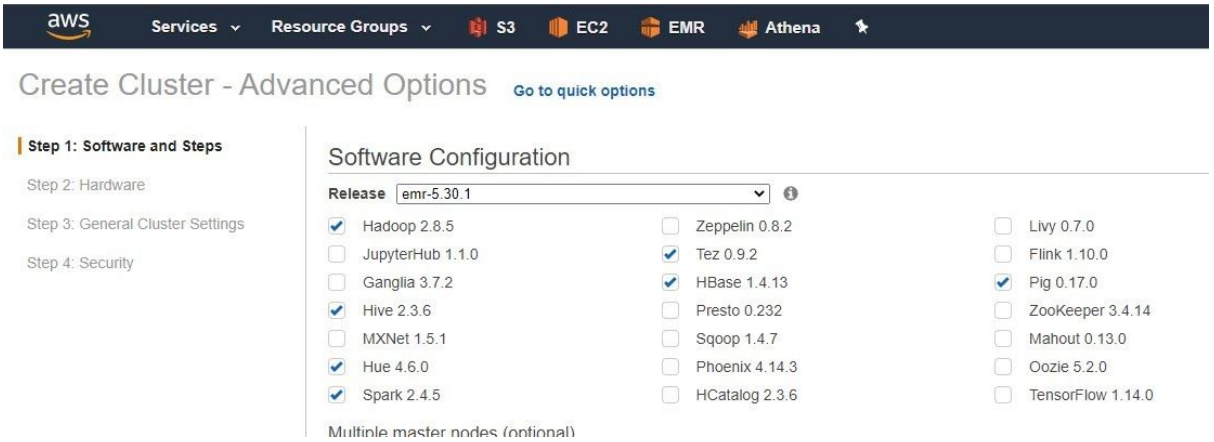
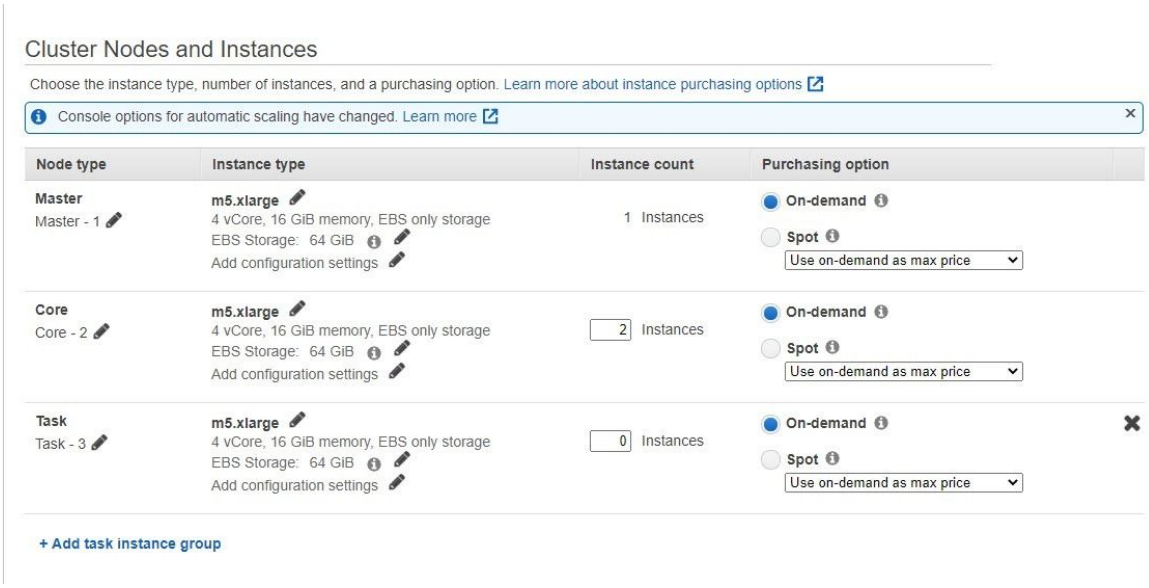
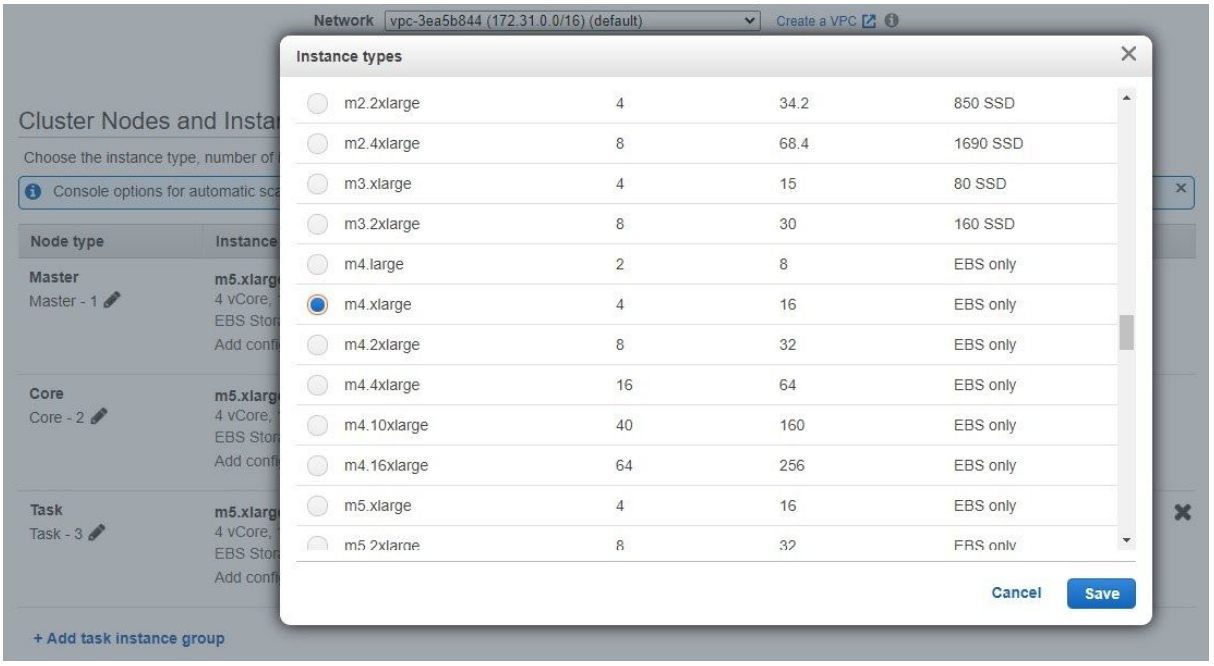
PLEASE HAVE IN MIND THAT THIS FILE WILL BE UPDATED.
THIS IS JUST QUICK RECAP WHAT THE APP DOES AND HOW IT IS WORKING FOR NOW.

Create AWS basic environment

1. Create cluster







Cluster Nodes and Instances

Choose the instance type, number of instances, and a purchasing option. Learn more about instance purchasing options

Console options for automatic scaling have changed. Learn more

Node type	Instance type	Instance count	Purchasing option
Master Master - 1	**m4.xlarge** 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB Add configuration settings	1 Instances	On-demand Spot Use on-demand as max price
Core Core - 2	**m4.xlarge** 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB Add configuration settings	2 Instances	On-demand Spot Use on-demand as max price
Task Task - 3	**m4.xlarge** 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB Add configuration settings	0 Instances	On-demand Spot Use on-demand as max price

Create Cluster - Advanced Options

Go to quick options

- Step 1: Software and Steps
- Step 2: Hardware
- Step 3: General Cluster Settings
- **Step 4: Security**

Security Options

EC2 key pair hadoopKey

Cluster visible to all IAM users in account

Permissions

Default Custom

Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.

EMR role EMR_DefaultRole

EC2 instance profile EMR_EC2_DefaultRole

Auto Scaling role EMR_AutoScaling_DefaultRole

▸ Security Configuration

▸ EC2 security groups

Cancel Previous Create cluster

2. Modify Security Groups

Clone | Terminate | AWS CLI export

Cluster: Hadoop_Hive_2020-09-17 Starting Configuring cluster software

Summary | Application user interfaces | Monitoring | Hardware | Configurations | Events | Steps | Bootstrap actions

Summary

ID: j-3O9N9MODM8W9N
Creation date: 2020-09-17 23:00 (UTC+2)
Elapsed time: 6 minutes
After last step completes: Cluster waits
Termination protection: On Change
Tags: -- View All / Edit
Master public DNS: ec2-3-95-187-130.compute-1.amazonaws.com
Connect to the Master Node Using SSH

Configuration details

Release label: emr-5.30.1
Hadoop distribution: Amazon 2.8.5
Applications: Hive 2.3.6, Pig 0.17.0, Hue 4.6.0, Spark 2.4.5, HBase 1.4.13, Tez 0.9.2
Log URI: s3://aws-logs-438477492770-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

Application user interfaces

Persistent user interfaces: --
On-cluster user interfaces: Not Enabled Enable an SSH Connection

Network and hardware

Availability zone: us-east-1d
Subnet ID: subnet-172ce336
Master: Bootstrapping 1 m4.xlarge
Core: Provisioning 2 m4.xlarge
Task: --
Cluster scaling: Not enabled

Security and access

Key name: hadoopKey
EC2 instance profile: EMR_EC2_DefaultRole
EMR role: EMR_DefaultRole
Auto Scaling role: EMR_AutoScaling_DefaultRole
Visible to all users: All Change
Security groups for Master: sg-01abbeeaafcde5be7 (ElasticMapReduce-master)
Security groups for Core & Task: sg-082c4611180041d01 (ElasticMapReduce-slave)

EC2 > Security Groups > sg-01abbeeaafcde5be7 - ElasticMapReduce-master > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
All TCP	TCP	0 - 65535	Custom	sg-01abbeeaafcde5be7		Delete
All TCP	TCP	0 - 65535	Custom	sg-082c4611180041d01		Delete
All traffic	All	All	My IP	185.93.94.32/32		Delete
SSH	TCP	22	Custom	93.174.24.146/32		Delete
Custom TCP	TCP	8443	Custom			Delete

EC2 > Security Groups > sg-082c4611180041d01 - ElasticMapReduce-slave > Edit inbound rules

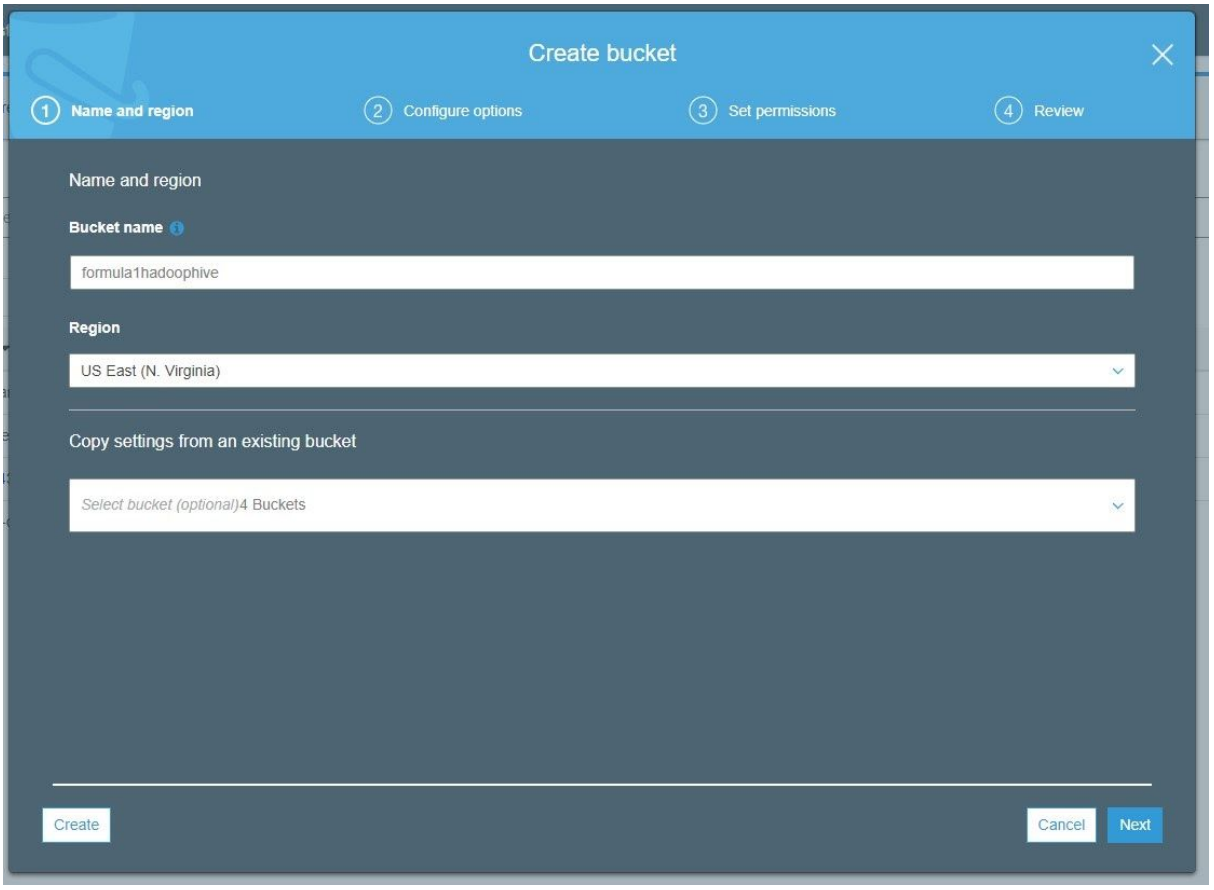
Edit inbound rules Info

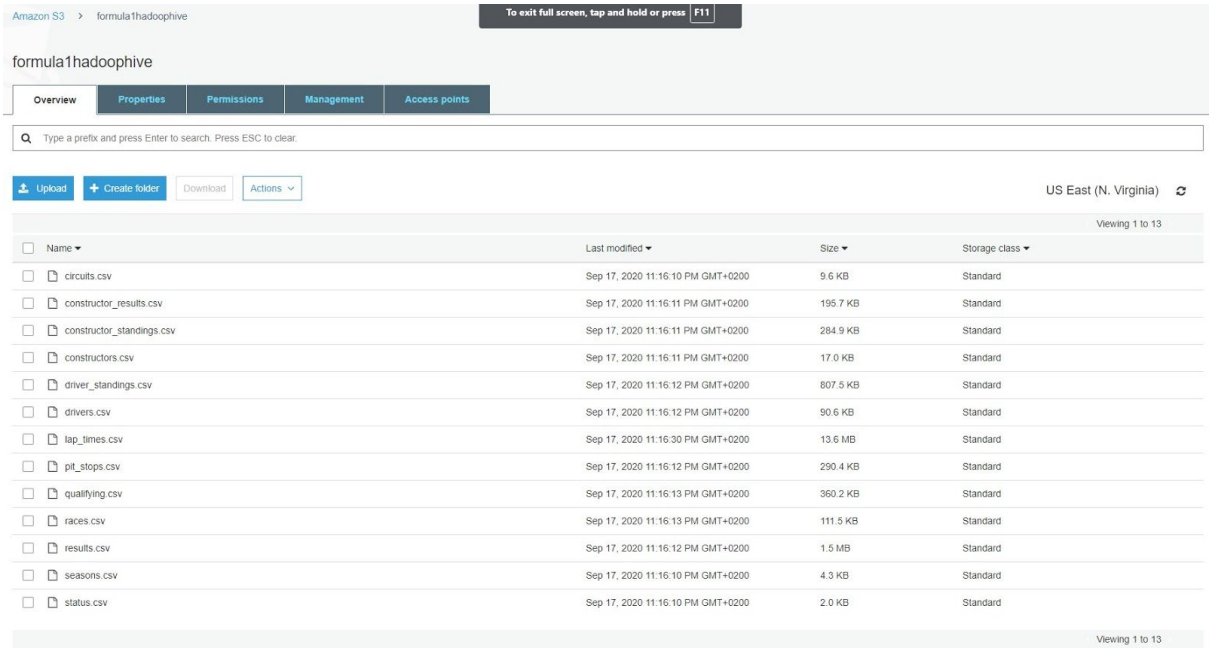
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

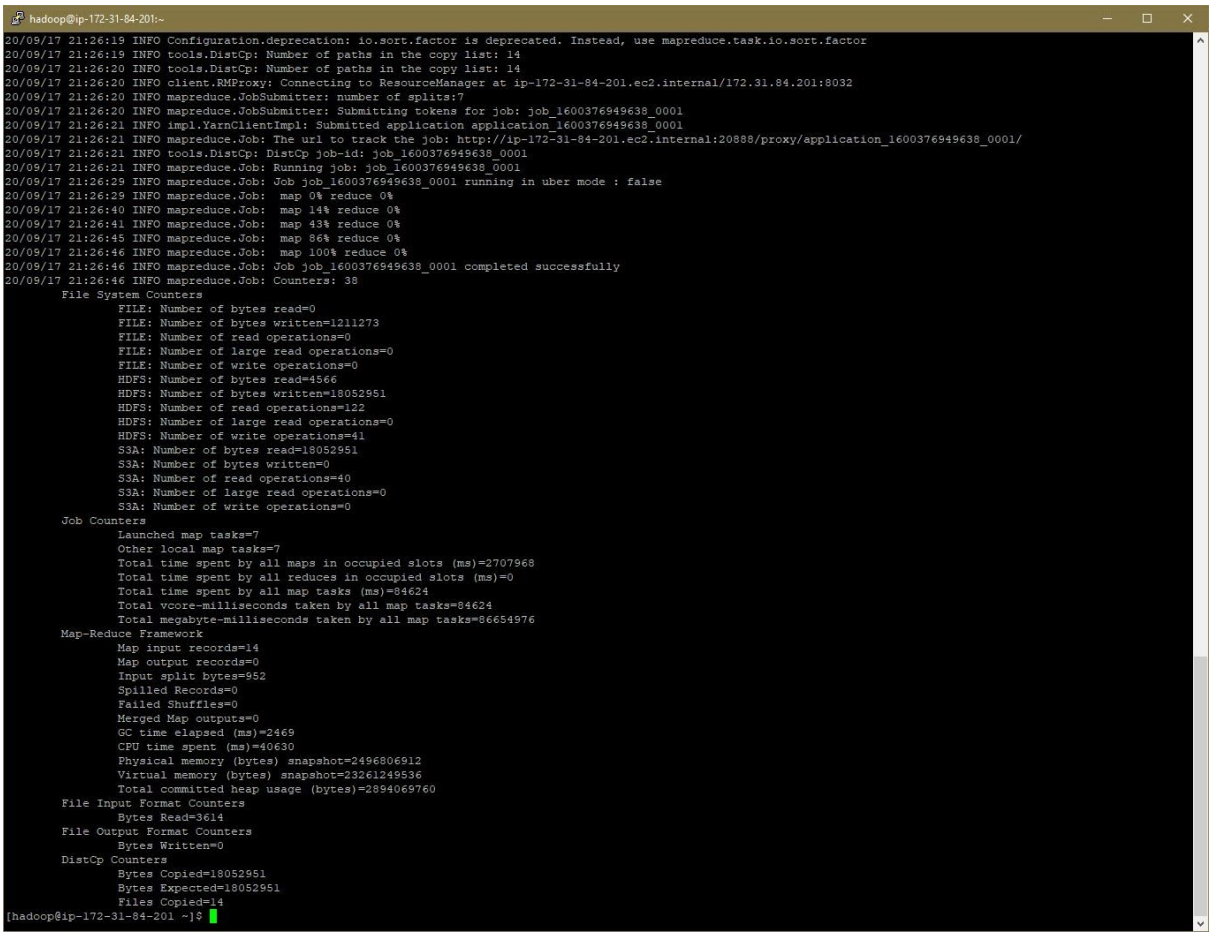
Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
All TCP	TCP	0 - 65535	Custom	sg-01abbeeaafcde5be7		Delete
All TCP	TCP	0 - 65535	Custom	sg-082c4611180041d01		Delete
All traffic	All	All	My IP	185.93.94.32/32		Delete
All UDP	UDP	0 - 65535	Custom			Delete

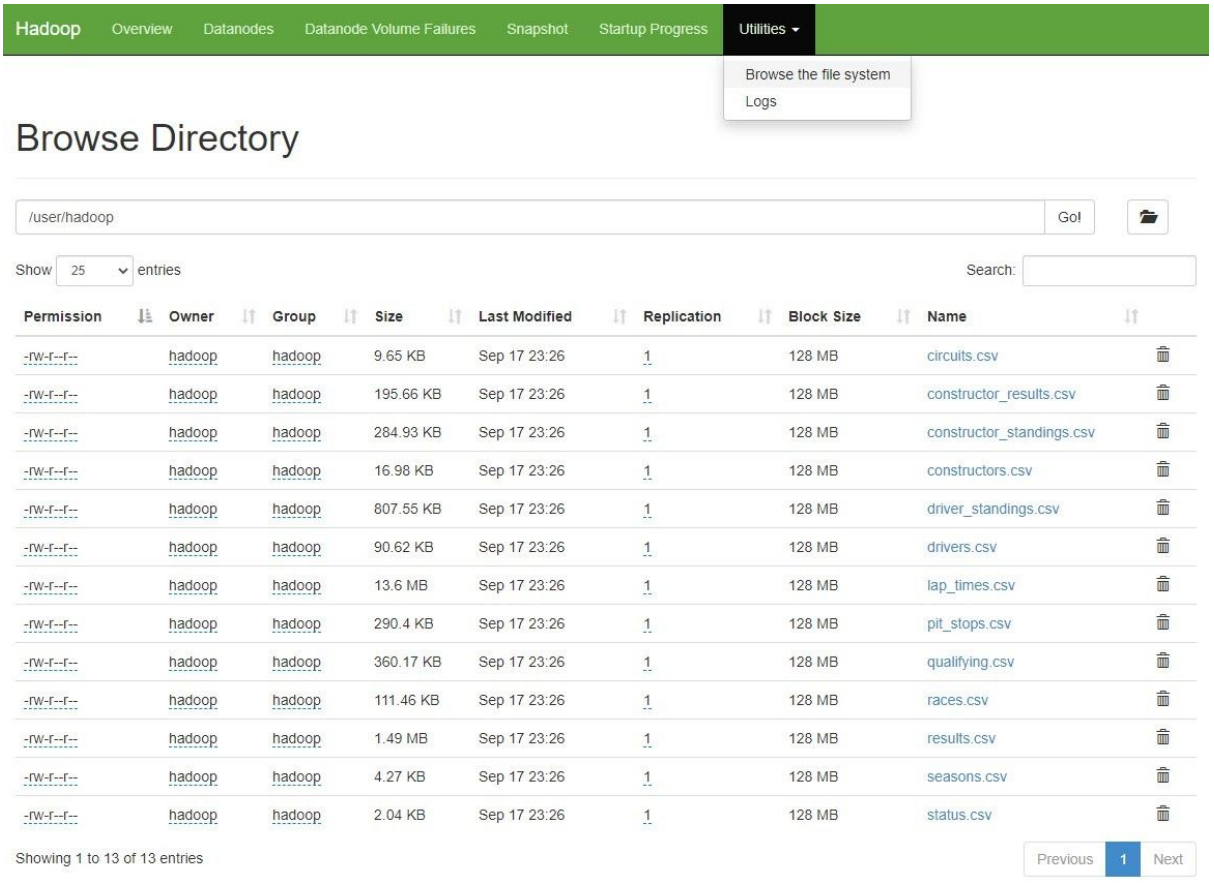
3. Create bucket on S3 and load data





4. Copy files from S3 to HDFS






```
[hadoop@ip-172-31-84-201 ~]$ hadoop fs -ls
Found 13 items
-rw-r--r--   1 hadoop hadoop       9878 2020-09-17 21:26 circuits.csv
-rw-r--r--   1 hadoop hadoop     200360 2020-09-17 21:26 constructor_results.csv
-rw-r--r--   1 hadoop hadoop     291772 2020-09-17 21:26 constructor_standings.csv
-rw-r--r--   1 hadoop hadoop      17387 2020-09-17 21:26 constructors.csv
-rw-r--r--   1 hadoop hadoop     826928 2020-09-17 21:26 driver_standings.csv
-rw-r--r--   1 hadoop hadoop      92796 2020-09-17 21:26 drivers.csv
-rw-r--r--   1 hadoop hadoop   14260968 2020-09-17 21:26 lap_times.csv
-rw-r--r--   1 hadoop hadoop     297371 2020-09-17 21:26 pit_stops.csv
-rw-r--r--   1 hadoop hadoop     368818 2020-09-17 21:26 qualifying.csv
-rw-r--r--   1 hadoop hadoop     114136 2020-09-17 21:26 races.csv
-rw-r--r--   1 hadoop hadoop    1566076 2020-09-17 21:26 results.csv
-rw-r--r--   1 hadoop hadoop       4376 2020-09-17 21:26 seasons.csv
-rw-r--r--   1 hadoop hadoop       2085 2020-09-17 21:26 status.csv
[hadoop@ip-172-31-84-201 ~]$
```

5. Open Hive shell

```
[hadoop@ip-172-31-84-201 ~]$ sudo hive shell

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive>
```

6. Create database

```
hive> show databases;
OK
default
Time taken: 0.555 seconds, Fetched: 1 row(s)
hive> CREATE DATABASE IF NOT EXISTS formula;
OK
Time taken: 0.081 seconds
hive> show databases;
OK
default
formula
Time taken: 0.013 seconds, Fetched: 2 row(s)
hive>
```

7. Create temp table (csv)

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS formula.races (
    > raceId INT,
    > year INT,
    > round INT,
    > curcuitId INT,
    > name STRING,
    > `date` DATE,
    > `time` STRING,
    > url STRING)
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE;
OK
Time taken: 0.283 seconds
hive>
```

8. Load data to table (csv)

```
hive> LOAD DATA INPATH '/user/hadoop/races.csv'
    > OVERWRITE INTO TABLE formula.races;
Loading data to table formula.races
chmod: changing permissions of 'hdfs://ip-172-31-84-
 of inode=/user/hive/warehouse/formula.db/races/race
OK
Time taken: 0.882 seconds
hive>
```

9. Verify data

```
hive> select * from formula.races limit 10;
OK
NULL    NULL    NULL    NULL    name    NULL    time    url
1       2009    1       1       "Australian Grand Prix" NULL    "06:00:00"      "http://en.wikipedia.org/wiki/2009_Australian_Grand_Prix"
2       2009    2       2       "Malaysian Grand Prix"  NULL    "09:00:00"      "http://en.wikipedia.org/wiki/2009_Malaysian_Grand_Prix"
3       2009    3       17      "Chinese Grand Prix"    NULL    "07:00:00"      "http://en.wikipedia.org/wiki/2009_Chinese_Grand_Prix"
4       2009    4       3       "Bahrain Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Bahrain_Grand_Prix"
5       2009    5       4       "Spanish Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Spanish_Grand_Prix"
6       2009    6       6       "Monaco Grand Prix"     NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Monaco_Grand_Prix"
7       2009    7       5       "Turkish Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Turkish_Grand_Prix"
8       2009    8       9       "British Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_British_Grand_Prix"
9       2009    9       20      "German Grand Prix"     NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_German_Grand_Prix"
Time taken: 1.346 seconds, Fetched: 10 row(s)
hive>
```

10. Create AVRO table

```
hive> CREATE TABLE IF NOT EXISTS formula.races_avro (
    > raceId INT,
    > year INT,
    > round INT,
    > curcuitId INT,
    > name STRING,
    > `date` DATE,
    > `time` STRING,
    > url STRING)
    > STORED AS AVRO;
OK
Time taken: 0.131 seconds
hive>
```

11. Load data from temp table (csv) to AVRO table

```
hive> INSERT INTO TABLE formula.races_avro SELECT * FROM formula.races;
Query ID = root_20200917224708_5d0127f3-b47e-4ab1-a2ec-30ab567bf878
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1600376949638_0017)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME: 5.40 s
--------------------------------------------------------------------------------
Loading data to table formula.races_avro
OK
Time taken: 12.743 seconds
hive>
```

12. Ready to write SQL!

```
hive> select * from formula.races_avro limit 25;
OK
NULL    NULL    NULL    NULL    name    NULL    time    url
1       2009    1       1       "Australian Grand Prix" NULL    "06:00:00"      "http://en.wikipedia.org/wiki/2009_Australian_Grand_Prix"
2       2009    2       2       "Malaysian Grand Prix"  NULL    "09:00:00"      "http://en.wikipedia.org/wiki/2009_Malaysian_Grand_Prix"
3       2009    3       17      "Chinese Grand Prix"    NULL    "07:00:00"      "http://en.wikipedia.org/wiki/2009_Chinese_Grand_Prix"
4       2009    4       3       "Bahrain Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Bahrain_Grand_Prix"
5       2009    5       4       "Spanish Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Spanish_Grand_Prix"
6       2009    6       6       "Monaco Grand Prix"     NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Monaco_Grand_Prix"
7       2009    7       5       "Turkish Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Turkish_Grand_Prix"
8       2009    8       9       "British Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_British_Grand_Prix"
9       2009    9       20      "German Grand Prix"     NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_German_Grand_Prix"
10      2009    10      11      "Hungarian Grand Prix"  NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Hungarian_Grand_Prix"
11      2009    11      12      "European Grand Prix"   NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_European_Grand_Prix"
12      2009    12      13      "Belgian Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Belgian_Grand_Prix"
13      2009    13      14      "Italian Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Italian_Grand_Prix"
14      2009    14      15      "Singapore Grand Prix"  NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2009_Singapore_Grand_Prix"
15      2009    15      22      "Japanese Grand Prix"   NULL    "05:00:00"      "http://en.wikipedia.org/wiki/2009_Japanese_Grand_Prix"
16      2009    16      18      "Brazilian Grand Prix"  NULL    "16:00:00"      "http://en.wikipedia.org/wiki/2009_Brazilian_Grand_Prix"
17      2009    17      24      "Abu Dhabi Grand Prix"  NULL    "11:00:00"      "http://en.wikipedia.org/wiki/2009_Abu_Dhabi_Grand_Prix"
18      2008    1       1       "Australian Grand Prix" NULL    "04:30:00"      "http://en.wikipedia.org/wiki/2008_Australian_Grand_Prix"
19      2008    2       2       "Malaysian Grand Prix"  NULL    "07:00:00"      "http://en.wikipedia.org/wiki/2008_Malaysian_Grand_Prix"
20      2008    3       3       "Bahrain Grand Prix"    NULL    "11:30:00"      "http://en.wikipedia.org/wiki/2008_Bahrain_Grand_Prix"
21      2008    4       4       "Spanish Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2008_Spanish_Grand_Prix"
22      2008    5       5       "Turkish Grand Prix"    NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2008_Turkish_Grand_Prix"
23      2008    6       6       "Monaco Grand Prix"     NULL    "12:00:00"      "http://en.wikipedia.org/wiki/2008_Monaco_Grand_Prix"
24      2008    7       7       "Canadian Grand Prix"   NULL    "17:00:00"      "http://en.wikipedia.org/wiki/2008_Canadian_Grand_Prix"
Time taken: 0.112 seconds, Fetched: 25 row(s)
hive>
```

```
hive> DESCRIBE FORMATTED formula.races_avro;
OK
# col_name              data_type               comment

raceid                  int
year                    int
round                   int
curcuitid               int
name                    string
date                    date
time                    string
url                     string

# Detailed Table Information
Database:               formula
Owner:                  root
CreateTime:             Thu Sep 17 22:46:19 UTC 2020
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://ip-172-31-84-201.ec2.internal:8020/user/hive/warehouse/formula.db/races_avro
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        numFiles                1
        numRows                 1036
        rawDataSize             0
        totalSize               98999
        transient_lastDdlTime   1600382841

# Storage Information
SerDe Library:          org.apache.hadoop.hive.serde2.avro.AvroSerDe
InputFormat:            org.apache.hadoop.hive.ql.io.avro.AvroContainerInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.avro.AvroContainerOutputFormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        serialization.format    1
Time taken: 0.08 seconds, Fetched: 37 row(s)
hive>
```

Data model

Tables

1. circuits
2. constructorResults
3. constructorStandings
4. constructors
5. driverStandings
6. drivers
7. lapTimes
8. pitStops
9. qualifying
10. races
11. results
12. seasons
13. status

Tables details

circuits.csv

Field	Type	Null	Key	Default	Extra
circuitId	int(11)	NO	PRI	NULL	auto_increment
circuitRef	varchar(255)	NO			
name	varchar(255)	NO			
location	varchar(255)	YES		NULL	
country	varchar(255)	YES		NULL	
lat	float	YES		NULL	
lng	float	YES		NULL	
alt	int(11)	YES		NULL	
url	varchar(255)	NO	UNI		

constructor_results.csv

Field	Type	Null	Key	Default	Extra
constructorResultsId	int(11)	NO	PRI	NULL	auto_increment
raceId	int(11)	NO		0	
constructorId	int(11)	NO		0	
points	float	YES		NULL	
status	varchar(255)	YES		NULL	

constructor_standings.csv

Field	Type	Null	Key	Default	Extra
constructorStandingsId	int(11)	NO	PRI	NULL	auto_increment
raceId	int(11)	NO		0	
constructorId	int(11)	NO		0	
points	float	NO		0	
position	int(11)	YES		NULL	
positionText	varchar(255)	YES		NULL	
wins	int(11)	NO		0	

constructors.csv

Field	Type	Null	Key	Default	Extra
constructorId	int(11)	NO	PRI	NULL	auto_increment
constructorRef	varchar(255)	NO			
name	varchar(255)	NO	UNI		
nationality	varchar(255)	YES		NULL	
url	varchar(255)	NO			

driver_standings.csv

Field	Type	Null	Key	Default	Extra
driverStandingsId	int(11)	NO	PRI	NULL	auto_increment
raceId	int(11)	NO		0	
driverId	int(11)	NO		0	
points	float	NO		0	
position	int(11)	YES		NULL	
positionText	varchar(255)	YES		NULL	
wins	int(11)	NO		0	

drivers.csv

Field	Type	Null	Key	Default	Extra
driverId	int(11)	NO	PRI	NULL	auto_increment
driverRef	varchar(255)	NO			
number	int(11)	YES		NULL	
code	varchar(3)	YES		NULL	
forename	varchar(255)	NO			
surname	varchar(255)	NO			
dob	date	YES		NULL	
nationality	varchar(255)	YES		NULL	
url	varchar(255)	NO	UNI		

lap_times.csv

Field	Type	Null	Key	Default	Extra
raceId	int(11)	NO	PRI	NULL	
driverId	int(11)	NO	PRI	NULL	
lap	int(11)	NO	PRI	NULL	
position	int(11)	YES		NULL	
time	varchar(255)	YES		NULL	
milliseconds	int(11)	YES		NULL	

pit_stops.csv

Field	Type	Null	Key	Default	Extra
raceId	int(11)	NO	PRI	NULL	
driverId	int(11)	NO	PRI	NULL	
stop	int(11)	NO	PRI	NULL	
lap	int(11)	NO		NULL	
time	time	NO		NULL	
duration	varchar(255)	YES		NULL	
milliseconds	int(11)	YES		NULL	

qualifying.csv

Field	Type	Null	Key	Default	Extra
qualifyId	int(11)	NO	PRI	NULL	auto_increment
raceId	int(11)	NO		0	
driverId	int(11)	NO		0	
constructorId	int(11)	NO		0	
number	int(11)	NO		0	
position	int(11)	YES		NULL	
q1	varchar(255)	YES		NULL	
q2	varchar(255)	YES		NULL	
q3	varchar(255)	YES		NULL	

races.csv

Field	Type	Null	Key	Default	Extra
raceId	int(11)	NO	PRI	NULL	auto_increment
year	int(11)	NO		0	
round	int(11)	NO		0	
circuitId	int(11)	NO		0	
name	varchar(255)	NO			
date	date	NO		0000-00-00	
time	time	YES		NULL	
url	varchar(255)	YES	UNI	NULL	

results.csv

Field	Type	Null	Key	Default	Extra
resultId	int(11)	NO	PRI	NULL	auto_increment
raceId	int(11)	NO		0	
driverId	int(11)	NO		0	
constructorId	int(11)	NO		0	
number	int(11)	YES		NULL	
grid	int(11)	NO		0	
position	int(11)	YES		NULL	
positionText	varchar(255)	NO			
positionOrder	int(11)	NO		0	
points	float	NO		0	
laps	int(11)	NO		0	
time	varchar(255)	YES		NULL	
milliseconds	int(11)	YES		NULL	
fastestLap	int(11)	YES		NULL	
rank	int(11)	YES		0	
fastestLapTime	varchar(255)	YES		NULL	
fastestLapSpeed	varchar(255)	YES		NULL	
statusId	int(11)	NO		0	

seasons.csv

Field	Type	Null	Key	Default	Extra
year	int(11)	NO	PRI	0	
url	varchar(255)	NO	UNI		

status.csv

Field	Type	Null	Key	Default	Extra
statusId	int(11)	NO	PRI	NULL	auto_increment
status	varchar(255)	NO			

Data model

to be done

Challenges

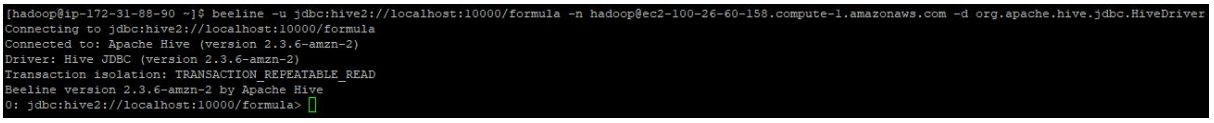
View examples (standard Hive shell vs Beeline shell)

Hive shell

Getting ready to start scripting Hive is simply as write
sudo hive shell
and we are ready to go. However, sometimes results of our queries could be not well formatted which I'll show on the first query below.

Beeline

To start scripting Hive could be more convenient in the beeline console. Beeline is a JDBC client using HiveServer2, so first it is required to connect to the database which we would like to query. To connect we need to write:
beeline -u jdbc:hive2://localhost:10000/formula -n hadoop@ec2-xxx-xx-xx-xxx.compute-1.amazonaws.com -d org.apache.hive.jdbc.HiveDriver



Queries

1. Top 10 drivers with the greatest number of won races ever.

Query

```
--Top 10 drivers with the greatest number of won races ever.
select count(*) wins, forename, surname
from results_csv left join drivers_csv on results_csv.driverId = drivers_csv.driverId
where results_csv.position = 1
group by drivers_csv.forename, drivers_csv.surname
order by wins desc
limit 10;
```

Results

Standard Hive shell

```
hive> use formula;
OK
Time taken: 0.048 seconds
hive> --Top 10 drivers with the greatest number of won races ever.
hive> select count(*) wins, forename, surname
    > from results_csv left join drivers_csv on results_csv.driverId = drivers_csv.driverId
    > where results_csv.position = 1
    > group by drivers_csv.forename, drivers_csv.surname
    > order by wins desc
    > limit 10;
Query ID = root_20200920112701_1427lb03-ac93-425d-acbb-371906438a6e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1600507775912_0018)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Map 4 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 5.91 s
----------------------------------------------------------------------------------------------
OK
91      "Michael"       "Schumacher"
90      "Lewis" "Hamilton"
53      "Sebastian"     "Vettel"
51      "Alain" "Prost"
41      "Ayrton"        "Senna"
32      "Fernando"      "Alonso"
31      "Nigel" "Mansell"
27      "Jackie"        "Stewart"
25      "Niki"  "Lauda"
25      "Jim"   "Clark"
Time taken: 9.944 seconds, Fetched: 10 row(s)
hive>
```

Beeline

```
[hadoop@ip-172-31-88-90 ~]$ beeline -u jdbc:hive2://localhost:10000/formula -n hadoop@ec2-100-26-60-158.compute-1.amazonaws.com -d org.apache.hive.jdbc.HiveDriver
Connecting to jdbc:hive2://localhost:10000/formula
Connected to: Apache Hive (version 2.3.6-amzn-2)
Driver: Hive JDBC (version 2.3.6-amzn-2)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 2.3.6-amzn-2 by Apache Hive
0: jdbc:hive2://localhost:10000/formula> --Top 10 drivers with the greatest number of won races ever.
0: jdbc:hive2://localhost:10000/formula> select count(*) wins, forename, surname
. . . . . .> from results_csv left join drivers_csv on results_csv.driverId = drivers_csv.driverId
. . . . . .> where results_csv.position = 1
. . . . . .> group by drivers_csv.forename, drivers_csv.surname
. . . . . .> order by wins desc
. . . . . .> limit 10;
+-------+-------------+---------------+
| wins  |   forename  |    surname    |
+-------+-------------+---------------+
| 91    | "Michael"   | "Schumacher"  |
| 90    | "Lewis"     | "Hamilton"    |
| 53    | "Sebastian" | "Vettel"      |
| 51    | "Alain"     | "Prost"       |
| 41    | "Ayrton"    | "Senna"       |
| 32    | "Fernando"  | "Alonso"      |
| 31    | "Nigel"     | "Mansell"     |
| 27    | "Jackie"    | "Stewart"     |
| 25    | "Niki"      | "Lauda"       |
| 25    | "Jim"       | "Clark"       |
+-------+-------------+---------------+
10 rows selected (12.814 seconds)
0: jdbc:hive2://localhost:10000/formula>
```

2. The best driver (having the greatest number of won races) from each country.

Query

Results

Visualization

to be done