TASK 2

This Notebook is to illustrate the first part of TASK 2 of the data engineering test task.

Three datasets in CSV format was provided.

These datasets contain information about customers, policies, and claims respectively.

I use Python and Pandas to extract data from these datasets and load them into a PostgreSQL database of your choice. Then after loadin g the datasets into postgreSQL server and the data analysis from there on is done using SQL query.

In [1]:
```
# Import all neccesary python modules

import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import psycopg2
from sqlalchemy import create_engine
print("Setup Complete")
```

```
/Users/shade/miniconda3/envs/ML/lib/python3.9/site-packages/scipy/__init
__.py:146: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required
for this version of SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversio
n}"
Setup Complete
```

In [2]:
```
claim_data = pd.read_csv('claims.csv')
customer_data = pd.read_csv('customers.csv')
policy_data = pd.read_csv('policies.csv')
```

In [3]:
```
customer_data
```

Out[3]:

	customer_id	customer_name	customer_email	customer_address	ag
0	1	Kevin Rodriguez	jonathonorr@example.net	56236 Danielle Center\nHuynhbury, SC 69099	3
1	2	Daniel Wong	nperez@example.org	99009 Kayla Orchard\nEast Andrea, WV 27256	3
2	3	Joanna Elliott	osbornemichael@example.org	1455 Garrett Mall Apt. 331\nBlevinstown, RI 53079	6
3	4	Michael Harmon	michael43@example.org	184 Laurie Lodge Apt. 577\nWest Jennifer, KY 7...	5
4	5	Vernon Stein	rhart@example.org	00681 Margaret Mission Suite 354\nPort Daniels...	3
...	...	...	...	...	
9995	9996	Stephanie Rodriguez	irobinson@example.net	Unit 6703 Box 8958\nDPO AA 13099	3
9996	9997	Joseph Gonzalez	vmcintosh@example.com	45396 Bill Isle Apt. 708\nAlexandertown, MT 27041	4
9997	9998	Samuel Bray	larrymartinez@example.org	4356 Jennifer Meadow Suite 242\nKennethland, N...	5
9998	9999	Mr. Michael Davis	kdaugherty@example.org	70292 Lindsey Avenue\nEast Tinaton, WA 85808	3
9999	10000	Tammy Williams	joseph78@example.net	46810 Thompson Road\nLake Timothy, FL 97993	3

10000 rows × 7 columns

In [4]:
```
claim_data
```

Out[4]:

	claim_id	claim_date	claim_amount	policy_id
0	1	2022-03-24	9821	8094
1	2	2022-09-08	7627	6511
2	3	2022-10-31	9541	4644
3	4	2023-01-01	1256	2772
4	5	2021-07-08	3948	8357
...	...	...	...	...
4995	4996	2022-03-05	4026	8804
4996	4997	2022-04-03	1955	6070
4997	4998	2021-10-14	2196	7717
4998	4999	2022-04-24	6295	7284
4999	5000	2021-08-06	4754	3447

5000 rows × 4 columns

In [5]:
```
policy_data
```

Out[5]:

	policy_id	policy_type	policy_start_date	policy_end_date	premium_amou
0	1	HEALTH_INSURANCE	2023-11-01	2024-10-28	328
1	2	HEALTH_INSURANCE	2022-04-22	2025-11-25	17
2	3	AUTO_INSURANCE	2022-07-01	2025-11-21	34
3	4	AUTO_INSURANCE	2022-08-04	2025-11-25	448
4	5	AUTO_INSURANCE	2023-12-05	2025-11-30	50
...	...	...	...	...	
8995	8996	AUTO_INSURANCE	2023-11-26	2024-11-18	39
8996	8997	AUTO_INSURANCE	2022-09-26	2024-08-08	21
8997	8998	HEALTH_INSURANCE	2023-01-18	2025-01-13	454
8998	8999	AUTO_INSURANCE	2022-05-07	2025-08-06	47
8999	9000	HEALTH_INSURANCE	2024-01-14	2024-05-31	40

9000 rows × 6 columns

In [6]:
```
claim_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   claim_id      5000 non-null   int64
 1   claim_date    5000 non-null   object
 2   claim_amount  5000 non-null   int64
 3   policy_id     5000 non-null   int64
dtypes: int64(3), object(1)
memory usage: 156.4+ KB
```

In [7]:
```
customer_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 7 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   customer_id       10000 non-null  int64
 1   customer_name     10000 non-null  object
 2   customer_email    10000 non-null  object
 3   customer_address  10000 non-null  object
 4   age               10000 non-null  int64
 5   gender            10000 non-null  object
 6   occupation        10000 non-null  object
dtypes: int64(2), object(5)
memory usage: 547.0+ KB
```

In [8]:
```
policy_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9000 entries, 0 to 8999
Data columns (total 6 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   policy_id         9000 non-null   int64
 1   policy_type       9000 non-null   object
 2   policy_start_date  9000 non-null   object
 3   policy_end_date   9000 non-null   object
 4   premium_amount    9000 non-null   int64
 5   customer_id       9000 non-null   int64
dtypes: int64(3), object(3)
memory usage: 422.0+ KB
```

In [ ]:
```

```

In [9]:
```
# Establish connection session between this python client program and the

pgconn = psycopg2.connect(host = 'localhost',
                          user = 'postgres',
                          password = 'Xingjin112',
                          database = 'postgres',
                          port = '5433',
                         )
```

In [10]:
```
# cursor
pgcursor = pgconn.cursor()
data_path = '/Users/shade/Desktop/Nash/Tech_stuffs/Cachet_Data_Engineer_H
```

In [11]:
```
# required code to temper errors
from psycopg2.extensions import ISOLATION_LEVEL_AUTOCOMMIT
pgconn.set_isolation_level(ISOLATION_LEVEL_AUTOCOMMIT)
```

pgcursor.execute('DROP DATABASE IF EXISTS policy_table') pgcursor.execute('DROP DATABASE IF EXISTS claims_table') pgcursor.execute('DROP DATABASE IF EXISTS customer_table')

In [12]:
```
# create POLICY data  table

pgcursor.execute("""
CREATE TABLE IF NOT EXISTS policy_table
(
    policy_id              SERIAL
  , policy_type            VARCHAR(50) NOT NULL
  , policy_start_date      DATE
  , policy_end_date        DATE
  , premium_amount         MONEY
  , customer_id            BIGINT
);

""")
```

In [ ]:
```

```

In [13]:
```
# create customer data  table

pgcursor.execute("""
CREATE TABLE IF NOT EXISTS customer_table
(
    customer_id             SERIAL
  , customer_name           VARCHAR(500)
  , customer_email          text not null unique
  , customer_address        VARCHAR
  , age                     INT
  , gender                  VARCHAR
  , occupation              VARCHAR
   , address       VARCHAR
);

""")
```

In [14]:
```
# create customer data  table

pgcursor.execute("""
CREATE TABLE IF NOT EXISTS claim_table
(
    claim_id             SERIAL
  , claim_date           DATE
  , claim_amount         MONEY
  , policy_id            BIGINT
);

""")
```

In [15]:
```
# Sending the dataframe into the postgreSQL database

claim_data.to_sql(name='claims_table', con=pgconn, if_exists='replace')

claim_data.to_sql(name='claims_table', con=pgconn, if_exists='replace')
```

```
claim_data.to_sql(name='claims_table', con=pgconn, if_exists='replace')

print(' data succesfully pushed to DataBase')
```

```
 data succesfully pushed to DataBase
```

In [16]:
```
pgconn.commit()
pgconn.close()

print('connectiion closed successfully')
```

```
connectiion closed successfully
```

In [ ]:

In [ ]: