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

In [2]:
```
df = pd.read_csv(r'C:\Users\ADMIN\Desktop\sql\dataset_1_202511181007.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	destination	passanger	weather	temperature	time	coupon	expiration	gender	age	maritalStatus	...	CarryAway
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d	Female	21	Unmarried partner	...	NaN
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h	Female	21	Unmarried partner	...	NaN
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h	Female	21	Unmarried partner	...	NaN
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h	Female	21	Unmarried partner	...	NaN
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d	Female	21	Unmarried partner	...	NaN

5 rows × 27 columns

In [4]:
```
df[['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]:
```
df.head(10)
```

Out[5]:

	destination	passanger	weather	temperature	time	coupon	expiration	gender	age	maritalStatus	...	CarryAway
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d	Female	21	Unmarried partner	...	NaN
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h	Female	21	Unmarried partner	...	NaN
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h	Female	21	Unmarried partner	...	NaN
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h	Female	21	Unmarried partner	...	NaN
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d	Female	21	Unmarried partner	...	NaN
5	No Urgent Place	Friend(s)	Sunny	80	6PM	Restaurant(<20)	2h	Female	21	Unmarried partner	...	NaN
6	No Urgent Place	Friend(s)	Sunny	55	2PM	Carry out & Take away	1d	Female	21	Unmarried partner	...	NaN
7	No Urgent Place	Kid(s)	Sunny	80	10AM	Restaurant(<20)	2h	Female	21	Unmarried partner	...	NaN
8	No Urgent Place	Kid(s)	Sunny	80	10AM	Carry out & Take away	2h	Female	21	Unmarried partner	...	NaN
9	No Urgent Place	Kid(s)	Sunny	80	10AM	Bar	1d	Female	21	Unmarried partner	...	NaN

10 rows × 27 columns

In [6]:
```
df[0:11]
```

Out[6]:

	destination	passanger	weather	temperature	time	coupon	expiration	gender	age	maritalStatus	...	CarryAwa
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d	Female	21	Unmarried partner	...	Nal
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h	Female	21	Unmarried partner	...	Nal
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h	Female	21	Unmarried partner	...	Nal
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h	Female	21	Unmarried partner	...	Nal
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d	Female	21	Unmarried partner	...	Nal
5	No Urgent Place	Friend(s)	Sunny	80	6PM	Restaurant(<20)	2h	Female	21	Unmarried partner	...	Nal
6	No Urgent Place	Friend(s)	Sunny	55	2PM	Carry out & Take away	1d	Female	21	Unmarried partner	...	Nal
7	No Urgent Place	Kid(s)	Sunny	80	10AM	Restaurant(<20)	2h	Female	21	Unmarried partner	...	Nal
8	No Urgent Place	Kid(s)	Sunny	80	10AM	Carry out & Take away	2h	Female	21	Unmarried partner	...	Nal
9	No Urgent Place	Kid(s)	Sunny	80	10AM	Bar	1d	Female	21	Unmarried partner	...	Nal
10	No Urgent Place	Kid(s)	Sunny	80	2PM	Restaurant(<20)	1d	Female	21	Unmarried partner	...	Nal

11 rows × 27 columns

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

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

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

Out[8]:

	destination	passanger	weather	temperature	time	coupon	expiration	gender	age	maritalStatus	...	CarryA
13	Home	Alone	Sunny	55	6PM	Bar	1d	Female	21	Unmarried partner	...	
14	Home	Alone	Sunny	55	6PM	Restaurant(20-50)	1d	Female	21	Unmarried partner	...	
15	Home	Alone	Sunny	80	6PM	Coffee House	2h	Female	21	Unmarried partner	...	
35	Home	Alone	Sunny	55	6PM	Bar	1d	Male	21	Single	...	
36	Home	Alone	Sunny	55	6PM	Restaurant(20-50)	1d	Male	21	Single	...	
...	...	...	...	...	...	...	...	...	...	...	...	
12675	Home	Alone	Snowy	30	10PM	Coffee House	2h	Male	26	Single	...	
12676	Home	Alone	Sunny	80	6PM	Restaurant(20-50)	1d	Male	26	Single	...	
12677	Home	Partner	Sunny	30	6PM	Restaurant(<20)	1d	Male	26	Single	...	
12678	Home	Partner	Sunny	30	10PM	Restaurant(<20)	2h	Male	26	Single	...	
12679	Home	Partner	Rainy	55	6PM	Carry out & Take away	1d	Male	26	Single	...	

3237 rows × 27 columns

In [9]:
```
df.sort_values('coupon')
```

Out[9]:

	destination	passanger	weather	temperature	time	coupon	expiration	gender	age	maritalStatus	...	Car
11702	Home	Partner	Sunny	30	10PM	Bar	2h	Female	50plus	Married partner	...	
9930	No Urgent Place	Alone	Snowy	30	2PM	Bar	1d	Female	21	Single	...	
10632	Home	Alone	Rainy	55	6PM	Bar	1d	Male	21	Single	...	
7997	No Urgent Place	Friend(s)	Rainy	55	10PM	Bar	2h	Male	26	Unmarried partner	...	
11166	Work	Alone	Snowy	30	7AM	Bar	1d	Female	41	Married partner	...	
...	...	...	...	...	...	...	...	...	...	...	...	
10476	Home	Alone	Sunny	80	6PM	Restaurant(<20)	1d	Female	31	Unmarried partner	...	
5447	Home	Alone	Sunny	80	10PM	Restaurant(<20)	2h	Female	50plus	Single	...	
10478	Home	Alone	Snowy	30	10PM	Restaurant(<20)	2h	Female	31	Unmarried partner	...	
5440	No Urgent Place	Alone	Sunny	80	2PM	Restaurant(<20)	2h	Female	50plus	Single	...	
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d	Female	21	Unmarried partner	...	

12684 rows × 27 columns

In [10]:
```
df.rename(columns={'destination':'Destination'},inplace=True)
```

In [12]:
```
df.columns
```

Out[12]:
```
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 [13]:
```
df.groupby('occupation').size().to_frame('Count').reset_index()
```

Out[13]:

	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

	occupation	Count
22	Student	1584
23	Transportation & Material Moving	218
24	Unemployed	1870

In [14]:
```
df.groupby('weather')['temperature'].mean().to_frame('avg_temp').reset_index()
```

Out[14]:

	weather	avg_temp
0	Rainy	55.000000
1	Snowy	30.000000
2	Sunny	68.946271

In [15]:
```
df.groupby('weather')['temperature'].size().to_frame('Count_temp').reset_index()
```

Out[15]:

	weather	Count_temp
0	Rainy	1210
1	Snowy	1405
2	Sunny	10069

In [16]:
```
df.groupby('weather')['temperature'].nunique().to_frame('count_distinct_temp').reset_index()
```

Out[16]:

	weather	count_distinct_temp
0	Rainy	1
1	Snowy	1
2	Sunny	3

In [17]:
```
df.groupby('weather')['temperature'].sum().to_frame('sum_temp').reset_index()
```

Out[17]:

	weather	sum_temp
0	Rainy	66550
1	Snowy	42150
2	Sunny	694220

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

Out[18]:

	weather	min_temp
0	Rainy	55
1	Snowy	30
2	Sunny	30

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

Out[19]:

	weather	max_temp
0	Rainy	55
1	Snowy	30
2	Sunny	80

In [21]:
```
df.groupby('occupation').filter(lambda x:x['occupation'].iloc[0]=='Student').groupby('occupation').size()
```

Out[21]:
```
occupation
Student    1584
dtype: int64
```

In [25]:
```
df1=df
```

In [27]:
```
df.columns
```

Out[27]:
```
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 [28]:
```
pd.concat([df,df1])['Destination'].drop_duplicates()
```

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

In [36]:
```
df[df['passanger']=='Alone'][['Destination','passanger']]
```

Out[36]:

	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 [37]:
```
df[df['weather'].str.startswith('Sun')]
```

Out[37]:

	Destination	passanger	weather	temperature	time	coupon	expiration	gender	age	maritalStatus	...	Carry
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d	Female	21	Unmarried partner	...	
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h	Female	21	Unmarried partner	...	
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h	Female	21	Unmarried partner	...	
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h	Female	21	Unmarried partner	...	
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d	Female	21	Unmarried partner	...	
...	...	...	...	...	...	...	...	...	...	...	...	
12673	Home	Alone	Sunny	30	6PM	Carry out & Take away	1d	Male	26	Single	...	
12676	Home	Alone	Sunny	80	6PM	Restaurant(20-50)	1d	Male	26	Single	...	
12677	Home	Partner	Sunny	30	6PM	Restaurant(<20)	1d	Male	26	Single	...	
12678	Home	Partner	Sunny	30	10PM	Restaurant(<20)	2h	Male	26	Single	...	
12683	Work	Alone	Sunny	80	7AM	Restaurant(20-50)	2h	Male	26	Single	...	

10069 rows × 27 columns

In [38]:
```
df[(df['temperature']>=29) & (df['temperature']<=75)]['temperature'].unique()
```

Out[38]:
```
array([55, 30], dtype=int64)
```

In [39]:
```
df[df['occupation'].isin(['Sales & Related','Management'])][['occupation']]
```

Out[39]:

	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

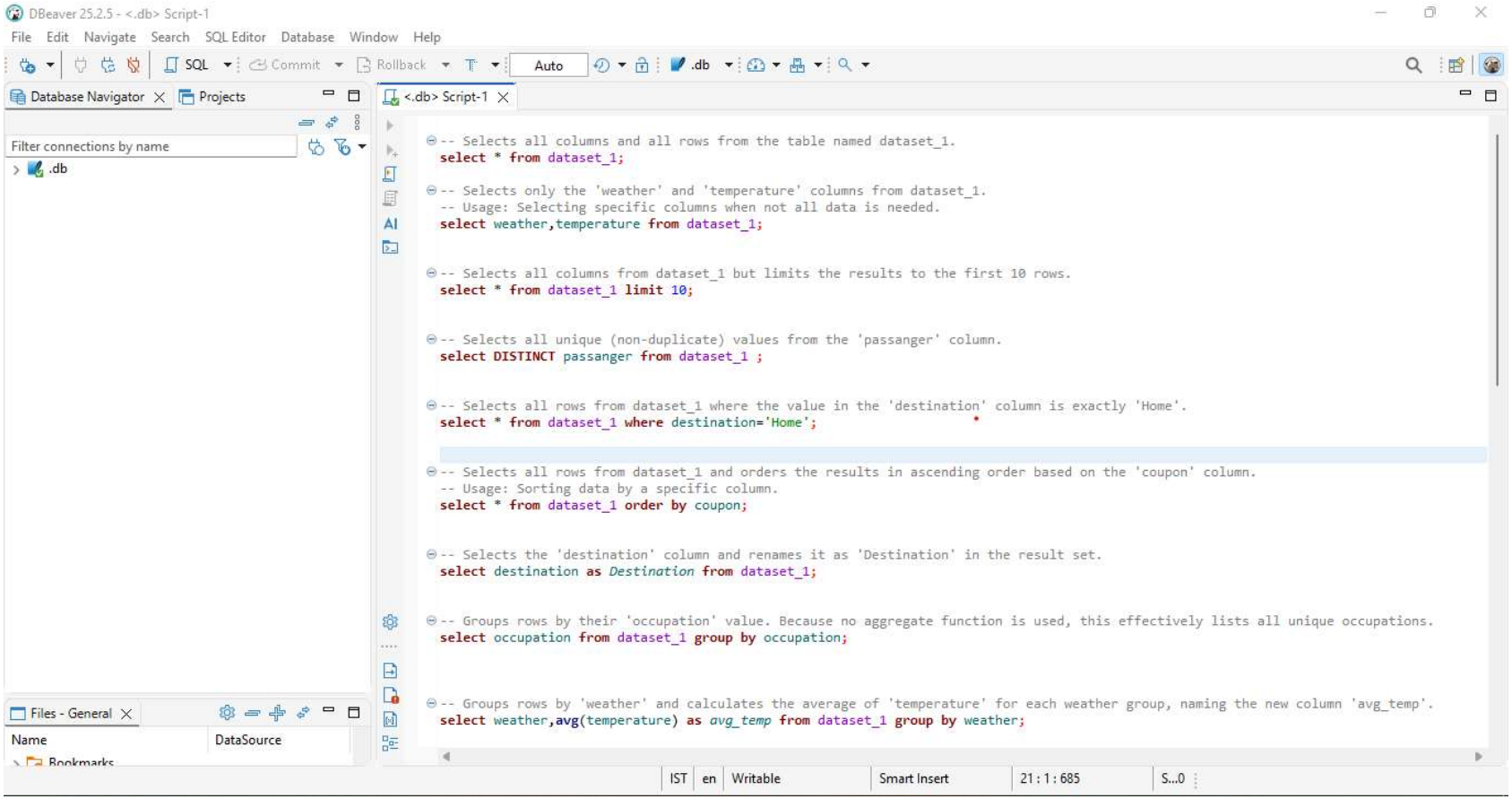
1931 rows × 1 columns

In [47]:

```
from PIL import Image
%matplotlib inline
sql1= Image.open(r"C:\Users\ADMIN\Pictures\SQL1.png")

sql1
```

Out[47]:

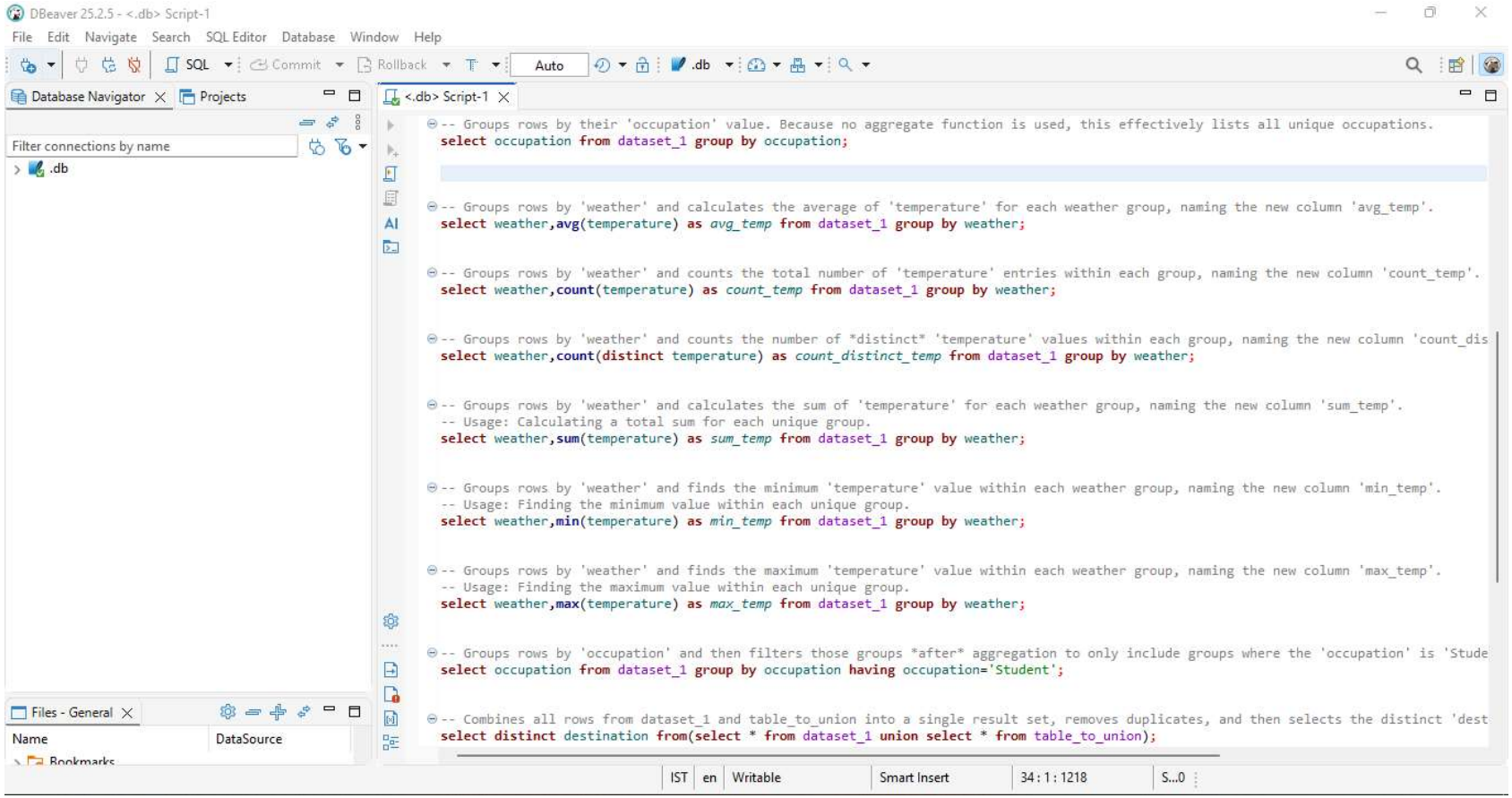


In [48]:

```
sql2= Image.open(r"C:\Users\ADMIN\Pictures\SQL2.png")

sql2
```

Out[48]:

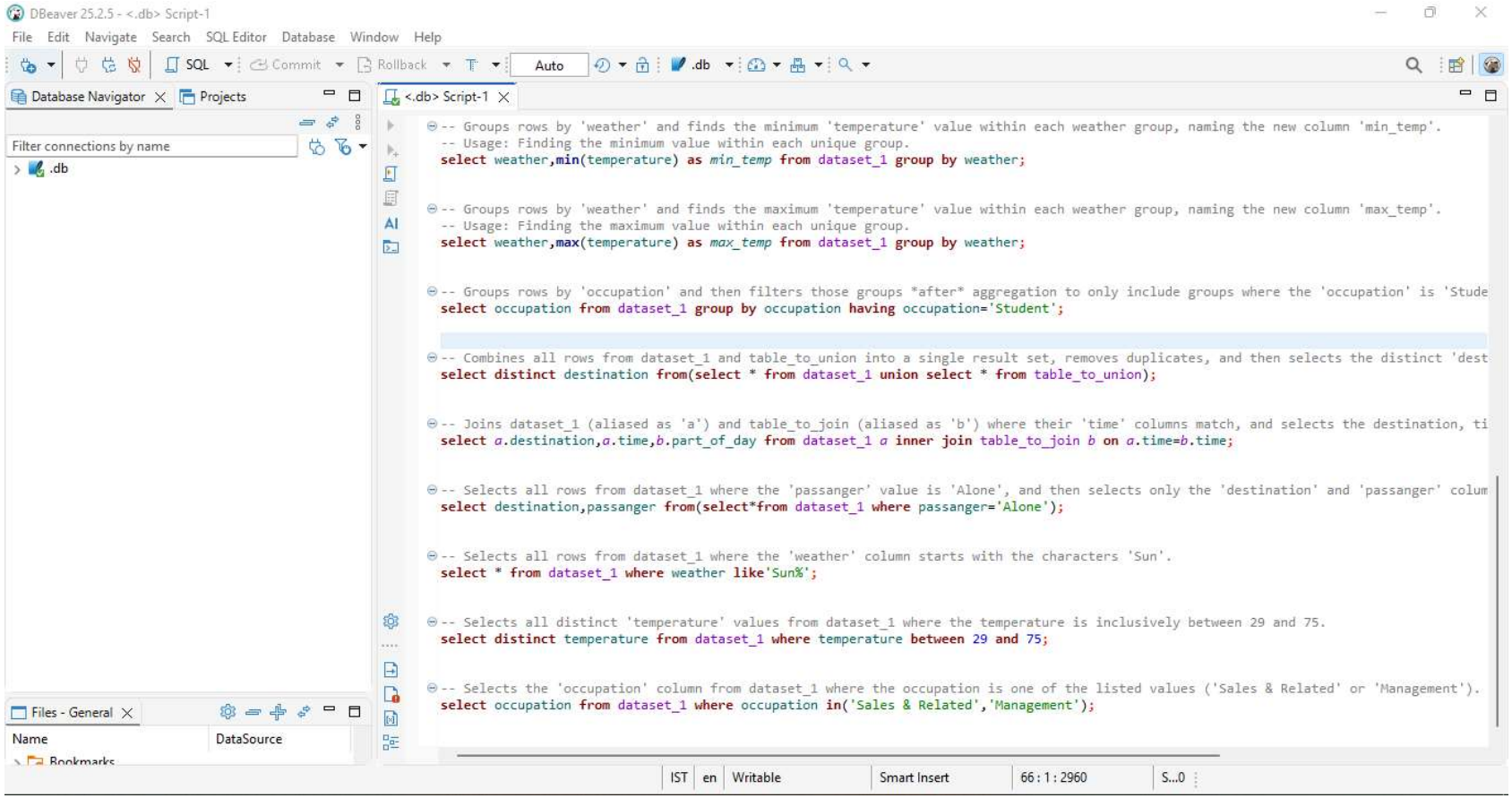


In [50]:

```
sql3= Image.open(r"C:\Users\ADMIN\Pictures\SQL3.png")

sql3
```

Out[50]:


DBeaver 25.2.5 - <.db> Script-1
File Edit Navigate Search SQL Editor Database Window Help
SQL Commit Rollback Auto .db
Database Navigator Projects
Filter connections by name
.db
<.db> Script-1
-- Groups rows by 'weather' and finds the minimum 'temperature' value within each weather group, naming the new column 'min_temp'.
-- Usage: Finding the minimum value within each unique group.
select weather,min(temperature) as min_temp from dataset_1 group by weather;
-- Groups rows by 'weather' and finds the maximum 'temperature' value within each weather group, naming the new column 'max_temp'.
-- Usage: Finding the maximum value within each unique group.
select weather,max(temperature) as max_temp from dataset_1 group by weather;
-- Groups rows by 'occupation' and then filters those groups *after* aggregation to only include groups where the 'occupation' is 'Stude
select occupation from dataset_1 group by occupation having occupation='Student';
-- Combines all rows from dataset_1 and table_to_union into a single result set, removes duplicates, and then selects the distinct 'dest
select distinct destination from(select * from dataset_1 union select * from table_to_union);
-- Joins dataset_1 (aliased as 'a') and table_to_join (aliased as 'b') where their 'time' columns match, and selects the destination, ti
select a.destination,a.time,b.part_of_day from dataset_1 a inner join table_to_join b on a.time=b.time;
-- Selects all rows from dataset_1 where the 'passanger' value is 'Alone', and then selects only the 'destination' and 'passanger' colum
select destination,passanger from(select*from dataset_1 where passanger='Alone');
-- Selects all rows from dataset_1 where the 'weather' column starts with the characters 'Sun'.
select * from dataset_1 where weather like'Sun%';
-- Selects all distinct 'temperature' values from dataset_1 where the temperature is inclusively between 29 and 75.
select distinct temperature from dataset_1 where temperature between 29 and 75;
-- Selects the 'occupation' column from dataset_1 where the occupation is one of the listed values ('Sales & Related' or 'Management').
select occupation from dataset_1 where occupation in('Sales & Related','Management');
Files - General
Name
DataSource
Bookmarks
IST en Writable Smart Insert 66 : 1 : 2960 S...0