In [ ]:
```
import pandas as pd
import numpy as np
import psycopg2, kaggle, sqlalchemy
import matplotlib.pyplot as plt

from kaggle.api.kaggle_api_extended import KaggleApi
```

In [ ]:
```
# Extract
api = KaggleApi()
api.authenticate()
```

In [ ]:
```
# Download

# Competition dataset
kaggle.KaggleApi.competition_download_file(self=kaggle.api, competition='titanic', force=F

# Standalone dataset
api.dataset_download_file('pruthvinavada/popular-cocktails', file_name='popular_cocktails.
api.dataset_download_file('willianoliveiragibin/data-jobs-salaries', file_name='salaries.c
```

```
train.csv: Skipping, found more recently modified local copy (use --force to force downloa
d)
```

Out[ ]:
```
False
```

In [ ]:
```
dbname = "assingmentfour"
user = "postgres"
password = "Kakkahata666"
host = "localhost"

# Establish a connection to the database
conn = psycopg2.connect(
    dbname=dbname,
    user=user,
    password=password,
    host=host
)
cursor = conn.cursor()
conn.autocommit = True
sql = '''CREATE DATABASE assingmentfour;'''
#cur.execute(sql)
conn.commit()
```

In [ ]:
```
# Transform
df1 = pd.DataFrame(pd.read_csv('popular_cocktails.csv'))
df2 = pd.DataFrame(pd.read_csv('train.csv'))
df3 = pd.DataFrame(pd.read_csv('salaries.csv'))
```

In [ ]:
```
# Drop the existing "cocktails" table if it exists
drop_table_query = "DROP TABLE IF EXISTS cocktails"
cursor.execute(drop_table_query)
conn.commit()

# Drop the existing "salary" table if it exists
drop_table_query = "DROP TABLE IF EXISTS salary"
cursor.execute(drop_table_query)
```

```
# Drop the existing "passenger" table if it exists
drop_table_query = "DROP TABLE IF EXISTS passenger"
cursor.execute(drop_table_query)
conn.commit()
```

In [ ]:

```
# Define the table structure for cocktails table
create_table_query = """
CREATE TABLE cocktails (
    idDrink INT PRIMARY KEY,
    strDrink TEXT,
    strDrinkAlternate TEXT,
    strTags TEXT,
    strVideo TEXT,
    strCategory TEXT,
    strIBA TEXT,
    strAlcoholic TEXT,
    strGlass TEXT,
    strInstructions TEXT,
    strInstructionsES TEXT,
    strInstructionsDE TEXT,
    strInstructionsFR TEXT,
    strInstructionsIT TEXT,
    strInstructionsZH_HANS TEXT,
    strInstructionsZH_HANT TEXT,
    strDrinkThumb TEXT,
    strIngredient1 TEXT,
    strIngredient2 TEXT,
    strIngredient3 TEXT,
    strIngredient4 TEXT,
    strIngredient5 TEXT,
    strIngredient6 TEXT,
    strIngredient7 TEXT,
    strIngredient8 TEXT,
    strIngredient9 TEXT,
    strIngredient10 TEXT,
    strIngredient11 TEXT,
    strIngredient12 TEXT,
    strIngredient13 TEXT,
    strIngredient14 TEXT,
    strIngredient15 TEXT,
    strMeasure1 TEXT,
    strMeasure2 TEXT,
    strMeasure3 TEXT,
    strMeasure4 TEXT,
    strMeasure5 TEXT,
    strMeasure6 TEXT,
    strMeasure7 TEXT,
    strMeasure8 TEXT,
    strMeasure9 TEXT,
    strMeasure10 TEXT,
    strMeasure11 TEXT,
    strMeasure12 TEXT,
    strMeasure13 TEXT,
    strMeasure14 TEXT,
    strMeasure15 TEXT,
    strImageSource TEXT,
    strImageAttribution TEXT,
    strCreativeCommonsConfirmed TEXT,
    dateModified TIMESTAMP
);
"""
```



```
table structure for the salary table
```

```
create_employee_salary_table_query = """
CREATE TABLE salary (
    work_year INT,
    experience_level TEXT,
    employment_type TEXT,
    job_title TEXT,
    salary NUMERIC,
    salary_currency TEXT,
    salary_in_usd NUMERIC,
    employee_residence TEXT,
    remote_ratio INT,
    company_location TEXT,
    company_size TEXT
);
"""

# Define the table structure for the passenger table
create_titanic_passenger_table_query = """
CREATE TABLE passenger (
    PassengerId INT PRIMARY KEY,
    Survived INT,
    Pclass INT,
    Name TEXT,
    Sex TEXT,
    Age NUMERIC,
    SibSp INT,
    Parch INT,
    Ticket TEXT,
    Fare NUMERIC,
    Cabin TEXT,
    Embarked TEXT
);
"""
```

In [ ]:

```
# Create the cocktails table
cursor.execute(create_table_query)
conn.commit()

# Create the salary table
cursor.execute(create_employee_salary_table_query)
conn.commit()

# Create the passenger table
cursor.execute(create_titanic_passenger_table_query)
conn.commit()
```

In [ ]:

```
# Insert data from DataFrames into the corresponding tables
for _, row in df1.iterrows():
    cursor.execute("INSERT INTO cocktails VALUES (%s)" % ', '.join(['%s'] * len(row)), tup
conn.commit()

for _, row in df2.iterrows():
    cursor.execute("INSERT INTO passenger VALUES (%s)" % ', '.join(['%s'] * len(row)), tup
conn.commit()

for _, row in df3.iterrows():
    cursor.execute("INSERT INTO salary VALUES (%s)" % ', '.join(['%s'] * len(row)), tuple(
conn.commit()

# Close the database connection
conn.close()
```

```
---------------------------------------------------------------------------
SyntaxError                               Traceback (most recent call last)
c:\Users\milli\DEAI\DE\KaggleTrainin.ipynb Cell 9 line 3
      <a href='vscode-notebook-cell:/c%3A/Users/milli/DEAI/DE/KaggleTrainin.ipynb#W6sZmlsZQ%3D%3D?line=0'>1</a> # Insert data from DataFrames into the corresponding tables
      <a href='vscode-notebook-cell:/c%3A/Users/milli/DEAI/DE/KaggleTrainin.ipynb#W6sZmlsZQ%3D%3D?line=1'>2</a> for _, row in df1.iterrows():
----> <a href='vscode-notebook-cell:/c%3A/Users/milli/DEAI/DE/KaggleTrainin.ipynb#W6sZmlsZQ%3D%3D?line=2'>3</a>     cursor.execute("INSERT INTO cocktails VALUES (%s)" % ', '.join(['%s'] * len(row)), tuple(row))
      <a href='vscode-notebook-cell:/c%3A/Users/milli/DEAI/DE/KaggleTrainin.ipynb#W6sZmlsZQ%3D%3D?line=3'>4</a> conn.commit()
      <a href='vscode-notebook-cell:/c%3A/Users/milli/DEAI/DE/KaggleTrainin.ipynb#W6sZmlsZQ%3D%3D?line=5'>6</a> for _, row in df2.iterrows():

SyntaxError: INSERT has more expressions than target columns
LINE 4: ...ps://pixabay.com/users/anilaha-16242978/', 'Yes', '2016-11-0...
                                                             ^
```