

TEAM -10 : Analysis of Health Insurance Claim Data

Introduction

Healthcare is a booming industry in today's market. As the value of healthcare is increasing the insurances are getting more premium. According to the government, the total Medicare spending increased exponentially due to increase in presence of chronic diseases. People affected by high blood pressure, heart diseases, depression are the highest insurance claimers.

As the cost of diagnosis is quite ambiguous, insurance companies have raised their premiums as more and more patients are claiming their insurance due to bad health conditions.

Some of the supporting facts are:

- People having Medical Insurance coverage in USA – 299.5 million
- Average number of claims in the country that saves lives – 68,000
- People having Medical Insurance coverage in Maryland - 5.9 million
- Most common disease or accident on which insurance was claimed – 1+ million

Problem Statement

Our problem is to analyze the trends of the claims from data and helping insurance companies in their decision making.

Role of Data Management

Data plays a crucial part to understand the pattern of the fraud claims. It is important to process and manage the data efficiently. Relational Databases are the simplest one which helps when data is small. It removes the redundancy and help in organizing the data in a better way. As cloud data bases are emerging and are easy to configure, we have leveraged AWS to store and query the data.

Tableau on Athena

Sample Architecture



Fig.1 : Flow of DataPipeline

TEAM -10 : Analysis of Health Insurance Claim Data

We have used the AWS standard architecture to integrate data pipeline with tableau for data visualization. The architecture has helped to seamless data flow from S3 to tableau without any hassle. The above picture has a brief explanation of the components.

Data Collection

Source: KAGGLE

Link: <https://www.kaggle.com/datasets/rohitrox/healthcare-provider-fraud-detection-analysis>

In the dataset we have three common separated files which are as follows:

1. Inpatient Data: This data provides insights about the claims filed for those patients who are admitted in the hospitals. It also provides additional details like their admission and discharge dates and admit diagnosis code.

Volume of data: 50K records and 27 attributes

2. Outpatient Data: This data provides details about the claims filed for those patients who visit hospitals and not admitted in it.

Volume of data: 643k records and 30 attributes

3. Beneficiary Details Data: This data contains beneficiary KYC details like health conditions, region they belong to etc.

Volume of data: 202k records and 27 attributes

Steps involved in Data Pipeline

- RDS Database created in AWS named 'database-aws'
- Assigned respective IAM roles for the team members
- We have stored the **3 CSV** files in **S3 bucket** of AWS to store the data.
- After that, we have crawled the data from AWS Glue Crawler and Connected Athena service to our cloud data.
- Integrated Tableau with AWS RDS database to plot charts.

TEAM -10 : Analysis of Health Insurance Claim Data

- Ran data queries on AWS Athena for Data Analysis.

ER DIAGRAM

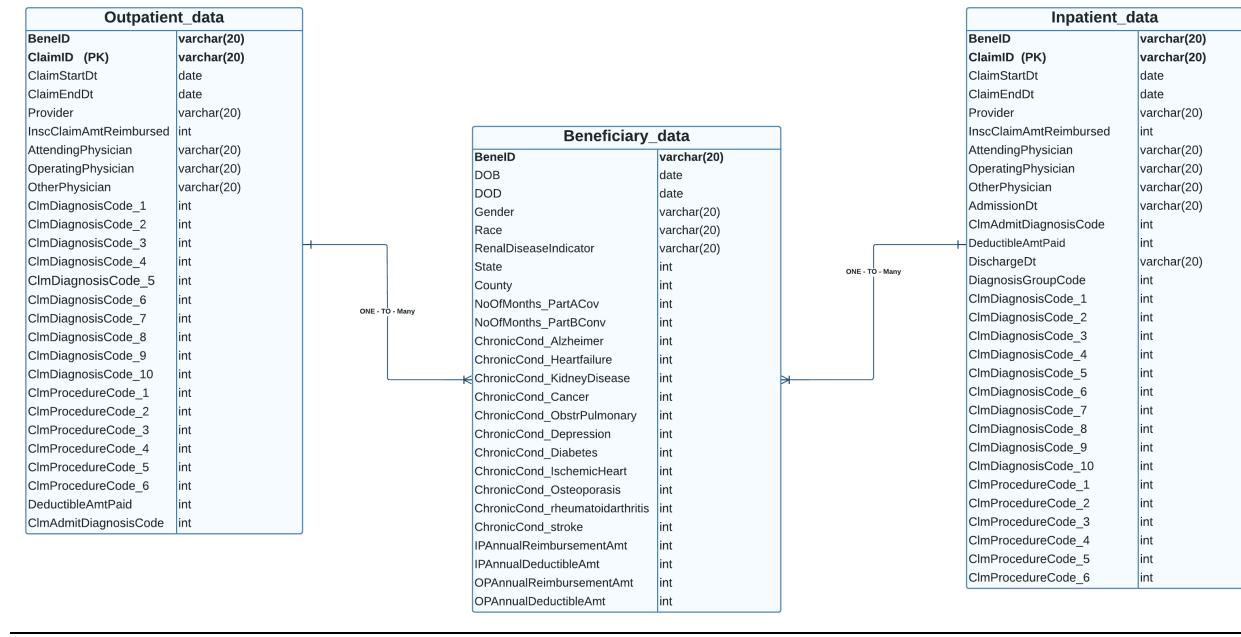


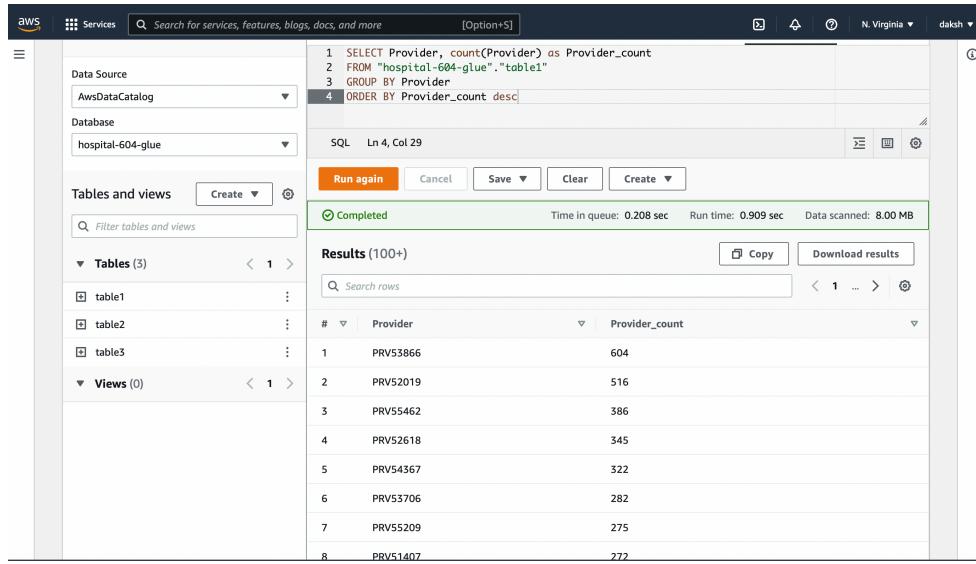
Fig. 2: Entity Relational Diagram

The ER diagram shows the relational between the 3 tables. This helped us to understand the relation between three entities.

TEAM -10 : Analysis of Health Insurance Claim Data

Data Analysis using SQL

1. Count of unique providers in Inpatient Data



The screenshot shows the AWS Lambda SQL interface. The left sidebar displays the Data Source (AwsDataCatalog) and Database (hospital-604-glue). The Tables and views section lists three tables: table1, table2, and table3. The main area contains a SQL query window with the following code:

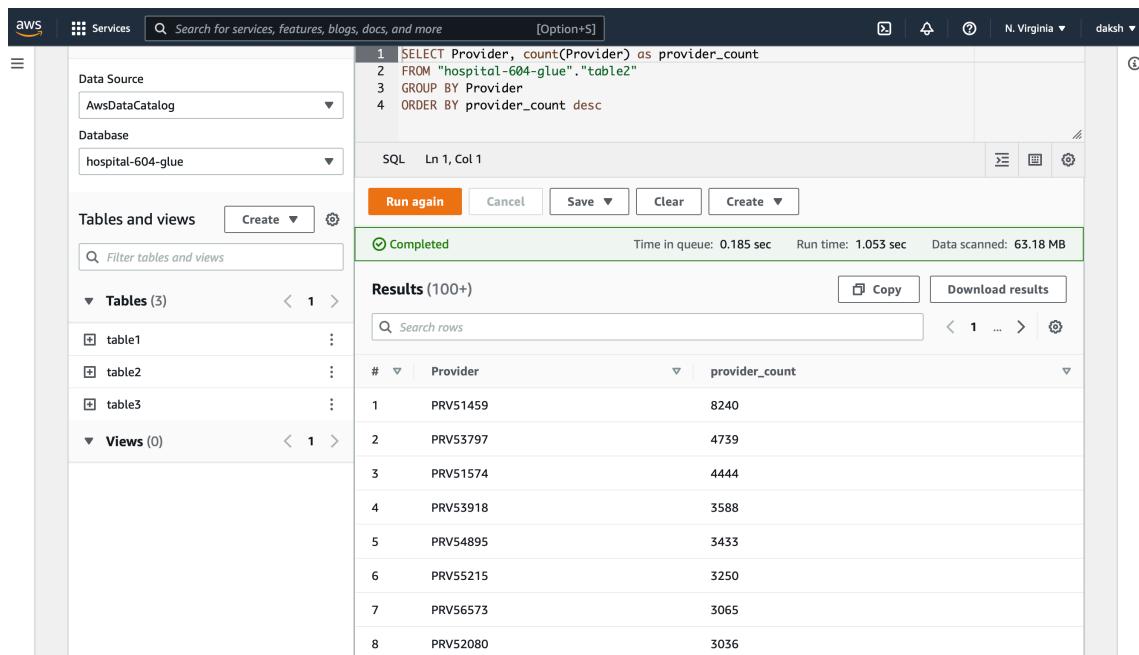
```
1 SELECT Provider, count(Provider) as Provider_count
2 FROM "hospital-604-glue"."table1"
3 GROUP BY Provider
4 ORDER BY Provider_count desc
```

The status bar indicates the query completed with a run time of 0.909 sec and 8.00 MB scanned.

Results (100+)

#	Provider	Provider_count
1	PRV53866	604
2	PRV52019	516
3	PRV55462	386
4	PRV52618	345
5	PRV54367	322
6	PRV53706	282
7	PRV55209	275
8	PRV51407	272

2. Count of unique providers in Outpatient Data



The screenshot shows the AWS Lambda SQL interface. The left sidebar displays the Data Source (AwsDataCatalog) and Database (hospital-604-glue). The Tables and views section lists three tables: table1, table2, and table3. The main area contains a SQL query window with the following code:

```
1 SELECT Provider, count(Provider) as provider_count
2 FROM "hospital-604-glue"."table2"
3 GROUP BY Provider
4 ORDER BY provider_count desc
```

The status bar indicates the query completed with a run time of 1.053 sec and 63.18 MB scanned.

Results (100+)

#	Provider	provider_count
1	PRV51459	8240
2	PRV53797	4739
3	PRV51574	4444
4	PRV53918	3588
5	PRV54895	3433
6	PRV55215	3250
7	PRV56573	3065
8	PRV52080	3036

TEAM -10 : Analysis of Health Insurance Claim Data

3. Count of unique Beneficiary ID in Inpatient Data

The screenshot shows the AWS Glue Data Catalog Editor interface. The left sidebar displays the Data Source (AwsDataCatalog) and Database (hospital-604-glue). The Tables and views section lists three tables: table1, table2, and table3. The main area shows a query editor with the following SQL code:

```
1 SELECT (count(DISTINCT beneid))
2 FROM "hospital-604-glue"."table3"
```

The results pane shows the output of the query:

#	_col0
1	148072

Below the results, the status bar indicates: Completed, Time in queue: 0.15 sec, Run time: 1.054 sec, Data scanned: 14.84 MB.

4. Count of unique Beneficiary ID in Outpatient Data

AWS

The screenshot shows the AWS Glue Data Catalog Editor interface. The left sidebar displays the Data Source (AwsDataCatalog) and Database (hospital-604-glue). The Tables and views section lists five tables: table1, table2, table3, table4, and table5. The main area shows a query editor with the following SQL code:

```
1 SELECT (count(DISTINCT beneid))
2 FROM "hospital-604-glue"."table1"
3
```

The results pane shows the output of the query:

#	_col0
1	37418

Below the results, the status bar indicates: Completed, Time in queue: 0.198 sec, Run time: 0.682 sec, Data scanned: 8.00 MB.

TEAM -10 : Analysis of Health Insurance Claim Data

5. %age of the Gender

The screenshot shows the AWS Glue Data Catalog interface. The left sidebar displays 'Data Source' set to 'AwsDataCatalog' and 'Database' set to 'hospital-604-glue'. The main area shows a list of tables ('table1', 'table2', 'table3') and views (0). A query editor window is open, showing the following SQL code:

```
1 SELECT count(Gender)*100/sum(count(Gender)) over(), '%'
2 FROM "hospital-604-glue"."table3"
3 GROUP BY Gender
```

The status bar indicates the query is 'Completed' with a run time of 0.774 sec and data scanned of 14.84 MB. The results table shows two rows:

#	_col0	_col1
1	57	%
2	42	%

6. Inner Join Query

The screenshot shows the AWS Glue Data Catalog interface. The left sidebar displays 'Data Source' set to 'AwsDataCatalog' and 'Database' set to 'hospital-604-glue'. The main area shows a list of tables ('table1', 'table2', 'table3') and views (0). A query editor window is open, showing the following SQL code:

```
1 SELECT "hospital-604-glue"."table1".BenefID , "hospital-604-glue"."table1".ClaimID , "hospital-604-glue"."table2".ClaimID AS Out_ClaimID , "hospital-604-glue"."table1".InscClaimAmtReimbursed , "hospital-604-glue"."table2".InscClaimAmtReimbursed AS Out_InscClaimAmtReimbursed
2 FROM "hospital-604-glue"
3 INNER JOIN "hospital-604-glue"."table1"
4 ON "hospital-604-glue"."table1".BenefID = "hospital-604-glue"."table2".BenefID
5 ORDER BY "hospital-604-glue"."table1".BenefID
```

The status bar indicates the query is 'Completed' with a run time of 2.685 sec and data scanned of 71.18 MB. The results table shows multiple rows of joined data:

#	BeneID	ClaimID	Out_ClaimID	InscClaimAmtReimbursed	Out_InscClaimAmtReimbursed
1	BENE100002	CLM47861	CLM262323	12000	80
2	BENE100002	CLM47861	CLM623189	12000	60
3	BENE100002	CLM47861	CLM240071	12000	300
4	BENE100002	CLM47861	CLM173783	12000	0
5	BENE100002	CLM47861	CLM689063	12000	300
6	BENE100002	CLM47861	CLM689063	12000	300

TEAM -10 : Analysis of Health Insurance Claim Data

7. Finding Null Values

The screenshot shows the AWS Glue Data Catalog interface. On the left, there's a sidebar with 'Data Source' set to 'AwsDataCatalog' and 'Database' set to 'hospital-604-glue'. Under 'Tables and views', there are three tables: 'table1', 'table2', and 'table3'. The 'table3' row is expanded to show its columns: DOB, DOD, BeneID, Gender, and State. A SQL query is being run in the main panel:

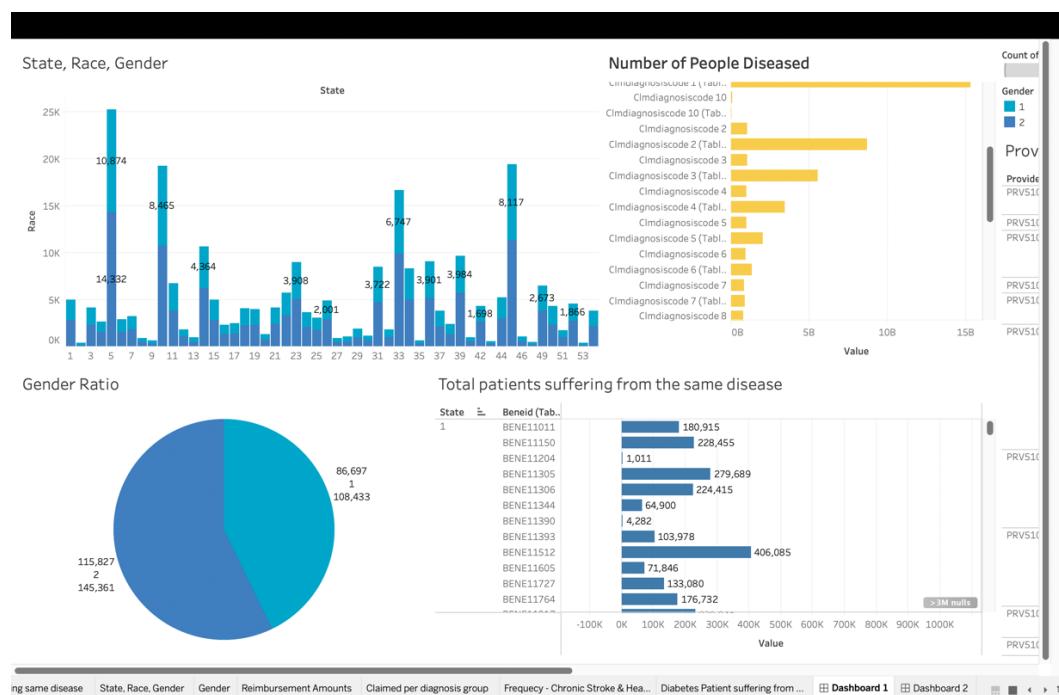
```

1 SELECT DOB, DOD, BeneID, Gender, State
2 FROM "hospital-604-glue"."table3"
3 WHERE IS_NAN(nan())
4
    
```

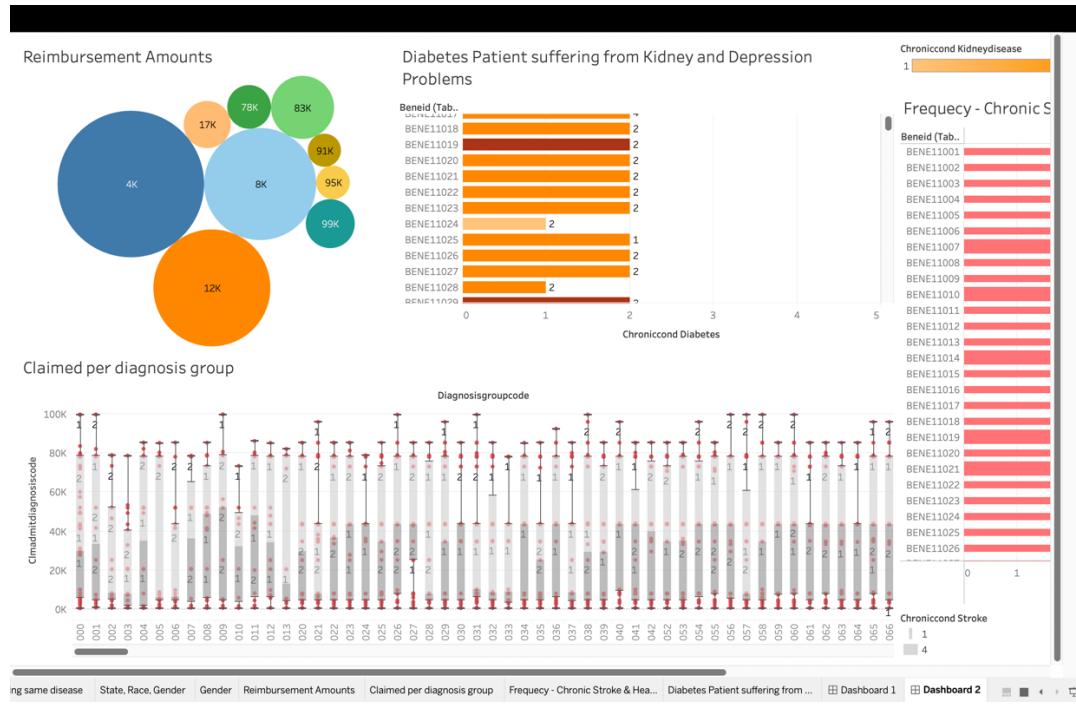
The status bar at the bottom indicates the query completed successfully with a time of 0.212 sec, a run time of 1.139 sec, and 14.84 MB of data scanned.

Data Visualization

Dashboard:

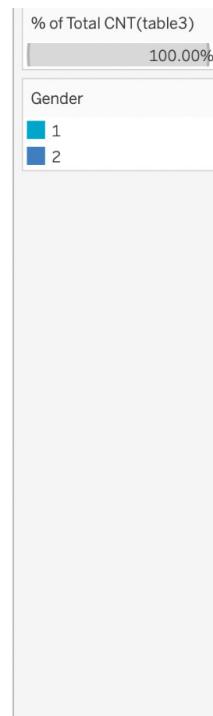
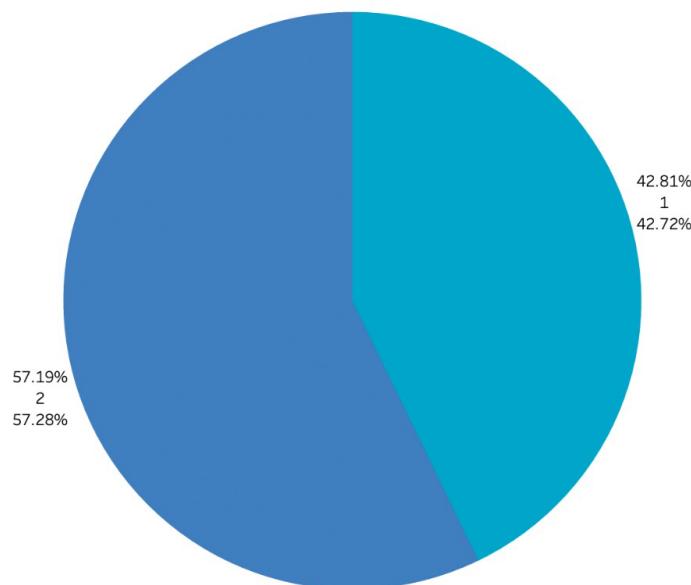


TEAM -10 : Analysis of Health Insurance Claim Data



1. Gender Ratio

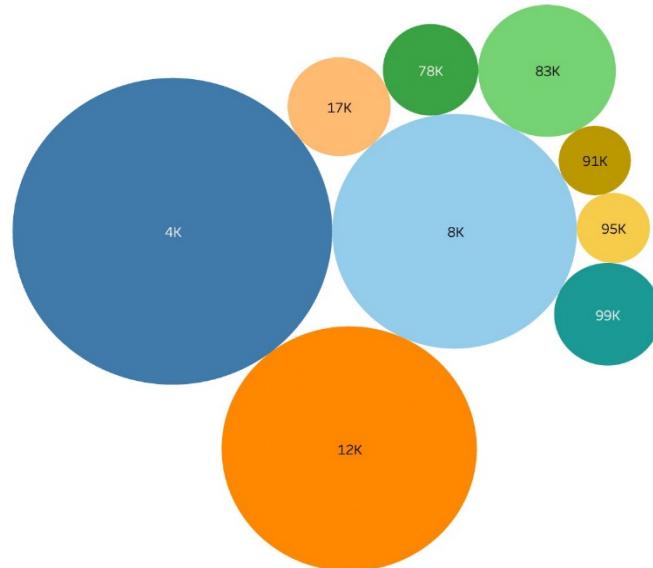
Gender Ratio



TEAM -10 : Analysis of Health Insurance Claim Data

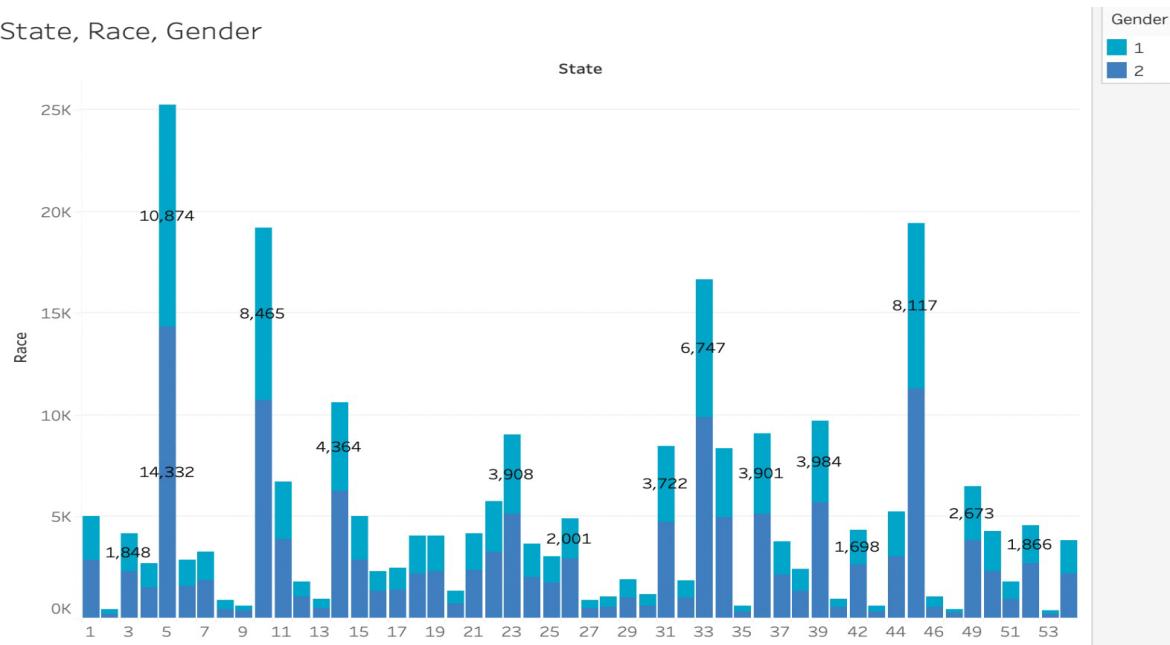
2. Reimbursement Amounts

Reimbursement Amounts



3. State VS Race VS Gender

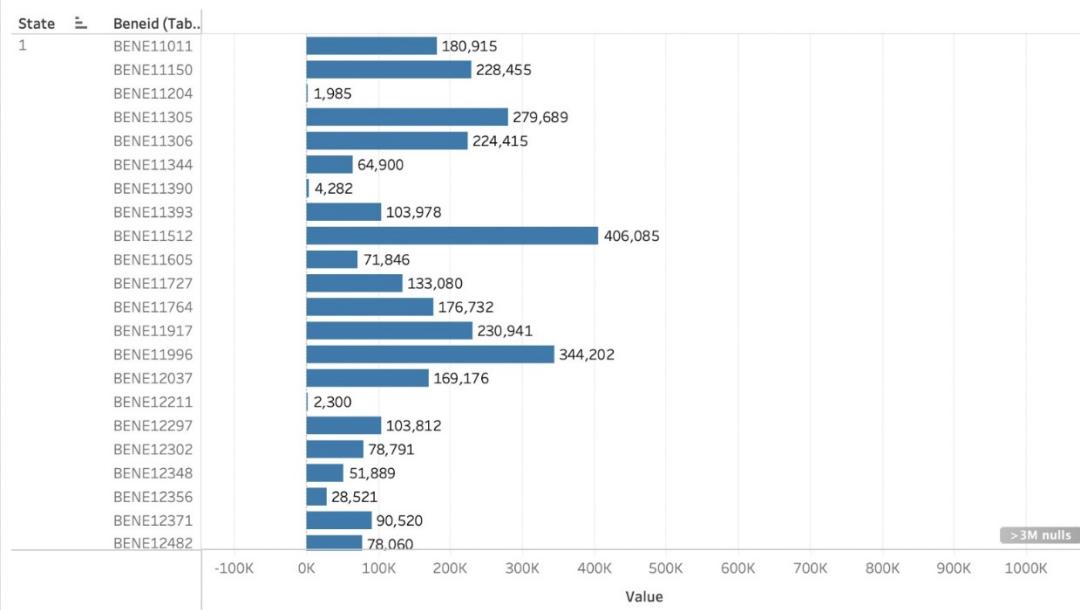
State, Race, Gender



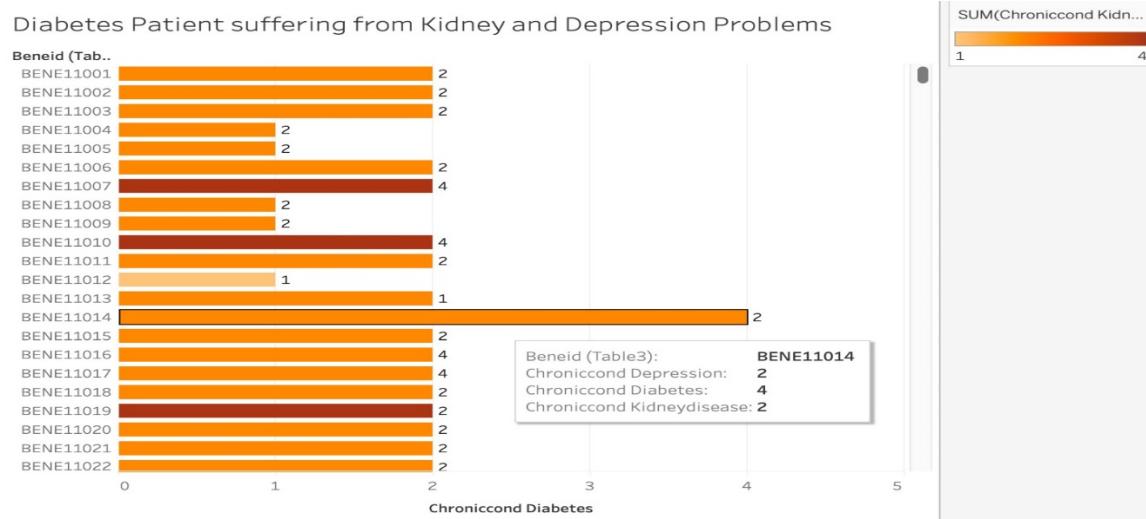
4. Patients suffering from disease

TEAM -10 : Analysis of Health Insurance Claim Data

Total patients suffering from the same disease



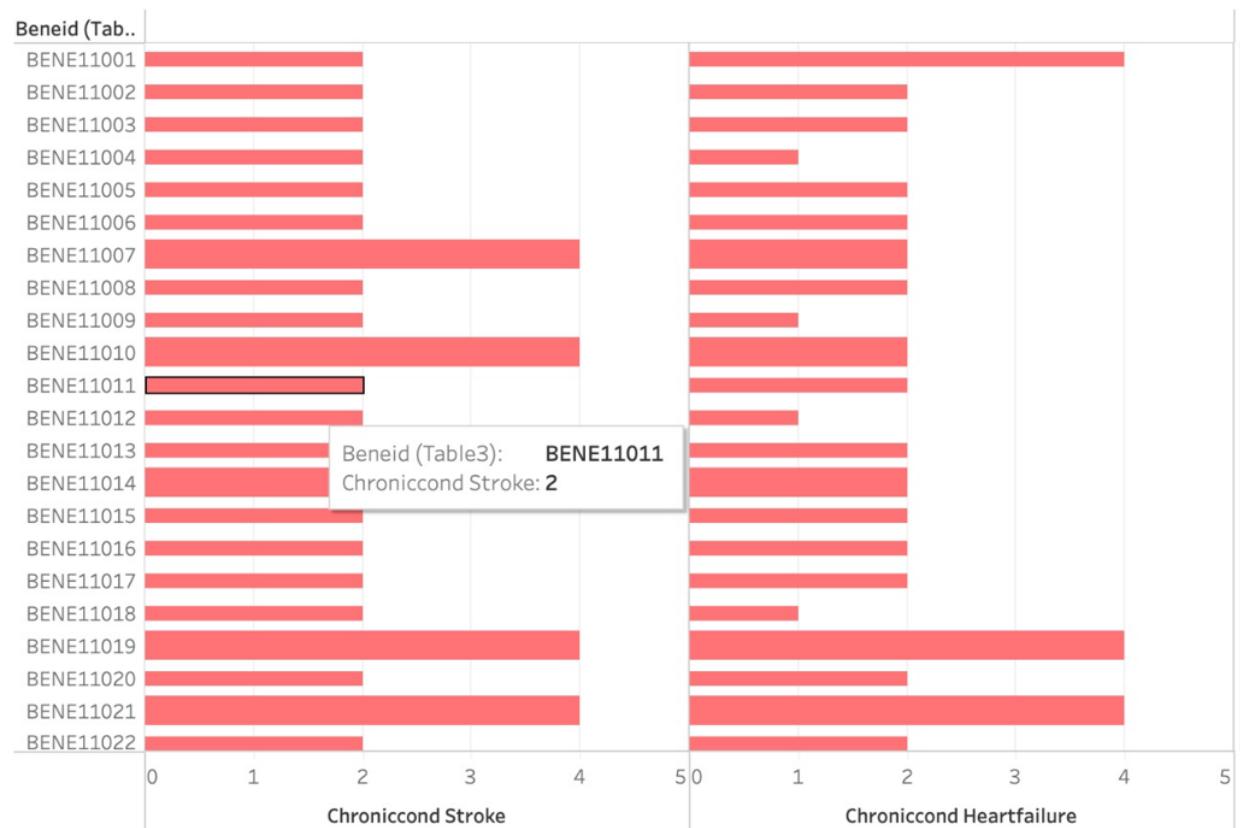
5. Diabetes Patients suffering other diseases



6. Number of Chronic Stroke & Heart Failures

TEAM -10 : Analysis of Health Insurance Claim Data

Frequency - Chronic Stroke & Heart Failure



TEAM -10 : Analysis of Health Insurance Claim Data

References

1. How Many Americans Don't Have Health Insurance In 2022?

<https://www.simplyinsurance.com/how-many-americans-dont-have-health-insurance/>

2. Galvani, A. P., Parpia, A. S., Foster, E. M., Singer, B. H., & Fitzpatrick, M. C. (2020). Improving the Prognosis of Healthcare in the United States. *Lancet* (London, England), 395(10223), 524.

[https://doi.org/10.1016/S0140-6736\(19\)33019-3](https://doi.org/10.1016/S0140-6736(19)33019-3)

3. Health insurance status Maryland population 2020 | Statista.

<https://www.statista.com/statistics/238765/health-insurance-status-of-the-total-population-of-maryland/>

4. *Top 5 critical illness claims* | BenefitsPRO.

<https://www.benefitspro.com/2012/06/22/top-5-critical-illness-claims/?slreturn=20220416173943>