Fetching Data From API and Creating DataFrame and Filtering Data IF Required

```
url = "https://api.themoviedb.org/3/movie/top_rated?language=en-
US&page=1"

headers = {
    "accept": "application/json",
    "Authorization": "Bearer
eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOiI4ZTE1OWRjODYwNWM2ZDk1NjMwMGI5ZjE0OGU0M
zlkMyIsInN1YiI6IjY0ZWQ5ODFkYzNjODkxMDBhZWRhZjA3MSIsInNjb3BlcyI6WyJhcGl
fcmVhZCJdLCJ2ZXJzaW9uIjoxfQ.ZQF5qbx6_klFgt9cceM003ldCCqS2q3MlIoUx6zGo9
I"
}

response = requests.get(url,headers= headers)
#print(response.text)


import pandas as pd

response.json()['results']

data = pd.DataFrame(response.json()['results'])

# data.head()

FilteredData =
data[['id','title','release_date','vote_average','vote_count']]

#FilteredData.head()

# Create an empty DataFrame to store the data
df = pd.DataFrame(columns=['id', 'title', 'release_date',
'vote_average', 'vote_count'])

# Your API request URL and headers
base_url = "https://api.themoviedb.org/3/movie/top_rated"
headers = {
    "accept": "application/json",
    "Authorization": "Bearer
eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOiI4ZTE1OWRjODYwNWM2ZDk1NjMwMGI5ZjE0OGU0M
zlkMyIsInN1YiI6IjY0ZWQ5ODFkYzNjODkxMDBhZWRhZjA3MSIsInNjb3BlcyI6WyJhcGl
fcmVhZCJdLCJ2ZXJzaW9uIjoxfQ.ZQF5qbx6_klFgt9cceM003ldCCqS2q3MlIoUx6zGo9
I"
}
```

```
# Loop through the desired range of pages
for i in range(1, 429):
    url = f"{base_url}?language=en-US&page={i}"

    # Make the API request
    response = requests.get(url, headers=headers)

    # Check if the request was successful
    if response.status_code == 200:
        data = response.json()
        results = data.get('results', [])

        if results:
            # Extract and append relevant data to the DataFrame
            filtered_data = pd.DataFrame(results, columns=['id', 
'title', 'release_date', 'vote_average', 'vote_count'])
            df = pd.concat([df, filtered_data], ignore_index=True)
    else:
        print(f"Failed to fetch data from page {i}")

# Now, 'df' contains all the data from the API

df.head()
```

```
      id                        title release_date  vote_average  
vote_count
0    238                The Godfather   1972-03-14           8.7  
18540
1    278     The Shawshank Redemption   1994-09-23           8.7  
24507
2    240        The Godfather Part II   1974-12-20           8.6  
11199
3    424             Schindler's List   1993-12-15           8.6  
14481
4  19404  Dilwale Dulhania Le Jayenge   1995-10-20           8.6  
4235
```

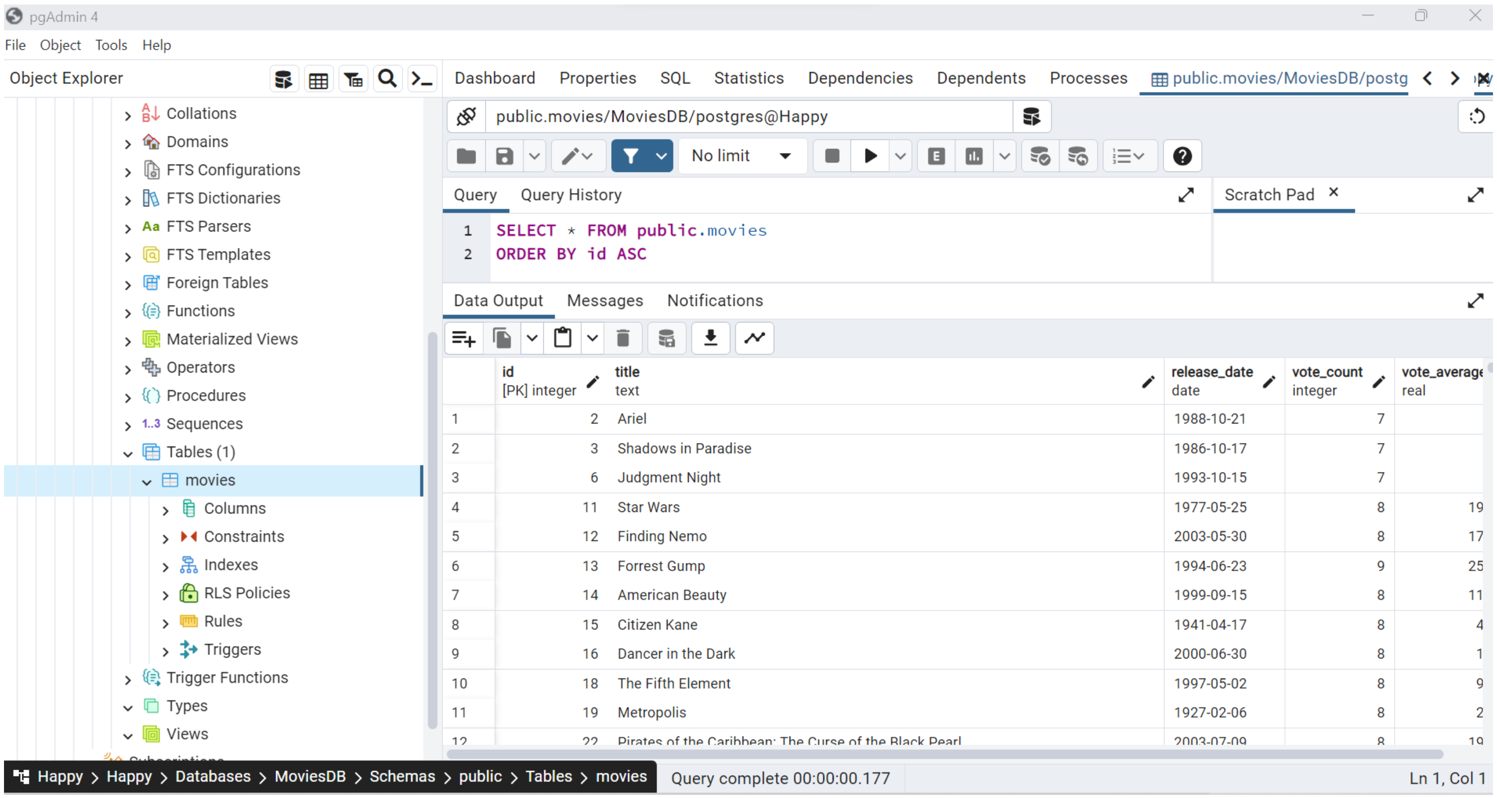
Connecting python with postgresql and Creating Data Base Objects to Dump Data from dataframe

```
#!pip install psycopg2   Install if not Available


import psycopg2
```

```
# Database parameters  of a sample DATABASE MADE ON LOCAL HOST

db_params = {
    'host': 'localhost',
    'database': 'MoviesDB',  # Use the name of your database
    'user': 'postgres',
    'password': 'Happy123@'
}

try:
    # Establish a connection to the database
    connection = psycopg2.connect(**db_params)
    print("Successfully connected to PostgreSQL.")

    # Create a cursor object to execute SQL commands
    cursor = connection.cursor()

    # Define the CREATE TABLE SQL statement using triple quotes
    create_table_sql = '''
    CREATE TABLE Movies (
        ID SERIAL PRIMARY KEY,
        TITLE TEXT,
        RELEASE_DATE DATE,
        VOTE_COUNT INTEGER,
        VOTE_AVERAGE REAL

    )
    '''

    # Execute the CREATE TABLE statement
    cursor.execute(create_table_sql)

      # Commit the transaction to save changes
    connection.commit()

    print("Table 'Movies' created successfully.")



    # Convert the DataFrame to a list of tuples   for  inserting data
from dataframe to this Table
    data_to_insert = [tuple(row) for row in
df.to_records(index=False)]

    # Define the INSERT INTO SQL statement
    insert_sql = '''
    INSERT INTO Movies (ID, TITLE, RELEASE_DATE, VOTE_COUNT,
VOTE_AVERAGE)
    VALUES (%s, %s, %s, %s, %s)
    '''

    # Execute the INSERT statement for each row of data
    cursor.executemany(insert_sql, data_to_insert)

    # Commit the transaction to save changes
    connection.commit()

    print("Data inserted into 'Movies' table successfully.")

    # Commit the transaction to save changes
    connection.commit()

    print("Table 'Movies' created successfully.")

except Exception as e:
    print("Error:", e)
finally:
    # Close the connection
    if connection:
        connection.close()
```

```
Successfully connected to PostgreSQL.
Table 'Movies' created successfully.
Data inserted into 'Movies' table successfully.
Table 'Movies' created successfully.
```

```
import psycopg2

# Database parameters
db_params = {
    'host': 'localhost',
    'database': 'MoviesDB',  # Use the name of your database
    'user': 'postgres',
    'password': 'Happy123@'
}

try:
    # Establish a connection to the database
    connection = psycopg2.connect(**db_params)
    print("Successfully connected to PostgreSQL.")

    # Create a cursor object to execute SQL commands
    cursor = connection.cursor()

    # Execute your SQL query
    cursor.execute('SELECT * FROM Movies LIMIT 5')  # Use "LIMIT"
instead of "TOP" in PostgreSQL

    # Fetch and print the results (if needed)
    results = cursor.fetchall()
    for row in results:
        print(row)

except Exception as e:
    print("Error:", e)
finally:
    # Close the cursor and the connection
    if cursor:
        cursor.close()
    if connection:
        connection.close()

Successfully connected to PostgreSQL.
(238, 'The Godfather', datetime.date(1972, 3, 14), 9, 18540.0)
(278, 'The Shawshank Redemption', datetime.date(1994, 9, 23), 9,
24507.0)
(240, 'The Godfather Part II', datetime.date(1974, 12, 20), 9,
11199.0)
(424, "Schindler's List", datetime.date(1993, 12, 15), 9, 14481.0)
(19404, 'Dilwale Dulhania Le Jayenge', datetime.date(1995, 10, 20), 9,
4235.0)
```



```
SELECT * FROM public.movies
ORDER BY id ASC
```

	id [PK] integer	title text	release_date date	vote_count integer	vote_average real
1	2	Ariel	1988-10-21	7	
2	3	Shadows in Paradise	1986-10-17	7	
3	6	Judgment Night	1993-10-15	7	
4	11	Star Wars	1977-05-25	8	19
5	12	Finding Nemo	2003-05-30	8	17
6	13	Forrest Gump	1994-06-23	9	25
7	14	American Beauty	1999-09-15	8	11
8	15	Citizen Kane	1941-04-17	8	4
9	16	Dancer in the Dark	2000-06-30	8	1
10	18	The Fifth Element	1997-05-02	8	9
11	19	Metropolis	1927-02-06	8	2
12	22	Pirates of the Caribbean: The Curse of the Black Pearl	2003-07-09	8	19