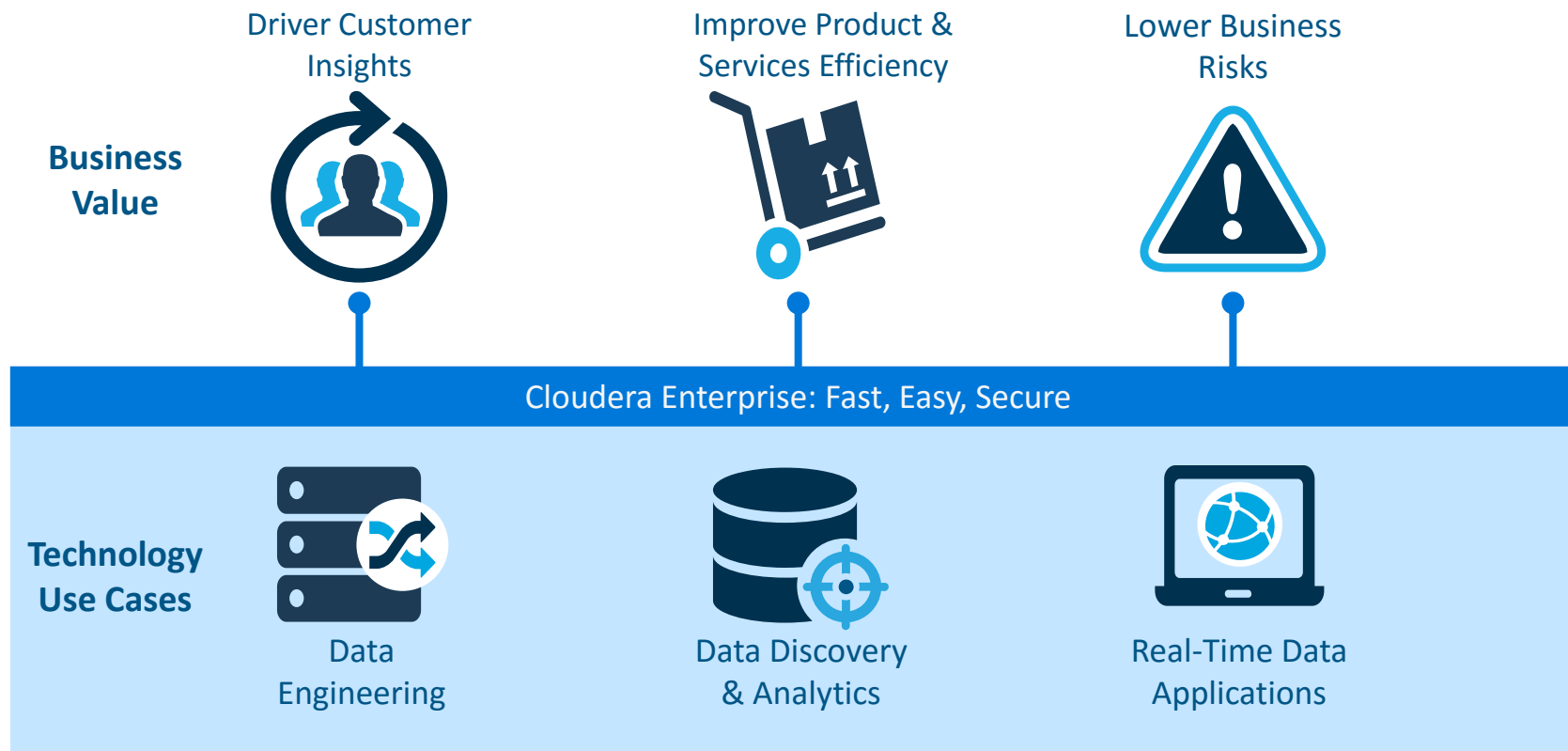
One Platform, Many Workloads



Batch, Interactive, and Real-Time.

- End-to-end analytic workflows
- Access more data
- Work with data in new ways
- Enable new users

One Platform. Many Applications.



Apache Impala

What is Impala ?

- Apache Impala is a high-performance, low-latency SQL queries engine on data stored on the popular Apache Hadoop file system (HDFS) .
- Impala integrates with the existing CDH ecosystem & shares same hive meta-store

Apache Impala: Open Source & Open Standard

- 1 > 1 MM downloads since GA
- 2 Majority adoption across Cloudera customers

- 3 Certification across key application partners:



- 4 De facto standard with multi-vendor support:



Database Cost per TB

MPP Database

- Massively Parallel Processing
- Specialty Database Appliances
- \$30-\$50K per TB
- Expensive

Apache Impala on Hadoop

- Massively Parallel Processing
- Really all Commodity Hardware
- \$2K per TB
- At least 10x Cheaper

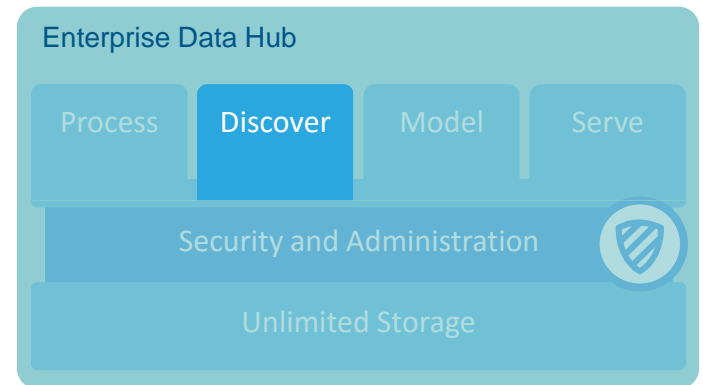
What is it good at?

Good at

- Schema on read – changing schemas
- Columnar Compression
- Ugly Wide tables
- Big data that doesn't fit anywhere else
- Interactive analysis at Scale
- Performance at 5-10% of the cost
- Data Consolidation – Data Lake, Data Hub

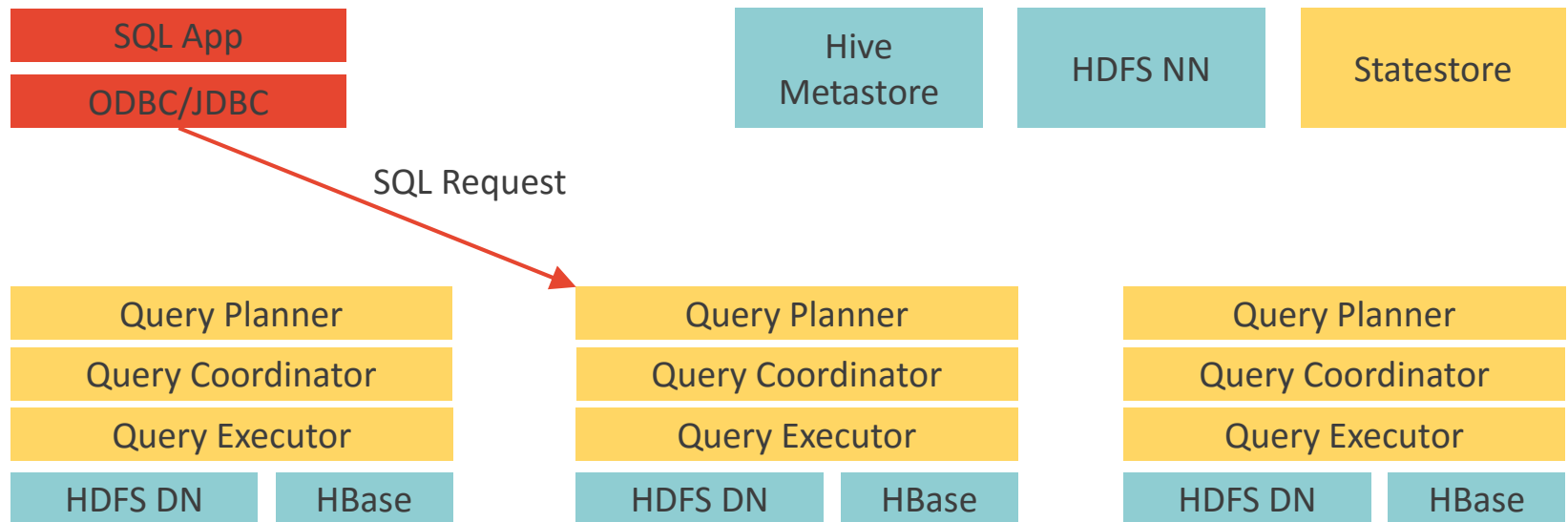
Successful Use Cases

- Interactive BI/analytics on “big data”
- Data discovery – Understanding your data
- Exploratory analytics – Top N , Percentiles etc.
- Queryable operational data store
- Active Archive



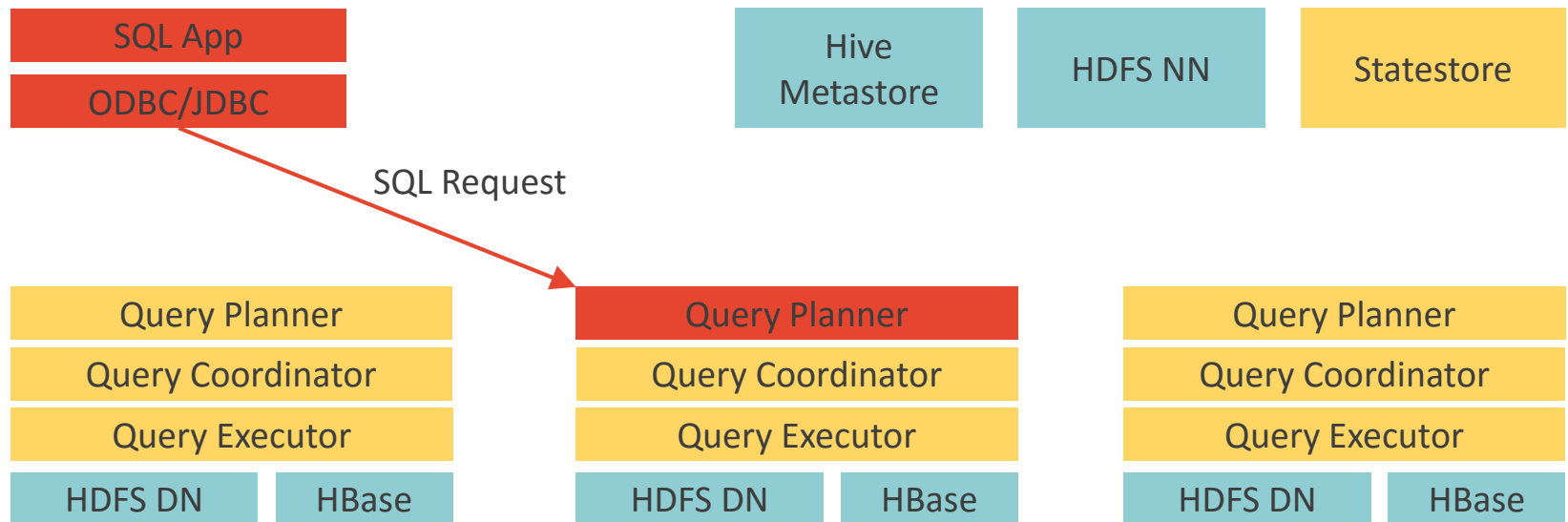
Impala Architecture

- MPP query engine built natively into Hadoop



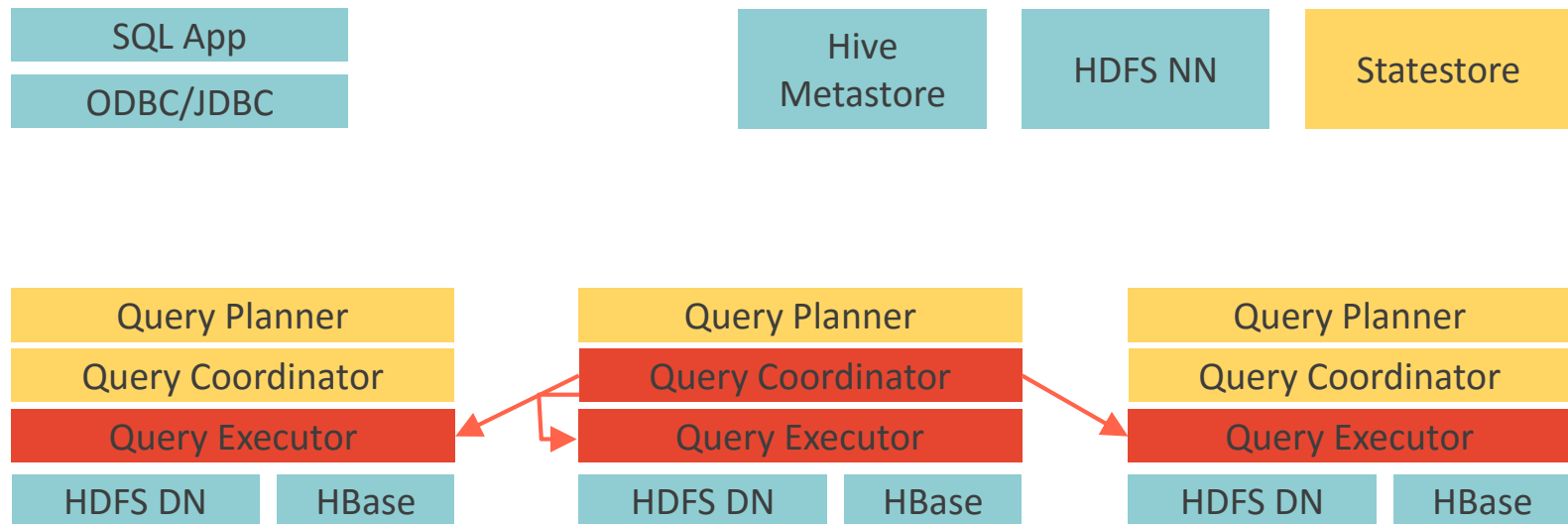
Impala Architecture: Query Execution

- Request arrives via ODBC/JDBC/Hue GUI/Shell



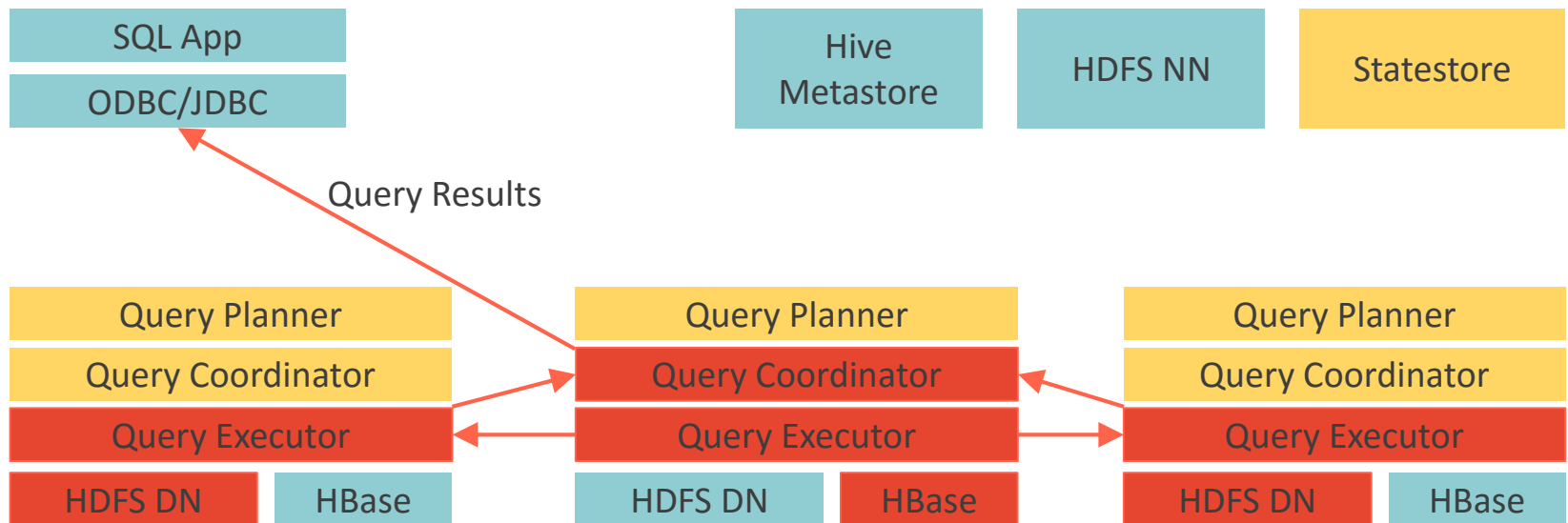
Impala Architecture: Query Execution

- Planner turns request into collections of plan fragments
- Coordinator initiates execution on impalad's local to data

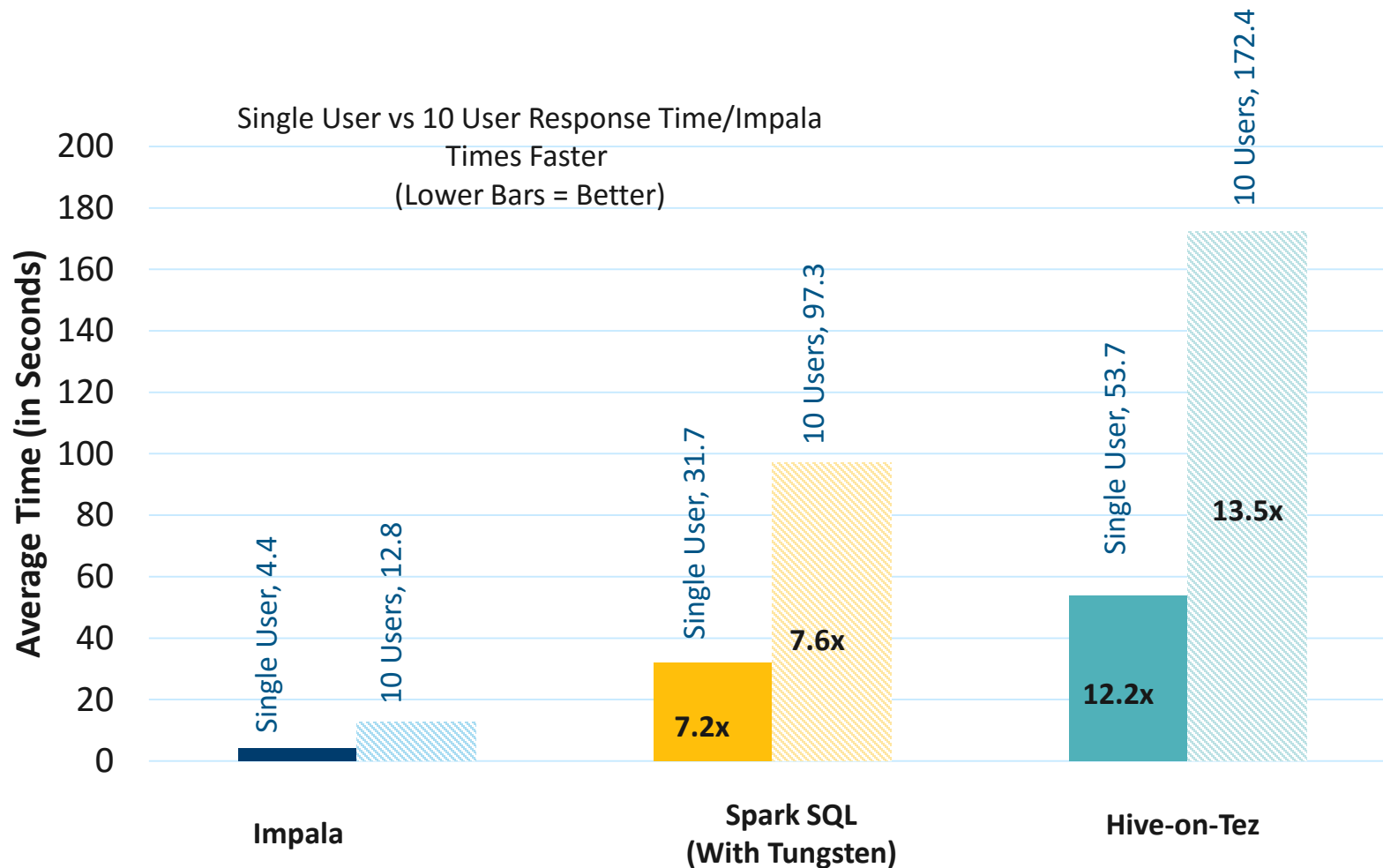


Impala Architecture: Query Execution

- Intermediate results are streamed between impalad's
- Query results are streamed back to client



Performance



Hadoop & Big Data

Our Customers

FAQs

Blog

Accumulo (1)

Using Impala at Scale at Allstate

by Justin Kestelyn (@kestelyn) | May 15, 2014 | 3 comments

Our thanks to Don Drake (@dondrake), an independent technology consultant who is currently working as a Principal Big Data Consultant at Allstate Insurance, for the guest post below about his experiences with Impala.

It started with a simple request from one of the managers in my group at Allstate to put together a demo of Tableau connecting to **Cloudera Impala**. I had previously worked on Impala with a large dataset about a year ago while it was still in beta, and was curious to see how Impala had improved since then in features and stability.

In this post, I'll share my experiences with Impala during the process of building my demo. As you'll see below, I

- <http://blog.cloudera.com/blog/2014/05/using-impala-at-scale-at-allstate/>

Blog Summary

- Earned Premium Dataset – 800 columns, 1.1B Rows, 2.3TB
- 35 node cluster – scans text table in 3 minutes
- Add Partitioning (time, geo, org, line) and Parquet Format
- Runs like lightning – where'd my 2TB go? Now 100GB
- Can scan the whole table in 10 seconds
- Partitioned queries run in 1 or 2 seconds

```
DROP TABLE IF EXISTS default.datasetp_2010_2013;
CREATE EXTERNAL TABLE IF NOT EXISTS default.datasetp_2010_2013
(
  PROC_MONTH STRING, PROC_YEAR STRING, POLCT STRING, STATST STRING, [800+ column definit
  EEXP REAL, EPREM REAL, EXP REAL, PREM REAL
)
PARTITIONED BY (ACTYR STRING, GEOST STRING, ALINE STRING, COMPNY STRING)
STORED AS PARQUET
LOCATION '/user/drake/cdf_impala_part';
```

Tuning & Best Practices

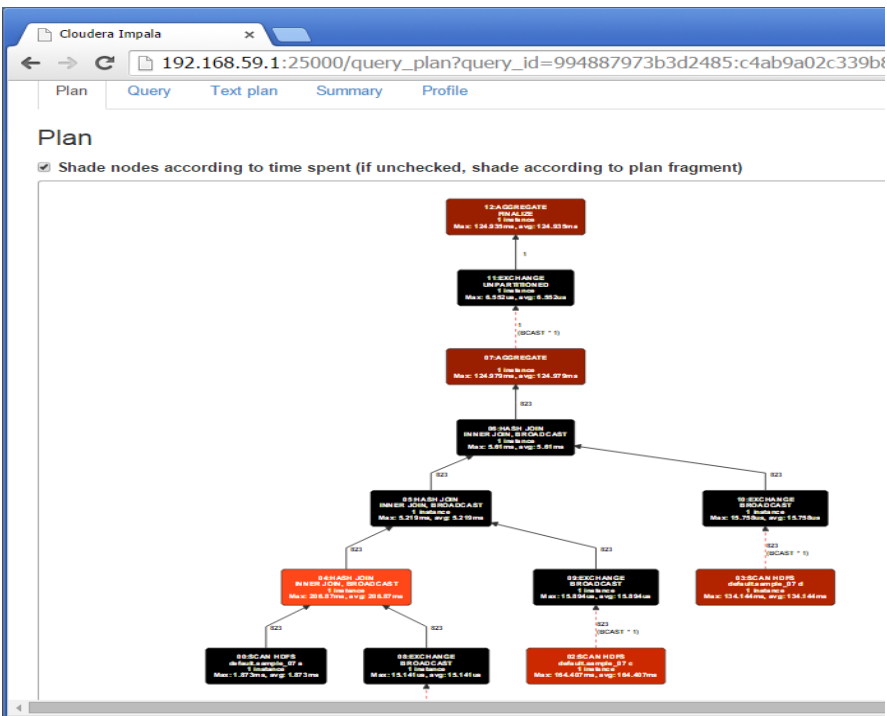
Partitioning

- Large tables can be partitioned
- Common partition field is date
- Partitions should be > 1GB
- Common where clause criteria

Why create a bunch of tables?

- 80% of data modeling is to save space or create performance
- Making Impala do a join is often a waste of time
- Parquet lets you build big ugly tables that work well
- Lazy data modeling advocate

Impala query analysis



Cloudera Impala

192.168.59.1:25000/query_summary?query_id=994887973b3d2485:c4ab9a02c339b881

impalad

/ /backends /catalog /hadoop-varz /logs /memz /metrics /queries /tcpz /sessions /threadz /varz

Query 994887973b3d2485:c4ab9a02c339b881

Status: OK

Plan Query Text plan Summary Profile

☒ Auto-refresh on Last updated: Wed Jul 08 2015 09:39:20 GMT-0400 (Eastern Daylight Time)

Exec Summary

Operator	#Hosts	Avg Time	Max Time	#Rows	Est. #Rows	Peak Mem	Est. Peak Mem	Detail
12:AGGREGATE	1	124.935ms	124.935ms	1	0	16.00 KB	-1.00 B	FINALIZE
11:EXCHANGE	1	6.552us	6.552us	1	0	0	-1.00 B	UNPARTITIONED
07:AGGREGATE	1	124.979ms	124.979ms	1	0	192.00 KB	10.00 MB	
06:HASH JOIN	1	5.61ms	5.61ms	823	0	3.05 MB	0	INNER JOIN, BROADCAST
--10:EXCHANGE	1	15.758us	15.758us	823	0	0	0	BROADCAST
03:SCAN HDFS	1	134.144ms	134.144ms	823	0	100.00 KB	32.00 MB	default.sample_07 d
05:HASH JOIN	1	5.219ms	5.219ms	823	0	3.05 MB	0	INNER JOIN, BROADCAST
--09:EXCHANGE	1	15.894us	15.894us	823	0	0	0	BROADCAST
02:SCAN HDFS	1	164.407ms	164.407ms	823	0	100.00 KB	32.00 MB	default.sample_07 c
04:HASH JOIN	1	206.87ms	206.87ms	823	0	3.04 MB	0	INNER JOIN, BROADCAST
--08:EXCHANGE	1	15.141us	15.141us	823	0	0	0	BROADCAST
01:SCAN HDFS	1	125.507ms	125.507ms	823	0	100.00 KB	32.00 MB	default.sample_07 b
00:SCAN HDFS	1	1.873ms	1.873ms	823	0	100.00 KB	32.00 MB	default.sample_07 a

Impala Best Practices [Hide Descriptions](#)

Filter charts, e.g. 'query_duration > 10s'

Filter

30m 1h

This page contains charts to help identify whether Impala best practices are being followed. See the individual charts for a description of that best practice and how to identify if it is being followed. The documentation should be consulted for further detail on each best practice and for additional best practices.

Adjust the time range to see data on queries run at different times. Click on the charts to get more detail on individual queries. The filter box at the top right of the page may be used to adjust what is shown on this page. For example it is possible to see just the queries that took more than ten seconds by making the filter **query_duration > 10s**.

It is possible to setup triggers based on each best practice by choosing the **Create Trigger** from the individual chart drop downs.

Statistics Missing

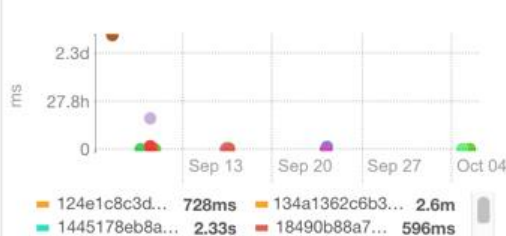


Chart showing the duration of queries with missing table or column statistics. Impala requires statistics to generate efficient plans. Missing statistics can result in bad plans and poor query performance.

HDFS Average Scan Range

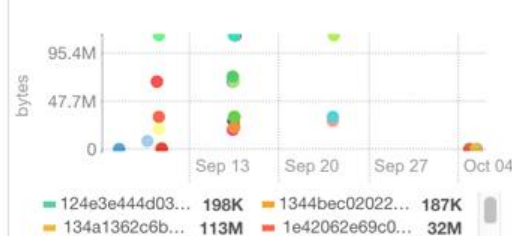


Chart showing the average HDFS scan size of each query. Low numbers for a query might indicate that the query read from many small files which negatively impacts performance.

HDFS Remote Bytes Read Percentage

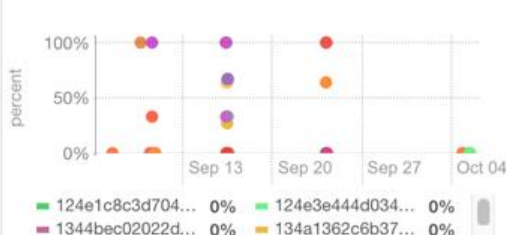


Chart showing the percentage of HDFS data read remotely by each query. For maximum performance, this number should be close to 0% for all queries.

HDFS Remote Bytes Read

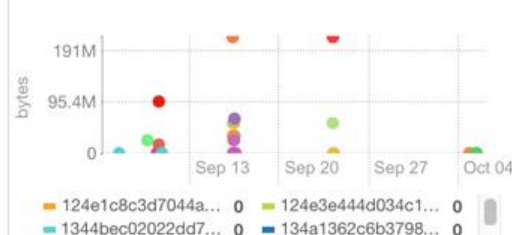


Chart showing the amount of HDFS data read remotely by each query. For maximum performance, this number should be as close to 0 for all queries.

Cloudera ODBC and JDBC Drivers

- Written by Simba Technologies
- Standard BI data access

The screenshot shows the Cloudera website header with the logo and navigation links: Why Cloudera, Products, Services & Support, Solutions, and Get Started. The main heading is 'Impala JDBC Connector 2.5.5 for Cloudera Enterprise' with the subtext 'Easily Build BI Applications with Open Source, Interactive SQL'. Below this, there are two columns. The left column is titled 'Enable your enterprise users' and contains two paragraphs explaining the driver's purpose and how it translates JDBC calls to SQL for the Impala engine. The right column is titled 'Impala JDBC Drivers 2.5.5' and features a dropdown menu labeled 'SELECT A DIFFERENT VERSION' with a list of available versions: 2.5.31, 2.5.30, 2.5.29, 2.5.28, and 2.5.24 for Cloudera Enterprise.

cloudera Why Cloudera Products Services & Support Solutions Get Started

Impala JDBC Connector 2.5.5 for Cloudera Enterprise

Easily Build BI Applications with Open Source, Interactive SQL

Enable your enterprise users

The Cloudera JDBC Driver for Impala enables your enterprise users to access Hadoop data through Business Intelligence (BI) applications with JDBC support.

The driver achieves this by translating Open Database Connectivity (JDBC) calls from the application into SQL and passing the SQL queries to the underlying Impala engine.

Impala JDBC Drivers 2.5.5

SELECT A DIFFERENT VERSION

- Impala JDBC Connector 2.5.31 for Cloudera Enterprise
- Impala JDBC Connector 2.5.30 for Cloudera Enterprise
- Impala JDBC Connector 2.5.29 for Cloudera Enterprise
- Impala JDBC Connector 2.5.28 for Cloudera Enterprise
- Impala JDBC Connector 2.5.24 for Cloudera Enterprise

Tableau

URL: ec2-54-201-244-144.us-west-2.compute.amazonaws.com

Port: 21050

Edit Connection

arrests (default.arrests) (default)

Cloudera Hadoop

Server: Port:

Select how to connect to the server:

Type:

Authentication:

Username:

Demo



cloudera®

Thank You

ishah@Cloudera.com

madhu@cloudera.com

murali@colaberry.com