**Overview**

The H-1B is an employment-based, non-immigrant visa category for temporary foreign workers in the United States. For a foreign national to apply for H1-B visa, an US employer must offer a job and petition for H-1B visa with the US immigration department. This is the most common visa status applied for and held by international students once they complete college/ higher education (Masters, PhD) and work in a full-time position.

* **Approach**

To perform this analysis, we will be using two major tools Spark and Hive from the Hadoop ecosystem. In this use case Spark is used as an ETL to clean and prepare the H1b dataset to be analyzed using Hive Query Language.

**Analysis Description**

Once the dataset is loaded, cleaned and ready to use we will be performing the following exploratory data analysis to find:

1. Number of applications for each year
2. Top 10 Employers to file H1B
3. Top job titles with highest application over the year 2008-2017
4. Average Salary for an analysts position from year 2011 to 2017
5. Highest Average Salaries Based on Job Titles
6. Highest Average Salaries Based on Worksite State
7. Top most hiring states
8. Full positions count
9. The Case\_status count
10. Is the number of petitions with Data Engineer job title increasing over time?
11. Which industry (SOC\_NAME) has the most number of Data Scientist positions?
12. The case\_status count per case\_status
13. The case\_status count per year
14. The most popular top 10 job positions for H1B visa applications for each year for only certified applications.
15. The average Prevailing Wage for each Job for each Year

To perform the Analysis we will be using Zeppelin note with many interpreters such as **spark, hive and shell** interpreter (**%spark, %hive, %sh**).

%spark

sc.version

**H1B DataSet Description**

The H1B16 dataset is provided in CSV format and . It contains five year’s worth of H-1B petition data and it has nearly 3 million records.

The dataset description is as follows:

* **CASE\_STATUS**: Status associated with the last significant event or decision. Valid values include “Certified”, “Certified-Withdrawn”, ” Denied”, and “Withdrawn”.
  + **Certified**: Employer filed the LCA, which was approved by DOL
  + **Certified Withdrawn**: LCA was approved but later withdrawn by employer
  + **Withdrawn**: LCA was withdrawn by employer before approval
  + **Denied:** LCA was denied by DOL
* **EMPLOYER\_NAME**: Name of employer submitting labour condition application.
* **SOC\_NAME**: the Occupational name associated with the SOC\_CODE. SOC\_CODE is the occupational code associated with the job being requested for temporary labour condition, as classified by the Standard Occupational Classification (SOC) System.
* **JOB\_TITLE**: Title of the job
* **FULL\_TIME\_POSITION**: Y = Full Time Position; N = Part Time Position
* **PREVAILING\_WAGE**: Prevailing Wage for the job being requested for temporary labour condition. The wage is listed at annual scale in USD. The prevailing wage for a job position is defined as the average wage paid to similarly employed workers in the requested occupation in the area of intended employment. The prevailing wage is based on the employer’s minimum requirements for the position.
* **YEAR**: Year in which the H1B visa petition was filed
* **WORKSITE**: City and State information of the foreign worker’s intended area of employment
* **lon**: longitude of the Worksite
* **lat**: latitude of the Worksite

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Column 1 | Column 2 | Column 3 | Column 4 | Column 5 | Column 6 | Column 7 | Column 8 | Column 9 | Column 10 |
| CASE\_STATUS | EMPLOYER\_NAME | SOC\_NAME | JOB\_TITLE | FULL\_TIME\_POSITION | PREVAILING\_WAGE | YEAR | WORKSITE | lon | lat |

**H1B DataSet Preparation**

In the data, few columns are enclosed by double quotes and also we have comma’s in a single column and the column is enclosed by double quotes. So we have to clean the dataset and remove all the extra quotes.

To perform this task we will be using Spark SQL DataFrame API. We first load the dataset, then perform some cleansing and write back the dataset on HDFS partitioned by year. Note that the default Spark output format is Parquet compressed using Snappy.

* **Placing H1B Dataset on HDFS**

First, extract the**h1b16.csv** dataset from the zip archive and upload it to HDFS.

%sh

# unzip the input archive and place the file on HDFS

cd /home/training/Data/

unzip h1b16.zip

hdfs dfs -mkdir -p /tutorials/uses\_cases/h1b/csv

hdfs dfs -put h1b16.csv /tutorials/uses\_cases/h1b/csv

* **Loading H1B Dataset With Spark**

Next, we load the h1b16.csv dataset from HDFS usinh Spark SQL API. We set the **read** function to:

* + Remove the CSV header.
  + Use the **‘,’** (coma) as column separator.
  + Infer the schema.
  + Remove the double quotes
  + Persist the dataframe in cache so input/output operations will perform faster.

%spark

// load the file using the SparkSQL context

val h1bRawDF = spark.read

.format("csv")

.option("header","true")

.option("inferSchema","true")

.option("delimiter", ",")

.option("quote", "\"")

.option("escape", "\"")

.load("/tutorials/uses\_cases/h1b/csv")

.cache

Now the input dataset is loaded in the h1bRawDF dataframe. You can print out the dataframe schema and explore the dataframe’s columns.

%spark

// print the dataframe schema

​

h1bRawDF.printSchema

* **H1B Dataset Cleansing**

Next we need to perform some cleansing and setting the data type for some columns. we will preform the following:

* + Remove the first column as it is an incremental ID not used in our analysis.
  + Remove all the rows with “NA”.
  + Convert **Year**, **PREVAILING\_WAGE**  columns data type from String to Integer.
  + Convert **lon**, **lat**columns data type from String to Double.

%spark

// H1B visa dataset cleansing

import org.apache.spark.sql.types.\_

val h1b = h1bRawDF

.drop("\_c0")

.filter(h1bRawDF("year") =!= "NA")

.withColumn("YEAR", col("YEAR").cast(IntegerType))

.withColumn("PREVAILING\_WAGE", col("PREVAILING\_WAGE").cast(IntegerType))

.withColumn("lon", col("lon").cast(DoubleType))

.withColumn("lat", col("lat").cast(DoubleType))

You can print out the new dataframe schema h1b and explore the columns.

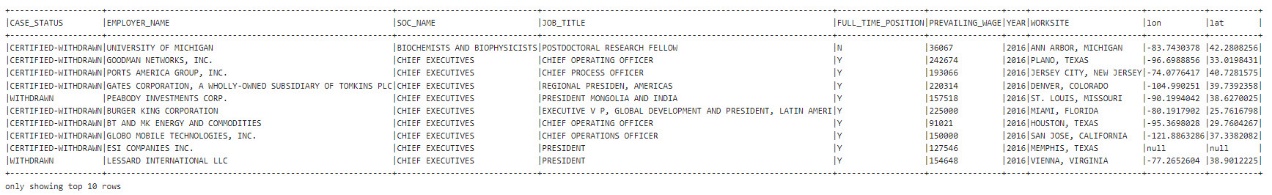
%spark

h1b.printSchema

And show a few rows to ensure that everything is fine.

%spark

​h1b.show(10, false)

[](http://localhost/wp-content/uploads/2023/07/h1b-dataframe-show-10.jpg)

* **Write Cleaned H1B Dataset To HDFS**

Next, after cleaning H1B dataset it is time to write it on HDFS.

We will be using the **write** function to write to the parquet directory on HDFS and set the PartitionBy column as ‘**year**‘ .

Note that the default writing format is Parquet with Snappy compression.

%spark

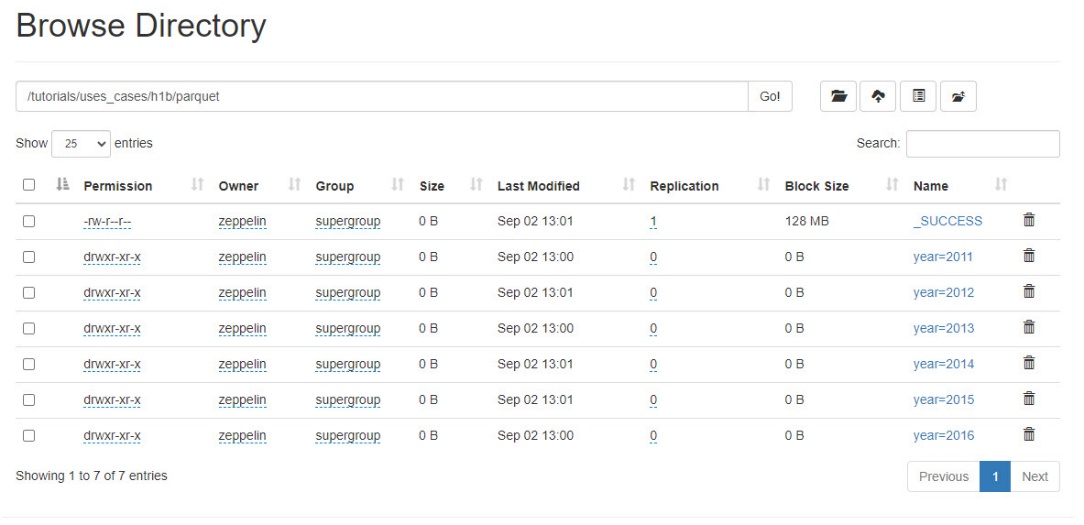
// write the dataframe in Parquet / Snappy and partition by 'year'

h1b

.write.mode("overwrite")

.partitionBy("year")

.save("/tutorials/uses\_cases/h1b/parquet")

[](http://localhost/wp-content/uploads/2023/07/h1b-dataframe-output-hdfs.jpg)

**Loading Cleaned H1B DataSet With Hive**

The H1B dataset has been cleaned, partitioned and outputted by Spark in Parquet format. We will be loading this cleaned and partitioned dataset using Hive and then perform the analysis using Hive Query Language (HiveQL).

* **Creating Hive Table**

First, we will create a User Managed table and read the data from HDFS.

%hive

-- Cerate the database tutorials

Create Database If Not Exists tutorials;

%hive

-- Create a User Managed table to read H1b parquet files

CREATE External TABLE tutorials.h1b\_parquet (

CASE\_STATUS string,

employer\_name string,

soc\_name string,

job\_title string,

full\_time\_position string,

prevailing\_wage int,

worksite string,

lon double,

lat double)

Partitioned By (year int)

Stored as parquet

location '/tutorials/uses\_cases/h1b/parquet'

TBLPROPERTIES ("parquet.compression"="SNAPPY");

* **Adding Missing Partitions**

With Spark we already partitioned the H1B dataset by **year**. For instance, Hive doesn’t know anything about these partitions. Let’s add all these partitions and check if Hive loaded them correctly.

%hive

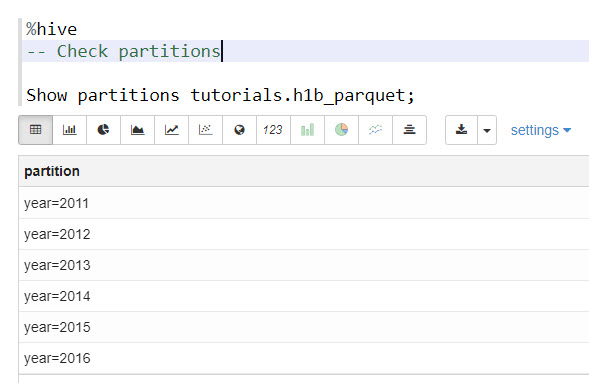
-- Add all missing partitions

msck repair table tutorials.h1b\_parquet;

%hive

-- Check partitions

Show partitions tutorials.h1b\_parquet;

[](http://localhost/wp-content/uploads/2023/07/h1b-hive-partitions.jpg)

Let’s show the top ten rows from h1b\_parquet table.

%hive

-- Show the first ten rows

select \* from tutorials.h1b\_parquet limit 10;

[](http://localhost/wp-content/uploads/2023/07/h1b-table-show-10.jpg)

* **Preparing H1B Table For Analysis**

To make HiveQL queries performing faster, we need to compute the h1b\_parquet table statistics. The following statistics currently supported for table and partitions:

* Number of rows
* Number of files
* Size in Bytes
* Number of partition if the table is partitioned.

Apache Hive uses ANALYZE TABLE command to collect statistics on a given table. This command collects statistics and stored in Hive metastore.

%hive

-- Prepare the table for the analysis

Analyze table tutorials.h1b\_parquet compute statistics;

**H1B Visa Analysis**

1. **Find the number of applications for each year**

%hive

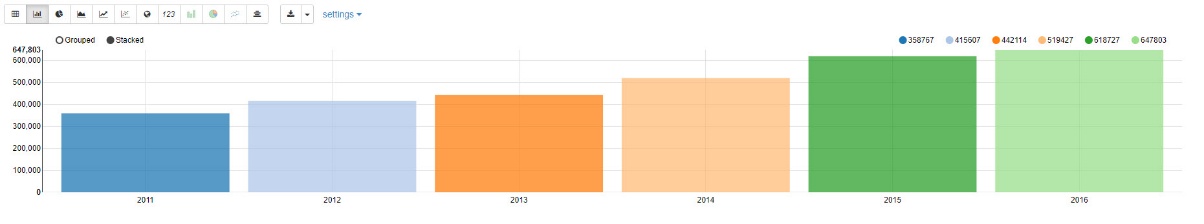
--2) Top 10 Employers to file H1B

SELECT employer\_name, count(employer\_name)as count

FROM tutorials.h1b\_parquet

GROUP BY employer\_name

ORDER BY count desc limit 10;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q1.jpg)

1. **Find the top 10 employers to file H1B**

Copy to Clipboard

Syntax Highlighter

1

%hive

2

-- Create a User Managed table to read H1b parquet files

3

​

4

CREATE External TABLE tutorials.h1b\_parquet (

5

CASE\_STATUS string,

6

employer\_name string,

7

soc\_name string,

8

job\_title string,

9

full\_time\_position string,

10

prevailing\_wage int,

11

worksite string,

12

lon double,

13

lat double)

14

Partitioned By (year int)

15

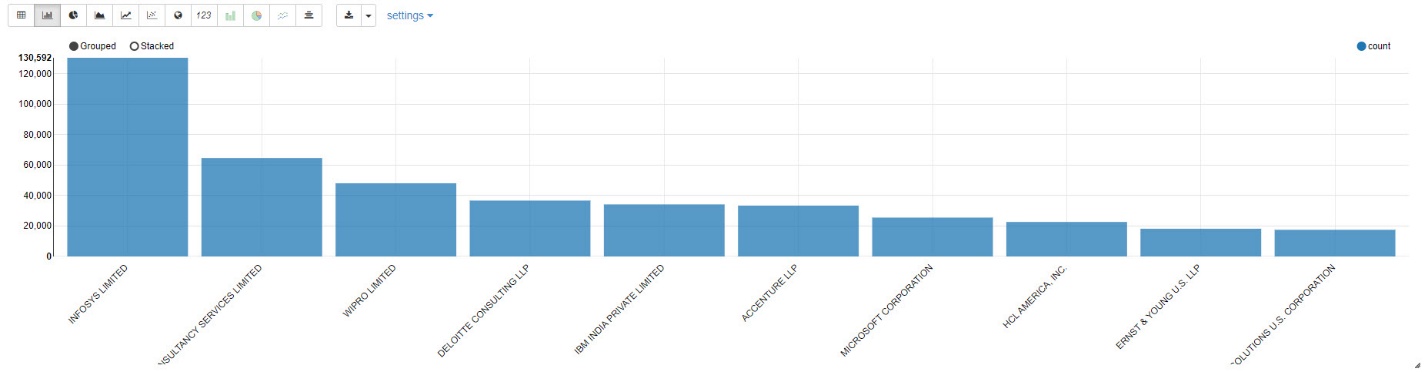
Stored as parquet

16

location '/tutorials/uses\_cases/h1b/parquet'

17

TBLPROPERTIES ("parquet.compression"="SNAPPY");

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q2.jpg)

1. **Find the top job titles with highest application**

Copy to Clipboard

Syntax Highlighter

1

%hive

2

-- 3) Top job titles with highest application over the year 2008-2017

3

​

4

SELECT year, job\_title, cnt

5

FROM

6

(SELECT year, job\_title, count(\*) as cnt, RANK()

7

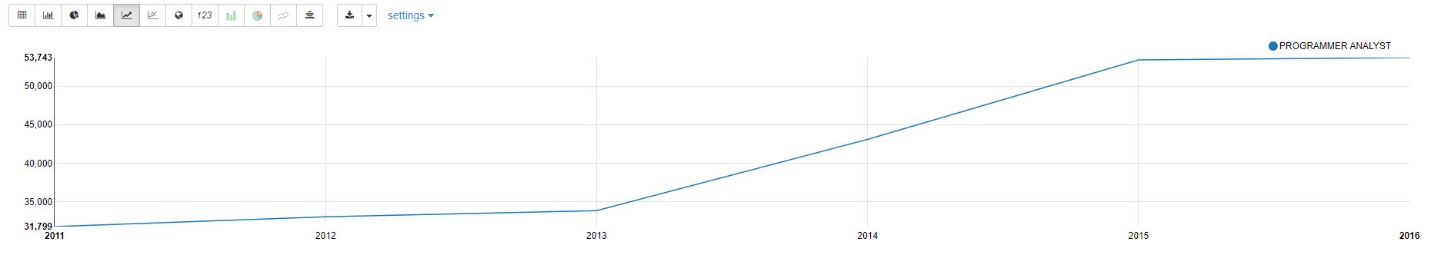
OVER (PARTITION BY year ORDER BY count(\*) DESC) as rnk

8

FROM tutorials.h1b\_parquet GROUP BY year, job\_title) as tg

9

WHERE rnk=1;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q3.jpg)

1. **Find the average salary for a Data Engineer position**

Copy to Clipboard

Syntax Highlighter

1

%hive

2

-- 4) Average Salary for an DATA ENGINEER position

3

​

4

SELECT year, round(AVG(prevailing\_wage),2) AS AVG\_SALARY FROM tutorials.h1b\_parquet

5

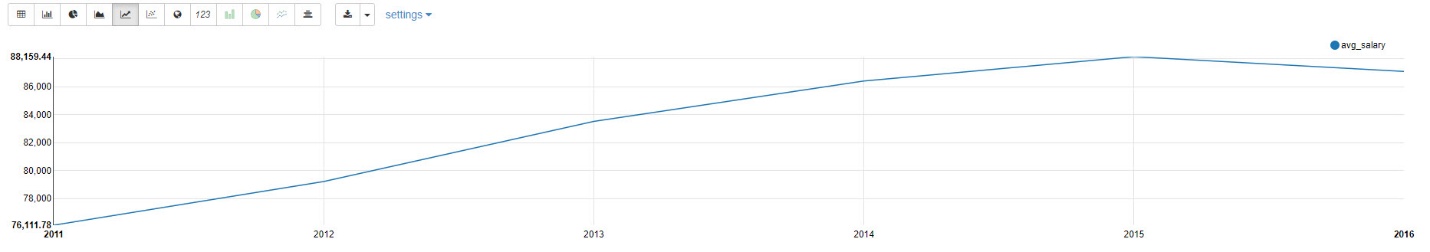
WHERE JOB\_TITLE LIKE 'DATA ENGINEER%'

6

GROUP BY year

7

ORDER BY year;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q4.jpg)

1. **Find highest average salaries based on job titles**

Copy to Clipboard

Syntax Highlighter

1

%hive

2

-- 5) Highest Average Salaries Based on Job Titles

3

​

4

SELECT JOB\_TITLE, round(AVG(prevailing\_wage),2) AS AVG\_SALARY from tutorials.h1b\_parquet

5

WHERE JOB\_TITLE = 'ANALYST' OR

6

JOB\_TITLE ='DATA SCIENTIST' OR

7

JOB\_TITLE ='DATA ENGINEER' OR

8

JOB\_TITLE ='CONSULTANT'

9

AND CASE\_STATUS = 'CERTIFIED'

10

GROUP BY JOB\_TITLE

11

ORDER BY AVG\_SALARY DESC

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q5.jpg)

1. **Find top 10 highest average salaries based on Worksite State**

%hive

-- 6) Top 10 Highest Average Salaries Based on Worksite State

SELECT WORKSITE,round(AVG(prevailing\_wage),2) AS AVG\_SALARY

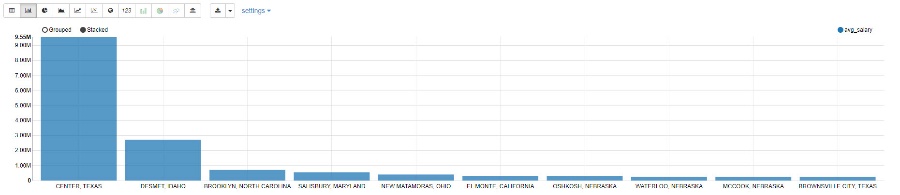
FROM tutorials.h1b\_parquet

WHERE CASE\_STATUS = 'CERTIFIED'

GROUP BY WORKSITE

ORDER BY AVG\_SALARY DESC

LIMIT 10;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q6.jpg)

1. **Find top most hiring states**

%hive

-- 7) top most hiring states

SELECT count(SUBSTR(WORKSITE,(INSTR(WORKSITE,",")+2), length(WORKSITE))) count,

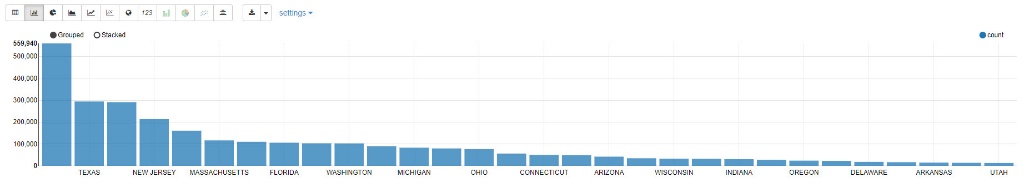
SUBSTR(WORKSITE,(INSTR(WORKSITE,",")+2), length(WORKSITE)) state

FROM tutorials.H1B\_parquet

GROUP BY (SUBSTR(WORKSITE,(INSTR(WORKSITE,",")+2),length(WORKSITE)))

order by count desc

limit 30;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q7.jpg)

1. **Find the ratio of Full time and Half time position**

%hive

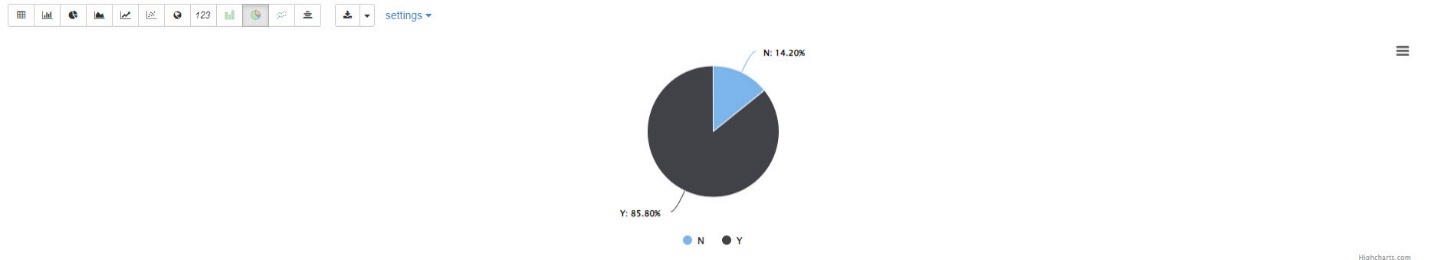
--8) Ratio of Full time and Half time position

select full\_time\_position, count(\*)

FROM tutorials.H1B\_parquet

Where full\_time\_position != 'NA'

Group by full\_time\_position;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q8.jpg)

1. **Find the ratio of all Case\_Status**

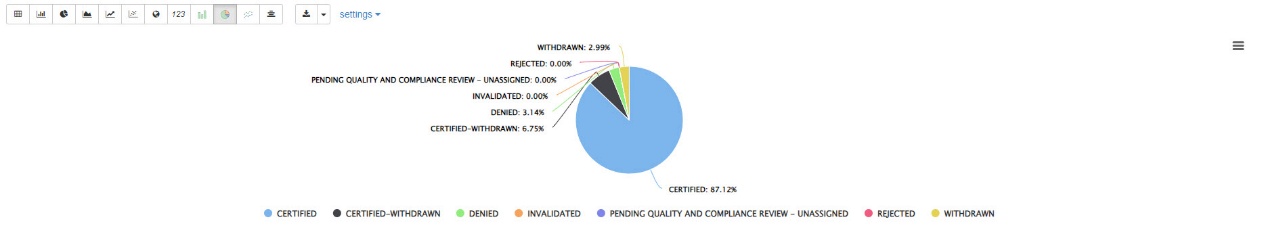
%hive

-- 9) Ratio of all case\_status

select case\_status, count(\*) count

from tutorials.h1b\_parquet

group by case\_status;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q9.jpg)

1. **Find if the number of petitions with Data Engineer job title increasing over time**

%hive

-- 10) Is the number of petitions with Data Engineer job title increasing over time?

select year,job\_title,count(\*) as count

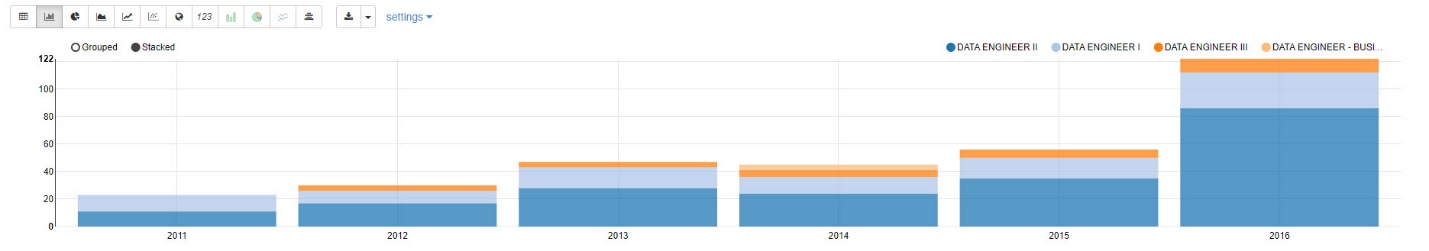
from tutorials.h1b\_parquet

where job\_title LIKE 'DATA ENGINEER %'

group by year,job\_title

having count > 2

order by year, count ;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q10.jpg)

1. **Find which industry has the most number of Data Scientist positions**

%hive

-- 11) Which industry (SOC\_NAME) has the most number of Data Scientist positions?

​select soc\_name,job\_title,count(\*)as total

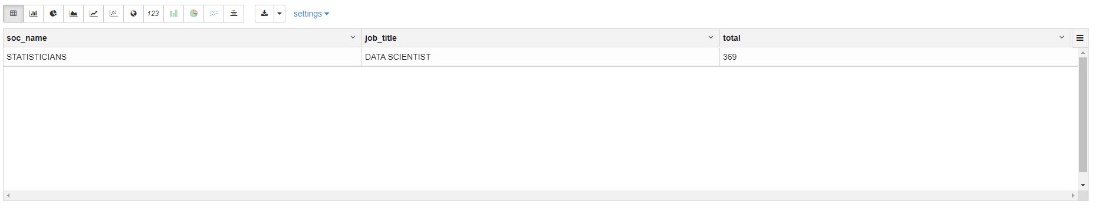
from tutorials.h1b\_parquet

where case\_status="CERTIFIED" and job\_title Like "%DATA SCIENTIST%"

group by soc\_name,job\_title

order by total desc

limit 1;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q11.jpg)

1. **Find the count of Case\_Status count per Case\_Status type**

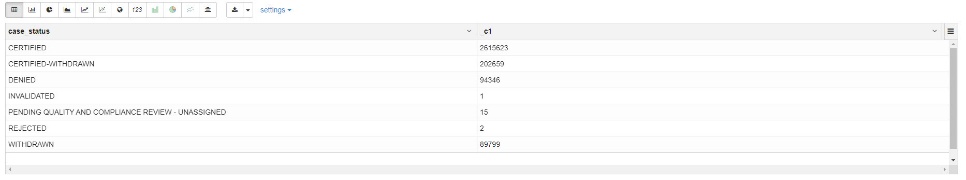
%hive

-- 12) case\_status count per case\_status type

​select case\_status, count(\*)

from tutorials.h1b\_parquet

group by case\_status;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q12.jpg)

1. **Find the count of Case\_Status per year**

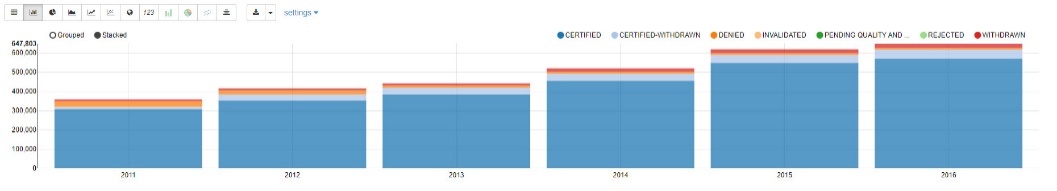
%hive

-- 13) case\_status count per year

select year, case\_status, count(case\_status)

from tutorials.h1b\_parquet

group by year, case\_status

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q13.jpg)

1. **Find the most popular top 10 job positions for each year**

%hive

--14) Find the most popular top 10 job positions for H1B visa applications for each year?

--b) for only certified applications.

select year,job\_title,count(\*)as total

from tutorials.h1b\_parquet

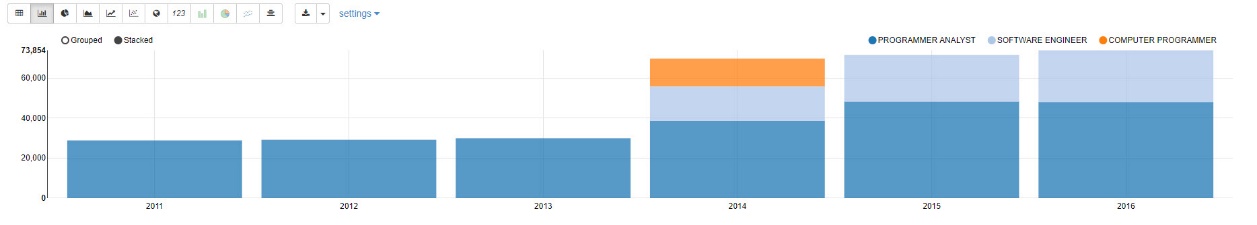
where case\_status='CERTIFIED'

group by year,job\_title

having total > 0

order by total desc

limit 10;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q14.jpg)

1. **Find the average Prevailing Wage for each job for each year**

%hive

--15) Find the average Prevailing Wage for each Job for each Year (take part time and full time

--separate). Arrange the output in descending order - [Certified and Certified Withdrawn.]

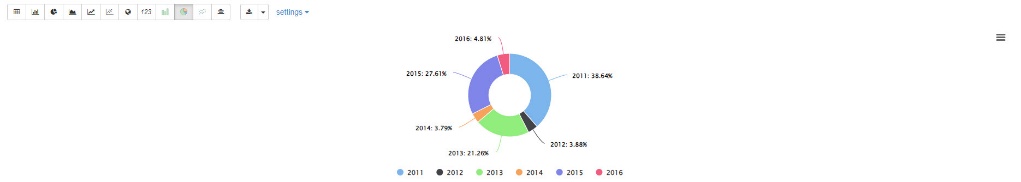
select year,job\_title,full\_time\_position,avg(prevailing\_wage)as average

from tutorials.h1b\_parquet

where full\_time\_position='Y' and (case\_status='CERTIFIED' or case\_status='CERTIFIED-WITHDRAWN')

group by year,job\_title,full\_time\_position

order by average desc;

[](http://localhost/wp-content/uploads/2023/07/h1b-analysis-q15.jpg)

Stop Spark context to free resources.

%spark

sc.stop

Open Zeppelin Note

[H1B Visa Application Analysis – Hive](http://localhost:19995/#/notebook/2J9RPQQZX)

[H1B Visa Application Analysis – Trino](http://localhost:19995/#/notebook/2KHVXW6TQ)