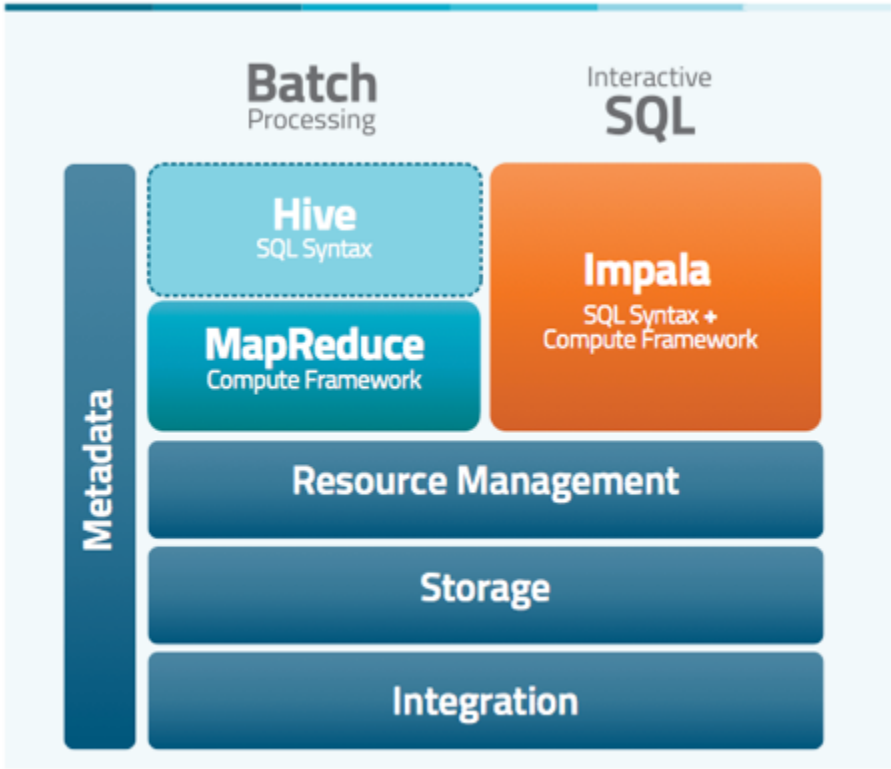
BIG DATA and AI
for business analytics

Hive and Impala

Decisions, Operations & Information Technologies
Robert H. Smith School of Business
Fall, 2020

K. Zhang (pic source: mapr.com/blog)

Where do Impala and Hive sit?



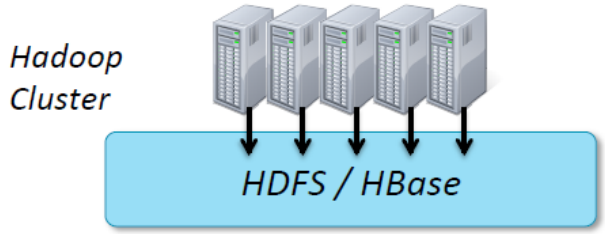
Hive processes data via MapReduce, Impala is a stand-alone MPP framework

What are Impala and Hive?

Introduction to Impala and Hive (1)

- **Impala and Hive are both tools that provide SQL querying of data stored in HDFS / HBase**

```
SELECT zipcode, SUM(cost) AS total
FROM customers
JOIN orders
ON (customers.cust_id = orders.cust_id)
WHERE zipcode LIKE '63%'
GROUP BY zipcode
ORDER BY total DESC;
```



Introduction to Impala and Hive (2)

- **Apache Hive is a high-level abstraction on top of MapReduce**
  - Uses HiveQL
  - Generates MapReduce or Spark* jobs that run on the Hadoop cluster
  - Originally developed at Facebook around 2007
    - Now an open-source Apache project

- **Cloudera Impala is a high-performance dedicated SQL engine**
  - Uses Impala SQL
  - Inspired by Google's Dremel project
  - Query latency measured in milliseconds
  - Developed at Cloudera in 2012
    - Open-source with an Apache license

* Hive-on-Spark is currently in beta testing

Differences between Hive and Impala

- **Hive has more features**
  - E.g. Complex data types (arrays, maps) and full support for windowing analytics
  - Highly extensible
  - Commonly used for batch processing

- **Impala is much faster**
  - Specialized SQL engine offers 5x to 50x better performance
  - Ideal for interactive queries and data analysis
  - More features being added over time

High-Level Overview

```
SELECT zipcode, SUM(cost) AS total
FROM customers
JOIN orders
ON (customers.cust_id = orders.cust_id)
WHERE zipcode LIKE '63%'
GROUP BY zipcode
ORDER BY total DESC;
```

- Parse HiveQL
- Make optimizations
- Plan execution
- Submit job(s) to cluster
- Monitor progress

Data Processing Engine (MapReduce)

Hadoop Cluster

HDFS

- Parse Impala SQL
- Make optimizations
- Plan execution
- Execute query on the cluster

Hadoop Cluster

HDFS

Why Use Hive and Impala?

- **Brings large-scale data analysis to a broader audience**
  - No software development experience required
  - Leverage existing knowledge of SQL
- **More productive than writing MapReduce or Spark directly**
  - Five lines of HiveQL/Impala SQL might be equivalent to 200 lines or more of Java
- **Offers interoperability with other systems**
  - Extensible through Java and external scripts
  - Many business intelligence (BI) tools support Hive and/or Impala

Simple Use Case: Log Files

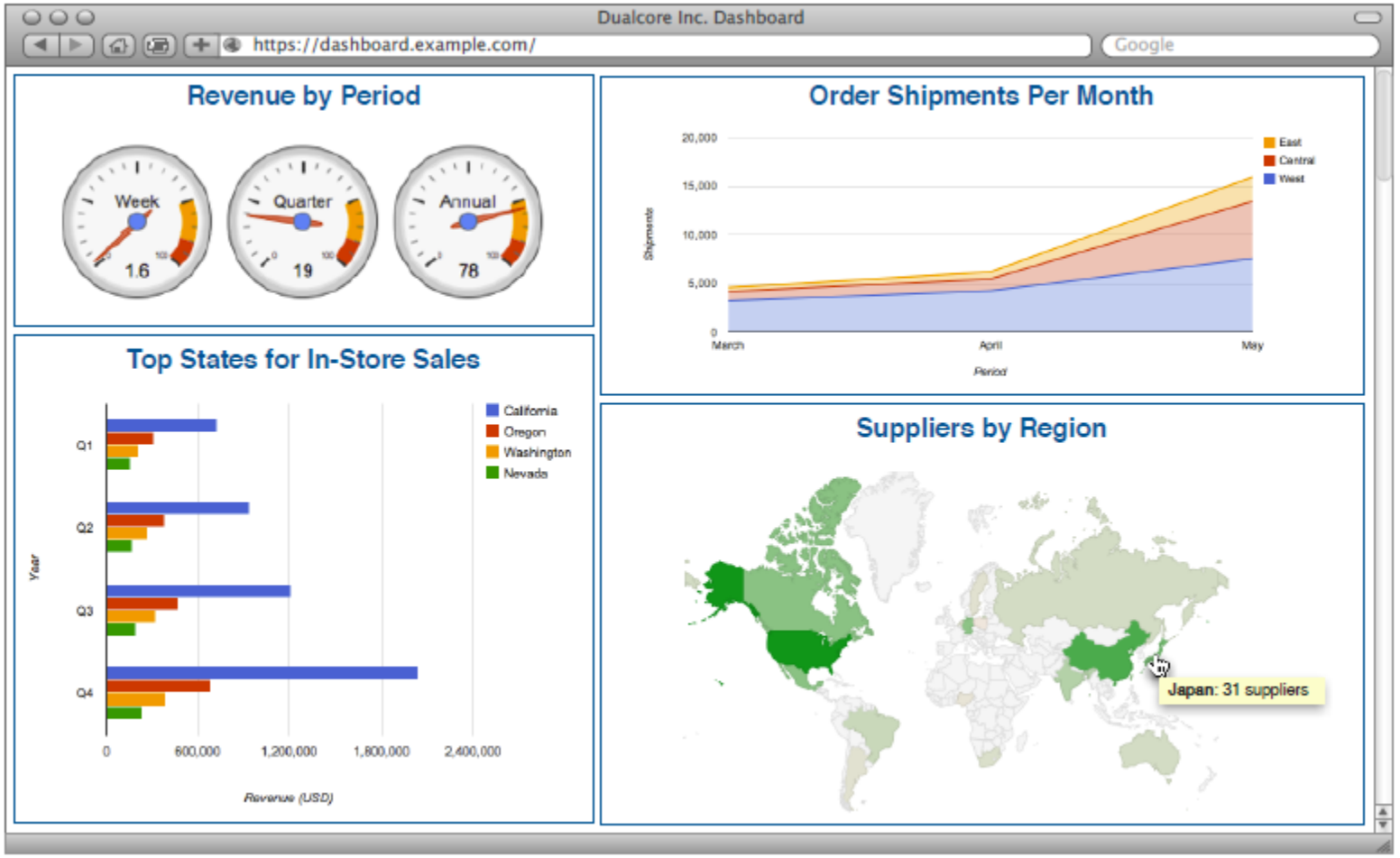
Use Case: Log File Analytics

- **Server log files are an important source of data**
- **Hive and Impala allow you to treat a directory of log files like a table**
  - Allows SQL-like queries against raw data

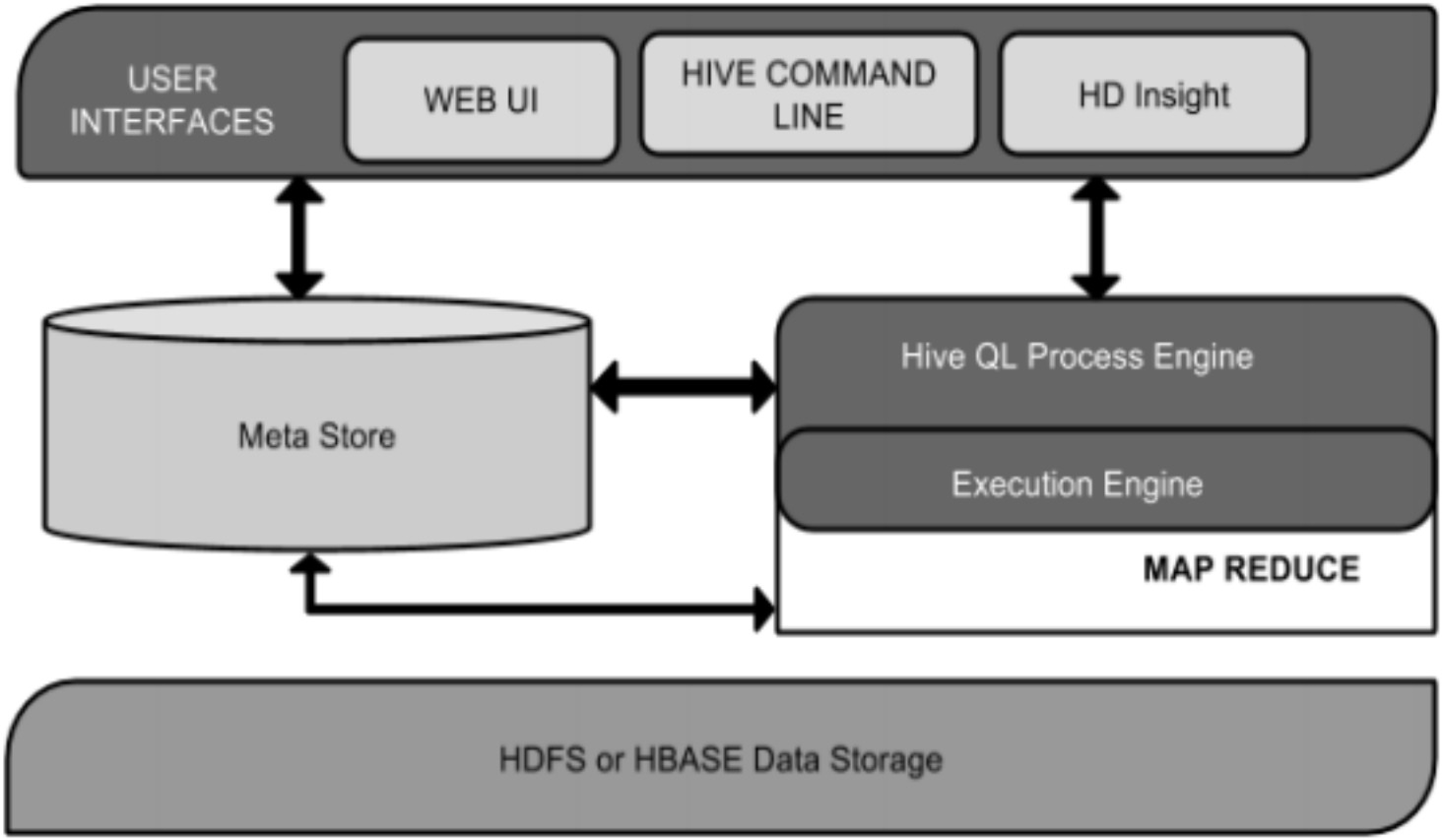
Dualcore Inc. Public Web Site (June 1 - 8)

Product	Unique Visitors	Page Views	Average Time on Page	Bounce Rate	Conversion Rate
Tablet	5,278	5,894	17 seconds	23%	65%
Notebook	4,139	4,375	23 seconds	47%	31%
Stereo	2,873	2,981	42 seconds	61%	12%
Monitor	1,749	1,862	26 seconds	74%	19%
Router	987	1,139	37 seconds	56%	17%
Server	314	504	53 seconds	48%	28%
Printer	86	97	34 seconds	27%	64%

Use Case: Getting Data for Dashboards

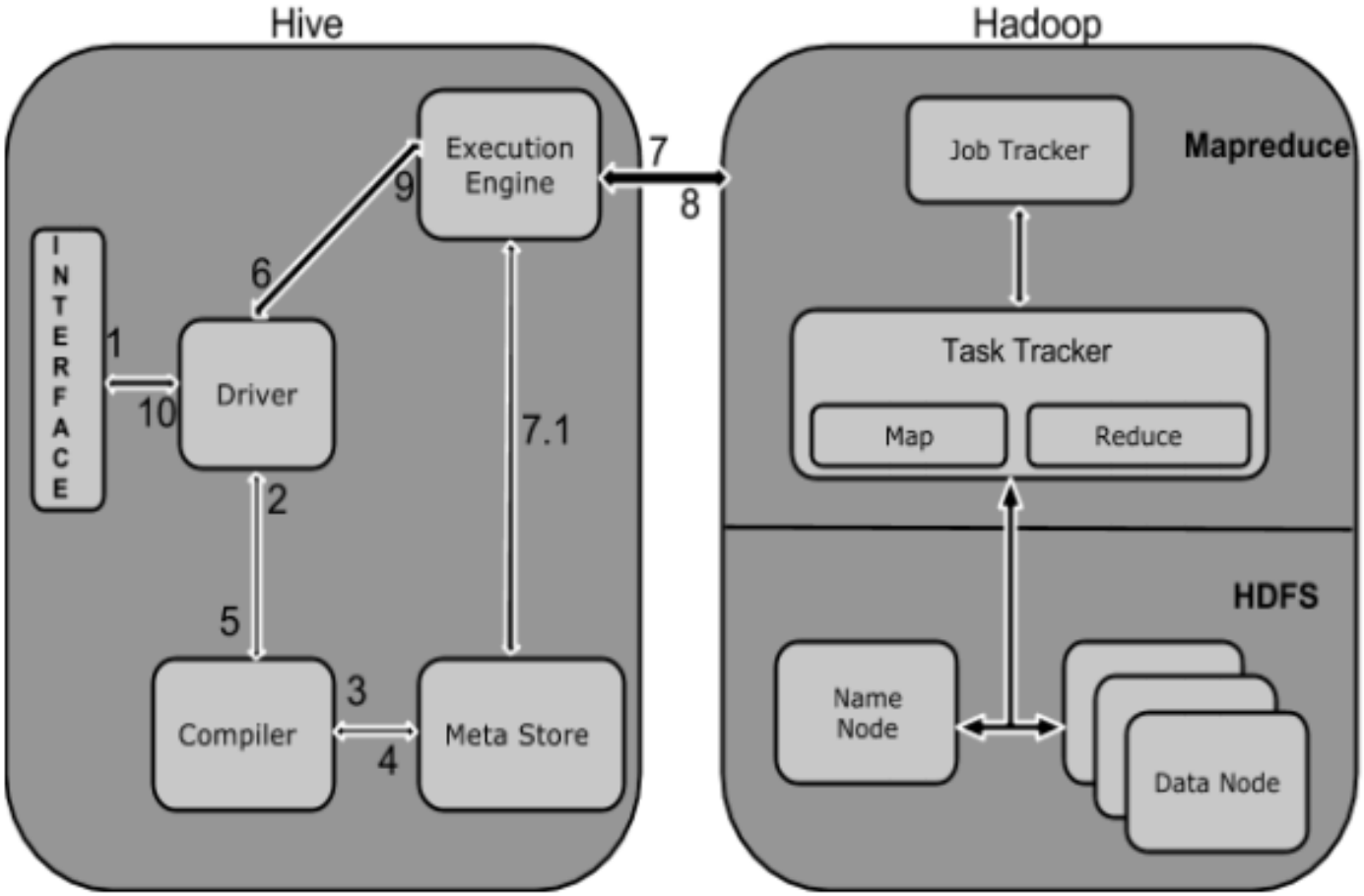


Architecture of Hive



Unite Name	Operation
User Interface	The interaction between user and HDFS. The user interfaces that Hive supports are Hive Web UI, Hive command line, and Hive HD Insight (In Windows server).
Meta Store	Hive chooses respective database servers to store the schema or Metadata of tables, databases, columns in a table, their data types, and HDFS mapping.
HiveQL Process Engine	HiveQL is similar to SQL for querying on schema info on Metastore. It is one of the replacements of traditional approach for MapReduce program. Instead of writing MapReduce program in Java, we can write a query for MapReduce job and process it.
Execution Engine	The conjunction part of HiveQL process Engine and MapReduce is Hive Execution Engine. Execution engine processes the query and generates results as same as MapReduce results. It uses the flavor of MapReduce.
HDFS or HBase	Hadoop distributed file system or HBase are the data storage techniques to store data into file system.

Working of Hive



Operations

Step No.	Operation
1	**Execute Query** The Hive interface such as Command Line or Web UI sends query to Driver (any database driver such as JDBC, ODBC, etc.) to execute.
2	**Get Plan** The driver takes the help of query compiler that parses the query to check the syntax and query plan or the requirement of query.
3	**Get Metadata** The compiler sends metadata request to Metastore (any database).
4	**Send Metadata** Metastore sends metadata as a response to the compiler.
5	**Send Plan** The compiler checks the requirement and resends the plan to the driver. Up to here, the parsing and compiling of a query is complete.

Operations (Cont'd)

Step No.	Operation
6	**Execute Plan** The driver sends the execute plan to the execution engine.
7	**Execute Job** Internally, the process of execution job is a MapReduce job. The execution engine sends the job to JobTracker, which is in Name node and it assigns this job to TaskTracker, which is in Data node. Here, the query executes MapReduce job.
7.1	**Metadata Ops** Meanwhile in execution, the execution engine can execute metadata operations with Metastore.
8	**Fetch Result** The execution engine receives the results from Data nodes.
9	**Send Results** The execution engine sends those resultant values to the driver.
10	**Send Results** The driver sends the results to Hive Interfaces.

Impala and Hive: Instructions and Commands

Interacting with Hive and Impala

- **Hive and Impala offer many interfaces for running queries**
  - Command-line shell
    - Impala: Impala shell
    - Hive: Beeline
  - Hue Web UI
    - Hive Query Editor
    - Impala Query Editor
    - Metastore Manager
  - ODBC / JDBC

Impala - Shell

Starting the Impala Shell

- **You can execute statements in the Impala shell**
  - This interactive tool is similar to the shell in MySQL
- **Execute the `impala-shell` command to start the shell**
  - Some log messages truncated to better fit the slide

```
$ impala-shell
Connected to localhost.localdomain:21000
Server version: impalad version 2.1.0-cdh5 (…)
Welcome to the Impala shell.
[localhost.localdomain:21000] >
```

- **Use `-i hostname:port` option to connect to a different server**

```
$ impala-shell -i myserver.example.com:21000
[myserver.example.com:21000] >
```

Impala - Shell

Using the Impala Shell

- **Enter semicolon-terminated statements at the prompt**
  - Hit [Enter] to execute a query or command
  - Use the `quit` command to exit the shell
- **Use `impala-shell --help` for a full list of options**

Executing Queries in the Impala Shell

```
> SELECT lname,fname FROM customers WHERE state = 'CA'
limit 50;

Query: select lname,fname FROM customers WHERE state =
'CA' limit 50
+-----------+-----------+
| lname     | fname     |
+-----------+-----------+
| Ham       | Marilyn   |
| Franks    | Gerard    |
| Preston   | Mason     |
| Cortez    | Pamela    |
…
| Falgoust  | Jennifer  |
+-----------+-----------+
Returned 50 row(s) in 0.17s

>
```

Note: shell prompt abbreviated as >

Hive Shell - Beeline

- **You can execute HiveQL statements in the Beeline shell**
  - Interactive shell based on the SQLLine utility
  - Similar to the Impala shell
- **Start Beeline by specifying the URL for a Hive2 server**
  - Plus username and password if required

```
$ beeline -u jdbc:hive2://host:10000 \
-n username -p password

0: jdbc:hive2://localhost:10000>
```

Executing Queries in Beeline

- **SQL commands are terminated with semi-colon (;)**
- **Similar to Impala shell**
  - Results formatting is slightly different

```
1: url> SELECT lname,fname FROM customers
. . . > WHERE state = 'CA' LIMIT 50;

+-------------+-------------+
|    lname    |    fname    |
+-------------+-------------+
| Ham         | Marilyn     |
| Franks      | Gerard      |
| Preston     | Mason       |
...
| Falgoust    | Jennifer    |
+-------------+-------------+
50 rows selected (15.829 seconds)

1: url>
```

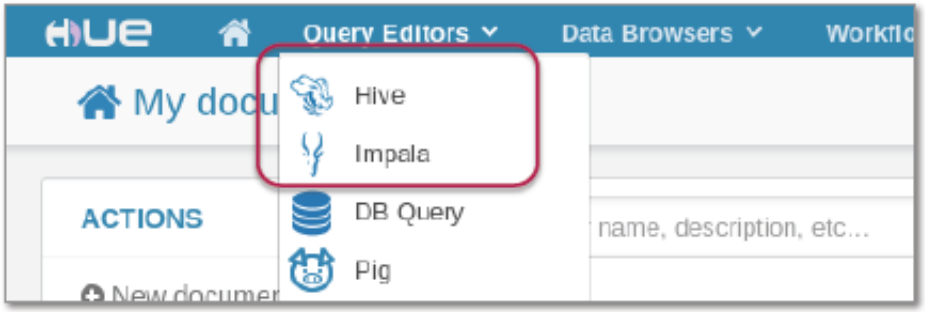
- **Execute Beeline commands with '!'**
  - No terminator character
- **Some commands**
  - `!connect url` – connect to a different Hive2 server
  - `!exit` – exit the shell
  - `!help` – show the full list of commands
  - `!verbose` – show added details of queries

```
0: jdbc:hive2://localhost:10000> !exit
```

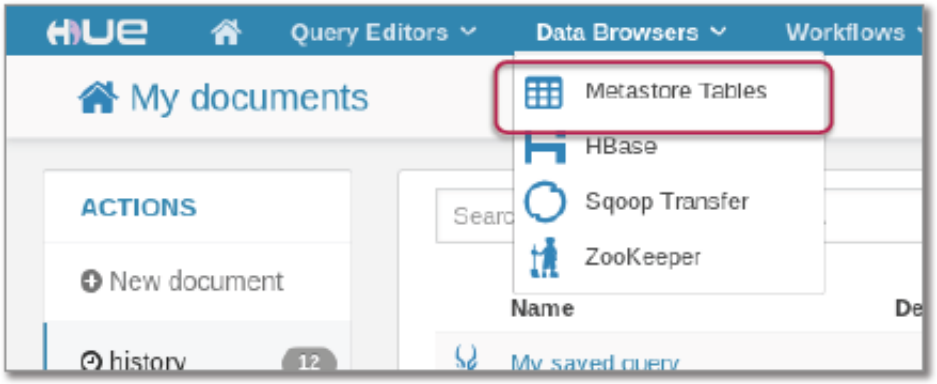
Using HUE with Hive and Impala

You can use Hue to…

Query data with Hive or Impala



View and manage the Metastore



The cluster is not a DBMS…

	Relational Database	Hive	Impala
Query language	SQL (full)	SQL (subset)	SQL (subset)
Update individual records	Yes	No	No
Delete individual records	Yes	No	No
Transactions	Yes	No	No
Index support	Extensive	Limited	No
Latency	Very low	High	Low
Data size	Terabytes	Petabytes	Petabytes

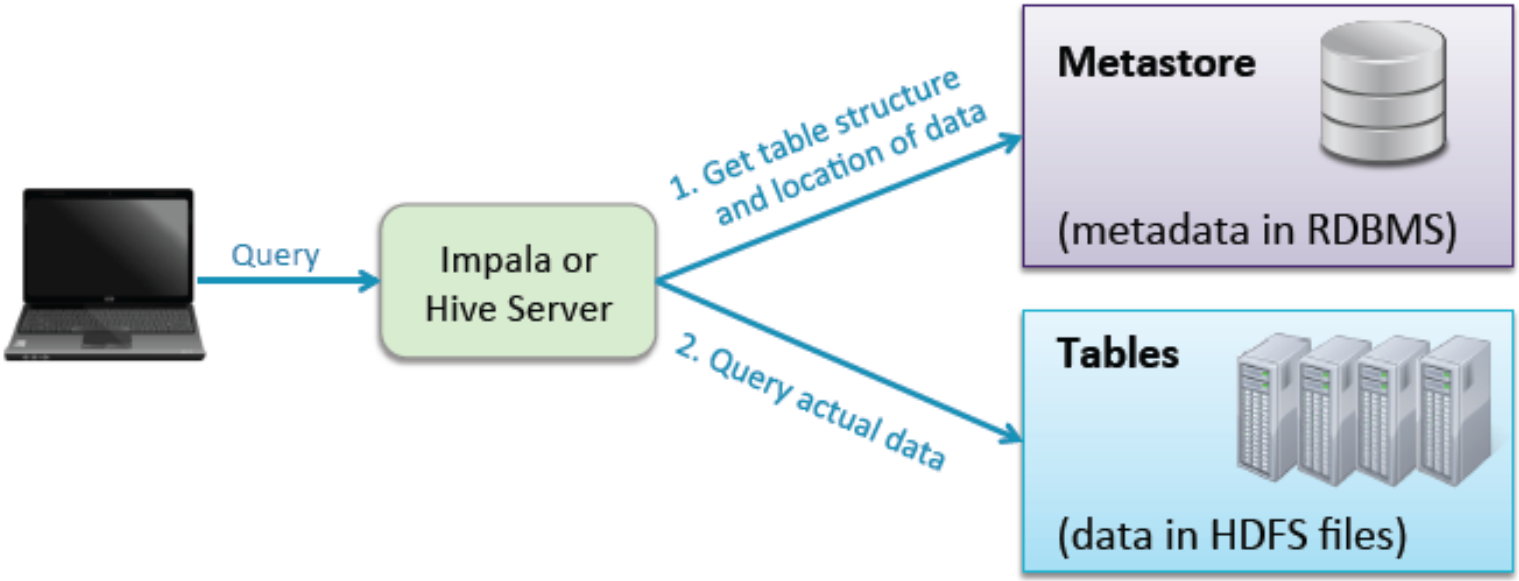
Impala and Hive: Modeling and Managing Data

Storing data on the cluster

- **Queries operate on tables, just like in an RDBMS**
  - A table is simply an HDFS directory containing one or more files
  - Default path: `/user/hive/warehouse/<table_name>`
  - Supports many formats for data storage and retrieval
- **What is the structure and location of tables?**
  - These are specified when tables are created
  - This metadata is stored in the *Metastore*
    - Contained in an RDBMS such as MySQL
- **Hive and Impala work with the same data**
  - Tables in HDFS, metadata in the Metastore

The Metastore

- **Hive and Impala use the Metastore to determine data format and location**
  - The query itself operates on data stored in HDFS



Metadata and data

- ***Data*** **refers to the information you store and process**
  - Billing records, sensor readings, and server logs are examples of data
- ***Metadata*** **describes important aspects of that data**
  - Field name and order are examples of metadata

Metadata

Data

cust_id	name	country
001	Alice	us
002	Bob	ca
003	Carlos	mx
…	…	…
392	Maria	it
393	Nigel	uk
394	Ophelia	dk
…	…	…

Operations on Data

- Creating databases and tables
  - Using HiveQL or Impala SQL data definition language (DDL)
    - Based on SQL DDL
    - See slides 6-10 to 6-27
  - External versus Internal tables
    - Dropping ‘external’ table does not delete data, only removes entry from metadata
  - Can also be done through HUE metastore manager (slide 6-28)

"External" Tables

- **CAUTION: Dropping a table removes its data in HDFS**
  - Tables are "managed" or "internal" by default
- **Using `EXTERNAL` when creating the table avoids this behavior**
  - Dropping an *external* table removes only its *metadata*

```
CREATE EXTERNAL TABLE adclicks
  ( campaign_id STRING,
    click_time TIMESTAMP,
    keyword STRING,
    site STRING,
    placement STRING,
    was_clicked BOOLEAN,
    cost SMALLINT)
  LOCATION '/loudacre/ad_data';
```

Recall…

Data Validation

- **Impala and Hive are 'schema on read'**
  - Unlike an RDBMS, they do not validate data on insert
    - Files are simply moved into place
  - Loading data into tables is therefore very fast
  - Errors in file format will be discovered when queries are performed
- **Missing or invalid data will be represented as `NULL`**

Adding data to the database

- Adding files through HDFS – adding files to the table's directory in HDFS
  - Slides 6-31 to 6-33
- Load data using the HUE Metastore Manager
  - Slide 6-34
- You also have the option of using Sqoop
  - Slide 6-35
- Alternatively, use Hcatalog, access the Metastore directly
  - Slides 6-37 to 6-40

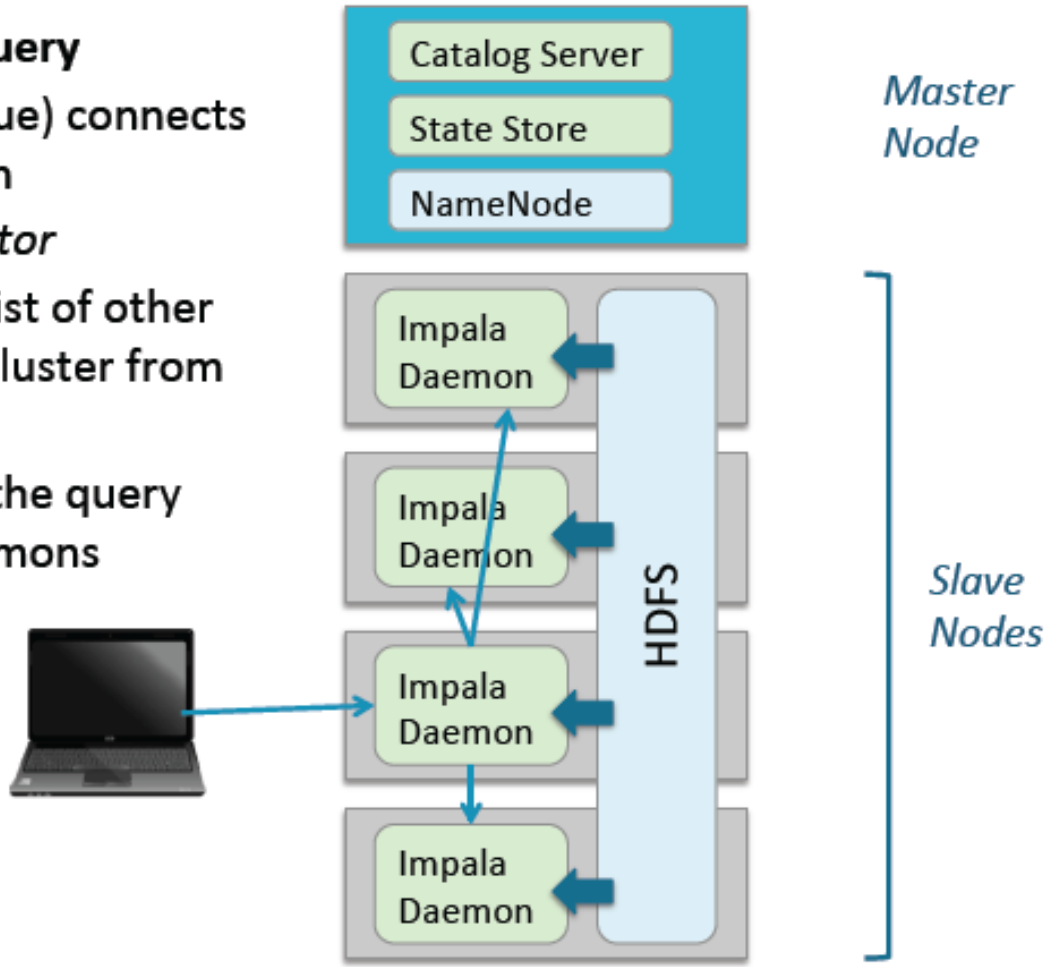
Impala – Works through the Metastore

Labs

Impala – Additional Information (Optional)

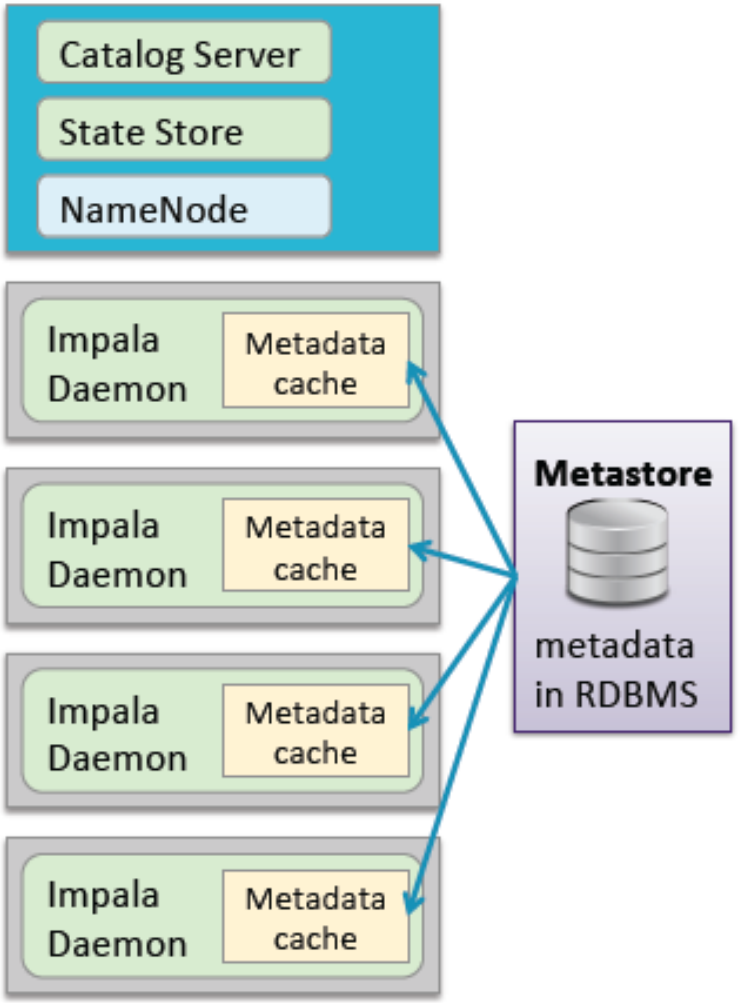
How Impala Executes a Query

- **Impala daemon plans the query**
  - Client (impala-shell or Hue) connects to a local impala daemon
    - This is the *coordinator*
  - Coordinator requests a list of other Impala daemons in the cluster from the State Store
  - Coordinator distributes the query across other Impala daemons
  - Streams results to client



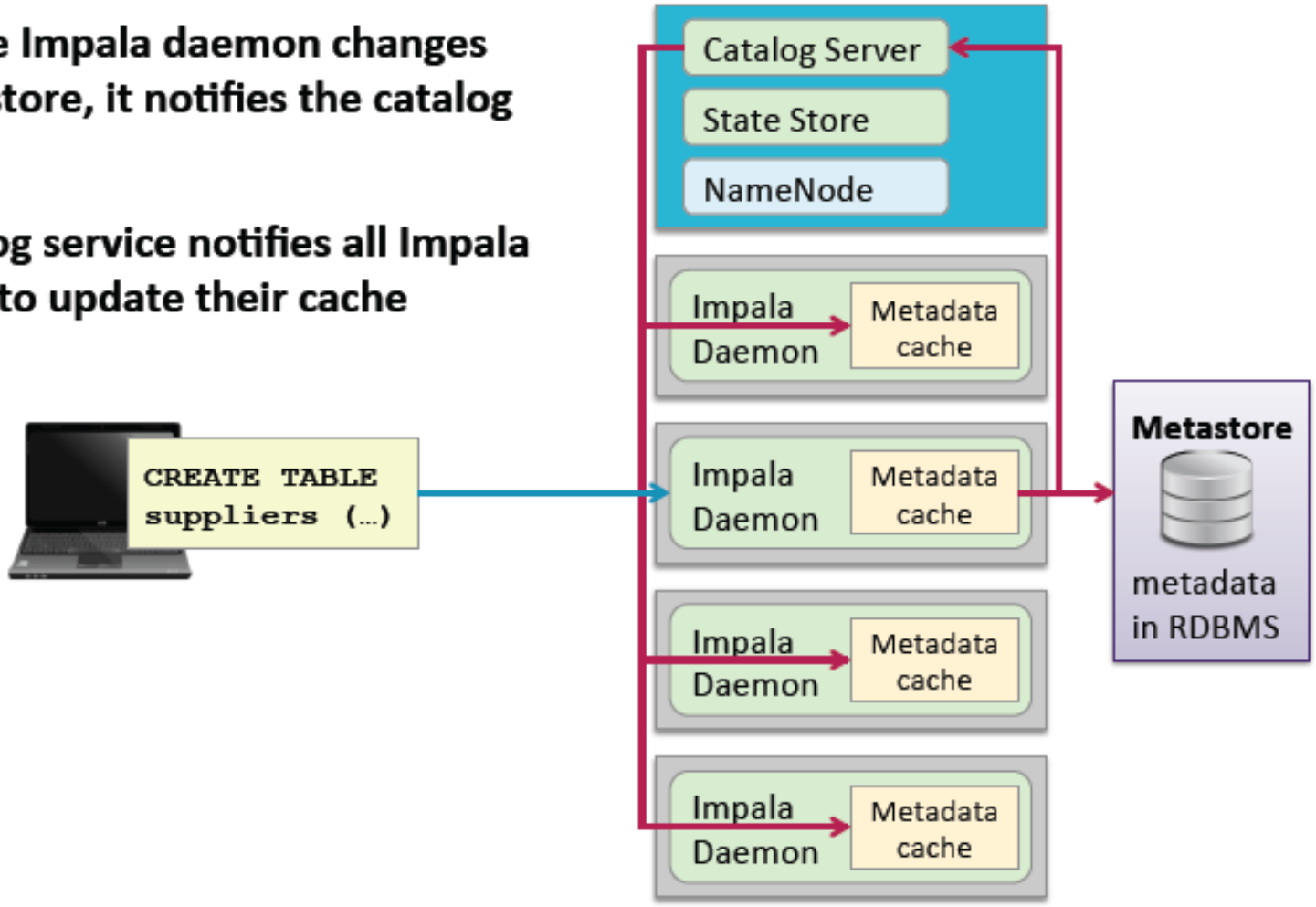
Metadata Caching (1)

- **Impala daemons cache metadata**
  - The tables' schema definitions
  - The locations of tables' HDFS blocks
- **Metadata is cached from the Metastore at startup**



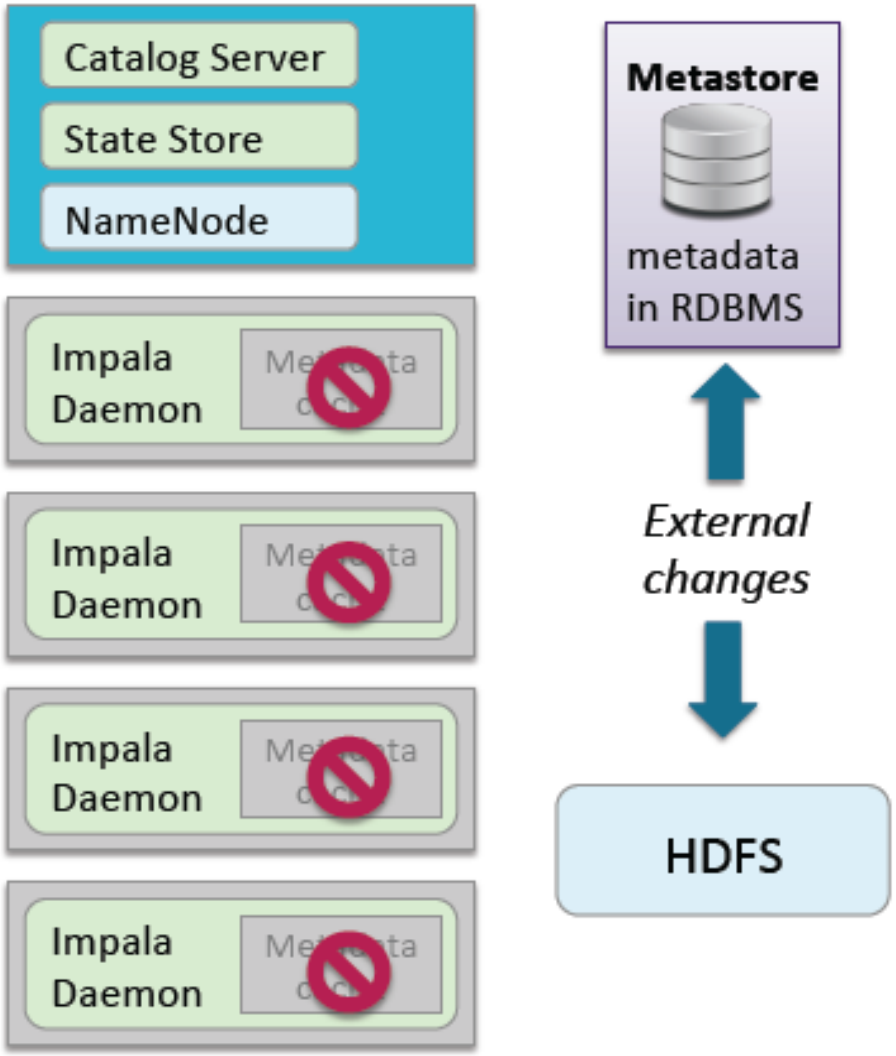
Metadata Caching (2)

- **When one Impala daemon changes the metastore, it notifies the catalog service**
- **The catalog service notifies all Impala daemons to update their cache**



External Changes and Metadata Caching

- **Metadata updates made *from outside of Impala* are not known to Impala, e.g.**
  - Changes via Hive, HCatalog or Hue Metadata Manager
  - Data added directly to directory in HDFS
- **Therefore the Impala metadata caches will be invalid**
- **You must manually refresh or invalidate Impala's metadata cache**



Updating the Impala Metadata Cache

External Metadata Change	Required Action	Effect on Local Caches
New table added	**`INVALIDATE METADATA`** (with no table name)	Marks the entire cache as stale; metadata cache is reloaded as needed.
Table schema modified *or* New data added to a table	**`REFRESH <table>`**	Reloads the metadata for one table *immediately*. Reloads HDFS block locations for new data files only.
Data in a table extensively altered, such as by HDFS balancing	**`INVALIDATE METADATA <table>`**	Marks the metadata for a single table as stale. When the metadata is needed, all HDFS block locations are retrieved.