Association of Data Science and Analytics

Workshop: Introduction to Hive

Ken Taylor, Agrible
October 11, 2015

Topics

- Hive Intro
- Language
- Examples
- References

Hive

- Tool with SQL interface
- Came out of Facebook
- Used to handle large log data files
- hive.apache.org

Hive

- Fast implementation
- Ad-hoc queries
- Pour in the data, start a query
- SQL compiles into MapReduce jobs

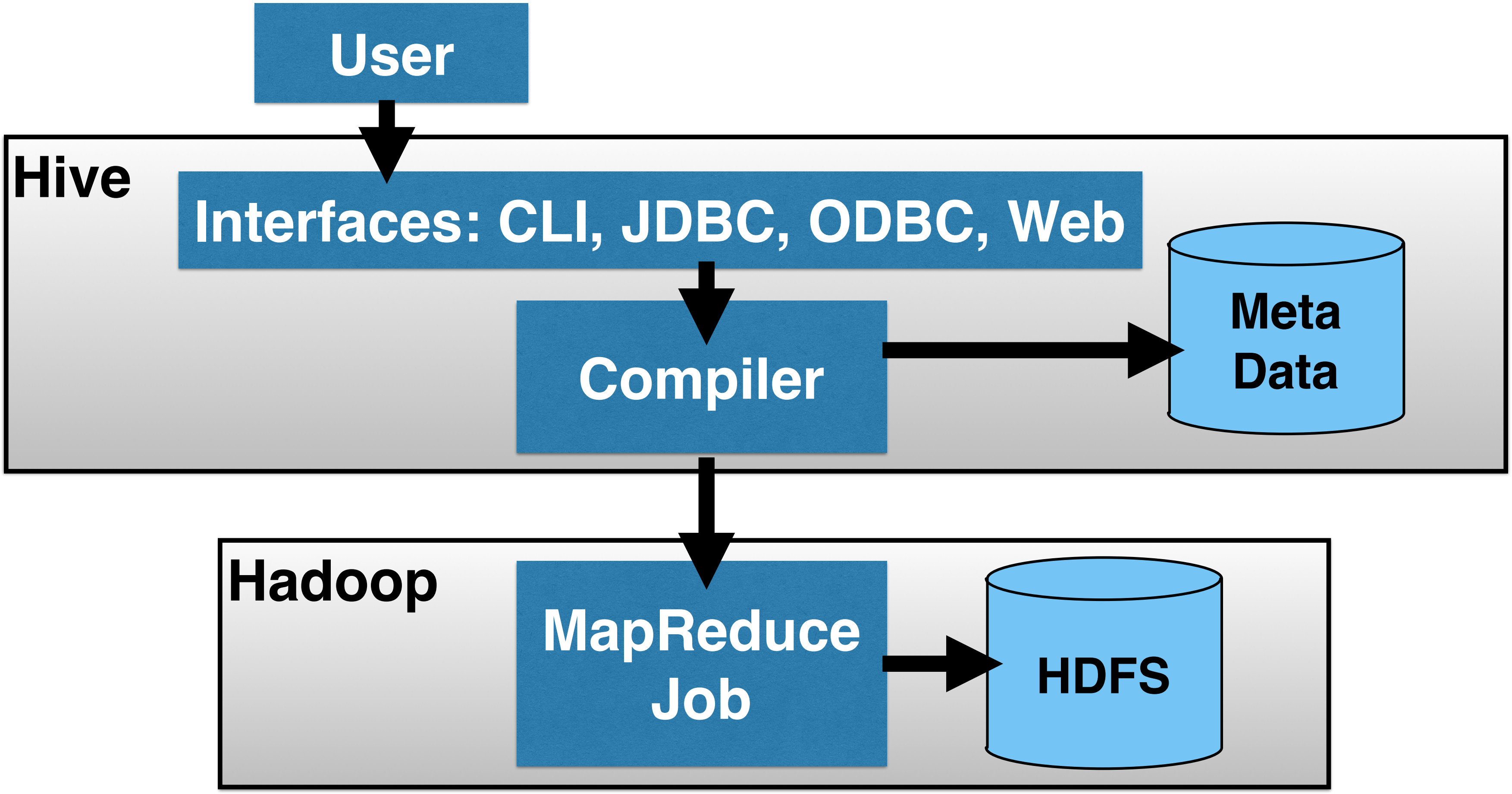
Hive

- Batch mode interaction
- Performance is not a requirement
- Minutes to generate results
- Scaling is required
- Good for large scale data analysis

Hive Use Cases

- Reporting
- Research
- Archive with occasional access
- Large scale data collection
- Not OLTP or performance environment

Hive


User
Hive
Interfaces: CLI, JDBC, ODBC, Web
Compiler
Meta Data
Hadoop
MapReduce Job
HDFS

Hive

- Metadata stores information about schema
- Compiler optimizes query performance
- HDFS is the data store

Hive

- HDFS stores read-only data files
- Inserts, Updates, Deletes now in Hive 0.14
  - Closer to full SQL compatibility
- Load temp results into new tables
  - Workflows: tableA ➔ tableB ➔ tableC

Hive

- Metadata is static
- Intermediate transformations are not stored, no reuse for future queries
- Temp tables only exist during the life of a job
- Save the temp table if you want to use it later

Hive

- Define Database
- Databases contain Tables
- Schema is an alias for database

Hive

- Language definition
- https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DDL

Create Database

```
CREATE (DATABASE|SCHEMA) [IF NOT
EXISTS] database_name
  [COMMENT database_comment]
  [LOCATION hdfs_path]
  [WITH DBPROPERTIES
  (property_name=property_value, ...)];
```

Drop Database

```
DROP (DATABASE|SCHEMA) [IF EXISTS]
database_name [RESTRICT|CASCADE];
```

Word Count Example In Hive

```
CREATE TABLE lines(line STRING);
LOAD DATA INPATH ’text1' OVERWRITE INTO
TABLE lines;
  SELECT count(*), word
    FROM lines
 LATERAL VIEW explode(split(text,'\s'))
         subView AS word
GROUP BY word
ORDER BY count DESC;
```

Word Count Example In Hive

```
CREATE TABLE lines(line STRING);
LOAD DATA INPATH ’text1' OVERWRITE INTO
TABLE lines;
  SELECT count(*), word
    FROM lines
 LATERAL VIEW explode(split(text,'\s'))
         subView AS word
GROUP BY word
ORDER BY count DESC;
```

Word Count Example In Hive

```
CREATE TABLE lines(line STRING);
LOAD DATA INPATH ’text1' OVERWRITE INTO
TABLE lines;
  SELECT count(*), word
    FROM lines
 LATERAL VIEW explode(split(text,'\s'))
         subView AS word
GROUP BY word
ORDER BY count DESC;
```

Word Count Example In Hive

```
CREATE TABLE lines(line STRING);
LOAD DATA INPATH ’text1' OVERWRITE INTO
TABLE lines;
  SELECT count(*), word
    FROM lines
 LATERAL VIEW explode(split(text,'\s'))
         subView AS word
GROUP BY word
ORDER BY count DESC;
```

Hive Data Types

- TINYINT, SMALLINT, INT, BIGINT
- FLOAT, DOUBLE, DECIMAL
- TIMESTAMP, DATE
- STRING, VARCHAR, CHAR
- BOOLEAN, BINART

Hive Data Types

- `ARRAY<data type>`
  - List of items that can be indexed
- `MAP<data type>`
  - Key:value pairs

Hive Data Types

- `STRUCT<col name : data type,…>`
  - C-struct or Java object
- `UNIONTYPE<data type1,data type2,…>`
  - Each record may be a different data type
  - Data types are indicated by first value

Complex Data Types

- Using complex data types allows data to be de-normalized
- Grouping more data together for faster access, more convenient access
- Avoids multiple table transactions

Sample Airline Flight Data

- Mixture of `INT` and `STRING` fields
- Fields delimited by commas
- We need to override the field delimiter

Delimiters

- Hive defaults to control characters to separate fields
- Fields: Ctrl-A `'\001'`
- Collections: Ctrl-B `'\002'`
- Maps: Ctrl-C `'\003'`

Table Definition

```
create table newtable(
   year int,
   month int,
   dayofmonth int,
   dayofweek int,
   deptime int,
   ...)
   ROW FORMAT DELIMITED FIELDS
   TERMINATED BY ',';
```

Schema

- Hive enforces schema at read time since tables are read-only
- Standard RDMS enforce schema at write time

Sample Data

- Commercial airline flight information
- Arrival (Arr)
- Computer Reservation System (CRS)
- Departure (Dep)
- National Aviation System (NAS)

Sample Data Header

- Year,Month,DayofMonth,DayOfWeek,DepTime,CRSDepTime,ArrTime,CRSArrTime,UniqueCarrier,FlightNum,TailNum,ActualElapsedTime,CRSElapsedTime,AirTime,ArrDelay,DepDelay,Origin,Dest,Distance,TaxiIn,TaxiOut,Cancelled,CancellationCode,Diverted,CarrierDelay,WeatherDelay,NASDelay,SecurityDelay,LateAircraftDelay

Sample Data

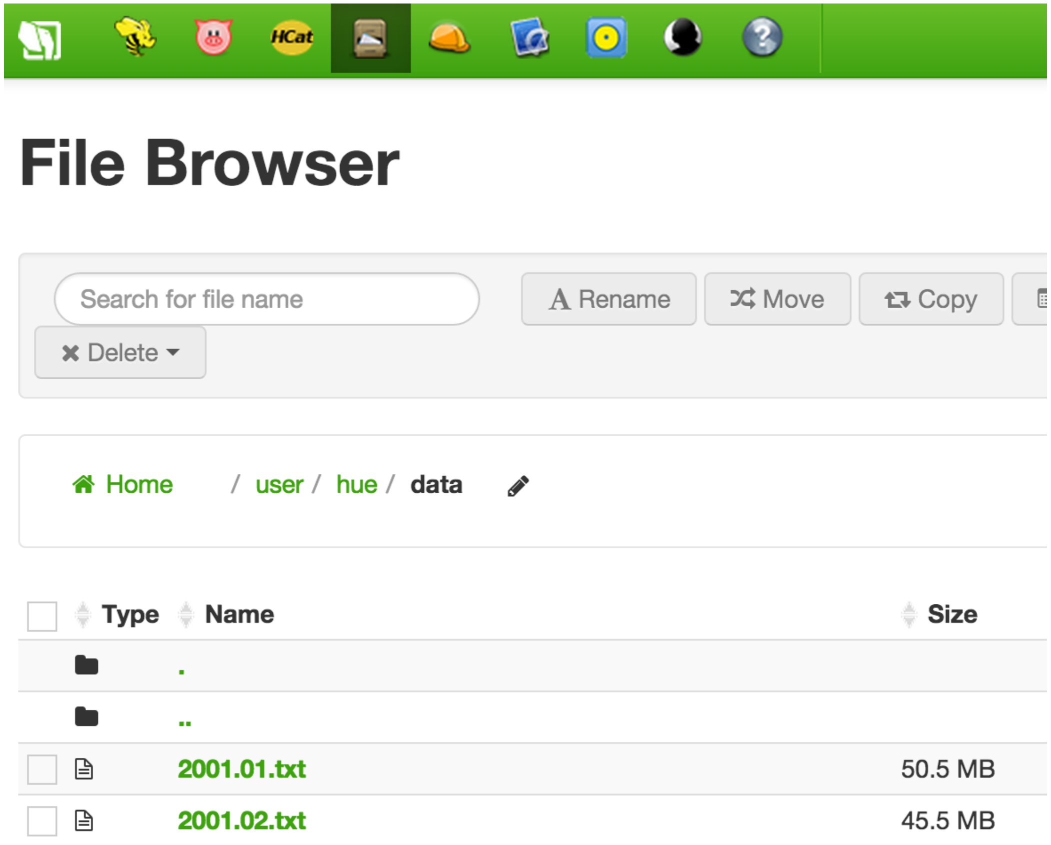
```
2001,1,17,3,1806,1810,1931,1934,US,
375,N700,85,84,60,-3,-4,BWI,CLT,
361,5,20,0,NA,0,NA,NA,NA,NA,NA

2001,1,18,4,1805,1810,1938,1934,US,
375,N713,93,84,64,4,-5,BWI,CLT,
361,9,20,0,NA,0,NA,NA,NA,NA,NA
```

Sample Data

- Create a directory
  - /user/hue/data
- Copy files into director
  - 2001.01.txt
  - 2001.02.txt

Sample Data



□	Type	Name	Size
	📁	.	
	📁	..	
□	📄	2001.01.txt	50.5 MB
□	📄	2001.02.txt	45.5 MB

Find All On-time Flights

```
select year,
       month,
       dayofmonth,
       flightnum,
       arrdelay
  from newtable
 where arrdelay < 1;
```

Find All On-time Flights

- Script
- On time flights - local data

Find Delayed Flights

```
select year,
       month,
       dayofmonth,
       flightnum,
       arrdelay
  from newtable
 where arrdelay > 300;
```

Find Delayed Flights

- Script
- Delayed flights - Local files

Data Location

- Hive will copy data files into default database
    - /apps/hive/warehouse
- Defined in config param file
    - /usr/hdp/current/hive-metastore/conf/conf.server/hive-site.xml

Local Data File

- To keep data files in local HDFS directory, use EXTERNAL table

```
CREATE EXTERNAL TABLE newtable(...)
ROW FORMAT DELIMITED FIELDS TERMINATED
BY ','
STORED AS TEXTFILE
LOCATION '/user/hue/data';
```

Local Data File

- Note that `/user/hue/data` is a directory, not a file
- `/user/hue/data` contains multiple files
  - 2001.01.csv
  - 2001.02.csv

Partitioning

- Problem: You need to collect log files from 1000 systems
- Collect one log file every hour from each system
- 24,000 log files a day
- 8,760,000 log files a year

Partioning

- Storing all log files together would degrade performance
- Need to avoid scanning 8 million data files for each query
- Need a better organization method

Partitioning

- Partitioning uses directory structure as a key to organize file

```
/data/year/month/day/hour/file1

/data/year/month/day/hour/file2
```

Partitioning

- The values of `year`, `month`, `day`, `hour` become keys, but are not stored in the file itself

```
/data/year/month/day/hour/file1

/data/year/month/day/hour/file2
```

Partitioning

- Each directory now contains only 1000 files, one per system per hour
- Query scans can be limited to 1000 files for a single hour

Partitioning

- Partitions can be added at the time of table definition
- Partitions can alter existing tables

Paritioning

```
CREATE TABLE logs(epoch BIGINT,
                  module STRING,
                  alert INT,
                  message STRING,)
COMMENT 'This is the log table'
PARTITIONED BY(year STRING,
               month STRING,
               day STRING,
               hour STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ‘;'
STORED AS TEXTFILE;
```

Functions

- round, floor, ceiling
- exponent, logarithm, trigonometric
- bitwise shiftleft, shiftright
- conv
- https://cwiki.apache.org/confluence/display/Hive/LanguageManual

Functions

- Order by
- Sort by, Cluster by, Distribute by (reducing)
- In, NOT In (similar to Case statement)
- Date functions
- String function
- Aggregate functions: sum, avg, min, max, stddev

Functions

- Group by
- Explode (similar to Flatten in Pig)
- User Defined Functions can be added

Joins

```
join_table:

  table_reference JOIN table_factor
     [join_condition]
| table_reference {LEFT|RIGHT|FULL} [OUTER] JOIN
     table_reference join_condition
| table_reference LEFT SEMI JOIN table_reference
     join_condition
| table_reference CROSS JOIN table_reference
     [join_condition]
```

Joins

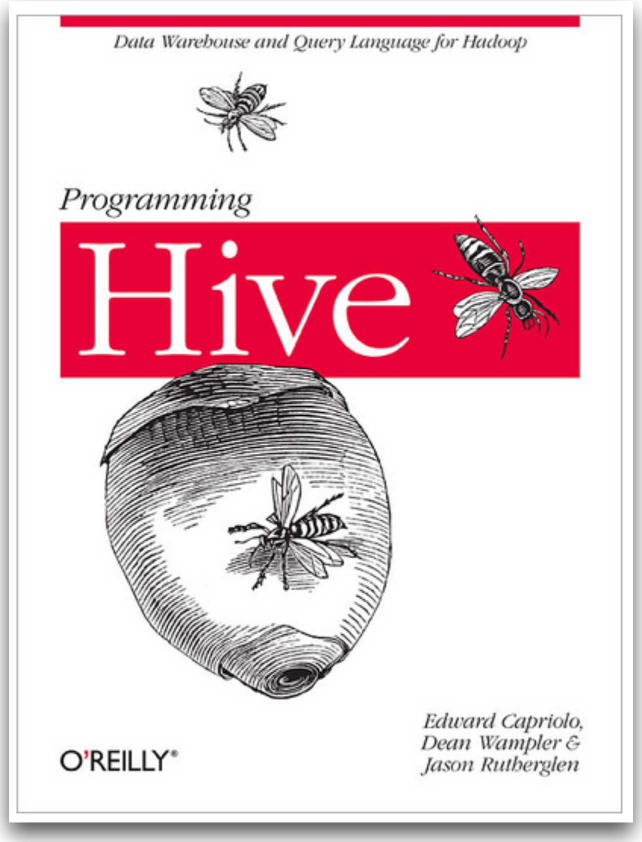
```
SELECT customers.ID,
       customers.NAME,
       orders.DATE,
       orders.AMOUNT,
       orders.ITEMS
  FROM CUSTOMERS
  LEFT OUTER JOIN ORDERS
    ON (customers.ID =
        orders.CUSTOMER_ID);
```

Joins

- Avoid large cartesian product joins
- Create smaller temp tables then join to improve performance
- If possible, consider de-normalizing data to avoid joins

References

- https://cwiki.apache.org/confluence/display/Hive/Home
- *Programming Hive*, by Edward Capriolo, Dean Wampler & Jason Rutherglen



Thank you

ken@agrible.com
LinkedIn: kentaylor7