Data Ingestion Using Scoop And Data Analysis Using Hive

DATA INGESTION FROM RDS TO HDFS USING SQOOP

1. Sqoop import command (importFromRDSToHDFS.sqoop)

   Note
   The table **Key_indicator_districtwise** available in RDS has some **NULL** values in some of the columns.

   Using the scoop command, the **NULL** values are replaced with **NA** for all String based columns and **\N** for all non-string based columns while importing data into HDFS.

   This is to make sure the **NULL** value is not written to the HDFS data.

2. Command to see the list of imported data (viewData.hdfs)

HIVE EXTERNAL TABLE CREATION AND LOADING INGESTED DATA

1. Command to create the external table (createHiveTable.hql)

   Note
   A database named **India_Annual_Health_Survey_2012_13_DB** is created. All the tables pertaining to this project will be created in this database.

   An external table named **IAHS_2012_13** is created with **645** columns. This table will be used as a master repository of data.

2. Command to load the ingested data into the external table (loadDataInHiveTable.hql)

3. Queries to verify that the ingestion is correctly accomplished
   1. Query to count the total number of rows of data fetched from RDS using MySQL Workbench and from Hive using Hue

      MySQL Workbench (verificationQuery1.sql)
      Hue (verificationQuery1.hql)

   2. Query to select the top 10 rows and first 8 columns of the data fetched from RDS using MySQL Workbench and from Hive using Hue

      MySQL Workbench (verificationQuery2.sql)
      Hue (verificationQuery2.hql)

Note

The above listed **02** queries and their results across the RDBMS table **Key_indicator_districtwise** and the HIVE table **IAHS_2012_13** should show that the data is correctly imported from RDS to HDFS using sqoop.

Later, the same imported data is correctly ingested into the HIVE table **IAHS_2012_13.**

SUBSET SCHEMA CREATION IN HIVE TO SUPPORT ANALYSIS

1. Columns used in the subset schema
   ID
   State_Name
   State_District_Name
   AA_Households_Total
   AA_Population_Total
   CC_Sex_Ratio_All_Ages_Total
   LL_Total_Fertility_Rate_Total
   YY_Under_Five_Mortality_Rate_U5MR_Total_Person

2. Storage format used [Benchmark the performance before finalizing the storage format to be used. Create one schema using default format and one in any other format such as ORC for the columns to be used. Insert data into both the tables created. Compare the runtimes of the following queries and decide which format to be used.
   1. select count(*) from <Table Name>;
   2. select State_Name, count(*) from <Table Name> group by State_Name;
   *3. select * from <Table Name> where State_Name = 'Uttar Pradesh';]*

   Note
   In point **03** below,
   A subset table named **IAHS_2012_13_TEXT** is created with default **TEXT** format.

   The subset table contains selected **08** columns.

   The data is ingested into this table from the master table **IAHS_2012_13.**

In point **04** below,

A subset table named **IAHS_2012_13_ORC** is created with **ORC** format.

The subset table contains selected **08** columns.

The data is ingested into this table from the master table **IAHS_2012_13.**

In point **05** below,

03 sets of queries are executed against both the tables (reference, point 3 and 4) and their execution time is noted.

On examining the execution time for all the **03** set of queries, it is observed that the queries executed on the table with **ORC** format has lower execution time in comparision to the execution time of queries executed on the table with default **TEXT** format.

The difference in execution time of queries is marginal as the data set is small in size.

The difference in execution time will increase for a voluminous production size data set.

Based on the benchmarking performed for all the **03** queries, I have choosen the ORC format to be used for this project.

Additionally, I have also used the compression algorithm **SNAPPY** with the **ORC** format as opposed to the non-compressed way of storing data with the default **TEXT** format.

The data stored in compressed format saves on disk space which is again helpful when the size of the data set is voluminous.

3. Create and insert command for the default format (createAndInsertDefaultFormat.hql)

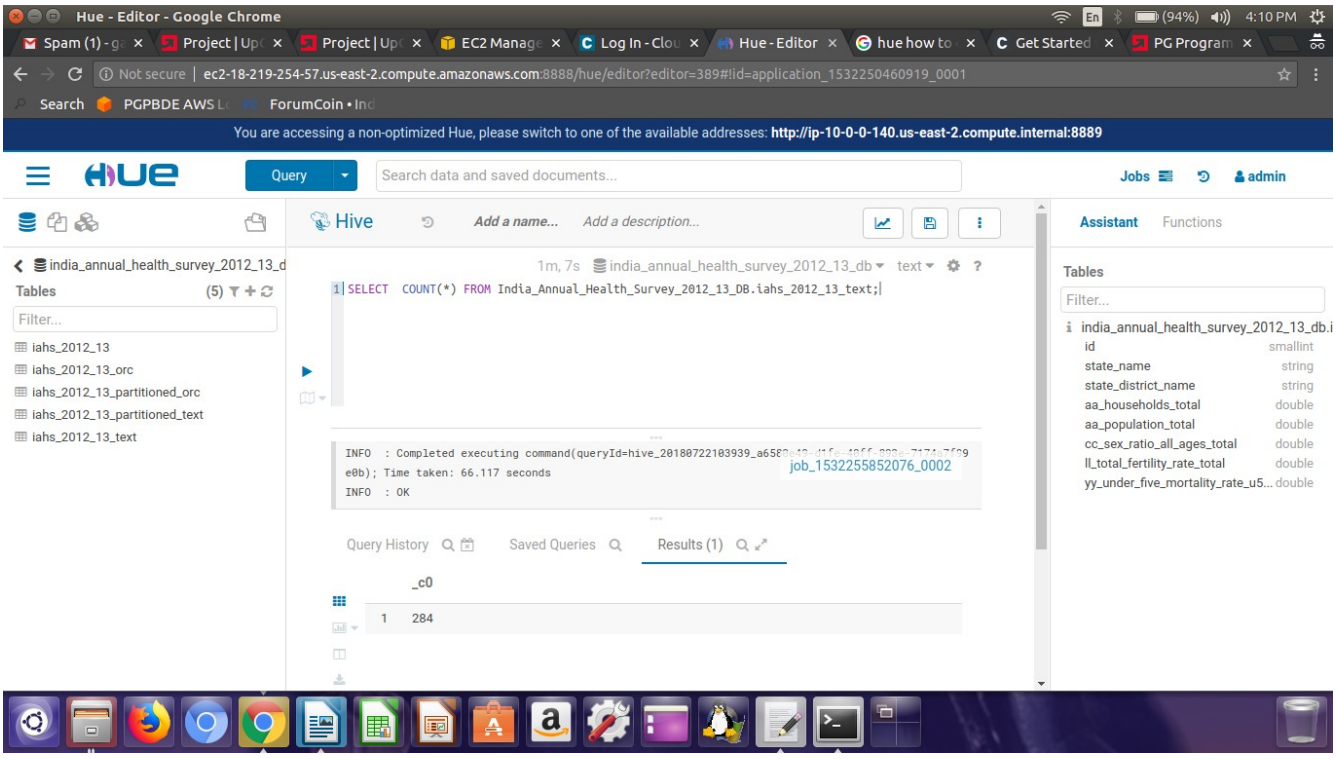
4. Create and insert command for the formats such as ORC (createAndInsertORCFormat.hql)

5. Screenshot of runtimes against each query given above for the default format as well as for the formats such as ORC

TEXT FORMAT

SELECT COUNT(*) FROM India_Annual_Health_Survey_2012_13_DB.iahs_2012_13_text;

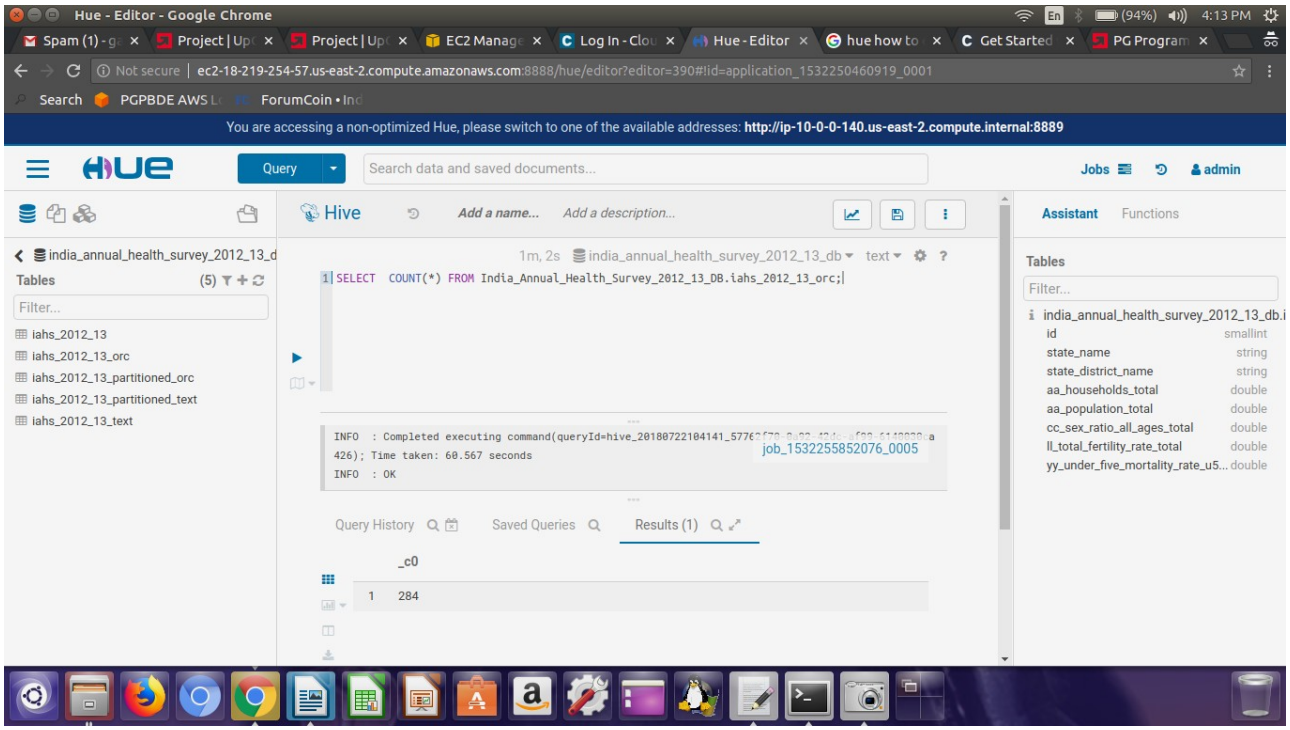
Time Taken: **66.117 seconds**



ORC FORMAT

SELECT COUNT(*) FROM India_Annual_Health_Survey_2012_13_DB.iahs_2012_13_orc;

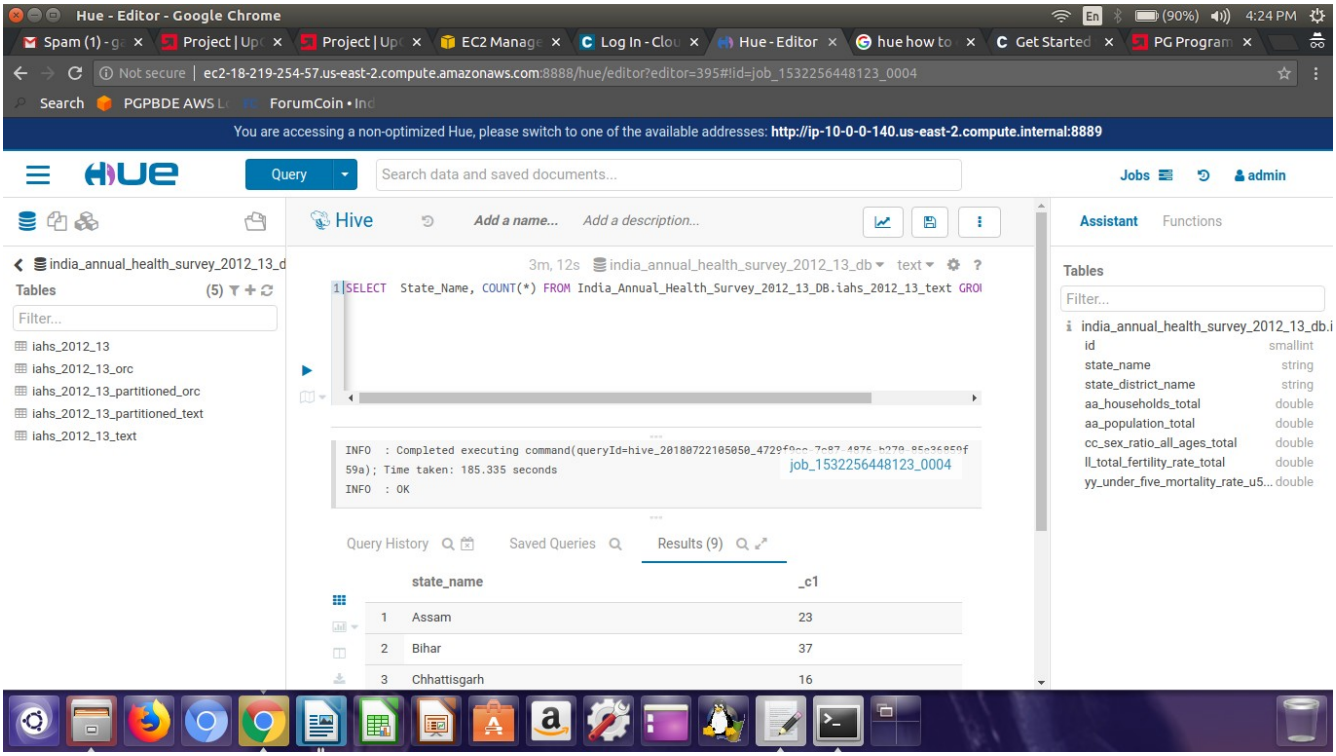
Time Taken: **60.657 seconds**



TEXT FORMAT

SELECT State_Name, COUNT(*) FROM India_Annual_Health_Survey_2012_13_DB.iahs_2012_13_text GROUP BY State_Name;

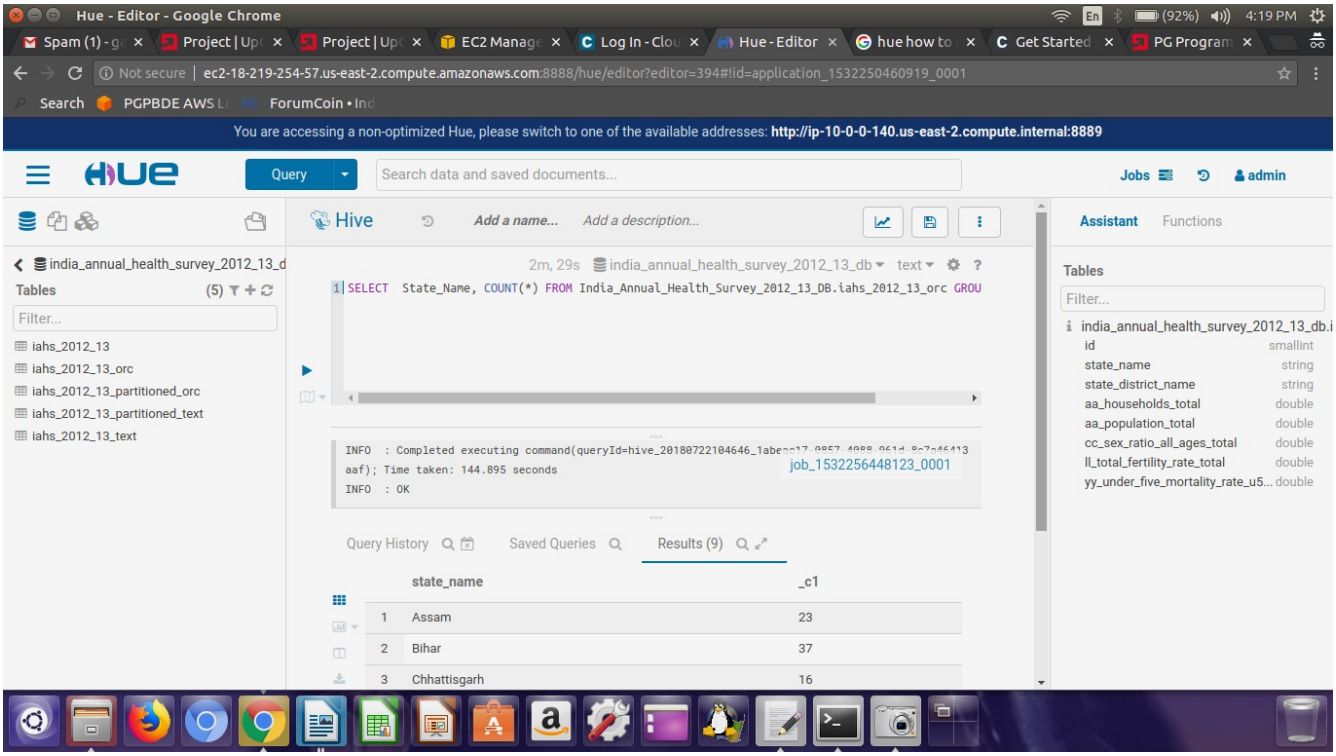
Time Taken: **185.335 seconds**



ORC FORMAT

SELECT State_Name, COUNT(*) FROM India_Annual_Health_Survey_2012_13_DB.iahs_2012_13_orc GROUP BY State_Name;

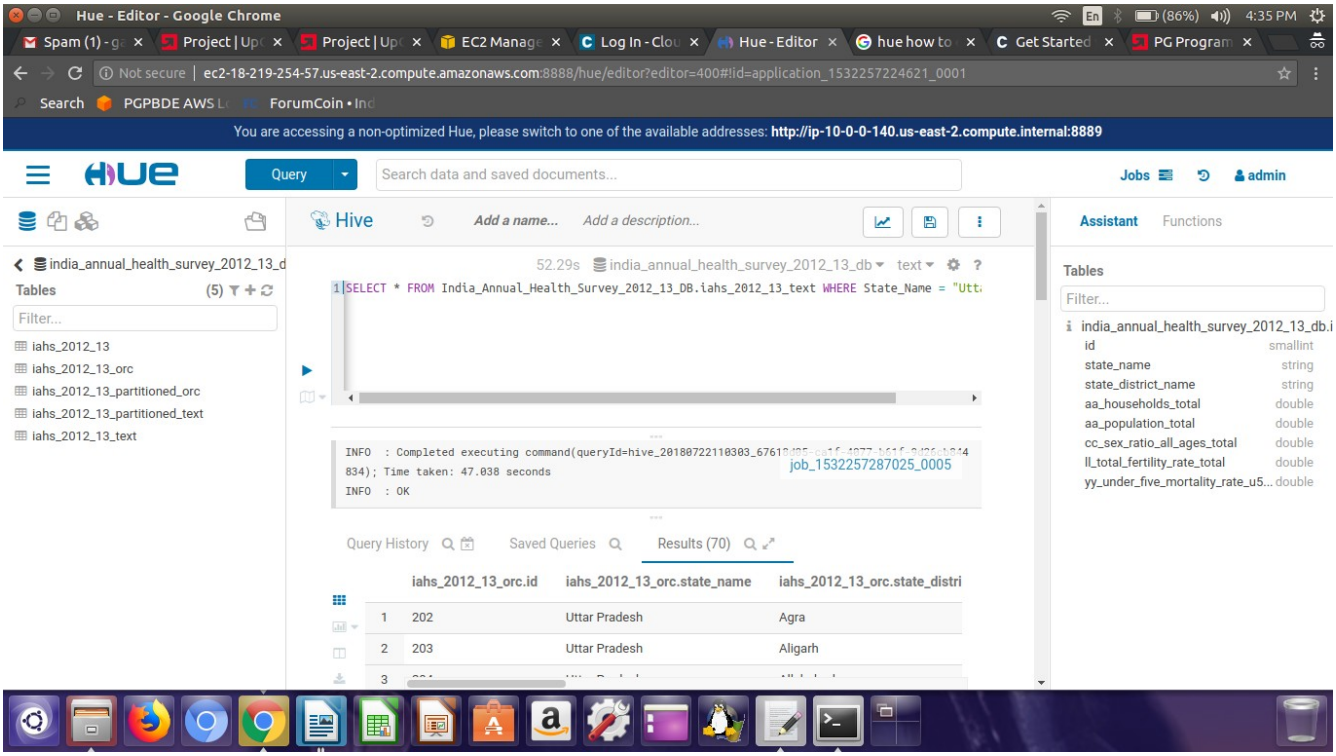
Time Taken: **144.896 seconds**



TEXT FORMAT

*SELECT * FROM India_Annual_Health_Survey_2012_13_DB.iahs_2012_13_text WHERE State_Name = "Uttar Pradesh";*

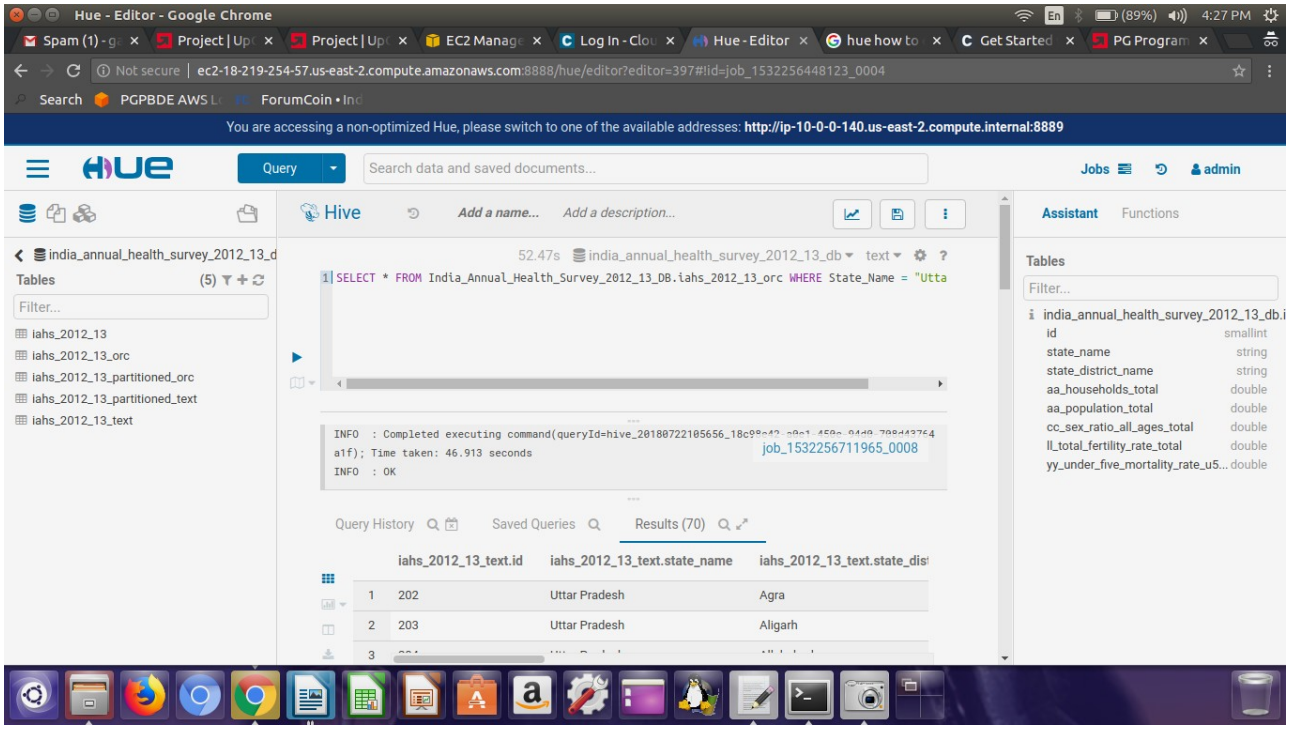
Time Taken: **47.038 seconds**



ORC FORMAT

*SELECT * FROM India_Annual_Health_Survey_2012_13_DB.iahs_2012_13_orc WHERE State_Name = "Uttar Pradesh";*

Time Taken: **46.913 seconds**



6. Create and insert command for the partition table for analyses 1 & 2. The partition table should be created using the table created above.
(createAndInsertORCFormatPartitioned.hql)

Note
For analyses 1 and 2,
A partitioned table named **IAHS_2012_13_PARTITIONED_ORC_FORMAT** is created with **ORC** format.

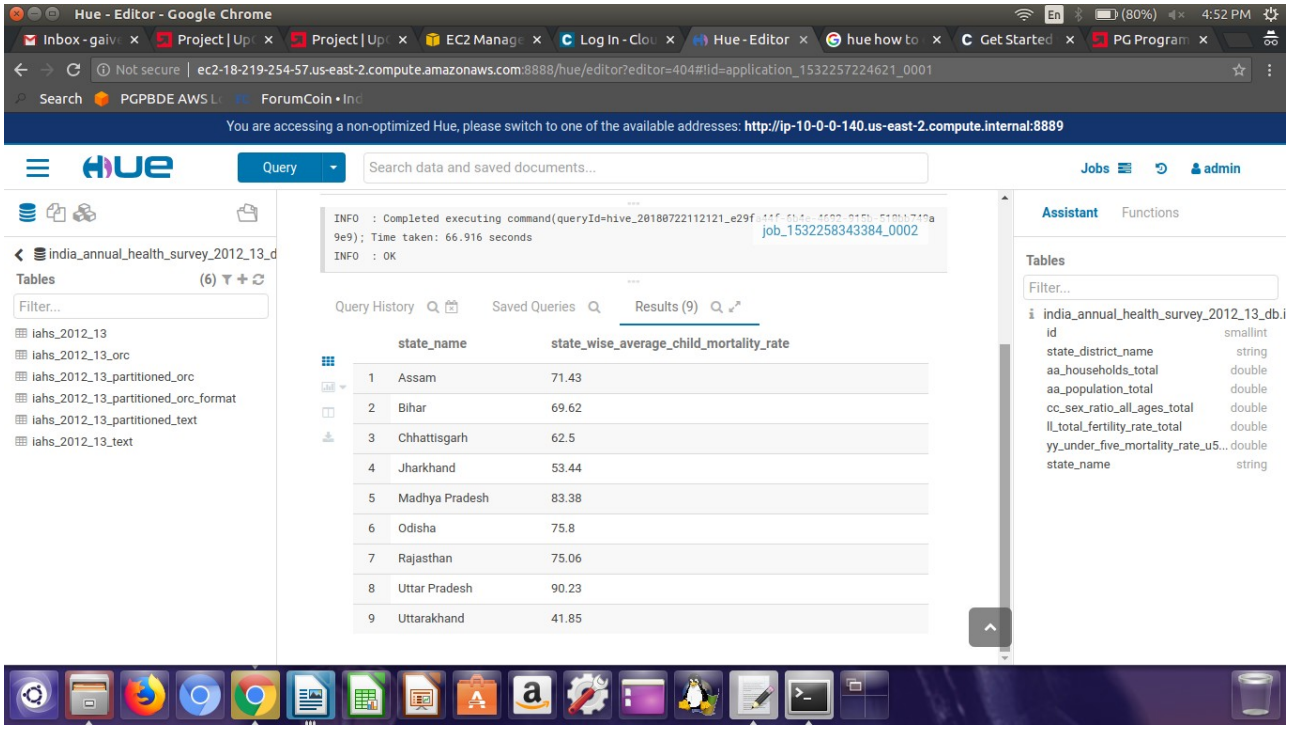
The data into this table is ingested from the master table **IAHS_2012_13.**

This table will be used only for writing queries for analyses 1 and 2.
For analyses 3, 4 and 5, the non-partitioned **ORC** format table **IAHS_2012_13_ORC** will be used.

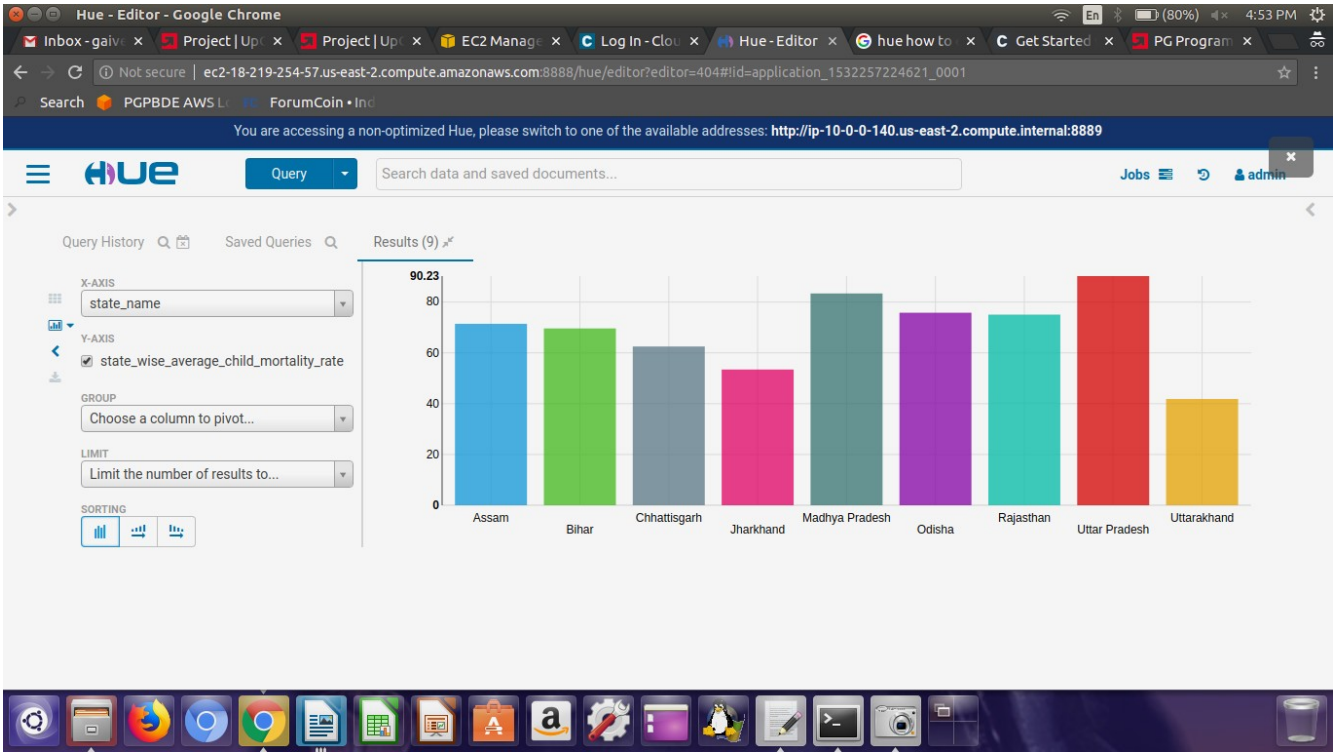
QUERY ANALYSIS, RESULT AND CHART

1. State wise child mortality rate (queryForAnalysis1.hql)

Screenshot of the result

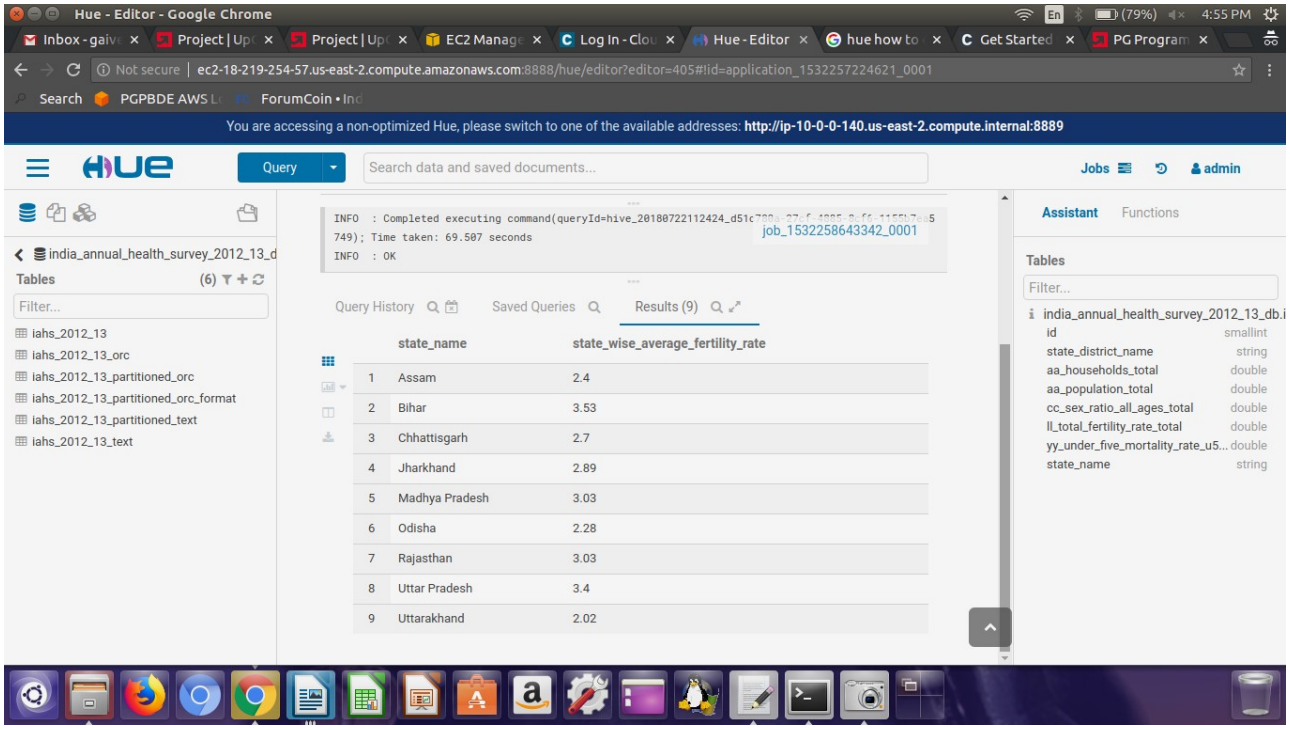


Chart

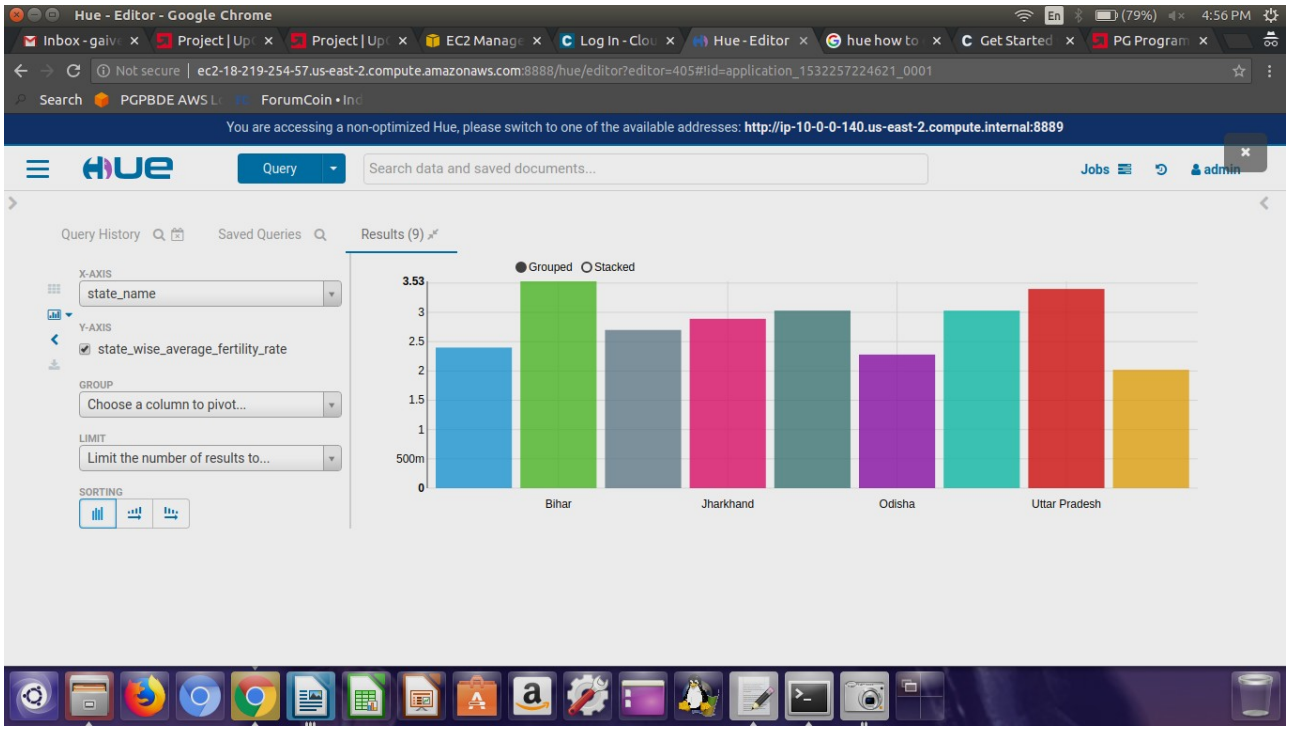

Hue - Editor - Google Chrome
Inbox - gaiv
Project | Up
Project | Up
EC2 Manage
Log In - Clou
Hue - Editor
hue how to
Get Started
PG Program
Not secure | ec2-18-219-254-57.us-east-2.compute.amazonaws.com:8888/hue/editor?editor=404#!id=application_1532257224621_0001
Search
PGPBDE AWS
ForumCoin • Ind
You are accessing a non-optimized Hue, please switch to one of the available addresses: http://ip-10-0-0-140.us-east-2.compute.internal:8889
HUE
Query
Search data and saved documents...
Jobs
admin
Query History
Saved Queries
Results (9)
X-AXIS
state_name
Y-AXIS
state_wise_average_child_mortality_rate
GROUP
Choose a column to pivot...
LIMIT
Limit the number of results to...
SORTING
90.23
80
60
40
20
0
Assam
Bihar
Chhattisgarh
Jharkhand
Madhya Pradesh
Odisha
Rajasthan
Uttar Pradesh
Uttarakhand

2. State wise fertility rate (queryForAnalysis2.hql)

Screenshot of the result

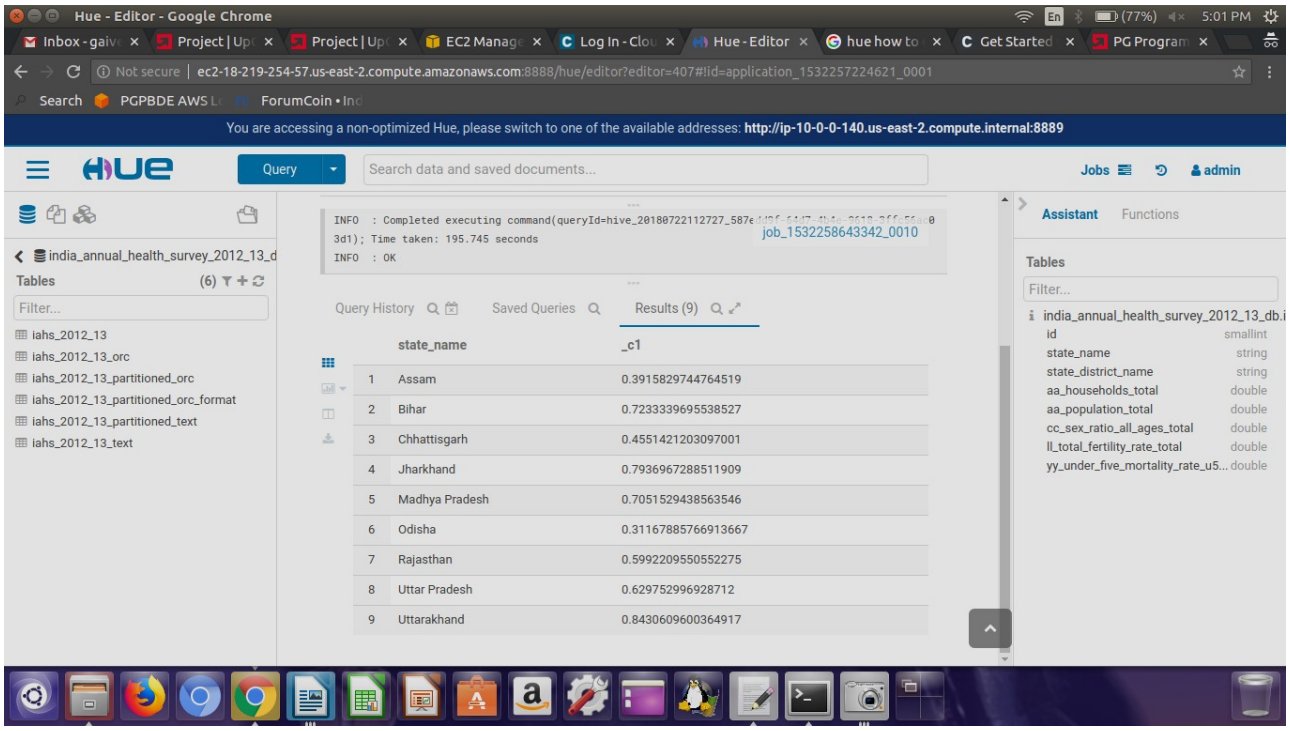


Chart

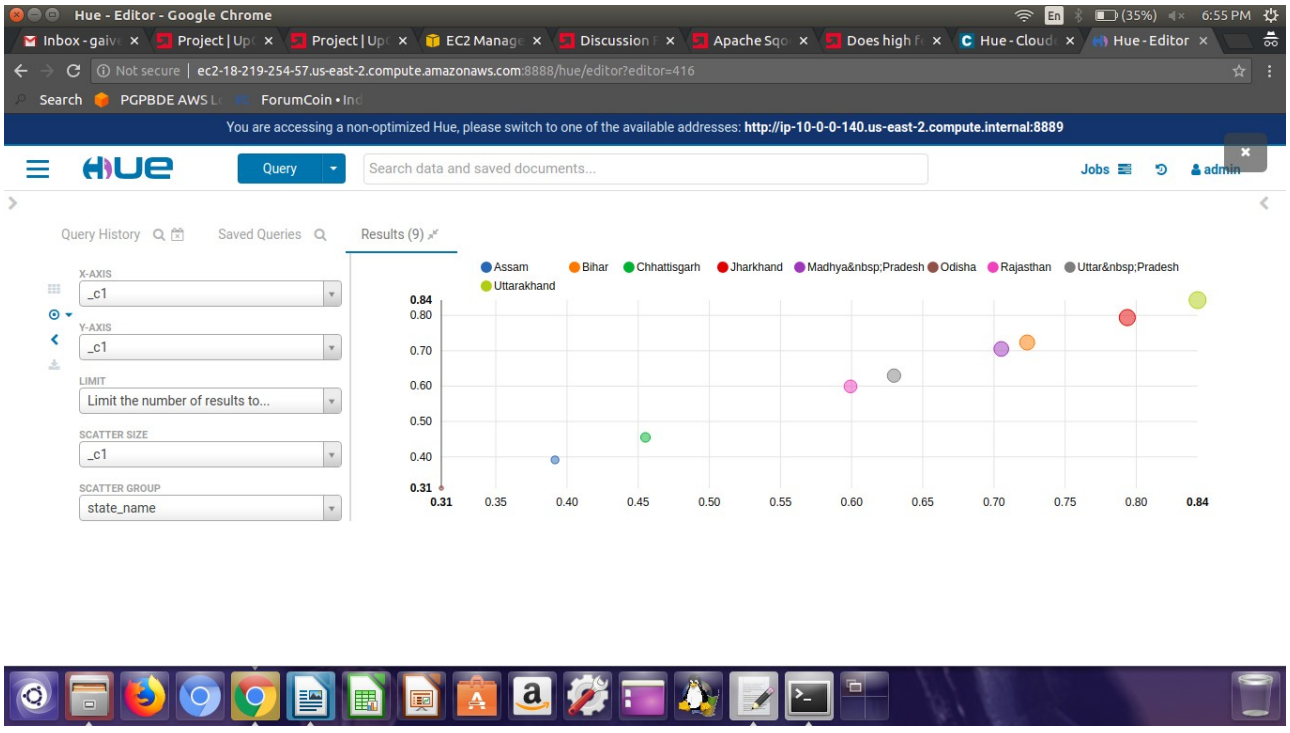


3. Does high fertility correlate with high child mortality? (queryForAnalysis3.hql)

Screenshot of the result



Chart

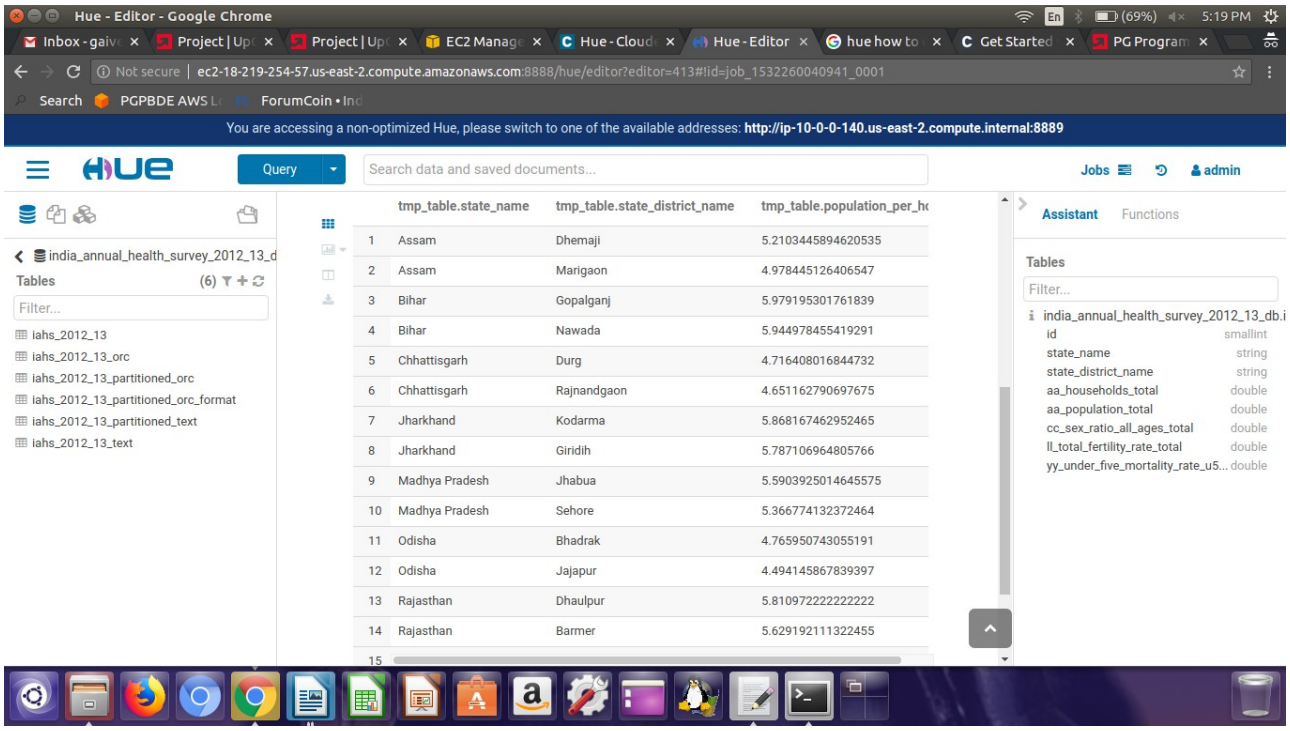


Note

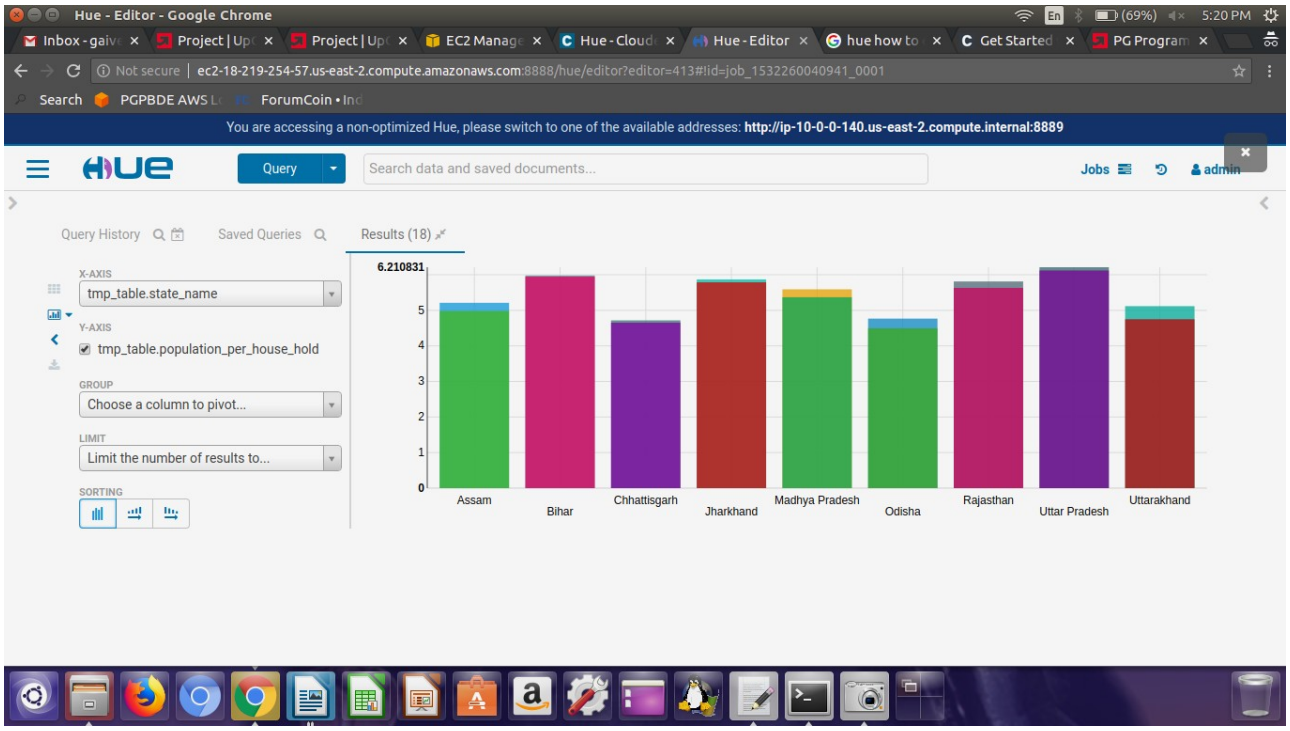
Based on the analysis of the output, we see a positive slope in the scatter plot above as all the correlation co-efficient lie in the range of 0.3 to 0.8

4. Find top 2 districts per state with the highest population per household (queryForAnalysis4.hql)

Screenshot of the result

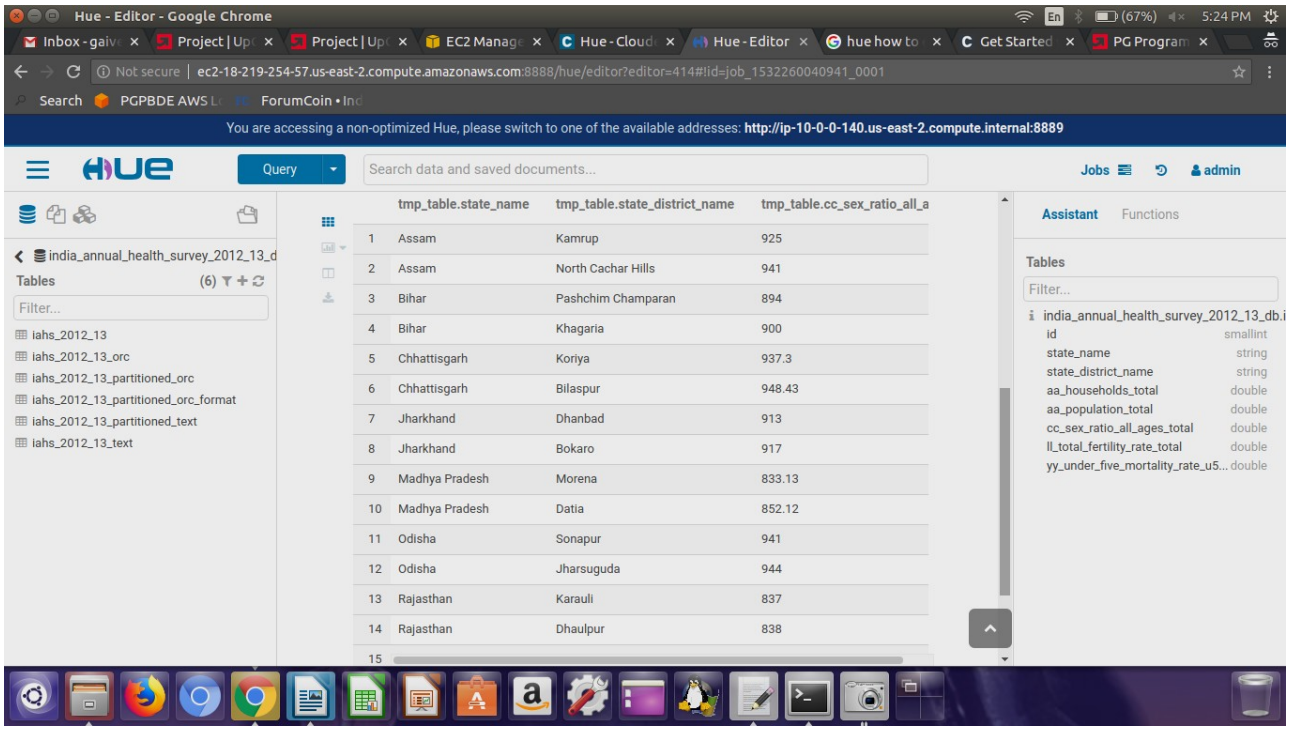


Chart



5. Find top 2 districts per state with the lowest sex ratios (queryForAnalysis5.hql)

Screenshot of the result



Chart