18-11-2025 SQL workshop-3

In [1]:
```
import pandas as pd
```

In [2]:
```
sql = pd.read_csv(r"C:\Users\karthik reddy\OneDrive\Desktop\sql_18-11-2025\dataset_
```

In [3]:
```
sql
```

Out[3]:

	destination	passanger	weather	temperature	time	coupon	expiration
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d
...	...	...	...	...	...	...	...
12679	Home	Partner	Rainy	55	6PM	Carry out & Take away	1d
12680	Work	Alone	Rainy	55	7AM	Carry out & Take away	1d
12681	Work	Alone	Snowy	30	7AM	Coffee House	1d
12682	Work	Alone	Snowy	30	7AM	Bar	1d
12683	Work	Alone	Sunny	80	7AM	Restaurant(20-50)	2h

12684 rows × 27 columns

In [4]:
```
sql[['weather','temperature']]
```

Out[4]:

	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 × 2 columns

In [5]:
```
sql.head(10)
```

Out[5]:

	destination	passanger	weather	temperature	time	coupon	expiration	gende
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d	Femal
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h	Femal
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h	Femal
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h	Femal
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d	Femal
5	No Urgent Place	Friend(s)	Sunny	80	6PM	Restaurant(<20)	2h	Femal
6	No Urgent Place	Friend(s)	Sunny	55	2PM	Carry out & Take away	1d	Femal
7	No Urgent Place	Kid(s)	Sunny	80	10AM	Restaurant(<20)	2h	Femal
8	No Urgent Place	Kid(s)	Sunny	80	10AM	Carry out & Take away	2h	Femal
9	No Urgent Place	Kid(s)	Sunny	80	10AM	Bar	1d	Femal

10 rows × 27 columns

In [6]:
```
sql['passanger'].unique()
```

Out[6]:
```
array(['Alone', 'Friend(s)', 'Kid(s)', 'Partner'], dtype=object)
```

In [7]:
```
sql[sql['destination']=='Home']
```

Out[7]:

	destination	passanger	weather	temperature	time	coupon	expiration
13	Home	Alone	Sunny	55	6PM	Bar	1d
14	Home	Alone	Sunny	55	6PM	Restaurant(20-50)	1d
15	Home	Alone	Sunny	80	6PM	Coffee House	2h
35	Home	Alone	Sunny	55	6PM	Bar	1d
36	Home	Alone	Sunny	55	6PM	Restaurant(20-50)	1d
...	...	...	...	...	...	...	...
12675	Home	Alone	Snowy	30	10PM	Coffee House	2h
12676	Home	Alone	Sunny	80	6PM	Restaurant(20-50)	1d
12677	Home	Partner	Sunny	30	6PM	Restaurant(<20)	1d
12678	Home	Partner	Sunny	30	10PM	Restaurant(<20)	2h
12679	Home	Partner	Rainy	55	6PM	Carry out & Take away	1d

3237 rows × 27 columns

In [8]:
```
sql.sort_values('coupon')
```

Out[8]:

	destination	passanger	weather	temperature	time	coupon	expiration	g
11702	Home	Partner	Sunny	30	10PM	Bar	2h	F
9930	No Urgent Place	Alone	Snowy	30	2PM	Bar	1d	F
10632	Home	Alone	Rainy	55	6PM	Bar	1d	
7997	No Urgent Place	Friend(s)	Rainy	55	10PM	Bar	2h	
11166	Work	Alone	Snowy	30	7AM	Bar	1d	F
...	...	...	...	...	...	...	...	
10476	Home	Alone	Sunny	80	6PM	Restaurant(<20)	1d	F
5447	Home	Alone	Sunny	80	10PM	Restaurant(<20)	2h	F
10478	Home	Alone	Snowy	30	10PM	Restaurant(<20)	2h	F
5440	No Urgent Place	Alone	Sunny	80	2PM	Restaurant(<20)	2h	F
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d	F

12684 rows × 27 columns

In [9]:
```
sql.rename(columns={'destination':'Destination'},inplace=True)
```

In [10]:
```
sql
```

Out[10]:

	Destination	passanger	weather	temperature	time	coupon	expiration
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d
...	...	...	...	...	...	...	...
12679	Home	Partner	Rainy	55	6PM	Carry out & Take away	1d
12680	Work	Alone	Rainy	55	7AM	Carry out & Take away	1d
12681	Work	Alone	Snowy	30	7AM	Coffee House	1d
12682	Work	Alone	Snowy	30	7AM	Bar	1d
12683	Work	Alone	Sunny	80	7AM	Restaurant(20-50)	2h

12684 rows × 27 columns

In [11]:
```
sql.groupby('occupation').size().to_frame('Count').reset_index()
```

Out[11]:

	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

In [12]:
```
sql.groupby('weather')['temperature'].mean().to_frame('avg_temp').reset_index()
```

Out[12]:

	weather	avg_temp
0	Rainy	55.000000
1	Snowy	30.000000
2	Sunny	68.946271

In [13]:
```
sql.groupby('weather')['temperature'].size().to_frame('Count_temp').reset_index()
```

Out[13]:

	weather	Count_temp
0	Rainy	1210
1	Snowy	1405
2	Sunny	10069

In [14]:
```
sql.groupby('weather')['temperature'].nunique().to_frame('count_distinct_temp').res
```

Out[14]:

	weather	count_distinct_temp
0	Rainy	1
1	Snowy	1
2	Sunny	3

In [15]:
```
sql.groupby('weather')['temperature'].sum().to_frame('sum_temp').reset_index()
```

Out[15]:

	weather	sum_temp
0	Rainy	66550
1	Snowy	42150
2	Sunny	694220

In [16]:
```
sql.groupby('weather')['temperature'].min().to_frame('min_temp').reset_index()
```

Out[16]:

	weather	min_temp
0	Rainy	55
1	Snowy	30
2	Sunny	30

In [17]:
```
sql.groupby('weather')['temperature'].max().to_frame('max_temp').reset_index()
```

Out[17]:

	weather	max_temp
0	Rainy	55
1	Snowy	30
2	Sunny	80

In [18]:
```
sql.groupby('occupation').filter(lambda x: x['occupation'].iloc[0] ==
'Student').groupby('occupation').size()
```

Out[18]:
```
occupation
Student    1584
dtype: int64
```

In [27]:
```
import pandas as pd

df = pd.read_csv(r"C:\Users\karthik reddy\OneDrive\Desktop\sql_18-11-2025\dataset_1
df1 = pd.read_csv(r"C:\Users\karthik reddy\OneDrive\Desktop\sql_18-11-2025\dataset_

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

In [28]:
```
result
```

Out[28]:
```
0     No Urgent Place
13               Home
16               Work
Name: destination, dtype: object
```

In [30]:
```
import pandas as pd

# Example DataFrame creations
df = pd.read_csv(r"C:\Users\karthik reddy\OneDrive\Desktop\sql_18-11-2025\dataset_1
df2 = pd.read_csv(r"C:\Users\karthik reddy\OneDrive\Desktop\sql_18-11-2025\dataset_

# Now the merge will work
merged = pd.merge(df, df2[['time', 'part_of_day']], on='time', how='inner')[['desti
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[30], line 8
      5 df2 = pd.read_csv(r"C:\Users\karthik reddy\OneDrive\Desktop\sql_18-11-2025\dataset_1_202511181007.csv")
      7 # Now the merge will work
----> 8 merged = pd.merge(df, df2[['time', 'part_of_day']], on='time', how='inner')[['destination', 'time', 'part_of_day']]

File D:\Anaconda\Lib\site-packages\pandas\core\frame.py:4108, in DataFrame.__getitem__(self, key)
   4106     if is_iterator(key):
   4107         key = list(key)
-> 4108     indexer = self.columns._get_indexer_strict(key, "columns")[1]
   4110 # take() does not accept boolean indexers
   4111 if getattr(indexer, "dtype", None) == bool:

File D:\Anaconda\Lib\site-packages\pandas\core\indexes\base.py:6200, in Index._get_indexer_strict(self, key, axis_name)
   6197 else:
   6198     keyarr, indexer, new_indexer = self._reindex_non_unique(keyarr)
-> 6200 self._raise_if_missing(keyarr, indexer, axis_name)
   6202 keyarr = self.take(indexer)
   6203 if isinstance(key, Index):
   6204     # GH 42790 - Preserve name from an Index

File D:\Anaconda\Lib\site-packages\pandas\core\indexes\base.py:6252, in Index._raise_if_missing(self, key, indexer, axis_name)
   6249     raise KeyError(f"None of [{key}] are in the [{axis_name}]")
   6251 not_found = list(ensure_index(key)[missing_mask.nonzero()[0]].unique())
-> 6252 raise KeyError(f"{not_found} not in index")

KeyError: "['part_of_day'] not in index"
```

In [31]:
```
print(df2.columns)
```

```
Index(['destination', 'passanger', 'weather', 'temperature', 'time', 'coupon',
       'expiration', 'gender', 'age', 'maritalStatus', 'has_children',
       'education', 'occupation', 'income', 'car', 'Bar', 'CoffeeHouse',
       'CarryAway', 'RestaurantLessThan20', 'Restaurant20To50',
       'toCoupon_GEQ5min', 'toCoupon_GEQ15min', 'toCoupon_GEQ25min',
       'direction_same', 'direction_opp', 'Y', 'row_count'],
      dtype='object')
```

In [32]:
```
print(df2.columns)  # See available columns

# Use only columns present in df2, example:
merged = pd.merge(df, df2[['time']], on='time', how='inner')
```

```
Index(['destination', 'passanger', 'weather', 'temperature', 'time', 'coupon',
       'expiration', 'gender', 'age', 'maritalStatus', 'has_children',
       'education', 'occupation', 'income', 'car', 'Bar', 'CoffeeHouse',
       'CarryAway', 'RestaurantLessThan20', 'Restaurant20To50',
       'toCoupon_GEQ5min', 'toCoupon_GEQ15min', 'toCoupon_GEQ25min',
       'direction_same', 'direction_opp', 'Y', 'row_count'],
      dtype='object')
```

In [33]:
```
merged = pd.merge(df, df2[['time', 'part_of_day']], on='time', how='inner')[['desti
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[33], line 1
----> 1 merged = pd.merge(df, df2[['time', 'part_of_day']], on='time', how='inner')
[['destination', 'time', 'part_of_day']]

File D:\Anaconda\Lib\site-packages\pandas\core\frame.py:4108, in DataFrame.__getitem
__(self, key)
   4106     if is_iterator(key):
   4107         key = list(key)
-> 4108     indexer = self.columns._get_indexer_strict(key, "columns")[1]
   4110 # take() does not accept boolean indexers
   4111 if getattr(indexer, "dtype", None) == bool:

File D:\Anaconda\Lib\site-packages\pandas\core\indexes\base.py:6200, in Index._get_i
ndexer_strict(self, key, axis_name)
   6197 else:
   6198     keyarr, indexer, new_indexer = self._reindex_non_unique(keyarr)
-> 6200 self._raise_if_missing(keyarr, indexer, axis_name)
   6202 keyarr = self.take(indexer)
   6203 if isinstance(key, Index):
   6204     # GH 42790 - Preserve name from an Index

File D:\Anaconda\Lib\site-packages\pandas\core\indexes\base.py:6252, in Index._raise
_if_missing(self, key, indexer, axis_name)
   6249     raise KeyError(f"None of [{key}] are in the [{axis_name}]")
   6251 not_found = list(ensure_index(key)[missing_mask.nonzero()[0]].unique())
-> 6252 raise KeyError(f"{not_found} not in index")

KeyError: "['part_of_day'] not in index"
```

In [34]:
```
df[df['passanger'] == 'Alone'][['destination', 'passanger']]
```

Out[34]:

	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 × 2 columns

In [35]:
```
df[df['weather'].str.startswith('Sun')]
```

Out[35]:

	destination	passanger	weather	temperature	time	coupon	expiration
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d
...	...	...	...	...	...	...	...
12673	Home	Alone	Sunny	30	6PM	Carry out & Take away	1d
12676	Home	Alone	Sunny	80	6PM	Restaurant(20-50)	1d
12677	Home	Partner	Sunny	30	6PM	Restaurant(<20)	1d
12678	Home	Partner	Sunny	30	10PM	Restaurant(<20)	2h
12683	Work	Alone	Sunny	80	7AM	Restaurant(20-50)	2h

10069 rows × 27 columns

In [36]:
```
df[(df['temperature'] >= 29) & (df['temperature'] <= 75)]['temperature'].unique()
```

Out[36]:
```
array([55, 30])
```

In [37]:
```
df[df['occupation'].isin(['Sales & Related', 'Management'])][['occupation']]
```

Out[37]:

	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 × 1 columns

In [ ]: