Day26_Data Analysis_Python_vs_SQL_Comparison

June 23, 2025

Introduction

This notebook is a practical guide to **Data Analysis using Python and SQL**, based on real-world queries performed on a dataset containing travel and customer behavior data.

It demonstrates how to:

- Load and explore data using **pandas** in Python
- Perform SQL-like operations such as `SELECT`, `WHERE`, `GROUP BY`, `ORDER BY`, and `JOIN` using Python code
- Compare each Python operation with its **equivalent SQL query**
- Apply core data analysis techniques like filtering, grouping, aggregating, and joining data

Whether you're a beginner learning pandas or someone transitioning from SQL to Python (or vice versa), this notebook will help you understand how the two languages mirror each other for data analysis tasks.

Most Common `groupby()` Functions in Pandas

Function	Description	Example Code
`mean()`	Average of the values	`df.groupby('col')['val'].mean()`
`sum()`	Sum of the values	`df.groupby('col')['val'].sum()`
`count()`	Count of non-null values	`df.groupby('col')['val'].count()`
`size()`	Count of total rows (includes NaNs)	`df.groupby('col').size()`
`min()`	Minimum value	`df.groupby('col')['val'].min()`
`max()`	Maximum value	`df.groupby('col')['val'].max()`
`median()`	Median (middle) value	`df.groupby('col')['val'].median()`
`std()`	Standard deviation	`df.groupby('col')['val'].std()`
`var()`	Variance	`df.groupby('col')['val'].var()`
`nunique()`	Number of unique values	`df.groupby('col')['val'].nunique()`
`unique()`	List/array of unique values	`df.groupby('col')['val'].unique()`
`first()`	First non-null value in group	`df.groupby('col')['val'].first()`
`last()`	Last non-null value in group	`df.groupby('col')['val'].last()`
`describe()`	Summary stats (count, mean, std, min, max, etc.)	`df.groupby('col')['val'].describe()`
`apply(func)`	Apply a custom function	`df.groupby('col')['val'].apply(lambda x: x.max() - x.min())`

Function	Description	Example Code
agg()	Apply multiple functions at once	df.groupby('col')['val'].agg(['mean', 'max', 'min'])

[2]:
```
import pandas as pd
```

[3]:
```
# Load dataset
df = pd.read_csv(r'C:\Users\aksha\OneDrive\Desktop\SQL_DATA_EXPORT\dataset1.
↪csv')
```

SQL Equivalent:

```
SELECT * FROM dataset_1;
```

1 Display specific columns

SQL Equivalent:

```
SELECT weather, temperature FROM dataset_1;
```

Python:

[5]:
```
df[['weather', 'temperature']]
```

[5]:
```
      weather  temperature
0       Sunny           55
1       Sunny           80
2       Sunny           80
3       Sunny           80
4       Sunny           80
…         …          …
12679   Rainy           55
12680   Rainy           55
12681   Snowy           30
12682   Snowy           30
12683   Sunny           80

[12684 rows x 2 columns]
```

2 View first 10 rows

SQL Equivalent:

```
SELECT * FROM dataset_1 LIMIT 10;
```

Python:

[6]:
```
df.head(10)
```

```
[6]:        destination  passanger weather  temperature  time  \
     0  No Urgent Place      Alone   Sunny           55   2PM
     1  No Urgent Place  Friend(s)   Sunny           80  10AM
     2  No Urgent Place  Friend(s)   Sunny           80  10AM
     3  No Urgent Place  Friend(s)   Sunny           80   2PM
     4  No Urgent Place  Friend(s)   Sunny           80   2PM
     5  No Urgent Place  Friend(s)   Sunny           80   6PM
     6  No Urgent Place  Friend(s)   Sunny           55   2PM
     7  No Urgent Place     Kid(s)   Sunny           80  10AM
     8  No Urgent Place     Kid(s)   Sunny           80  10AM
     9  No Urgent Place     Kid(s)   Sunny           80  10AM

                       coupon expiration  gender age      maritalStatus  …  \
     0        Restaurant(<20)         1d  Female  21  Unmarried partner  …
     1           Coffee House         2h  Female  21  Unmarried partner  …
     2  Carry out & Take away         2h  Female  21  Unmarried partner  …
     3           Coffee House         2h  Female  21  Unmarried partner  …
     4           Coffee House         1d  Female  21  Unmarried partner  …
     5        Restaurant(<20)         2h  Female  21  Unmarried partner  …
     6  Carry out & Take away         1d  Female  21  Unmarried partner  …
     7        Restaurant(<20)         2h  Female  21  Unmarried partner  …
     8  Carry out & Take away         2h  Female  21  Unmarried partner  …
     9                    Bar         1d  Female  21  Unmarried partner  …

        CarryAway RestaurantLessThan20 Restaurant20To50 toCoupon_GEQ5min  \
     0       NaN                  4~8              1~3                1
     1       NaN                  4~8              1~3                1
     2       NaN                  4~8              1~3                1
     3       NaN                  4~8              1~3                1
     4       NaN                  4~8              1~3                1
     5       NaN                  4~8              1~3                1
     6       NaN                  4~8              1~3                1
     7       NaN                  4~8              1~3                1
     8       NaN                  4~8              1~3                1
     9       NaN                  4~8              1~3                1

       toCoupon_GEQ15min toCoupon_GEQ25min direction_same direction_opp  Y  \
     0                 0                 0              0             1  1
     1                 0                 0              0             1  0
     2                 1                 0              0             1  1
     3                 1                 0              0             1  0
     4                 1                 0              0             1  0
     5                 1                 0              0             1  1
     6                 1                 0              0             1  1
     7                 1                 0              0             1  1
     8                 1                 0              0             1  1
     9                 1                 0              0             1  0
```

```
  row_count
0         1
1         2
2         3
3         4
4         5
5         6
6         7
7         8
8         9
9        10

[10 rows x 27 columns]
```

3 Unique values in a column

SQL Equivalent:

```
SELECT DISTINCT passanger FROM dataset_1;
```

Python:

[7]:
```
df['passanger'].unique()
```

[7]:
```
array(['Alone', 'Friend(s)', 'Kid(s)', 'Partner'], dtype=object)
```

4 Filter rows based on condition

SQL Equivalent:

```
SELECT * FROM dataset_1 WHERE destination = 'Home';
```

Python:

[8]:
```
df[df['destination'] == 'Home']
```

[8]:
```
      destination passanger weather  temperature  time               coupon  \
13           Home     Alone   Sunny           55   6PM                  Bar
14           Home     Alone   Sunny           55   6PM      Restaurant(20-50)
15           Home     Alone   Sunny           80   6PM           Coffee House
35           Home     Alone   Sunny           55   6PM                  Bar
36           Home     Alone   Sunny           55   6PM      Restaurant(20-50)
…             …         …       …           …     …                   …
12675        Home     Alone   Snowy           30  10PM           Coffee House
12676        Home     Alone   Sunny           80   6PM      Restaurant(20-50)
12677        Home   Partner   Sunny           30   6PM        Restaurant(<20)
12678        Home   Partner   Sunny           30  10PM        Restaurant(<20)
12679        Home   Partner   Rainy           55   6PM  Carry out & Take away
```

```
      expiration  gender age      maritalStatus  …  CarryAway  \
13            1d  Female  21  Unmarried partner  …        NaN
14            1d  Female  21  Unmarried partner  …        NaN
15            2h  Female  21  Unmarried partner  …        NaN
35            1d    Male  21             Single  …        4~8
36            1d    Male  21             Single  …        4~8
…          …     …  ..                …  …        …
12675         2h    Male  26             Single  …        1~3
12676         1d    Male  26             Single  …        1~3
12677         1d    Male  26             Single  …        1~3
12678         2h    Male  26             Single  …        1~3
12679         1d    Male  26             Single  …        1~3

      RestaurantLessThan20 Restaurant20To50 toCoupon_GEQ5min  \
13                     4~8              1~3                1
14                     4~8              1~3                1
15                     4~8              1~3                1
35                     4~8            less1                1
36                     4~8            less1                1
…                    …              …              …
12675                  4~8              1~3                1
12676                  4~8              1~3                1
12677                  4~8              1~3                1
12678                  4~8              1~3                1
12679                  4~8              1~3                1

      toCoupon_GEQ15min toCoupon_GEQ25min direction_same direction_opp  Y  \
13                    0                 0              1             0  1
14                    1                 0              0             1  1
15                    0                 0              0             1  0
35                    0                 0              1             0  1
36                    1                 0              0             1  0
…                 …               …            …           … ..
12675                 1                 0              0             1  0
12676                 0                 0              1             0  1
12677                 1                 1              0             1  1
12678                 1                 0              1             0  0
12679                 0                 0              1             0  1

      row_count
13           14
14           15
15           16
35           36
36           37
…         …
```

```
12675     12676
12676     12677
12677     12678
12678     12679
12679     12680

[3237 rows x 27 columns]
```

5 Order by a column

SQL Equivalent:

```
SELECT * FROM dataset_1 ORDER BY coupon;
```

Python:

[9]:

```
df.sort_values('coupon')
```

[9]:

```
           destination  passanger weather  temperature  time           coupon   \
11702             Home    Partner   Sunny           30  10PM              Bar
9930   No Urgent Place      Alone   Snowy           30   2PM              Bar
10632             Home      Alone   Rainy           55   6PM              Bar
7997   No Urgent Place  Friend(s)   Rainy           55  10PM              Bar
11166             Work      Alone   Snowy           30   7AM              Bar
…                …        …     …          …   …              …
10476             Home      Alone   Sunny           80   6PM  Restaurant(<20)
5447              Home      Alone   Sunny           80  10PM  Restaurant(<20)
10478             Home      Alone   Snowy           30  10PM  Restaurant(<20)
5440   No Urgent Place      Alone   Sunny           80   2PM  Restaurant(<20)
0      No Urgent Place      Alone   Sunny           55   2PM  Restaurant(<20)

      expiration  gender     age      maritalStatus  …  CarryAway   \
11702         2h  Female  50plus    Married partner  …        4~8
9930          1d  Female      21             Single  …        gt8
10632         1d    Male      21             Single  …        gt8
7997          2h    Male      26  Unmarried partner  …        4~8
11166         1d  Female      41    Married partner  …        gt8
…          …     …     …                …  …        …
10476         1d  Female      31  Unmarried partner  …        1~3
5447          2h  Female  50plus             Single  …      less1
10478         2h  Female      31  Unmarried partner  …        1~3
5440          2h  Female  50plus             Single  …      less1
0             1d  Female      21  Unmarried partner  …        NaN

      RestaurantLessThan20 Restaurant20To50 toCoupon_GEQ5min   \
11702                  1~3            less1                1
9930                   gt8              4~8                1
10632                less1            less1                1
```

```
7997                 never              1~3                1
11166                  1~3            less1                1
…                    …              …              …
10476                  1~3            less1                1
5447                 less1            never                1
10478                  1~3            less1                1
5440                 less1            never                1
0                      4~8              1~3                1

      toCoupon_GEQ15min toCoupon_GEQ25min direction_same direction_opp  Y  \
11702                 1                 1              0             1  0
9930                  0                 0              0             1  0
10632                 1                 1              0             1  0
7997                  1                 0              0             1  1
11166                 1                 1              0             1  0
…                 …               …            …           … ..
10476                 0                 0              1             0  1
5447                  0                 0              1             0  0
10478                 1                 1              0             1  0
5440                  1                 0              0             1  0
0                     0                 0              0             1  1

      row_count
11702     11703
9930       9931
10632     10633
7997       7998
11166     11167
…         …
10476     10477
5447       5448
10478     10479
5440       5441
0             1

[12684 rows x 27 columns]
```

6 Rename a column

SQL Equivalent:

```
SELECT destination AS Destination FROM dataset_1;
```

Python:

[10]:

```
df.rename(columns={'destination':'Destination'}, inplace=True)
```

7 Group by with count

SQL Equivalent:

```
SELECT occupation, COUNT(*) AS Count FROM dataset_1 GROUP BY occupation;
```

Python:

```
[11]: df.groupby('occupation').size()
```

```
[11]: occupation
      Architecture & Engineering                   175
      Arts Design Entertainment Sports & Media     629
      Building & Grounds Cleaning & Maintenance     44
      Business & Financial                         544
      Community & Social Services                  241
      Computer & Mathematical                     1408
      Construction & Extraction                    154
      Education&Training&Library                   943
      Farming Fishing & Forestry                    43
      Food Preparation & Serving Related           298
      Healthcare Practitioners & Technical         244
      Healthcare Support                           242
      Installation Maintenance & Repair            133
      Legal                                        219
      Life Physical Social Science                 170
      Management                                   838
      Office & Administrative Support              639
      Personal Care & Service                      175
      Production Occupations                       110
      Protective Service                           175
      Retired                                      495
      Sales & Related                             1093
      Student                                     1584
      Transportation & Material Moving             218
      Unemployed                                  1870
      dtype: int64
```

```
[12]: df.groupby('occupation').size().to_frame('Count').reset_index()
```

```
[12]:                                  occupation  Count
      0                 Architecture & Engineering    175
      1   Arts Design Entertainment Sports & Media    629
      2  Building & Grounds Cleaning & Maintenance     44
      3                       Business & Financial    544
      4                Community & Social Services    241
      5                    Computer & Mathematical   1408
      6                  Construction & Extraction    154
      7                 Education&Training&Library    943
```

```
8                 Farming Fishing & Forestry     43
9         Food Preparation & Serving Related    298
10      Healthcare Practitioners & Technical    244
11                        Healthcare Support    242
12         Installation Maintenance & Repair    133
13                                     Legal    219
14              Life Physical Social Science    170
15                                Management    838
16           Office & Administrative Support    639
17                   Personal Care & Service    175
18                    Production Occupations    110
19                        Protective Service    175
20                                   Retired    495
21                           Sales & Related   1093
22                                   Student   1584
23          Transportation & Material Moving    218
24                                Unemployed   1870
```

8 Group by with average

SQL Equivalent:

```
SELECT weather, AVG(temperature) AS avg_temp FROM dataset_1 GROUP BY weather;
```

Python:

[13]:

```
df.groupby('weather')['temperature'].mean().to_frame('avg_temp').reset_index()
```

[13]:

```
  weather   avg_temp
0   Rainy  55.000000
1   Snowy  30.000000
2   Sunny  68.946271
```

# Group by with count of temperature entries **SQL Equivalent:**

```
SELECT weather, COUNT(temperature) AS Count_temp FROM dataset_1 GROUP BY weather;
```

Python:

[14]:

```
df.groupby('weather')['temperature'].size().to_frame('Count_temp').reset_index()
```

[14]:

```
  weather  Count_temp
0   Rainy        1210
1   Snowy        1405
2   Sunny       10069
```

9 Group by with count of distinct values

SQL Equivalent:

```
SELECT weather, COUNT(DISTINCT temperature) AS count_distinct_temp FROM dataset_1 GROUP BY wea
```

Python:

```
[15]: df.groupby('weather')['temperature'].nunique().to_frame('count_distinct_temp').
       ↪reset_index()
```

```
[15]:   weather  count_distinct_temp
      0   Rainy                    1
      1   Snowy                    1
      2   Sunny                    3
```

10 Group by with sum

SQL Equivalent:

```
SELECT weather, SUM(temperature) AS sum_temp FROM dataset_1 GROUP BY weather;
```

Python:

```
[16]: df.groupby('weather')['temperature'].sum().to_frame('sum_temp').reset_index()
```

```
[16]:   weather  sum_temp
      0   Rainy     66550
      1   Snowy     42150
      2   Sunny    694220
```

11 Group by with min and max

SQL Equivalent:

```
SELECT weather, MIN(temperature) AS min_temp FROM dataset_1 GROUP BY weather;
SELECT weather, MAX(temperature) AS max_temp FROM dataset_1 GROUP BY weather;
```

Python:

```
[18]: df.groupby('weather')['temperature'].min().to_frame('min_temp').reset_index()
```

```
[18]:   weather  min_temp
      0   Rainy        55
      1   Snowy        30
      2   Sunny        30
```

```
[19]: df.groupby('weather')['temperature'].max().to_frame('max_temp').reset_index()
```

```
[19]:   weather  max_temp
      0   Rainy        55
      1   Snowy        30
      2   Sunny        80
```

12 Group by with HAVING clause logic

SQL Equivalent:

```
SELECT occupation FROM dataset_1 GROUP BY occupation HAVING occupation = 'Student';
```

Python:

[20]:

```
df.groupby('occupation').filter(lambda x: x['occupation'].iloc[0] == 'Student').
  ↪groupby('occupation').size()
```

[20]:

```
occupation
Student    1584
dtype: int64
```

13 Union + Drop duplicates

SQL Equivalent:

```
SELECT DISTINCT destination FROM (
    SELECT * FROM dataset_1
    UNION
    SELECT * FROM table_to_union
);
```

If you're running this notebook yourself, make sure to define or load `df1` and `df2` from appropriate CSVs or simulated data before using these examples.

Python:

```
pd.concat([df, df1])['destination'].drop_duplicates()
```

14 INNER JOIN

SQL Equivalent:

```
SELECT a.destination, a.time, b.part_of_day
FROM dataset_1 a
INNER JOIN table_to_join b ON a.time = b.time;
```

Python:

```
pd.merge(df, df2[['time', 'part_of_day']], on='time', how='inner')[['destination', 'time', 'pa
```

If you're running this notebook yourself, make sure to define or load `df1` and `df2` from appropriate CSVs or simulated data before using these examples.

15 Filtering rows by value

SQL Equivalent:

```
SELECT destination, passanger FROM dataset_1 WHERE passanger = 'Alone';
```

Python:

```
[21]: df[df['passanger'] == 'Alone'][['Destination', 'passanger']]
```

```
[21]:          Destination passanger
      0      No Urgent Place     Alone
      13                Home     Alone
      14                Home     Alone
      15                Home     Alone
      16                Work     Alone
      …                  …        …
      12676             Home     Alone
      12680             Work     Alone
      12681             Work     Alone
      12682             Work     Alone
      12683             Work     Alone

      [7305 rows x 2 columns]
```

16 Filtering rows by prefix

SQL Equivalent:

```
SELECT * FROM dataset_1 WHERE weather LIKE 'Sun%';
```

Python:

```
[22]: df[df['weather'].str.startswith('Sun')]
```

```
[22]:          Destination  passanger weather  temperature  time  \
      0      No Urgent Place      Alone   Sunny           55   2PM
      1      No Urgent Place  Friend(s)   Sunny           80  10AM
      2      No Urgent Place  Friend(s)   Sunny           80  10AM
      3      No Urgent Place  Friend(s)   Sunny           80   2PM
      4      No Urgent Place  Friend(s)   Sunny           80   2PM
      …                  …         …     …             …    …
      12673             Home      Alone   Sunny           30   6PM
      12676             Home      Alone   Sunny           80   6PM
      12677             Home    Partner   Sunny           30   6PM
      12678             Home    Partner   Sunny           30  10PM
      12683             Work      Alone   Sunny           80   7AM

                            coupon expiration  gender age      maritalStatus  …  \
      0            Restaurant(<20)         1d  Female  21  Unmarried partner  …
      1               Coffee House         2h  Female  21  Unmarried partner  …
      2      Carry out & Take away         2h  Female  21  Unmarried partner  …
      3               Coffee House         2h  Female  21  Unmarried partner  …
      4               Coffee House         1d  Female  21  Unmarried partner  …
```

```
…                      …        …     …  ..                  …  …
12673  Carry out & Take away       1d    Male  26              Single  …
12676      Restaurant(20-50)       1d    Male  26              Single  …
12677        Restaurant(<20)       1d    Male  26              Single  …
12678        Restaurant(<20)       2h    Male  26              Single  …
12683      Restaurant(20-50)       2h    Male  26              Single  …

       CarryAway RestaurantLessThan20 Restaurant20To50 toCoupon_GEQ5min  \
0            NaN                  4~8              1~3                1
1            NaN                  4~8              1~3                1
2            NaN                  4~8              1~3                1
3            NaN                  4~8              1~3                1
4            NaN                  4~8              1~3                1
…          …                  …              …              …
12673        1~3                  4~8              1~3                1
12676        1~3                  4~8              1~3                1
12677        1~3                  4~8              1~3                1
12678        1~3                  4~8              1~3                1
12683        1~3                  4~8              1~3                1

      toCoupon_GEQ15min toCoupon_GEQ25min direction_same direction_opp  Y  \
0                     0                 0              0             1  1
1                     0                 0              0             1  0
2                     1                 0              0             1  1
3                     1                 0              0             1  0
4                     1                 0              0             1  0
…                  …               …            …           … ..
12673                 0                 0              0             1  0
12676                 0                 0              1             0  1
12677                 1                 1              0             1  1
12678                 1                 0              1             0  0
12683                 0                 0              1             0  0

      row_count
0             1
1             2
2             3
3             4
4             5
…         …
12673     12674
12676     12677
12677     12678
12678     12679
12683     12684

[10069 rows x 27 columns]
```

17 Filter values within a range

SQL Equivalent:

```
SELECT DISTINCT temperature FROM dataset_1 WHERE temperature BETWEEN 29 AND 75;
```

Python:

[23]:

```
df[(df['temperature'] >= 29) & (df['temperature'] <= 75)]['temperature'].
 ↪unique()
```

[23]:

```
array([55, 30], dtype=int64)
```

18 Filter rows with specific values

SQL Equivalent:

```
SELECT occupation FROM dataset_1 WHERE occupation IN ('Sales & Related', 'Management');
```

Python:

[24]:

```
df[df['occupation'].isin(['Sales & Related','Management'])][['occupation']]
```

[24]:

```
            occupation
193    Sales & Related
194    Sales & Related
195    Sales & Related
196    Sales & Related
197    Sales & Related
…                  …
12679  Sales & Related
12680  Sales & Related
12681  Sales & Related
12682  Sales & Related
12683  Sales & Related

[1931 rows x 1 columns]
```

Conclusion

In this notebook, we explored how to perform essential data analysis tasks using both **Python (pandas)** and **SQL** side-by-side. From filtering and sorting to grouping and aggregating data, we covered a wide range of operations commonly used in real-world data workflows.

This comparative approach not only strengthens your Python and SQL skills but also helps you transition smoothly between the two, depending on the data environment you're working with.

Keep practicing these techniques on different datasets to build a strong foundation in data analysis. Remember — clean, well-understood data is the first step toward building powerful insights and models.

Happy Learning!