```
import sqlite3

# DB connection
conn = sqlite3.connect("retail_orders.db")
cursor = conn.cursor()

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

```
<IPython.core.display.HTML object>
```

```
Saving Walmart_Sales.csv to Walmart_Sales.csv
```

```
import pandas as pd

df = pd.read_csv("Walmart_Sales.csv")
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 6435,\n  \"fields\":
[\n    {\n      \"column\": \"Store\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 12,\n        \"min\": 1,\n
\"max\": 45,\n        \"num_unique_values\": 45,\n        \"samples\":
[\n          40,\n          26,\n          27\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Date\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 143,\n
\"samples\": [\n          \"04-05-2012\",\n          \"18-06-2010\",\n
\"02-09-2011\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Weekly_Sales\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 564366.6220536974,\n        \"min\":
209986.25,\n        \"max\": 3818686.45,\n
\"num_unique_values\": 6435,\n        \"samples\": [\n
1138800.32,\n          1304850.67,\n          1769296.25\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Holiday_Flag\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Temperature\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 18.444932875811585,\n        \"min\": -
2.06,\n        \"max\": 100.14,\n        \"num_unique_values\": 3528,\
n        \"samples\": [\n          51.13,\n          98.15\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Fuel_Price\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.4590197071928516,\n        \"min\":
2.472,\n        \"max\": 4.468,\n        \"num_unique_values\": 892,\n
```

\"samples\": [\n 2.84,\n 3.95\n ],\n \"semantic_type\": \"\",\n \"description\": \"\"\n }\n },\n {\n \"column\": \"CPI\",\n \"properties\": {\n \"dtype\": \"number\",\n \"std\": 39.35671229566413,\n \"min\": 126.064,\n \"max\": 227.2328068,\n \"num_unique_values\": 2145,\n \"samples\": [\n 184.613419,\n 214.1083654\n ],\n \"semantic_type\": \"\",\n \"description\": \"\"\n }\n },\n {\n \"column\": \"Unemployment\",\n \"properties\": {\n \"dtype\": \"number\",\n \"std\": 1.8758847818628084,\n \"min\": 3.879,\n \"max\": 14.313,\n \"num_unique_values\": 349,\n \"samples\": [\n 8.185,\n 7.804\n ],\n \"semantic_type\": \"\",\n \"description\": \"\"\n }\n }\n ]\n}","type":"dataframe","variable_name":"df"}

```
import sqlite3

conn = sqlite3.connect("retail_orders.db")
df.to_sql("orders", conn, if_exists="replace", index=False)

print("Data loaded into SQL table 'orders' successfully!")
```

```
Data loaded into SQL table 'orders' successfully!
```

```
pd.read_sql("SELECT Store, Date, Weekly_Sales FROM orders LIMIT 10;", conn)
```

{"summary":"{\n \"name\": \"pd\",\n \"rows\": 10,\n \"fields\": [\n {\n \"column\": \"Store\",\n \"properties\": {\n \"dtype\": \"number\",\n \"std\": 0,\n \"min\": 1,\n \"max\": 1,\n \"num_unique_values\": 1,\n \"samples\": [\n 1\n ],\n \"semantic_type\": \"\",\n \"description\": \"\"\n }\n },\n {\n \"column\": \"Date\",\n \"properties\": {\n \"dtype\": \"string\",\n \"num_unique_values\": 10,\n \"samples\": [\n \"02-04-2010\"\n ],\n \"semantic_type\": \"\",\n \"description\": \"\"\n }\n },\n {\n \"column\": \"Weekly_Sales\",\n \"properties\": {\n \"dtype\": \"number\",\n \"std\": 93652.49664286087,\n \"min\": 1404429.92,\n \"max\": 1643690.9,\n \"num_unique_values\": 10,\n \"samples\": [\n 1594968.28\n ],\n \"semantic_type\": \"\",\n \"description\": \"\"\n }\n }\n ]\n}","type":"dataframe"}

```
df.columns
```

```
Index(['Store', 'Date', 'Weekly_Sales', 'Holiday_Flag', 'Temperature',
       'Fuel_Price', 'CPI', 'Unemployment'],
      dtype='object')
```

```
pd.read_sql("SELECT * FROM orders WHERE Weekly_Sales > 100000;", conn)
```


```
{"summary":"{\n  \"name\": \"pd\",\n  \"rows\": 6435,\n  \"fields\":
[\n    {\n      \"column\": \"Store\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 12,\n        \"min\": 1,\n
\"max\": 45,\n        \"num_unique_values\": 45,\n        \"samples\":
[\n          40,\n          26,\n          27\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Date\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 143,\n
\"samples\": [\n          \"04-05-2012\",\n          \"18-06-2010\",\n
\"02-09-2011\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Weekly_Sales\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 564366.6220536974,\n        \"min\":
209986.25,\n        \"max\": 3818686.45,\n
\"num_unique_values\": 6435,\n        \"samples\": [\n
1138800.32,\n          1304850.67,\n          1769296.25\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Holiday_Flag\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Temperature\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 18.444932875811585,\n        \"min\": -
2.06,\n        \"max\": 100.14,\n        \"num_unique_values\": 3528,\
n        \"samples\": [\n          51.13,\n          98.15\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Fuel_Price\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.4590197071928516,\n        \"min\":
2.472,\n        \"max\": 4.468,\n        \"num_unique_values\": 892,\n
\"samples\": [\n          2.84,\n          3.95\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"CPI\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 39.35671229566413,\n
\"min\": 126.064,\n        \"max\": 227.2328068,\n
\"num_unique_values\": 2145,\n        \"samples\": [\n
184.613419,\n          214.1083654\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Unemployment\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1.8758847818628084,\n        \"min\": 3.879,\n        \"max\":
14.313,\n        \"num_unique_values\": 349,\n        \"samples\": [\n
8.185,\n          7.804\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe"}
```

```
pd.read_sql("SELECT * FROM orders ORDER BY Weekly_Sales DESC LIMIT
10;", conn)
```


```
{"summary":"{\n  \"name\": \"pd\",\n  \"rows\": 10,\n  \"fields\": [\n
{\n      \"column\": \"Store\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 6,\n        \"min\": 2,\n
\"max\": 20,\n        \"num_unique_values\": 6,\n        \"samples\":
[\n          14,\n          20,\n          2\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Date\",\n      \"properties\": {\n
\"dtype\": \"object\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"23-12-2011\",\n          \"24-12-2010\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Weekly_Sales\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
128873.59315014462,\n        \"min\": 3436007.68,\n        \"max\":
3818686.45,\n        \"num_unique_values\": 10,\n        \"samples\":
[\n          3487986.89,\n          3766687.43\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Holiday_Flag\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 0,\n
\"num_unique_values\": 1,\n        \"samples\": [\n          0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Temperature\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
10.771998731278549,\n        \"min\": 24.76,\n        \"max\": 57.06,\
n        \"num_unique_values\": 10,\n        \"samples\": [\n
48.36\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Fuel_Price\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.22367446583521025,\n        \"min\":
2.846,\n        \"max\": 3.541,\n        \"num_unique_values\": 9,\n
\"samples\": [\n          3.541\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"CPI\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 39.12284719628951,\n
\"min\": 126.9835806,\n        \"max\": 212.2360401,\n
\"num_unique_values\": 6,\n        \"samples\": [\n
182.54459\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unemployment\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1.1311911175590288,\n        \"min\":
5.143,\n        \"max\": 9.003,\n        \"num_unique_values\": 10,\n
\"samples\": [\n          7.874\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```


```
pd.read_sql("""
SELECT Store, SUM(Weekly_Sales) AS Total_Sales, AVG(Weekly_Sales) AS
```

```
Avg_Sales
FROM orders
GROUP BY Store
ORDER BY Total_Sales DESC;
""", conn)
```

```
{"summary":"{\n  \"name\": \"\\\"\\\"\\\", conn)\",\n  \"rows\": 45,\n
\"fields\": [\n    {\n      \"column\": \"Store\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
13,\n        \"min\": 1,\n        \"max\": 45,\n
\"num_unique_values\": 45,\n        \"samples\": [\n          3,\n
8,\n          17\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Total_Sales\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 78167556.49249025,\n        \"min\":
37160221.960000016,\n        \"max\": 301397792.46000004,\n
\"num_unique_values\": 45,\n        \"samples\": [\n
57586735.07,\n          129951181.13,\n          127782138.83000007\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Avg_Sales\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
546626.2691782535,\n        \"min\": 259861.69202797214,\n
\"max\": 2107676.8703496507,\n        \"num_unique_values\": 45,\n
\"samples\": [\n          402704.44104895106,\n
908749.5183916084,\n          893581.390419581\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```

```
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
```

```
df.to_sql("orders", conn, if_exists="replace", index=False)
```

```
6435
```

```
pd.read_sql("""
SELECT strftime('%m', Date) AS Month, SUM(Weekly_Sales) AS
Monthly_Sales
FROM orders
GROUP BY Month
ORDER BY Monthly_Sales DESC;
""", conn)
```

```
{"summary":"{\n  \"name\": \"\\\"\\\"\\\", conn)\",\n  \"rows\": 13,\n
\"fields\": [\n    {\n      \"column\": \"Month\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 12,\n        \"samples\": [\n          \"07\",\
n          \"08\",\n          \"10\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Monthly_Sales\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1052012366.9732459,\n        \"min\": 181559036.28000015,\n
```

```
\"max\": 4018647429.0999904,\n        \"num_unique_values\": 13,\n
\"samples\": [\n          186820778.95999998,\n
232764958.4299998,\n          4018647429.0999904\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```

```
pd.read_sql("SELECT * FROM orders WHERE Holiday_Flag = 1 ORDER BY
Weekly_Sales DESC LIMIT 10;", conn)
```

```
{"repr_error":"Out of range float values are not JSON compliant:
nan","type":"dataframe"}
```

```
pd.read_sql("SELECT Store, Weekly_Sales AS Revenue FROM orders LIMIT
5;", conn)
```

```
{"summary":"{\n  \"name\": \"pd\",\n  \"rows\": 5,\n  \"fields\": [\n
{\n      \"column\": \"Store\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0,\n        \"min\": 1,\n
\"max\": 1,\n        \"num_unique_values\": 1,\n        \"samples\":
[\n          1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Revenue\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 97798.39377032383,\n        \"min\": 1409727.59,\n
\"max\": 1643690.9,\n        \"num_unique_values\": 5,\n
\"samples\": [\n          1641957.44\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```

```
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
df.to_sql("orders", conn, if_exists="replace", index=False)
```

```
6435
```

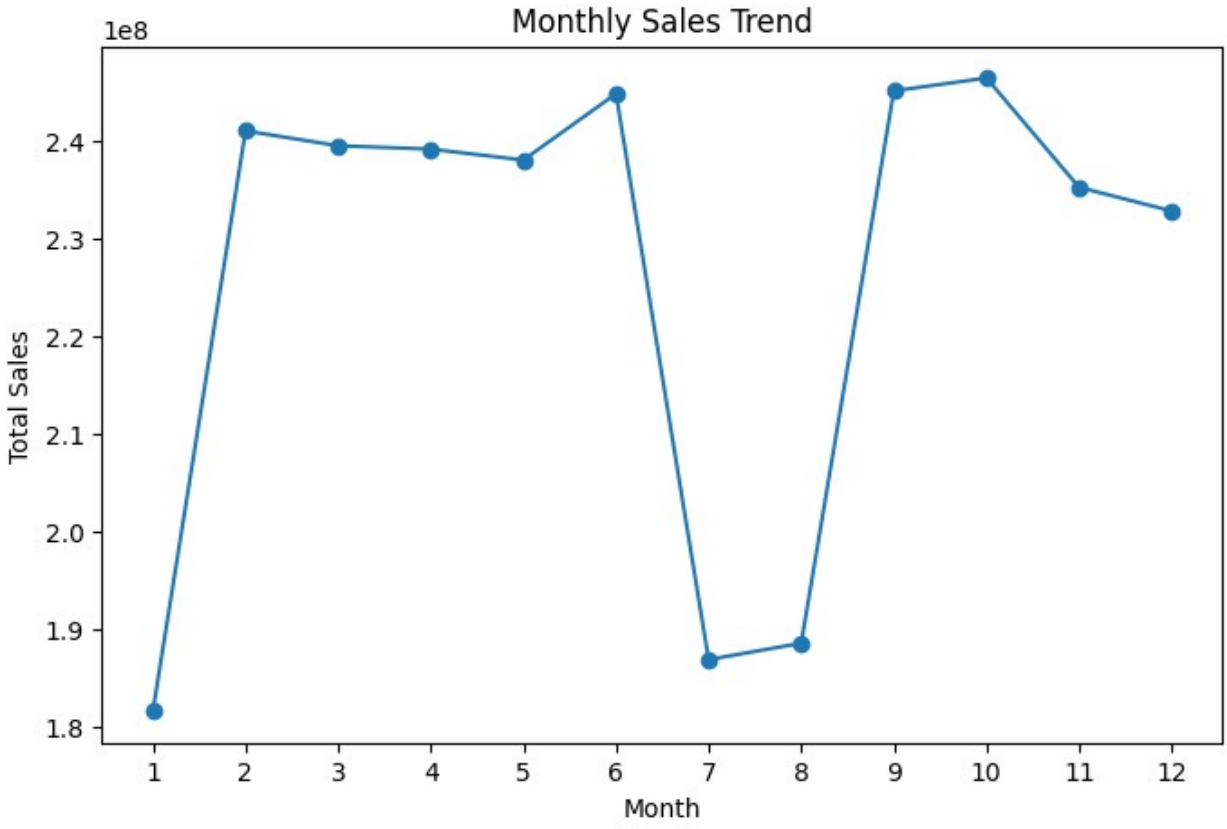
```
monthly_sales = pd.read_sql("""
SELECT strftime('%m', Date) AS Month, SUM(Weekly_Sales) AS
Monthly_Sales
FROM orders
WHERE Date IS NOT NULL
GROUP BY Month
ORDER BY Month;
""", conn)
```

```
monthly_sales = monthly_sales.dropna()
monthly_sales['Month'] = monthly_sales['Month'].astype(int)
```

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

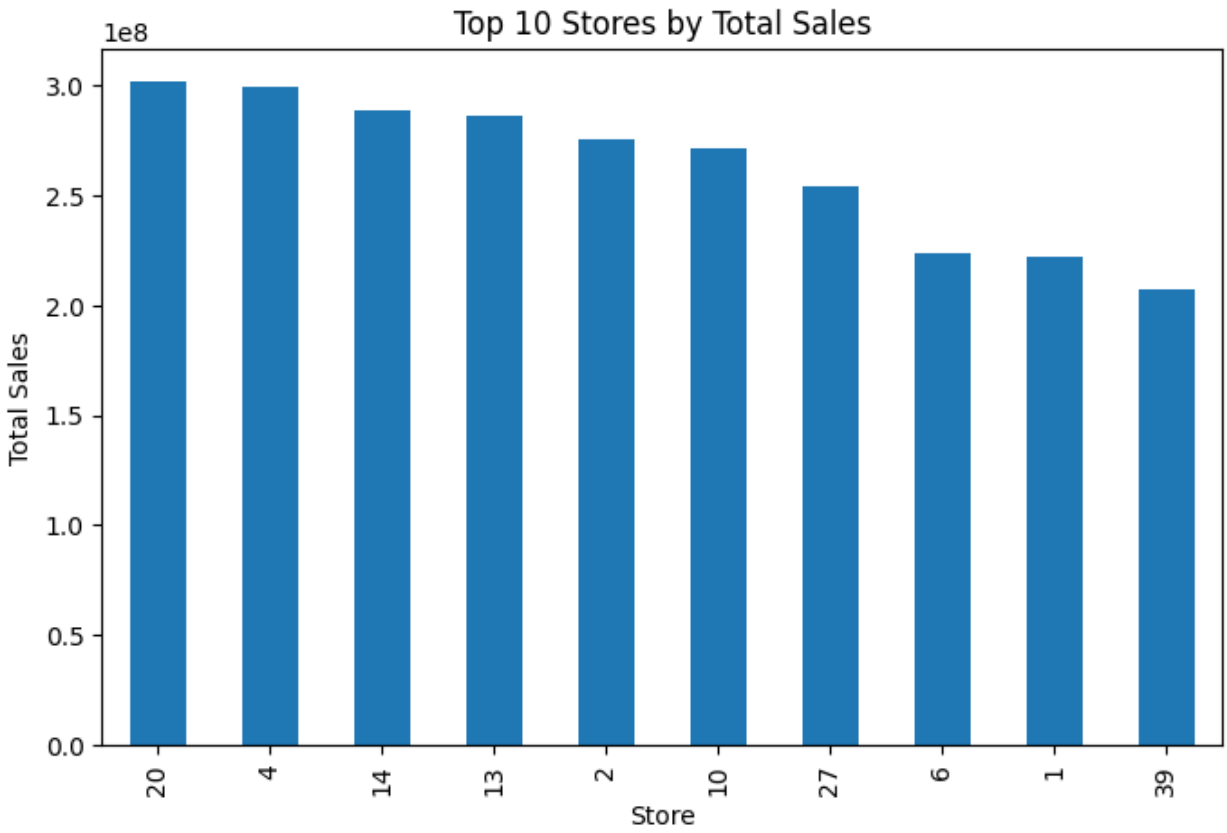
```
plt.figure(figsize=(8,5))
plt.plot(monthly_sales['Month'], monthly_sales['Monthly_Sales'],
marker='o')
```

```
plt.title("Monthly Sales Trend")
plt.xlabel("Month")
plt.ylabel("Total Sales")
plt.xticks(range(1,13))
plt.show()
```



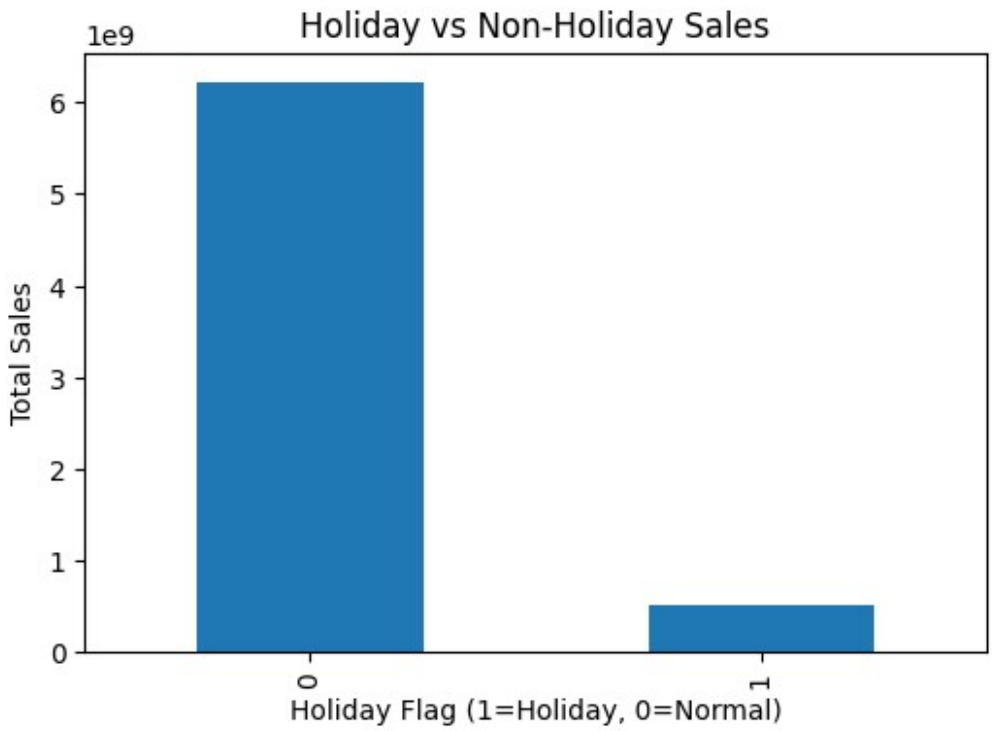

```
store_sales = pd.read_sql("""
SELECT Store, SUM(Weekly_Sales) AS Total_Sales
FROM orders
GROUP BY Store
ORDER BY Total_Sales DESC
LIMIT 10;
""", conn)

store_sales.plot(kind="bar", x="Store", y="Total_Sales",
figsize=(8,5), legend=False)
plt.title("Top 10 Stores by Total Sales")
plt.xlabel("Store")
plt.ylabel("Total Sales")
plt.show()
```




```
holiday_sales = pd.read_sql("""
SELECT Holiday_Flag, SUM(Weekly_Sales) AS Total_Sales
FROM orders
GROUP BY Holiday_Flag;
""", conn)

holiday_sales.plot(kind="bar", x="Holiday_Flag", y="Total_Sales",
figsize=(6,4), legend=False)
plt.title("Holiday vs Non-Holiday Sales")
plt.xlabel("Holiday Flag (1=Holiday, 0=Normal)")
plt.ylabel("Total Sales")
plt.show()
```




```
pd.read_sql("SELECT DISTINCT Store FROM orders;", conn)
```


```
{"summary":"{\n  \"name\": \"pd\",\n  \"rows\": 45,\n  \"fields\": [\n{\n      \"column\": \"Store\",\n      \"properties\": {\n\"dtype\": \"number\",\n        \"std\": 13,\n        \"min\": 1,\n\"max\": 45,\n        \"num_unique_values\": 45,\n        \"samples\": [\n          40,\n          26,\n          27\n        ],\n\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```


```
pd.read_sql("SELECT * FROM orders WHERE Weekly_Sales BETWEEN 50000 AND 100000;", conn)
```


```
{"repr_error":"Out of range float values are not JSON compliant: nan","type":"dataframe"}
```


```
pd.read_sql("""
SELECT Store, SUM(Weekly_Sales) AS Total_Sales
FROM orders
GROUP BY Store
HAVING Total_Sales > 10000000
ORDER BY Total_Sales DESC;
""", conn)
```


```
{"summary":"{\n  \"name\": \"\\\"\\\"\\\", conn)\",\n  \"rows\": 45,\n\"fields\": [\n    {\n      \"column\": \"Store\",\n
```

\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
13,\n        \"min\": 1,\n        \"max\": 45,\n
\"num_unique_values\": 45,\n        \"samples\": [\n          3,\n
8,\n          17\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Total_Sales\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 78167556.49249025,\n        \"min\":
37160221.960000016,\n        \"max\": 301397792.46000004,\n
\"num_unique_values\": 45,\n        \"samples\": [\n
57586735.07,\n          129951181.13,\n          127782138.83000007\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe"}

```
pd.read_sql("""
SELECT
    CASE WHEN Holiday_Flag = 1 THEN 'Holiday' ELSE 'Normal Day' END AS
Day_Type,
    SUM(Weekly_Sales) AS Total_Sales
FROM orders
GROUP BY Day_Type;
""", conn)
```

{"summary":"{\n  \"name\": \"\\\"\\\"\\\"\", conn)\",\n  \"rows\": 2,\n
\"fields\": [\n    {\n      \"column\": \"Day_Type\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n          \"Normal
Day\",\n          \"Holiday\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Total_Sales\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 4049331753.247053,\n
\"min\": 505299551.55999994,\n        \"max\": 6231919435.550006,\n
\"num_unique_values\": 2,\n        \"samples\": [\n
6231919435.550006,\n          505299551.55999994\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}

```
conn.close()
print("SQL session closed. Task completed ")
```

```
SQL session closed. Task completed 
```