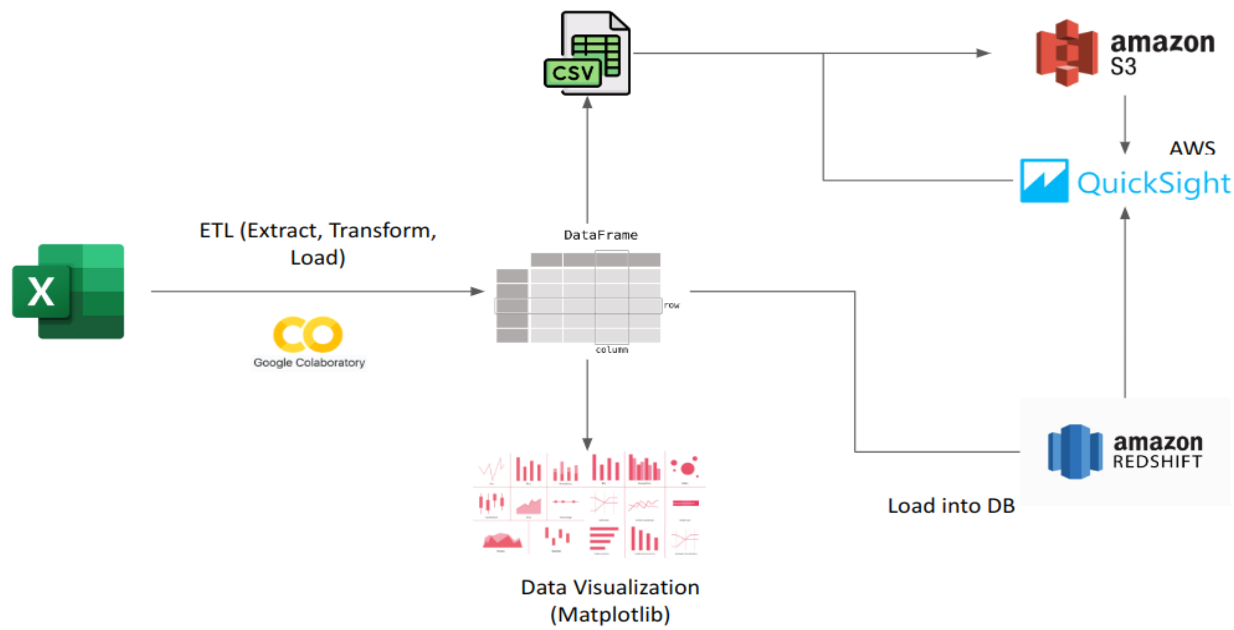
# ETL Pipeline Documentation

The following ETL Pipeline is built using Python programming language and its associated tools & frameworks. The end-to- end pipeline works on cloud environments using Google Colab Notebook and Amazon Web Services (AWS) tools.

Below is the architecture of ETL pipeline being built -




Components Involved -

1. Google Colab Notebook (Python 3.9 configured)
2. AWS SDK (with associated AWS account)
    a. AWS S3
    b. Amazon Redshift
    c. AWS QuickSight
3. Python packages (Pandas,NumPy,Matplotlib,SQLAlchemy,boto3,psycopg2)

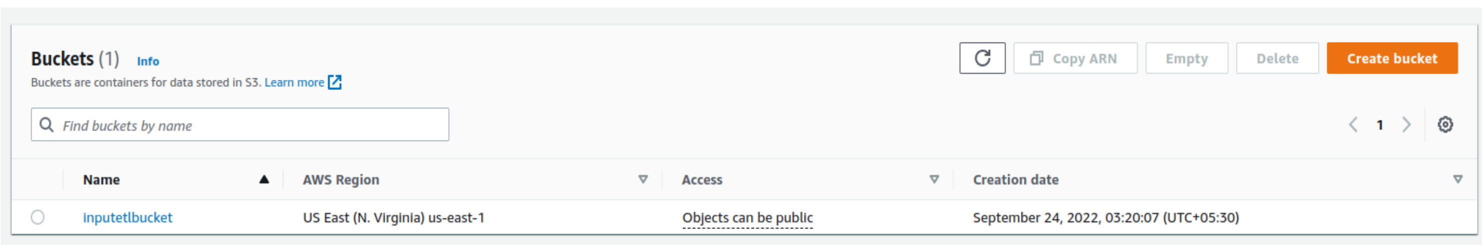
The steps of processes involved in creating and triggering the pipeline are as follows -

# 1. Setting up AWS components

## Create AWS S3 Bucket

S3 Bucket will store the processed CSV files which can then be used as a source for Data Analytics and Visualization.

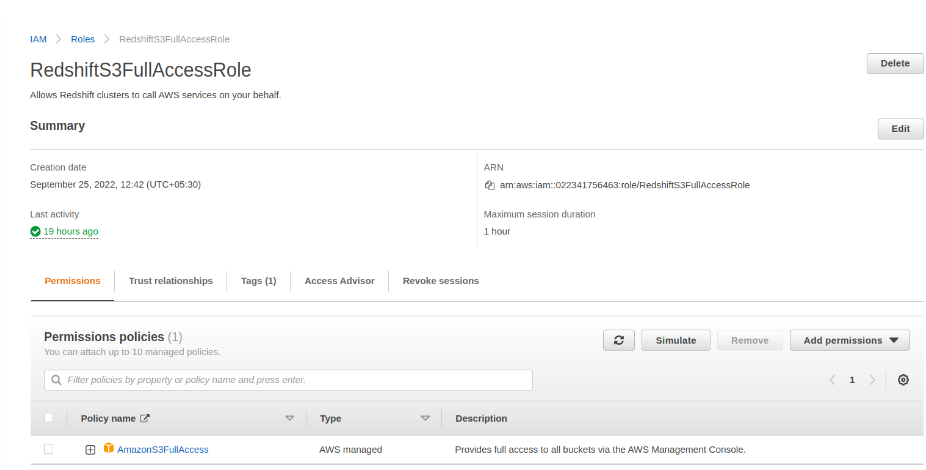
1. Create S3 bucket (public) using AWS Management Console
2. Using Python AWS SDK (boto3) connect with S3 API.
3. Bucket Name - inputetlbucket




## Create AWS IAM Role

The IAM Role created will give access to Redshift to access S3 bucket files to load data if needed. (*optional)

1. IAM Role Name - RedshiftS3FullAccessRole
2. Policy Attached - AmazonS3FullAccess



# 2. Installing Python Packages

SInce the ETL Pipeline is run using Google Colab, external python packages are essential.

The list of packages which are installed first are -

```
[ ]  import boto3
     import pandas as pd
     import numpy as np
     import psycopg2
     import json
     import matplotlib.pyplot as plt
     import sqlalchemy as sa
     from sqlalchemy.engine.url import URL
     from sqlalchemy import orm as sa_orm
```

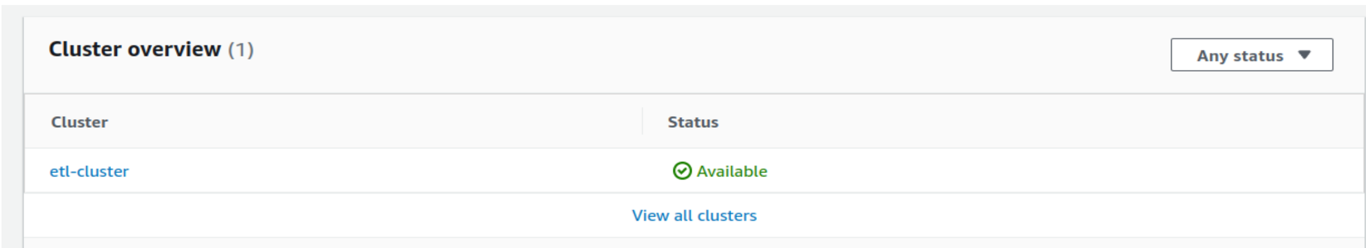
# 3. Python- AWS SDK Configuration

1. Using boto3 we will connect with EC2 resource, S3 resource, Redshift client and IAM resource for code integration in AWS platform.
2. EC2 resource is used to get the default VPC and attach that to AWS Redshift Cluster.
3. S3 client will be used fetch bucket and add files to bucket during Load process of ETL pipeline
4. The Redshift client will be used to load data from dataframe to database.
5. IAM role will be used to perform the following actions -
    a. Attach RedshiftS3FullAccessRole to Redshift
    b. Get Redshift Cluster ARN to perform actions on it.

# 4. Create Redshift Cluster

Redshift Cluster is our target DB (Data Warehouse) to store the final dataset entity for Data Analytics and Visualization.

The created cluster is assigned into VPC configured default.




```python
# Create Redshift Cluster
try:
    response = redshift.create_cluster(
        ClusterType=CLUSTER_TYPE,
        NodeType=NODE_TYPE,
        DBName=DB_NAME,
        ClusterIdentifier=CLUSTER_IDENTIFIER,
        MasterUsername=DB_USER,
        MasterUserPassword=DB_PASSWORD,
        IamRoles=[redshift_role_arn]
      )
except Exception as e:
    print(e)
```

# 5. Extract data from Spreadsheet

Pandas is used to extract the data from Spreadsheet ( CSV or XLSX format) to create the data frame first. Data Nature is then described using pandas utility functions like head(), isna(), mean(), duplicated(),info(), shape etc.

Based on the nature of data and data types, it is then transformed from a raw data frame to a processed meaningful dataframe.

```
[ ] df = pd.read_excel('WS.xlsx')
```

```
# daata type che check for DB loading
print("Dataframe Shape : ", df.shape)
df.info()
```

```
Dataframe Shape :  (1629, 18)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1629 entries, 0 to 1628
Data columns (total 18 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   ID                          1629 non-null   object
 1   Company Name                1629 non-null   object
 2   Number of Employees         1613 non-null   float64
 3   % Developers Count          1209 non-null   float64
 4   Estimated Developers Count  1248 non-null   float64
 5   Developers on LinkedIn      425 non-null    object
 6   Business Description        1608 non-null   object
 7   Region of Headquarters      1611 non-null   object
 8   Country of Headquarters     1611 non-null   object
 9   Revenue                     1394 non-null   float64
 10  Currency                    1629 non-null   object
 11  TRBC Economic Sector Name   1606 non-null   object
 12  TRBC Business Sector Name   1373 non-null   object
 13  TRBC Industry Group Name    1371 non-null   object
 14  URL                         1591 non-null   object
 15  Sales Person                1629 non-null   object
 16  Sales Person Email          1629 non-null   object
 17  Days Since First Contact    1620 non-null   float64
dtypes: float64(5), object(13)
memory usage: 229.2+ KB
```

```
# check missing data
df.isna().mean()*100
```

```
ID                             0.000000
Company Name                   0.000000
Number of Employees            0.982198
% Developers Count            25.782689
Estimated Developers Count    23.388582
Developers on LinkedIn        73.910374
Business Description           1.289134
Region of Headquarters         1.104972
Country of Headquarters        1.104972
Revenue                       14.426028
Currency                       0.000000
TRBC Economic Sector Name      1.411909
TRBC Business Sector Name     15.715163
TRBC Industry Group Name      15.837937
URL                            2.332719
Sales Person                   0.000000
Sales Person Email             0.000000
Days Since First Contact       0.552486
dtype: float64
```

```
[ ] # check for duplicate data
df.duplicated().mean()*100
```

```
0.0
```

# 6. Transform Data

The set of processes performed during dataset transformation are -

1. Removing unwanted columns
2. Filling Empty cells ( Nan, Null valued) with Aggregate values
3. Populating columns through peer dependent columns ( A & B -> C)
4. Removing rows having empty cells ( > 50-60%)
5. Bringing consistency and uniformity in column by maintaining a single data type.
6. Precising cell values using round functions and type casting.
7. Changing value of denormalized cells.

Some common utility functions used during the process includes -
drop(), isna(), fillna(), loc(),value_counts(),
apply(),mean(),median(),astype()

# 7. Load Data

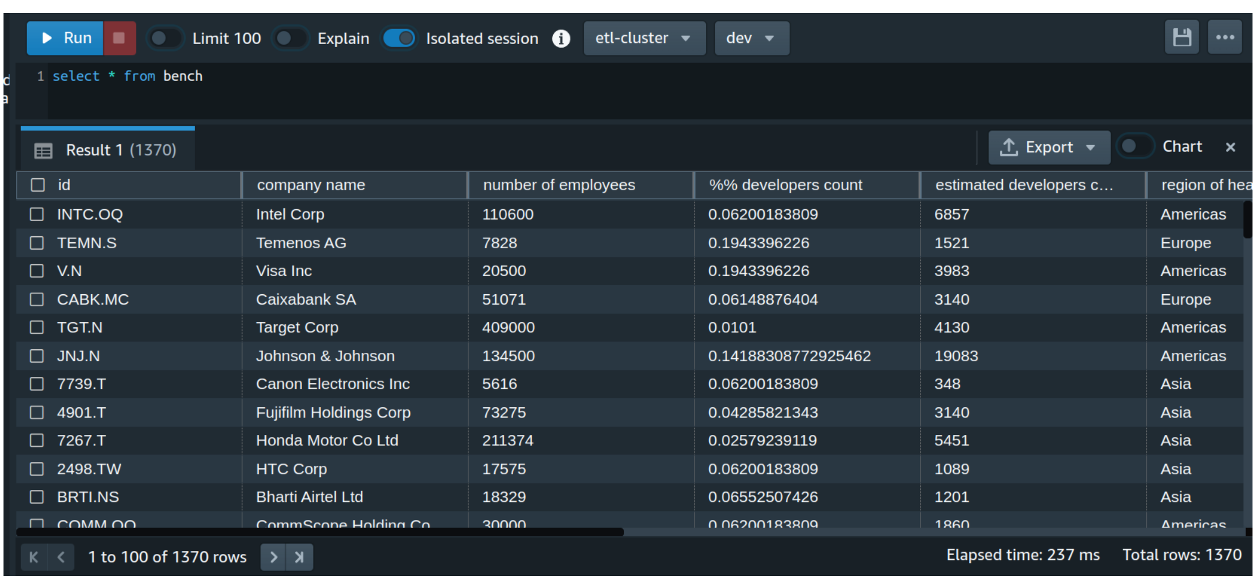
Transformed data is important in terms of insights and analytics so the data need to be stored in Database (DynamoDB) or Data Lake (S3) or Data Warehouse (Redshift).

## 1. Load data into Redshift

The rows and columns of dataframe are mapped with DB Schema and then rows are copied into redshift db.

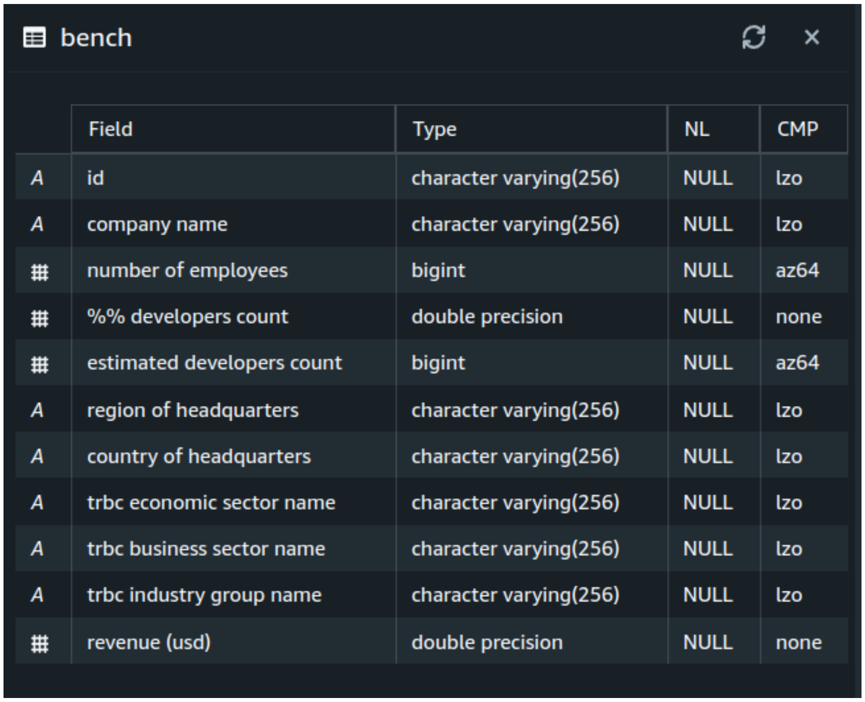
- SQLAlchemy's utility tools like create_engine() and URL prepare to transfer data to DB. to_sql() commands copies data into the result set using the engine.
- Psygopg2 handles IO Errors during data loading.
- Session() is maintained for concurrent and multiple data loading use cases.

```
# load data to db
try:
    df.to_sql('bench', engine, index=False, if_exists='replace')
except psycopg2.Error as e:
    print('Error : Failed creating table !')
    print(e)
```




| id | company name | number of employees | %% developers count | estimated developers c... | region of hea |
|---|---|---|---|---|---|
| INTC.OQ | Intel Corp | 110600 | 0.06200183809 | 6857 | Americas |
| TEMN.S | Temenos AG | 7828 | 0.1943396226 | 1521 | Europe |
| V.N | Visa Inc | 20500 | 0.1943396226 | 3983 | Americas |
| CABK.MC | Caixabank SA | 51071 | 0.06148876404 | 3140 | Europe |
| TGT.N | Target Corp | 409000 | 0.0101 | 4130 | Americas |
| JNJ.N | Johnson & Johnson | 134500 | 0.14188308772925462 | 19083 | Americas |
| 7739.T | Canon Electronics Inc | 5616 | 0.06200183809 | 348 | Asia |
| 4901.T | Fujifilm Holdings Corp | 73275 | 0.04285821343 | 3140 | Asia |
| 7267.T | Honda Motor Co Ltd | 211374 | 0.02579239119 | 5451 | Asia |
| 2498.TW | HTC Corp | 17575 | 0.06200183809 | 1089 | Asia |
| BRTI.NS | Bharti Airtel Ltd | 18329 | 0.06552507426 | 1201 | Asia |
| COMM.OQ | CommScope Holding Co | 30000 | 0.06200183809 | 1860 | Americas |

***Database schema***



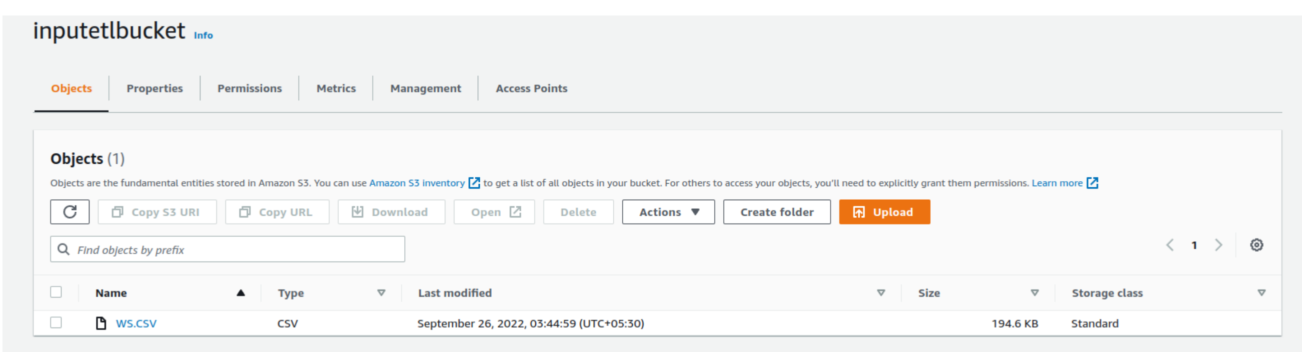
bench

| | Field | Type | NL | CMP |
|---|---|---|---|---|
| A | id | character varying(256) | NULL | lzo |
| A | company name | character varying(256) | NULL | lzo |
| # | number of employees | bigint | NULL | az64 |
| # | %% developers count | double precision | NULL | none |
| # | estimated developers count | bigint | NULL | az64 |
| A | region of headquarters | character varying(256) | NULL | lzo |
| A | country of headquarters | character varying(256) | NULL | lzo |
| A | trbc economic sector name | character varying(256) | NULL | lzo |
| A | trbc business sector name | character varying(256) | NULL | lzo |
| A | trbc industry group name | character varying(256) | NULL | lzo |
| # | revenue (usd) | double precision | NULL | none |

## 2. Load data into S3

The processed data frame is also converted into a csv file which is then stored into AWS S3 bucket using boto3 module.

```
[ ]  # save transformed file as csv and upload to AWS S3
     df.to_csv("WS.csv",index=False)
     s3.Bucket('inputetlbucket').upload_file('WS.csv','WS.CSV')
```



# 8. Metrics and Visualization

The processed data frame as it is ready for data analytics , few metrics have been produced and those metrics and insights are visualized using Data visualization tools and libraries.

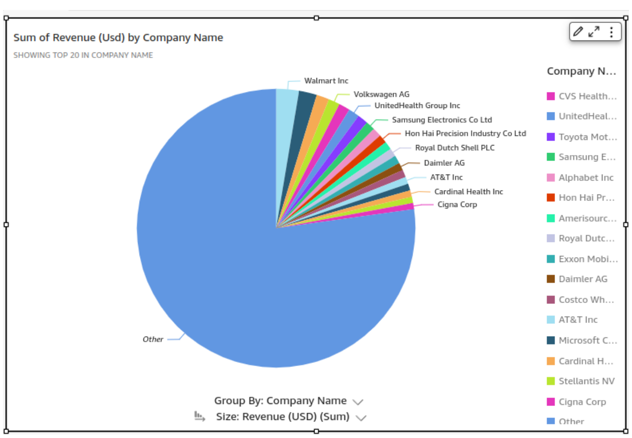
Some of the key metrics produced are as follows -

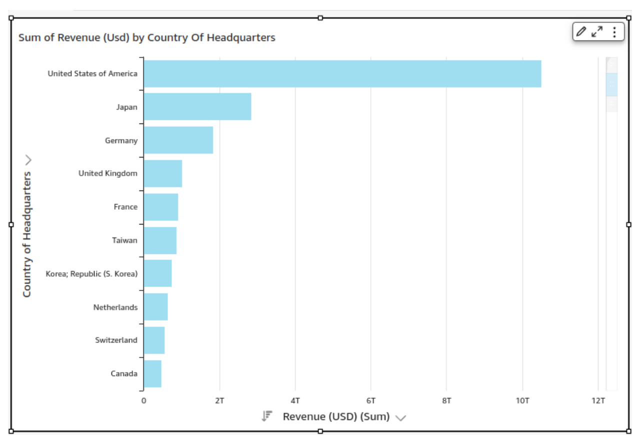
1. Total Revenue by Region of Headquarters
2. Total Revenue by Business Sector
3. Total Revenue by Technology Sector
4. Total Revenue by Economic Sector
5. Employes data by headquarter
6. Estimated developers count by Region of Headquarters
7. Sum of revenue by company

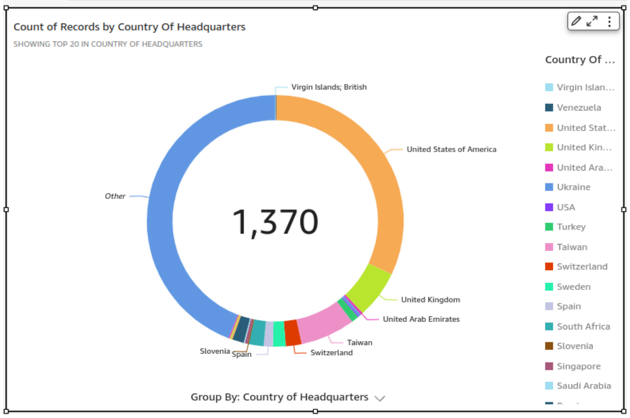
## 1. AWS QuickSight

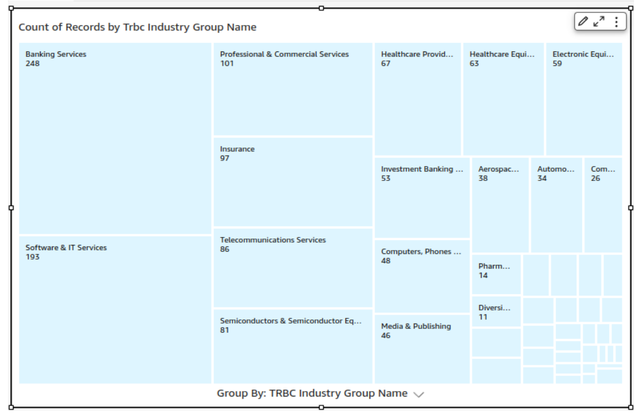
AWS Quicksight is then finally connected to our ETL Pipeline which fetches data through various sources like Redshift, AWS S3 and CSV File.

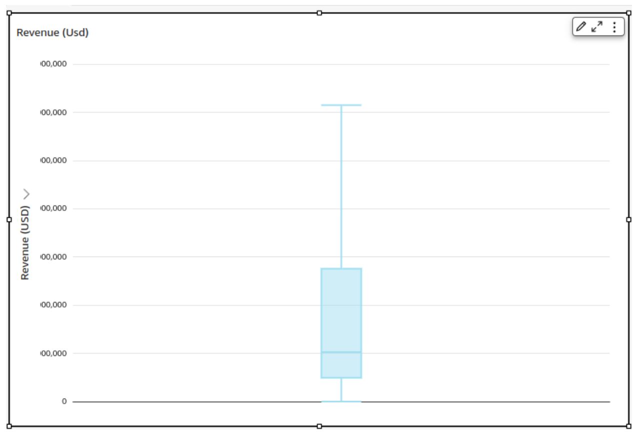
The data is being fed to Quicksight which analyzes the data and creates data insights.






## 2. Matplotlib

Python libraries like Matplotlib are also used to visualize metrics derived from dataframe. Line Chart enhances data visualization showing trends of data.