

Lecture 12

Hadoop Hive

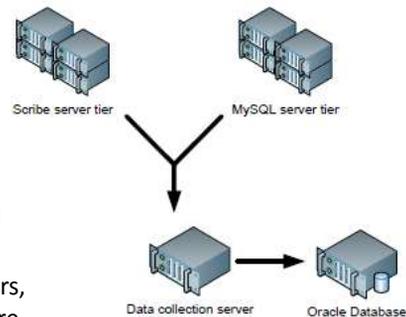
Zoran B. Djordjević
Csci e63 Big Data Analytics

@Zoran B. Djordjevic

1

Facebook 2006

- Started at Facebook.
- At that time, 2006, everything at Facebook was ran on MySQL.
- Facebook has a lot of Web Servers that produce a lot of logs with user related information that they wanted to extract.
- User data were held in farms of MySQL servers.
- All logs went through a locally developed log aggregation framework called Scribe.
- To run large scale reports: how many users, how many emails, who the messages were sent to, etc., a nightly cron job pushed data through an ETL process (Python scripts) into an Oracle database. Oracle spilled the reports.



@Zoran B. Djordjevic

2

Facebook Scales, 2007 > 2009 > 2011

- In 2006 Facebook produced several tens of GBs of data and schema worked well.
- In mid 2007, Facebook logs accumulated 1TB of data per day.
- In 2009 the volume grew to 10 TB per day.
- Facebook claims that Oracle solution was not scaling at all. (Actually, they forgot to call in Oracle Consulting Services :)
- Since Facebook was aware of Hadoop, they decided to push log data into Hadoop and try processing logs by MapReduce techniques.
- Facebook added 625 TB of compressed data during January 2011.
- In July 2011 Facebook Hadoop cluster had 30 PB of data.

@Zoran B. Djordjevic

3

Hadoop is an Enterprise Data Management System

- They started loading data from Scribe and MySQL data into Hadoop HDFS
- Facebook people started writing Hadoop MapReduce jobs to process data.
- To write MapReduce jobs you have to be a relatively sophisticated Java programmer. Hadoop was developed by geeks for geeks.
- It soon became apparent that some things were missing. Most notably:
 - Command-line interface for “end users” was not there.
 - End users are typically “marketing” people and they had no facilities to write queries on the fly. They had to bag real engineers to develop MapReduce programs and even run those programs for them.
 - The above broke the proper social hierarchy of Facebook Corp. Marketing people should never speak with geeks. In military they call that fraternization and can shoot you for it.
 - Information on data structures (schema) was not readily available.
 - Log data files have an implicit schema.
 - That schema is embedded in the code that knows how to read log files.
 - Schema of data is not readily visible.

@Zoran B. Djordjevic

4

Hive was Conceived

- In response to those challenges, Facebook developed Hive so that the “users” could perform:
 - Ad-hoc queries without writing full MapReduce jobs
 - Extract or create Schema information(Even Hive queries are still written and run by true engineers.)
- Hive could be used for
 - Log processing
 - Text mining
 - Document indexing
 - Customer-facing business intelligence (e.g., Google Analytics)
 - General Statistical Analysis, Predictive modeling, Hypothesis testing, etc.
- Hive has support for various aggregations and joins.
- Hive is considerably closer to standard SQL than PIG.
- Hive is more a data warehouse query language. PIG is more process oriented.

@Zoran B. Djordjevic

5

Hive Components

Hive has the following five major components:

- Shell, Driver, Compiler, Execution Engine and Meta Store

Shell

- A tool for user interaction. Allows interactive queries, just like DB shell connected to database. Supports web and JDBC clients.

Driver

- Management core of Hive engine. Driver manages query lifecycle, submits queries to the compiler, handles session, fetches the results and returns them to shell.

Compiler:

- Processing core that takes HQL language statements, parses them, creates query plan considering the schema of the database, selects optimal set of HDFS fetches, and other operations.

@Zoran B. Djordjevic

6

Hive Components

Execution engine:

- Directed Acyclic Graph of stages implemented as a set (tree) of
- Map/Reduce jobs, direct unfiltered fetches from HDFS and perhaps communications with Meta Store.

Meta Store:

- An actual relational database: Java Derby, MySQL, or any other.
- Meta Store contains schema of your tables,
- where in HDFS the table data are located, and
- a system called: SerDe (Serializer-Deserializer) which describes how to load data from HDFS or outside files and represent data as tables.

@Zoran B. Djordjevic

7

Hive vs. OLAP

- OLAP databases create cubes which are basically materialized views.
- OLAP cubes are not scalable to 500 machines.
- Hive has no automatically generated materialized views.
- If you know what your marketing users are looking for nothing in Hive prevents you from prefabricating data sets that would allow the “users” to quickly find what they are looking for.
- Hive could operate on clusters of 500 or 10,000 machines
- There is no solution that could work on 1TB and return results in minutes.
- If you have 100 GB of data you are better off loading data into an Oracle database. Let Oracle index everything and then run your queries.
- Once you are in TB range Hadoop or Hive are faster than Oracle or any other RDBMS.
- Hive is approximately as fast as Hadoop itself. Hive simplifies your work.
- Hadoop and Hive are not targeting small incremental changes of data sets.
- Hadoop and Hive are meant for global enterprises.

@Zoran B. Djordjevic

8

Data Model

- The Major benefit of Hive is that it could make unstructured data look like tables.
- Simply organized data like comma separated values naturally look like tables.
- In `CREATE EXTERNAL TABLE` command you just specify "delimited by ', ' and data will be loaded properly.
- You could also write elaborate serializers/deserializers that could read complex files and populate tables with data contained in those files.
- Hive is a database (warehouse) with strongly typed tables.
- Columns could have atomic types: `int`, `float`, `string`, `date`, `boolean`
- Composite types: `list`, `map` (associative array) or `struct` (convenient for JSON-like data).
- Elements of composite types could be any types, including composite types, meaning that types could be arbitrarily complex.

@Zoran B. Djordjevic

9

Hive Extends SQL

- Hive has various extensions of SQL. For example:
 - `EXPLODE` operator would take lists of data and create several columns with atomic data.
 - `COLAPSE` operator takes lists of data and pushes them into a single column of comma separated data.

@Zoran B. Djordjevic

10

Partitions

- You can break large tables by ranges of values in a column, for example by date.
- Date partitioned tables are stored in directories which have subdirectories with names stamped with date.
- You can make queries against individual partitions. Such queries are naturally much faster than queries over entire domain of partition column.
- Within partitions you have sub-partitions called Buckets
- Buckets are Hash partitions within ranges.
- Buckets are useful for sampling. For example: you can perform 5% of query on a valid sample.
- Buckets are also used by optimizer .

@Zoran B. Djordjevic

11

Meta Store

- Meta store does not reside in HDFS. Usually it is a Java Derby or MySQL database. Could use almost any other relational databases with a JDBC connector.
- Meta store uses derby by default;
- Meta store is a Database and:
 - Contains a namespace containing a set of tables
 - Holds table definitions (column types, physical layout (where in HDFS tables live as files, etc.))
 - Partitioning data (what are partition boundaries, etc.)
- Database storage location of Meta Store is determined by the hive configuration variable named `javax.jdo.option.ConnectionURL`.
- By default (see `conf/hive-site.xml`), this location is `./metastore_db`
- Right now, in the default configuration, this metadata can only be seen by one user at a time.

@Zoran B. Djordjevic

12

Meta Store

- The location and the type of the RDBMS are specified by `'javax.jdo.option.ConnectionURL'` and `'javax.jdo.option.ConnectionDriverName'`.
- Those are many other parameters of Hive are can be controlled by parameters in file `$HIVE_HOME/conf/hive-site.xml` and
- JDO (or JPOX) documentation provides more details on supported databases. The database schema is defined in JDO metadata annotations file `package.jdo` at `src/contrib/hive/metastore/src/model`.
- In the future, the metastore itself could be a standalone server.
- To run the metastore as a network server so it can be accessed from multiple nodes one should switch to `HiveDerbyServerMode`.

@Zoran B. Djordjevic

13

Physical Layout

- Warehouse directory is stored in HDFS e.g. `/home/hive/warehouse` or a similarly named directory
- Every table is stored in a subdirectory of `/home/hive/warehouse`
- Partitions, buckets form subdirectories of tables.
- Those files are under Hive control.
- Hive documentation suggests that you could backup those files by just making a copy to another directory or machine.
- Table data stored in
 - Flat Control char-delimited text files (`ctrl A` is the default delimiter)
 - SequenceFiles which are native to Hadoop.
 - With custom serializer-deserializers, called `SerDe`, files could use arbitrary data organization format

@Zoran B. Djordjevic

14

Installing Hive

- If installing Hive on a bare machine, you could go to www.apache.org and navigate to Hadoop > Hive.
- You need to have Hadoop installed and `HADOOP_HOME` environmental variable in your path.
- Hive could be downloaded from Facebook Subversion site:

```
svn co http://svn.apache.org/repos/asf/hadoop/hive/trunk hive
$ cd hive -- Run Ant
$ ant package
$ cd build/dist
$ ls
README.txt
bin/ (all the shell scripts)
lib/ (required jar files)
conf/ (configuration files)
examples/ (sample input and query files)
```

- You can still do this, even if you just want examples. I did it.

@Zoran B. Djordjevic

15

Cloudera VM

- You used to be able to go to www.cloudera.com and download a 64 bit VM with preconfigured versions of Hadoop and Hive.
- `cloudera-quickstart-vm-4.6.0-0-vmware.7z` or whatever was the name of the downloaded file has approximately 2GB and takes some time to download. You need utility 7-zip to open the archive.
- Once you open the archive, that is it. You can run the VM using VMWare VMPlayer or if you have VMWare Workstation installed, simply open the VM.
- Check the option that VMWare Tools are updated at Startup time.
- Once you launch the VM, log in with the following account details:
 - `username: cloudera`
 - `password: cloudera`
- Some of us will have a problem. Cloudera offer only 64 bit VMs and those require more than 4GB of RAM. If you have a 32bit OS or have less than 4GB of RAM on your machine, you are almost out of luck.

@Zoran B. Djordjevic

16

Where to run Hive, AWS

- If you still using older 32 bit VM, you will not be able to make the VM downloaded from Cloudera work on your machines.
- However, you are not out of luck. You can use AWS EMR machines.
- Go to Elastic MapReduce service, click "Create New Cluster". Check "Run your own application". As the type of job flow select "Hive Program".

@Zoran B. Djordjevic

17

Start an Interactive Hive Session

- Subsequently select number of instance (2 for initial test) and type (small).
- Select Amazon EC2 Key Pair

@Zoran B. Djordjevic

18

Add Key Pair, Enable Debugging

- Hive is very resilient and you might not need to debug anything. Just for the sake of an example, select Enable Debugging and provide a path to a folder in one of your S3 buckets.

Create a New Job Flow Cancel X

DETAILED JOB FLOW | SPECIFY PARAMETERS | CONFIGURE EC2 INSTANCES | **ADVANCED OPTIONS** | BOOTSTRAP ACTIONS | REVIEW

Here you enter advanced details about your job flow, such as an EC2 key pair, to use VPC, and your job flow debugging options.

Amazon EC2 Key Pair:
Use an existing key pair to SSH into the master node of the Amazon EC2 cluster as the user "hadoop".

Amazon VPC Subnet ID:
To run this job flow in a Virtual Private Cloud (VPC), select a subnet. See [Create a VPC](#).

Configure your logging options. [Learn more.](#)

Amazon S3 Log Path:
Optional: To save log files from the job flow to Amazon S3, specify an Amazon S3 bucket.

Enable Debugging: Yes No
Yes means EMR will store an index of your logs (requires an Amazon S3 Log Path).

Set advanced job flow options.

Keep Alive: Yes No You selected an interactive session; it requires manual termination.

Termination Protection: Yes No Yes prevents your nodes from shutting down due to accident or error.

Visible To All IAM Users: Yes No Yes means the job flow will be visible to all IAM users under your account.

[Back](#) **Create Job Flow** * Required field

@Zoran B. Djordjevic

19

Review your Job Setup

Create a New Job Flow Cancel X

DETAILED JOB FLOW | SPECIFY PARAMETERS | CONFIGURE EC2 INSTANCES | ADVANCED OPTIONS | BOOTSTRAP ACTIONS | **REVIEW**

Please review the details of your job flow and click "Create Job Flow" when you are ready to launch your Hadoop Cluster.

Job Flow Name: My Job Flow [Edit Job Flow Definition](#)

Type: Interactive Hive Session

Parameters: Interactive hive session has no parameters [Edit Job Flow Parameters](#)

Master Instance Type: m1.small **Instance Count:** 1

Core Instance Type: m1.small **Instance Count:** 2

[Edit EC2 Configs](#)

Amazon EC2 Key Pair: ec2_hu

Amazon Subnet ID:

Amazon S3 Log Path: s3://zoranlog/hive03162013

Enable Debugging: Yes **Keep Alive:** Yes

Termination Protected: No **Visible To All Users:** No

[Edit Advanced Options](#)

Bootstrap Actions: No Bootstrap Actions created for this Job Flow [Edit Bootstrap Actions](#)

[Back](#) **Create Job Flow**

Note: Once you click "Create Job Flow," instances will be launched and you will be charged accordingly.

- Click Create Job Flow

@Zoran B. Djordjevic

20

Wait until the job is RUNNING

The screenshot shows the AWS Elastic MapReduce console. At the top, there's a table of job flows:

Name	Status	Creation Date	Elapsed Time	Normalized Instance Hours
My Job Flow	RUNNING	2012-11-30 15:58 EST	0 hours 3 minutes	3
My Job Flow	TERMINATED	2012-11-29 16:41 EST	3 hours 35 minutes	3
hivejob	TERMINATED	2012-11-28 20:17 EST	3 hours 2 minutes	12
MyJob01	FAILED	2012-11-16 16:42 EST	0 hours 3 minutes	2

Below the table, the details for the selected 'My Job Flow' (Job Flow ID: j-1C2LFMQTHDZ4A) are shown. The 'Last State Change' is 'running step'. The 'Description' tab is active, showing the following details:

Description	Steps	Bootstrap Actions	Instance Groups	Monitoring
Name:	My Job Flow	Creation Date:	2012-11-30 15:08 EST	
Start Date:	2012-11-30 15:14 EST	End Date:	-	
Availability Zone:	us-east-1b	Instance Count:	-	
Master Instance Type:	-	Slave Instance Type:	-	
Key Name:	ec2_fu	Log URI:	s3n://zoran111@log/develop/	
Ami Version:	2.2.3	Master Public DNS Name:	ec2-107-71-101-218.compute-1.amazonaws.com	
Hadoop Version:	1.0.3	Keep Alive:	true	
Termination Protected:	false	Visible To All Users:	false	
Subnet ID:	-	Supported Products:	-	

Select DNS name of Hadoop master node

@Zoran B. Djordjevic

21

Alternatively go to EC2 Service

- Select Instances and then once your instances are running (green) identify the master. We have 2 slaves and 1 master. Connect to the master.

The screenshot shows the AWS Management Console 'INSTANCES' page. A table of instances is displayed:

Name	Instance	AMI ID	Root Device	Type	State	Status Checks	Alarm Status
empty	i-c9800e48	ami-5abc2933	instance store	m1.small	terminated		none
empty	i-c9800e4b	ami-5abc2933	instance store	m1.small	terminated		none
empty	i-45324625	ami-5abc2933	instance store	m1.small	terminated		none
empty	i-b2ac4f6a	ami-5abc2933	instance store	m1.small	running	2/2 checks p.	none
empty	i-b4a14f6c	ami-5abc2933	instance store	m1.small	running	2/2 checks p.	none
empty	i-f0e059a	ami-5abc2933	instance store	m1.small	running	2/2 checks p.	none

The 'EC2 Instance selected' details for the instance 'i-f0e059a' are shown below:

Description	Status Checks	Monitoring	Tags
AMI:	Amazon Elastic MapReduce 2013-02-21-03-43-50-82 pvm/s3 (ami-5abc2933)	Alarm Status:	none
Zone:	us-east-1b	Security Group:	ElasticMapReduce-master-sec-
Type:	m1.small	Status:	running
Scheduled Events:	No scheduled events	Owner:	951414139794

@Zoran B. Djordjevic

22

Right Click, Select Connect, Copy DNS

- Checkbox master instance, right click onto master, select Connect. On the wizard that pops-up highlight and copy Public DNS. Use the Public DNS to connect to the master through ssh on your Cygwin prompt

Wait for State to change to RUNNING

- Connect with `ssh` from your Cygwin terminal. Do not use root. Use hadoop user

```
$ ssh -i ec2_hu.pem hadoop@ec2-54-224-57-35.compute-1.amazonaws.com
```

```
The authenticity of host 'ec2-107-21-161-216.compute-1.amazonaws.com
(107.21.161.216)' can't be established.
```

```
RSA key fingerprint is f0:36:45:13:e5:9d:de:72:fd:d4:9d:06:6d:e2:58:8a.
```

```
Are you sure you want to continue connecting (yes/no)? yes
```

```
Warning: Permanently added 'ec2-107-21-161-216.compute-
1.amazonaws.com,107.21.161.216' (RSA) to the list of known hosts.
```

```
Linux (none) 2.6.35.11-83.9.amzn1.i686 #1 SMP Sat Feb 19 23:41:56 UTC 2011 i686
```

```
Welcome to Amazon Elastic MapReduce running Hadoop and Debian/Squeeze.
```

```
Hadoop is installed in /home/hadoop. Log files are in /mnt/var/log/hadoop. Check
/mnt/var/log/hadoop/steps for diagnosing step failures.
```

```
The Hadoop UI can be accessed via the following commands:
```

```
JobTracker      lynx http://localhost:9100/
```

```
NameNode       lynx http://localhost:9101/
```

```
hadoop@domU-12-31-39-0A-32-B0:~$ pwd
```

```
/home/hadoop
```

```
hadoop@domU-12-31-39-0A-32-B0:~$
```

Where to run Hive

- Either at an AWS Hadoop machine or on Cludera preconfigured VM you just type `hive` on a command prompt and Hive shell opens:
- `hadoop@domU-12-31-39-0A-32-B0:~$ hive`
- Logging initialized using configuration in `file:/home/hadoop/.versions/hive-0.8.1/conf/hive-log4j.properties`
- Hive history `file=/mnt/var/lib/hive_081/tmp/history/hive_job_log_hadoop_201211302021_1687633784.txt`
- `hive>`
- We might need one more terminal windows to run some prepackaged Hadoop jobs and produce standard sample data.
- Open another terminal window:
- Hadoop file systems should be empty;


```
$ hadoop fs -ls
```

@Zoran B. Djordjevic

25

Fetch Example Files and Queries

- Hive installation comes with a fair number of examples. On hadoop command prompt, type:


```
hadoop@domU-12-31-39-0E-14-28:~$ cd hive/example/files
hadoop@domU-12-31-39-0E-14-28:~/hive/examples/files$ ls
T1.txt          in2.txt      kv6.txt      srcbucket1.txt
T2.txt          in3.txt      lineitem.txt srcbucket20.txt
T3.txt          in4.txt      lt100.sorted.txt srcbucket21.txt
TestSerDe.jar  in5.txt      lt100.txt    srcbucket22.txt
apache.access.2.log in6.txt    lt100.txt.deflate srcbucket23.txt
hadoop@domU-12-31-39-0E-14-28:~/hive/examples/files$ cd ../queries
hadoop@domU-12-31-39-0E-14-28:~/hive/examples/queries$ ls
case_sensitivity.q input1.q input8.q join3.q sample3.q udf6.q
cast1.q input2.q input9.q join4.q sample4.q udf_case.q
groupby1.q input20.q input_part1.q join5.q sample5.q udf_when.q
groupby2.q input3.q input_testsequencefile.q join6.q sample6.q union.q
groupby3.q input4.q input_testxpath.q join7.q sample7.q
```
- Study them. They are excellent educational tools
- You can tar the examples directory and copy it to you computer using `scp` command


```
hadoop@..~/hive$ tar cvf ex.tar example # On EC2 Linux prompt
$ scp -i ec2_hu.pem hadoop@ec2-54-224-57-35.com..-1.amazonaws.com: home/hadoop/hive/ex.tar .
$ tar xvf ex.tar
```
- The last 2 commands are run on your Cygwin prompt, on your local machine

@Zoran B. Djordjevic

26

To Get Examples, Get Hive Release

- If you need hive code and examples, you could get the entire hive release. Go to: <http://hive.apache.org/releases.html>
- Select **Download a release now!**
- Select a Download Mirror
- Select `hive-0.12.0/` (Note, any version will work for our examples)
- Download `hive-0.12.tar.gz`
- Un-tar it in your Cygwin windows.

```
$ tar zxf hive-0.12.0.tar.gz # Note: z tells tar to run gzip filter first
[cloudera@localhost dist2]$ cd hive-0.12.0
[cloudera@localhost hive-0.12.0]$ ls
bin conf docs examples lib LICENSE NOTICE README.txt
[cloudera@localhost hive-0.12.0]$ cd examples
[cloudera@localhost examples]$ ls
files queries test-plugin
[cloudera@localhost examples]$ cd files
[cloudera@localhost files]$ ls
apache.access.2.log      in7.txt                  1t100.txt
srcbucket22.txt apache.access.log      in8.txt                  1t100.txt.deflate
```

@Zoran B. Djordjevic

27

Sample Data, Unpack Shakespeare

- On my machine in directory `~date` there are files: `shakespeare.tar.gz` and `bible.tar.gz` files.
- One contains complete works of Shakespeare and the other King James Bible. Both works use somewhat archaic form of English.
- We can unzip (`un tar`) both files. Command `$ tar zxf shakespeare.tar.gz`
- will un-tar Shakespeare's works and create directory `~/input`
- which contains a file `all-shakespeare` with all of Shakespeare works.
- You could examine the file by perhaps doing:

```
$ cat all-shakespeare | wc or
$ cat all-shakespeare | tail -n 100
```

@Zoran B. Djordjevic

28

Copy Shakespeare into HDFS

- We will copy local directory "input" into the HDFS directory input:

```
$hadoop fs -put input input
```
- We could convince ourselves that the data inside HDFS is still the same Shakespeare by typing something like:
 - ```
$ hadoop fs -cat input/all-shakespeare | head -n 20
```
  - ```
1 KING HENRY IV
```
 - ```
DRAMATIS PERSONAE
```
  - ```
KING HENRY      the Fourth. (KING HENRY IV:)
```

@Zoran B. Djordjevic

29

Unpack King James Bible

- We un-tar bible.tar.gz, what creates new local directory bible:

```
$ tar xzf bible.tar.gz
```
- We copy that directory to HDFS, as well

```
$hadoop fs -put bible bible
```
- If we now list files/directories in HDFS we will see both input and bible.

@Zoran B. Djordjevic

30

Running a MapReduce `grep` Job

- Hadoop example MapReduce scrips (jobs) are contained in `$HADOOP_HOME/hadoop-examples.jar` file. On Cloudera VM in: `/usr/lib/hadoop-0.20-mapreduce/hadoop-examples.jar`
- One of the scripts is a `grep` job which counts how many times every word appears in the analyzed corpus.
- In our case, Hadoop `grep` would scan the file (with all Shakespeare's works) placed in the specified (HDFS) directory "input" and create a tab delimited report named `shakespeare_freq`.
- Hadoop `grep` uses regular exp '`\w+`' to select all multi-character words.
- This `grep` is different from Unix (Linux) `grep`. Unix `grep` returns lines where a pattern appears. Hadoop `grep` counts word frequencies.
- The command to run Hadoop `grep` reads:

```
$ hadoop jar /usr/lib/hadoop-0.20-mapreduce/hadoop-examples.jar grep input/all-shakespeare shakespeare_freq '\w+'
```
- Job takes a few minutes. You could monitor progress of all map jobs and
- reduce jobs. The output is placed in HDFS directory `shakespeare_freq`

@Zoran B. Djordjevic

31

Examine Result of `grep`

- Examin the output in HDFS directory `shakespeare_freq` by typing:

```
$ cloudera@localhost:~/data/input$ hadoop fs -ls
Found 2 items
drwxr-xr-x - cloudera /user/cloudera/input
drwxr-xr-x - cloudera /user/cloudera/shakespeare_freq
$ hadoop fs -ls shakespeare_freq
Found 2 items
drwxr-xr-x - cloudera /user/cloudera/shakespeare_freq/_logs
-rw-r--r-- 1 cloudera supergroup 324840 /user/cloudera/shakespeare_freq/part-00000
cloudera@localhost:~/data/input$
```
- subdirectory `_logs` is full of logs that would mess up our future import.
- So we will remove it. Type: `$ hadoop fs -rmr shakespeare_freq/_logs`
- Remaining `part-00000` is a single HDFS file.

@Zoran B. Djordjevic

32

Examine Content of the output file

- We could also see partial content of the output file:

```
$ hadoop fs -cat shakespeare_freq/part-00000 | head -n 20
25848 the
23031 I
19671 and
18038 to
16700 of
14170 a
12702 you
11297 my
10797 in
6817 his
6773 be
6309 for
cat: Unable to write to output stream.
```

- These are frequency - word pairs, as expected.

@Zoran B. Djordjevic

33

Create Table to Accept grep data

- In preparation for import of Shakespeare frequency data we on hive prompt we create table shakespeare.

- Note, whenever you enter hive shell, type the following:

```
hive> add jar hive-contrib-0.10.0-cdh4.6.0.jar;
```

- Then, let us create the table

```
hive> create table shakespeare (freq INT, word STRING) ROW FORMAT
      DELIMITED FIELDS TERMINATED BY '\t' stored as textfile;
```

```
hive> show tables;
```

```
OK
```

```
shakespeare
```

```
Time taken: 8.268 seconds
```

```
hive> describe shakespeare;
```

```
OK
```

```
freq int
```

```
word string
```

```
Time taken: 1.253 seconds
```

```
hive>
```

- This created table shakespeare with out any data.

@Zoran B. Djordjevic

34

Load grep Data into `shakespeare` Table

- Remember, we must delete `_logs` directory first:


```
$ hadoop fs -rmr shakespeare_freq/_logs
Deleted hdfs://localhost:8022/user/cloudera/shakespeare_freq/_logs
```
- To load data we go back to the `hive` shell and type:


```
hive> LOAD DATA INPATH "shakespeare_freq" INTO TABLE shakespeare;
Loading data to table shakespeare
OK
Time taken: 0.213 seconds
```
- On the load command, Hive moved HDFS directory `shakespeare_freq` into its own HDFS directory. That directory is specified in `hive-site.xml` file


```
cloudera@localhost:~/hive/conf$ vi hive-site.xml
. . . .
<property>
  <name>hive.metastore.warehouse.dir</name>
  <value>/usr/hive/warehouse</value>
  <description>location of default database for the warehouse</description>
</property>
```
- Note again, the directory `/usr/hive/warehouse` is in HDFS, not on Linux OS.

@Zoran B. Djordjevic

35

Verify that `shakespeare` has `grep` Data

```
hive> select * from shakespeare limit 10;
OK
25848 the
23031 I
19671 and
18038 to
16700 of
14170 a
12702 you
11297 my
10797 in
8882 is
Time taken: 0.095 seconds
hive>
```

- This statement read from the table (actually as part of optimization, it read directly from the HDFS file) and presented us with the first 10 lines.
- This is the same data we saw previously.

@Zoran B. Djordjevic

36

More Advanced Query

- Slightly more advanced query would perhaps be this one:

```
hive> SELECT * FROM shakespeare
WHERE freq > 100 SORT BY freq ASC
LIMIT 10;
```

- Notice that for a large data set this is not an entirely trivial job.
- Data has to be sorted before we could see 10 rows of words that have frequency just above 100.
- Notice how hive reports on map-reduce job it is starting.
- If the job takes too long you are given the job id and the command that you could execute to tell Hadoop to kill the job:

```
Starting Job = job_201404021324_0005, Tracking URL =
  http://localhost:50030/jobdetails.jsp?jobid=job_201404021324_0005
Kill Command = /usr/lib/hadoop/bin/hadoop job -
  Dmapred.job.tracker=localhost:8021 -kill job_201404021324_0005
```

@Zoran B. Djordjevic

37

Even More Complex Query

- The “users”, linguists perhaps, would like to know the number of words which appear with the most common frequencies.

```
hive> SELECT freq, COUNT(1) AS f2
FROM shakespeare GROUP BY freq SORT BY f2 DESC LIMIT 10;
```

- OK
- 1 13426
- 2 4274
- 3 2342
- 4 1502
- 5 1111
- 6 873
- 7 656
- 8 598
- 9 474
- 10 381

- This tells us that there are 13426 words that appears only once.
- 4274 words appear twice. 2342 words appear three times, etc.
- SQL command with minor deviation: ORDER BY is replaced by SORT BY.

@Zoran B. Djordjevic

38

Zipf's Law

- **Rank (r)**: The numerical position of a word in a list sorted by decreasing frequency (f).
- Zipf (1949) "discovered" that:
- If probability of word of rank r is p_r and N is the total number of word occurrences:

$$f \cdot r = k \quad (\text{for constant } k)$$

$$p_r = \frac{f}{N} = \frac{A}{r} \quad \text{for corpus indep. const. } A \approx 0.1$$

@Zoran B. Djordjevic

39

Stop Word or the most Frequent Words

- Most frequent words are simple to find:

```
hive> select freq, word
      from shakespeare
      sort by freq desc limit 20;
```

OK

```
25848 the
23031 I
19671 and
18038 to
16700 of
14170 a
12702 you
11297 my
10797 in
. . . .
```

- Those words we call stop words. In Google-like analysis of relevance for text finding, we simply ignore stop word. When we create Tf-Idf weighted vectors we by rule do not include "stop words".

@Zoran B. Djordjevic

40

Zipf and Term Weighting

- Luhn (1958) suggested that both extremely common and extremely uncommon words were not very useful for indexing.

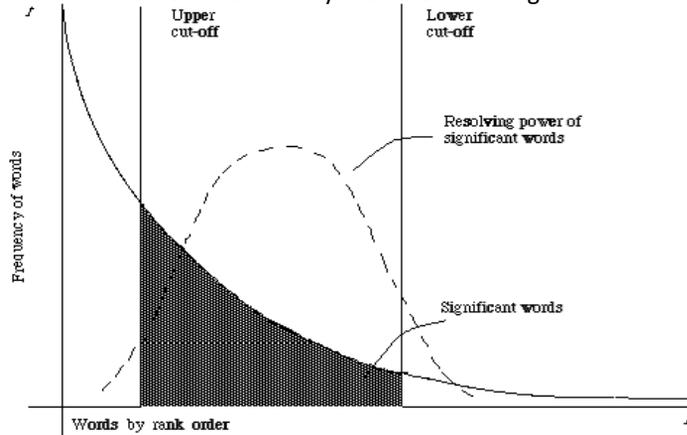


Figure 2.1. A plot of the hyperbolic curve relating f , the frequency of occurrence and r , the rank order (Adapted from Schultz⁴⁴, page 123)

@Zoran B. Djordjevic

41

How is Query Executed, Explain

- If we are curious how a query is executed we could use Explain command:

```
hive> EXPLAIN SELECT freq, COUNT(1) AS f2
FROM shakespeare GROUP BY freq
SORT BY f2 DESC LIMIT 10;
```

ABSTRACT SYNTAX TREE:

```
(TOK_QUERY (TOK_FROM (TOK_TABREF shakespeare)) (TOK_INSERT
(TOK_DESTINATION (TOK_DIR TOK_TMP_FILE)) (TOK_SELECT
(TOK_SELEXPR (TOK_TABLE_OR_COL freq)) (TOK_SELEXPR
(TOK_FUNCTION COUNT 1) f2)) (TOK_GROUPBY (TOK_TABLE_OR_COL
freq)) (TOK_SORTBY (TOK_TABSORTCOLNAMEDESC (TOK_TABLE_OR_COL
f2))) (TOK_LIMIT 10)))
```

STAGE DEPENDENCIES:

```
Stage-1 is a root stage
Stage-2 depends on stages: Stage-1
Stage-3 depends on stages: Stage-2
Stage-0 is a root stage
```

@Zoran B. Djordjevic

42

How is Query Executed, Explain

```

STAGE PLANS:
  Stage: Stage-1
    Map Reduce
      Alias -> Map Operator Tree:
        shakespeare
          TableScan
            alias: shakespeare
          Select Operator
            expressions:
              expr: freq
              type: int
            outputColumnNames: freq
          Reduce Output Operator
            key expressions:
              expr: freq
              type: int
            sort order: +

```

@Zoran B. Djordjevic

43

How is Query Executed, Explain

```

Map-reduce partition columns:
  expr: freq
  type: int
  tag: -1
  value expressions:
    expr: 1
    type: int
Reduce Operator Tree:
  Group By Operator
    aggregations:
      expr: count(VALUE._col0)
    keys:
      expr: KEY._col0
      type: int
    mode: complete

```

.....

@Zoran B. Djordjevic

44

Joining Tables

- One of the most powerful feature of Hive is the ability to create queries that joins tables together using regular SQL syntax.
- We have (freq, word) data for Shakespeare
- We could generate similar data for King James Bible and then examine which words show up in both volumes of text.
- To generate `grep` data for King James Bible we run Hadoop `grep` command:

```
$ hadoop jar $HADOOP_HOME/hadoop-examples.jar grep
    bible bible_freq '\w+'
```

- This will generate HDFS directory `bible_freq`

```
$ hadoop fs -ls
```

```
Found 4 items
```

```
drwxr-xr-x - cloudera supergroup          0 2010-05-02 16:38
/user/cloudera/bible
```

```
drwxr-xr-x - cloudera supergroup          0 2010-05-02 17:44
/user/cloudera/bible_freq
```

- Again we remove `_logs` directory.

```
$ hadoop fs -rmr bible_freq/_logs
```

@Zoran B. Djordjevic

45

Create KingJamesBible Table

```
hive> CREATE TABLE KingJamesBible (freq INT, word STRING)
ROW FORMAT DELIMITED
    FIELDS TERMINATED BY '\t' STORED AS TEXTFILE;
```

```
hive> show tables;
```

```
OK
```

```
kingjamesbible
```

```
shakespeare
```

```
Time taken: 0.165 seconds
```

```
hive> describe kingjamesbible;
```

```
OK
```

```
freq  int
```

```
word  string
```

```
Time taken: 0.228 seconds
```

```
hive>
```

@Zoran B. Djordjevic

46

Import data into KingJamesBible

```
hive> LOAD DATA INPATH "bible_freq" INTO TABLE KingJamesBible;
OK
Time taken: 0.781 seconds
hive> select * from kingjamesbible limit 20;
OK
62394  the
38985  and
34654  of
13526  to
12846  And
12603  that
12445  in
6913   be
6884   is
6649   him
6647   LORD
. . .
Time taken: 0.111 seconds
hive>
```

@Zoran B. Djordjevic

47

Examine bible_freq directory in HDFS

- Once you imported data into KingJamesBible table examine bible_freq directory in HDFS.

```
$ hadoop fs -ls bible_freq
```

- There is nothing there.
- Hive took part-00000 out and moved it somewhere else

@Zoran B. Djordjevic

48

Create an Intermediate Table

- We need a table that will list most common words in both volumes with corresponding frequencies

```
hive> CREATE TABLE merged
      (word STRING, shake_f INT, kjb_f INT);
```

- For this table we do not need to specify how will data be stored.
- Hive will determine that by itself.
- Next, we will run a query that will select data from tables: `shakespeare` and `kingjamesbible`, create a join and insert, i.e. overwrite the content of new table.
- In our case the table happens to be empty. If it were not empty and we insist on overwriting, table data would be lost. If we only perform an insert, new data would be appended to the old.

@Zoran B. Djordjevic

49

Populate merged table

```
hive> INSERT OVERWRITE TABLE merged
SELECT s.word, s.freq, k.freq FROM
shakespeare s JOIN kingjamesbible k ON
(s.word = k.word)
WHERE s.freq >= 1 AND k.freq >= 1;
....
Ended Job = job_201005021324_0013
Loading data to table merged
7826 Rows loaded to merged
hive> . . . .
A      2027    236
AND    102      5
AS     25       2
Aaron  26       350
Abel   2         16
Abhor  2         1
Abide  1         5
About  41        6
Above  25        3
Abraham 4      250
Time taken: 0.107 seconds
```

@Zoran B. Djordjevic

50

Most common common words

- What words appeared most frequently in both corpuses?

```
hive> SELECT word, shake_f, kjb_f, (shake_f + kjb_f) AS ss
FROM merged SORT BY ss DESC LIMIT 20;
the      25848    62394    88242
and      19671    38985    58656
of       16700    34654    51354
I        23031    8854     31885
to       18038    13526    31564
in       10797    12445    23242
a        14170    8057     22227
that     8869     12603    21472
And      7800     12846    20646
is       8882     6884     15766
my       11297    4135     15432
you      12702    2720     15422
he       5720     9672     15392
his      6817     8385     15202
not      8409     6591     15000
be       6773     6913     13686
for      6309     7270     13579
with     7284     6057     13341
it       7178     5917     13095
shall   3293     9764     13057
```

@Zoran B. Djordjevic

51

To examine common non-Stop Word, go deeper

```
SELECT word, shake_f, kjb_f, (shake_f + kjb_f) AS ss
FROM merged SORT BY ss DESC LIMIT 200;
```

```
. . . . .
heaven 626    578    1204
When   847    349    1196
Of 1006 63     1191
most   1017   135    1152
where  813    335    1148
tell   960    188    1148
blood  699    447    1146
doth   961    63     1146
set    451    694    1145
It 890  241    1131
ever   634    475    1109
Which 977    130    1107
whom   375    732    1107
Time taken: 46.988 seconds
```

@Zoran B. Djordjevic

52

Hive's DDL Operations, Create Table

- We already know how to create Hive tables and browse through them
hive> CREATE TABLE pokes (foo INT, bar STRING);
- Creates a table called `pokes` with two columns, the first being an integer and the other a string
hive> CREATE TABLE invites (foo INT, bar STRING)
PARTITIONED BY (ds STRING);
- Creates a table called `invites` with two columns and a partition column called `ds`.
- The partition column is a virtual column. It is not a part of the data itself but is derived from the partition that a particular dataset is loaded into.
- By default, tables are assumed to be of text input format and the delimiters are assumed to be `^A(ctrl-a)`.

@Zoran B. Djordjevic

53

Show Tables Command

- hive> SHOW TABLES '.*s';
- OK
- invites
- ip_locations
- pokes
- Show tables; command lists all the table that end with an 's'.
- The pattern matching follows Java regular expressions.

@Zoran B. Djordjevic

54

Alter Table Command

- As for altering tables, table names can be changed and additional columns can be dropped:

```
hive> ALTER TABLE pokes ADD COLUMNS (new_col INT);
hive> ALTER TABLE invites ADD COLUMNS (new_col2 INT COMMENT
'this is a comment');
hive> ALTER TABLE pokes RENAME TO happenings;
```

OK

Time taken: 0.17 seconds

```
hive> ALTER TABLE happenings RENAME TO pokes;
```

@Zoran B. Djordjevic

55

Running commands on OS

- From within a `hive` shell you could run outside (OS) commands by placing an exclamation mark in front of the command. Like:

```
hive> !pwd;
/home/cloudera/dist/hive-0.12.0-bin/examples/files
Hive> !ls;
union_input.txt
x.txt
y.txt
z.txt
```

@Zoran B. Djordjevic

56

DML Operations, Loading Data from Flat Files

- To use examples provided with Elastic Map Reduce AMI distribution go to:
/home/hadoop/hive/ directory

- Open hive shell

```
hive> LOAD DATA LOCAL INPATH './examples/files/kv1.txt' OVERWRITE
      INTO TABLE pokes;
```

Copying data from

```
file:/home/cloudera/hive/build/dist/examples/files/kv1.txt
```

Loading data to table pokes.

- Key word LOCAL forces Hive to load data from a local flat file.
- If the keyword is not present, Hive presumes that you are loading data from HDFS.
- The keyword 'overwrite' signifies that existing data in the table is deleted.
- If the 'overwrite' keyword is omitted, data files are appended to existing data sets.

@Zoran B. Djordjevic

57

Every Table corresponds to a Directory

```
[cloudera@localhost files]$ hadoop fs -ls /mnt/hive_081/warehouse
```

Found 4 items

```
drwxr-xr-x - cloudera supergroup          0 2012-11-30 06:51
/mnt/hive_081/warehouse/kingjamesbible
drwxr-xr-x - cloudera supergroup          0 2012-11-30 07:48
/mnt/hive_081/warehouse/merged
drwxr-xr-x - cloudera supergroup          0 2012-11-30 08:42
/mnt/hive_081/warehouse/pokes
drwxr-xr-x - cloudera supergroup          0 2012-11-30 06:50
/mnt/hive_081/warehouse/shakespeare
```

```
[cloudera@localhost files]$
```

```
[cloudera@localhost files]$ hadoop fs -ls
```

```
/mnt/hive_081/warehouse/kingjamesbible
```

Found 2 items

```
-rw-r--r--  1 cloudera supergroup          0 2012-11-30 06:42
/mnt/hive_081/warehouse/kingjamesbible/_SUCCESS
-rw-r--r--  1 cloudera supergroup    147408 2012-11-30 06:42
/mnt
/hive_081/warehouse/kingjamesbible/part-00000
```

- Data of a non-partitioned table are contained in a single file

@Zoran B. Djordjevic

58

Every Partition corresponds to a Directory

```
[..@localhost files]$ hadoop fs -ls /mnt/hive_081/warehouse/invites
Found 2 items
drwxr-xr-x  - cloudera supergroup          0 2012-11-30 09:17
/mnt/hive_081/warehouse/invites/ds=2008-08-08
drwxr-xr-x  - cloudera supergroup          0 2012-11-30 09:16
/mnt/hive_081/warehouse/invites/ds=2008-08-15
```

- Note that partitioned indexes are used as new directory names.
- Insider every “partition” directory there is a file containing data for that partition:

```
[..@..files]$ hadoop fs -ls /mnt/hive_081/warehouse/invites/ds=2008-08-08
Found 1 items
-rw-r--r--  1 cloudera supergroup          216 2012-11-30 09:17
/mnt/hive_081/warehouse/invites/ds=2008-08-08/kv3.txt
[..@..]hadoop fs -cat /mnt/hive_081/warehouse/invites/ds=2008-08-08/kv3.txt
213 val_213
146 val_146
406 val_406
```

@Zoran B. Djordjevic

59

Verify Presence of Data in pokes

```
hive> select * from pokes limit 20;
OK
238    val_238 NULL
86     val_86  NULL
311    val_311 NULL
27     val_27  NULL
165    val_165 NULL
409    val_409 NULL
255    val_255 NULL
278    val_278 NULL
98     val_98  NULL
484    val_484 NULL
265    val_265 NULL
193    val_193 NULL
401    val_401 NULL
150    val_150 NULL
273    val_273 NULL
224    val_224 NULL
369    val_369 NULL . . . .
Time taken: 0.272 seconds
hive>
```

NOTE:

- NO verification of data against the schema is performed by the load command.
- If the file is in HDFS, it is moved into the Hive-controlled file system namespace.
- The root of the Hive directory is specified by the option 'hive.metastore.warehouse.dir' in conf/hive-site.xml.
- If that directory is not there, the users should create this directory before trying to create tables via Hive.

@Zoran B. Djordjevic

60

Loading data into Partitioned Table

```
hive> LOAD DATA LOCAL INPATH './examples/files/kv2.txt'
      OVERWRITE INTO TABLE invites PARTITION (ds='2008-08-15');
hive> LOAD DATA LOCAL INPATH './examples/files/kv3.txt'
      OVERWRITE INTO TABLE invites PARTITION (ds='2008-08-08');
```

- The two LOAD statements above load data into two different partitions of the table invites.
- Table invites must be created as partitioned by the key ds for this to succeed.
- To verify that data is loaded run:

```
SELECT a.foo FROM invites a WHERE a.ds='2008-08-08';
```

- The statement selects column 'foo' from all rows of partition <2008-08-08> of invites table. The results are not stored anywhere, but are displayed on the console.
- **For fast access to data, partitioned tables should have a partition index selected in the WHERE clause of the statement.**

@Zoran B. Djordjevic

61

Without Partition the Query makes a Full Table Scan

```
hive> select count(*) from invites;
Starting Job = job_201211300612_0015, Tracking URL =
http://localhost:50030/jobdetails.jsp?jobid=job_201211300612_0015
Kill Command = /usr/lib/hadoop/bin/hadoop job -
Dmapred.job.tracker=localhost:8021 -kill job_201211300612_0015
Ended Job = job_201211300612_0015
OK
525
Time taken: 54.988 seconds
hive> select count(*) from invites where ds='2008-08-08';
25
Time taken: 15.792 seconds
hive> select count(*) from invites where ds='2008-08-15';
500
Time taken: 31.454 seconds
hive>
```

@Zoran B. Djordjevic

62

Exporting Data from a Table into an HDFS directory

- The following command will move data in table `invites` to HDFS directory `hdfs_out`

```
INSERT OVERWRITE DIRECTORY './hdfs_out' SELECT a.* FROM invites a
WHERE a.ds='2008-08-08';
Total MapReduce jobs = 2
Number of reduce tasks is set to 0 since there's no reduce operator
....
Ended Job = job_201005021324_0025
Moving data to: hdfs_out
25 Rows loaded to hdfs_out
OK
Time taken: 39.014 seconds
hive>
```

- Verify presence of the directory

```
$ hadoop fs -ls
Found 5 items . . . .
0 2010-05-02 18:32 /user/cloudera/hdfs_out
```

@Zoran B. Djordjevic

63

Exporting Data into a Local Directory

- You could as well send data into a local directory:

```
INSERT OVERWRITE LOCAL DIRECTORY './tmp/local_out' SELECT a.* FROM
pokes a;
cloudera@localhost files]$ cd /tmp/local_out
$ ls
000000_0
$ cat 000000_0
87 val_87 \N
364 val_364 \N
179 val_179 \N
118 val_118 \N
134 val_134 \N
395 val_395 \N
282 val_282 \N
138 val_138 \N
238 val_238 \N . . . .
```

@Zoran B. Djordjevic

64

GROUP BY Statements

```
hive> create table events (bar string, foo int);
```

- Note that the following statements are equivalent.

```
hive> FROM invites a INSERT OVERWRITE TABLE events SELECT a.bar,
count(*) WHERE a.foo > 0 GROUP BY a.bar;
```

```
hive> INSERT OVERWRITE TABLE events SELECT a.bar, count(1) FROM
invites a WHERE a.foo > 0 GROUP BY a.bar;
```

- Note that COUNT(*) does not work on older hive installations. You have to use COUNT(1) instead.
- You can use SUM, AVG, MIN, MAX operators on any column as well

```
INSERT OVERWRITE TABLE events SELECT a.bar, sum(a.foo) FROM
invites a WHERE a.foo > 0 GROUP BY a.bar;
```

- The following syntax works:

```
hive> FROM pokes t1 JOIN invites t2 ON (t1.bar = t2.bar) INSERT
OVERWRITE TABLE events SELECT t1.bar, t2.foo;
```

@Zoran B. Djordjevic

65

Multi-Table Insert

- Modified syntax, where query starts with a FROM clause has its benefits.
- Could you do this in your favorite RDBMS?

```
FROM src
INSERT OVERWRITE TABLE dest1 SELECT src.* WHERE src.key < 100
INSERT OVERWRITE TABLE dest2 SELECT src.key, src.value WHERE
src.key >= 100 and src.key < 200
INSERT OVERWRITE TABLE dest3 PARTITION(ds='2008-04-08', hr='12')
SELECT src.key WHERE src.key >= 200 and src.key < 300
INSERT OVERWRITE LOCAL DIRECTORY '/tmp/dest4.out' SELECT
src.value WHERE src.key >= 300;
```

- Apparently, this syntax allows you to perform inserts into several tables while visiting the original table only once. Since your table contains “big data”, Hive’s SQL engine has achieved a significant optimization.

@Zoran B. Djordjevic

66

Apache Weblog Data

- Regular expression serializer, deserializer `RegexSerDe` need to be loaded into Hive from `hive-contrib.jar`. The file is introduced into Hive by copying it to HDFS and then adding it to hive:

```
$hadoop fs -copyFromLocal hive/contrib/hive-contrib-0.8.1.jar \
hive/contrib/hive-contrib.jar
```

```
hive> add jar hive/contrib/hive_contrib.jar;
```

- For default Apache weblog, you can create a table with the following command

```
hive> CREATE TABLE apachelog (
host STRING,
identity STRING,
user STRING,
time STRING,
request STRING,
status STRING,
size STRING,
referer STRING,
agent STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.RegexSerDe'
WITH SERDEPROPERTIES ( "input.regex" = "([^ ]*) ([^ ]*) ([^ ]*) (-
|\\[[^\\]]*\\]) ([^ \\"]*|\"[^\"]*\") (-|[0-9]*) (-|[0-9]*) (?:(^[^
\"]*|\"[^\"]*\") ([^ \\"]*|\"[^\"]*\"))?)", "output.format.string" =
"%1$s %2$s %3$s %4$s %5$s %6$s %7$s %8$s %9$s" )
STORED AS TEXTFILE;
```

@Zoran B. Djordjevic

67

Insert data into apachelog

```
hive> load data local inpath
'./hive/examples/files/apache.access.2.log' into table
apachelog;
Copying data from
file:/home/hadoop/hive/examples/files/apache.access.2.log
hive> load data local inpath
'./hive/examples/files/apache.access.log' into table
apachelog;
Copying data from
file:/home/hadoop/hive/examples/files/apache.access.log
Hive> select * from apachelog;
127.0.0.1 - - [26/May/2009:00:00:00 +0000]
"GET /someurl/?t rack=Blabla(Main) HTTP/1.1"
200 5864 - "Mozilla/5.0 (Windows; U
; Windows NT 6.0; en-US) AppleWebKit/525.19 (KHTML, like Gecko)
Chrome/1.0.154.6 5 Safari/525.19"
127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700]
"GET /apache_pb.gif HTTP/1.0" 200 2326
NULL NULL
Time taken: 0.269 seconds
```

@Zoran B. Djordjevic

68

Test data in pig-apache log s3 folder

```
$ hadoop fs -ls s3n://elasticmapreduce/samples/pig-apache/input
Found 6 items
-rwxrwxrwx  1    8754118 2009-08-04 20:33 /samples/pig-
apache/input/access_log_1
-rwxrwxrwx  1    8902171 2009-08-04 20:33 /samples/pig-
apache/input/access_log_2
. . .
-rwxrwxrwx  1    8892828 2009-08-04 20:34 /samples/pig-
apache/input/access_log_6
```

- We will copy apache log data from the S3 bucket into local directory.

```
$ hadoop fs -copyToLocal \
    s3n://elasticmapreduce/samples/pig-apache/input .
$ ls input
access_log_1
access_log_2
. . .
access_log_6
```

@Zoran B. Djordjevic

69

Load Data

```
LOAD DATA LOCAL INPATH './input/access_log_1' into table apachelog;
Copying data from file:/home/hadoop/input/access_log_1
Loading data to table default.apachelog
OK
Time taken: 3.794 seconds
LOAD DATA LOCAL INPATH './input/access_log_2' into table apachelog;
SELECT * from apachelog;
46      "http://example.org/"      "Mozilla/5.0 (Macintosh; U; Intel Mac OS X
10_5_7; en-us) AppleWebKit/530.17 (KHTML, like Gecko) Version/4.0
Safari/530.17"
66.249.67.3      -      -      [20/Jul/2009:20:13:21 -0700]      "GET
/gallery/main.php?g2_controller=exif.SwitchDetailMode&g2_mode=detailed&g2_retur
n=%2Fgallery%2Fmain.php%3Fg2_itemId%3D30893&g2_returnName=photo HTTP/1.1"
302      5      "-"      "Mozilla/5.0 (compatible; Googlebot/2.1;
+http://www.google.com/bot.html)"
66.249.67.3      -      -      [20/Jul/2009:20:13:24 -0700]      "GET
/gallery/main.php?g2_itemId=30893&g2_fromNavId=xfc647d65 HTTP/1.1"      200
8196      "-"      "Mozilla/5.0 (compatible; Googlebot/2.1;
+http://www.google.com/bot.html)"
```

@Zoran B. Djordjevic

70

Impression Logs

- Let us look at impressions logs, first. On remote Linux prompt we enter:

```
$hadoop fs -ls s3://elasticmapreduce/samples/hive-ads/tables/impressions
drwxrwxrwx - 0 1970-01-01 00:00 /samples/hive-ads/tables/impressions/dt=2009-04-14-12-15
drwxrwxrwx - 0 1970-01-01 00:00 /samples/hive-ads/tables/impressions/dt=2009-04-14-12-20
drwxrwxrwx - 0 1970-01-01 00:00 /samples/hive-ads/tables/impressions/dt=2009-04-14-13-00
$hadoop fs -ls s3://elasticmapreduce/samples/hive-ads/tables/impressions/dt=2009-04-14-13-00
-rwxrwxrwx 1 31733 2009-10-01 01:09 /samples/hive-ads/tables/impressions/dt=2009-04-14-13-00/ec2-65-23-84-97.amazon.com-2009-04-14-13-00.log
-rwxrwxrwx 1 32369 2009-10-01 01:09 /samples/hive-ads/tables/impressions/dt=2009-04-14-13-00/ec2-93-80-66-68.amazon.com-2009-04-14-13-00.log
hadoop fs -cat s3://elasticmapreduce/samples/hive-ads/tables/impressions/dt=2009-04-14-13-00/ec2-93-80-66-68.amazon.com-2009-04-14-13-00.log
{"number": "945155", "referrer": "naver.com", "processId": "1739", "adId":
"AJ93SFCDQ6IKa7BdRhn4CMmAKhAGhT", "browserCookie": "gtxxljhucu", "userCookie":
"D7D0xTR8E3ESjL36n3h4bvErdB83hV", "requestEndTime": "1239714024000", "impressionId":
"6nhxSxUMHCQF4bqIhqEaG3wV9cbVsG", "userAgent": "Mozilla/4.0 (compatible; MSIE 7.0; Windows
NT 5.1; InfoPath.1)", "timers": {"modelLookup": "0.394", "requestTime": "0.8062"}, "threadId": "51",
"ip": "39.204.178.7", "modelId": "bxxiuxduad", "hostname": "ec2-93-80-66-68.amazon.com",
"sessionId": "le4T4IVVQPssc1GDFOQxNqAiEg9Fn", "requestBeginTime": "1239714023000"}
```

@Zoran B. Djordjevic

71

Add Jason Serializer-Deserializer

- Impression and Click logs are collections of Jason structures.
- Complex data organizations require specific serializers & deserializers (SERDE)
- A few come with Hive distribution. On `hive>` prompt we load Jason SERDE.

```
add jar s3://elasticmapreduce/samples/hive-ads/libs/jsonserde.jar;
```

- With SERDE in place, we can read the data and push them into a table.
- We will actually treat data as an EXTERNAL TABLE

@Zoran B. Djordjevic

72

External Table

```
create external table if not exists impressions (
  requestBeginTime STRING,
  adId STRING,
  impressionId STRING,
  referrer STRING,
  userAgent STRING,
  userCookie STRING,
  ip STRING)
partitioned by (dt STRING)
ROW FORMAT
SERDE 'com.amazon.elasticmapreduce.JsonSerde'
WITH SERDEPROPERTIES
('paths'='requestBeginTime,adId,impressionId,
referrer,userAgent, userCookie, ip')
LOCATION 's3://elasticmapreduce/samples/hive-
ads/tables/impressions';
```

@Zoran B. Djordjevic

73

Describe impressions

```
hive> desc impressions;
OK
requestbeginTime      string  from deserializer
adid                  string  from deserializer
impressionid          string  from deserializer
referrer              string  from deserializer
useragent             string  from deserializer
usercookie            string  from deserializer
ip                    string  from deserializer
dt                    string
Time taken: 0.918 seconds
hive>
show partitions impressions;
alter table impressions recover partitions;
```

@Zoran B. Djordjevic

74

Session History

- Hive writes session history to files it stores in the local directory:

```
$ls /mnt/var/lib/hive_081/tmp/history/
hive_job_log_hadoop_201303012359_200507540.txt
hive_job_log_hadoop_201303020005_660836981.txt
hive_job_log_hadoop_201303020026_176357633.txt
hive_job_log_hadoop_201303020030_1224796588.txt
hive_job_log_hadoop_201303020036_281675702.txt
hive_job_log_hadoop_201303020459_688140619.txt
hive_job_log_hadoop_201303020511_511897674.txt
$ vi
/mnt/var/lib/hive_081/tmp/history/hive_job_log_hadoop_20130301235
9_200507540.txt
```

@Zoran B. Djordjevic

75

hive_job_log_hadoop_201303012359_200507540.txt

```
SessionStart SESSION_ID="hadoop_201303012359"
TIME="1362182359726"
QueryStart QUERY_STRING="create table shakespeare (freq INT, word
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
stored as textfile" QUERY_ID="hadoop_20130301235959_3456b91d-
e074-4f05-a39e-b3c51e183164" TIME="1362182369190"
Counters plan="{\"queryId\":\"hadoop_20130301235959_3456b91d-e074-
4f05-a39e-
b3c51e183164\",\"queryType\":null,\"queryAttributes\":{\"queryString\":
create table shakespeare (freq INT, word STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '\t' stored as
textfile\"},\"queryCounters\":null,\"stageGraph\":{\"nodeType\":\"STAGE
\",\"roots\":null,\"adjacencyList\":null\"},\"stageList\":[{\"stageId\":\"St
age-
0\",\"stageType\":\"DDL\",\"stageAttributes\":null,\"stageCounters\":}
\",\"taskList\":[{\"taskId\":\"Stage-
0_OTHER\",\"taskType\":\"OTHER\",\"taskAttributes\":null,\"taskCounters
\":null,\"operatorGraph\":null,\"operatorList\":null,\"done\":false
\",\"started\":false\"}],\"done\":false,\"started\":false\"}],\"done\":f
alse\",\"started\":true\"}" TIME="1362182369220"
TaskStart TASK_NAME="org.apache.hadoop.hive ql.exec.DDLTask"
TASK_ID="Stage-0" QUERY_ID="hadoop_20130301235959_3456b91d-e074-
4f05-a39e-b3c51e183164" TIME="1362182369223"
```

@Zoran B. Djordjevic

76

References

- <http://wiki.apache.org/hadoop/Hive/GettingStarted>
- http://www.cloudera.com/videos/introduction_to_hive
- http://www.cloudera.com/videos/hive_tutorial
- <http://issues.apache.org/jira/browse/HIVE-662>

i