Theory Activity No. 1

Name : Om Shinkar

Division : CS2

Roll No. : CS2-01

PRN : 202401040177

Dataset : Sales Dataset

1.

```
import pandas as pd
import numpy as np

# Simulating a mini Sales dataset
data = {
    'product': [
        'Laptop', 'Tablet', 'Smartphone', 'Printer', 'Monitor',
        'Keyboard', 'Mouse', 'Headphones', 'Webcam', 'Charger'
    ],
    'region': [
        'North', 'South', 'East', 'West', 'North',
        'East', 'South', 'West', 'North', 'East'
    ],
    'units_sold': [120, 85, 150, 60, 90, 200, 300, 130, 75, 160],
    'unit_price': [700, 300, 500, 150, 200, 50, 30, 80, 100, 25],
    'sale_date': pd.date_range(start='2024-01-01', periods=10, freq='M')
}

sales_df = pd.DataFrame(data)
sales_df['revenue'] = sales_df['units_sold'] * sales_df['unit_price']
sales_df
```

2.

	product	region	units_sold	unit_price	sale_date	revenue
0	Laptop	North	120	700	2024-01-31	84000
1	Tablet	South	85	300	2024-02-29	25500
2	Smartphone	East	150	500	2024-03-31	75000
3	Printer	West	60	150	2024-04-30	9000
4	Monitor	North	90	200	2024-05-31	18000
5	Keyboard	East	200	50	2024-06-30	10000
6	Mouse	South	300	30	2024-07-31	9000
7	Headphones	West	130	80	2024-08-31	10400
8	Webcam	North	75	100	2024-09-30	7500
9	Charger	East	160	25	2024-10-31	4000

3.

```
[3] # 1. Total number of products sold (sum of units)
    sales_df['units_sold'].sum()
```

```
np.int64(1370)
```

```
[4] # 2. List all unique products
    sales_df['product'].unique()
```

```
array(['Laptop', 'Tablet', 'Smartphone', 'Printer', 'Monitor', 'Keyboard',
       'Mouse', 'Headphones', 'Webcam', 'Charger'], dtype=object)
```

```
[5] # 3. Find the product with maximum units sold
    sales_df.loc[sales_df['units_sold'].idxmax()]
```

	6
product	Mouse
region	South
units_sold	300
unit_price	30
sale_date	2024-07-31 00:00:00
revenue	9000

dtype: object

4.

```
# 4. Find the product with minimum units sold
sales_df.loc[sales_df['units_sold'].idxmin()]
```

	3
product	Printer
region	West
units_sold	60
unit_price	150
sale_date	2024-04-30 00:00:00
revenue	9000

dtype: object

5.

```
# 5. Sort products by revenue generated (descending)
sales_df.sort_values('revenue', ascending=False)
```

	product	region	units_sold	unit_price	sale_date	revenue
0	Laptop	North	120	700	2024-01-31	84000
2	Smartphone	East	150	500	2024-03-31	75000
1	Tablet	South	85	300	2024-02-29	25500
4	Monitor	North	90	200	2024-05-31	18000
7	Headphones	West	130	80	2024-08-31	10400
5	Keyboard	East	200	50	2024-06-30	10000
3	Printer	West	60	150	2024-04-30	9000
6	Mouse	South	300	30	2024-07-31	9000
8	Webcam	North	75	100	2024-09-30	7500
9	Charger	East	160	25	2024-10-31	4000

+ Code + Text

```
[8] # 6. Average revenue per product
sales_df['revenue'].mean()
```

np.float64(25240.0)

6.

```
[9]  # 7. Total revenue from the 'North' region
     sales_df[sales_df['region'] == 'North']['revenue'].sum()
```

```
np.int64(109500)
```

```
[10] # 8. Number of products sold in 'South' region
     sales_df[sales_df['region'] == 'South']['units_sold'].sum()
```

```
np.int64(385)
```

7.

```
# 9. Add a new column 'high_sales' (units_sold > 100)
sales_df['high_sales'] = sales_df['units_sold'] > 100
sales_df[['product', 'high_sales']]
```

	product	high_sales
0	Laptop	True
1	Tablet	False
2	Smartphone	True
3	Printer	False
4	Monitor	False
5	Keyboard	True
6	Mouse	True
7	Headphones	True
8	Webcam	False
9	Charger	True

```
[12] # 10. Count how many products had high sales
     sales_df['high_sales'].sum()
```

```
np.int64(6)
```

8.

```
[13] # 11. Find all products priced above 100
     sales_df[sales_df['unit_price'] > 100]
```

	product	region	units_sold	unit_price	sale_date	revenue	high_sales
0	Laptop	North	120	700	2024-01-31	84000	True
1	Tablet	South	85	300	2024-02-29	25500	False
2	Smartphone	East	150	500	2024-03-31	75000	True
3	Printer	West	60	150	2024-04-30	9000	False
4	Monitor	North	90	200	2024-05-31	18000	False

```
[14] # 12. Find the month with the highest total sales revenue
     sales_df.loc[sales_df['revenue'].idxmax(), 'sale_date'].month
```

```
1
```

```
# 13. Average unit price of products
sales_df['unit_price'].mean()
```

```
np.float64(213.5)
```

9.

```
[16] # 14. Group products by region and get total revenue
     sales_df.groupby('region')['revenue'].sum()
```

region	revenue
East	89000
North	109500
South	34500
West	19400

dtype: int64

10.

```
[16] # 14. Group products by region and get total revenue
     sales_df.groupby('region')['revenue'].sum()
```

region	revenue
East	89000
North	109500
South	34500
West	19400

dtype: int64

11.

```
# 16. Create a new column 'profit_estimate' assuming 20% profit margin
sales_df['profit_estimate'] = sales_df['revenue'] * 0.2
sales_df[['product', 'profit_estimate']]
```

	product	profit_estimate
0	Laptop	16800.0
1	Tablet	5100.0
2	Smartphone	15000.0
3	Printer	1800.0
4	Monitor	3600.0
5	Keyboard	2000.0
6	Mouse	1800.0
7	Headphones	2080.0
8	Webcam	1500.0
9	Charger	800.0

12.

```
[19] # 17. Find product with highest profit estimate
     sales_df.loc[sales_df['profit_estimate'].idxmax()]
```

	0
product	Laptop
region	North
units_sold	120
unit_price	700
sale_date	2024-01-31 00:00:00
revenue	84000
high_sales	True
profit_estimate	16800.0

dtype: object