case-studies-using-sql

September 1, 2024

1 Author : Maruf khan

- linkedin : https://www.linkedin.com/in/maruf-khan-1516a4224/
- Github : https://github.com/Maruf-Ahmad-khan?tab=repositories

2 General SQL Queries:

3 you can find the dataset on github for practice.

- 1. Write a query to find the total number of users in the dataset.
- 2. How would you retrieve all unique states present in the dataset?
- 3. Write a SQL query to count how many users are in each state.
- 4. How can you find the number of users in a specific city, for example, Ahmedabad?
- 5. Write a query to find the number of users in each city within a specific state, such as Maharashtra.
- 6. How would you retrieve all users whose names start with the letter 'B'?
- 7. Write a query to list users who live in cities that start with the letter 'J'.
- 8. How can you find the user with the highest user_id?
- 9. Write a query to return users from the state 'Gujarat' and the city 'Ahmedabad'.
- 10. How would you get the count of users grouped by state and then by city?

4 Establish the connect with the database

[1]:
```
import mysql.connector as connection

try:
    mydb = connection.connect(host="localhost", user="root", passwd="",␣
 ↪database="demo", use_pure=True)

    if mydb.is_connected():
        print("Successfully connected to the demo database.")
except Exception as e:
```

```
    print("Error:", str(e))
```

```
Successfully connected to the demo database.
```

5 Read the table

[2]:
```
import pandas as pd
try:
    df = pd.read_sql("SELECT * FROM users;", mydb)
    print(df)
except Exception as e:
    print("Error:", str(e))
```

```
     user_id      name              state       city
0          1    Bharat            Gujarat  Ahmedabad
1          2     Pearl        Maharashtra       Pune
2          3     Jahan     Madhya Pradesh     Bhopal
3          4    Divsha          Rajasthan     Jaipur
4          5   Kasheen        West Bengal    Kolkata
..       …       …                  …         …
396      407   Shubham  Jammu and Kashmir    Kashmir
397      408   Kalyani         Tamil Nadu    Chennai
398      409     Komal      Uttar Pradesh    Lucknow
399      410  Kartikay              Bihar      Patna
400      464    Ankita        Maharashtra     Mumbai

[401 rows x 4 columns]
c:\Users\mk744\anaconda3\lib\site-packages\pandas\io\sql.py:761: UserWarning:
pandas only support SQLAlchemy connectable(engine/connection) ordatabase string
URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please
consider using SQLAlchemy
  warnings.warn(
```

6 Read the table

[3]:
```
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
df = pd.read_sql("SELECT * FROM users;", mydb)
df
```

[3]:
```
     user_id      name              state       city
0          1    Bharat            Gujarat  Ahmedabad
1          2     Pearl        Maharashtra       Pune
2          3     Jahan     Madhya Pradesh     Bhopal
3          4    Divsha          Rajasthan     Jaipur
```

```
4          5   Kasheen        West Bengal    Kolkata
..       …       …                 …         …
396      407   Shubham  Jammu and Kashmir    Kashmir
397      408   Kalyani         Tamil Nadu    Chennai
398      409     Komal      Uttar Pradesh    Lucknow
399      410  Kartikay              Bihar      Patna
400      464    Ankita        Maharashtra     Mumbai

[401 rows x 4 columns]
```

7 Write a query to find the total number of users in the dataset

[4]:
```
df_count = pd.read_sql("SELECT COUNT(*) AS Total_number_of_users FROM users;",␣
 ↪mydb)
df_count
```

[4]:
```
   Total_number_of_users
0                    401
```

8 How would you retrieve all unique states present in the dataset?

[5]:
```
df_unique_state = pd.read_sql("SELECT DISTINCT state AS Unique_state FROM␣
 ↪users", mydb)
df_unique_state
```

[5]:
```
         Unique_state
0             Gujarat
1          Maharashtra
2        Madhya Pradesh
3            Rajasthan
4          West Bengal
5            Karnataka
6     Jammu and Kashmir
7           Tamil Nadu
8         Uttar Pradesh
9                 Bihar
10              Kerala
11              Punjab
12             Haryana
13    Himachal Pradesh
14              Sikkim
15                 Goa
16            Nagaland
17       Andhra Pradesh
18               Delhi
```

9 Write a SQL query to count how many users are in each state.

[6]:
```
user_count = pd.read_sql("SELECT state, COUNT(*) AS user_count FROM users GROUP␣
 ↪BY city ORDER BY user_count ASC", mydb)
user_count
```

[6]:

	state	user_count
0	Tamil Nadu	6
1	Jammu and Kashmir	8
2	Sikkim	9
3	Uttar Pradesh	9
4	Punjab	9
5	Goa	10
6	Gujarat	10
7	Uttar Pradesh	11
8	Kerala	11
9	Himachal Pradesh	11
10	Nagaland	11
11	Bihar	12
12	Rajasthan	12
13	Gujarat	13
14	Rajasthan	13
15	Andhra Pradesh	13
16	Karnataka	15
17	Maharashtra	16
18	Madhya Pradesh	16
19	West Bengal	16
20	Punjab	22
21	Delhi	24
22	Maharashtra	61
23	Madhya Pradesh	63

10 How can you find the number of users in a specific city, for example, Ahmedabad?

[7]:
```
User_count_Ahmedabad = pd.read_sql("SELECT COUNT(*) AS user_count FROM users␣
 ↪WHERE city = 'Ahmedabad';", mydb)
User_count_Ahmedabad
```

[7]:

	user_count
0	13

11 Write a query to find the number of users in each city within a specific state, such as Maharashtra.

[8]:
```
No_of_users_Maharashtra = pd.read_sql("SELECT COUNT(*) user_count FROM users␣
 ↪WHERE state = 'Maharashtra';", mydb)
No_of_users_Maharashtra
```

[8]:

	user_count
0	77

12 How would you retrieve all users whose names start with the letter 'B'?

[9]:
```
users_Name_Startwith_B = pd.read_sql("SELECT * FROM users WHERE name LIKE␣
 ↪'b%'", mydb)
users_Name_Startwith_B
```

[9]:

	user_id	name	state	city
0	1	Bharat	Gujarat	Ahmedabad
1	15	Bhavna	Sikkim	Gangtok
2	26	Bhishm	Maharashtra	Mumbai
3	78	Bathina	Karnataka	Bangalore
4	81	Bhawna	Madhya Pradesh	Indore
5	89	Bhaggyasree	Maharashtra	Mumbai
6	276	Bhosale	Punjab	Amritsar
7	286	Brijesh	Rajasthan	Udaipur
8	305	Bhargav	Madhya Pradesh	Delhi
9	332	Bhutekar	Madhya Pradesh	Indore

13 Write a query to list users who live in cities that start with the letter 'J'.

[10]:
```
user_city_startwith_J = pd.read_sql("SELECT * FROM users WHERE city LIKE 'j%'",␣
 ↪mydb)
user_city_startwith_J
```

[10]:

	user_id	name	state	city
0	4	Divsha	Rajasthan	Jaipur
1	22	Monisha	Rajasthan	Jaipur
2	40	Paridhi	Rajasthan	Jaipur
3	58	Shefali	Rajasthan	Jaipur
4	76	Chandni	Rajasthan	Jaipur
5	94	Subhasmita	Rajasthan	Jaipur
6	112	Adhvaita	Rajasthan	Jaipur

```
7       130     Rishabh  Rajasthan  Jaipur
8       148      Nitant  Rajasthan  Jaipur
9       166     Surabhi  Rajasthan  Jaipur
10      184       Rohit  Rajasthan  Jaipur
11      220     Dheeraj  Rajasthan  Jaipur
12      404     Nandita  Rajasthan  Jaipur
```

14 How can you find the user with the highest user_id?

[11]:
```
# Highest_user_id = pd.read_sql("SELECT MAX(user_id) AS Highest_user_id FROM␣
↪users", mydb)
Highest_user_id = pd.read_sql("SELECT * FROM users ORDER BY user_id LIMIT 1",␣
↪mydb)
Highest_user_id
```

[11]:
```
   user_id    name    state       city
0        1  Bharat  Gujarat  Ahmedabad
```

15 Write a query to return users from the state 'Gujarat' and the city 'Ahmedabad'.

[12]:
```
Gujarat_Ahmedabad = pd.read_sql("SELECT * FROM users WHERE state = 'Gujarat'␣
↪AND city = 'Ahmedabad'", mydb)
Gujarat_Ahmedabad
```

[12]:
```
    user_id      name    state       city
0         1    Bharat  Gujarat  Ahmedabad
1        19    Ramesh  Gujarat  Ahmedabad
2        73   Arsheen  Gujarat  Ahmedabad
3       145    Kartik  Gujarat  Ahmedabad
4       163   Noshiba  Gujarat  Ahmedabad
5       181    Rutuja  Gujarat  Ahmedabad
6       199  Divyansh  Gujarat  Ahmedabad
7       235   Moumita  Gujarat  Ahmedabad
8       253    Gaurav  Gujarat  Ahmedabad
9       343   Shardul  Gujarat  Ahmedabad
10      347    Chetan  Gujarat  Ahmedabad
11      354    Trupti  Gujarat  Ahmedabad
12      361    Surbhi  Gujarat  Ahmedabad
```

16 How would you get the count of users grouped by state and then by city?

[13]:
```
users_state_city = pd.read_sql("SELECT state, city, COUNT(*) AS user_count FROM␣
↪users GROUP BY state ,city ORDER BY state, city DESC;", mydb)
users_state_city
```

[13]:

	state	city	user_count
0	Andhra Pradesh	Hyderabad	13
1	Bihar	Patna	12
2	Delhi	Delhi	21
3	Goa	Goa	10
4	Gujarat	Surat	10
5	Gujarat	Ahmedabad	13
6	Haryana	Chandigarh	10
7	Himachal Pradesh	Simla	11
8	Jammu and Kashmir	Kashmir	8
9	Karnataka	Bangalore	15
10	Kerala	Thiruvananthapuram	11
11	Madhya Pradesh	Indore	63
12	Madhya Pradesh	Delhi	3
13	Madhya Pradesh	Bhopal	16
14	Maharashtra	Pune	16
15	Maharashtra	Mumbai	61
16	Nagaland	Kohima	11
17	Punjab	Chandigarh	12
18	Punjab	Amritsar	9
19	Rajasthan	Udaipur	12
20	Rajasthan	Jaipur	13
21	Sikkim	Gangtok	9
22	Tamil Nadu	Chennai	6
23	Uttar Pradesh	Lucknow	11
24	Uttar Pradesh	Allahabad	9
25	West Bengal	Kolkata	16

17 Data Aggregation and Grouping:

- 1. Write a query to find the average user_id for users in each state.
- 2. How would you find the city with the maximum number of users?
- 3. Write a query to find the state with the minimum number of cities represented.
- 4. How would you group users by the first letter of their city name?
- 5. Write a query to find the top 5 cities with the most users.

18 Write a query to find the average user_id for users in each state.

[14]:
```
Average_user_id = pd.read_sql("SELECT user_id, state, AVG(user_id) AS␣
 ↪Avg_user_id FROM users GROUP BY state ORDER BY Avg_user_id;", mydb)
Average_user_id
```

[14]:

	user_id	state	Avg_user_id
0	7	Jammu and Kashmir	117.7500
1	13	Haryana	140.8000
2	16	Goa	143.8000
3	11	Kerala	147.9091
4	6	Karnataka	153.4667
5	14	Himachal Pradesh	159.6364
6	10	Bihar	162.8333
7	17	Nagaland	168.6364
8	15	Sikkim	173.0000
9	5	West Bengal	177.1875
10	8	Tamil Nadu	179.6667
11	2	Maharashtra	196.3506
12	3	Madhya Pradesh	200.8537
13	12	Punjab	226.5714
14	4	Rajasthan	227.5600
15	18	Andhra Pradesh	229.6154
16	9	Uttar Pradesh	247.3500
17	1	Gujarat	251.0435
18	304	Delhi	368.2857

19 How would you find the city with the maximum number of users?

[15]:
```
Max_num_users = pd.read_sql("SELECT  city, COUNT(*) AS Maxm_num_users FROM␣
 ↪users GROUP BY city ORDER BY Maxm_num_users DESC LIMIT 1", mydb)
Max_num_users
```

[15]:

	city	Maxm_num_users
0	Indore	63

20 Write a query to find the state with the minimum number of cities represented.

[16]:
```
Min_city_count = pd.read_sql("SELECT state, COUNT(DISTINCT city) AS␣
 ↪Minm_num_city FROM users GROUP BY state ORDER BY Minm_num_city ASC LIMIT 1 ;
 ↪", mydb)
```

```
Min_city_count
```

[16]:

	state	Minm_num_city
0	Andhra Pradesh	1

21 How would you group users by the first letter of their city name?

[17]:
```
City_First_Letter = pd.read_sql("SELECT LEFT(city, 1) AS First_Letter, COUNT(*)␣
↪AS city_count FROM users GROUP BY LEFT(city, 1) ORDER BY city_count DESC;",␣
↪mydb)
City_First_Letter
```

[17]:

	First_Letter	city_count
0	I	63
1	M	61
2	K	35
3	B	31
4	A	31
5	C	28
6	P	28
7	D	24
8	S	21
9	G	19
10	J	13
11	H	13
12	U	12
13	L	11
14	T	11

22 Write a query to find the top 5 cities with the most users.

[18]:
```
Top_5_cities = pd.read_sql("SELECT city, COUNT(*) AS Top_five_cities FROM users␣
↪GROUP BY city ORDER BY Top_five_cities DESC LIMIT 5", mydb)
Top_5_cities
```

[18]:

	city	Top_five_cities
0	Indore	63
1	Mumbai	61
2	Delhi	24
3	Chandigarh	22
4	Pune	16

23 Subqueries and Nested Queries:

- 1. Write a query to find all users who live in the same city as the user with user_id 1.
- 2. How would you find states that have more users than the state 'Rajasthan'?
- 3. Write a query to retrieve users whose user_id is higher than the average user_id.
- 4. How would you find the second most populated city in the dataset?
- 5. Write a query to list users who live in cities that have fewer than 5 users.

24 Write a query to find all users who live in the same city as the user with user_id 1 .

[27]:
```
same_user_id = pd.read_sql(
    "SELECT user_id, name, state, city FROM users WHERE city = (SELECT city␣
 ↪FROM users WHERE user_id = 1)"
    , mydb)
same_user_id
```

[27]:

	user_id	name	state	city
0	1	Bharat	Gujarat	Ahmedabad
1	19	Ramesh	Gujarat	Ahmedabad
2	73	Arsheen	Gujarat	Ahmedabad
3	145	Kartik	Gujarat	Ahmedabad
4	163	Noshiba	Gujarat	Ahmedabad
5	181	Rutuja	Gujarat	Ahmedabad
6	199	Divyansh	Gujarat	Ahmedabad
7	235	Moumita	Gujarat	Ahmedabad
8	253	Gaurav	Gujarat	Ahmedabad
9	343	Shardul	Gujarat	Ahmedabad
10	347	Chetan	Gujarat	Ahmedabad
11	354	Trupti	Gujarat	Ahmedabad
12	361	Surbhi	Gujarat	Ahmedabad

25 How would you find states that have more users than the state 'Rajasthan'?

[31]:
```
More_users = pd.read_sql("SELECT state, COUNT(*) AS user_count FROM users GROUP␣
 ↪BY state HAVING COUNT(*) > (SELECT COUNT(*) FROM users WHERE state =␣
 ↪'Rajasthan') ORDER BY user_count DESC", mydb)
More_users
```

[31]:

	state	user_count
0	Madhya Pradesh	82
1	Maharashtra	77

26 Write a query to retrieve users whose user_id is higher than the average user_id.

[34]:
```
Avg_user_id = pd.read_sql("SELECT user_id, name, state, city FROM users WHERE␣
 ↪user_id > (SELECT AVG(user_id) FROM users)", mydb)
Avg_user_id
```

[34]:

	user_id	name	state	city
0	206	Dhanraj	Madhya Pradesh	Indore
1	207	Vipul	Uttar Pradesh	Lucknow
2	208	Apsingekar	Bihar	Patna
3	209	Suman	Kerala	Thiruvananthapuram
4	210	Nripraj	Punjab	Chandigarh
..	…	…	…	…
195	407	Shubham	Jammu and Kashmir	Kashmir
196	408	Kalyani	Tamil Nadu	Chennai
197	409	Komal	Uttar Pradesh	Lucknow
198	410	Kartikay	Bihar	Patna
199	464	Ankita	Maharashtra	Mumbai

[200 rows x 4 columns]

27 How would you find the second most populated city in the dataset?

[41]:
```
Second_populated_city = pd.read_sql("SELECT city, COUNT(*) user_city FROM users␣
 ↪GROUP BY city ORDER BY user_city DESC LIMIT 1 OFFSET 1", mydb)
Second_populated_city
```

[41]:

	city	user_city
0	Mumbai	61

28 Write a query to list users who live in cities that have fewer than 5 users.

[42]:
```
Less_user = pd.read_sql("SELECT user_id, name, state, city FROM users WHERE␣
 ↪city IN (SELECT city FROM users GROUP BY city HAVING COUNT(*) < 5)", mydb)
Less_user
```

[42]:
```
Empty DataFrame
Columns: [user_id, name, state, city]
Index: []
```

[43]:
```
import subprocess

# Specify the path to your notebook
notebook_path = "Case Studies using sql.ipynb"

# Convert the notebook to PDF using nbconvert
subprocess.run(["jupyter", "nbconvert", "--to", "pdf", notebook_path])
```

[43]:
```
CompletedProcess(args=['jupyter', 'nbconvert', '--to', 'pdf', 'Case Studies
using sql.ipynb'], returncode=1)
```

[ ]:

[ ]: