ConnectingSQL

April 1, 2025

1 Performing ELT process by extracting data from kaggle, connecting to Google Cloud SQL and Performing SQL Queries

1.1 Read data from CSV file

[10]:
```
import pandas as pd
df = pd.read_csv('orders.csv', na_values =['Not Available','unknown'])
df.head(20)
```

[10]:
```
    Order Id  Order Date       Ship Mode      Segment        Country  \
0          1  2023-03-01    Second Class     Consumer  United States
1          2  2023-08-15    Second Class     Consumer  United States
2          3  2023-01-10    Second Class    Corporate  United States
3          4  2022-06-18  Standard Class     Consumer  United States
4          5  2022-07-13  Standard Class     Consumer  United States
5          6  2022-03-13             NaN     Consumer  United States
6          7  2022-12-28  Standard Class     Consumer  United States
7          8  2022-01-25  Standard Class     Consumer  United States
8          9  2023-03-23             NaN     Consumer  United States
9         10  2023-05-16  Standard Class     Consumer  United States
10        11  2023-03-31             NaN     Consumer  United States
11        12  2023-12-25             NaN     Consumer  United States
12        13  2022-02-11  Standard Class     Consumer  United States
13        14  2023-07-18  Standard Class     Consumer  United States
14        15  2023-11-09             NaN  Home Office  United States
15        16  2022-06-18  Standard Class  Home Office  United States
16        17  2022-02-04  Standard Class     Consumer  United States
17        18  2023-08-04    Second Class     Consumer  United States
18        19  2022-01-23    Second Class     Consumer  United States
19        20  2022-01-11    Second Class     Consumer  United States

               City       State  Postal Code Region         Category  \
0         Henderson    Kentucky        42420  South        Furniture
1         Henderson    Kentucky        42420  South        Furniture
2       Los Angeles  California        90036   West  Office Supplies
3   Fort Lauderdale     Florida        33311  South        Furniture
4   Fort Lauderdale     Florida        33311  South  Office Supplies
5       Los Angeles  California        90032   West        Furniture
```

```
6       Los Angeles      California        90032     West  Office Supplies
7       Los Angeles      California        90032     West       Technology
8       Los Angeles      California        90032     West  Office Supplies
9       Los Angeles      California        90032     West  Office Supplies
10      Los Angeles      California        90032     West        Furniture
11      Los Angeles      California        90032     West       Technology
12          Concord  North Carolina        28027    South  Office Supplies
13          Seattle      Washington        98103     West  Office Supplies
14       Fort Worth           Texas        76106  Central  Office Supplies
15       Fort Worth           Texas        76106  Central  Office Supplies
16          Madison       Wisconsin        53711  Central  Office Supplies
17      West Jordan            Utah        84084     West  Office Supplies
18    San Francisco      California        94109     West  Office Supplies
19    San Francisco      California        94109     West       Technology

   Sub Category       Product Id  cost price  List Price  Quantity  \
0     Bookcases  FUR-BO-10001798         240         260         2
1        Chairs  FUR-CH-10000454         600         730         3
2        Labels  OFF-LA-10000240          10          10         2
3        Tables  FUR-TA-10000577         780         960         5
4       Storage  OFF-ST-10000760          20          20         2
5   Furnishings  FUR-FU-10001487          50          50         7
6           Art  OFF-AR-10002833          10          10         4
7        Phones  TEC-PH-10002275         860         910         6
8       Binders  OFF-BI-10003910          20          20         3
9    Appliances  OFF-AP-10002892          90         110         5
10       Tables  FUR-TA-10001539        1470        1710         9
11       Phones  TEC-PH-10002033         750         910         4
12        Paper  OFF-PA-10002365          20          20         3
13      Binders  OFF-BI-10003656         360         410         3
14   Appliances  OFF-AP-10002311          60          70         5
15      Binders  OFF-BI-10000756           0           0         3
16      Storage  OFF-ST-10004186         610         670         6
17      Storage  OFF-ST-10000107          60          60         2
18          Art  OFF-AR-10003056          10          10         2
19       Phones  TEC-PH-10001949         170         210         3

    Discount Percent
0                  2
1                  3
2                  5
3                  2
4                  5
5                  3
6                  3
7                  5
8                  2
```

```
9                      3
10                     3
11                     3
12                     3
13                     2
14                     5
15                     5
16                     3
17                     4
18                     4
19                     3
```

1.2 Handle null values

[11]:
```
# Check for null values and also for distinct values
df['Ship Mode'].unique()
```

[11]:
```
array(['Second Class', 'Standard Class', nan, 'First Class', 'Same Day'],
      dtype=object)
```

1.3 Rename column names to lower case and remove spaces

[18]:
```
#df.columns
#df.columns = df.columns.str.lower() #This will convert column names in lower␣
 ↪case
#df.columns = df.columns.str.replace(' ','_') # This will remove space and␣
 ↪add'_'
df.columns
df.head()
```

[18]:
```
   order_id  order_date       ship_mode    segment        country  \
0         1  2023-03-01    Second Class   Consumer  United States
1         2  2023-08-15    Second Class   Consumer  United States
2         3  2023-01-10    Second Class  Corporate  United States
3         4  2022-06-18  Standard Class   Consumer  United States
4         5  2022-07-13  Standard Class   Consumer  United States

              city       state  postal_code region         category  \
0        Henderson    Kentucky        42420  South        Furniture
1        Henderson    Kentucky        42420  South        Furniture
2      Los Angeles  California        90036   West  Office Supplies
3  Fort Lauderdale     Florida        33311  South        Furniture
4  Fort Lauderdale     Florida        33311  South  Office Supplies

  sub_category       product_id  cost_price  list_price  quantity  \
0    Bookcases  FUR-BO-10001798         240         260         2
1       Chairs  FUR-CH-10000454         600         730         3
```

```
2       Labels  OFF-LA-10000240         10          10          2
3       Tables  FUR-TA-10000577        780         960          5
4      Storage  OFF-ST-10000760         20          20          2

   discount_percent
0                 2
1                 3
2                 5
3                 2
4                 5
```

1.3.1 Create a new column of discount, sale_price, and profit

[24]:
```
#Since we already have list_price and discount percent, we will calculate␣
↪discount amount and create a new column called 'discount'
df['discount'] = df['list_price']*df['discount_percent']*0.01 #discount
df.head(20)

#Now we have to calculate sale_price
df['sale_price'] = df['list_price']-df['discount'] #sale_price
df.head()

#Now we have to calulate the profit
df['profit'] = df['sale_price'] - df['cost_price'] #Profit
df.head()
```

[24]:
```
   order_id  order_date       ship_mode    segment        country  \
0         1  2023-03-01    Second Class   Consumer  United States
1         2  2023-08-15    Second Class   Consumer  United States
2         3  2023-01-10    Second Class  Corporate  United States
3         4  2022-06-18  Standard Class   Consumer  United States
4         5  2022-07-13  Standard Class   Consumer  United States

              city       state  postal_code region         category  \
0        Henderson    Kentucky        42420  South        Furniture
1        Henderson    Kentucky        42420  South        Furniture
2      Los Angeles  California        90036   West  Office Supplies
3  Fort Lauderdale     Florida        33311  South        Furniture
4  Fort Lauderdale     Florida        33311  South  Office Supplies

  sub_category       product_id  cost_price  list_price  quantity  \
0    Bookcases  FUR-BO-10001798         240         260         2
1       Chairs  FUR-CH-10000454         600         730         3
2       Labels  OFF-LA-10000240          10          10         2
3       Tables  FUR-TA-10000577         780         960         5
4      Storage  OFF-ST-10000760          20          20         2
```

```
   discount_percent  discount  sale_price  profit
0                 2       5.2       254.8    14.8
1                 3      21.9       708.1   108.1
2                 5       0.5         9.5    -0.5
3                 2      19.2       940.8   160.8
4                 5       1.0        19.0    -1.0
```

1.3.2 convert order_date in datetime format

[28]:
```
#current data type of order_date is 'object'
df['order_date']=pd.to_datetime(df['order_date'],format="%Y-%m-%d") #We have␣
 ↪entered capital Y because its 4 numbers
df.dtypes
```

[28]:
```
order_id                     int64
order_date          datetime64[ns]
ship_mode                   object
segment                     object
country                     object
city                        object
state                       object
postal_code                  int64
region                      object
category                    object
sub_category                object
product_id                  object
cost_price                   int64
list_price                   int64
quantity                     int64
discount_percent             int64
discount                   float64
sale_price                 float64
profit                     float64
dtype: object
```

1.3.3 Drop cost_price, list_price, and discount_percent columns as we don't need them now

[34]:
```
df.drop(columns=['cost_price','list_price','discount_percent'])
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[34], line 1
----> 1 df.drop(columns=['cost_price','list_price','discount_percent'])
      2 df.head
```

```
File c:
↪\users\singh\appdata\local\programs\python\python39\lib\site-packages\pandas\util\_decorat
↪py:311, in deprecate_nonkeyword_arguments.<locals>.decorate.<locals>.
↪wrapper(*args, **kwargs)
    305 if len(args) > num_allow_args:
    306     warnings.warn(
    307         msg.format(arguments=arguments),
    308         FutureWarning,
    309         stacklevel=stacklevel,
    310     )
--> 311 return func(*args, **kwargs)

File c:
↪\users\singh\appdata\local\programs\python\python39\lib\site-packages\pandas\core\frame.
↪py:4954, in DataFrame.drop(self, labels, axis, index, columns, level, inplace ␣
↪errors)
   4806 @deprecate_nonkeyword_arguments(version=None, allowed_args=["self",␣
↪"labels"])
   4807 def drop(
   4808     self,
   (…)
   4815     errors: str = "raise",
   4816 ):
   4817     """
   4818     Drop specified labels from rows or columns.
   4819
   (…)
   4952             weight  1.0     0.8
   4953     """
-> 4954     return super().drop(
   4955         labels=labels,
   4956         axis=axis,
   4957         index=index,
   4958         columns=columns,
   4959         level=level,
   4960         inplace=inplace,
   4961         errors=errors,
   4962     )

File c:
↪\users\singh\appdata\local\programs\python\python39\lib\site-packages\pandas\core\generic.
↪py:4267, in NDFrame.drop(self, labels, axis, index, columns, level, inplace,␣
↪errors)
   4265 for axis, labels in axes.items():
   4266     if labels is not None:
-> 4267         obj = obj._drop_axis(labels, axis, level=level, errors=errors)
   4269 if inplace:
   4270     self._update_inplace(obj)
```

```
File c:
↪\users\singh\appdata\local\programs\python\python39\lib\site-packages\pandas\core\generic.
↪py:4311, in NDFrame._drop_axis(self, labels, axis, level, errors, consolidate ␣
↪only_slice)
   4309         new_axis = axis.drop(labels, level=level, errors=errors)
   4310     else:
-> 4311         new_axis = axis.drop(labels, errors=errors)
   4312     indexer = axis.get_indexer(new_axis)
   4314 # Case for non-unique axis
   4315 else:

File c:
↪\users\singh\appdata\local\programs\python\python39\lib\site-packages\pandas\core\indexes\
↪py:6644, in Index.drop(self, labels, errors)
   6642 if mask.any():
   6643     if errors != "ignore":
-> 6644         raise KeyError(f"{list(labels[mask])} not found in axis")
   6645     indexer = indexer[~mask]
   6646 return self.delete(indexer)

KeyError: "['cost_price', 'list_price', 'discount_percent'] not found in axis"
```

[35]:
```
df
```

[35]:
```
      order_id order_date       ship_mode    segment        country  \
0            1 2023-03-01    Second Class   Consumer  United States
1            2 2023-08-15    Second Class   Consumer  United States
2            3 2023-01-10    Second Class  Corporate  United States
3            4 2022-06-18  Standard Class   Consumer  United States
4            5 2022-07-13  Standard Class   Consumer  United States
…          …          …             …          …              …
9989      9990 2023-02-18    Second Class   Consumer  United States
9990      9991 2023-03-17  Standard Class   Consumer  United States
9991      9992 2022-08-07  Standard Class   Consumer  United States
9992      9993 2022-11-19  Standard Class   Consumer  United States
9993      9994 2022-07-17    Second Class   Consumer  United States

                 city       state  postal_code region         category  \
0           Henderson    Kentucky        42420  South        Furniture
1           Henderson    Kentucky        42420  South        Furniture
2         Los Angeles  California        90036   West  Office Supplies
3     Fort Lauderdale     Florida        33311  South        Furniture
4     Fort Lauderdale     Florida        33311  South  Office Supplies
…                 …           …          …      …              …
9989            Miami     Florida        33180  South        Furniture
9990       Costa Mesa  California        92627   West        Furniture
9991       Costa Mesa  California        92627   West       Technology
```

```
9992       Costa Mesa  California        92627   West  Office Supplies
9993      Westminster  California        92683   West  Office Supplies

     sub_category       product_id  quantity  discount  sale_price  profit
0       Bookcases  FUR-BO-10001798         2       5.2       254.8    14.8
1          Chairs  FUR-CH-10000454         3      21.9       708.1   108.1
2          Labels  OFF-LA-10000240         2       0.5         9.5    -0.5
3          Tables  FUR-TA-10000577         5      19.2       940.8   160.8
4         Storage  OFF-ST-10000760         2       1.0        19.0    -1.0
…             …               …       …         …         …      …
9989  Furnishings  FUR-FU-10001889         3       1.2        28.8    -1.2
9990  Furnishings  FUR-FU-10000747         2       3.6        86.4    16.4
9991       Phones  TEC-PH-10003645         2       5.2       254.8    34.8
9992        Paper  OFF-PA-10004041         4       0.9        29.1    -0.9
9993   Appliances  OFF-AP-10002684         2       7.2       232.8    22.8

[9994 rows x 16 columns]
```

1.3.4 Now we are done with the cleaning above, changed data type of date, dropped unnecessary columns, and done with calculations

1.3.5 Next we will load the data in SQL Server

[36]:
```
pip install pymysql sqlalchemy pandas
```

```
Collecting pymysqlNote: you may need to restart the kernel to use updated
packages.


[notice] A new release of pip is available: 25.0 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip


  Downloading PyMySQL-1.1.1-py3-none-any.whl.metadata (4.4 kB)
Collecting sqlalchemy
  Downloading sqlalchemy-2.0.40-cp39-cp39-win_amd64.whl.metadata (9.9 kB)
Requirement already satisfied: pandas in
c:\users\singh\appdata\local\programs\python\python39\lib\site-packages (1.4.3)
Collecting greenlet>=1 (from sqlalchemy)
  Downloading greenlet-3.1.1-cp39-cp39-win_amd64.whl.metadata (3.9 kB)
Requirement already satisfied: typing-extensions>=4.6.0 in
c:\users\singh\appdata\local\programs\python\python39\lib\site-packages (from
sqlalchemy) (4.12.2)
Requirement already satisfied: python-dateutil>=2.8.1 in
c:\users\singh\appdata\local\programs\python\python39\lib\site-packages (from
pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
c:\users\singh\appdata\local\programs\python\python39\lib\site-packages (from
pandas) (2022.2.1)
```

```
Requirement already satisfied: numpy>=1.18.5 in
c:\users\singh\appdata\local\programs\python\python39\lib\site-packages (from
pandas) (1.23.2)
Requirement already satisfied: six>=1.5 in
c:\users\singh\appdata\local\programs\python\python39\lib\site-packages (from
python-dateutil>=2.8.1->pandas) (1.16.0)
Downloading PyMySQL-1.1.1-py3-none-any.whl (44 kB)
Downloading sqlalchemy-2.0.40-cp39-cp39-win_amd64.whl (2.1 MB)
   ---------------------------------------- 0.0/2.1 MB ? eta -:--:--
   ------------------- -------------------- 1.0/2.1 MB 5.6 MB/s eta 0:00:01
   ----------------------------- ---------- 1.6/2.1 MB 4.2 MB/s eta 0:00:01
   ---------------------------------------- 2.1/2.1 MB 4.1 MB/s eta 0:00:00
Downloading greenlet-3.1.1-cp39-cp39-win_amd64.whl (298 kB)
Installing collected packages: pymysql, greenlet, sqlalchemy
Successfully installed greenlet-3.1.1 pymysql-1.1.1 sqlalchemy-2.0.40
```

[60]:

```
from urllib.parse import quote_plus
from sqlalchemy import create_engine

# Replace with your details
USER = "singhpravesh882"  # Default MySQL root user
PASSWORD = "pravesh882"
HOST = "34.55.16.96"  # Find this in Cloud SQL instance details
DATABASE = "SQL_PYTHON_PROJECT"
connection_string = "mysql+pymysql://{}:{}@{}/{}".format("singhpravesh882",␣
 ↪"pravesh882", "34.55.16.96", "SQL_PYTHON_PROJECT")

print("Connection String:", connection_string)  # Debugging step

#  Create the SQLAlchemy Engine
engine = create_engine(connection_string)

print(df)
```

```
Connection String:
mysql+pymysql://singhpravesh882:pravesh882@34.55.16.96/SQL_PYTHON_PROJECT
      order_id order_date       ship_mode    segment        country  \
0            1 2023-03-01    Second Class   Consumer  United States
1            2 2023-08-15    Second Class   Consumer  United States
2            3 2023-01-10    Second Class  Corporate  United States
3            4 2022-06-18  Standard Class   Consumer  United States
4            5 2022-07-13  Standard Class   Consumer  United States
…          …          …               …          …              …
9989      9990 2023-02-18    Second Class   Consumer  United States
9990      9991 2023-03-17  Standard Class   Consumer  United States
9991      9992 2022-08-07  Standard Class   Consumer  United States
9992      9993 2022-11-19  Standard Class   Consumer  United States
9993      9994 2022-07-17    Second Class   Consumer  United States
```

```
                 city       state  postal_code region         category  \
0           Henderson    Kentucky        42420  South        Furniture
1           Henderson    Kentucky        42420  South        Furniture
2         Los Angeles  California        90036   West  Office Supplies
3     Fort Lauderdale     Florida        33311  South        Furniture
4     Fort Lauderdale     Florida        33311  South  Office Supplies
…                 …           …           …      …                …
9989            Miami     Florida        33180  South        Furniture
9990       Costa Mesa  California        92627   West        Furniture
9991       Costa Mesa  California        92627   West       Technology
9992       Costa Mesa  California        92627   West  Office Supplies
9993      Westminster  California        92683   West  Office Supplies

     sub_category       product_id  quantity  discount  sale_price  profit
0       Bookcases  FUR-BO-10001798         2       5.2       254.8    14.8
1          Chairs  FUR-CH-10000454         3      21.9       708.1   108.1
2          Labels  OFF-LA-10000240         2       0.5         9.5    -0.5
3          Tables  FUR-TA-10000577         5      19.2       940.8   160.8
4         Storage  OFF-ST-10000760         2       1.0        19.0    -1.0
…             …              …         …       …          …      …
9989  Furnishings  FUR-FU-10001889         3       1.2        28.8    -1.2
9990  Furnishings  FUR-FU-10000747         2       3.6        86.4    16.4
9991       Phones  TEC-PH-10003645         2       5.2       254.8    34.8
9992        Paper  OFF-PA-10004041         4       0.9        29.1    -0.9
9993   Appliances  OFF-AP-10002684         2       7.2       232.8    22.8

[9994 rows x 16 columns]
```

[42]:
```
import sys
print(sys.version)
```

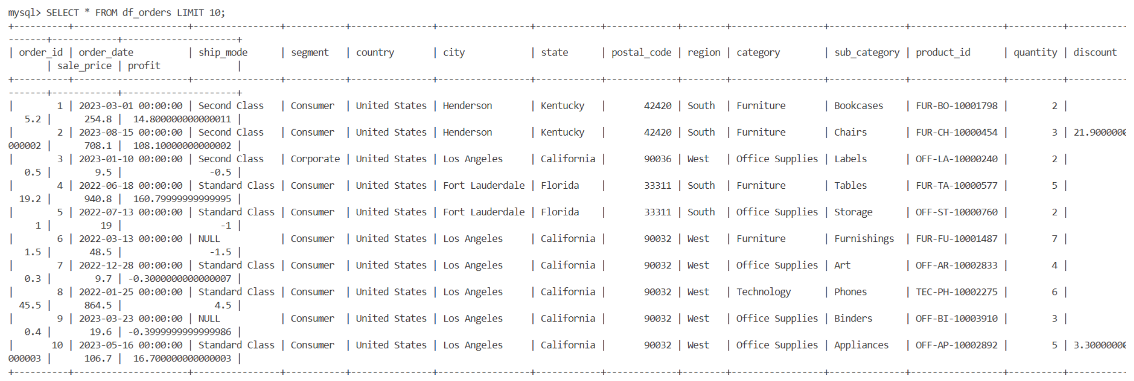
```
3.9.0 (tags/v3.9.0:9cf6752, Oct  5 2020, 15:34:40) [MSC v.1927 64 bit (AMD64)]
```

[62]:
```
df.to_sql('df_orders', con= connection_string, index=False, if_exists='replace')
```

[62]:
```
9994
```

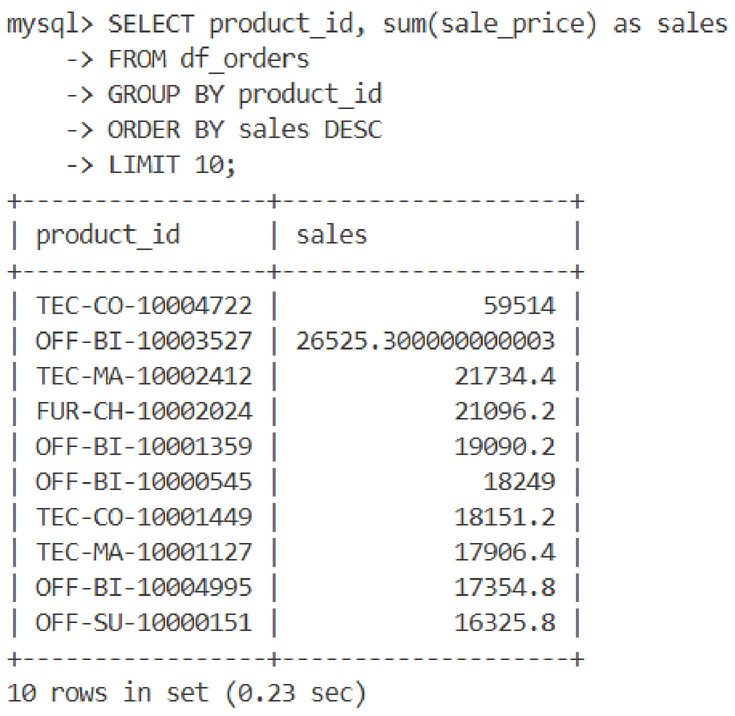
1.4 The connection to Google Cloud SQl was Successful and below are the queries that were performed for df_orders table

1.4.1 1. View top 10 rows of df_orders



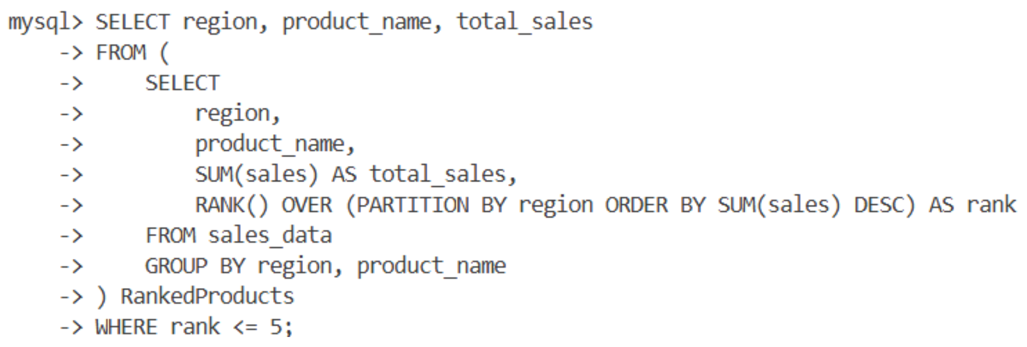
```
mysql> SELECT * FROM df_orders LIMIT 10;
+----------+---------------------+----------------+-----------+---------------+-----------------+------------+-------------+--------+-----------------+--------------+-----------------+----------+----------
-------+-----------+--------------------+
| order_id | order_date          | ship_mode      | segment   | country       | city            | state      | postal_code | region | category        | sub_category | product_id      | quantity | discount
       | sale_price | profit             |
+----------+---------------------+----------------+-----------+---------------+-----------------+------------+-------------+--------+-----------------+--------------+-----------------+----------+----------
-------+-----------+--------------------+
|        1 | 2023-03-01 00:00:00 | Second Class   | Consumer  | United States | Henderson       | Kentucky   |       42420 | South  | Furniture       | Bookcases    | FUR-BO-10001798 |        2 |
   5.2 |      254.8 |  14.800000000000011 |
|        2 | 2023-08-15 00:00:00 | Second Class   | Consumer  | United States | Henderson       | Kentucky   |       42420 | South  | Furniture       | Chairs       | FUR-CH-10000454 |        3 | 21.9000000
000002 |      708.1 |  108.10000000000002 |
|        3 | 2023-01-10 00:00:00 | Second Class   | Corporate | United States | Los Angeles     | California |       90036 | West   | Office Supplies | Labels       | OFF-LA-10000240 |        2 |
   0.5 |        9.5 |                -0.5 |
|        4 | 2022-06-18 00:00:00 | Standard Class | Consumer  | United States | Fort Lauderdale | Florida    |       33311 | South  | Furniture       | Tables       | FUR-TA-10000577 |        5 |
  19.2 |      940.8 |  160.79999999999995 |
|        5 | 2022-07-13 00:00:00 | Standard Class | Consumer  | United States | Fort Lauderdale | Florida    |       33311 | South  | Office Supplies | Storage      | OFF-ST-10000760 |        2 |
     1 |         19 |                  -1 |
|        6 | 2022-03-13 00:00:00 | NULL           | Consumer  | United States | Los Angeles     | California |       90032 | West   | Furniture       | Furnishings  | FUR-FU-10001487 |        7 |
   1.5 |       48.5 |                -1.5 |
|        7 | 2022-12-28 00:00:00 | Standard Class | Consumer  | United States | Los Angeles     | California |       90032 | West   | Office Supplies | Art          | OFF-AR-10002833 |        4 |
   0.3 |        9.7 | -0.3000000000000007 |
|        8 | 2022-01-25 00:00:00 | Standard Class | Consumer  | United States | Los Angeles     | California |       90032 | West   | Technology      | Phones       | TEC-PH-10002275 |        6 |
  45.5 |      864.5 |                 4.5 |
|        9 | 2023-03-23 00:00:00 | NULL           | Consumer  | United States | Los Angeles     | California |       90032 | West   | Office Supplies | Binders      | OFF-BI-10003910 |        3 |
   0.4 |       19.6 | -0.3999999999999986 |
|       10 | 2023-05-16 00:00:00 | Standard Class | Consumer  | United States | Los Angeles     | California |       90032 | West   | Office Supplies | Appliances   | OFF-AP-10002892 |        5 | 3.3000000
000003 |      106.7 |  16.700000000000003 |
+----------+---------------------+----------------+-----------+---------------+-----------------+------------+-------------+--------+-----------------+--------------+-----------------+----------+----------
```

1.4.2 2. Select top 10 products by revenue



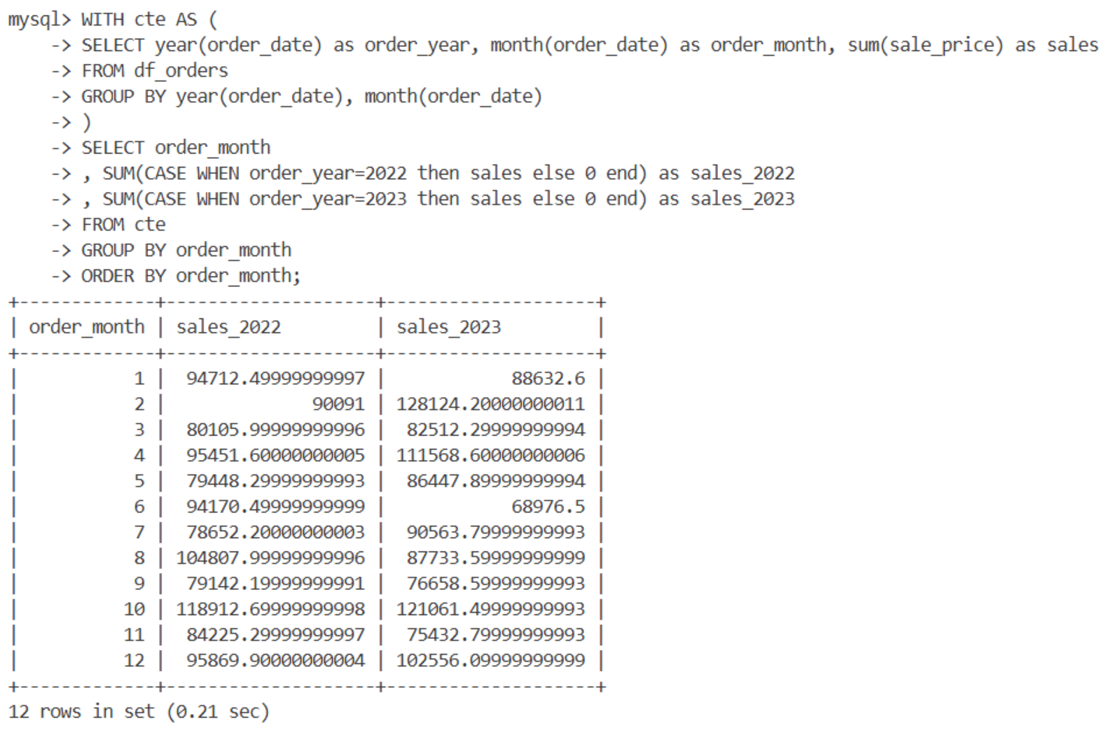
```
mysql> SELECT product_id, sum(sale_price) as sales
    -> FROM df_orders
    -> GROUP BY product_id
    -> ORDER BY sales DESC
    -> LIMIT 10;
+-----------------+--------------------+
| product_id      | sales              |
+-----------------+--------------------+
| TEC-CO-10004722 |              59514 |
| OFF-BI-10003527 | 26525.300000000003 |
| TEC-MA-10002412 |            21734.4 |
| FUR-CH-10002024 |            21096.2 |
| OFF-BI-10001359 |            19090.2 |
| OFF-BI-10000545 |              18249 |
| TEC-CO-10001449 |            18151.2 |
| TEC-MA-10001127 |            17906.4 |
| OFF-BI-10004995 |            17354.8 |
| OFF-SU-10000151 |            16325.8 |
+-----------------+--------------------+
10 rows in set (0.23 sec)
```

1.4.3 3. Find top 5 highest selling products in each region



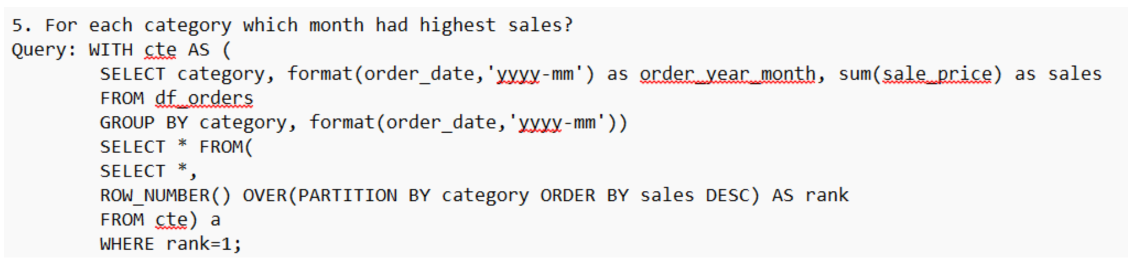
```
mysql> SELECT region, product_name, total_sales
    -> FROM (
    ->     SELECT
    ->         region,
    ->         product_name,
    ->         SUM(sales) AS total_sales,
    ->         RANK() OVER (PARTITION BY region ORDER BY SUM(sales) DESC) AS rank
    ->     FROM sales_data
    ->     GROUP BY region, product_name
    -> ) RankedProducts
    -> WHERE rank <= 5;
```

1.4.4 4. Find month over month growth comparison for 2022 and 2023 sales (Example: Jan2022 vs Jan2023)



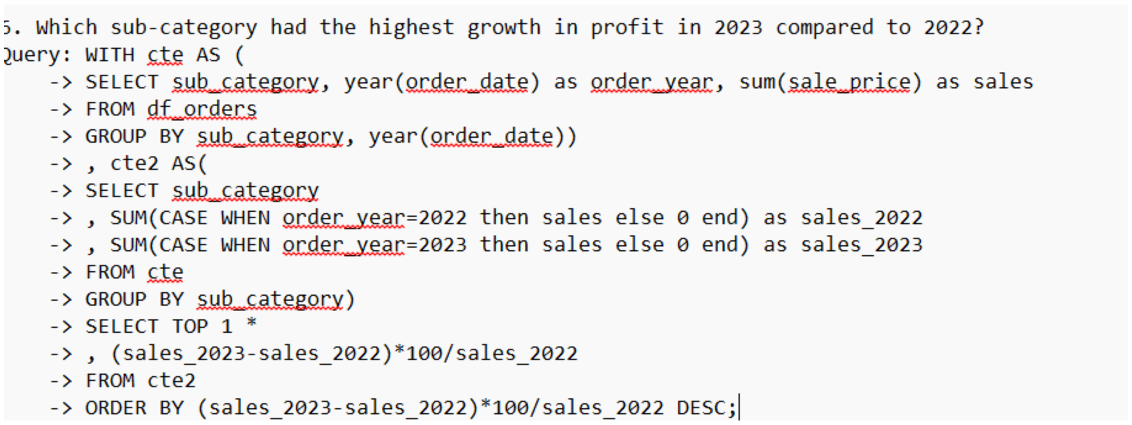
```
mysql> WITH cte AS (
    -> SELECT year(order_date) as order_year, month(order_date) as order_month, sum(sale_price) as sales
    -> FROM df_orders
    -> GROUP BY year(order_date), month(order_date)
    -> )
    -> SELECT order_month
    -> , SUM(CASE WHEN order_year=2022 then sales else 0 end) as sales_2022
    -> , SUM(CASE WHEN order_year=2023 then sales else 0 end) as sales_2023
    -> FROM cte
    -> GROUP BY order_month
    -> ORDER BY order_month;
+-------------+-------------------+-------------------+
| order_month | sales_2022        | sales_2023        |
+-------------+-------------------+-------------------+
|           1 |  94712.49999999997 |          88632.6 |
|           2 |             90091 | 128124.20000000011 |
|           3 |  80105.99999999996 |  82512.29999999994 |
|           4 |  95451.60000000005 | 111568.60000000006 |
|           5 |  79448.29999999993 |  86447.89999999994 |
|           6 |  94170.49999999999 |           68976.5 |
|           7 |  78652.20000000003 |  90563.79999999993 |
|           8 | 104807.99999999996 |  87733.59999999999 |
|           9 |  79142.19999999991 |  76658.59999999993 |
|          10 | 118912.69999999998 | 121061.49999999993 |
|          11 |  84225.29999999997 |  75432.79999999993 |
|          12 |  95869.90000000004 | 102556.09999999999 |
+-------------+-------------------+-------------------+
12 rows in set (0.21 sec)
```

1.4.5 5. For each category which month had highest sales?



```
5. For each category which month had highest sales?
Query: WITH cte AS (
        SELECT category, format(order_date,'yyyy-mm') as order_year_month, sum(sale_price) as sales
        FROM df_orders
        GROUP BY category, format(order_date,'yyyy-mm'))
        SELECT * FROM(
        SELECT *,
        ROW_NUMBER() OVER(PARTITION BY category ORDER BY sales DESC) AS rank
        FROM cte) a
        WHERE rank=1;
```

1.4.6 6. Which sub-category had the highest growth in profit in 2023 compared to 2022?



```
5. Which sub-category had the highest growth in profit in 2023 compared to 2022?
Query: WITH cte AS (
    -> SELECT sub_category, year(order_date) as order_year, sum(sale_price) as sales
    -> FROM df_orders
    -> GROUP BY sub_category, year(order_date))
    -> , cte2 AS(
    -> SELECT sub_category
    -> , SUM(CASE WHEN order_year=2022 then sales else 0 end) as sales_2022
    -> , SUM(CASE WHEN order_year=2023 then sales else 0 end) as sales_2023
    -> FROM cte
    -> GROUP BY sub_category)
    -> SELECT TOP 1 *
    -> , (sales_2023-sales_2022)*100/sales_2022
    -> FROM cte2
    -> ORDER BY (sales_2023-sales_2022)*100/sales_2022 DESC;
```

```
[ ]:
```