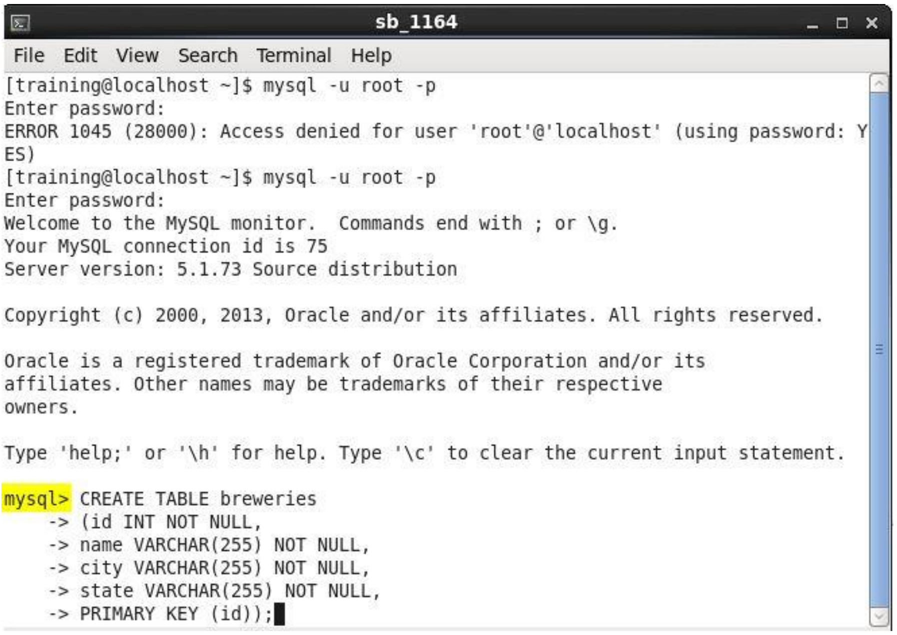
SPARK ESSENTIALS – SUHAIL BARI

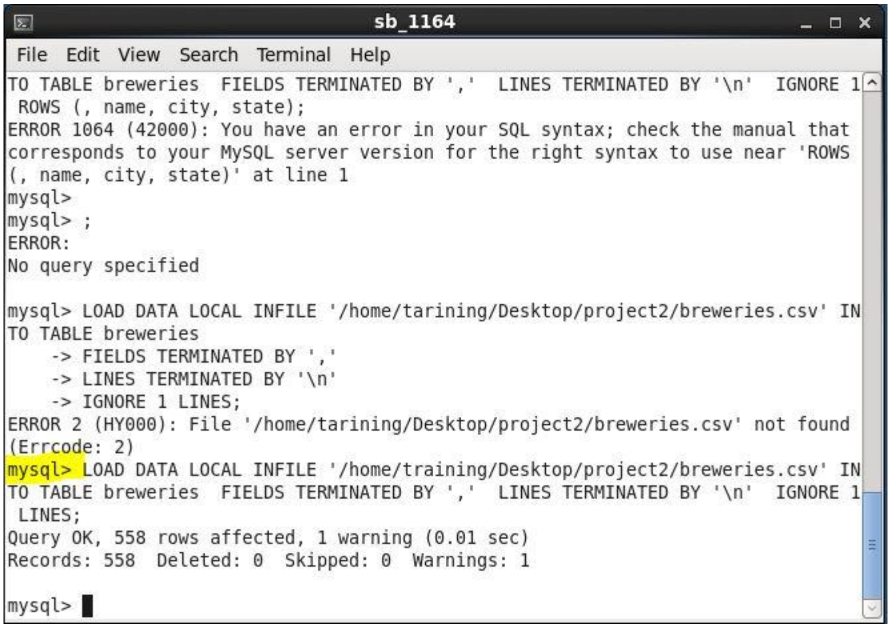
1. AIM - Move the brewery.csv into Spark from Sqoop

   A. Goto SQL command line and create table “breweries” in MYSQL after going through the csv file for datatypes.



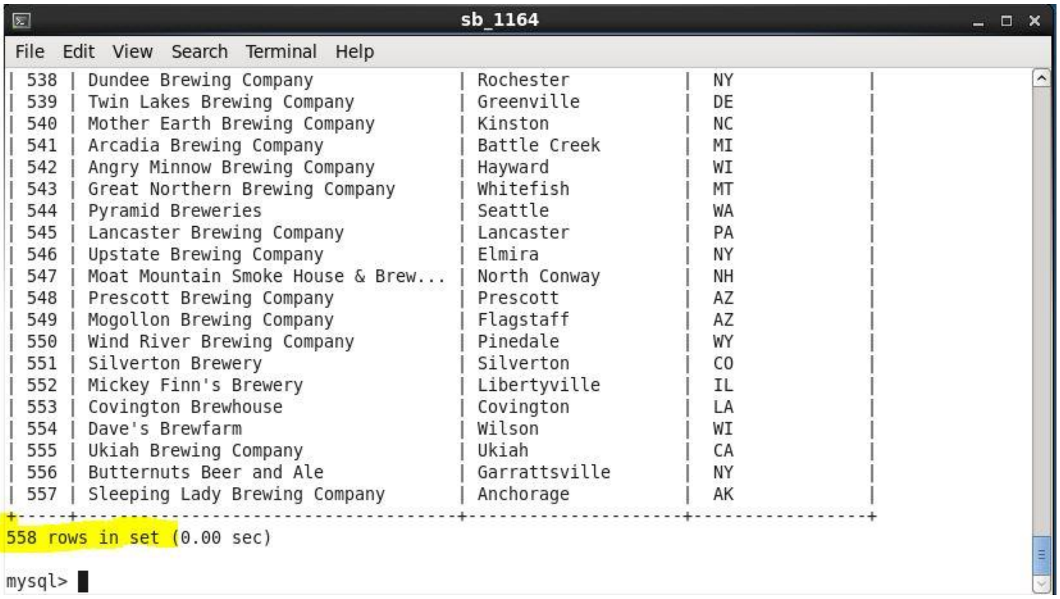
```
[training@localhost ~]$ mysql -u root -p
Enter password:
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: Y
ES)
[training@localhost ~]$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 75
Server version: 5.1.73 Source distribution

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE TABLE breweries
    -> (id INT NOT NULL,
    -> name VARCHAR(255) NOT NULL,
    -> city VARCHAR(255) NOT NULL,
    -> state VARCHAR(255) NOT NULL,
    -> PRIMARY KEY (id));
```

   B. Load the table with data from the CSV file.



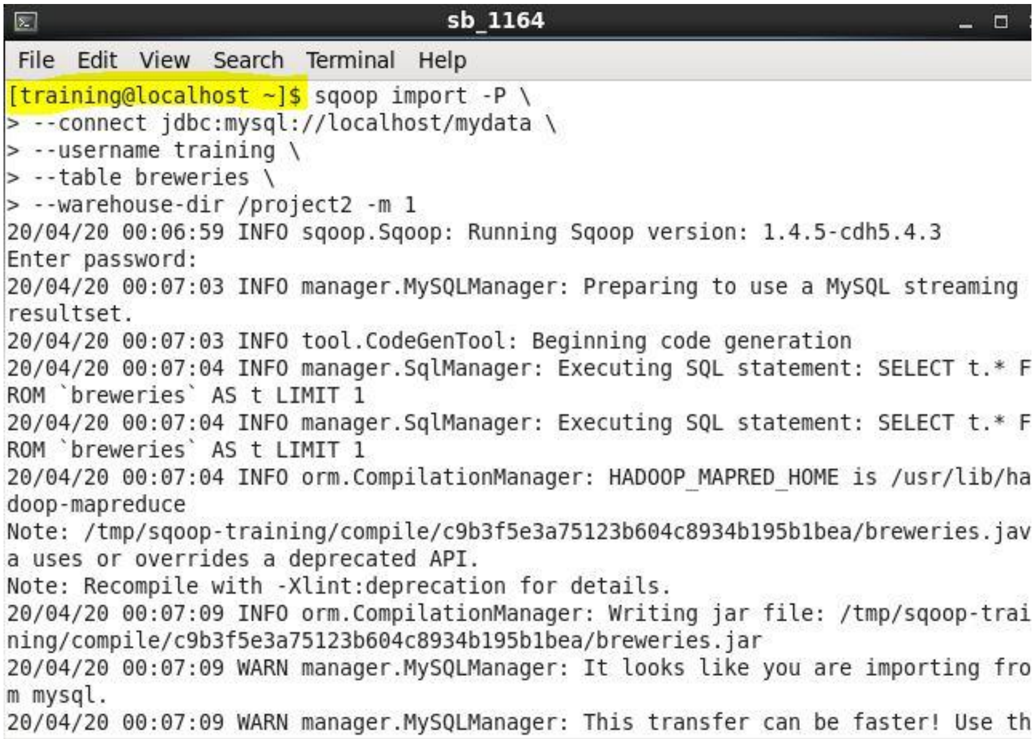
```
TO TABLE breweries  FIELDS TERMINATED BY ','  LINES TERMINATED BY '\n'  IGNORE 1
 ROWS (, name, city, state);
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near 'ROWS
(, name, city, state)' at line 1
mysql>
mysql> ;
ERROR:
No query specified

mysql> LOAD DATA LOCAL INFILE '/home/tarining/Desktop/project2/breweries.csv' IN
TO TABLE breweries
    -> FIELDS TERMINATED BY ','
    -> LINES TERMINATED BY '\n'
    -> IGNORE 1 LINES;
ERROR 2 (HY000): File '/home/tarining/Desktop/project2/breweries.csv' not found
(Errcode: 2)
mysql> LOAD DATA LOCAL INFILE '/home/training/Desktop/project2/breweries.csv' IN
TO TABLE breweries  FIELDS TERMINATED BY ','  LINES TERMINATED BY '\n'  IGNORE 1
 LINES;
Query OK, 558 rows affected, 1 warning (0.01 sec)
Records: 558  Deleted: 0  Skipped: 0  Warnings: 1

mysql> 
```

   C. Verify whether the data was loaded properly. Cross check the number of rows in the csv file.



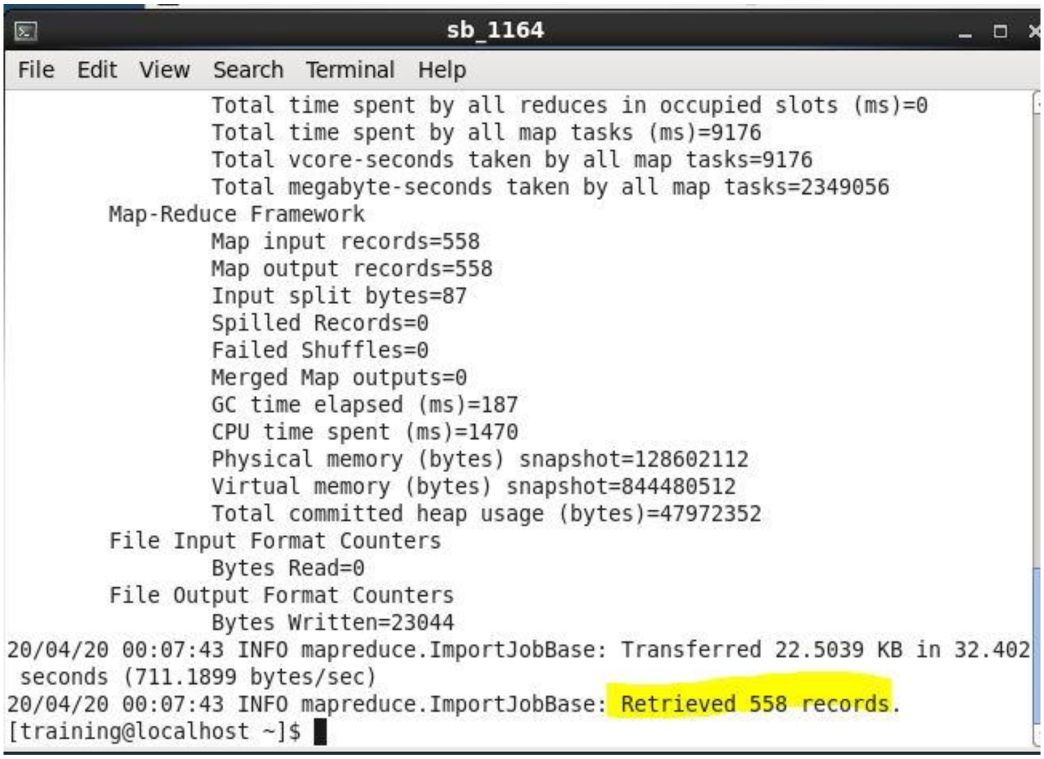
```
| 538 | Dundee Brewing Company              | Rochester           |  NY             |
| 539 | Twin Lakes Brewing Company          | Greenville          |  DE             |
| 540 | Mother Earth Brewing Company        | Kinston             |  NC             |
| 541 | Arcadia Brewing Company             | Battle Creek        |  MI             |
| 542 | Angry Minnow Brewing Company        | Hayward             |  WI             |
| 543 | Great Northern Brewing Company      | Whitefish           |  MT             |
| 544 | Pyramid Breweries                   | Seattle             |  WA             |
| 545 | Lancaster Brewing Company           | Lancaster           |  PA             |
| 546 | Upstate Brewing Company             | Elmira              |  NY             |
| 547 | Moat Mountain Smoke House & Brew... | North Conway        |  NH             |
| 548 | Prescott Brewing Company            | Prescott            |  AZ             |
| 549 | Mogollon Brewing Company            | Flagstaff           |  AZ             |
| 550 | Wind River Brewing Company          | Pinedale            |  WY             |
| 551 | Silverton Brewery                   | Silverton           |  CO             |
| 552 | Mickey Finn's Brewery               | Libertyville        |  IL             |
| 553 | Covington Brewhouse                 | Covington           |  LA             |
| 554 | Dave's Brewfarm                     | Wilson              |  WI             |
| 555 | Ukiah Brewing Company               | Ukiah               |  CA             |
| 556 | Butternuts Beer and Ale             | Garrattsville       |  NY             |
| 557 | Sleeping Lady Brewing Company       | Anchorage           |  AK             |
+-----+-------------------------------------+---------------------+-----------------+
558 rows in set (0.00 sec)

mysql> 
```

D. Goto Linux command line Import the data into HDFS using Sqoop.



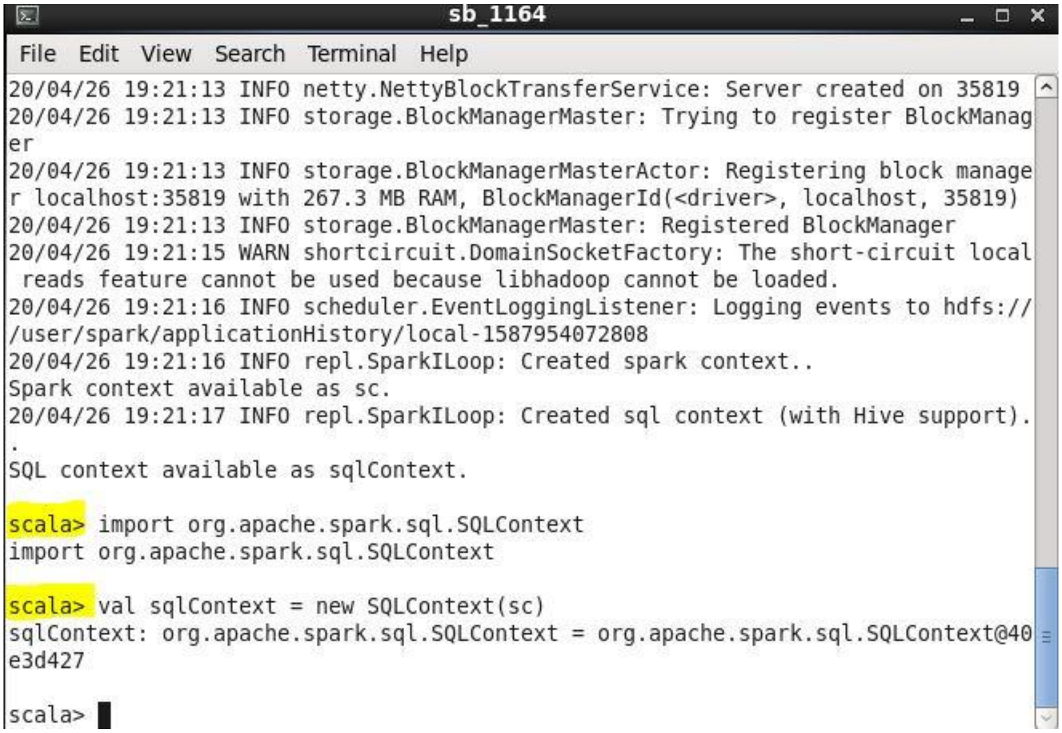
```
[training@localhost ~]$ sqoop import -P \
> --connect jdbc:mysql://localhost/mydata \
> --username training \
> --table breweries \
> --warehouse-dir /project2 -m 1
20/04/20 00:06:59 INFO sqoop.Sqoop: Running Sqoop version: 1.4.5-cdh5.4.3
Enter password:
20/04/20 00:07:03 INFO manager.MySQLManager: Preparing to use a MySQL streaming
resultset.
20/04/20 00:07:03 INFO tool.CodeGenTool: Beginning code generation
20/04/20 00:07:04 INFO manager.SqlManager: Executing SQL statement: SELECT t.* F
ROM `breweries` AS t LIMIT 1
20/04/20 00:07:04 INFO manager.SqlManager: Executing SQL statement: SELECT t.* F
ROM `breweries` AS t LIMIT 1
20/04/20 00:07:04 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /usr/lib/ha
doop-mapreduce
Note: /tmp/sqoop-training/compile/c9b3f5e3a75123b604c8934b195b1bea/breweries.jav
a uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
20/04/20 00:07:09 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-trai
ning/compile/c9b3f5e3a75123b604c8934b195b1bea/breweries.jar
20/04/20 00:07:09 WARN manager.MySQLManager: It looks like you are importing fro
m mysql.
20/04/20 00:07:09 WARN manager.MySQLManager: This transfer can be faster! Use th
```

E. Verify whether the import was a success or not.



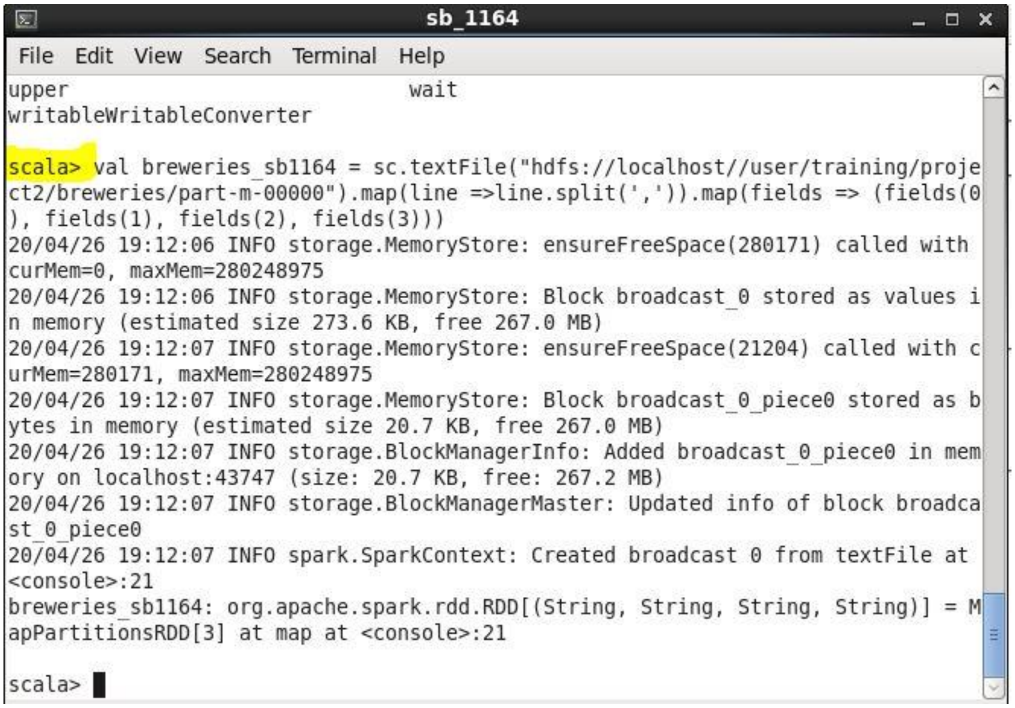
```
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=9176
                Total vcore-seconds taken by all map tasks=9176
                Total megabyte-seconds taken by all map tasks=2349056
        Map-Reduce Framework
                Map input records=558
                Map output records=558
                Input split bytes=87
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=187
                CPU time spent (ms)=1470
                Physical memory (bytes) snapshot=128602112
                Virtual memory (bytes) snapshot=844480512
                Total committed heap usage (bytes)=47972352
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=23044
20/04/20 00:07:43 INFO mapreduce.ImportJobBase: Transferred 22.5039 KB in 32.402
 seconds (711.1899 bytes/sec)
20/04/20 00:07:43 INFO mapreduce.ImportJobBase: Retrieved 558 records.
[training@localhost ~]$ 
```

F. Goto Scala command line by “spark-shell” and create variable sqlContext.



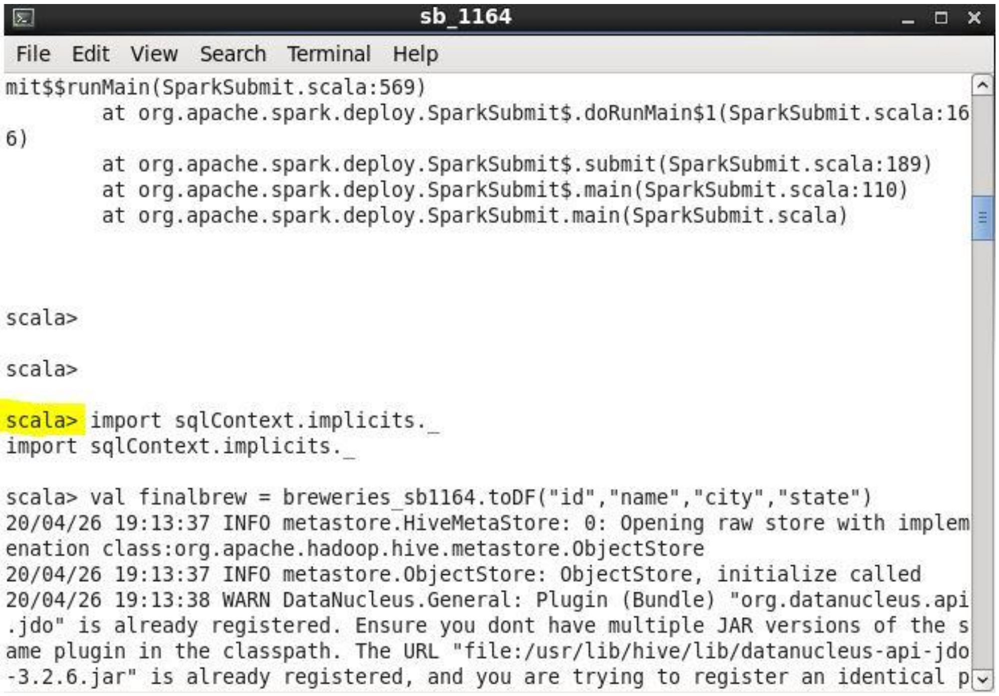
```
20/04/26 19:21:13 INFO netty.NettyBlockTransferService: Server created on 35819
20/04/26 19:21:13 INFO storage.BlockManagerMaster: Trying to register BlockManag
er
20/04/26 19:21:13 INFO storage.BlockManagerMasterActor: Registering block manage
r localhost:35819 with 267.3 MB RAM, BlockManagerId(<driver>, localhost, 35819)
20/04/26 19:21:13 INFO storage.BlockManagerMaster: Registered BlockManager
20/04/26 19:21:15 WARN shortcircuit.DomainSocketFactory: The short-circuit local
 reads feature cannot be used because libhadoop cannot be loaded.
20/04/26 19:21:16 INFO scheduler.EventLoggingListener: Logging events to hdfs://
/user/spark/applicationHistory/local-1587954072808
20/04/26 19:21:16 INFO repl.SparkILoop: Created spark context..
Spark context available as sc.
20/04/26 19:21:17 INFO repl.SparkILoop: Created sql context (with Hive support).
.
SQL context available as sqlContext.

scala> import org.apache.spark.sql.SQLContext
import org.apache.spark.sql.SQLContext

scala> val sqlContext = new SQLContext(sc)
sqlContext: org.apache.spark.sql.SQLContext = org.apache.spark.sql.SQLContext@40
e3d427

scala> 
```

G. Create an RDD “breweries_sb1164” and load it with from the HDFS location where the table “breweries” is stored. Map it so that the rows are split by ‘,’ and fields are displayed as fields(0), fields(1), fields(2), fields(3).



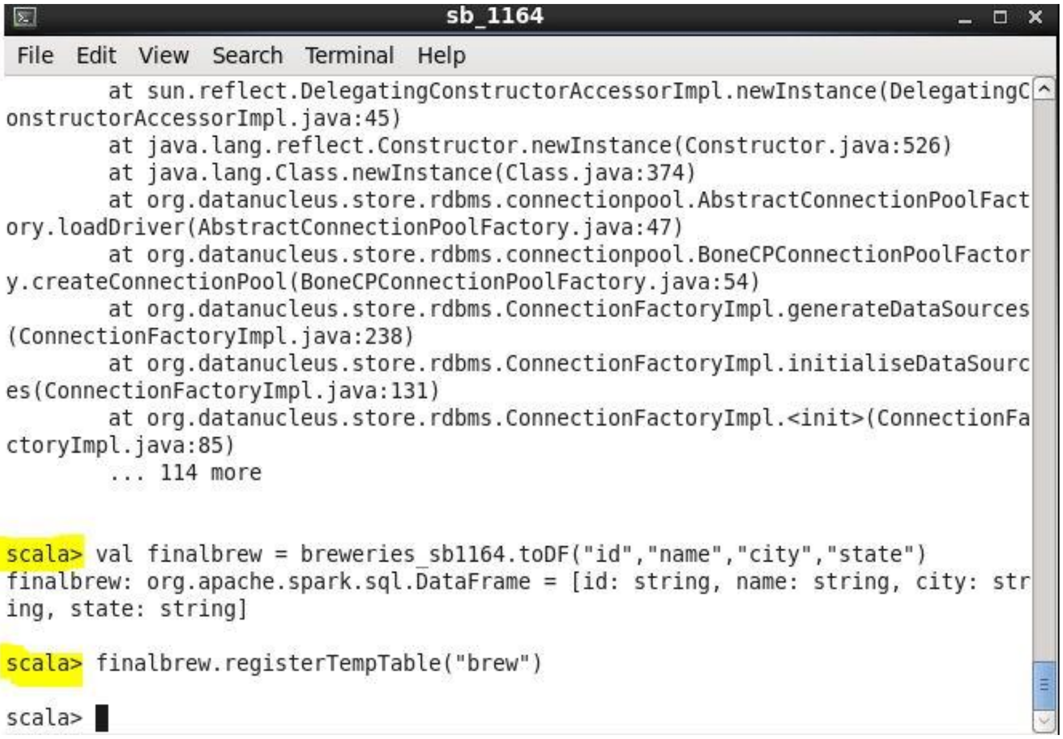
```
upper                            wait
writableWritableConverter

scala> val breweries_sb1164 = sc.textFile("hdfs://localhost//user/training/proje
ct2/breweries/part-m-00000").map(line =>line.split(',')).map(fields => (fields(0
), fields(1), fields(2), fields(3)))
20/04/26 19:12:06 INFO storage.MemoryStore: ensureFreeSpace(280171) called with
curMem=0, maxMem=280248975
20/04/26 19:12:06 INFO storage.MemoryStore: Block broadcast_0 stored as values i
n memory (estimated size 273.6 KB, free 267.0 MB)
20/04/26 19:12:07 INFO storage.MemoryStore: ensureFreeSpace(21204) called with c
urMem=280171, maxMem=280248975
20/04/26 19:12:07 INFO storage.MemoryStore: Block broadcast_0_piece0 stored as b
ytes in memory (estimated size 20.7 KB, free 267.0 MB)
20/04/26 19:12:07 INFO storage.BlockManagerInfo: Added broadcast_0_piece0 in mem
ory on localhost:43747 (size: 20.7 KB, free: 267.2 MB)
20/04/26 19:12:07 INFO storage.BlockManagerMaster: Updated info of block broadca
st_0_piece0
20/04/26 19:12:07 INFO spark.SparkContext: Created broadcast 0 from textFile at
<console>:21
breweries_sb1164: org.apache.spark.rdd.RDD[(String, String, String, String)] = M
apPartitionsRDD[3] at map at <console>:21

scala> █
```

H. Import sqlContext.implicits._



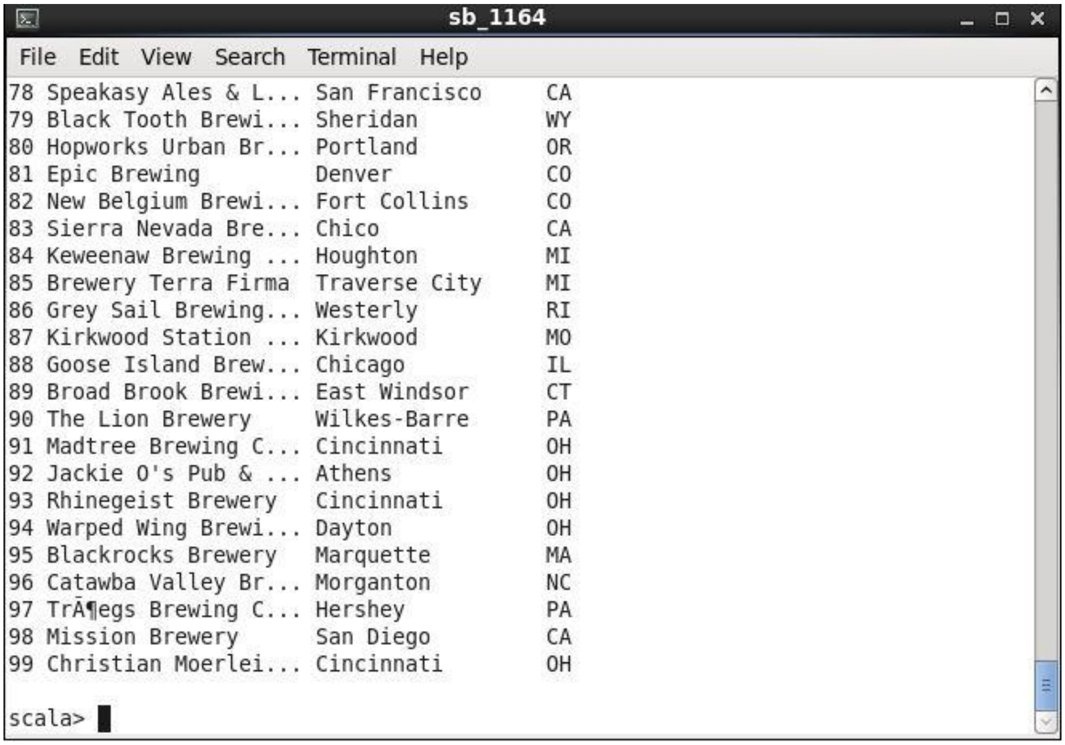
```
mit$$runMain(SparkSubmit.scala:569)
        at org.apache.spark.deploy.SparkSubmit$.doRunMain$1(SparkSubmit.scala:16
6)
        at org.apache.spark.deploy.SparkSubmit$.submit(SparkSubmit.scala:189)
        at org.apache.spark.deploy.SparkSubmit$.main(SparkSubmit.scala:110)
        at org.apache.spark.deploy.SparkSubmit.main(SparkSubmit.scala)



scala>

scala>

scala> import sqlContext.implicits._
import sqlContext.implicits._

scala> val finalbrew = breweries_sb1164.toDF("id","name","city","state")
20/04/26 19:13:37 INFO metastore.HiveMetaStore: 0: Opening raw store with implem
enation class:org.apache.hadoop.hive.metastore.ObjectStore
20/04/26 19:13:37 INFO metastore.ObjectStore: ObjectStore, initialize called
20/04/26 19:13:38 WARN DataNucleus.General: Plugin (Bundle) "org.datanucleus.api
.jdo" is already registered. Ensure you dont have multiple JAR versions of the s
ame plugin in the classpath. The URL "file:/usr/lib/hive/lib/datanucleus-api-jdo
-3.2.6.jar" is already registered, and you are trying to register an identical p
```

I. Convert the RDD to a Dataframe by using toDF and rename the columns as id,name,city,state. Create a temp table “brew”



```
        at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingC
onstructorAccessorImpl.java:45)
        at java.lang.reflect.Constructor.newInstance(Constructor.java:526)
        at java.lang.Class.newInstance(Class.java:374)
        at org.datanucleus.store.rdbms.connectionpool.AbstractConnectionPoolFact
ory.loadDriver(AbstractConnectionPoolFactory.java:47)
        at org.datanucleus.store.rdbms.connectionpool.BoneCPConnectionPoolFactor
y.createConnectionPool(BoneCPConnectionPoolFactory.java:54)
        at org.datanucleus.store.rdbms.ConnectionFactoryImpl.generateDataSources
(ConnectionFactoryImpl.java:238)
        at org.datanucleus.store.rdbms.ConnectionFactoryImpl.initialiseDataSourc
es(ConnectionFactoryImpl.java:131)
        at org.datanucleus.store.rdbms.ConnectionFactoryImpl.<init>(ConnectionFa
ctoryImpl.java:85)
        ... 114 more


scala> val finalbrew = breweries_sb1164.toDF("id","name","city","state")
finalbrew: org.apache.spark.sql.DataFrame = [id: string, name: string, city: str
ing, state: string]

scala> finalbrew.registerTempTable("brew")

scala>
```

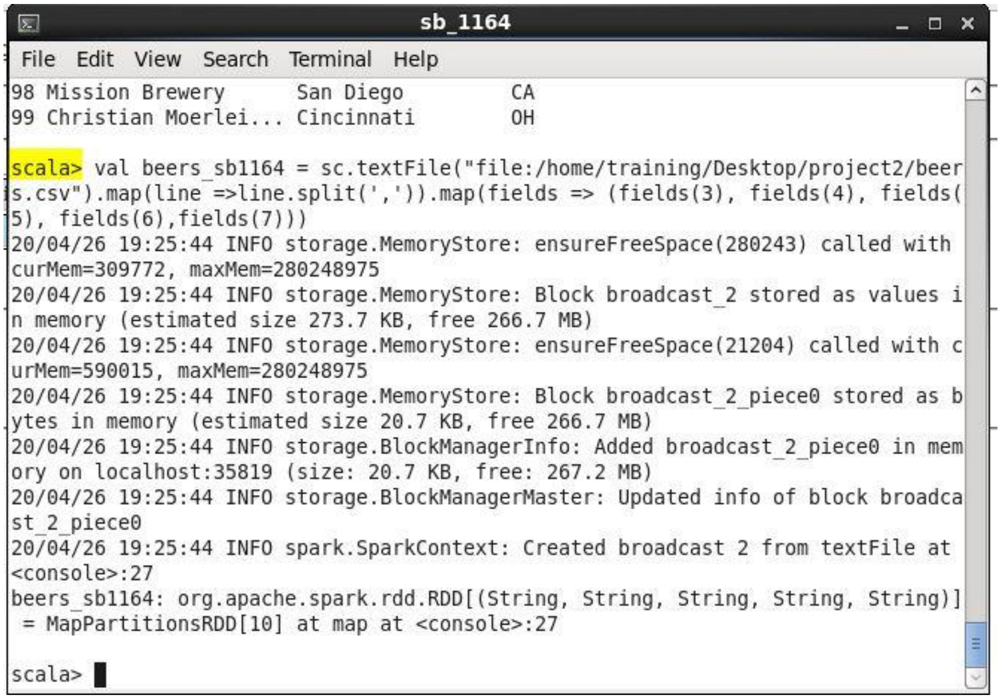
J. Verify whether the temp table has all the data by command show(100)



```
78 Speakasy Ales & L... San Francisco    CA
79 Black Tooth Brewi... Sheridan         WY
80 Hopworks Urban Br... Portland         OR
81 Epic Brewing         Denver           CO
82 New Belgium Brewi... Fort Collins     CO
83 Sierra Nevada Bre... Chico            CA
84 Keweenaw Brewing ... Houghton         MI
85 Brewery Terra Firma  Traverse City    MI
86 Grey Sail Brewing... Westerly         RI
87 Kirkwood Station ... Kirkwood         MO
88 Goose Island Brew... Chicago          IL
89 Broad Brook Brewi... East Windsor     CT
90 The Lion Brewery     Wilkes-Barre     PA
91 Madtree Brewing C... Cincinnati       OH
92 Jackie O's Pub & ... Athens           OH
93 Rhinegeist Brewery   Cincinnati       OH
94 Warped Wing Brewi... Dayton           OH
95 Blackrocks Brewery   Marquette        MA
96 Catawba Valley Br... Morganton        NC
97 TrÃ¶egs Brewing C... Hershey          PA
98 Mission Brewery      San Diego        CA
99 Christian Moerlei... Cincinnati       OH

scala>
```

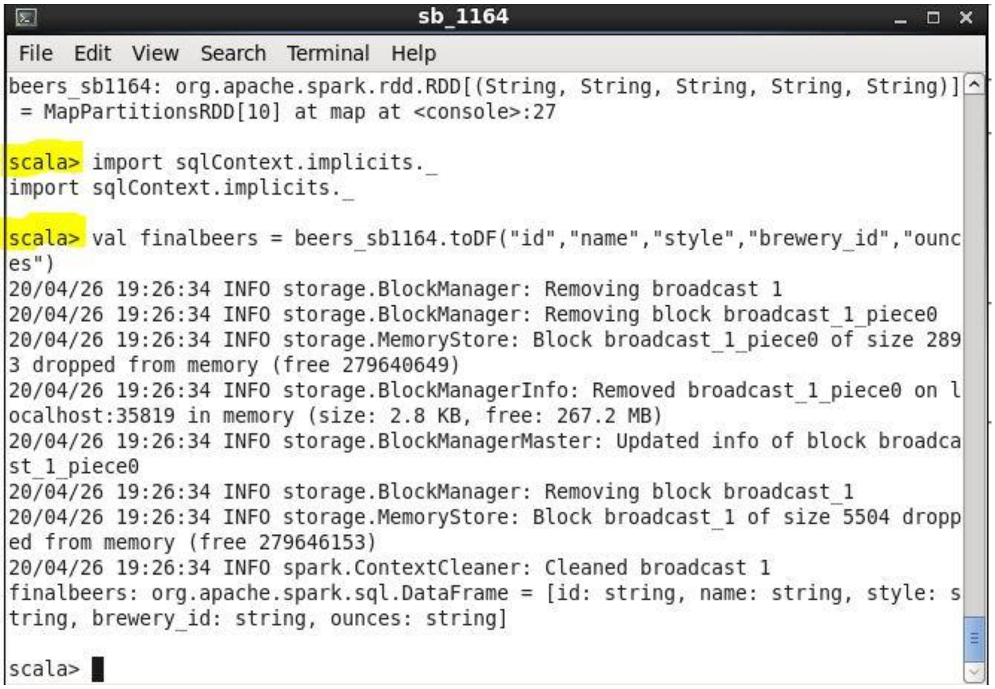
2. Create a RDD for Beers

   A. Create an RDD beer by pulling the file from a local location. Map it so that the rows are split by ‘,’ and fields are displayed as fields(3), fields(4), fields(5), fields(6), fields(7)



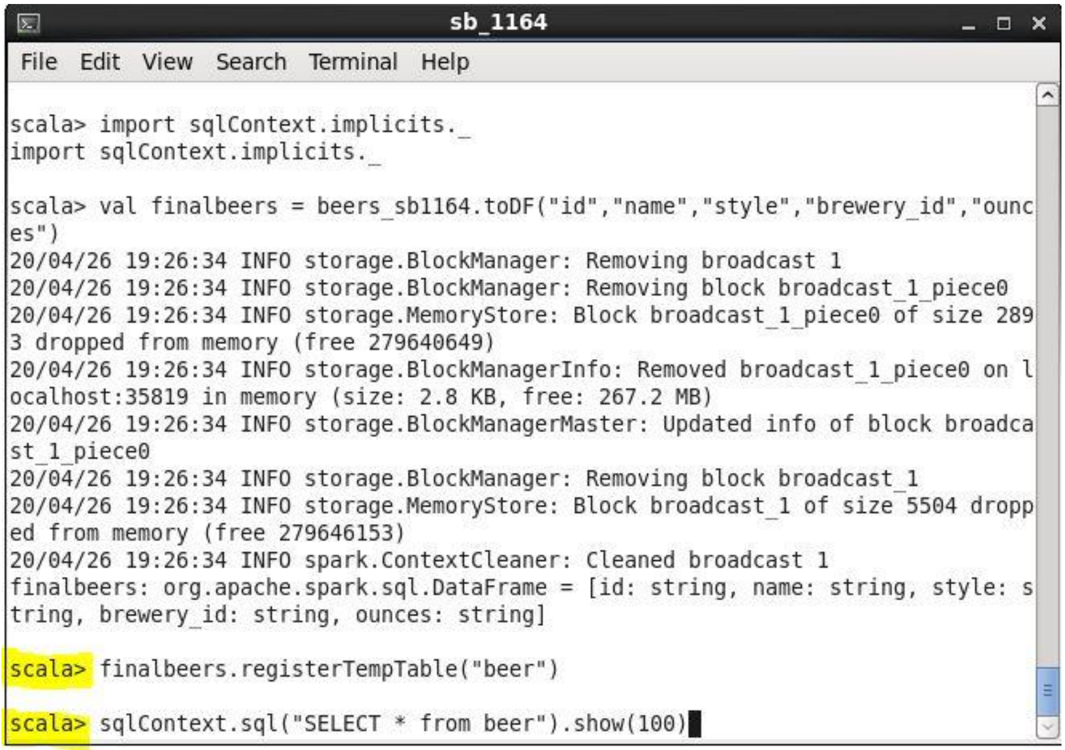
```
98 Mission Brewery      San Diego        CA
99 Christian Moerlei... Cincinnati       OH

scala> val beers_sb1164 = sc.textFile("file:/home/training/Desktop/project2/beer
s.csv").map(line =>line.split(',')).map(fields => (fields(3), fields(4), fields(
5), fields(6),fields(7)))
20/04/26 19:25:44 INFO storage.MemoryStore: ensureFreeSpace(280243) called with
curMem=309772, maxMem=280248975
20/04/26 19:25:44 INFO storage.MemoryStore: Block broadcast_2 stored as values i
n memory (estimated size 273.7 KB, free 266.7 MB)
20/04/26 19:25:44 INFO storage.MemoryStore: ensureFreeSpace(21204) called with c
urMem=590015, maxMem=280248975
20/04/26 19:25:44 INFO storage.MemoryStore: Block broadcast_2_piece0 stored as b
ytes in memory (estimated size 20.7 KB, free 266.7 MB)
20/04/26 19:25:44 INFO storage.BlockManagerInfo: Added broadcast_2_piece0 in mem
ory on localhost:35819 (size: 20.7 KB, free: 267.2 MB)
20/04/26 19:25:44 INFO storage.BlockManagerMaster: Updated info of block broadca
st_2_piece0
20/04/26 19:25:44 INFO spark.SparkContext: Created broadcast 2 from textFile at
<console>:27
beers_sb1164: org.apache.spark.rdd.RDD[(String, String, String, String, String)]
 = MapPartitionsRDD[10] at map at <console>:27

scala> 
```

   B. Import sqlContext.implicits._ and convert the Rdd to a Dataframe. Rename columns to id, name, style, brewery_id, ounces.



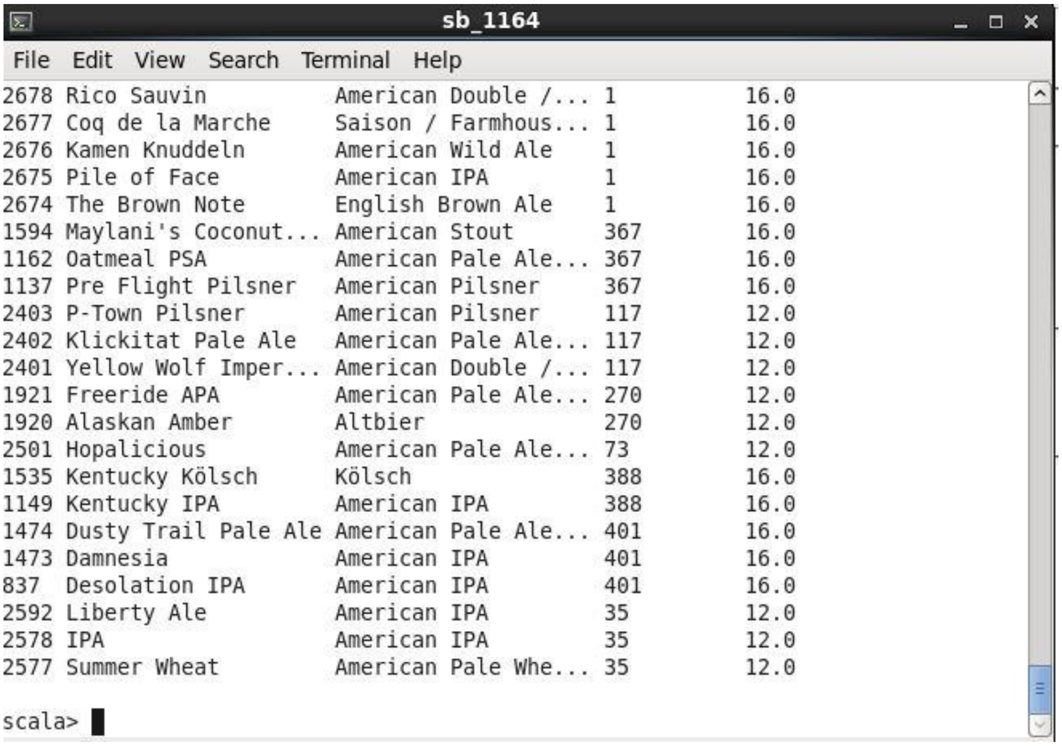
```
beers_sb1164: org.apache.spark.rdd.RDD[(String, String, String, String, String)]
 = MapPartitionsRDD[10] at map at <console>:27

scala> import sqlContext.implicits._
import sqlContext.implicits._

scala> val finalbeers = beers_sb1164.toDF("id","name","style","brewery_id","ounc
es")
20/04/26 19:26:34 INFO storage.BlockManager: Removing broadcast 1
20/04/26 19:26:34 INFO storage.BlockManager: Removing block broadcast_1_piece0
20/04/26 19:26:34 INFO storage.MemoryStore: Block broadcast_1_piece0 of size 289
3 dropped from memory (free 279640649)
20/04/26 19:26:34 INFO storage.BlockManagerInfo: Removed broadcast_1_piece0 on l
ocalhost:35819 in memory (size: 2.8 KB, free: 267.2 MB)
20/04/26 19:26:34 INFO storage.BlockManagerMaster: Updated info of block broadca
st_1_piece0
20/04/26 19:26:34 INFO storage.BlockManager: Removing block broadcast_1
20/04/26 19:26:34 INFO storage.MemoryStore: Block broadcast_1 of size 5504 dropp
ed from memory (free 279646153)
20/04/26 19:26:34 INFO spark.ContextCleaner: Cleaned broadcast 1
finalbeers: org.apache.spark.sql.DataFrame = [id: string, name: string, style: s
tring, brewery_id: string, ounces: string]

scala> 
```

C. Create a temp table and verify if the data was loaded properly.



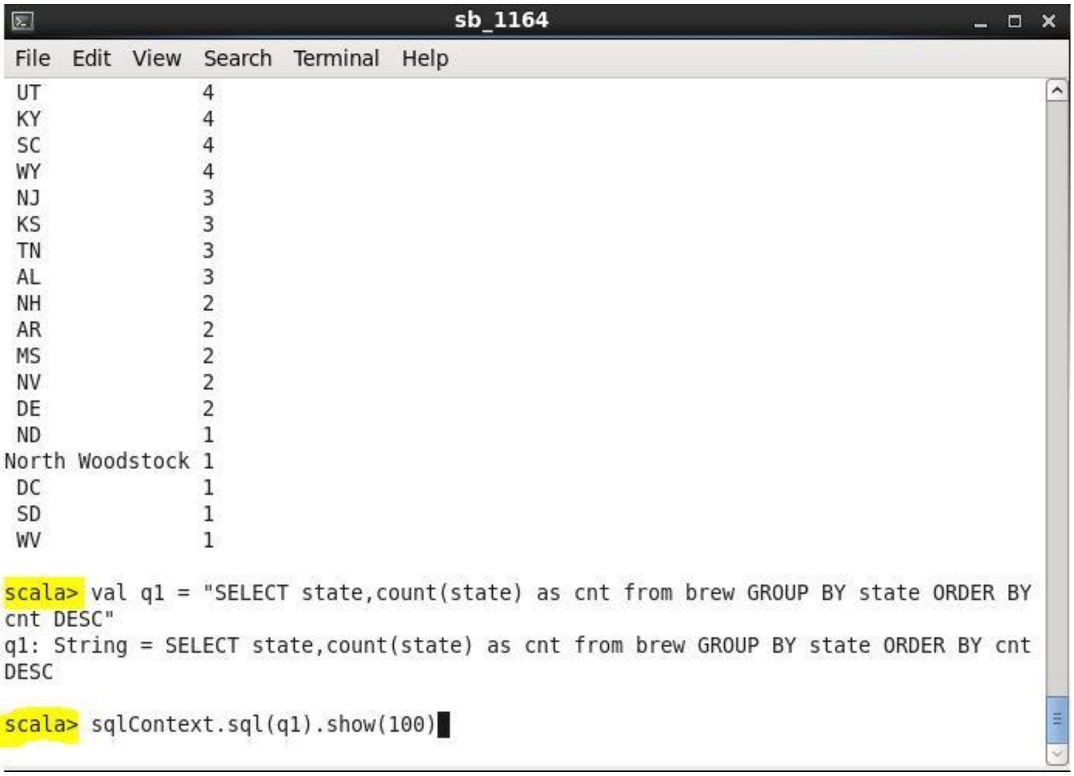
```
scala> import sqlContext.implicits._
import sqlContext.implicits._

scala> val finalbeers = beers_sb1164.toDF("id","name","style","brewery_id","ounc
es")
20/04/26 19:26:34 INFO storage.BlockManager: Removing broadcast 1
20/04/26 19:26:34 INFO storage.BlockManager: Removing block broadcast_1_piece0
20/04/26 19:26:34 INFO storage.MemoryStore: Block broadcast_1_piece0 of size 289
3 dropped from memory (free 279640649)
20/04/26 19:26:34 INFO storage.BlockManagerInfo: Removed broadcast_1_piece0 on l
ocalhost:35819 in memory (size: 2.8 KB, free: 267.2 MB)
20/04/26 19:26:34 INFO storage.BlockManagerMaster: Updated info of block broadca
st_1_piece0
20/04/26 19:26:34 INFO storage.BlockManager: Removing block broadcast_1
20/04/26 19:26:34 INFO storage.MemoryStore: Block broadcast_1 of size 5504 dropp
ed from memory (free 279646153)
20/04/26 19:26:34 INFO spark.ContextCleaner: Cleaned broadcast 1
finalbeers: org.apache.spark.sql.DataFrame = [id: string, name: string, style: s
tring, brewery_id: string, ounces: string]

scala> finalbeers.registerTempTable("beer")

scala> sqlContext.sql("SELECT * from beer").show(100)
```

D. Verified.

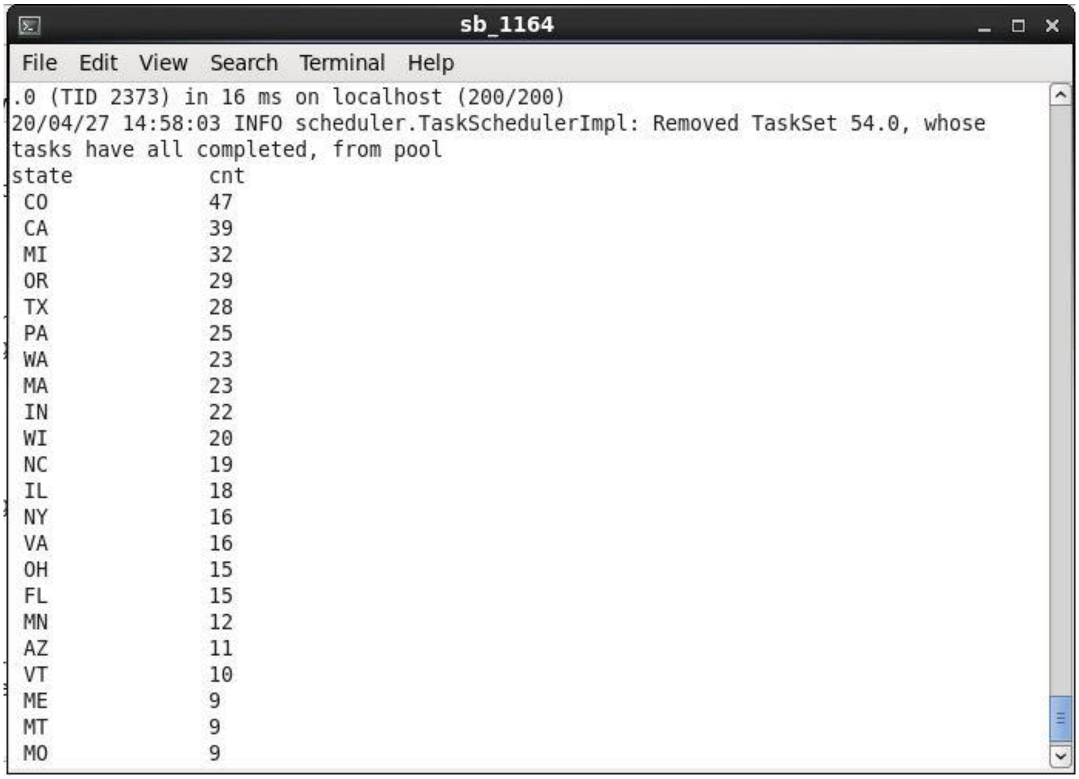


```
2678 Rico Sauvin          American Double /... 1          16.0
2677 Coq de la Marche     Saison / Farmhous... 1          16.0
2676 Kamen Knuddeln       American Wild Ale    1          16.0
2675 Pile of Face         American IPA         1          16.0
2674 The Brown Note       English Brown Ale    1          16.0
1594 Maylani's Coconut... American Stout       367        16.0
1162 Oatmeal PSA          American Pale Ale... 367        16.0
1137 Pre Flight Pilsner   American Pilsner     367        16.0
2403 P-Town Pilsner       American Pilsner     117        12.0
2402 Klickitat Pale Ale   American Pale Ale... 117        12.0
2401 Yellow Wolf Imper... American Double /... 117        12.0
1921 Freeride APA         American Pale Ale... 270        12.0
1920 Alaskan Amber        Altbier              270        12.0
2501 Hopalicious          American Pale Ale... 73         12.0
1535 Kentucky Kölsch      Kölsch               388        16.0
1149 Kentucky IPA         American IPA         388        16.0
1474 Dusty Trail Pale Ale American Pale Ale... 401        16.0
1473 Damnesia             American IPA         401        16.0
837  Desolation IPA       American IPA         401        16.0
2592 Liberty Ale          American IPA         35         12.0
2578 IPA                  American IPA         35         12.0
2577 Summer Wheat         American Pale Whe... 35         12.0

scala> 
```

3. Use SqarkQL to determine the answers
   A. Determine the number of breweries in each state.

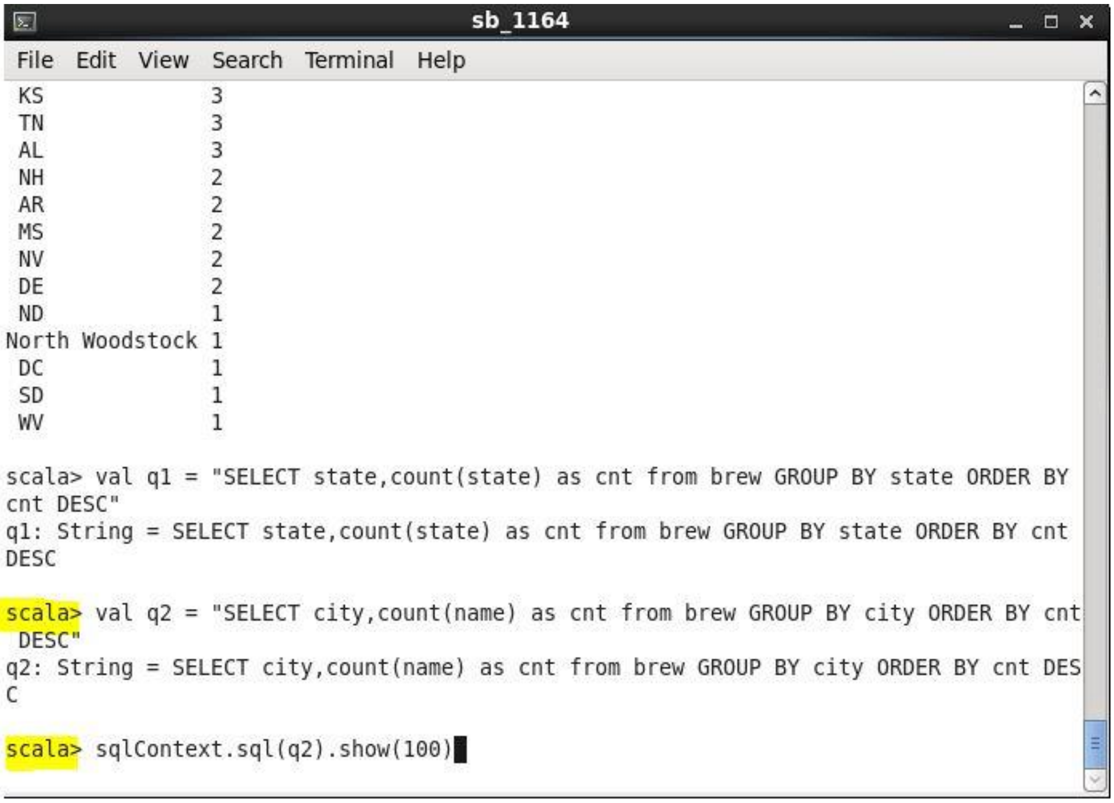


```
 UT             4
 KY             4
 SC             4
 WY             4
 NJ             3
 KS             3
 TN             3
 AL             3
 NH             2
 AR             2
 MS             2
 NV             2
 DE             2
 ND             1
North Woodstock 1
 DC             1
 SD             1
 WV             1

scala> val q1 = "SELECT state,count(state) as cnt from brew GROUP BY state ORDER BY
cnt DESC"
q1: String = SELECT state,count(state) as cnt from brew GROUP BY state ORDER BY cnt
DESC

scala> sqlContext.sql(q1).show(100)
```

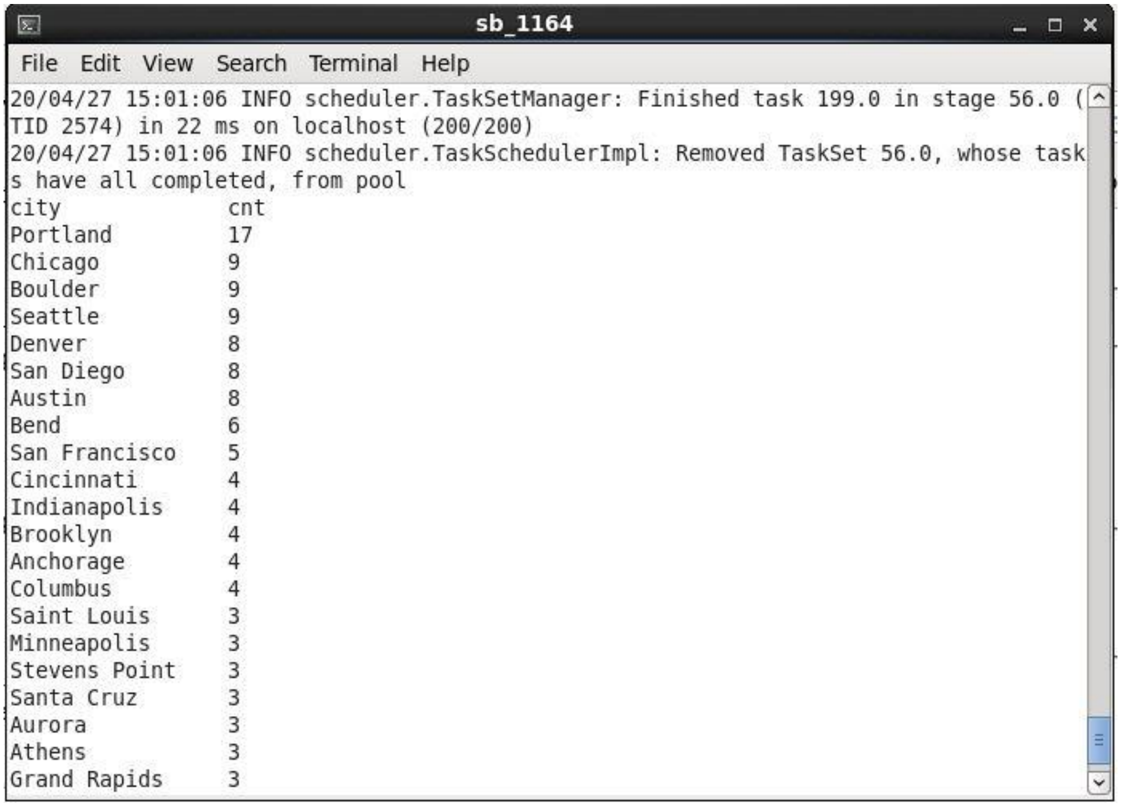


```
.0 (TID 2373) in 16 ms on localhost (200/200)
20/04/27 14:58:03 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 54.0, whose
tasks have all completed, from pool
state           cnt
 CO             47
 CA             39
 MI             32
 OR             29
 TX             28
 PA             25
 WA             23
 MA             23
 IN             22
 WI             20
 NC             19
 IL             18
 NY             16
 VA             16
 OH             15
 FL             15
 MN             12
 AZ             11
 VT             10
 ME             9
 MT             9
 MO             9
```

B. Determine the cities with most breweries

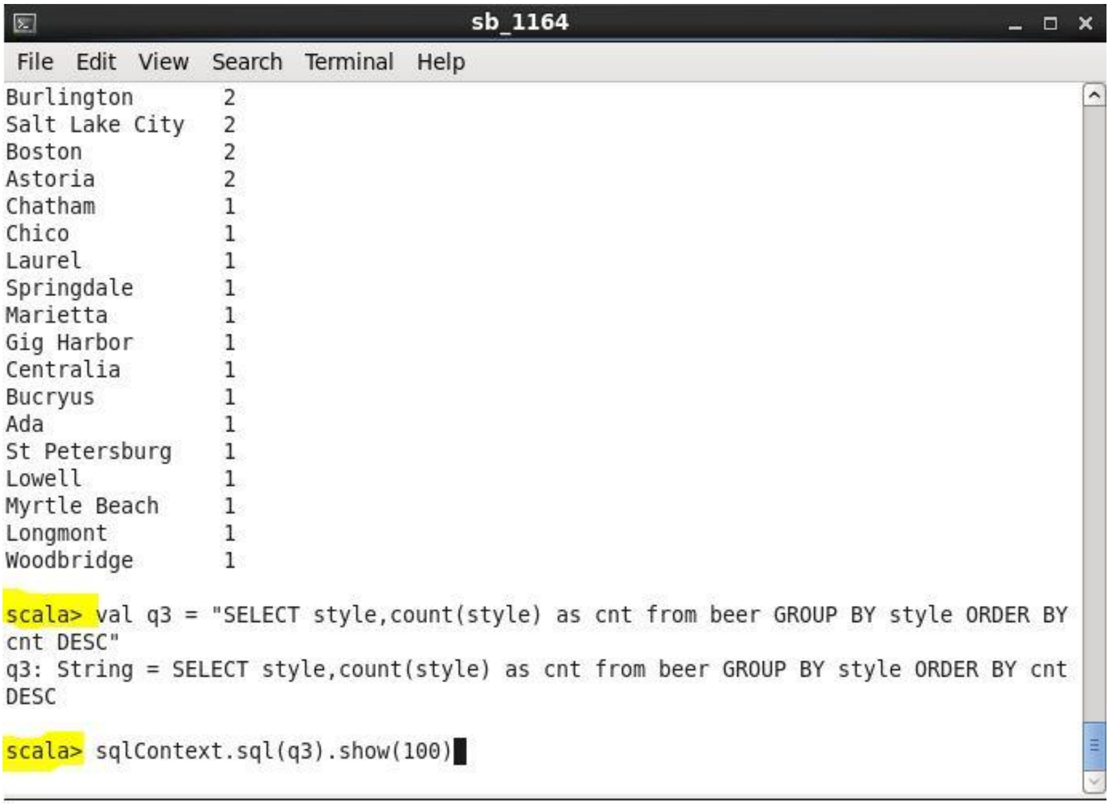


```
 KS             3
 TN             3
 AL             3
 NH             2
 AR             2
 MS             2
 NV             2
 DE             2
 ND             1
North Woodstock 1
 DC             1
 SD             1
 WV             1

scala> val q1 = "SELECT state,count(state) as cnt from brew GROUP BY state ORDER BY
cnt DESC"
q1: String = SELECT state,count(state) as cnt from brew GROUP BY state ORDER BY cnt
DESC

scala> val q2 = "SELECT city,count(name) as cnt from brew GROUP BY city ORDER BY cnt
 DESC"
q2: String = SELECT city,count(name) as cnt from brew GROUP BY city ORDER BY cnt DES
C

scala> sqlContext.sql(q2).show(100)
```

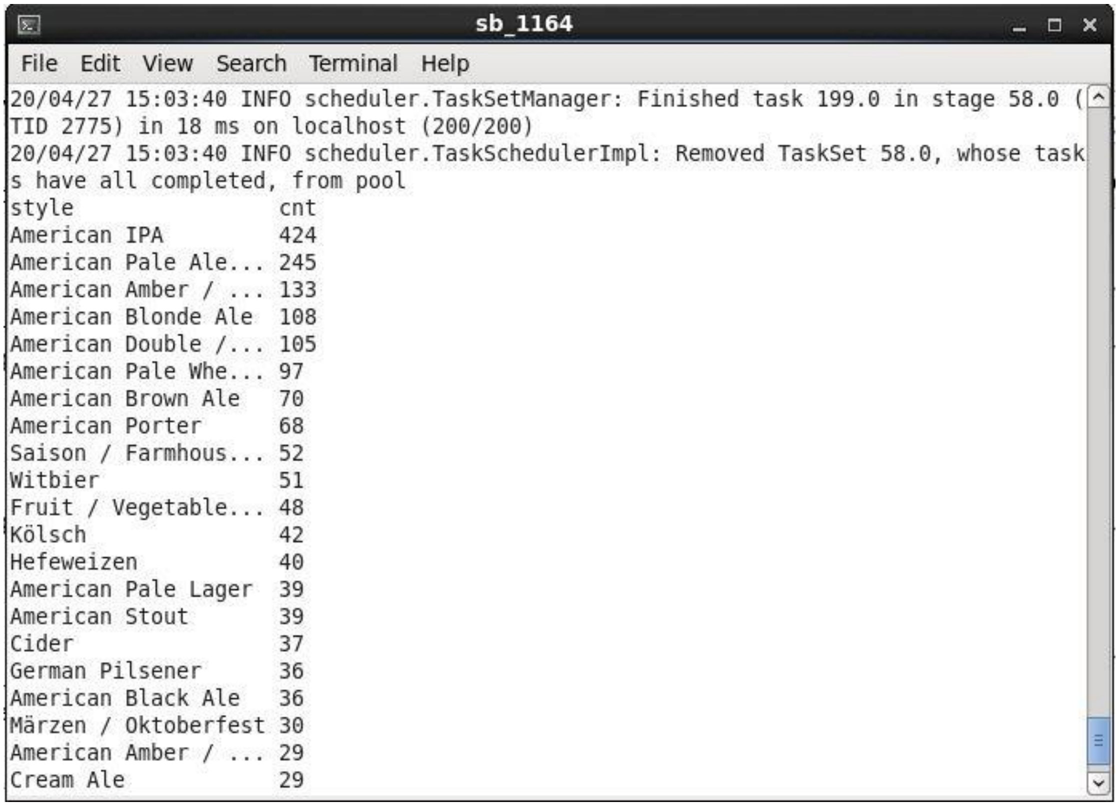


```
20/04/27 15:01:06 INFO scheduler.TaskSetManager: Finished task 199.0 in stage 56.0 (
TID 2574) in 22 ms on localhost (200/200)
20/04/27 15:01:06 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 56.0, whose task
s have all completed, from pool
city             cnt
Portland         17
Chicago          9
Boulder          9
Seattle          9
Denver           8
San Diego        8
Austin           8
Bend             6
San Francisco    5
Cincinnati       4
Indianapolis     4
Brooklyn         4
Anchorage        4
Columbus         4
Saint Louis      3
Minneapolis      3
Stevens Point    3
Santa Cruz       3
Aurora           3
Athens           3
Grand Rapids     3
```

C. Determine the most brewed beer style



```
Burlington       2
Salt Lake City   2
Boston           2
Astoria          2
Chatham          1
Chico            1
Laurel           1
Springdale       1
Marietta         1
Gig Harbor       1
Centralia        1
Bucryus          1
Ada              1
St Petersburg    1
Lowell           1
Myrtle Beach     1
Longmont         1
Woodbridge       1

scala> val q3 = "SELECT style,count(style) as cnt from beer GROUP BY style ORDER BY
cnt DESC"
q3: String = SELECT style,count(style) as cnt from beer GROUP BY style ORDER BY cnt
DESC

scala> sqlContext.sql(q3).show(100)
```



```
20/04/27 15:03:40 INFO scheduler.TaskSetManager: Finished task 199.0 in stage 58.0 (
TID 2775) in 18 ms on localhost (200/200)
20/04/27 15:03:40 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 58.0, whose task
s have all completed, from pool
style                cnt
American IPA         424
American Pale Ale... 245
American Amber / ... 133
American Blonde Ale  108
American Double /... 105
American Pale Whe... 97
American Brown Ale   70
American Porter      68
Saison / Farmhous... 52
Witbier              51
Fruit / Vegetable... 48
Kölsch               42
Hefeweizen           40
American Pale Lager  39
American Stout       39
Cider                37
German Pilsener      36
American Black Ale   36
Märzen / Oktoberfest 30
American Amber / ... 29
Cream Ale            29
```