EECSE6893 Big Data Analytics

Homework 1

Nan Zhao nz2250

1. Install Hadoop
    a. Example 1 Running Result

```
nan@nan-ThundeRobot: /usr/local/Cellar/hadoop/2.7.3
                HDFS: Number of bytes read=2620
                HDFS: Number of bytes written=215
                HDFS: Number of read operations=43
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=3
        Job Counters
                Launched map tasks=10
                Launched reduce tasks=1
                Data-local map tasks=10
                Total time spent by all maps in occupied slots (ms)=49550
                Total time spent by all reduces in occupied slots (ms)=3726
                Total time spent by all map tasks (ms)=49550
                Total time spent by all reduce tasks (ms)=3726
                Total vcore-milliseconds taken by all map tasks=49550
                Total vcore-milliseconds taken by all reduce tasks=3726
                Total megabyte-milliseconds taken by all map tasks=50739200
                Total megabyte-milliseconds taken by all reduce tasks=3815424
        Map-Reduce Framework
                Map input records=10
                Map output records=20
                Map output bytes=180
                Map output materialized bytes=280
                Input split bytes=1440
                Combine input records=0
                Combine output records=0
                Reduce input groups=2
                Reduce shuffle bytes=280
                Reduce input records=20
                Reduce output records=0
                Spilled Records=40
                Shuffled Maps =10
                Failed Shuffles=0
                Merged Map outputs=10
                GC time elapsed (ms)=2168
                CPU time spent (ms)=8680
                Physical memory (bytes) snapshot=2952298496
                Virtual memory (bytes) snapshot=21359263744
                Total committed heap usage (bytes)=2152726528
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=1180
        File Output Format Counters
                Bytes Written=97
Job Finished in 21.634 seconds
Estimated value of Pi is 3.14800000000000000000
nan@nan-ThundeRobot:/usr/local/Cellar/hadoop/2.7.3$
```

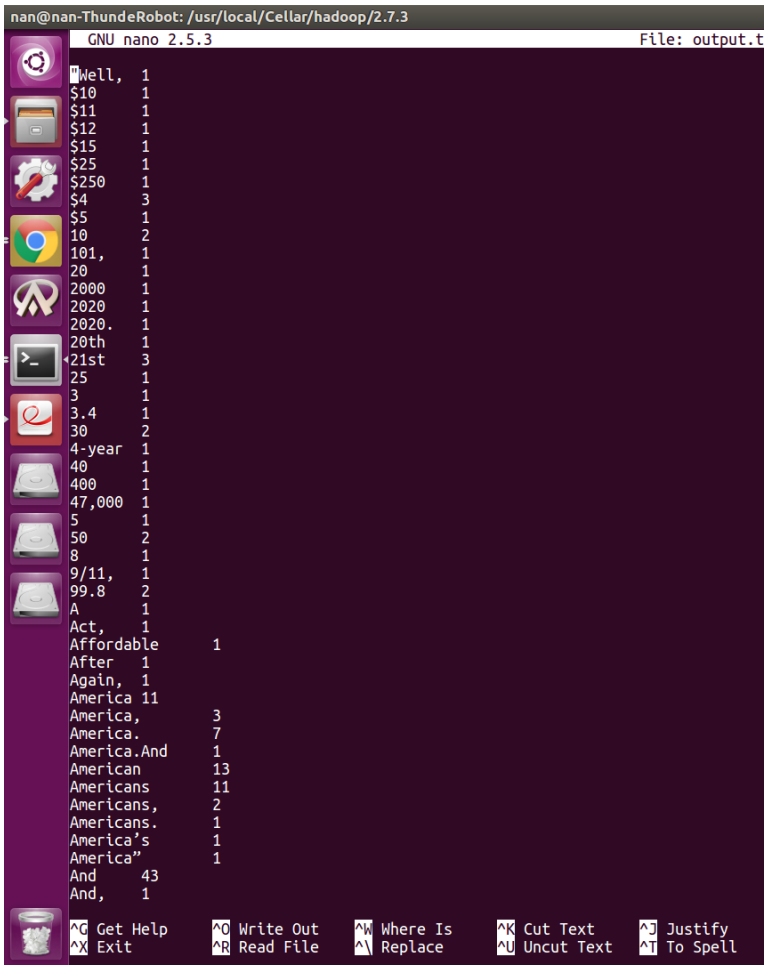
    b. Example 2 Running Result

```
                Bytes Read=1180
        File Output Format Counters
                Bytes Written=97
Job Finished in 21.634 seconds
Estimated value of Pi is 3.14800000000000000000
nan@nan-ThundeRobot:/usr/local/Cellar/hadoop/2.7.3$ hadoop jar hadoop-mapreduce-examples-2.6.0.jar sudoku puzzle1.dta
Solving puzzle1.dta
8 5 1 3 9 2 6 4 7
4 3 2 6 7 8 1 9 5
7 9 6 5 1 4 3 8 2
6 1 4 8 2 3 7 5 9
5 7 8 9 6 1 4 2 3
3 2 9 4 5 7 8 1 6
9 4 7 2 8 6 5 3 1
1 8 5 7 3 9 2 6 4
2 6 3 1 4 5 9 7 8

Found 1 solutions
nan@nan-ThundeRobot:/usr/local/Cellar/hadoop/2.7.3$
```

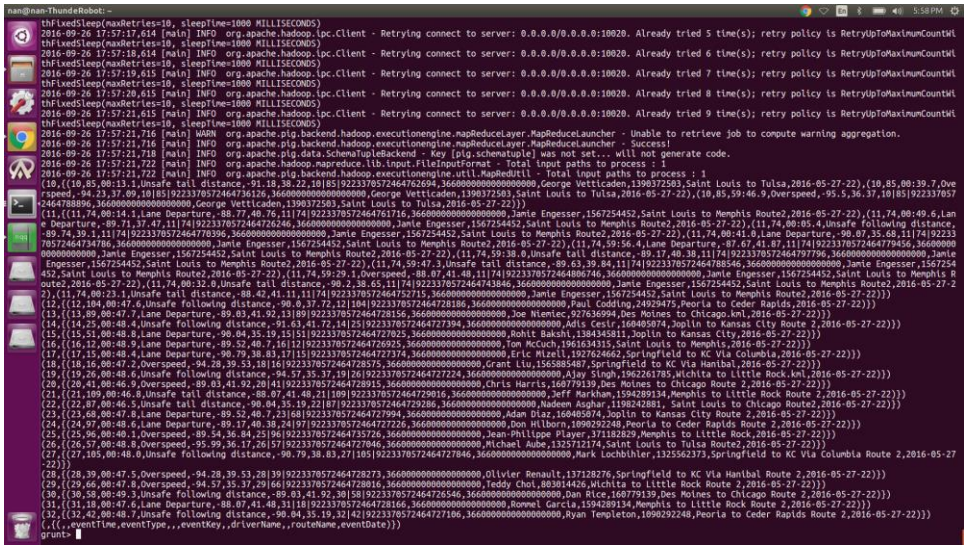
    c. Example 3 Running Result

```
nan@nan-ThundeRobot: /usr/local/Cellar/hadoop/2.7.3
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=31862
                HDFS: Number of bytes written=17113
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=1857
                Total time spent by all reduces in occupied slots (ms)=1859
                Total time spent by all map tasks (ms)=1857
                Total time spent by all reduce tasks (ms)=1859
                Total vcore-milliseconds taken by all map tasks=1857
                Total vcore-milliseconds taken by all reduce tasks=1859
                Total megabyte-milliseconds taken by all map tasks=1901568
                Total megabyte-milliseconds taken by all reduce tasks=1903616
        Map-Reduce Framework
                Map input records=167
                Map output records=5592
                Map output bytes=54123
                Map output materialized bytes=24151
                Input split bytes=107
                Combine input records=5592
                Combine output records=1783
                Reduce input groups=1783
                Reduce shuffle bytes=24151
                Reduce input records=1783
                Reduce output records=1783
                Spilled Records=3566
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=98
                CPU time spent (ms)=1790
                Physical memory (bytes) snapshot=459309056
                Virtual memory (bytes) snapshot=3890855936
                Total committed heap usage (bytes)=349175808
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=31755
        File Output Format Counters
                Bytes Written=17113
nan@nan-ThundeRobot:/usr/local/Cellar/hadoop/2.7.3$
```

Content of output.txt



2. Download Airline Data
3. Learn to use PIG
    a. Pig Example Running Result



b. Pig Example on 2008 Airline Data

```
(2008,1,1450)
(2008,1,2245)
(2008,1,615)
(2008,1,1150)
(2008,1,2025)
(2008,1,1038)
(2008,1,1900)
(2008,1,700)
(2008,1,948)
(2008,1,646)
(2008,1,1110)
(2008,1,1535)
(2008,1,1919)
(2008,1,1053)
(2008,1,1433)
(2008,1,2015)
(2008,1,2139)
(2008,1,1500)
(2008,1,850)
(2008,1,646)
(2008,1,1221)
(2008,1,1738)
(2008,1,1813)
(2008,1,802)
(2008,1,1820)
(2008,1,821)
(2008,1,1734)
(2008,1,712)
(2008,1,1318)
(2008,1,958)
(2008,1,1859)
(2008,1,1538)
(2008,1,933)
(2008,1,2248)
(2008,1,1327)
(2008,1,624)
(2008,1,1614)
(2008,1,1917)
(2008,1,1832)
(2008,1,1229)
(2008,1,1256)
(2008,1,2118)
(2008,1,905)
(2008,1,1739)
(2008,1,906)
(2008,1,816)
(2008,1,1325)
(2008,1,1506)
(2008,1,2039)
(2008,1,924)
(2008,1,1611)
grunt>
```

Below is the script I ran:

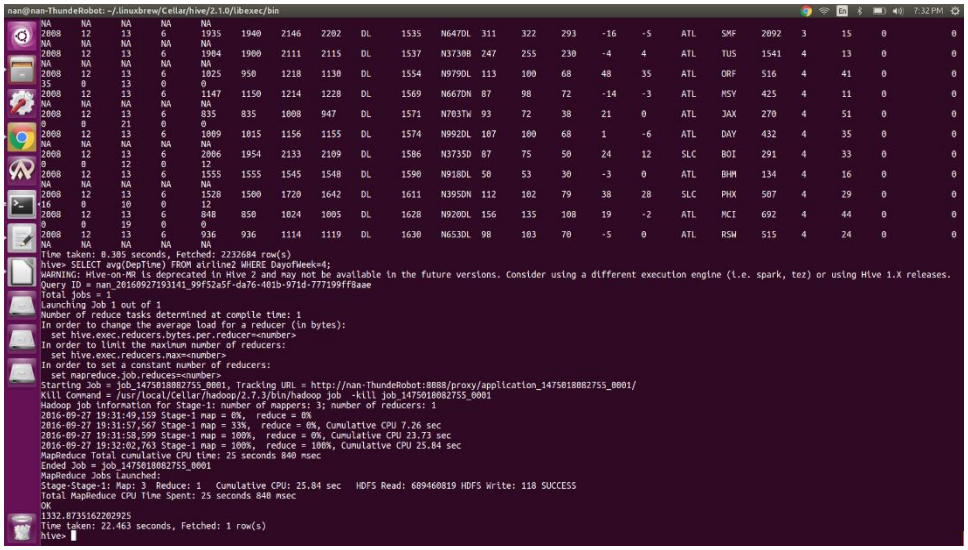
```
*2008_Example_Running_Script_Pig (~/Downloads) - gedit
Open ▾ Save
*2008_Example_Running_Script_Pig   2008_Example_Running_Script

Airline_Info = LOAD '/user/pig_example/2008.csv' USING PigStorage(',') AS
(Year:int,Month:int,DayofMonth:int,DayOfWeek:int,DepTime:int,CRSDepTime:int,ArrTime:int,CRSArrTime:int,UniqueCarrier:chararray,FlightNum:chararray,TailNum:chararray,ActualElapsedTime:i
DESCRIBE Airline_Info;

Airline_Info_subset = LIMIT Airline_Info 100;
DESCRIBE Airline_Info_subset;

specific_columns = FOREACH Airline_Info_subset GENERATE Year, Month, DepTime;
DESCRIBE specific_columns;

DUMP specifici_columns;
```

4. Learn to use Hive

a. Running Example on 2007 Airline data

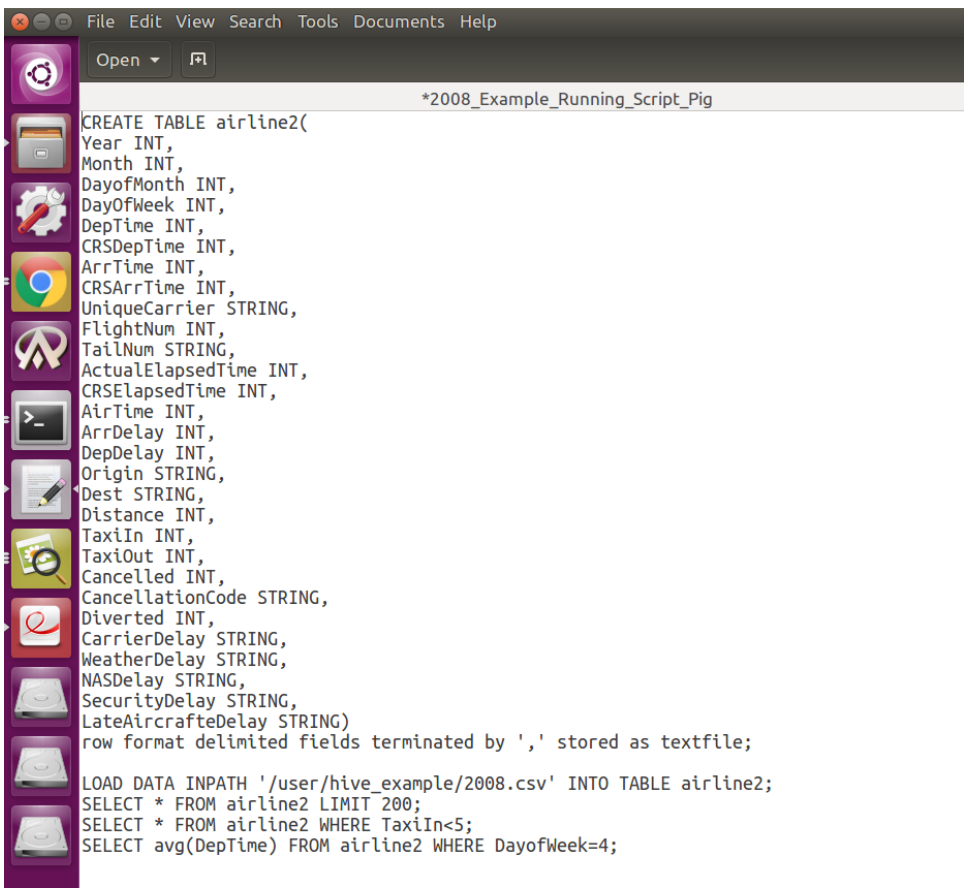
```
nan@nan-ThundeRobot: ~/.linuxbrew/Cellar/hive/2.1.0/libexec/bin
0       0       0       0       0
2007    12      7       5       802     745     943     854     DL      1122    N937DL  101     69      47      49      17      CAE     ATL     191     13      41      0       0
0       0       49      0       0
2007    12      8       6       802     885     1522    1517    DL      870     N664DN  260     252     237     5       -3      PDX     CVG     1975    11      12      0       0
0       0       0       0       0
2007    12      8       6       802     711     858     828     DL      1255    N914DL  116     137     96      30      51      ATL     IAH     689     4       16      0       0
30      0       0       0       0
2007    12      9       7       802     810     1138    1138    DL      983     N948DL  156     148     126     0       -8      MSP     ATL     906     13      17      0       0
0       0       0       0       0
2007    12      10      1       802     800     1039    1008    DL      619     N919DL  157     128     101     31      2       DTW     ATL     594     8       48      0       0
0       0       31      0       0
2007    12      10      1       802     810     1132    1138    DL      983     N917DE  150     148     124     -6      -8      MSP     ATL     906     8       18      0       0
0       0       0       0       0
2007    12      11      2       802     805     1059    1058    DL      1895    N929DL  177     173     154     1       -3      FLL     LGA     1076    8       15      0       0
0       0       0       0       0
2007    12      13      4       802     805     954     1004    DL      1623    N638DL  112     119     86      -10     -3      FLL     ATL     581     11      15      0       0
0       0       0       0       0
2007    12      14      5       802     800     1442    1447    DL      22      N135DL  220     227     199     -5      2       LAS     ATL     1747    8       13      0       0
0       0       0       0       0
2007    12      14      5       802     755     1147    1105    DL      1711    N982DL  225     190     161     42      7       LGA     PBI     1035    3       61      0       0
0       7       35      0       0
2007    12      14      5       802     815     1105    1131    DL      1880    N931DL  183     196     155     -26     -13     JFK     TPA     1005    3       25      0       0
0       0       0       0       0
Time taken: 0.318 seconds, Fetched: 6600 row(s)
hive> SELECT avg(DepTime) FROM airline WHERE Month=2;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = nan_20160926191418_fd7703df-acc2-4ac4-a1a3-64fb2d1832c1
Total jobs = 1
Launching Job 1 out of 1
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_1474925334805_0005, Tracking URL = http://nan-ThundeRobot:8088/proxy/application_1474925334805_0005/
Kill Command = /usr/local/Cellar/hadoop/2.7.3/bin/hadoop job  -kill job_1474925334805_0005
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 1
2016-09-26 19:14:25,232 Stage-1 map = 0%,  reduce = 0%
2016-09-26 19:14:33,854 Stage-1 map = 33%,  reduce = 0%, Cumulative CPU 7.33 sec
2016-09-26 19:14:34,890 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 25.81 sec
2016-09-26 19:14:40,218 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 28.8 sec
MapReduce Total cumulative CPU time: 28 seconds 800 msec
Ended Job = job_1474925334805_0005
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 1   Cumulative CPU: 28.8 sec   HDFS Read: 70292541S HDFS Write: 118 SUCCESS
Total MapReduce CPU Time Spent: 28 seconds 800 msec
OK
1347.4624698457249
Time taken: 23.578 seconds, Fetched: 1 row(s)
hive>
```

a. Running Example on 2008 Airline Data



```
nan@nan-ThundeRobot: ~/.linuxbrew/Cellar/hive/2.1.0/libexec/bin
NA      NA      NA      NA      NA
2008    12      13      6       1935    1940    2146    2202    DL      1535    N647DL  311     322     293     -16     -5      ATL     SMF     2092    3       15      0       0
NA      NA      NA      NA      NA
2008    12      13      6       1904    1900    2111    2115    DL      1537    N3730B  247     255     230     -4      4       ATL     TUS     1541    4       13      0       0
NA      NA      NA      NA      NA
2008    12      13      6       1025    950     1218    1130    DL      1554    N979DL  113     100     68      48      35      ATL     ORF     516     4       41      0       0
35      0       13      0       0
2008    12      13      6       1147    1150    1214    1228    DL      1569    N667DN  87      98      72      -14     -3      ATL     MSY     425     4       11      0       0
NA      NA      NA      NA      NA
2008    12      13      6       835     835     1008    947     DL      1571    N703TW  93      72      38      21      0       ATL     JAX     270     4       51      0       0
0       0       21      0       0
2008    12      13      6       1009    1015    1156    1155    DL      1574    N992DL  107     100     68      1       -6      ATL     DAY     432     4       35      0       0
NA      NA      NA      NA      NA
2008    12      13      6       2006    1954    2133    2109    DL      1586    N3735D  87      75      50      24      12      SLC     BOI     291     4       33      0       0
0       0       12      0       12
2008    12      13      6       1555    1555    1545    1548    DL      1590    N918DL  50      53      30      -3      0       ATL     BHM     134     4       16      0       0
NA      NA      NA      NA      NA
2008    12      13      6       1528    1500    1720    1642    DL      1611    N395DN  112     102     79      38      28      SLC     PHX     507     4       29      0       0
116     0       30      0       12
2008    12      13      6       848     850     1024    1005    DL      1628    N920DL  156     135     108     19      -2      ATL     MCI     692     4       44      0       0
0       0       19      0       0
2008    12      13      6       936     936     1114    1119    DL      1630    N653DL  98      103     70      -5      0       ATL     RSW     515     4       24      0       0
NA      NA      NA      NA      NA
Time taken: 8.385 seconds, Fetched: 2232684 row(s)
hive> SELECT avg(DepTime) FROM airline2 WHERE DayofWeek=4;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = nan_20160927193141_99f52a5f-da76-401b-971d-777199ff8aae
Total jobs = 1
Launching Job 1 out of 1
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_1475018082755_0001, Tracking URL = http://nan-ThundeRobot:8088/proxy/application_1475018082755_0001/
Kill Command = /usr/local/Cellar/hadoop/2.7.3/bin/hadoop job  -kill job_1475018082755_0001
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 1
2016-09-27 19:31:49,159 Stage-1 map = 0%,  reduce = 0%
2016-09-27 19:31:57,567 Stage-1 map = 33%,  reduce = 0%, Cumulative CPU 7.26 sec
2016-09-27 19:31:58,599 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 23.73 sec
2016-09-27 19:32:02,763 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 25.84 sec
MapReduce Total cumulative CPU time: 25 seconds 840 msec
Ended Job = job_1475018082755_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 1   Cumulative CPU: 25.84 sec   HDFS Read: 689460819 HDFS Write: 118 SUCCESS
Total MapReduce CPU Time Spent: 25 seconds 840 msec
OK
1332.8735162202925
Time taken: 22.463 seconds, Fetched: 1 row(s)
hive>
```

Below is the scrip I ran:



```
CREATE TABLE airline2(
Year INT,
Month INT,
DayofMonth INT,
DayOfWeek INT,
DepTime INT,
CRSDepTime INT,
ArrTime INT,
CRSArrTime INT,
UniqueCarrier STRING,
FlightNum INT,
TailNum STRING,
ActualElapsedTime INT,
CRSElapsedTime INT,
AirTime INT,
ArrDelay INT,
DepDelay INT,
Origin STRING,
Dest STRING,
Distance INT,
TaxiIn INT,
TaxiOut INT,
Cancelled INT,
CancellationCode STRING,
Diverted INT,
CarrierDelay STRING,
WeatherDelay STRING,
NASDelay STRING,
SecurityDelay STRING,
LateAircrafteDelay STRING)
row format delimited fields terminated by ',' stored as textfile;

LOAD DATA INPATH '/user/hive_example/2008.csv' INTO TABLE airline2;
SELECT * FROM airline2 LIMIT 200;
SELECT * FROM airline2 WHERE TaxiIn<5;
SELECT avg(DepTime) FROM airline2 WHERE DayofWeek=4;
```

5. Learn to use HBASE
    a. HBASE example

    I build a table named 'test', put in some data.

nan@nan-ThundeRobot: ~/.linuxbrew/Cellar/hbase-1.2.3/bin

```
  hbase> scan 't1', {RAW => true, VERSIONS => 10}

Besides the default 'toStringBinary' format, 'scan' supports custom formatting
by column.  A user can define a FORMATTER by adding it to the column name in
the scan specification.  The FORMATTER can be stipulated:

 1. either as a org.apache.hadoop.hbase.util.Bytes method name (e.g, toInt, toString)
 2. or as a custom class followed by method name: e.g. 'c(MyFormatterClass).format'.

Example formatting cf:qualifier1 and cf:qualifier2 both as Integers:
  hbase> scan 't1', {COLUMNS => ['cf:qualifier1:toInt',
    'cf:qualifier2:c(org.apache.hadoop.hbase.util.Bytes).toInt'] }

Note that you can specify a FORMATTER by column only (cf:qualifier).  You cannot
specify a FORMATTER for all columns of a column family.

Scan can also be used directly from a table, by first getting a reference to a
table, like such:

  hbase> t = get_table 't'
  hbase> t.scan

Note in the above situation, you can still provide all the filtering, columns,
options, etc as described above.


hbase(main):003:0> creat 'test', 'cf'
NoMethodError: undefined method `creat' for #<Object:0x4997552e>

hbase(main):004:0> create 'test', 'cf'
0 row(s) in 1.2490 seconds

=> Hbase::Table - test
hbase(main):005:0> put 'test', 'row1', 'cf:a', 'value1'
0 row(s) in 0.0670 seconds

hbase(main):006:0> put 'test', 'row2', 'cf:b', 'value2'
0 row(s) in 0.0070 seconds

hbase(main):007:0> put 'test', 'row3', 'cf:c', 'value3'
0 row(s) in 0.0020 seconds

hbase(main):008:0> scan 'test'
ROW                                                    COLUMN+CELL
 row1                                                  column=cf:a, timestamp=1475610460902, value=value1
 row2                                                  column=cf:b, timestamp=1475610473951, value=value2
 row3                                                  column=cf:c, timestamp=1475610483904, value=value3
3 row(s) in 0.0200 seconds

hbase(main):009:0>
```