Session 28 – Spark MLLib 1

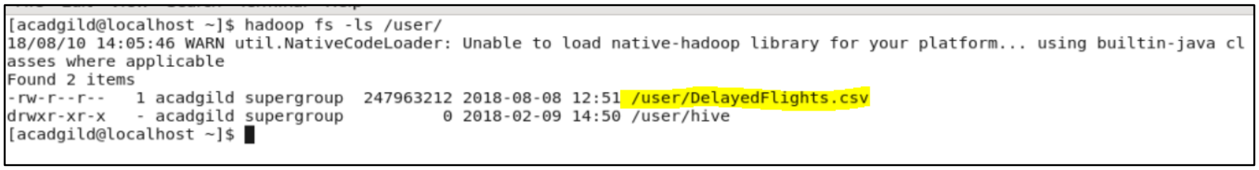
Assignment 1

The U.S. Department of Transportation's (DOT) Bureau of Transportation Statistics (BTS) tracks the on-time performance of domestic flights operated by large air carriers. Summary information on the number of on-time, delayed, canceled, and diverted flights appears in DOT's monthly Air Travel Consumer Report, published about 30 days after the month's end, as well as in summary tables posted on this website. Summary statistics and raw data are made available to the public at the time the Air Travel Consumer Report is released.

<u>Step 1: Loading raw data into the root directory of HDFS.</u>

The dataset provided 'DelayedFlights.csv' is loaded onto HDFS using the command:

hadoop fs –put /home/acadgild/DelayedFlights.csv /user/

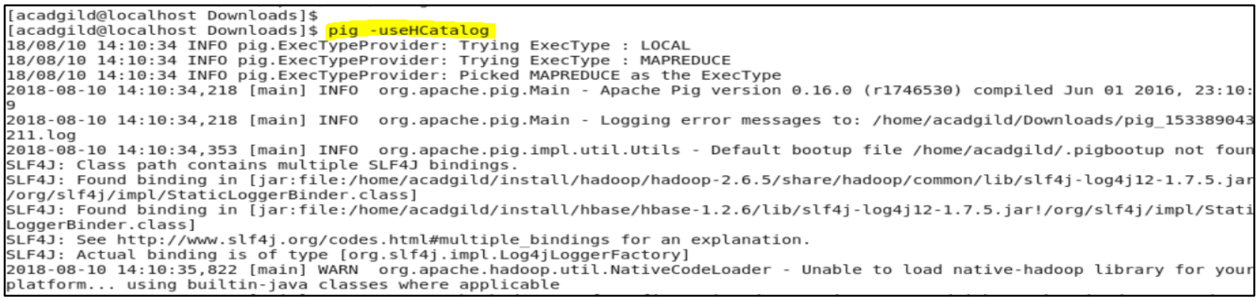


```
[acadgild@localhost ~]$ hadoop fs -ls /user/
18/08/10 14:05:46 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Found 2 items
-rw-r--r--   1 acadgild supergroup  247963212 2018-08-08 12:51 /user/DelayedFlights.csv
drwxr-xr-x   - acadgild supergroup          0 2018-02-09 14:50 /user/hive
[acadgild@localhost ~]$ 
```

The file can be seen on HDFS on doing an ls on that folder.

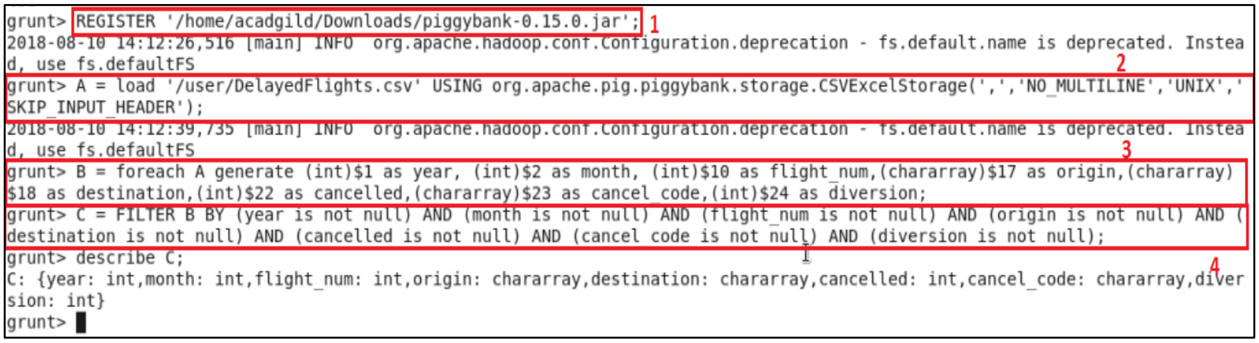
<u>Step 2: Pre-processing using Pig.</u>

Open pig using the parameter :   *pig –useHCatalog*



```
[acadgild@localhost Downloads]$
[acadgild@localhost Downloads]$ pig -useHCatalog
18/08/10 14:10:34 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
18/08/10 14:10:34 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
18/08/10 14:10:34 INFO pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2018-08-10 14:10:34,218 [main] INFO  org.apache.pig.Main - Apache Pig version 0.16.0 (r1746530) compiled Jun 01 2016, 23:10:
9
2018-08-10 14:10:34,218 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/acadgild/Downloads/pig_153389043
211.log
2018-08-10 14:10:34,353 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home/acadgild/.pigbootup not foun
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar
/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/Stati
LoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
2018-08-10 14:10:35,822 [main] WARN  org.apache.hadoop.util.NativeCodeLoader - Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable
```

Then, the raw data is processed as below :



```
grunt> REGISTER '/home/acadgild/Downloads/piggybank-0.15.0.jar';
2018-08-10 14:12:26,516 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> A = load '/user/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','
SKIP_INPUT_HEADER');
2018-08-10 14:12:39,735 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> B = foreach A generate (int)$1 as year, (int)$2 as month, (int)$10 as flight_num,(chararray)$17 as origin,(chararray)
$18 as destination,(int)$22 as cancelled,(chararray)$23 as cancel_code,(int)$24 as diversion;
grunt> C = FILTER B BY (year is not null) AND (month is not null) AND (flight_num is not null) AND (origin is not null) AND (
destination is not null) AND (cancelled is not null) AND (cancel_code is not null) AND (diversion is not null);
grunt> describe C;
C: {year: int,month: int,flight_num: int,origin: chararray,destination: chararray,cancelled: int,cancel_code: chararray,diver
sion: int}
grunt> 
```

With reference to the screenshot above,

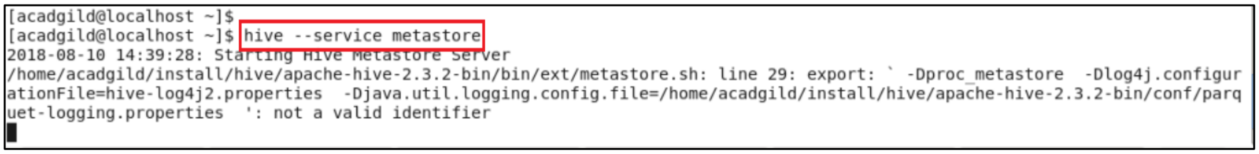
1: As the file is comma separated, we will register and use piggybank jar in order to use the CSVExcelStorage class.
2: In relation A, we are loading the data using CSVExcelStorage because of its effective technique to handle double quotes and headers.
3: In relation B, we are generating the columns that are required for processing and explicitly typecasting each of them.
4: In relation C, we are filtering out the null values if any, from the generated columns.

<u>Step 3: Loading pre-processed data from pig to hive using HCatalog.</u>

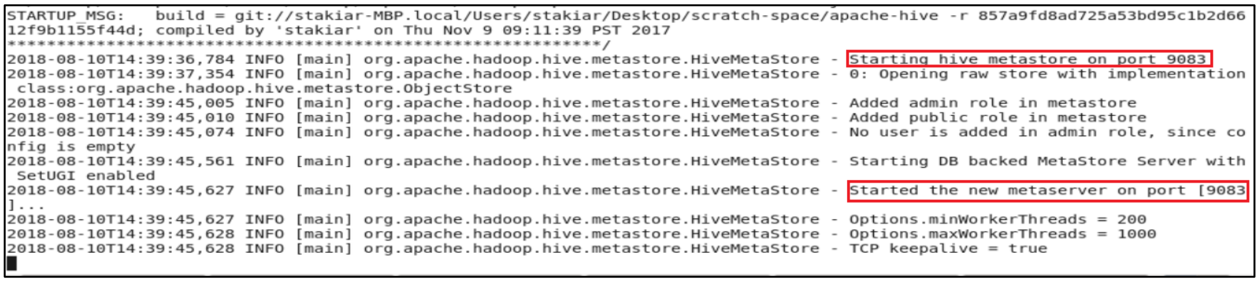
Once the data is cleaned, we need to transfer it to process and gain insights. We will be using HCatalog and sending the cleansed data directly from pig to hive using it.

We need to start hive metastore service before loading data using HCatalog using below command.

hive –service metastore

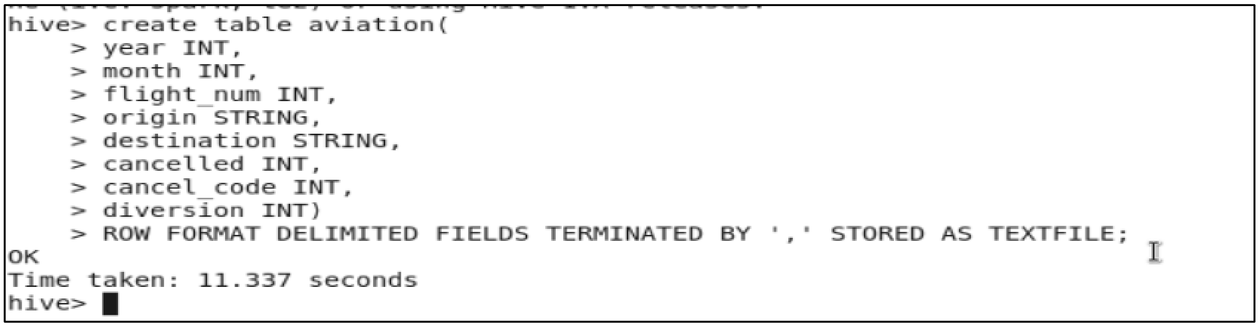


```
[acadgild@localhost ~]$
[acadgild@localhost ~]$ hive --service metastore
2018-08-10 14:39:28: Starting Hive Metastore Server
/home/acadgild/install/hive/apache-hive-2.3.2-bin/bin/ext/metastore.sh: line 29: export: ` -Dproc_metastore  -Dlog4j.configur
ationFile=hive-log4j2.properties  -Djava.util.logging.config.file=/home/acadgild/install/hive/apache-hive-2.3.2-bin/conf/parq
uet-logging.properties  ': not a valid identifier
█
```



```
STARTUP_MSG:   build = git://stakiar-MBP.local/Users/stakiar/Desktop/scratch-space/apache-hive -r 857a9fd8ad725a53bd95c1b2d66
12f9b1155f44d; compiled by 'stakiar' on Thu Nov 9 09:11:39 PST 2017
************************************************************/
2018-08-10T14:39:36,784 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Starting hive metastore on port 9083
2018-08-10T14:39:37,354 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - 0: Opening raw store with implementation
 class:org.apache.hadoop.hive.metastore.ObjectStore
2018-08-10T14:39:45,005 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Added admin role in metastore
2018-08-10T14:39:45,010 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Added public role in metastore
2018-08-10T14:39:45,074 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - No user is added in admin role, since co
nfig is empty
2018-08-10T14:39:45,561 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Starting DB backed MetaStore Server with
 SetUGI enabled
2018-08-10T14:39:45,627 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Started the new metaserver on port [9083
]...
2018-08-10T14:39:45,627 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Options.minWorkerThreads = 200
2018-08-10T14:39:45,628 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Options.maxWorkerThreads = 1000
2018-08-10T14:39:45,628 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - TCP keepalive = true
█
```

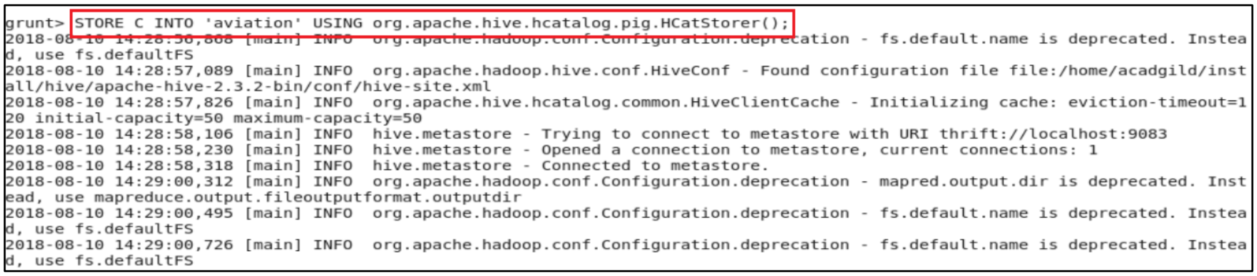
Next, create a hive table with the same schema as you had pre-processed in the Pig.



```
hive> create table aviation(
    > year INT,
    > month INT,
    > flight_num INT,
    > origin STRING,
    > destination STRING,
    > cancelled INT,
    > cancel_code INT,
    > diversion INT)
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;
OK
Time taken: 11.337 seconds
hive> █
```

We need to maintain the same order as well as same datatypes while creating the table.

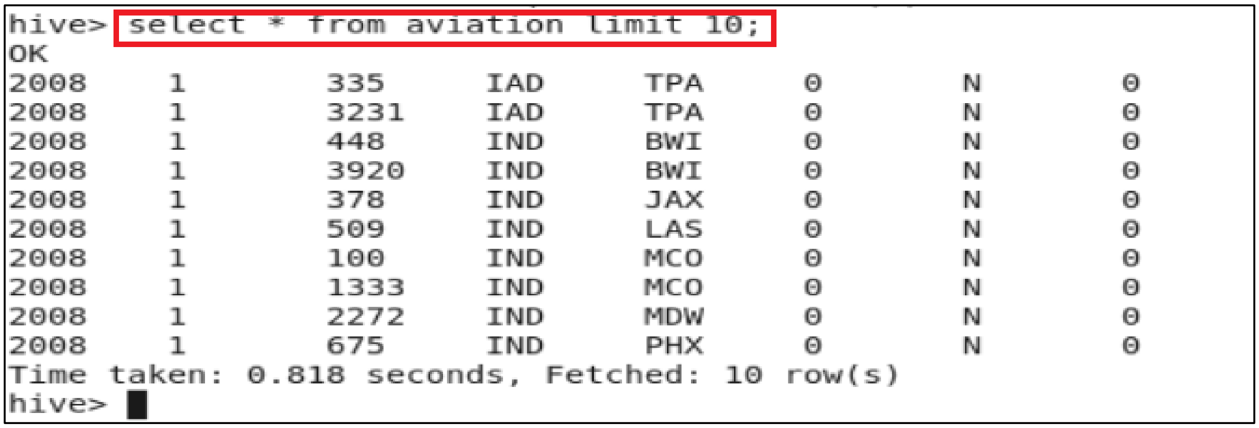
Use below command in pig grunt shell to load the data to hive.



```
grunt> STORE C INTO 'aviation' USING org.apache.hive.hcatalog.pig.HCatStorer();
2018-08-10 14:28:56,868 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
2018-08-10 14:28:57,089 [main] INFO  org.apache.hadoop.hive.conf.HiveConf - Found configuration file file:/home/acadgild/inst
all/hive/apache-hive-2.3.2-bin/conf/hive-site.xml
2018-08-10 14:28:57,826 [main] INFO  org.apache.hive.hcatalog.common.HiveClientCache - Initializing cache: eviction-timeout=1
20 initial-capacity=50 maximum-capacity=50
2018-08-10 14:28:58,106 [main] INFO  hive.metastore - Trying to connect to metastore with URI thrift://localhost:9083
2018-08-10 14:28:58,230 [main] INFO  hive.metastore - Opened a connection to metastore, current connections: 1
2018-08-10 14:28:58,318 [main] INFO  hive.metastore - Connected to metastore.
2018-08-10 14:29:00,312 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.output.dir is deprecated. Inst
ead, use mapreduce.output.fileoutputformat.outputdir
2018-08-10 14:29:00,495 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
2018-08-10 14:29:00,726 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
```

This will load the data into hive table which we had already created.
We can cross check the same using in hive shell.

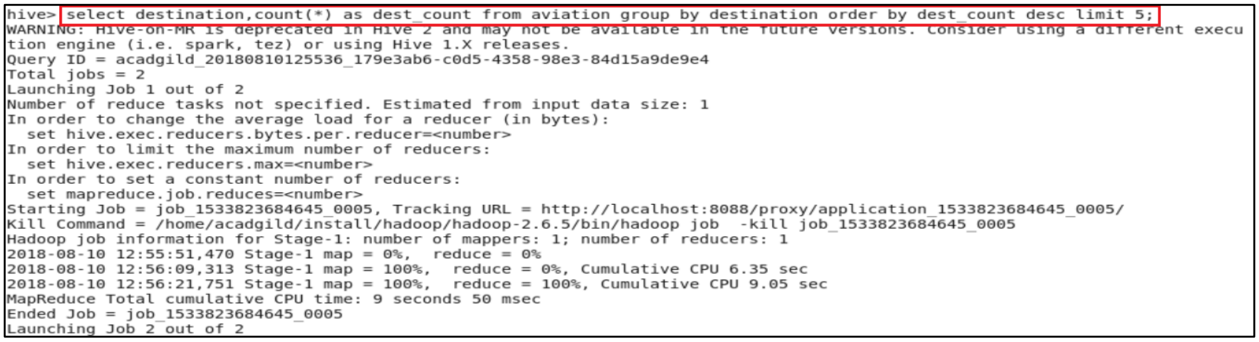
*SELECT * FROM aviation LIMIT 10;*



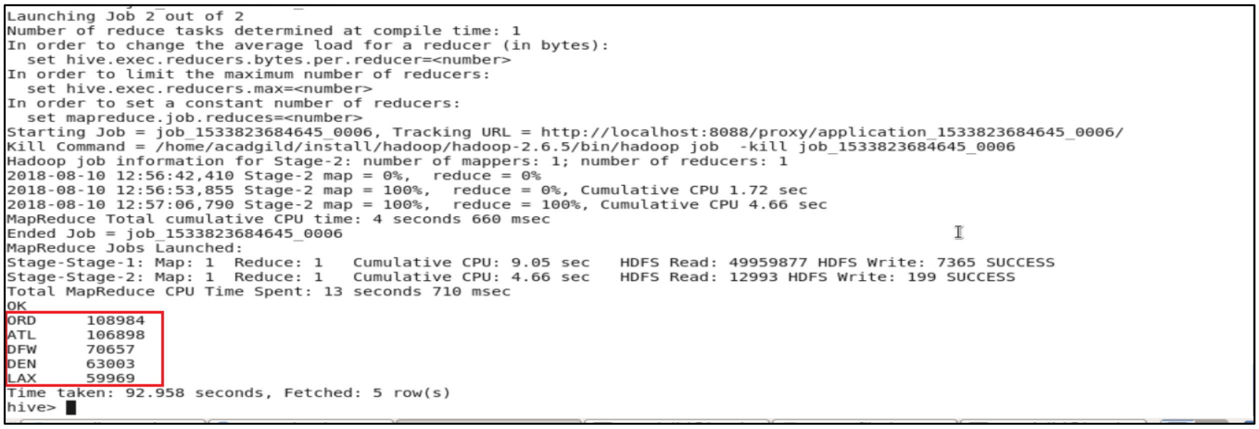
```
hive> select * from aviation limit 10;
OK
2008    1       335     IAD     TPA     0       N       0
2008    1       3231    IAD     TPA     0       N       0
2008    1       448     IND     BWI     0       N       0
2008    1       3920    IND     BWI     0       N       0
2008    1       378     IND     JAX     0       N       0
2008    1       509     IND     LAS     0       N       0
2008    1       100     IND     MCO     0       N       0
2008    1       1333    IND     MCO     0       N       0
2008    1       2272    IND     MDW     0       N       0
2008    1       675     IND     PHX     0       N       0
Time taken: 0.818 seconds, Fetched: 10 row(s)
hive> █
```

Problem Statement 1
Find out the top 5 most visited destinations.

SELECT destination,count(*) as dest_count
FROM aviation
GROUP BY destination
ORDER BY dest_count desc
LIMIT 5;



```
hive> select destination,count(*) as dest_count from aviation group by destination order by dest_count desc limit 5;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execu
tion engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180810125536_179e3ab6-c0d5-4358-98e3-84d15a9de9e4
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1533823684645_0005, Tracking URL = http://localhost:8088/proxy/application_1533823684645_0005/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1533823684645_0005
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-08-10 12:55:51,470 Stage-1 map = 0%,  reduce = 0%
2018-08-10 12:56:09,313 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 6.35 sec
2018-08-10 12:56:21,751 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 9.05 sec
MapReduce Total cumulative CPU time: 9 seconds 50 msec
Ended Job = job_1533823684645_0005
Launching Job 2 out of 2
```

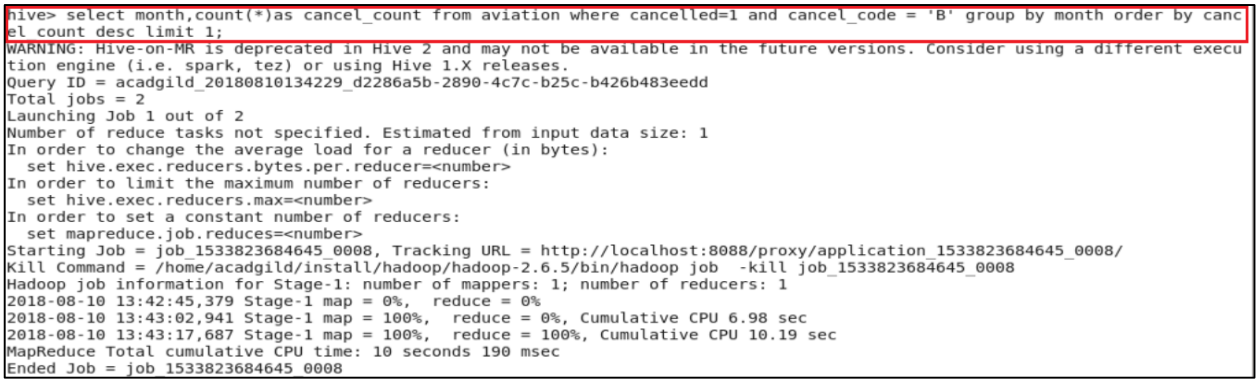


```
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1533823684645_0006, Tracking URL = http://localhost:8088/proxy/application_1533823684645_0006/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1533823684645_0006
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-08-10 12:56:42,410 Stage-2 map = 0%,  reduce = 0%
2018-08-10 12:56:53,855 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 1.72 sec
2018-08-10 12:57:06,790 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 4.66 sec
MapReduce Total cumulative CPU time: 4 seconds 660 msec
Ended Job = job_1533823684645_0006
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 9.05 sec   HDFS Read: 49959877 HDFS Write: 7365 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 4.66 sec   HDFS Read: 12993 HDFS Write: 199 SUCCESS
Total MapReduce CPU Time Spent: 13 seconds 710 msec
OK
ORD     108984
ATL     106898
DFW     70657
DEN     63003
LAX     59969
Time taken: 92.958 seconds, Fetched: 5 row(s)
hive> █
```

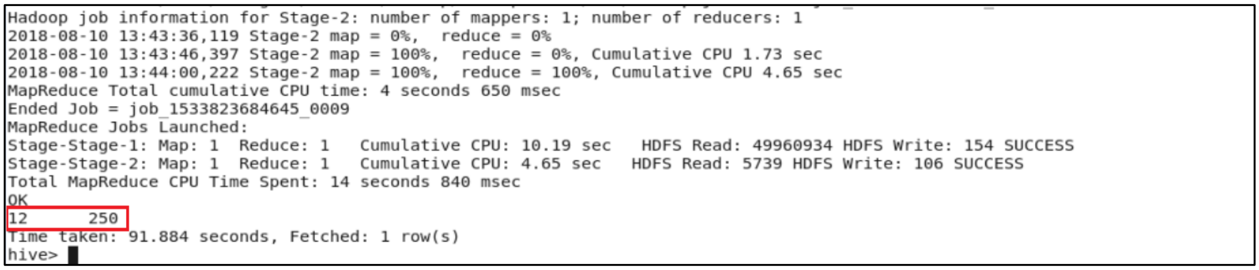
Top 5 destinations are displayed after the job is successfully completed.

Problem Statement 2
Which month has seen the most number of cancellations due to bad weather?

SELECT month,count(*) as cancel_count
FROM aviation
WHERE cancelled = 1 and cancel_code = 'B'
GROUP BY month
ORDER BY cancel_count
LIMIT 1;



```
hive> select month,count(*)as cancel_count from aviation where cancelled=1 and cancel_code = 'B' group by month order by canc
el_count desc limit 1;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execu
tion engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180810134229_d2286a5b-2890-4c7c-b25c-b426b483eedd
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1533823684645_0008, Tracking URL = http://localhost:8088/proxy/application_1533823684645_0008/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1533823684645_0008
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-08-10 13:42:45,379 Stage-1 map = 0%,  reduce = 0%
2018-08-10 13:43:02,941 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 6.98 sec
2018-08-10 13:43:17,687 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 10.19 sec
MapReduce Total cumulative CPU time: 10 seconds 190 msec
Ended Job = job_1533823684645_0008
```

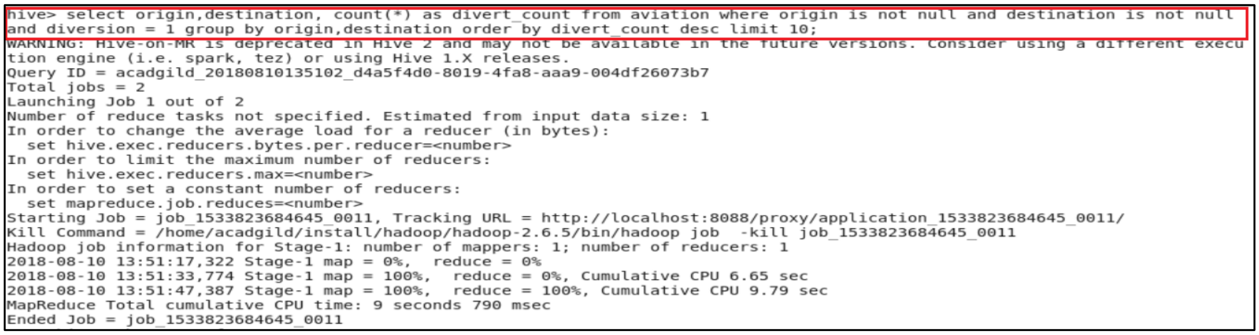


```
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-08-10 13:43:36,119 Stage-2 map = 0%,  reduce = 0%
2018-08-10 13:43:46,397 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 1.73 sec
2018-08-10 13:44:00,222 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 4.65 sec
MapReduce Total cumulative CPU time: 4 seconds 650 msec
Ended Job = job_1533823684645_0009
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 10.19 sec   HDFS Read: 49960934 HDFS Write: 154 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 4.65 sec   HDFS Read: 5739 HDFS Write: 106 SUCCESS
Total MapReduce CPU Time Spent: 14 seconds 840 msec
OK
12      250
Time taken: 91.884 seconds, Fetched: 1 row(s)
hive> █
```

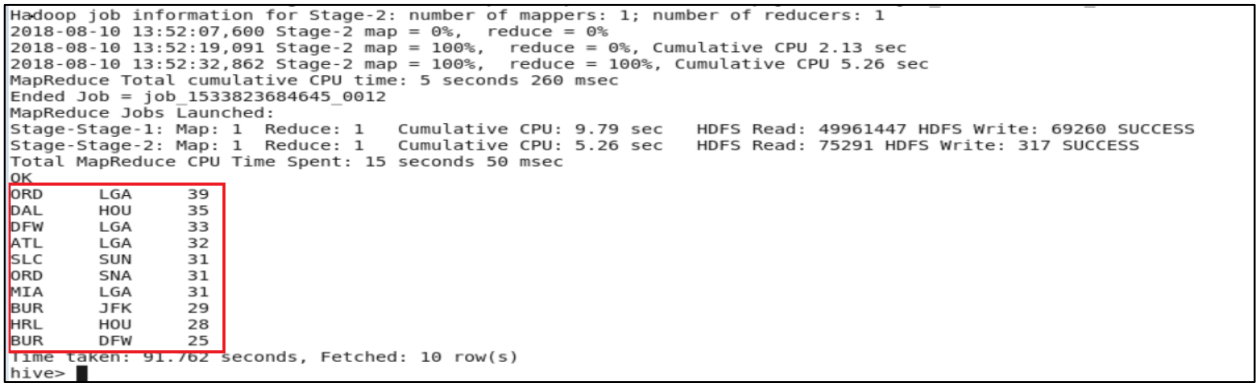
The month 12 i.e., December has seen the most number of cancellations due to bad weather.

Problem Statement 3
Which route (origin & destination) has seen the maximum diversion?

SELECT origin,destination,count(*) as divert_count
FROM aviation
WHERE origin is NOT NULL
AND destination is NOT NULL
AND diversion = 1
GROUP BY origin,destination
ORDER BY divert_count desc
LIMIT 10;



```
hive> select origin,destination, count(*) as divert_count from aviation where origin is not null and destination is not null
and diversion = 1 group by origin,destination order by divert_count desc limit 10;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execu
tion engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180810135102_d4a5f4d0-8019-4fa8-aaa9-004df26073b7
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1533823684645_0011, Tracking URL = http://localhost:8088/proxy/application_1533823684645_0011/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1533823684645_0011
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-08-10 13:51:17,322 Stage-1 map = 0%,  reduce = 0%
2018-08-10 13:51:33,774 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 6.65 sec
2018-08-10 13:51:47,387 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 9.79 sec
MapReduce Total cumulative CPU time: 9 seconds 790 msec
Ended Job = job_1533823684645_0011
```



```
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-08-10 13:52:07,600 Stage-2 map = 0%,  reduce = 0%
2018-08-10 13:52:19,091 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 2.13 sec
2018-08-10 13:52:32,862 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 5.26 sec
MapReduce Total cumulative CPU time: 5 seconds 260 msec
Ended Job = job_1533823684645_0012
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 9.79 sec   HDFS Read: 49961447 HDFS Write: 69260 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 5.26 sec   HDFS Read: 75291 HDFS Write: 317 SUCCESS
Total MapReduce CPU Time Spent: 15 seconds 50 msec
OK
ORD     LGA     39
DAL     HOU     35
DFW     LGA     33
ATL     LGA     32
SLC     SUN     31
ORD     SNA     31
MIA     LGA     31
BUR     JFK     29
HRL     HOU     28
BUR     DFW     25
Time taken: 91.762 seconds, Fetched: 10 row(s)
hive> 
```

The top 10 routes with maximum number of diversions are displayed in the result.