ASSIGNMENT 22.1

Problem Statement:

Go through below blog and reiterate the same at your end.

https://docs.google.com/document/d/1csLBlMiEXs_hXWV2Z8VpBlrj_R6RoDQLlZUnA0uBTCk

Solution:

I have downloaded the dataset from the link below:

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

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"

Let’s load this dataset into Spark memory:

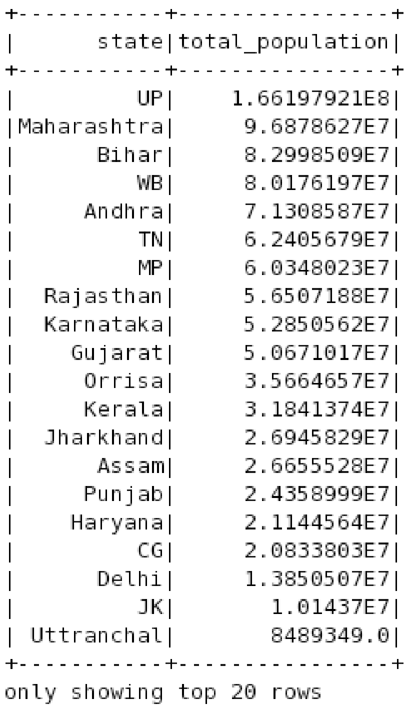
```
val census_data = sc.textFile("file:///home/kiran/Documents/datasets/census.csv")

          .map (x => x.split(","))            // split each line by comma to access each
                                              // field by its index position

          .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)))

          .toDF("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")      //define schema for the dataset

          .registerTempTable("census")                  // register a temporary table
                                                        // with name ‘census’
```

Problem 1: Find out the state wise population and order by state

Here is an SQL query embedded into Spark’s SQL context to find this out:

```
val population = spark.sql(“select state,sum(persons) as total_population ” +
                            “from census ” +
                            “group by state ” +
                            “order by total_population desc ”
                           ).show
```

Output:



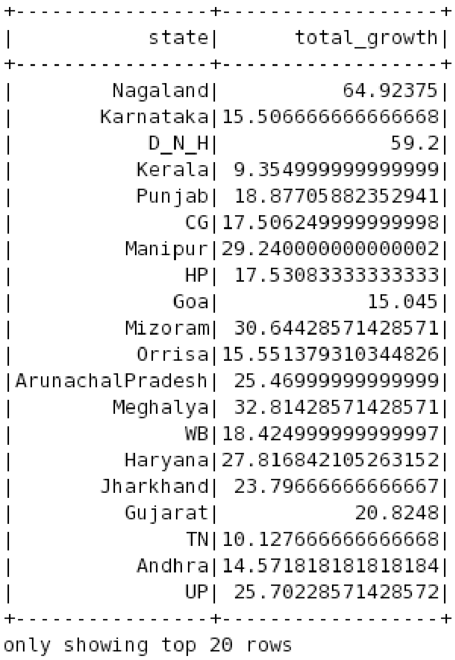
```
+-----------+---------------+
|      state|total_population|
+-----------+---------------+
|         UP|    1.66197921E8|
|Maharashtra|     9.6878627E7|
|      Bihar|     8.2998509E7|
|         WB|     8.0176197E7|
|     Andhra|     7.1308587E7|
|         TN|     6.2405679E7|
|         MP|     6.0348023E7|
|  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
```

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

Here is an SQL query embedded into Spark’s SQL context to find this out:

```
val growth_rate = spark.sql(“select state, avg(Growth_1991_2001) as total_growth ” +

                             “from census group by state”

                            ).show
```

Output:



```
+----------------+------------------+
|           state|      total_growth|
+----------------+------------------+
|        Nagaland|          64.92375|
|       Karnataka|15.506666666666668|
|           D_N_H|              59.2|
|          Kerala| 9.354999999999999|
|          Punjab| 18.87705882352941|
|              CG|17.506249999999998|
|         Manipur|29.240000000000002|
|              HP| 17.53083333333333|
|             Goa|            15.045|
|         Mizoram| 30.64428571428571|
|          Orrisa|15.551379310344826|
|ArunachalPradesh| 25.46999999999999|
|        Meghalya| 32.81428571428571|
|              WB|18.424999999999997|
|         Haryana|27.816842105263152|
|       Jharkhand| 23.79666666666667|
|         Gujarat|           20.8248|
|              TN|10.127666666666668|
|          Andhra|14.571818181818184|
|              UP| 25.70228571428572|
+----------------+------------------+
only showing top 20 rows
```

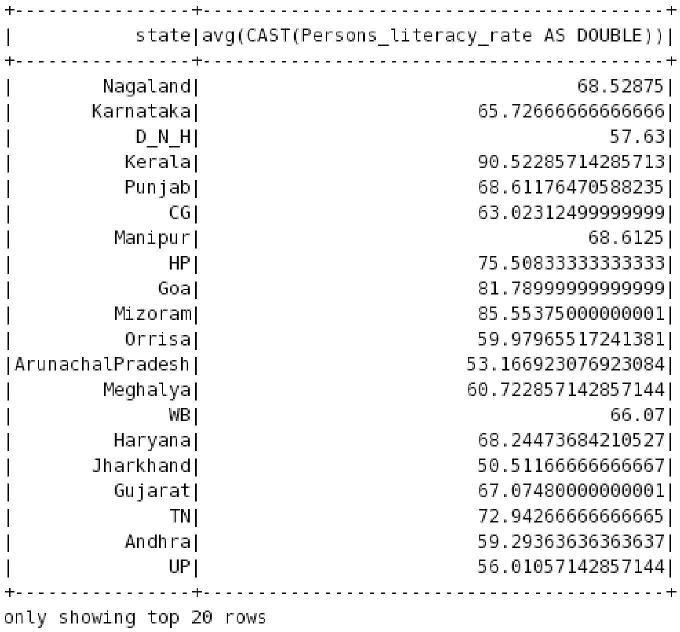
3. Find the literacy rate of each state.

Here is an SQL query embedded into Spark’s SQL context to find this out:

val literacy = spark.sql(“select state,avg(Persons_literacy_rate) ” +

“from census group by state”).show

Output:



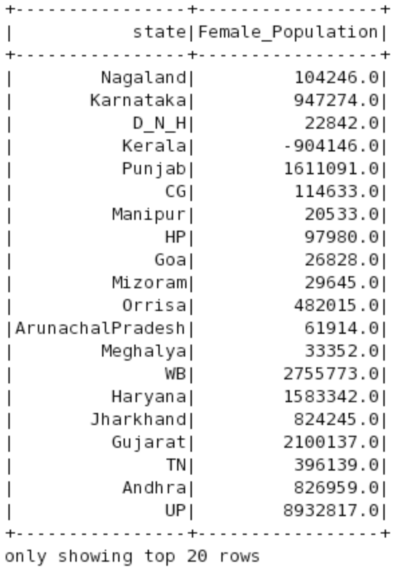
```
+----------------+------------------------------------------+
|           state|avg(CAST(Persons_literacy_rate AS DOUBLE))|
+----------------+------------------------------------------+
|        Nagaland|                                  68.52875|
|       Karnataka|                         65.72666666666666|
|           D_N_H|                                     57.63|
|          Kerala|                         90.52285714285713|
|          Punjab|                         68.61176470588235|
|              CG|                         63.02312499999999|
|         Manipur|                                   68.6125|
|              HP|                         75.50833333333333|
|             Goa|                         81.78999999999999|
|         Mizoram|                         85.55375000000001|
|          Orrisa|                         59.97965517241381|
|ArunachalPradesh|                        53.166923076923084|
|        Meghalya|                        60.722857142857144|
|              WB|                                     66.07|
|         Haryana|                         68.24473684210527|
|       Jharkhand|                         50.51166666666667|
|         Gujarat|                         67.07480000000001|
|              TN|                         72.94266666666665|
|          Andhra|                         59.29363636363637|
|              UP|                         56.01057142857144|
+----------------+------------------------------------------+
only showing top 20 rows
```

4. Find out the States with More Female Population.

Here is an SQL query embedded into Spark’s SQL context to find this out:

```
val female_pop = spark.sql(“select state, sum(Males)-sum(Females) as Female_Population ” +
                           “from census” +
                           “group by state”).show
```

Output:

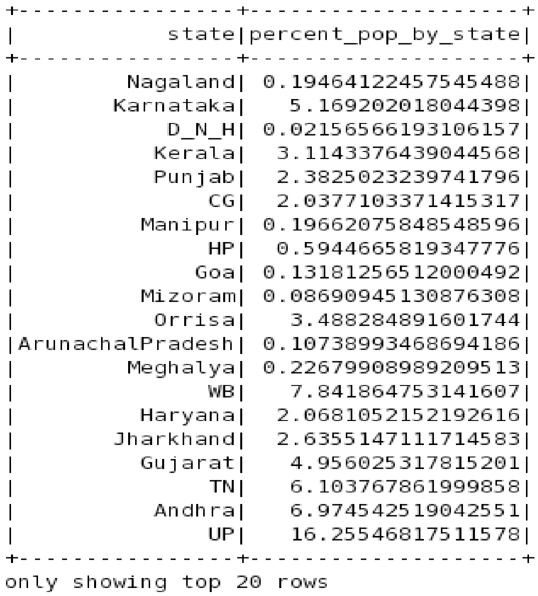


```
+----------------+-----------------+
|           state|Female_Population|
+----------------+-----------------+
|        Nagaland|         104246.0|
|       Karnataka|         947274.0|
|           D_N_H|          22842.0|
|          Kerala|        -904146.0|
|          Punjab|        1611091.0|
|              CG|         114633.0|
|         Manipur|          20533.0|
|              HP|          97980.0|
|             Goa|          26828.0|
|         Mizoram|          29645.0|
|          Orrisa|         482015.0|
|ArunachalPradesh|          61914.0|
|        Meghalya|          33352.0|
|              WB|        2755773.0|
|         Haryana|        1583342.0|
|       Jharkhand|         824245.0|
|         Gujarat|        2100137.0|
|              TN|         396139.0|
|          Andhra|         826959.0|
|              UP|        8932817.0|
+----------------+-----------------+
only showing top 20 rows
```

5. Find out the Percentage of Population in Every State

```
val percenet_pop = spark.sql(“select state, (sum(persons) * 100.0) / SUM(sum(persons)) over() ” +
                             “as percent_pop_by_state from census group by state”).show
```

Output:



```
+----------------+--------------------+
|           state|percent_pop_by_state|
+----------------+--------------------+
|        Nagaland| 0.19464122457545488|
|       Karnataka|   5.169202018044398|
|           D_N_H| 0.02156566193106157|
|          Kerala|  3.1143376439044568|
|          Punjab|  2.3825023239741796|
|              CG|  2.0377103371415317|
|         Manipur| 0.19662075848548596|
|              HP|  0.5944665819347776|
|             Goa| 0.13181256512000492|
|         Mizoram| 0.08690945130876308|
|          Orrisa|   3.488284891601744|
|ArunachalPradesh| 0.10738993468694186|
|        Meghalya| 0.22679908989209513|
|              WB|   7.841864753141607|
|         Haryana|  2.0681052152192616|
|       Jharkhand|  2.6355147111714583|
|         Gujarat|   4.956025317815201|
|              TN|   6.103767861999858|
|          Andhra|   6.974542519042551|
|              UP|   16.25546817511578|
+----------------+--------------------+
only showing top 20 rows
```