

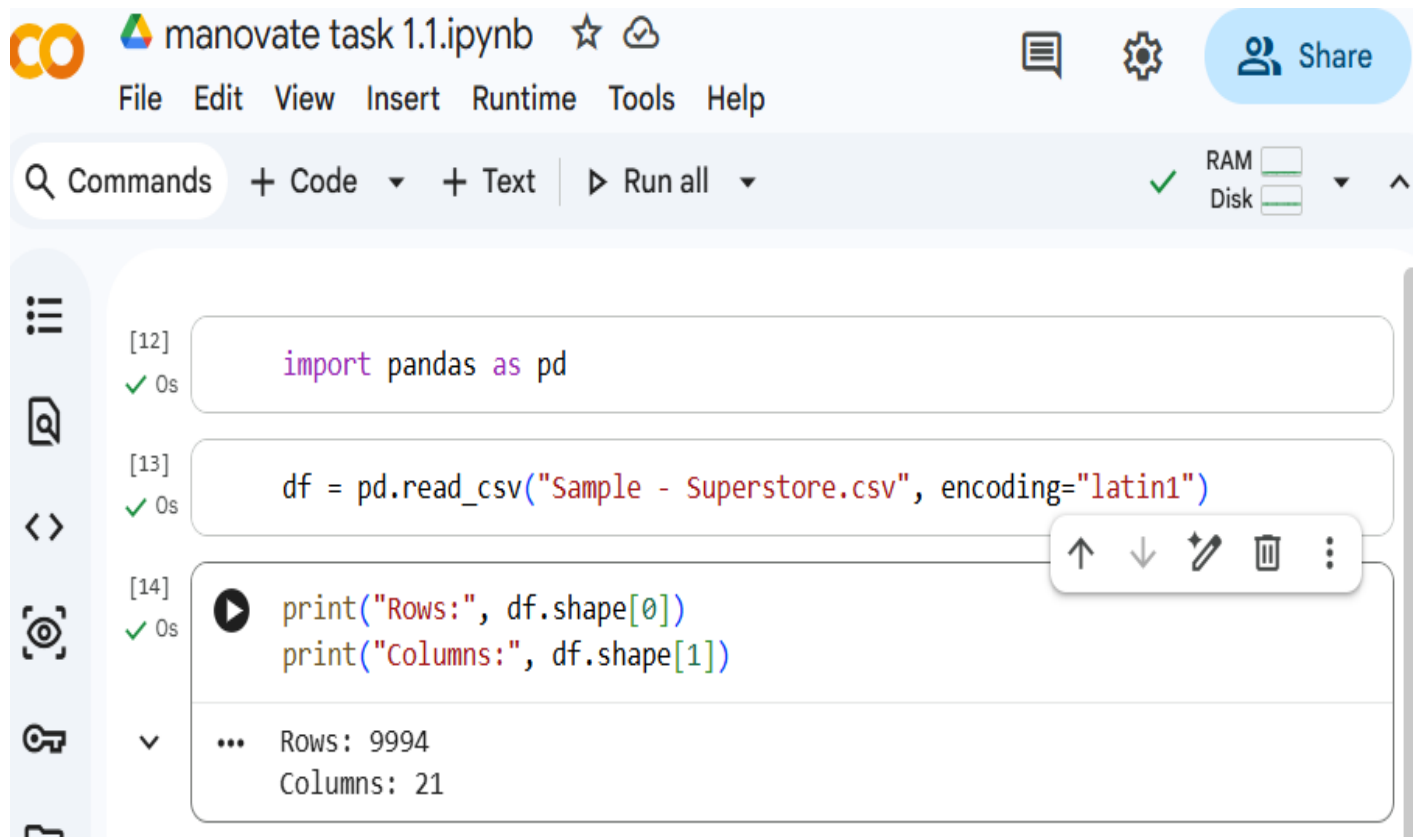
NAME: NIKHIL KRISHNA

EMP ID:IND-CHN-SD-327

TASK 1 — DATA PROFILING

1. Number of Rows & Columns

- **Rows: 9,994**
- **Columns: 21**



The screenshot displays a Jupyter Notebook interface with the title "manovate task 1.1.ipynb". The top bar includes a search bar with "Commands", a dropdown menu with "+ Code" and "+ Text", and a "Run all" button. On the right, there are icons for chat, settings, and a "Share" button. Below the top bar, the notebook content is visible, showing three code cells. The first cell contains the import statement for pandas. The second cell reads a CSV file. The third cell prints the shape of the DataFrame. The output of the third cell is displayed below the code, showing the number of rows and columns.

```
[12] import pandas as pd
[13] df = pd.read_csv("Sample - Superstore.csv", encoding="latin1")
[14] print("Rows:", df.shape[0])
    print("Columns:", df.shape[1])
```

... Rows: 9994
Columns: 21

2. Column Names & Data Types

```
5] print("\nColumn Names & Data Types:")
0s print(df.dtypes)

...
Column Names & Data Types:
Row ID          int64
Order ID        object
Order Date      object
Ship Date       object
Ship Mode       object
Customer ID     object
Customer Name   object
Segment        object
Country         object
City           object
State          object
Postal Code     int64
Region         object
Product ID     object
Category       object
Sub-Category   object
Product Name   object
Sales          float64
Quantity       int64
Discount       float64
Profit         float64
dtype: object
```

3. Missing value percentage per column

- All columns have 0% missing values.

```
3] print("\nMissing Value % per Column:")
print(df.isnull().mean() * 100)

...
Missing Value % per Column:
Row ID          0.0
Order ID        0.0
Order Date      0.0
Ship Date       0.0
Ship Mode       0.0
Customer ID     0.0
Customer Name   0.0
Segment        0.0
Country         0.0
City           0.0
State          0.0
Postal Code     0.0
Region         0.0
Product ID     0.0
Category       0.0
Sub-Category   0.0
Product Name   0.0
Sales          0.0
Quantity       0.0
Discount       0.0
Profit         0.0
dtype: float64
```

4. Duplicate records (count)

- Duplicate row count: 0

```
print("\nDuplicate Records:", df.duplicated().sum())
```

...

Duplicate Records: 0

5. Outliers in Sales / Profit

- Outliers in Sales: 1,167
- Outliers in Profit: 1,881

```
def find_outliers(series):  
    Q1 = series.quantile(0.25)  
    Q3 = series.quantile(0.75)  
    IQR = Q3 - Q1  
    lower = Q1 - (1.5 * IQR)  
    upper = Q3 + (1.5 * IQR)  
    return ((series < lower) | (series > upper)).sum()
```

```
print("\nOutliers in Sales:", find_outliers(df['Sales']))  
print("Outliers in Profit:", find_outliers(df['Profit']))
```

...

Outliers in Sales: 1167
Outliers in Profit: 1881

6.Relationship check (Discount vs Profit, Sales vs Quantity)

Relationship	Correlation	Interpretation
Discount vs Profit	-0.219	Higher discount → lower profit (weak negative relationship)
Sales vs Quantity	0.201	Higher quantity → slightly higher sales (weak positive relationship)

```
]
)s
print("\nCorrelation: Discount vs Profit:", df['Discount'].corr(df['Profit']))
print("Correlation: Sales vs Quantity:", df['Sales'].corr(df['Quantity']))
```

```
Correlation: Discount vs Profit: -0.21948745637176803
```

```
Correlation: Sales vs Quantity: 0.20079477137389765
```