```
Start coding or generate with AI.
```

```
from google.colab import files
uploaded = files.upload()
```

Choose files luxury_cos...sis_2025.csv

luxury_cosmetics_fraud_analysis_2025.csv(text/csv) - 425294 bytes, last modified: 04/09/2025 - 100% done

```
Saving luxury_cosmetics_fraud_analysis_2025.csv to luxury_cosmetics_fraud_analysis_2025 (1).csv
```

```
import pandas as pd
```

```
df = pd.read_csv("luxury_cosmetics_fraud_analysis_2025.csv")
```

```
df.head()
```

	Transaction_ID	Customer_ID	Transaction_Date	Transaction_Time	Customer_Age	Customer_Loyalty_Ti
0	702bdd9b-9c93-41e3-9dbb-a849b2422080	119dca0b-8554-4b2d-9bec-e964eaf6af97	2025-07-27	04:04:15	56.0	Silv
1	2e64c346-36bc-4acf-bc2b-8b0fdf46abc5	299df086-26c4-4708-b6d7-fcaeceb14637	2025-03-14	20:23:23	46.0	Platinu
2	29ad1278-70ce-421f-8d81-23816b39f4ac	dfa3d24d-b935-49a5-aa1d-7d57a44d8773	2025-02-20	12:36:02	32.0	Silv
3	07dc4894-e0eb-48f1-99a7-1942b1973d9b	7a67e184-9369-49ee-aeac-18f5b51b230f	2025-04-25	19:09:43	60.0	Bron
4	ae407054-5543-429c-918a-cdcc42ea9782	cf14730a-8f5a-453d-b527-39a278852b27	2025-04-17	14:23:23	NaN	Platinu

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns


sns.set(style="whitegrid")


df["Transaction_Date"] = pd.to_datetime(df["Transaction_Date"])
df["Customer_Age"].fillna(df["Customer_Age"].median(), inplace=True)
```

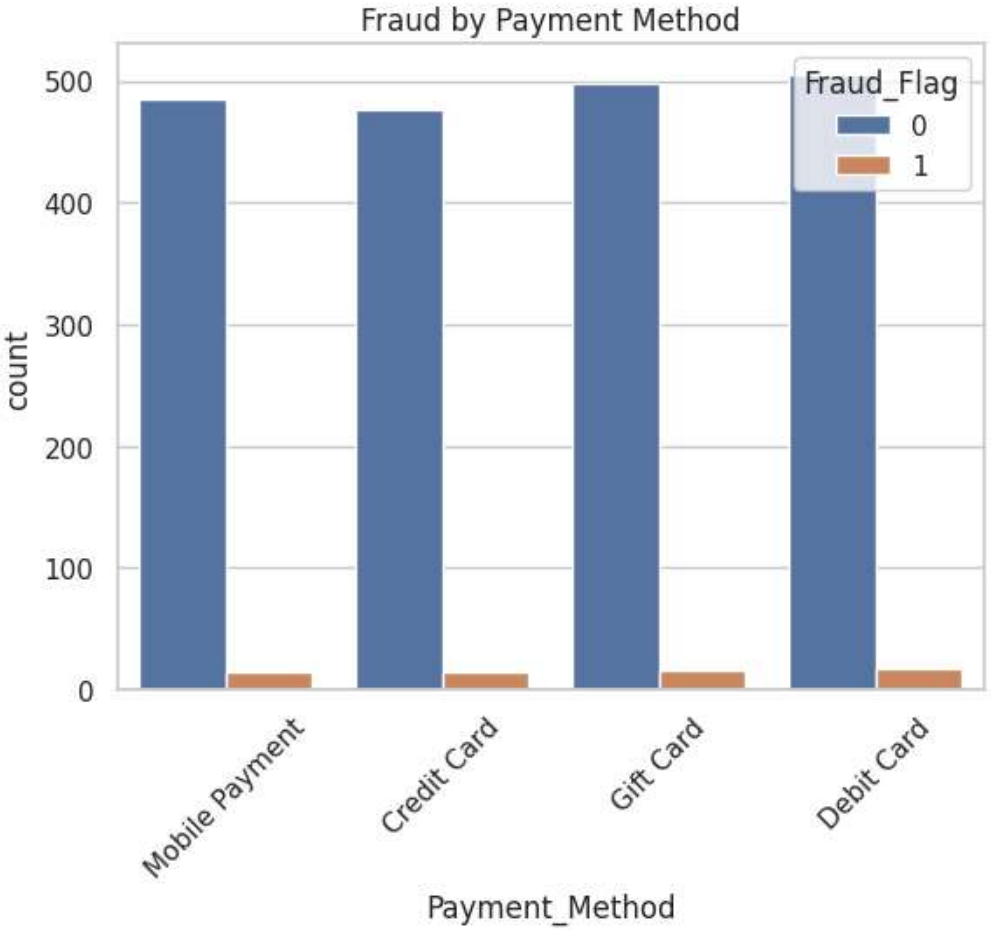
```
/tmp/ipython-input-1630701826.py:2: FutureWarning: A value is trying to be set on a copy of a DataFram
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate o

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inpl


  df["Customer_Age"].fillna(df["Customer_Age"].median(), inplace=True)
```

```
df["Customer_Age"].fillna(df["Customer_Age"].median(), inplace=True)
```

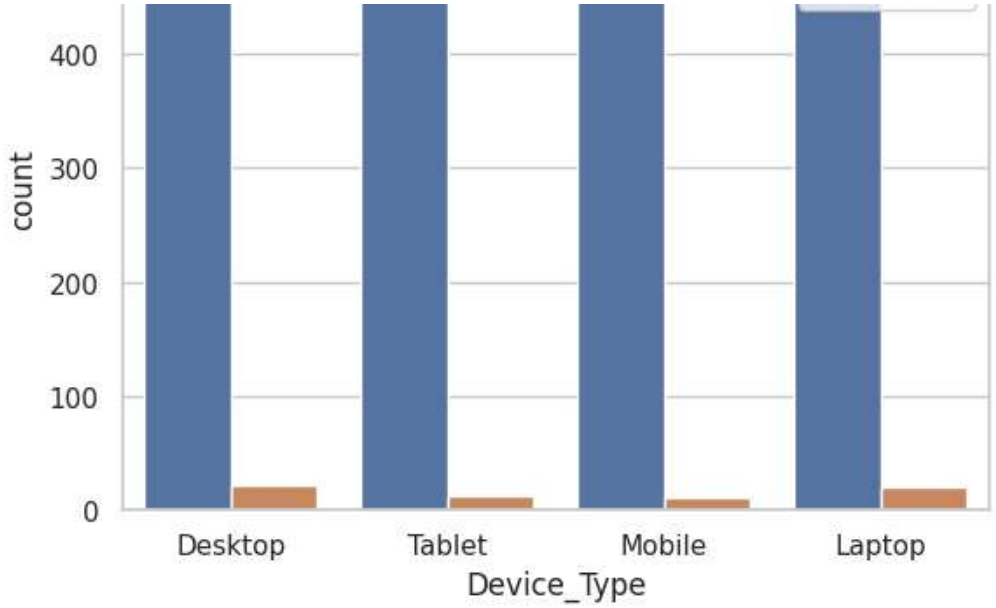
```
/tmp/ipython-input-547931938.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate o

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inpl


  df["Customer_Age"].fillna(df["Customer_Age"].median(), inplace=True)
```

```
sns.countplot(x="Fraud_Flag", data=df)
plt.title("Fraud vs Non-Fraud Transactions")
plt.show()

sns.countplot(x="Payment_Method", hue="Fraud_Flag", data=df)
plt.xticks(rotation=45)
plt.title("Fraud by Payment Method")
plt.show()

sns.countplot(x="Device_Type", hue="Fraud_Flag", data=df)
plt.title("Fraud by Device Type")
plt.show()

sns.boxplot(x="Fraud_Flag", y="Purchase_Amount", data=df)
plt.title("Purchase Amount vs Fraud")
plt.show()
```


Fraud vs Non-Fraud Transactions
count
2000
1750
1500
1250
1000
750
500
250
0
0
1
Fraud_Flag


Fraud by Payment Method
count
500
400
300
200
100
0
Fraud_Flag
0
1
Mobile Payment
Credit Card
Gift Card
Debit Card
Payment_Method


Fraud by Device Type
500
Fraud_Flag
0
1


400
300
200
100
0
count
Desktop
Tablet
Mobile
Laptop
Device_Type


Purchase Amount vs Fraud
300
250
200
150
100
50
Purchase_Amount
0
1
Fraud_Flag

```
fraud_by_location = df.groupby("Location")["Fraud_Flag"].mean().sort_values(ascending=False)
print("Fraud Rate by Location:\n", fraud_by_location.head())

fraud_by_loyalty = df.groupby("Customer_Loyalty_Tier")["Fraud_Flag"].mean().sort_values(ascending=False)
print("\nFraud Rate by Loyalty Tier:\n", fraud_by_loyalty)
```

```
Fraud Rate by Location:
 Location
Las Vegas    0.078431
Shanghai     0.051282
Miami        0.047619
Sydney       0.046875
Singapore    0.044643
Name: Fraud_Flag, dtype: float64

Fraud Rate by Loyalty Tier:
 Customer_Loyalty_Tier
VIP         0.054054
Platinum    0.039106
Silver      0.031621
Bronze      0.029703
Gold        0.026005
Name: Fraud_Flag, dtype: float64
```

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

X = df[["Customer_Age", "Purchase_Amount", "Footfall_Count"]]
y = df["Fraud_Flag"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

model = RandomForestClassifier()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98       620
           1       0.00      0.00      0.00        20

    accuracy                           0.97       640
   macro avg       0.48      0.50      0.49       640
weighted avg       0.94      0.97      0.95       640

/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarnin
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarnin
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarnin
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
fraud_counts = df["Fraud_Flag"].value_counts()
fraud_counts.plot.pie(autopct='%1.1f%%', labels=["Non-Fraud", "Fraud"], colors=["skyblue", "salmon"])
```