Problem Statement

Go through below blog and reiterate the same at your end.
https://docs.google.com/document/d/1csLBlMiEXs_hXWV2Z8VpBlrj_R6RoDQLlZUnA0uBTCk/

(Census data analysis.pdf)

DataSet

You can download the dataset from the below link

https://drive.google.com/open?id=0ByJLBTmJojjzWllGZFJFaXFVbU0

(census.csv)

Due to the limitation of 22 elements for a map function, we are taking only 22 columns from the data set.
Here is the total dataset description

State String,District String,Persons String,Males int,Females int,Growth_1991_2001 int,Rural int,Urban int,Scheduled_Caste_population int,Percentage_SC_to_total int,Number_of_households int,Household_size_per_household int,Sex_ratio_females_per_1000_males int ,Sex_ratio_0_6_years int,Scheduled_Tribe_population int,Percentage_to_total_population_ST int,Persons_literate int,Males_Literate int,Females_Literate int,Persons_literacy_rate int,Males_Literatacy_Rate int,Females_Literacy_Rate int,Total_Educated int,Data_without_level int,Below_Primary int,Primary int,Middle int,Matric_Higher_Secondary_Diploma int,Graduate_and_Above int,X0_4_years int,X5_14_years int,X15_59_years int,X60_years_and_above_Incl_ANS int,Total_workers int,Main_workers int,Marginal_workers int,Non_workers int,SC_1_Name String,SC_1_Population int,SC_2_Name String,SC_2_Population int,SC_3_Name String,SC_3_Population int,Religeon_1_Name String,Religeon_1_Population int,Religeon_2_Name String,Religeon_2_Population int,Religeon_3_Name String,Religeon_3_Population int,ST_1_Name String,ST_1_Population int,ST_2_Name String,ST_2_Population int,ST_3_Name String,ST_3_Population int,Imp_Town_1_Name String,Imp_Town_1_Population int,Imp_Town_2_Name String,Imp_Town_2_Population int,Imp_Town_3_Name String,Imp_Town_3_Population int,Total_Inhabited_Villages int,Drinking_water_facilities int,Safe_Drinking_water int,Electricity_Power_Supply int,Electricity_domestic int,Electricity_Agriculture int,Primary_school int,Middle_schools int,Secondary_Sr_Secondary_schools int,College int,Medical_facility int,Primary_Health_Centre int,Primary_Health_Sub_Centre int,Post_telegraph_and_telephone_facility int,Bus_services int,Paved_approach_road int,Mud_approach_road int,Permanent_House int,Semi_permanent_House int,Temporary_House int

Here is what we are taking

"State" ,"Persons","Males" ,"Females" ,"Growth_1991_2001" ,"Rural" ,"Urban" ,"Scheduled_Caste_population" ,"Percentage_SC_to_total" ,"Number_of_households" ,"Household_size_per_household" ,"Sex_ratio_females_per_1000_males " ,"Sex_ratio_0_6_years" ,"Scheduled_Tribe_population" ,"Percentage_to_total_population_ST" ,"Persons_literate" ,"Males_Literate" ,"Females_Literate" ,"Persons_literacy_rate" ,"Males_Literatacy_Rate" ,"Females_Literacy_Rate" ,"Total_Educated"

Solution:

Importing spark and sql context packages to be use for dataframes

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

scala>

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

scala>
```

Reading the CSV text into RDD and mapping the required columns to RDD

```
scala> val census_data = sc.textFile("file:///home/acadgild/Downloads/census.csv").map(x => x.split(",")).map(x => (x(0),x(2)
,x(3),x(4),x(5),x(6),x(7),x(8),x(9),x(10),x(11),x(12),x(13),x(14),x(15),x(16),x(17),x(18),x(19),x(20),x(21),x(22)))
census_data: org.apache.spark.rdd.RDD[(String, String, String, String, String, String, String, String, String, String, String
, String, String, String, String, String, String, String, String, String, String)] = MapPartitionsRDD[13] at map at <
console>:33

scala>
```

Converting Data RDD into Dataframes with named columns

```
scala> val censusDf = census_data.toDF("State" ,"Persons","Males" ,"Females" ,"Growth_1991_2001" ,"Rural" ,"Urban" ,"Schedule
d_Caste_population" ,"Percentage_SC_to_total" ,"Number_of_households" ,"Household_size_per_household" ,"Sex_ratio_females_per
_1000_males " ,"Sex_ratio_0_6_years" ,"Scheduled_Tribe_population" ,"Percentage_to_total_population_ST" ,"Persons_literate" ,
"Males_Literate" ,"Females_Literate" ,"Persons_literacy_rate" ,"Males_Literatacy_Rate" ,"Females_Literacy_Rate" ,"Total_Educa
ted")
censusDf: org.apache.spark.sql.DataFrame = [State: string, Persons: string, Males: string, Females: string, Growth_1991_2001:
 string, Rural: string, Urban: string, Scheduled_Caste_population: string, Percentage_SC_to_total: string, Number_of_household
ds: string, Household_size_per_household: string, Sex_ratio_females_per_1000_males : string, Sex_ratio_0_6_years: string, Sch
eduled_Tribe_population: string, Percentage_to_total_population_ST: string, Persons_literate: string, Males_Literate: string,
 Females_Literate: string, Persons_literacy_rate: string, Males_Literatacy_Rate: string, Females_Literacy_Rate: string, Total
_Educated: string]

scala>
```

Creating of Dataframes into Temporary Table

```
scala> val censustf = censusDf.registerTempTable("census")
censustf: Unit = ()

scala>
```

1. **Find out the state wise population and order by state**

Code:

```
scala> val population = sqlContext.sql("select state,sum(persons) as total_population from census group by state order by tot
al_population desc")
population: org.apache.spark.sql.DataFrame = [state: string, total_population: double]

scala>
```

Output:

```
scala> population.show()
+-----------+----------------+
|      state|total_population|
+-----------+----------------+
|         UP|     1.66197921E8|
|Maharashtra|      9.6878627E7|
|      Bihar|      8.2998509E7|
|         WB|      8.0176197E7|
|     Andhra|      7.1308587E7|
|         TN|      6.2405679E7|
|         MP|      6.0348023E7|
|  Rajasthan|      5.6507188E7|
|  Karnataka|      5.2850562E7|
|    Gujarat|      5.0671017E7|
|     Orrisa|      3.5664657E7|
|     Kerala|      3.1841374E7|
|  Jharkhand|      2.6945829E7|
|      Assam|      2.6655528E7|
|     Punjab|      2.4358999E7|
|    Haryana|      2.1144564E7|
|         CG|      2.0833803E7|
|      Delhi|      1.3850507E7|
|         JK|         1.01437E7|
| Uttranchal|         8489349.0|
+-----------+----------------+
only showing top 20 rows

scala>
```

2. Find out the Growth Rate of Each State Between 1991-2001

Code:

```
scala> val growth_rate = sqlContext.sql("select state,avg(Growth_1991_2001) as total_growth from census group by state")
growth_rate: org.apache.spark.sql.DataFrame = [state: string, total_growth: double]

scala>
```

Output:

```
scala> growth_rate.show()
+----------------+------------------+
|           state|       total_growth|
+----------------+------------------+
|     Maharashtra|19.607142857142865|
|              TN|10.127666666666668|
|         Gujarat|           20.8248|
|          Orrisa|15.551379310344826|
|          Sikkim|31.834999999999997|
|              AN|            18.665|
|       Chandigarh|             40.33|
|           Bihar|28.605945945945955|
|              HP| 17.53083333333333|
|              UP| 25.70228571428572|
|ArunachalPradesh| 25.46999999999999|
|          Tripura|15.405000000000001|
|            D_N_H|              59.2|
|       Uttranchal|17.092307692307692|
|          Haryana|27.816842105263152|
|              CG|17.506249999999998|
|              WB|18.424999999999997|
|          Manipur|29.240000000000002|
|              JK|28.785714285714285|
|       Lakshdweep|             17.19|
+----------------+------------------+
only showing top 20 rows

scala>
```

3. Find the literacy rate of each state

Code:

```
scala> val literacy = sqlContext.sql("select state,avg(Persons_literacy_rate) as Avg_Persons_Literacy from census group by st
ate")
literacy: org.apache.spark.sql.DataFrame = [state: string, Avg_Persons_Literacy: double]

scala>
```

Output:

```
scala> literacy.show()
+----------------+--------------------+
|           state|Avg_Persons_Literacy|
+----------------+--------------------+
|     Maharashtra|    74.55342857142857|
|              TN|    72.94266666666665|
|         Gujarat|    67.07480000000001|
|          Orrisa|    59.97965517241381|
|          Sikkim|             66.9975|
|              AN|    77.41999999999999|
|       Chandigarh|               81.94|
|           Bihar|    46.42135135135135|
|              HP|    75.50833333333333|
|              UP|    56.01057142857144|
|ArunachalPradesh|   53.166923076923084|
|          Tripura|    70.27000000000001|
|            D_N_H|               57.63|
|       Uttranchal|    72.01769230769231|
|          Haryana|    68.24473684210527|
|              CG|    63.02312499999999|
|              WB|               66.07|
|          Manipur|              68.6125|
|              JK|   54.867142857142845|
|       Lakshdweep|               86.66|
+----------------+--------------------+
only showing top 20 rows

scala>
```

4. Find out the States with More Female Population

Code:

```
scala> val female_pop = sqlContext.sql("select state, sum(Males)-sum(Females) from census group by state")
female_pop: org.apache.spark.sql.DataFrame = [state: string, _c1: double]

scala>
```

Output:

```
scala> female_pop.show()
+----------------+---------+
|           state|      _c1|
+----------------+---------+
|     Maharashtra|3922565.0|
|              TN| 396139.0|
|         Gujarat|2100137.0|
|          Orrisa| 482015.0|
|          Sikkim|  36117.0|
|              AN|  29792.0|
|       Chandigarh| 113241.0|
|           Bihar|3489081.0|
|              HP|  97980.0|
|              UP|8932817.0|
|ArunachalPradesh|  61914.0|
|          Tripura| 185247.0|
|            D_N_H|  22842.0|
|       Uttranchal| 162499.0|
|          Haryana|1583342.0|
|              CG| 114633.0|
|              WB|2755773.0|
|          Manipur|  20533.0|
|              JK| 578152.0|
|       Lakshdweep|   1612.0|
+----------------+---------+
only showing top 20 rows

scala>
```

5. Find out the Percentage of Population in Every State

Code:

```
scala> val percenet_pop = sqlContext.sql("select state, (sum(persons) * 100.0) / SUM(sum(persons)) over() as percent_pop_by_s
tate from census group by state")
percenet_pop: org.apache.spark.sql.DataFrame = [state: string, percent_pop_by_state: double]

scala>
```

Output:

```
scala> percenet_pop.show()
17/12/19 15:32:01 WARN Window: No Partition Defined for Window operation! Moving all data to a single partition, this can cau
se serious performance degradation.
+----------------+--------------------+
|           state|percent_pop_by_state|
+----------------+--------------------+
|     Maharashtra|   9.475494209385522|
|              TN|   6.103767861999858|
|         Gujarat|   4.956025317815201|
|          Orrisa|   3.488284891601744|
|          Sikkim| 0.05289949576432755|
|              AN| 0.03483447606726582|
|       Chandigarh| 0.08808921009243792|
|           Bihar|   8.117909138174843|
|              HP|  0.5944665819347776|
|              UP|   16.25546817511578|
|ArunachalPradesh| 0.10738993468694186|
|          Tripura| 0.31290729895613395|
|            D_N_H| 0.02156566193106157|
|       Uttranchal|  0.8303253233652121|
|          Haryana|  2.0681052152192616|
|              CG|  2.0377103371415317|
|              WB|   7.841864753141607|
|          Manipur| 0.19662075848548596|
|              JK|  0.9921339059826262|
|       Lakshdweep|0.005932048601382...|
+----------------+--------------------+
only showing top 20 rows

scala>
```