Import thư viện

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import random
```

1. Đọc file dữ liệu

In [2]:
```
df = pd.read_csv('../data/orginal_sales_data_edit.csv')
```

3. Hiển thị 5 dòng đầu tiên

In [3]:
```
df.head(5)
```

Out[3]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
0	10107	30	95.70	2	2871.00	2/24/20(
1	10121	34	81.35	5	2765.90	5/7/20(
2	10134	41	94.74	2	3884.34	7/1/20(
3	10145	45	83.26	6	3746.70	8/25/20(
4	10159	49	100.00	14	5205.27	10/10/20(

5 rows × 28 columns

4. Hiển thị 5 dòng cuối cùng

In [4]:
```
df.tail(5)
```

Out[4]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERI
2818	10350	20	100.00	15	2244.40	12/2
2819	10373	29	100.00	1	3978.51	1/31
2820	10386	43	100.00	4	5417.57	3/1
2821	10397	34	62.24	1	2116.16	3/28
2822	10414	47	65.52	9	3079.44	5/6

5 rows × 28 columns

5. Chuyển kiểu dữ liệu cho 1 cột nào đó

In [5]:
```
# Chuyển đổi dữ liệu từ int64 sang int32

df["YEAR_ID"] = df["YEAR_ID"].astype("int32")
```

In [6]:
```
# Kiểm tra kiểu dữ liệu của cột YEAR_ID

df["YEAR_ID"].dtype
```

Out[6]: dtype('int32')

6. Xem chiều dài của DataFram df, tương đương shape[0]

In [7]:
```
# Trả về kết quả là số dòng trong DataFrame df

print('Len: ', len(df))
```

```
Len:  2823
```

7. Xem thông tin DataFrame vừa đọc được

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2823 entries, 0 to 2822
Data columns (total 28 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   ORDERNUMBER      2823 non-null   int64
 1   QUANTITYORDERED  2823 non-null   int64
 2   PRICEEACH        2823 non-null   float64
 3   ORDERLINENUMBER  2823 non-null   int64
 4   SALES            2823 non-null   float64
 5   ORDERDATE        2823 non-null   object
 6   STATUS           2823 non-null   object
 7   QTR_ID           2823 non-null   int64
 8   MONTH_ID         2823 non-null   int64
 9   YEAR_ID          2823 non-null   int32
 10  PRODUCTLINE      2823 non-null   object
 11  MSRP             2823 non-null   int64
 12  PRODUCTCODE      2823 non-null   object
 13  CATEGORY         2823 non-null   object
 14  SUBCATEGORY      2823 non-null   object
 15  CUSTOMERNAME     2823 non-null   object
 16  PHONE            2823 non-null   object
 17  ADDRESSLINE1     2823 non-null   object
 18  ADDRESSLINE2     302 non-null    object
 19  CITY             2823 non-null   object
 20  STATE            1337 non-null   object
 21  POSTALCODE       2747 non-null   object
 22  COUNTRY          2823 non-null   object
 23  TERRITORY        1749 non-null   object
 24  CONTACTLASTNAME  2823 non-null   object
 25  CONTACTFIRSTNAME 2823 non-null   object
 26  DEALSIZE         2823 non-null   object
 27  PAYMENTFULLNAME  2823 non-null   object
dtypes: float64(2), int32(1), int64(6), object(19)
memory usage: 606.6+ KB
```

8. Xem kích thước của DataFrame

In [9]:
```
# Trả về kết quả 1 tuple chứa thông tin về số hàng và số cột trong DataFrame đó
# Kết quả là: DataFrame có 2823 dòng và 28 cột

print('Shape: ', df.shape)
```

```
Shape:  (2823, 28)
```

9. Hiển thị dữ liệu của cột thứ 10

In [10]:
```
# Trả về kết quả là dạng DataFrame của cột PRODUCTLINE

df[['PRODUCTLINE']]
```

Out[10]:

	PRODUCTLINE
0	Motorcycles
1	Motorcycles
2	Motorcycles
3	Motorcycles
4	Motorcycles
...	...
2818	Ships
2819	Ships
2820	Ships
2821	Ships
2822	Ships

2823 rows × 1 columns

10. Hiển thị dữ liệu của cột 1,2,3,5,6

In [11]:
```
# Gộp cột trả về dạng DataFrame

df[['ORDERNUMBER', 'QUANTITYORDERED', 'PRICEEACH', 'PRODUCTLINE']]
```

Out[11]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	PRODUCTLINE
0	10107	30	95.70	Motorcycles
1	10121	34	81.35	Motorcycles
2	10134	41	94.74	Motorcycles
3	10145	45	83.26	Motorcycles
4	10159	49	100.00	Motorcycles
...	...	...	...	...
2818	10350	20	100.00	Ships
2819	10373	29	100.00	Ships
2820	10386	43	100.00	Ships
2821	10397	34	62.24	Ships
2822	10414	47	65.52	Ships

2823 rows × 4 columns

11. Hiển thị 5 dòng dữ liệu đầu tiên gồm các cột 1,2,3,5,6

In [12]:
```
df[['ORDERNUMBER', 'QUANTITYORDERED', 'PRICEEACH', 'PRODUCTLINE']].head(5)
```

Out[12]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	PRODUCTLINE
0	10107	30	95.70	Motorcycles
1	10121	34	81.35	Motorcycles
2	10134	41	94.74	Motorcycles
3	10145	45	83.26	Motorcycles
4	10159	49	100.00	Motorcycles

12. Hiển thị 5 dòng dữ liệu đầu tiên theo chỉ số

In [13]:
```
df[0:5]
```

Out[13]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
0	10107	30	95.70	2	2871.00	2/24/20(
1	10121	34	81.35	5	2765.90	5/7/20(
2	10134	41	94.74	2	3884.34	7/1/20(
3	10145	45	83.26	6	3746.70	8/25/20(
4	10159	49	100.00	14	5205.27	10/10/20(

5 rows × 28 columns

13. Hiển thị 5 dòng dữ liệu đầu tiên theo chỉ số gồm các cột 1,2,3,5,6

In [14]:
```
df[['ORDERNUMBER', 'QUANTITYORDERED', 'PRICEEACH', 'PRODUCTLINE']][:5]
```

Out[14]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	PRODUCTLINE
0	10107	30	95.70	Motorcycles
1	10121	34	81.35	Motorcycles
2	10134	41	94.74	Motorcycles
3	10145	45	83.26	Motorcycles
4	10159	49	100.00	Motorcycles

14. Loại bỏ các dòng trùng nhau

In [15]:
```
df.drop_duplicates(inplace=True)
```

In [16]:
```
# Kiểm tra df sau khi loại bỏ các dòng trùng nhau

df
```

Out[16]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERI
0	10107	30	95.70	2	2871.00	2/24
1	10121	34	81.35	5	2765.90	5/7
2	10134	41	94.74	2	3884.34	7/1
3	10145	45	83.26	6	3746.70	8/25
4	10159	49	100.00	14	5205.27	10/10
...	...	...	...	...	...	
2818	10350	20	100.00	15	2244.40	12/2
2819	10373	29	100.00	1	3978.51	1/31
2820	10386	43	100.00	4	5417.57	3/1
2821	10397	34	62.24	1	2116.16	3/28
2822	10414	47	65.52	9	3079.44	5/6

2823 rows × 28 columns

15. Loại bỏ các dòng trống có dữ liệu của 1 cột là trống, có 2823 dòng

In [17]:
```
# Tính số lượng các dòng có giá trị NaN trong cột "ADDRESSLINE2"
# Kết quả là: có 2521 dòng

df["ADDRESSLINE2"].isna().sum()
```

Out[17]: 2521

16. Loại bỏ các dòng không biết của 1 cột có giá trị là Unknown, có 2521 dòng

In [18]:
```
# Thay thế tất cả các dòng có giá trị NaN trong cột "ADDRESSLINE2" bằng chuỗi 'Unknown'

df["ADDRESSLINE2"].fillna('Unknown', inplace=True)
```

In [19]:
```
# Kiểm tra dữ liệu cột ADDRESSLINE2 có giá trị NaN không ?
# Kết quả là: có 0 dòng

df["ADDRESSLINE2"].isna().sum()
```

Out[19]: 0

17. Lấy dữ liệu theo cột dạng chuỗi

In [20]:
```
df['QUANTITYORDERED']
```

Out[20]:
```
0       30
1       34
2       41
3       45
4       49
       ..
2818    20
2819    29
2820    43
2821    34
2822    47
Name: QUANTITYORDERED, Length: 2823, dtype: int64
```

18. Lấy dữ liệu về 1 mảng

In [21]:
```
df['QUANTITYORDERED'].values
```

Out[21]:
```
array([30, 34, 41, ..., 43, 34, 47], dtype=int64)
```

20. Lấy dữ liệu từ dòng số 4 đến dòng số 9

In [22]:
```
df[4:10]
```

Out[22]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
4	10159	49	100.00	14	5205.27	10/10/20(
5	10168	36	96.66	1	3479.76	10/28/20(
6	10180	29	86.13	9	2497.77	11/11/20(
7	10188	48	100.00	1	5512.32	11/18/20(
8	10201	22	98.57	2	2168.54	12/1/20(
9	10211	41	100.00	14	4708.44	1/15/20(

6 rows × 28 columns

21. Đọc dữ liệu từ dòng số 4 đến dòng số 9

In [23]:
```
# KQ trả về là dữ liệu của dòng số 4 đến dòng số 10 (bao gồm cả dòng 10)

df.loc[4:10]
```

Out[23]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDA
4	10159	49	100.00	14	5205.27	10/10/2(
5	10168	36	96.66	1	3479.76	10/28/2(
6	10180	29	86.13	9	2497.77	11/11/2(
7	10188	48	100.00	1	5512.32	11/18/2(
8	10201	22	98.57	2	2168.54	12/1/2(
9	10211	41	100.00	14	4708.44	1/15/2(
10	10223	37	100.00	1	3965.66	2/20/2(

7 rows × 28 columns

In [24]:
```
# KQ trả về là dữ liệu của dòng số 4 đến dòng số 9 (KHÔNG bao gồm cả dòng 10)

df.iloc[4:10]
```

Out[24]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
4	10159	49	100.00	14	5205.27	10/10/20(
5	10168	36	96.66	1	3479.76	10/28/20(
6	10180	29	86.13	9	2497.77	11/11/20(
7	10188	48	100.00	1	5512.32	11/18/20(
8	10201	22	98.57	2	2168.54	12/1/20(
9	10211	41	100.00	14	4708.44	1/15/20(

6 rows × 28 columns

22. Lấy thông tin tại dòng có chỉ số là 2

In [25]:
```
df.loc[2]
```

Out[25]:
```
ORDERNUMBER                10134
QUANTITYORDERED               41
PRICEEACH                  94.74
ORDERLINENUMBER                2
SALES                    3884.34
ORDERDATE                7/1/2003
STATUS                    Shipped
QTR_ID                          3
MONTH_ID                        7
YEAR_ID                      2003
PRODUCTLINE            Motorcycles
MSRP                            95
PRODUCTCODE                S10_1678
CATEGORY                  Furniture
SUBCATEGORY                Cabinet
CUSTOMERNAME          Lyon Souveniers
PHONE                +33 1 46 62 7555
ADDRESSLINE1    27 rue du Colonel Pierre Avia
ADDRESSLINE2                Unknown
CITY                          Paris
STATE                           NaN
POSTALCODE                    75508
COUNTRY                      France
TERRITORY                      EMEA
CONTACTLASTNAME            Da Cunha
CONTACTFIRSTNAME             Daniel
DEALSIZE                     Medium
PAYMENTFULLNAME        DaCunha Daniel
Name: 2, dtype: object
```

23. Lấy thông tin từ dòng 4 đến dòng 10 của một số cột

In [26]:
```
# KQ trả về là dữ liệu từ dòng 4 đến dòng 10 của 2 cột QUANTITYORDERED và SALES

df.loc[4:10, ['QUANTITYORDERED', 'SALES']]
```

Out[26]:

	QUANTITYORDERED	SALES
4	49	5205.27
5	36	3479.76
6	29	2497.77
7	48	5512.32
8	22	2168.54
9	41	4708.44
10	37	3965.66

24. Lấy thông tin dòng 2 đến dòng 9, từ cột 4 đến cột 7

In [27]:
```
df.iloc[2:9, 4:7]
```

Out[27]:

	SALES	ORDERDATE	STATUS
2	3884.34	7/1/2003	Shipped
3	3746.70	8/25/2003	Shipped
4	5205.27	10/10/2003	Shipped
5	3479.76	10/28/2003	Shipped
6	2497.77	11/11/2003	Shipped
7	5512.32	11/18/2003	Shipped
8	2168.54	12/1/2003	Shipped

25. Lấy dữ liệu tại chỉ số (index) là 2

In [28]:
```
df.iloc[2]
```

Out[28]:
```
ORDERNUMBER                 10134
QUANTITYORDERED                41
PRICEEACH                   94.74
ORDERLINENUMBER                 2
SALES                     3884.34
ORDERDATE                 7/1/2003
STATUS                    Shipped
QTR_ID                          3
MONTH_ID                        7
YEAR_ID                      2003
PRODUCTLINE              Motorcycles
MSRP                            95
PRODUCTCODE                S10_1678
CATEGORY                  Furniture
SUBCATEGORY                  Cabinet
CUSTOMERNAME          Lyon Souveniers
PHONE                +33 1 46 62 7555
ADDRESSLINE1     27 rue du Colonel Pierre Avia
ADDRESSLINE2                 Unknown
CITY                           Paris
STATE                            NaN
POSTALCODE                     75508
COUNTRY                       France
TERRITORY                       EMEA
CONTACTLASTNAME              Da Cunha
CONTACTFIRSTNAME               Daniel
DEALSIZE                     Medium
PAYMENTFULLNAME         DaCunha Daniel
Name: 2, dtype: object
```

26. Lấy dữ liệu từ dòng đầu tiên đến dòng 9 dùng iloc

In [29]:
```
df.iloc[:10]
```

Out[29]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
0	10107	30	95.70	2	2871.00	2/24/20(
1	10121	34	81.35	5	2765.90	5/7/20(
2	10134	41	94.74	2	3884.34	7/1/20(
3	10145	45	83.26	6	3746.70	8/25/20(
4	10159	49	100.00	14	5205.27	10/10/20(
5	10168	36	96.66	1	3479.76	10/28/20(
6	10180	29	86.13	9	2497.77	11/11/20(
7	10188	48	100.00	1	5512.32	11/18/20(
8	10201	22	98.57	2	2168.54	12/1/20(
9	10211	41	100.00	14	4708.44	1/15/20(

10 rows × 28 columns

27. Lấy dữ liệu từ dòng đầu tiên đến dòng 9 gồm cột 4 đến cột 7 dùng iloc

In [30]:
```
df.iloc[2:9, 4:7]
```

Out[30]:

	SALES	ORDERDATE	STATUS
2	3884.34	7/1/2003	Shipped
3	3746.70	8/25/2003	Shipped
4	5205.27	10/10/2003	Shipped
5	3479.76	10/28/2003	Shipped
6	2497.77	11/11/2003	Shipped
7	5512.32	11/18/2003	Shipped
8	2168.54	12/1/2003	Shipped

28. Sắp xếp dữ liệu theo sales tăng dần

In [31]:
```
df.sort_values(by='SALES', ascending=True)
```

Out[31]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDEI
2249	10425	11	43.83	6	482.13	5/3
1287	10407	6	90.19	3	541.14	4/2
2044	10408	15	36.93	1	553.95	4/2
1551	10280	20	28.88	12	577.60	8/1
1818	10419	15	42.67	7	640.05	5/1
...	...	...	...	...	...	
104	10403	66	100.00	9	11886.60	4/
1062	10412	60	100.00	9	11887.80	5/
53	10424	50	100.00	6	12001.00	5/3
744	10322	50	100.00	6	12536.50	11/
598	10407	76	100.00	2	14082.80	4/2

2823 rows × 28 columns

29. Sắp xếp dữ liệu theo nhiều tiêu chí

In [32]:
```
# KQ trả về là cột QUANTITYORDERED sắp xếp tăng dần và cột SALES sắp xếp giảm dần

df.sort_values(by=['QUANTITYORDERED', 'SALES'], ascending=[True, False])
```

Out[32]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDER
751	10409	6	100.00	2	785.64	4/2
1287	10407	6	90.19	3	541.14	4/2
315	10419	10	100.00	11	1092.20	5/1
907	10423	10	88.14	1	881.40	5/3
2507	10401	11	100.00	8	1135.31	4/
...	...	...	...	...	...	
1995	10405	76	100.00	3	11739.70	4/1
1714	10407	76	94.50	6	7182.00	4/2
2689	10401	77	92.00	9	7084.00	4/
2586	10401	85	88.75	10	7543.75	4/
418	10405	97	93.28	5	9048.16	4/1

2823 rows × 28 columns

30. Lọc dữ liệu theo 1 điều kiện

In [33]:
```
df[df['SALES']>5000]
```

Out[33]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERI
4	10159	49	100.0	14	5205.27	10/10
7	10188	48	100.0	1	5512.32	11/18
20	10341	41	100.0	9	7737.93	11/24
25	10417	66	100.0	2	7516.08	5/13
26	10103	26	100.0	11	5404.62	1/29
...	...	...	...	...	...	
2685	10361	44	100.0	10	5001.92	12/17
2686	10375	44	100.0	11	5208.72	2/3
2689	10401	77	92.0	9	7084.00	4/3
2715	10397	48	100.0	3	5192.64	3/28
2820	10386	43	100.0	4	5417.57	3/1

549 rows × 28 columns

In [34]:
```
df.loc[df['SALES']>5000]
```

Out[34]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERI
4	10159	49	100.0	14	5205.27	10/10
7	10188	48	100.0	1	5512.32	11/18
20	10341	41	100.0	9	7737.93	11/24
25	10417	66	100.0	2	7516.08	5/13
26	10103	26	100.0	11	5404.62	1/29
...	...	...	...	...	...	
2685	10361	44	100.0	10	5001.92	12/17
2686	10375	44	100.0	11	5208.72	2/3
2689	10401	77	92.0	9	7084.00	4/3
2715	10397	48	100.0	3	5192.64	3/28
2820	10386	43	100.0	4	5417.57	3/1

549 rows × 28 columns

31. Lọc dữ liệu theo nhiều điều kiện

In [35]:
```
df[(df['SALES']>5000) & (df['QUANTITYORDERED']>40)]
```

Out[35]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDEI
4	10159	49	100.0	14	5205.27	10/1
7	10188	48	100.0	1	5512.32	11/1
20	10341	41	100.0	9	7737.93	11/2
25	10417	66	100.0	2	7516.08	5/1
30	10150	45	100.0	8	10993.50	9/1
...	...	...	...	...	...	
2685	10361	44	100.0	10	5001.92	12/1
2686	10375	44	100.0	11	5208.72	2/
2689	10401	77	92.0	9	7084.00	4/
2715	10397	48	100.0	3	5192.64	3/2
2820	10386	43	100.0	4	5417.57	3/

374 rows × 28 columns

32. Lọc giá trị và gán điều kiện dùng loc

In [36]:
```
# Gán nhãn FLAG là EXPENSIVE cho các giá trị >= 65 của cột PRICEEACH
# Gán nhãn FLAG là CHEAP cho các giá trị < 65 của cột PRICEEACH

df.loc[df['PRICEEACH']>=65, 'FLAG']='EXPENSIVE'
df.loc[df['PRICEEACH']<65, 'FLAG']='CHEAP'
```

```
C:\Users\Lan Anh\AppData\Local\Temp\ipykernel_9000\2734652281.py:4: FutureWarning: Setting an ite
m of incompatible dtype is deprecated and will raise in a future error of pandas. Value 'EXPENSIVE' has d
type incompatible with float64, please explicitly cast to a compatible dtype first.
  df.loc[df['PRICEEACH']>=65, 'FLAG']='EXPENSIVE'
```

In [37]:
```
# Kiểm tra cột PRICEEACH và FLAG sau khi gán

df[['PRICEEACH', 'FLAG']]
```

Out[37]:

	PRICEEACH	FLAG
0	95.70	EXPENSIVE
1	81.35	EXPENSIVE
2	94.74	EXPENSIVE
3	83.26	EXPENSIVE
4	100.00	EXPENSIVE
...	...	...
2818	100.00	EXPENSIVE
2819	100.00	EXPENSIVE
2820	100.00	EXPENSIVE
2821	62.24	CHEAP
2822	65.52	EXPENSIVE

2823 rows × 2 columns

33. Viết hàm trả về giá trị có nhiều điều kiện và áp dụng hàm giá trị trả về cho 1 cột

In [38]:
```
def foo(x):
  if x<10:
    return 'BAD'
  elif x>=10 and x<50:
    return 'GOOD'
  else:
    return 'EXCELLENT'

df['WORTH'] = df[['QUANTITYORDERED']].applymap(foo)
df[['QUANTITYORDERED', 'WORTH']]
```

```
C:\Users\Lan Anh\AppData\Local\Temp\ipykernel_9000\910735475.py:9: FutureWarning: DataFrame.ap
plymap has been deprecated. Use DataFrame.map instead.
  df['WORTH'] = df[['QUANTITYORDERED']].applymap(foo)
```

Out[38]:

	QUANTITYORDERED	WORTH
0	30	GOOD
1	34	GOOD
2	41	GOOD
3	45	GOOD
4	49	GOOD
...	...	...
2818	20	GOOD
2819	29	GOOD
2820	43	GOOD
2821	34	GOOD
2822	47	GOOD

2823 rows × 2 columns

34. Ánh xạ giá trị tới 1 cột

In [39]:
```
dict_map1 = {1: 'Qui_1', 2: 'Qui_2', 3: 'Qui_3', 4: 'Qui_4'}
df['QTR_ID'] = df['QTR_ID'].map(dict_map1)
df['QTR_ID']
```

Out[39]:
```
0       Qui_1
1       Qui_2
2       Qui_3
3       Qui_3
4       Qui_4
       ...
2818    Qui_4
2819    Qui_1
2820    Qui_1
2821    Qui_1
2822    Qui_2
Name: QTR_ID, Length: 2823, dtype: object
```

In [62]:
```
df[['QTR_ID']]
```

Out[62]:

	QTR_ID
0	Quy
1	Quy
2	Quy
3	Quy
4	Quy
...	...
2818	Quy
2819	Quy
2820	Quy
2821	Quy
2822	Quy

2823 rows × 1 columns

In [63]:
```
df.head(5)
```

Out[63]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
0	10107	30	95.70	2	2871.00	200
1	10121	34	81.35	5	2765.90	200
2	10134	41	94.74	2	3884.34	200
3	10145	45	83.26	6	3746.70	200
4	10159	49	100.00	14	5205.27	200

5 rows × 32 columns

35. Lấy những dòng dữ liệu bằng 1 điều kiện nào đó

In [41]:
```
df[['QUANTITYORDERED', 'PRICEEACH']].loc[df['YEAR_ID']==2003]
```

Out[41]:

	QUANTITYORDERED	PRICEEACH
0	30	95.70
1	34	81.35
2	41	94.74
3	45	83.26
4	49	100.00
...	...	...
2801	50	60.06
2802	38	48.59
2803	40	50.23
2804	28	64.43
2805	42	50.23

1000 rows × 2 columns

36. Hiển thị các bản ghi có số lượng hơn 25

In [42]:
```
df[df['QUANTITYORDERED']>25]
print("Hiển thị 5 dòng")
TuoiTre = df[df['QUANTITYORDERED']>25]
TuoiTre[:5]
```

Hiển thị 5 dòng

Out[42]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
0	10107	30	95.70	2	2871.00	2/24/20(
1	10121	34	81.35	5	2765.90	5/7/20(
2	10134	41	94.74	2	3884.34	7/1/20(
3	10145	45	83.26	6	3746.70	8/25/20(
4	10159	49	100.00	14	5205.27	10/10/20(

5 rows × 30 columns

37. Hiển thị các hóa đơn đã được giao

In [43]:
```
dg = df[df['STATUS'] == 'Shipped']
dg[:10]
```

Out[43]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
0	10107	30	95.70	2	2871.00	2/24/20(
1	10121	34	81.35	5	2765.90	5/7/20(
2	10134	41	94.74	2	3884.34	7/1/20(
3	10145	45	83.26	6	3746.70	8/25/20(
4	10159	49	100.00	14	5205.27	10/10/20(
5	10168	36	96.66	1	3479.76	10/28/20(
6	10180	29	86.13	9	2497.77	11/11/20(
7	10188	48	100.00	1	5512.32	11/18/20(
8	10201	22	98.57	2	2168.54	12/1/20(
9	10211	41	100.00	14	4708.44	1/15/20(

10 rows × 30 columns

38. So sánh chuỗi

In [44]:
```
sosanh = df['STATUS'].str.contains('Shipped')
sosanh.head(5)
```

Out[44]:
```
0    True
1    True
2    True
3    True
4    True
Name: STATUS, dtype: bool
```

39. Lấy giá trị trả về mảng

In [45]:
```
df['STATUS'].values
```

Out[45]:
```
array(['Shipped', 'Shipped', 'Shipped', ..., 'Resolved', 'Shipped',
       'On Hold'], dtype=object)
```

40. Thêm 1 cột vào file dữ liệu

In [46]:
```
df_len = len(df)
ngayLap = [random.randrange(2003, 2005, 1) for i in range(df_len)]
df['ORDERDATE'] = ngayLap
df.tail(5)
```

Out[46]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERI
2818	10350	20	100.00	15	2244.40	
2819	10373	29	100.00	1	3978.51	
2820	10386	43	100.00	4	5417.57	
2821	10397	34	62.24	1	2116.16	
2822	10414	47	65.52	9	3079.44	

5 rows × 30 columns

41. Thêm 1 cột (daGiao) vào dữ liệu theo tiêu chí. Nếu điều kiện thỏa thì giá trị mặc định là True, ngược lại là False

In [47]:
```
df['DAGIAO'] = df['STATUS'] == 'Shipped'
df.head(5)
```

Out[47]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
0	10107	30	95.70	2	2871.00	200
1	10121	34	81.35	5	2765.90	200
2	10134	41	94.74	2	3884.34	200
3	10145	45	83.26	6	3746.70	200
4	10159	49	100.00	14	5205.27	200

5 rows × 31 columns

42. Tạo 1 cột mới có giá trị rỗng

In [48]:
```
df['TONGTIEN'] = None
df
```

Out[48]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERI
0	10107	30	95.70	2	2871.00	
1	10121	34	81.35	5	2765.90	
2	10134	41	94.74	2	3884.34	
3	10145	45	83.26	6	3746.70	
4	10159	49	100.00	14	5205.27	
...	...	...	...	...	...	
2818	10350	20	100.00	15	2244.40	
2819	10373	29	100.00	1	3978.51	
2820	10386	43	100.00	4	5417.57	
2821	10397	34	62.24	1	2116.16	
2822	10414	47	65.52	9	3079.44	

2823 rows × 32 columns

44. Sửa giá trị của cột

In [49]:
```
df['QTR_ID'] = None
df['QTR_ID'] = 'Quy'
df.head(5)
```

Out[49]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
0	10107	30	95.70	2	2871.00	200
1	10121	34	81.35	5	2765.90	200
2	10134	41	94.74	2	3884.34	200
3	10145	45	83.26	6	3746.70	200
4	10159	49	100.00	14	5205.27	200

5 rows × 32 columns

45. Xóa cột trong DataFrame

In [50]:
```
df.drop(['TONGTIEN'], axis=1)
df.head(5)
```

Out[50]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDAT
0	10107	30	95.70	2	2871.00	20(
1	10121	34	81.35	5	2765.90	20(
2	10134	41	94.74	2	3884.34	20(
3	10145	45	83.26	6	3746.70	20(
4	10159	49	100.00	14	5205.27	20(

5 rows × 32 columns

46. Xóa bản ghi theo chỉ số

In [51]:
```
# Xóa bản ghi theo chỉ số 1 và 2

df.drop([0,1])
```

Out[51]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERI
2	10134	41	94.74	2	3884.34	
3	10145	45	83.26	6	3746.70	
4	10159	49	100.00	14	5205.27	
5	10168	36	96.66	1	3479.76	
6	10180	29	86.13	9	2497.77	
...	...	...	...	...	...	
2818	10350	20	100.00	15	2244.40	
2819	10373	29	100.00	1	3978.51	
2820	10386	43	100.00	4	5417.57	
2821	10397	34	62.24	1	2116.16	
2822	10414	47	65.52	9	3079.44	

2821 rows × 32 columns

47. Sử dụng hàm describe() để thống kê dữ liệu

In [52]:
```
df.describe()
```

Out[52]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	C
count	2823.000000	2823.000000	2823.000000	2823.000000	2823.000000	
mean	10258.725115	35.092809	83.658544	6.466171	3553.889072	
std	92.085478	9.741443	20.174277	4.225841	1841.865106	
min	10100.000000	6.000000	26.880000	1.000000	482.130000	
25%	10180.000000	27.000000	68.860000	3.000000	2203.430000	
50%	10262.000000	35.000000	95.700000	6.000000	3184.800000	
75%	10333.500000	43.000000	100.000000	9.000000	4508.000000	
max	10425.000000	97.000000	100.000000	18.000000	14082.800000	

48. Xem thống kê trên từng cột

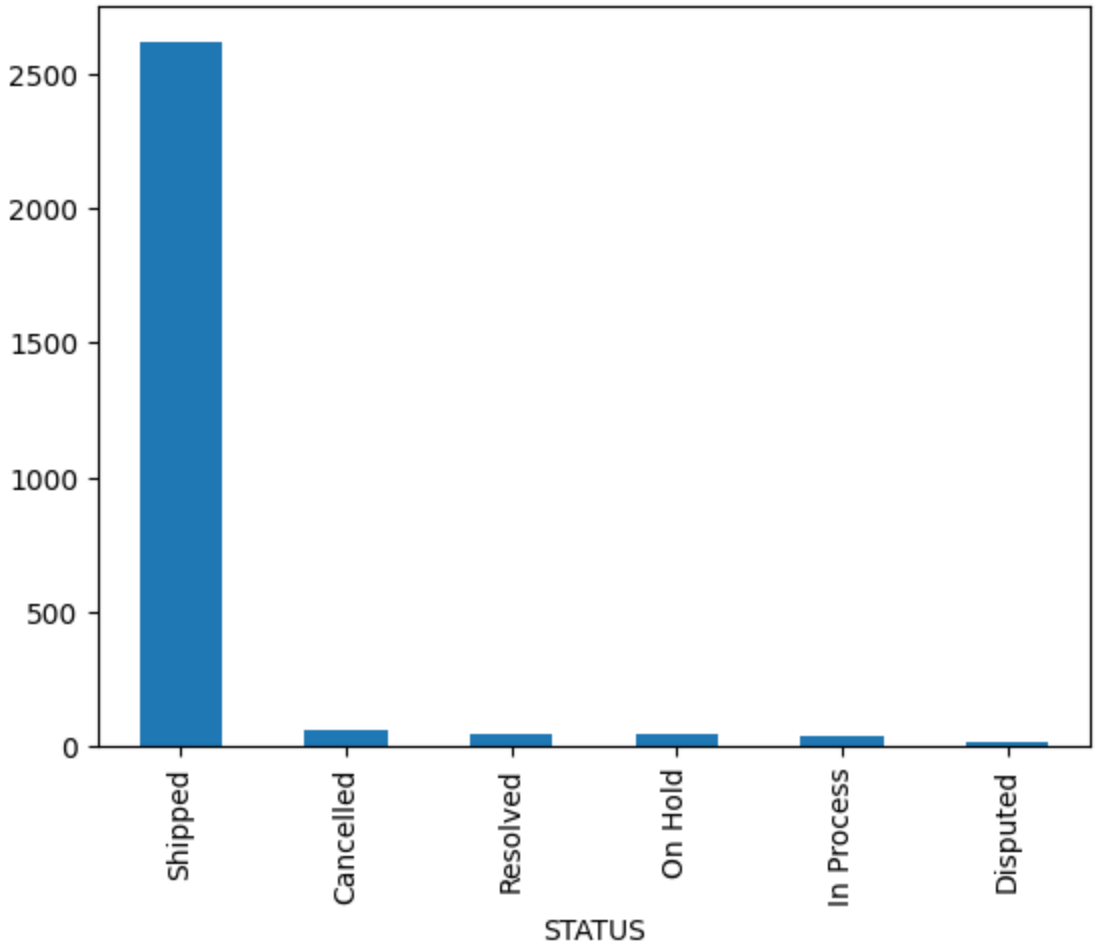
In [53]:
```
df['STATUS'].value_counts()
```

Out[53]:
```
STATUS
Shipped       2617
Cancelled       60
Resolved        47
On Hold         44
In Process      41
Disputed        14
Name: count, dtype: int64
```

49. Vẽ đồ thị xem phân bố giá trị của 1 trường trong DataFrame

In [54]:
```
df['STATUS'].value_counts().plot(kind='bar')
```

Out[54]: <Axes: xlabel='STATUS'>



50. Tạo mới DataFrame từ các Python List

In [55]:
```
peoples = {
   'name': ['Nguyễn Văn Hiếu', 'Hiếu Nguyễn Văn'],
   'age': [28, 28],
   'website': ['https://blog.luyencode.net', None]
}

df1 = pd.DataFrame(peoples)
print(df1)
```

```
            name  age                website
0  Nguyễn Văn Hiếu   28  https://blog.luyencode.net (https://blog.luyencode.net)
1  Hiếu Nguyễn Văn   28                   None
```

Tạo mới DataFrame từ các Python List

In [56]:
```
txts = ['chỗ này ăn cũng khá ngon', 'ngon, nhất định sẽ quay lại', 'thái độ phục vụ quá tệ']
labels = [1, 1, 0]
df1 = pd.DataFrame()
df1['txt'] = txts
df1['label'] = labels
print(df1)
```

```
                    txt  label
0    chỗ này ăn cũng khá ngon      1
1  ngon, nhất định sẽ quay lại      1
2       thái độ phục vụ quá tệ      0
```

51. Sắp xếp DataFrame

In [57]:
```
# Sắp xếp DataFrame df tăng dần theo cột nào đó
df1 = pd.DataFrame({'name': ['Nam', 'Hiếu', 'Mai', 'Hoa'], 'age': [18, 18, 17, 19]})
print('Before sort\n', df1)
df1 = df1.sort_values('age', ascending=True)
print('After sort\n', df1)
```

```
Before sort
   name  age
0   Nam   18
1  Hiếu   18
2   Mai   17
3   Hoa   19
After sort
   name  age
2   Mai   17
0   Nam   18
1  Hiếu   18
3   Hoa   19
```

52. Nối 2 DataFrame

In [58]:
```
# Gộp 2 DataFrame

import pandas as pd

df_1 = pd.DataFrame({'name': ['Hiếu'], 'age': [18], 'gender': ['male']})
df_2 = pd.DataFrame({'name': ['Nam', 'Mai', 'Hoa'], 'age': [15, 17, 19]})
df_3 = pd.concat([df_1, df_2], ignore_index=True)
print(df_3)
```

```
   name  age gender
0  Hiếu   18   male
1   Nam   15    NaN
2   Mai   17    NaN
3   Hoa   19    NaN
```

53. Xóa trộn các bản ghi trong DataFrame

In [59]:
```
df_1 = pd.DataFrame({'name': ['Hiếu', 'Nam', 'Mai', 'Hoa'], 'age': [18, 15, 17, 19]})
print('\nBefore shuffle\n', df)
df1 = df.sample(frac=1).reset_index(drop=True)
print('\nAfter shuffle\n', df1)
```

```
Before shuffle
     ORDERNUMBER  QUANTITYORDERED  PRICEEACH  ORDERLINENUMBER    SALES \
0          10107               30      95.70                2  2871.00
1          10121               34      81.35                5  2765.90
2          10134               41      94.74                2  3884.34
3          10145               45      83.26                6  3746.70
4          10159               49     100.00               14  5205.27
...          ...              ...        ...              ...      ...
2818       10350               20     100.00               15  2244.40
2819       10373               29     100.00                1  3978.51
2820       10386               43     100.00                4  5417.57
2821       10397               34      62.24                1  2116.16
2822       10414               47      65.52                9  3079.44

     ORDERDATE   STATUS QTR_ID  MONTH_ID  YEAR_ID  ... COUNTRY  TERRITORY \
0         2003  Shipped    Quy         2     2003  ...     USA        NaN
1         2004  Shipped    Quy         5     2003  ...  France       EMEA
2         2004  Shipped    Quy         7     2003  ...  France       EMEA
```

54. Lưu DataFrame về file csv

In [60]:
```
df1.to_csv('../data/KHACHHANG.csv')
```

55. Tối ưu hóa bộ nhớ khi dùng pandas

In [61]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, csv file I/O (e.g. pd.read_csv)

def reduce_mem_usage(df):
    """iterate through all the columns of a dataframe and modify the data type
       to reduce memory usage
    """

    start_mem = df.memory_usage().sum()/1024**2
    print('Memory usage of dataframe is {:.2f} MB'.format(start_mem))
    for col in df.columns:
        col_type = df[col].dtype

        if col_type != object and col_type.name != 'category' and 'datetime' not in col_type.name:
            c_min = df[col].min()
            c_max = df[col].max()
            if str(col_type)[:3] == 'int':
                if c_min > np.iinfo(np.int8).min and c_max < np.iinfo(np.int8).max:
                    df[col] = df[col].asptype(np.int8)
                elif c_min > np.iinfo(np.int16).min and c_max < np.iinfo(np.int16).max:
                    df[col] = df[col].asptype(np.int16)
                elif c_min > np.iinfo(np.int32).min and c_max < np.iinfo(np.int32).max:
                    df[col] = df[col].asptype(np.int32)
                elif c_min > np.iinfo(np.int64).min and c_max < np.iinfo(np.int64).max:
                    df[col] = df[col].asptype(np.int64)
            else:
                if c_min > np.finfo(np.float16).min and c_max < np.finfo(np.float16).max:
                    df[col] = df[col].astype(np.float16)
                elif c_min > np.finfo(np.float32).min and c_max < np.finfo(np.float32).max:
                    df[col] = df[col].astype(np.float32)
                else:
                    df[col] = df[col].astype(np.float64)
        elif 'datetime' not in col_type.name:
            df[col] = df[col].astype('category')
        end_mem = df.memory_usage().sum() / 1024**2
        print('Memory usage after optimization is: {:.2f} MB'.format(end_mem))
        print('Decreased bt {:.1f}'.format(100 * (start_mem - end_mem)/ start_mem))

        return df

df
```

Out[61]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERI
0	10107	30	95.70	2	2871.00	
1	10121	34	81.35	5	2765.90	
2	10134	41	94.74	2	3884.34	
3	10145	45	83.26	6	3746.70	
4	10159	49	100.00	14	5205.27	
...	...	...	...	...	...	
2818	10350	20	100.00	15	2244.40	
2819	10373	29	100.00	1	3978.51	
2820	10386	43	100.00	4	5417.57	
2821	10397	34	62.24	1	2116.16	
2822	10414	47	65.52	9	3079.44	

2823 rows × 32 columns