1. **Import the brewery.csv into Sqoop**

Step 1: Create table in MySQL

Use loudacre;

CREATE TABLE breweries_export (ID int, NAME VARCHAR(50), CITY VARCHAR(50), STATE VARCHAR(20));

```
mysql> use loudacre;
Database changed
mysql> CREATE TABLE breweries_export (ID int, NAME VARCHAR(50), CITY VARCHAR(50)
, STATE VARCHAR(20));
Query OK, 0 rows affected (0.02 sec)
```

Step 2: Insert data into table (Note: Delete the column names before inserting)
load data local infile '/home/training/Desktop/breweries.csv' into table breweries_export;

```
mysql> load data local infile '/home/training/Desktop/breweries.csv' into table
breweries_export;
Query OK, 558 rows affected, 2232 warnings (0.00 sec)
Records: 558  Deleted: 0  Skipped: 0  Warnings: 2232
```

Step3: Sqoop import
sqoop import --connect jdbc:mysql://localhost/loudacre --username training --password training --table breweries_export -m1 --target-dir /user/training/rawdata/breweries.csv

Contents of directory /user/training/rawdata/breweries.csv

Goto : /user/training/rawdata/brew go

Go to parent directory

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
_SUCCESS	file	0 B	1	128 MB	2017-12-10 18:51	rw-rw-rw-	training	supergroup
part-m-00000	file	10.25 KB	1	128 MB	2017-12-10 18:51	rw-rw-rw-	training	supergroup

Go back to DFS home

2.Create RDD for Beers.csv

```
val beerRDD = sc.textFile("file:/home/training/Desktop/beers.csv")

beerRDD.take(5).foreach(println)
```

Creating data frame for sql computations

Step 1: Load the file into the Hadoop FS into the rawdata directory.

```
hdfs dfs -put Desktop/beers.csv rawdata;
hdfs dfs -cat rawdata/beers.csv;
```

```
557,Sleeping Lady Brewing Company,Anchorage, AK
[training@localhost ~]$ hdfs dfs -put Desktop/beers.csv rawdata;
[training@localhost ~]$ hdfs dfs -cat rawdata/beers.csv;
0,0.05,,1436,Pub Beer,American Pale Lager,408,12.0
1,0.066,,2265,Devil's Cup,American Pale Ale (APA),177,12.0
2,0.071,,2264,Rise of the Phoenix,American IPA,177,12.0
3,0.09,,2263,Sinister,American Double / Imperial IPA,177,12.0
4,0.075,,2262,Sex and Candy,American IPA,177,12.0
5,0.077,,2261,Black Exodus,Oatmeal Stout,177,12.0
6,0.045,,2260,Lake Street Express,American Pale Ale (APA),177,12.0
7,0.065,,2259,Foreman,American Porter,177,12.0
8,0.055,,2258,Jade,American Pale Ale (APA),177,12.0
9,0.086,,2131,Cone Crusher,American Double / Imperial IPA,177,12.0
10,0.07200000000000001,,2099,Sophomoric Saison,Saison / Farmhouse Ale,177,12.0
11,0.073,,2098,Regional Ring Of Fire,Saison / Farmhouse Ale,177,12.0
12,0.069,,2097,Garce Selé,Saison / Farmhouse Ale,177,12.0
13,0.085,,1980,Troll Destroyer,Belgian IPA,177,12.0
14,0.061,60.0,1979,Bitter Bitch,American Pale Ale (APA),177,12.0
15,0.06,,2318,Ginja Ninja,Cider,154,12.0
16,0.06,,2170,Cherried Away,Cider,154,12.0
17,0.06,,2169,Rhubarbarian,Cider,154,12.0
18,0.06,,1502,BrightCider,Cider,154,12.0
19,0.08199999999999999,,1593,He Said Baltic-Style Porter,Baltic Porter,368,12.0
20,0.08199999999999999,,1592,He Said Belgian-Style Tripel,Tripel,368,12.0
```

Step 2: Create and insert data into table in Hive (Note: Delete the column names before inserting)

CREATE TABLE beers (sno int, abv string, ibu string, id int, name string, style string, brewery_id int, ounces double) row format delimited fields terminated by ',';

load data inpath 'rawdata/beers.csv' into table beers;

```
hive> CREATE TABLE beers (sno int, abv string, ibu  string, id int, name string,
 style string, brewery_id int, ounces double) row format delimited fields termin
ated by ',';
OK
Time taken: 0.374 seconds
hive> load data inpath 'rawdata/beers.csv' into table beers;
Loading data to table buan6346.beers
chgrp: changing ownership of 'hdfs://localhost:8020/user/hive/warehouse/buan6346
.db/beers/beers.csv': User does not belong to hive
Table buan6346.beers stats: [numFiles=1, totalSize=159492]
OK
Time taken: 0.55 seconds
```

Step 3: Create table in MySQL

Use loudacre;

drop table beers_export;

CREATE TABLE beers_export (sno int, abv VARCHAR(50), ibu VARCHAR(50), id int, name VARCHAR(100), style VARCHAR(100), brewery_id int, ounces VARCHAR(50));

```
mysql> use loudacre;
Database changed
mysql> drop table beers_export;
Query OK, 0 rows affected (0.00 sec)

mysql> CREATE TABLE beers_export (sno int, abv VARCHAR(50), ibu VARCHAR(50), id
int, name VARCHAR(100), style VARCHAR(100), brewery_id int, ounces VARCHAR(50));
Query OK, 0 rows affected (0.04 sec)
```

Step 4: Import data into MySQL

load data local infile '/home/training/Desktop/beers.csv' into table beers_export fields terminated by ',';

```
mysql> load data local infile '/home/training/Desktop/beers.csv' into table beer
s_export fields terminated by ',';
Query OK, 2410 rows affected, 4 warnings (0.07 sec)
Records: 2410  Deleted: 0  Skipped: 0  Warnings: 2
```

Step 5: Create sql context

import org.apache.spark.sql.SQLContext
val sqlCtx = new SQLContext(sc)
import sqlCtx._

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

scala> val sqlCtx = new SQLContext(sc)
17/12/09 23:23:31 INFO storage.BlockManager: Removing broadcast 1
sqlCtx: org.apache.spark.sql.SQLContext = org.apache.spark.sql.SQLContext@4fdedb58

scala> 17/12/09 23:23:31 INFO storage.BlockManager: Removing block broadcast_1_piece0
17/12/09 23:23:31 INFO storage.MemoryStore: Block broadcast_1_piece0 of size 1668 dropped from memory (free 279634751
)
17/12/09 23:23:31 INFO storage.BlockManagerInfo: Removed broadcast_1_piece0 on localhost:34431 in memory (size: 1668.
0 B, free: 267.2 MB)
17/12/09 23:23:31 INFO storage.BlockManagerMaster: Updated info of block broadcast_1_piece0
17/12/09 23:23:31 INFO storage.BlockManager: Removing block broadcast_1
17/12/09 23:23:31 INFO storage.MemoryStore: Block broadcast_1 of size 2688 dropped from memory (free 279637439)
17/12/09 23:23:31 INFO spark.ContextCleaner: Cleaned broadcast 1
17/12/09 23:23:31 INFO storage.BlockManager: Removing broadcast 2
17/12/09 23:23:31 INFO storage.BlockManager: Removing block broadcast_2
17/12/09 23:23:31 INFO storage.MemoryStore: Block broadcast_2 of size 2688 dropped from memory (free 279640127)
17/12/09 23:23:31 INFO storage.BlockManager: Removing block broadcast_2_piece0
17/12/09 23:23:31 INFO storage.MemoryStore: Block broadcast_2_piece0 of size 1668 dropped from memory (free 279641795
)
17/12/09 23:23:31 INFO storage.BlockManagerInfo: Removed broadcast_2_piece0 on localhost:34431 in memory (size: 1668.
0 B, free: 267.2 MB)
17/12/09 23:23:31 INFO storage.BlockManagerMaster: Updated info of block broadcast_2_piece0
17/12/09 23:23:31 INFO spark.ContextCleaner: Cleaned broadcast 2
17/12/09 23:23:31 INFO storage.BlockManager: Removing broadcast 4
17/12/09 23:23:31 INFO storage.BlockManager: Removing block broadcast_4
17/12/09 23:23:31 INFO storage.MemoryStore: Block broadcast_4 of size 2688 dropped from memory (free 279644483)
17/12/09 23:23:31 INFO storage.BlockManager: Removing block broadcast_4_piece0
17/12/09 23:23:31 INFO storage.MemoryStore: Block broadcast_4_piece0 of size 1670 dropped from memory (free 279646153
)
17/12/09 23:23:31 INFO storage.BlockManagerInfo: Removed broadcast_4_piece0 on localhost:34431 in memory (size: 1670.
0 B, free: 267.2 MB)
17/12/09 23:23:31 INFO storage.BlockManagerMaster: Updated info of block broadcast_4_piece0
17/12/09 23:23:31 INFO spark.ContextCleaner: Cleaned broadcast 4


scala> import sqlCtx._
import sqlCtx._
```

Step 6: Load data from MySQL to spark- Create a new DataFrame based on the breweries_export table from the database and then RDD

val beers=sqlCtx.load("jdbc",Map("url"->"jdbc:mysql://localhost/loudacre?user=training&password=training","dbtable" -> "beers_export"))

beers.registerTempTable("beersA")

```
scala> val beers=sqlCtx.load("jdbc",Map("url"->"jdbc:mysql://localhost/loudacre?
user=training&password=training","dbtable" -> "beers_export"))
beers: org.apache.spark.sql.DataFrame = [sno: int, abv: string, ibu: string, id:
 int, name: string, style: string, brewery_id: int, ounces: string]

scala> beers.registerTempTable("beersA")
```

3.Move the brewery.csv into Spark from Sqoop (intermediate movement to another component in Hadoop is fine)

Moving from sqoop:

Note: Data is already moved into sqoop via MySQL in Q1. We can use the table from MySQL to move the breweries into spark and sql context is already created for Q2.

val brewery = sc.textFile("/user/training/rawdata/breweries.csv/part-m-00000")
brewery.take(5).foreach(println)

```
scala> val brewery = sc.textFile("/user/training/rawdata/breweries.csv/part-m-00000")
17/12/10 20:26:33 INFO storage.MemoryStore: ensureFreeSpace(280243) called with curMem=607665, maxMem=280248975
17/12/10 20:26:33 INFO storage.MemoryStore: Block broadcast_6 stored as values in memory (estimated size 273.7 KB, free 266.4 MB)
17/12/10 20:26:33 INFO storage.MemoryStore: ensureFreeSpace(21204) called with curMem=887908, maxMem=280248975
17/12/10 20:26:33 INFO storage.MemoryStore: Block broadcast_6_piece0 stored as bytes in memory (estimated size 20.7 KB, free 266.4 MB)
17/12/10 20:26:33 INFO storage.BlockManagerInfo: Added broadcast_6_piece0 in memory on localhost:60461 (size: 20.7 KB, free: 267.2 MB)
17/12/10 20:26:33 INFO storage.BlockManagerMaster: Updated info of block broadcast_6_piece0
17/12/10 20:26:33 INFO spark.SparkContext: Created broadcast 6 from textFile at <console>:33
brewery: org.apache.spark.rdd.RDD[String] = /user/training/rawdata/breweries.csv/part-m-00000 MapPartitionsRDD[7] at textFile at <console>:33

scala> brewery.take(5).foreach(println)
17/12/10 20:27:25 INFO mapred.FileInputFormat: Total input paths to process : 1
17/12/10 20:27:25 INFO spark.SparkContext: Starting job: take at <console>:36
17/12/10 20:27:25 INFO scheduler.DAGScheduler: Got job 4 (take at <console>:36) with 1 output partitions (allowLocal=true)
17/12/10 20:27:25 INFO scheduler.DAGScheduler: Final stage: Stage 4(take at <console>:36)
17/12/10 20:27:25 INFO scheduler.DAGScheduler: Parents of final stage: List()
17/12/10 20:27:25 INFO scheduler.DAGScheduler: Missing parents: List()
17/12/10 20:27:25 INFO scheduler.DAGScheduler: Submitting Stage 4 (/user/training/rawdata/breweries.csv/part-m-00000 MapPartitionsRDD[7] at textFile at <console>:33), which has no missing parents
17/12/10 20:27:25 INFO storage.MemoryStore: ensureFreeSpace(2696) called with curMem=909112, maxMem=280248975
17/12/10 20:27:25 INFO storage.MemoryStore: Block broadcast_7 stored as values in memory (estimated size 2.6 KB, free 266.4 MB)
17/12/10 20:27:25 INFO storage.MemoryStore: ensureFreeSpace(1676) called with curMem=911808, maxMem=280248975
17/12/10 20:27:25 INFO storage.MemoryStore: Block broadcast_7_piece0 stored as bytes in memory (estimated size 1676.0 B, free 266.4 MB)
17/12/10 20:27:25 INFO storage.BlockManagerInfo: Added broadcast_7_piece0 in memory on localhost:60461 (size: 1676.0 B, free: 267.2 MB)
17/12/10 20:27:25 INFO storage.BlockManagerMaster: Updated info of block broadcast_7_piece0
17/12/10 20:27:25 INFO spark.SparkContext: Created broadcast 7 from broadcast at DAGScheduler.scala:839
17/12/10 20:27:25 INFO scheduler.DAGScheduler: Submitting 1 missing tasks from Stage 4 (/user/training/rawdata/breweries.csv/part-m-00000 MapPartitionsRDD[7] at textFile at <console>:33)
17/12/10 20:27:25 INFO scheduler.TaskSchedulerImpl: Adding task set 4.0 with 1 tasks
17/12/10 20:27:25 INFO scheduler.TaskSetManager: Starting task 0.0 in stage 4.0 (TID 4, localhost, PROCESS_LOCAL, 1334 bytes)
17/12/10 20:27:25 INFO executor.Executor: Running task 0.0 in stage 4.0 (TID 4)
17/12/10 20:27:25 INFO rdd.HadoopRDD: Input split: hdfs://localhost:8020/user/training/rawdata/breweries.csv/part-m-00000:0+23601
17/12/10 20:27:25 INFO executor.Executor: Finished task 0.0 in stage 4.0 (TID 4). 2029 bytes result sent to driver
17/12/10 20:27:25 INFO scheduler.DAGScheduler: Stage 4 (take at <console>:36) finished in 0.051 s
17/12/10 20:27:25 INFO scheduler.DAGScheduler: Job 4 finished: take at <console>:36, took 0.103234 s
0,NorthGate Brewing ,Minneapolis, MN
1,Against the Grain Brewery,Louisville, KY
2,Jack's Abby Craft Lagers,Framingham, MA
3,Mike Hess Brewing Company,San Diego, CA
4,Fort Point Beer Company,San Francisco, CA
```

Creating data frame for sql computations

Step 1: Load data from MySQL to spark- Create a new DataFrame based on the breweries_export table from the database and then RDD

val breweries=sqlCtx.load("jdbc",Map("url"->"jdbc:mysql://localhost/loudacre?user=training&password=training","dbtable" -> "breweries_export"))

breweries.registerTempTable("breweriesA")

Step 2:

breweries.take(10).foreach(println)

```
scala> breweries.take(10).foreach(println)
17/12/10 19:33:40 INFO spark.SparkContext: Starting job: runJob at SparkPlan.scala:121
17/12/10 19:33:40 INFO scheduler.DAGScheduler: Got job 36 (runJob at SparkPlan.scala:121) with 1 output partitions (allowLocal=false)
17/12/10 19:33:40 INFO scheduler.DAGScheduler: Final stage: Stage 77(runJob at SparkPlan.scala:121)
17/12/10 19:33:40 INFO scheduler.DAGScheduler: Parents of final stage: List()
17/12/10 19:33:40 INFO scheduler.DAGScheduler: Missing parents: List()
17/12/10 19:33:40 INFO scheduler.DAGScheduler: Submitting Stage 77 (MapPartitionsRDD[146] at map at SparkPlan.scala:96), which has no missing parents
17/12/10 19:33:40 INFO storage.MemoryStore: ensureFreeSpace(4072) called with curMem=1211869, maxMem=280248975
17/12/10 19:33:40 INFO storage.MemoryStore: Block broadcast_62 stored as values in memory (estimated size 4.0 KB, free 266.1 MB)
17/12/10 19:33:40 INFO storage.MemoryStore: ensureFreeSpace(2081) called with curMem=1215941, maxMem=280248975
17/12/10 19:33:40 INFO storage.MemoryStore: Block broadcast_62_piece0 stored as bytes in memory (estimated size 2.0 KB, free 266.1 MB)
17/12/10 19:33:40 INFO storage.BlockManagerInfo: Added broadcast_62_piece0 in memory on localhost:54626 (size: 2.0 KB, free: 267.2 MB)
17/12/10 19:33:40 INFO storage.BlockManagerMaster: Updated info of block broadcast_62_piece0
17/12/10 19:33:40 INFO spark.SparkContext: Created broadcast 62 from broadcast at DAGScheduler.scala:839
17/12/10 19:33:40 INFO scheduler.DAGScheduler: Submitting 1 missing tasks from Stage 77 (MapPartitionsRDD[146] at map at SparkPlan.scala:96)
17/12/10 19:33:40 INFO scheduler.TaskSchedulerImpl: Adding task set 77.0 with 1 tasks
17/12/10 19:33:40 INFO scheduler.TaskSetManager: Starting task 0.0 in stage 77.0 (TID 2835, localhost, PROCESS_LOCAL, 1062 bytes)
17/12/10 19:33:40 INFO executor.Executor: Running task 0.0 in stage 77.0 (TID 2835)
17/12/10 19:33:40 INFO jdbc.JDBCRDD: closed connection
17/12/10 19:33:40 INFO executor.Executor: Finished task 0.0 in stage 77.0 (TID 2835). 1189 bytes result sent to driver
17/12/10 19:33:40 INFO scheduler.DAGScheduler: Stage 77 (runJob at SparkPlan.scala:121) finished in 0.016 s
17/12/10 19:33:40 INFO scheduler.DAGScheduler: Job 36 finished: runJob at SparkPlan.scala:121, took 0.028520 s
17/12/10 19:33:40 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 77.0 (TID 2835) in 15 ms on localhost (1/1)
17/12/10 19:33:40 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 77.0, whose tasks have all completed, from pool
]0,NorthGate Brewing ,Minneapolis, MN
]1,Against the Grain Brewery,Louisville, KY
]2,Jack's Abby Craft Lagers,Framingham, MA
]3,Mike Hess Brewing Company,San Diego, CA
]4,Fort Point Beer Company,San Francisco, CA
]5,COAST Brewing Company,Charleston, SC
]6,Great Divide Brewing Company,Denver, CO
]7,Tapistry Brewing,Bridgman, MI
]8,Big Lake Brewing,Holland, MI
]9,The Mitten Brewing Company,Grand Rapids, MI
```

4. For the following questions use Spark SQL

a) Determine the number of breweries in each state

sqlCtx.sql("""Select STATE, count(ID) from breweriesA group by STATE""").count

```
scala> sqlCtx.sql("""Select STATE, count(ID) from breweriesA group by STATE""").count
17/12/10 12:14:00 INFO spark.SparkContext: Starting job: collect at SparkPlan.scala:83
17/12/10 12:14:00 INFO scheduler.DAGScheduler: Registering RDD 25 (mapPartitions at Exchange.scala:64)
17/12/10 12:14:00 INFO scheduler.DAGScheduler: Registering RDD 30 (mapPartitions at Exchange.scala:100)
17/12/10 12:14:00 INFO scheduler.DAGScheduler: Got job 5 (collect at SparkPlan.scala:83) with 1 output partitions (al
lowLocal=false)
```

```
17/12/10 12:14:03 INFO storage.ShuffleBlockFetcherIterator: Getting 200 non-empty blocks out of 200 blocks
17/12/10 12:14:03 INFO storage.ShuffleBlockFetcherIterator: Started 0 remote fetches in 1 ms
17/12/10 12:14:03 INFO executor.Executor: Finished task 0.0 in stage 11.0 (TID 805). 1223 bytes result sent to driver
17/12/10 12:14:03 INFO scheduler.DAGScheduler: Stage 11 (collect at SparkPlan.scala:83) finished in 0.205 s
17/12/10 12:14:03 INFO scheduler.DAGScheduler: Job 5 finished: collect at SparkPlan.scala:83, took 3.638559 s
res19: Long = 52
```

sqlCtx.sql("""Select STATE, count(ID) from breweriesA group by STATE""").show(52)

```
scala> sqlCtx.sql("""Select STATE, count(ID) from breweriesA group by STATE""").show(52)
17/12/10 12:17:11 INFO spark.SparkContext: Starting job: runJob at SparkPlan.scala:121
17/12/10 12:17:11 INFO scheduler.DAGScheduler: Registering RDD 44 (mapPartitions at Exchange.scala:64)
17/12/10 12:17:11 INFO scheduler.DAGScheduler: Got job 8 (runJob at SparkPlan.scala:121) with 1 output partitions (al
lowLocal=false)
17/12/10 12:17:11 INFO scheduler.DAGScheduler: Final stage: Stage 17(runJob at SparkPlan.scala:121)
17/12/10 12:17:11 INFO scheduler.DAGScheduler: Parents of final stage: List(Stage 16)
17/12/10 12:17:11 INFO scheduler.DAGScheduler: Missing parents: List(Stage 16)
17/12/10 12:17:11 INFO scheduler.DAGScheduler: Submitting Stage 16 (MapPartitionsRDD[44] at mapPartitions at Exchange
.scala:64), which has no missing parents
 FL              15
 MA              23
 MD              7
 ME              9
 MI              32
 MN              12
 GA              7
 MO              9
 MS              2
 MT              9

scala>
```

b) Determine the cities with most breweries

sqlCtx.sql("""Select CITY, count(ID) from breweriesA group by CITY order by c1 desc""").count

```
scala> sqlCtx.sql("""Select CITY, count(ID) from breweriesA group by CITY order by c1 desc""").count
17/12/10 12:20:30 INFO spark.SparkContext: Starting job: RangePartitioner at Exchange.scala:88
17/12/10 12:20:30 INFO scheduler.DAGScheduler: Registering RDD 51 (mapPartitions at Exchange.scala:64)
17/12/10 12:20:30 INFO scheduler.DAGScheduler: Got job 10 (RangePartitioner at Exchange.scala:88) with 200 output partitions (allowLocal=false)
17/12/10 12:20:30 INFO scheduler.DAGScheduler: Final stage: Stage 21(RangePartitioner at Exchange.scala:88)
17/12/10 12:20:30 INFO scheduler.DAGScheduler: Parents of final stage: List(Stage 20)
17/12/10 12:20:30 INFO scheduler.DAGScheduler: Missing parents: List(Stage 20)
...
17/12/10 12:20:44 INFO storage.ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
17/12/10 12:20:44 INFO executor.Executor: Finished task 0.0 in stage 25.0 (TID 1618). 1224 bytes result sent to driver
17/12/10 12:20:44 INFO scheduler.DAGScheduler: Stage 25 (collect at SparkPlan.scala:83) finished in 0.027 s
17/12/10 12:20:44 INFO scheduler.DAGScheduler: Job 11 finished: collect at SparkPlan.scala:83, took 10.306988 s
res22: Long = 384
```

sqlCtx.sql("""Select CITY, count(ID) from breweriesA group by CITY order by c1 desc""").show(50)

Note: Since the list is too long, only top 50 are shown

```
17/12/10 12:24:21 INFO scheduler.DAGScheduler: Stage 27 (takeOrdered at basicOperators.scala:137) finished in 6.907 s
17/12/10 12:24:21 INFO scheduler.DAGScheduler: Job 12 finished: takeOrdered at basicOperators.scala:137, took 7.495673 s
17/12/10 12:24:21 INFO scheduler.TaskSetManager: Finished task 199.0 in stage 27.0 (TID 1819) in 26 ms on localhost (200/200)
17/12/10 12:24:21 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 27.0, whose tasks have all completed, from pool
CITY             c1
Portland         17
Seattle          9
Boulder          9
Chicago          9
Denver           8
Austin           8
San Diego        8
Bend             6
San Francisco    5
Anchorage        4
Brooklyn         4
Indianapolis     4
Columbus         4
Cincinnati       4
Edwards          3
Albuquerque      3
Traverse City    3
Minneapolis      3
Saint Louis      3
Aurora           3
Jacksonville     3
Charlotte        3
Cleveland        3
Tampa            3
Santa Cruz       3
San Antonio      3
Lexington        3
Baltimore        3
Grand Rapids     3
Stevens Point    3
Athens           3
Lake Havasu City 2
Bloomfield       2
Dallas           2
Greenville       2
Richmond         2
Pottstown        2
Middlebury       2
Tucson           2
Louisville       2
Garden City      2
Charlottesville  2
Asheville        2
Bellingham       2
Springfield      2
Philadelphia     2
Fort Collins     2
Kalamazoo        2
Bloomington      2
Hilo             2
```

c) Calculate the average alcohol % by volume in each state

sqlCtx.sql("""select breweriesA.STATE, AVG(beersA.abv) from beersA,breweriesA where beersA.brewery_id=breweriesA.ID group by breweriesA.STATE""").count

```
scala> sqlCtx.sql("""select breweriesA.STATE, AVG(beersA.abv) from beersA,breweriesA where beersA.brewery_id=breweriesA.ID group by breweriesA.STATE""").count
17/12/10 12:33:30 INFO spark.SparkContext: Starting job: collect at SparkPlan.scala:83
17/12/10 12:33:30 INFO scheduler.DAGScheduler: Registering RDD 80 (mapPartitions at Exchange.scala:64)
17/12/10 12:33:30 INFO scheduler.DAGScheduler: Registering RDD 83 (mapPartitions at Exchange.scala:64)
17/12/10 12:33:30 INFO scheduler.DAGScheduler: Registering RDD 89 (mapPartitions at Exchange.scala:64)
17/12/10 12:33:30 INFO scheduler.DAGScheduler: Registering RDD 94 (mapPartitions at Exchange.scala:100)
17/12/10 12:33:30 INFO scheduler.DAGScheduler: Got job 14 (collect at SparkPlan.scala:83) with 1 output partitions (allowLocal=false)


17/12/10 12:34:11 INFO executor.Executor: Finished task 0.0 in stage 33.0 (TID 2223). 1223 bytes result sent to driver
17/12/10 12:34:11 INFO scheduler.DAGScheduler: Stage 33 (collect at SparkPlan.scala:83) finished in 0.059 s
17/12/10 12:34:11 INFO scheduler.DAGScheduler: Job 14 finished: collect at SparkPlan.scala:83, took 41.463948 s
res28: Long = 52
```

sqlCtx.sql("""select breweriesA.STATE, AVG(beersA.abv) from beersA,breweriesA where

beersA.brewery_id=breweriesA.ID group by breweriesA.STATE""").show(52)

```
scala> sqlCtx.sql("""select breweriesA.STATE, AVG(beersA.abv) from beersA,breweriesA where beersA.brewery_id=breweriesA.ID group by breweriesA.STATE""").show(52)
17/12/10 12:36:31 INFO spark.SparkContext: Starting job: runJob at SparkPlan.scala:121
17/12/10 12:36:31 INFO scheduler.DAGScheduler: Registering RDD 101 (mapPartitions at Exchange.scala:64)
17/12/10 12:36:31 INFO scheduler.DAGScheduler: Registering RDD 104 (mapPartitions at Exchange.scala:64)
17/12/10 12:36:31 INFO scheduler.DAGScheduler: Registering RDD 110 (mapPartitions at Exchange.scala:64)
17/12/10 12:36:31 INFO scheduler.DAGScheduler: Got job 15 (runJob at SparkPlan.scala:121) with 1 output partitions (allowLocal=false)
MD              0.059047619047619036
ME              0.05781481481481482
MI              0.06337748344370862
MN              0.059824561403508766
GA              0.056374999999999995
MO              0.054794871794871795
MS              0.058909090909090904
MT              0.0564871794871795

scala>
```

d) Determine the beer styles with highest average alcohol by volume

sqlCtx.sql("""select style, AVG(abv) as avg_alcohol from beersA group by style order by avg_alcohol desc""").count

```
scala> sqlCtx.sql("""select style, AVG(abv) as avg_alcohol from beersA group by style order by avg_alcohol desc""").count
17/12/10 12:41:27 INFO spark.SparkContext: Starting job: RangePartitioner at Exchange.scala:88
17/12/10 12:41:27 INFO scheduler.DAGScheduler: Registering RDD 117 (mapPartitions at Exchange.scala:64)
17/12/10 12:41:27 INFO scheduler.DAGScheduler: Got job 17 (RangePartitioner at Exchange.scala:88) with 200 output partitions (allowLocal=false)
17/12/10 12:41:27 INFO scheduler.DAGScheduler: Final stage: Stage 43(RangePartitioner at Exchange.scala:88)
17/12/10 12:41:27 INFO scheduler.DAGScheduler: Parents of final stage: List(Stage 42)
17/12/10 12:41:27 INFO scheduler.DAGScheduler: Missing parents: List(Stage 42)
17/12/10 12:41:27 INFO scheduler.DAGScheduler: Submitting Stage 42 (MapPartitionsRDD[117] at mapPartitions at Exchange.scala:64), which has no missing parents
17/12/10 12:41:27 INFO storage.MemoryStore: ensureFreeSpace(8944) called with curMem=33470, maxMem=280248975
17/12/10 12:41:27 INFO storage.MemoryStore: Block broadcast_35 stored as values in memory (estimated size 8.7 KB, free 267.2 MB)
17/12/10 12:41:27 INFO storage.MemoryStore: ensureFreeSpace(4251) called with curMem=42414, maxMem=280248975
17/12/10 12:41:27 INFO storage.MemoryStore: Block broadcast_35_piece0 stored as bytes in memory (estimated size 4.2 KB, free 267.2 MB)
17/12/10 12:41:27 INFO storage.BlockManagerInfo: Added broadcast_35_piece0 in memory on localhost:32854 (size: 4.2 KB, free: 267.3 MB)
17/12/10 12:41:41 INFO executor.Executor: Finished task 0.0 in stage 47.0 (TID 3124). 1224 bytes result sent to driver
17/12/10 12:41:41 INFO scheduler.DAGScheduler: Stage 47 (collect at SparkPlan.scala:83) finished in 0.032 s
17/12/10 12:41:41 INFO scheduler.DAGScheduler: Job 18 finished: collect at SparkPlan.scala:83, took 10.860373 s
res30: Long = 102
```

sqlCtx.sql("""select style, AVG(abv) as avg_alcohol from beersA group by style order by avg_alcohol desc""").show(50)

Note: Since the list is too long, only top 50 are shown

```
scala> sqlCtx.sql("""select style, AVG(abv) as avg_alcohol from beersA group by style order by avg_alcohol desc""").show(50)
17/12/10 12:43:48 INFO spark.SparkContext: Starting job: takeOrdered at basicOperators.scala:137
17/12/10 12:43:48 INFO scheduler.DAGScheduler: Registering RDD 144 (mapPartitions at Exchange.scala:64)
17/12/10 12:43:48 INFO scheduler.DAGScheduler: Got job 20 (takeOrdered at basicOperators.scala:137) with 200 output partitions (allowLocal=false)
17/12/10 12:43:48 INFO scheduler.DAGScheduler: Final stage: Stage 51(takeOrdered at basicOperators.scala:137)
17/12/10 12:43:48 INFO scheduler.DAGScheduler: Parents of final stage: List(Stage 50)
17/12/10 12:43:48 INFO scheduler.DAGScheduler: Missing parents: List(Stage 50)
Rauchbier             0.0605
Wheat Ale             0.06
Oatmeal Stout         0.05966666666666667
American White IPA    0.05963636363636365
American Porter       0.05959999999999998
Maibock / Helles ...  0.05925
Pumpkin Ale           0.058434782608695654
American Brown Ale    0.05792647058823529
MÃ¤rzen / Oktober... 0.05772413793103448
```

e) Determine the most brewed beer style

sqlCtx.sql("""select style,count(sno) from beersA group by style order by c1 desc""").count

```
scala> sqlCtx.sql("""select style,count(sno) from beersA group by style order by c1 desc""").count
17/12/10 12:48:37 INFO spark.SparkContext: Starting job: RangePartitioner at Exchange.scala:88
17/12/10 12:48:37 INFO scheduler.DAGScheduler: Registering RDD 152 (mapPartitions at Exchange.scala:64)
17/12/10 12:48:37 INFO scheduler.DAGScheduler: Got job 21 (RangePartitioner at Exchange.scala:88) with 200 output partitions (allowLocal=false)
17/12/10 12:48:37 INFO scheduler.DAGScheduler: Final stage: Stage 53(RangePartitioner at Exchange.scala:88)
17/12/10 12:48:37 INFO scheduler.DAGScheduler: Parents of final stage: List(Stage 52)
17/12/10 12:48:47 INFO storage.ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
17/12/10 12:48:47 INFO executor.Executor: Finished task 0.0 in stage 57.0 (TID 3968). 1224 bytes result sent to driver
17/12/10 12:48:47 INFO scheduler.DAGScheduler: Stage 57 (collect at SparkPlan.scala:83) finished in 0.041 s
17/12/10 12:48:47 INFO scheduler.DAGScheduler: Job 22 finished: collect at SparkPlan.scala:83, took 7.294213 s
res34: Long = 102
```

sqlCtx.sql("""select style,count(sno) from beersA group by style order by c1 desc""").show(50)

Note: Since the list is too long, only top 50 are shown

```
scala> sqlCtx.sql("""select style,count(sno) from beersA group by style order by c1 desc""").show(50)
17/12/10 12:50:55 INFO spark.SparkContext: Starting job: takeOrdered at basicOperators.scala:137
17/12/10 12:50:55 INFO scheduler.DAGScheduler: Registering RDD 171 (mapPartitions at Exchange.scala:64)
17/12/10 12:50:55 INFO scheduler.DAGScheduler: Got job 23 (takeOrdered at basicOperators.scala:137) with 200 output partitions (allowLocal=false)
17/12/10 12:50:55 INFO scheduler.DAGScheduler: Final stage: Stage 59(takeOrdered at basicOperators.scala:137)
17/12/10 12:50:55 INFO scheduler.DAGScheduler: Parents of final stage: List(Stage 58)
Tripel                11
Russian Imperial ...  11
American White IPA    11
Belgian Dark Ale      11
Milk / Sweet Stout    10
Schwarzbier           9
American Double /...  9
```

The most brewed beer style is American IPA.