Hive and Pig

Enterprise Architectures for Big Data

Hadoop Ecosystem

oozie
(Work flow)

HCatalog
Table & schema Management

Pig
(Scripting)

Hive
(Sql Query)

mahout
(Machine Learning)

Drill
(Interactive Analysis)

AVRO
(JSON)

Thrift
(Cross Language Service)

APACHE HBASE
HBASE
(Columnar Store)

Sqoop
(Data Collection)

Zookeeper
(Coordination)

Ambari

Apache Ambari
(Management & Monitoring)

Mapreduce
(Data Processing)

Yarn
(Cluster Resource Management)

HDFS
(Hadoop Distributed File system)

FLUME
Flume
(Data Collection)

Hadoop Ecosystem

- Hive – provides a SQL like query capability
- Pig – a high-level language for creating MapReduce jobs
- HCatalog – takes Hive's metadata and makes it available across the Hadoop ecosystem
- Hbase – a column-oriented NoSQL data store
- Mahout – a library of algorithms for clustering, classification, and filtering
- Sqoop – accelerates bulk loads of data between Hadoop and Relational Databases
- Flume – streams large volumes of log data from multiple sources into Hadoop

Source: Hortonworks

Need for High-Level Languages

- Hadoop is great for large-data processing!
  - But writing MapReduce programs for everything is verbose and slow
  - Not everyone wants to (or can) write MapReduce code

- Solution: develop higher-level data processing languages
  - Hive
  - Pig

Hive and Pig

- Hive: data warehousing application in Hadoop
  - Query language is HQL, variant of SQL
  - Tables stored on HDFS as flat files
  - Developed by Facebook, now open source
- Pig: large-scale data processing system
  - Scripts are written in Pig Latin, a dataflow language
  - Developed by Yahoo!, now open source
- Common idea:
  - Provide higher-level language to facilitate large-data processing
  - Higher-level language “compiles down” to Hadoop jobs

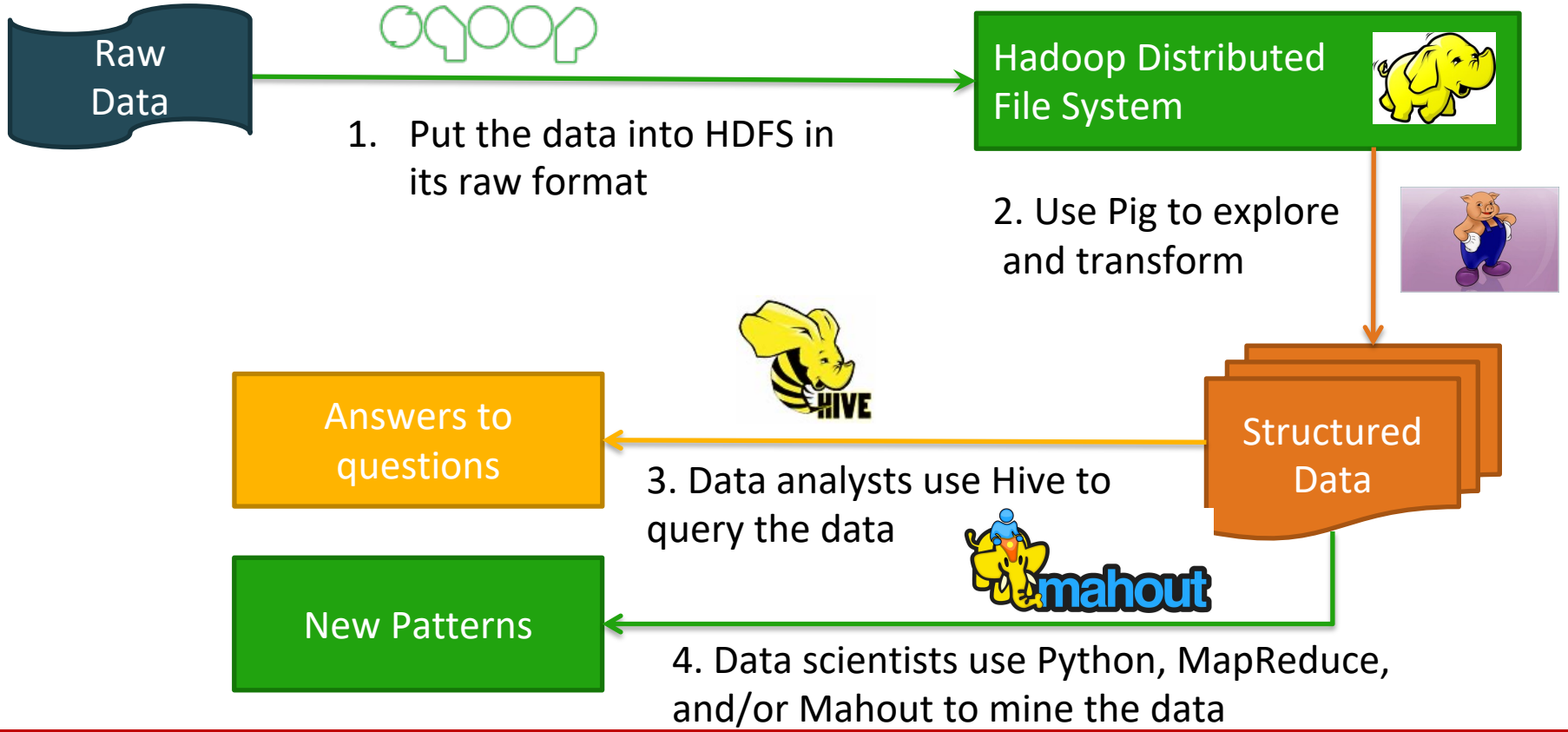
Source: Hortonworks

Hive and Pig

Hive	Apache Pig
For querying data	For ETL (Extract, Transform, Load)
Hive uses a language called HiveQL. It was originally created at Facebook.	Apache Pig uses a language called Pig Latin. It was originally created at Yahoo.
HiveQL is a query processing language very similar to SQL	Pig Latin is a data flow language.
HiveQL is like SQL a declarative language.	Pig Latin is a procedural language and it fits in pipeline paradigm.
Hive is mostly for structured data.	Apache Pig can handle structured, unstructured, and semi-structured data.

Source: Hortonworks

Interplay of different Hadoop Elements



Example Use Case

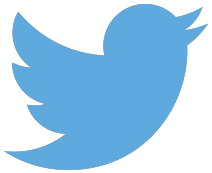
Sentiment Use Case

- Analyze customer sentiment on the days leading up to and following the release of the latest Avenger movie

- Questions to answer:
  - How did the public feel about the debut?
  - How might the sentiment data have been used to better promote the launch of the movie?

Source: Hortonworks

Getting Twitter Feeds into Hadoop with Flume



Flume Agent

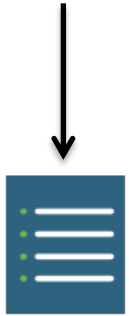
Avengers Endgame was awesome. I want to go see it again!
Avengers Endgame = 7.7 stars
Tony Stark has 42 different Iron Man suits in Avengers Endgame
Wow as good as or better than the last one
Endgame was way better than Infinity War.

Flume is a tool for streaming data into Hadoop.

Hadoop cluster

HIVE and HCatalog for Defining a Schema

```
CREATE EXTERNAL TABLE tweets_raw
(
   id BIGINT,
   created_at STRING,
   source STRING,
   favorited BOOLEAN,
   retweet_count INT,
   text STRING
)
```

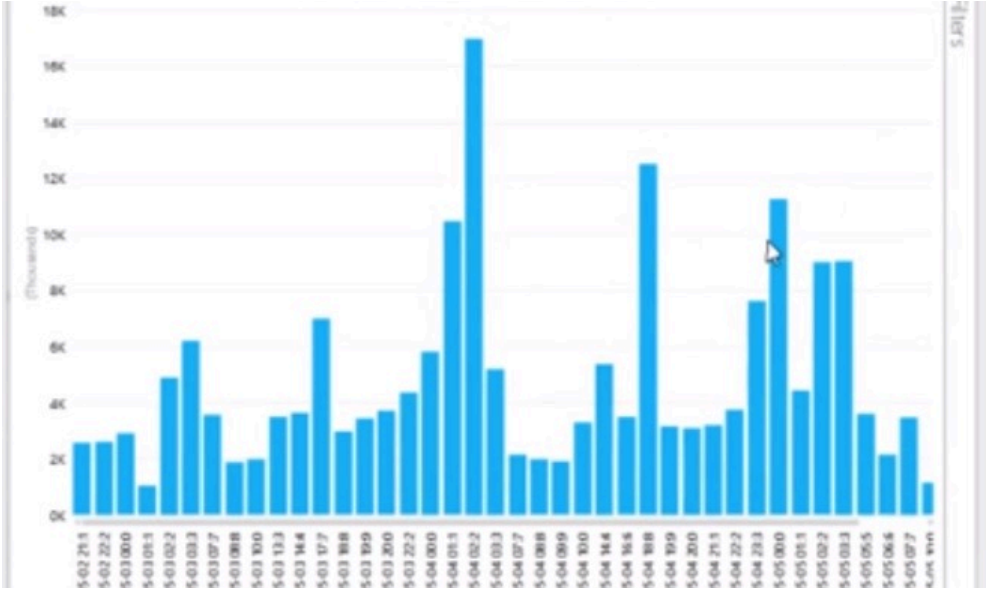


HCatalog metastore

Use Hive to Determine Sentiment

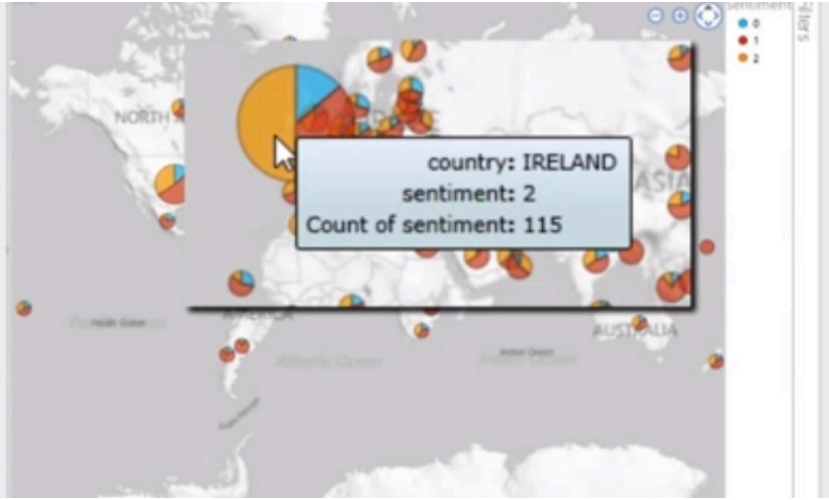
```
CREATE TABLE tweetsbi
STORED AS RCFile
AS
SELECT
  t.*,
  case s.sentiment
    when 'positive' then 2
    when 'neutral' then 1
    when 'negative' then 0
  end as sentiment
FROM tweets_clean t LEFT OUTER JOIN tweets_sentiment s on t.id = s.id;
```

Analyze Tweet Volume in Jupyter or Zeppelin

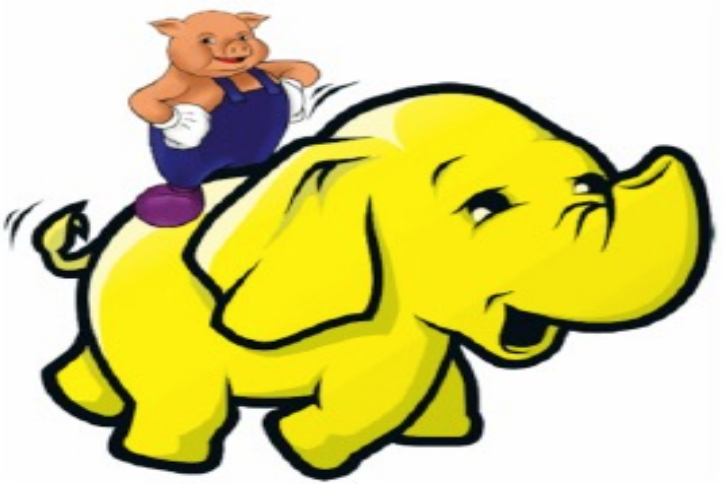


A large spike in tweets around the Thursday midnight opening and spikes around the Friday evening, Saturday afternoon, and Saturday evening showings

Connect Hive to Tableau
View Sentiment by Country

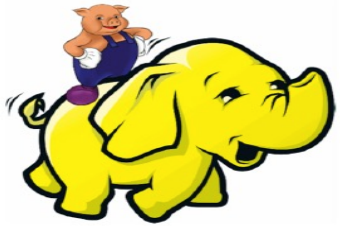


- Viewing the tweets on a map shows the sentiment of the movie by country.
- For example, Ireland had 50% positive tweets, while 67% of tweets from Mexico were neutral.



Pig

Pig

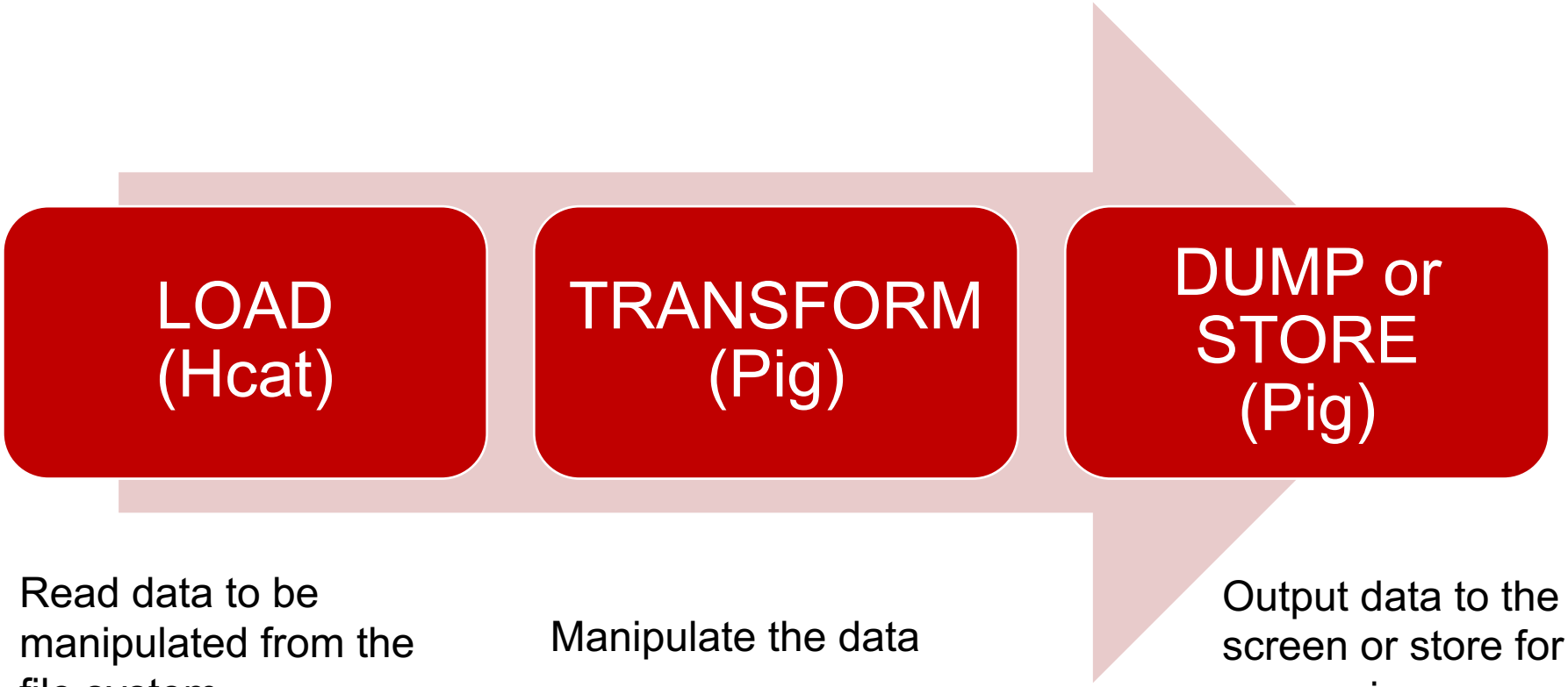


- An engine for executing programs on top of Hadoop
- It provides a language, Pig Latin, to specify these programs
- Why the name Pig?
  - **Pigs eat anything**
    - Pig can process any data, structured or unstructured
  - **Pigs live anywhere**
    - Pig can run on any parallel data processing framework, so Pig scripts do not have to run just on Hadoop
  - **Pigs are domestic animals**
    - Pig is designed to be easily controlled and modified by its users
  - **Pigs fly**
    - Pig is designed to process data quickly

Pig Latin

- High-level data-flow scripting language
- Pig executes in a unique fashion:
  - During execution, each statement is processed by the Pig interpreter
  - If a statement is valid, it gets added to a logical plan built by the interpreter
  - The steps in the logical plan do not actually execute until a DUMP or STORE command is used

Pig ETL Flow


LOAD
(Hcat)
TRANSFORM
(Pig)
DUMP or
STORE
(Pig)
Read data to be manipulated from the file system
Manipulate the data
Output data to the screen or store for processing

Pig vs. MapReduce

Apache Pig	MapReduce
Apache Pig is a data flow language.	MapReduce is a data processing paradigm.
It is a high level language.	MapReduce is low level and rigid.
Performing a Join operation in Apache Pig is pretty simple.	It is quite difficult in MapReduce to perform a Join operation between datasets.
Any novice programmer with a basic knowledge of SQL can work conveniently with Apache Pig.	Exposure to Java is must to work with MapReduce.
Apache Pig uses multi-query approach, thereby reducing the length of the codes to a great extent.	MapReduce will require almost 20 times more the number of lines to perform the same task.
There is no need for compilation. On execution, every Apache Pig operator is converted internally into a MapReduce job.	MapReduce jobs have a long compilation process.

Source: Hortonworks

Pig vs. SQL

Pig	SQL
Pig Latin is a **procedural** language.	SQL is a **declarative** language.
In Apache Pig, **schema** is optional. We can store data without designing a schema (values are stored as $01, $02 etc.)	Schema is mandatory in SQL.
The data model in Apache Pig is **nested relational**.	The data model used in SQL **is flat relational**.
Apache Pig provides limited opportunity for **Query optimization**.	There is more opportunity for query optimization in SQL.

Source: Hortonworks

Pig Latin – Data Model

- Data model is fully nested
- A **Relation** is the outermost structure of the Pig Latin data model.
- And it is a **bag** where
  - A bag is a collection of tuples.
  - A tuple is an ordered set of fields.
  - A field is a piece of data.

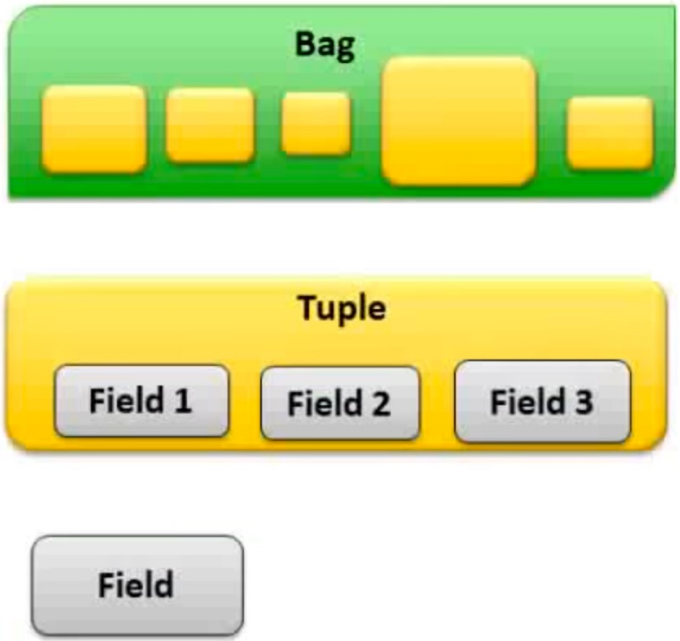
Pig Latin – Data Model

1. A relation is a bag (more specifically, an outer bag).

1. A bag is a collection of unordered tuples (can be different sizes).

2. A tuple is an ordered set of fields.

3. A field is a piece of data.



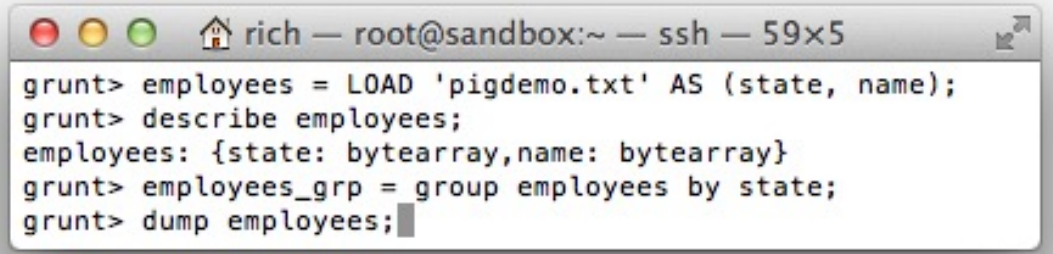
Pig Commands

Command	Description
LOAD	Read data from file system
STORE	Write data to file system
FOREACH	Apply expression to each record and output 1+ records
FILTER	Apply predicate and remove records that do not return true
GROUP/COGROUP	Collect records with the same key from one or more inputs
JOIN	Joint 2+ inputs based on a key; various join algorithms exist
ORDER	Sort records based on a key
DISTINCT	Remove duplicate records
UNION	Merge two data sets
SPLIT	Split data into 2+ more sets based on filter conditions
STREAM	Send all records through a user provided executable
SAMPLE	Read a random sample of the data
LIMIT	Limit the number of records

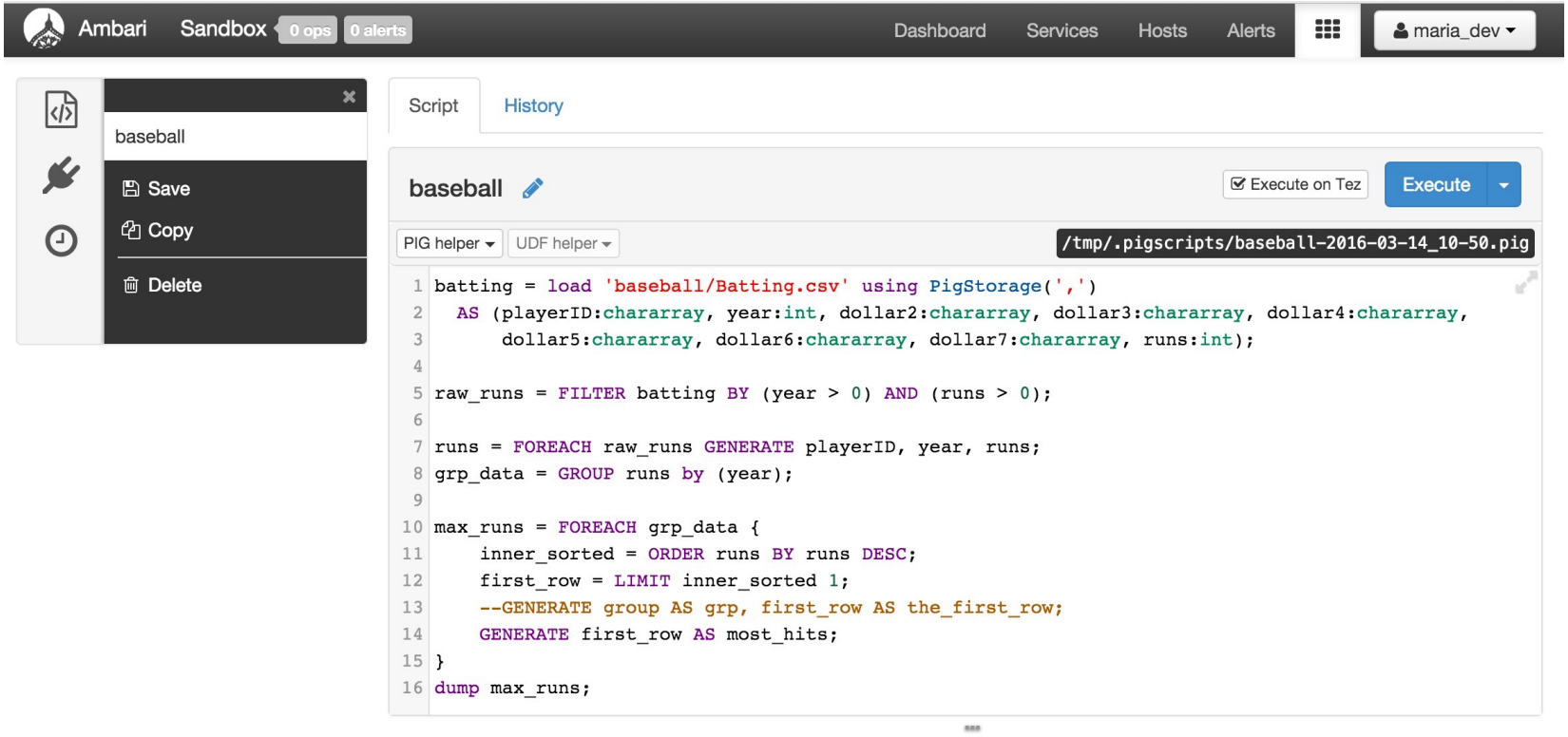
Source: Hortonworks

The Grunt Shell

- An interactive shell for entering Pig Latin statements
- Started by running the pig executable



Executing Pig Scripts in Ambari Pig View



Why Use Pig?

- Example Scenario: We want to target a subset of our users and then determine the most popular pages they access

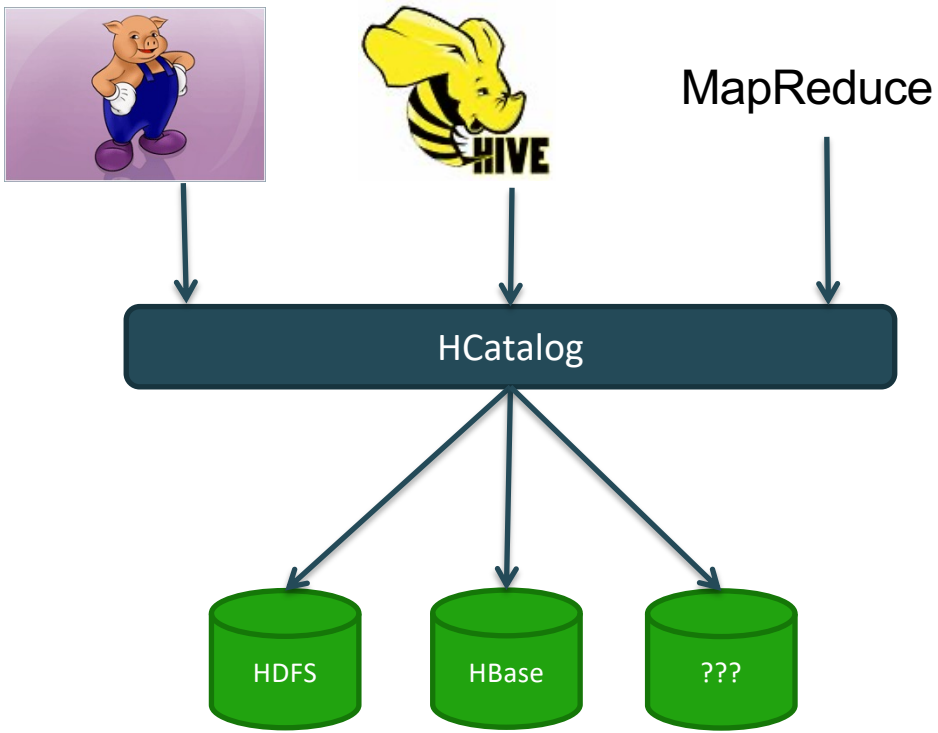
```
1  users = LOAD 'input/users' USING PigStorage(',')
2          AS (name:chararray, age:int);
3
4  filtrd = FILTER users BY age >= 18 and age <= 25;
5
6  pages = LOAD 'input/pages' USING PigStorage(',')
7          AS (user:chararray, url:chararray);
8
9  jnd = JOIN filtrd BY name, pages BY user;
10
11 grpd = GROUP jnd BY url;
12
13 smmd = FOREACH grpd GENERATE group, COUNT(jnd) AS clicks;
14
15 srtd = ORDER smmd BY clicks DESC;
16
17 top5 = LIMIT srtd 5;
18
19 STORE Top5 INTO 'output/top5sites' USING PigStorage(',');
```

Pig DataFu Library

- A collection of Pig UDFs for data analysis on Hadoop
- Started by LinkedIn and open-sourced under the Apache 2.0 license

- Includes functions for:
  - Bag and set operations
  - PageRank
  - Quantiles
  - Variance
  - Sessionization

HCatalog in the Ecosystem



Summary Pig

- Pig is a high-level data-flow scripting language
- Scripts do not execute until an I/O operation like DUMP or STORE are reached
- Can be run via the interactive shell or as a script
- Has a comprehensive set of commands available to Pig programmers
- DataFu library is a collection of Pig UDFs for data analysis on Hadoop
- HCatalog provides a consistent data model for the various tools that use Hadoop

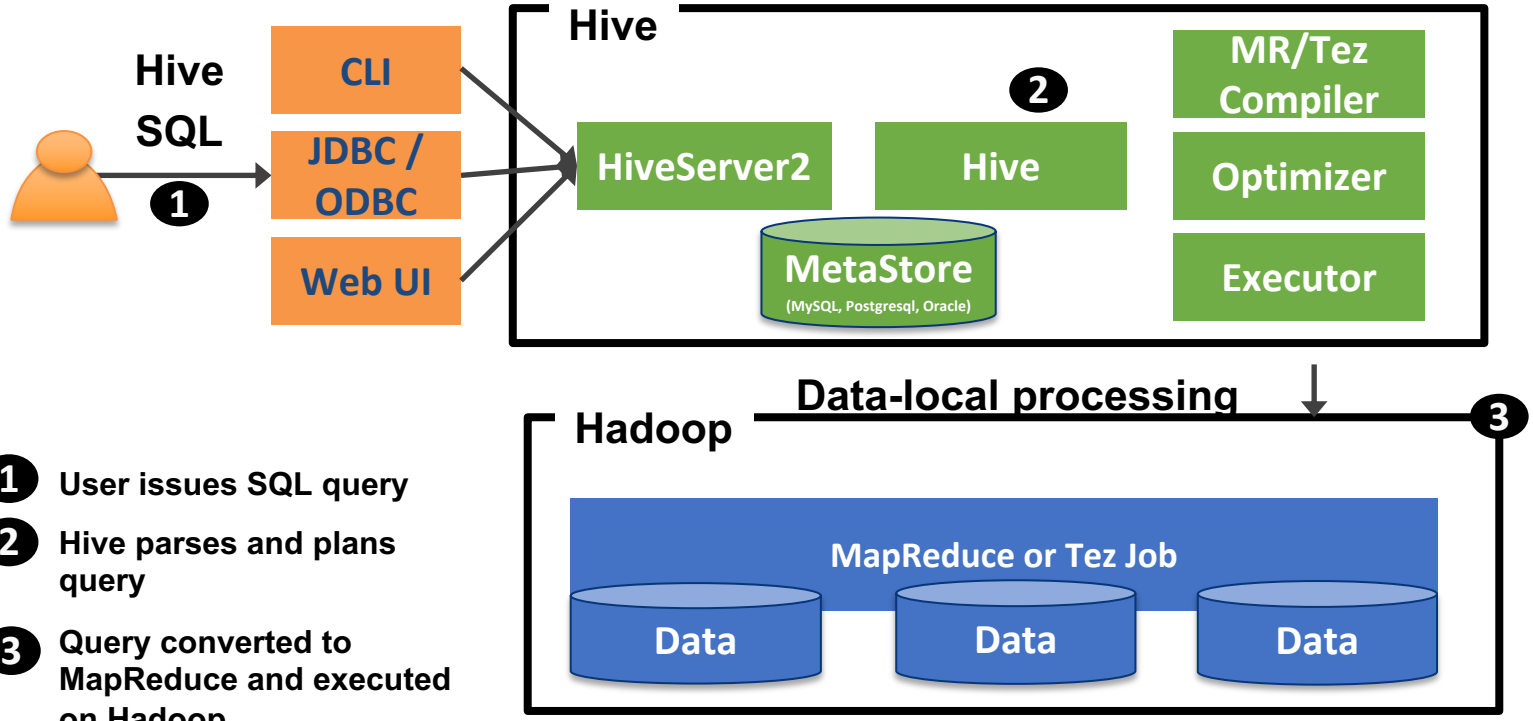
Source: Hortonworks

Hive

What is Hive?

- Data warehouse system for Hadoop
- Create schemas/table definitions that point to data in Hadoop
- Treat your data in Hadoop as tables
- SQL 92
- Interactive queries at scale

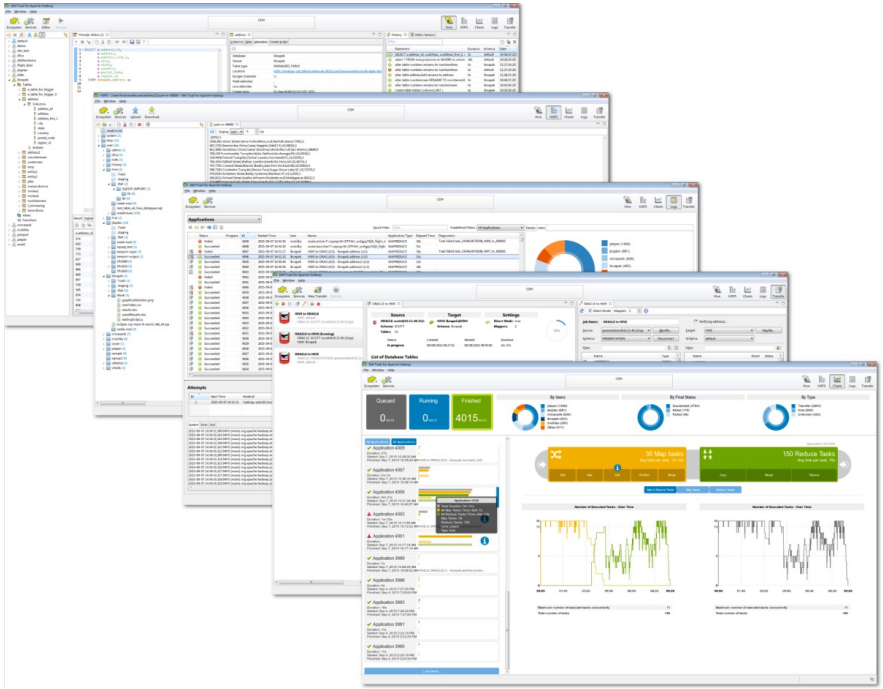
Hive Query Process



1. User issues SQL query
2. Hive parses and plans query
3. Query converted to MapReduce and executed on Hadoop

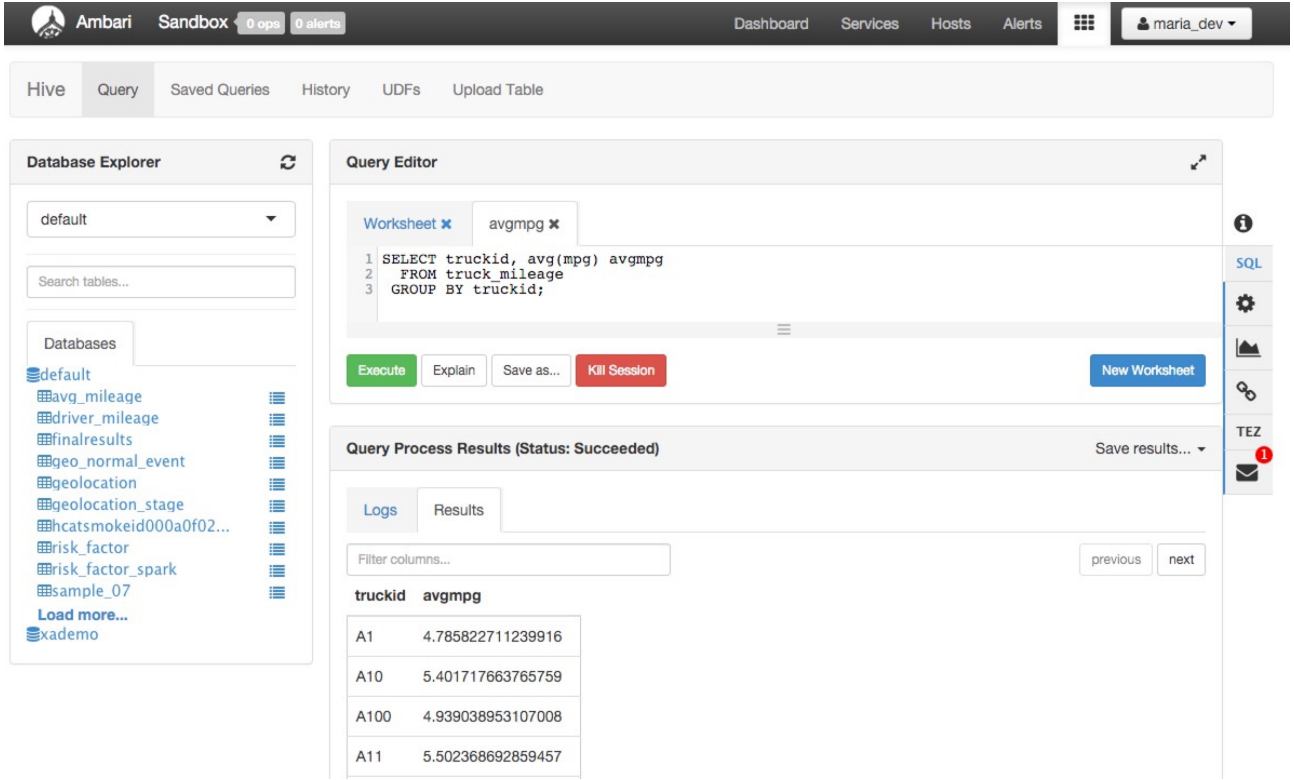
Submitting Hive Queries – CLI and GUI Tools

```
[it1@sandbox ~]$ beeline -u jdbc:hive2://localhost:10000
WARNING: Use "yarn jar" to launch YARN applications.
Connecting to jdbc:hive2://localhost:10000
Connected to: Apache Hive (version 1.2.1000.2.4.0.0-169)
Driver: Hive JDBC (version 1.2.1000.2.4.0.0-169)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 1.2.1000.2.4.0.0-169 by Apache Hive
0: jdbc:hive2://localhost:10000> show tables;
+-----------------------------------+--+
|             tab_name              |
+-----------------------------------+--+
| avg_mileage                       |
| driver_mileage                    |
| finalresults                      |
| geo_normal_event                  |
| geolocation                       |
| geolocation_stage                 |
| hcatsmokeid000a0f02_date250116    |
| risk_factor                       |
| risk_factor_spark                 |
| sample_07                         |
| sample_08                         |
| truck_mileage                     |
| trucks                            |
| trucks_stage                      |
+-----------------------------------+--+
14 rows selected (0.696 seconds)
0: jdbc:hive2://localhost:10000>
```



Source: Hortonworks

Submitting Hive Queries – Ambari Hive View



Defining a Hive-Managed Table

```
CREATE TABLE customer (
          customerID INT,
          firstName STRING,
          lastName STRING,
          birthday TIMESTAMP,
  ) ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ',';
```

Defining an External Table

```
CREATE EXTERNAL TABLE salaries (
    gender string,
    age int,
    salary double,
    zip int
 ) ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ',';
```

Defining a Table LOCATION

```
CREATE EXTERNAL TABLE SALARIES (
   gender string,
   age int,
   salary double,
   zip int
 ) ROW FORMAT DELIMITED
   FIELDS TERMINATED BY ','
   LOCATION '/user/train/salaries/';
```

Loading Data into Hive

```
LOAD DATA LOCAL INPATH '/tmp/customers.csv' OVERWRITE
INTO TABLE customers;
```

```
LOAD DATA INPATH '/user/train/customers.csv' OVERWRITE
INTO TABLE customers;
```

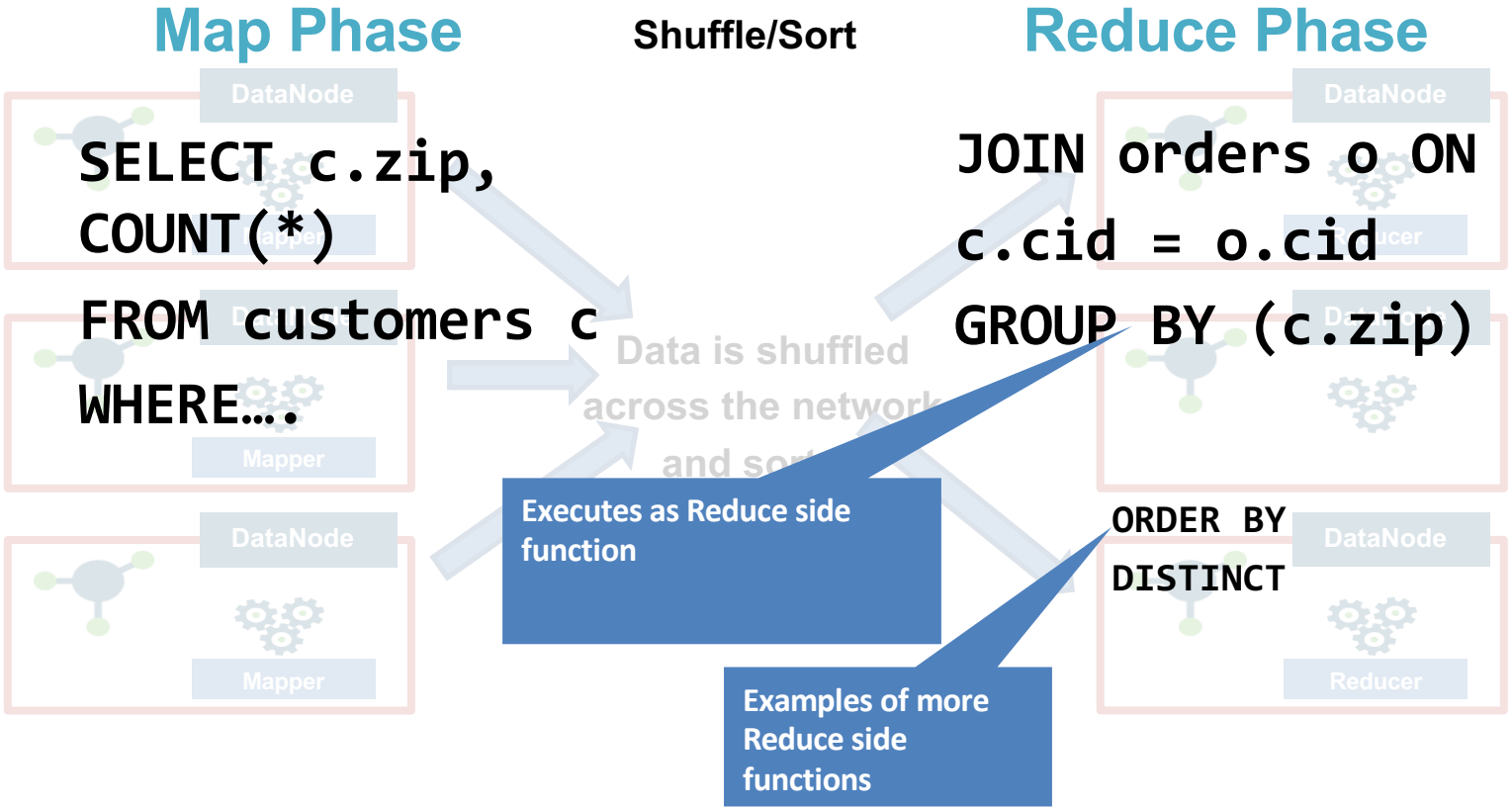
Performing Queries

```
SELECT * FROM customers;

SELECT firstName, lastName, address, zip
FROM customers
WHERE orderID > 0
GROUP BY zip;

SELECT customers.*, orders.*
FROM customers
JOIN orders ON
(customers.customerID = orders.customerID);
```

Internal Compilation to MapReduce



Views

```
CREATE VIEW 2010_visitors AS
  SELECT fname, lname,
     time_of_arrival, info_comment
  FROM wh_visits
  WHERE
  cast(substring(time_of_arrival,6,4)   AS int) >= 2010
  AND
  cast(substring(time_of_arrival,6,4)   AS int) < 2011;
```

Hive is NOT…

- … a relational database
  - Hive uses a database to store metadata, but the data that Hive processes is stored in HDFS
- … designed for online transaction processing
  - Hive runs on Hadoop (a batch-processing system where jobs can have high latency with substantial overhead)
- … suited for real-time queries and row-level updates
  - Hive is best used for batch jobs over large sets of immutable data (such as web logs)

Summary Hive

- Hive is the data warehouse system for Hadoop and uses the familiar table and SQL metaphors that are used with classic RDBMS solutions
- The MetaStore maintains the logical view of tables as well as the physical characteristics such as where the data is stored and in what format it is in
- Clients, using JDBC or ODBC, connect to the HiveServer2 component on a master node which in turn submits queries into the worker nodes for processing
- Hive can create, populate and query tables
- Views are supported, but they are not materialized
- Significant performance improvements have surfaced from the Stinger initiative including the use of the ORC file format and Tez as the execution engine

Source: Hortonworks