```
!pip install openpyxl
```

```
Requirement already satisfied: openpyxl in
/usr/local/lib/python3.11/dist-packages (3.1.5)
Requirement already satisfied: et-xmlfile in
/usr/local/lib/python3.11/dist-packages (from openpyxl) (2.0.0)
```

```
import pandas as pd
import matplotlib.pyplot as plt
```

```
!wget https://archive.ics.uci.edu/static/public/352/online+retail.zip
```

```
--2025-02-22 21:35:30--
https://archive.ics.uci.edu/static/public/352/online+retail.zip
Resolving archive.ics.uci.edu (archive.ics.uci.edu)... 128.195.10.252
Connecting to archive.ics.uci.edu (archive.ics.uci.edu)|
128.195.10.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified
Saving to: ‘online+retail.zip’

online+retail.zip       [      <=>           ]  22.62M  18.9MB/s    in
1.2s

2025-02-22 21:35:32 (18.9 MB/s) - ‘online+retail.zip’ saved [23715478]
```

```
!unzip online+retail.zip
```

```
Archive:  online+retail.zip
 extracting: Online Retail.xlsx
```

```
import time
stime = time.time()

df1 = pd.read_excel("Online Retail.xlsx", dtype = {"InvoiceNo" :
"string", "StockCode" : "string", "Description" : "string",
"Country" : "string"})
df1
```

```
{"type":"dataframe","variable_name":"df1"}
```

```
df1.shape
```

```
(541909, 8)
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
```

```
---  ------       --------------   -----
 0   InvoiceNo    541909 non-null  string
 1   StockCode    541909 non-null  string
 2   Description  540455 non-null  string
 3   Quantity     541909 non-null  int64
 4   InvoiceDate  541909 non-null  datetime64[ns]
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  string
dtypes: datetime64[ns](1), float64(2), int64(1), string(4)
memory usage: 33.1 MB
```

Here I have download the file in my local computer.

```
# df1.to_excel("Online Retail.xlsx", index=False)

# from google.colab import files
# files.download("Online Retail.xlsx")
```

Data Cleaning and Handling Missing Values Process began

```
df1.isna().sum()
```

```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

```
df1[df1.Description.isnull()]
```


```
{"summary":"{\n  \"name\": \"df1[df1\",\n  \"rows\": 1454,\n
\"fields\": [\n    {\n      \"column\": \"InvoiceNo\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 1454,\n        \"samples\": [\n
\"548997\",\n          \"569347\",\n          \"547957\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"StockCode\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 960,\n        \"samples\": [\n
\"48173C\",\n          \"84858C\",\n          \"84678\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Description\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 0,\n        \"samples\": [],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
```

```
n    },\n    {\n      \"column\": \"Quantity\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 243,\n
\"min\": -3667,\n        \"max\": 5568,\n
\"num_unique_values\": 266,\n        \"samples\": [],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"InvoiceDate\",\n
\"properties\": {\n        \"dtype\": \"date\",\n        \"min\":
\"2010-12-01 11:52:00\",\n        \"max\": \"2011-12-08 14:06:00\",\n
\"num_unique_values\": 1121,\n        \"samples\": [],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"UnitPrice\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.0,\n        \"min\": 0.0,\n        \"max\": 0.0,\n
\"num_unique_values\": 1,\n        \"samples\": [],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"CustomerID\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
null,\n        \"min\": null,\n        \"max\": null,\n
\"num_unique_values\": 0,\n        \"samples\": [],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Country\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\": 1,\n
\"samples\": [],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```


```
df1[df1.StockCode == "22139"]
```


```
{"summary":"{\n  \"name\": \"df1[df1\",\n  \"rows\": 993,\n
\"fields\": [\n    {\n      \"column\": \"InvoiceNo\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 983,\n        \"samples\": [\n
\"576089\",\n          \"575919\",\n          \"576269\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"StockCode\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n
\"22139\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Description\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          \"amazon\"\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Quantity\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 7,\n        \"min\": -14,\n
\"max\": 96,\n        \"num_unique_values\": 22,\n        \"samples\":
[\n          23\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"InvoiceDate\",\n      \"properties\": {\n        \"dtype\":
\"date\",\n        \"min\": \"2010-12-01 09:41:00\",\n        \"max\":
\"2011-12-09 11:59:00\",\n        \"num_unique_values\": 978,\n
```

```
\"samples\": [\n          \"2011-04-12 12:40:00\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"UnitPrice\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
2.056683418428866,\n        \"min\": 0.0,\n        \"max\": 11.02,\n
\"num_unique_values\": 5,\n        \"samples\": [\n          0.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"CustomerID\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1758.7728831262718,\n        \"min\": 12360.0,\n        \"max\":
18283.0,\n        \"num_unique_values\": 489,\n        \"samples\": [\
n          13471.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Country\",\n      \"properties\": {\n        \"dtype\": \"string\",\
n        \"num_unique_values\": 22,\n        \"samples\": [\n
\"United Kingdom\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```


This is the most frequent Description of "22139" StockCode. We will use this Description to replace all the null/inaccurate values of this StockCode.

```
df1[df1.StockCode == "22139"].Description.mode()

0    RETROSPOT TEA SET CERAMIC 11 PC
Name: Description, dtype: string

df1[["StockCode", "Description"]].value_counts()

StockCode  Description
85123A     WHITE HANGING HEART T-LIGHT HOLDER    2302
22423      REGENCY CAKESTAND 3 TIER              2200
85099B     JUMBO BAG RED RETROSPOT               2159
47566      PARTY BUNTING                         1727
20725      LUNCH BAG RED RETROSPOT               1638
                                                 ...
35954      counted                                  1
           check                                    1
35923      check                                    1
35915C     damaged                                  1
m          Manual                                   1
Name: count, Length: 4792, dtype: int64

most_freq = df1[["StockCode",
"Description"]].value_counts().reset_index()
most_freq
```


```
{"summary":"{\n  \"name\": \"most_freq\",\n  \"rows\": 4792,\n
\"fields\": [\n    {\n      \"column\": \"StockCode\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 3958,\n        \"samples\": [\n
```

\"21843\",\n   \"22586\",\n   \"21291\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Description\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 4223,\n   \"samples\": [\n   \"OOH
LA LA DOGS COLLAR\",\n   \"MULTI HEARTS STICKERS\",\n
\"METALIC LEAVES BAG CHARMS\"\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"count\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 186,\n   \"min\": 1,\n
\"max\": 2302,\n   \"num_unique_values\": 617,\n
\"samples\": [\n   919,\n   36,\n   742\
n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   }\n]\
n}","type":"dataframe","variable_name":"most_freq"}

```
most_freq[most_freq.StockCode == '22423']
```

{"summary":"{\n \"name\": \"most_freq[most_freq\",\n \"rows\": 3,\n
\"fields\": [\n   {\n   \"column\": \"StockCode\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 1,\n   \"samples\": [\n
\"22423\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Description\",\n   \"properties\": {\n   \"dtype\":
\"string\",\n   \"num_unique_values\": 3,\n   \"samples\":
[\n   \"REGENCY CAKESTAND 3 TIER\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"count\",\n   \"properties\": {\
n   \"dtype\": \"number\",\n   \"std\": 1269,\n
\"min\": 1,\n   \"max\": 2200,\n   \"num_unique_values\":
3,\n   \"samples\": [\n   2200\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe"}

```
most_freq = most_freq.groupby("StockCode").head(1)
most_freq
```

{"summary":"{\n \"name\": \"most_freq\",\n \"rows\": 3958,\n
\"fields\": [\n   {\n   \"column\": \"StockCode\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 3958,\n   \"samples\": [\n
\"21843\",\n   \"22586\",\n   \"21291\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Description\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 3819,\n   \"samples\": [\n
\"BLACK LOVE BIRD T-LIGHT HOLDER\",\n   \"POLKADOT COFFEE CUP &
SAUCER PINK\",\n   \"ENCHANTED BIRD PLANT CAGE\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\

```
n    },\n    {\n      \"column\": \"count\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 198,\n
\"min\": 1,\n        \"max\": 2302,\n        \"num_unique_values\":
616,\n        \"samples\": [\n          764,\n          445,\n
46\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"most_freq"}
```


```
most_freq.columns = ["StockCode", "freq_description", "count"]

df2 = df1.merge(most_freq, on = "StockCode", how = "left")

df2
```


```
{"type":"dataframe","variable_name":"df2"}
```


```
df2[df2.StockCode == "85123A"]
```


```
{"summary":"{\n  \"name\": \"df2[df2\",\n  \"rows\": 2313,\n
\"fields\": [\n    {\n      \"column\": \"InvoiceNo\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 2246,\n        \"samples\": [\n
\"568210\",\n          \"543909\",\n          \"564530\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"StockCode\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n
\"85123A\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Description\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          \"?\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Quantity\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 117,\n        \"min\": -1930,\n
\"max\": 4000,\n        \"num_unique_values\": 67,\n
\"samples\": [\n          33\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"InvoiceDate\",\n      \"properties\": {\n
\"dtype\": \"date\",\n        \"min\": \"2010-12-01 08:26:00\",\n
\"max\": \"2011-12-09 11:34:00\",\n        \"num_unique_values\":
2229,\n        \"samples\": [\n          \"2010-12-06 10:19:00\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"UnitPrice\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.8172110151691304,\n        \"min\": 0.0,\n        \"max\": 5.91,\n
\"num_unique_values\": 8,\n        \"samples\": [\n          2.95\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"CustomerID\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
```

```
1615.627886636571,\n        \"min\": 12370.0,\n        \"max\":
18283.0,\n        \"num_unique_values\": 858,\n        \"samples\": [\
n          16978.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Country\",\n      \"properties\": {\n        \"dtype\": \"string\",\
n        \"num_unique_values\": 16,\n        \"samples\": [\n
\"United Kingdom\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"freq_description\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 1,\n        \"samples\":
[\n          \"WHITE HANGING HEART T-LIGHT HOLDER\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"count\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 0.0,\n
\"min\": 2302.0,\n        \"max\": 2302.0,\n
\"num_unique_values\": 1,\n        \"samples\": [\n          2302.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe"}
```


```
df2.Description = df2.freq_description
df2.head()
```


```
{"type":"dataframe","variable_name":"df2"}
```


```
df2[df2.StockCode == "85123A"]
```


```
{"summary":"{\n  \"name\": \"df2[df2\",\n  \"rows\": 2313,\n
\"fields\": [\n    {\n      \"column\": \"InvoiceNo\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 2246,\n        \"samples\": [\n
\"568210\",\n          \"543909\",\n          \"564530\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"StockCode\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n
\"85123A\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Description\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 1,\n        \"samples\":
[\n          \"WHITE HANGING HEART T-LIGHT HOLDER\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Quantity\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 117,\n
\"min\": -1930,\n        \"max\": 4000,\n
\"num_unique_values\": 67,\n        \"samples\": [\n          33\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"InvoiceDate\",\n
\"properties\": {\n        \"dtype\": \"date\",\n        \"min\":
\"2010-12-01 08:26:00\",\n        \"max\": \"2011-12-09 11:34:00\",\n
\"num_unique_values\": 2229,\n        \"samples\": [\n
```

\"2010-12-06 10:19:00\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"UnitPrice\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0.8172110151691304,\n   \"min\": 0.0,\n   \"max\": 5.91,\n   \"num_unique_values\": 8,\n   \"samples\": [\n   2.95\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"CustomerID\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1615.627886636571,\n   \"min\": 12370.0,\n   \"max\": 18283.0,\n   \"num_unique_values\": 858,\n   \"samples\": [\n   16978.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Country\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 16,\n   \"samples\": [\n   \"United Kingdom\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"freq_description\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 1,\n   \"samples\": [\n   \"WHITE HANGING HEART T-LIGHT HOLDER\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"count\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0.0,\n   \"min\": 2302.0,\n   \"max\": 2302.0,\n   \"num_unique_values\": 1,\n   \"samples\": [\n   2302.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe"}

```
df2.isnull().sum()
```

```
InvoiceNo                0
StockCode                0
Description            112
Quantity                 0
InvoiceDate              0
UnitPrice                0
CustomerID          135080
Country                  0
freq_description       112
count                  112
dtype: int64
```

We have replace all the invalid Description with the Frequent Description. Now i can remove the actual null values from the Dataset.

```
df2.dropna(subset="Description", inplace = True)

df2.isnull().sum()
```

```
InvoiceNo                0
StockCode                0
Description              0
Quantity                 0
InvoiceDate              0
UnitPrice                0
CustomerID          134968
Country                  0
freq_description         0
count                    0
dtype: int64
```

We have successfully handle the invalid and the null Description values. We can ignore the CustomerID column because in the Sales Transaction it is not necessary to fix. Now we can drop the extra columns from our dataset.

```
df2.drop(columns= ["freq_description", "count"], inplace = True)
df2
```

```
{"type":"dataframe","variable_name":"df2"}
```

```
df2.describe()
```

```
{"summary":"{\n  \"name\": \"df2\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"Quantity\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 196373.7975324299,\n
\"min\": -80995.0,\n        \"max\": 541797.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
9.555918545137754,\n          10.0,\n          541797.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"InvoiceDate\",\n
\"properties\": {\n        \"dtype\": \"date\",\n        \"min\":
\"1970-01-01 00:00:00.000541797\",\n        \"max\": \"2011-12-09
12:50:00\",\n        \"num_unique_values\": 7,\n        \"samples\":
[\n          \"541797\",\n          \"2011-07-04
14:06:48.671255296\",\n          \"2011-10-19 11:41:00\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"UnitPrice\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
190712.59619572025,\n        \"min\": -11062.06,\n        \"max\":
541797.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n
4.612066833149686,\n          4.13,\n          541797.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"CustomerID\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
139204.16800694188,\n        \"min\": 1713.6003033216632,\n
\"max\": 406829.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          15287.690570239585,\n          16791.0,\n
406829.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```

We can see we have some Negative value in the Quantity and the UnitPrice columns. Quantity and UnitPrice can't be a negative value so we are considering them as an invalid value too.

```
df2[df2.Quantity<0]
```


```
{"summary":"{\n  \"name\": \"df2[df2\",\n  \"rows\": 10527,\n
\"fields\": [\n    {\n      \"column\": \"InvoiceNo\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 5075,\n        \"samples\": [\n
\"C550771\",\n          \"558449\",\n          \"C536854\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"StockCode\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 2462,\n
\"samples\": [\n          \"22068\",\n          \"21933\",\n
\"22227\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Description\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 2442,\n
\"samples\": [\n          \"SET OF 6 ICE CREAM SKITTLES\",\n
\"BLACK/WHITE GLASS/SILVER BRACELET\",\n          \"MINI JIGSAW CIRCUS
PARADE \"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Quantity\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1097,\n        \"min\": -80995,\n
\"max\": -1,\n        \"num_unique_values\": 329,\n
\"samples\": [\n          -10,\n          -65,\n          -66\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"InvoiceDate\",\n
\"properties\": {\n        \"dtype\": \"date\",\n        \"min\":
\"2010-12-01 09:41:00\",\n        \"max\": \"2011-12-09 11:58:00\",\n
\"num_unique_values\": 4798,\n        \"samples\": [\n
\"2011-01-21 16:58:00\",\n          \"2011-02-21 13:58:00\",\n
\"2011-04-27 16:28:00\"\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"UnitPrice\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 626.3344527561717,\n
\"min\": 0.0,\n        \"max\": 38970.0,\n
\"num_unique_values\": 574,\n        \"samples\": [\n
233.25,\n          5.45,\n          82.5\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"CustomerID\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1706.7723571934944,\n        \"min\": 12346.0,\n        \"max\":
18282.0,\n        \"num_unique_values\": 1589,\n        \"samples\":
[\n          12830.0,\n          15694.0,\n          12670.0\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Country\",\n      \"properties\": {\n        \"dtype\": \"string\",\
n        \"num_unique_values\": 30,\n        \"samples\": [\n
```

\"Greece\",\n   \"Netherlands\",\n   \"Czech Republic\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe"}

```
df2[df2.UnitPrice<0]
```

{"summary":"{\n \"name\": \"df2[df2\",\n \"rows\": 2,\n \"fields\": [\n   {\n   \"column\": \"InvoiceNo\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"A563187\",\n   \"A563186\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"StockCode\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 1,\n   \"samples\": [\n   \"B\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Description\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 1,\n   \"samples\": [\n   \"Adjust bad debt\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Quantity\",\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\": \"InvoiceDate\",\n   \"properties\": {\n   \"dtype\": \"date\",\n   \"min\": \"2011-08-12 14:51:00\",\n   \"max\": \"2011-08-12 14:52:00\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"2011-08-12 14:52:00\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"UnitPrice\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0.0,\n   \"min\": -11062.06,\n   \"max\": -11062.06,\n   \"num_unique_values\": 1,\n   \"samples\": [\n   -11062.06\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"CustomerID\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": null,\n   \"min\": null,\n   \"max\": null,\n   \"num_unique_values\": 0,\n   \"samples\": [],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Country\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 1,\n   \"samples\": [],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe"}

We are making another dataset except these invalid values.

```
df3 = df2[(df2.Quantity>0) & (df2.UnitPrice>0)]
df3
```


```
{"type":"dataframe","variable_name":"df3"}
```


```
df3.describe()
```


```
{"summary":"{\n  \"name\": \"df3\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"Quantity\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 185496.45754809008,\n
\"min\": 1.0,\n        \"max\": 530104.0,\n
\"num_unique_values\": 7,\n        \"samples\": [\n
530104.0,\n          10.542037034242338,\n          80995.0\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"InvoiceDate\",\n      \"properties\": {\n        \"dtype\":
\"date\",\n        \"min\": \"1970-01-01 00:00:00.000530104\",\n
\"max\": \"2011-12-09 12:50:00\",\n        \"num_unique_values\": 7,\n
\"samples\": [\n          \"530104\",\n          \"2011-07-04
20:16:05.225087744\",\n          \"2011-10-19 12:39:00\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"UnitPrice\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
186793.77949489455,\n        \"min\": 0.001,\n        \"max\":
530104.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n
3.90762524712132,\n          4.13,\n          530104.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"CustomerID\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
136042.15806984302,\n        \"min\": 1713.1415604398278,\n
\"max\": 397884.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          15294.423452564064,\n          16795.0,\n
397884.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```


In the Quantity Column we can see 75% of values is in 10.00 but the max value is 80995. We have to see whether it is an Outlier of not.

```
df3['Quantity'].quantile(0.9999)
```

```
1439.8763999990188
```

```
df3[df3.Quantity>1500]
```


```
{"summary":"{\n  \"name\": \"df3[df3\",\n  \"rows\": 41,\n
\"fields\": [\n    {\n      \"column\": \"InvoiceNo\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 34,\n        \"samples\": [\n
\"554868\",\n          \"562439\",\n          \"573261\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
```

```
n    },\n    {\n      \"column\": \"StockCode\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 26,\n        \"samples\": [\n
\"18007\",\n          \"84879\",\n          \"84950\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Description\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 26,\n        \"samples\": [\n
\"ESSENTIAL BALM 3.5g TIN IN ENVELOPE\",\n          \"ASSORTED COLOUR
BIRD ORNAMENT\",\n          \"ASSORTED COLOUR T-LIGHT HOLDER\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Quantity\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
16450,\n        \"min\": 1515,\n        \"max\": 80995,\n
\"num_unique_values\": 28,\n        \"samples\": [\n          3906,\n
1820,\n          2592\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"InvoiceDate\",\n      \"properties\": {\n        \"dtype\":
\"date\",\n        \"min\": \"2010-12-02 16:48:00\",\n        \"max\":
\"2011-12-09 09:15:00\",\n        \"num_unique_values\": 34,\n
\"samples\": [\n          \"2011-05-27 10:52:00\",\n          \"2011-
08-04 18:06:00\",\n          \"2011-10-28 12:32:00\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"UnitPrice\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.7329412015148614,\n        \"min\": 0.06,\n        \"max\": 2.55,\n
\"num_unique_values\": 24,\n        \"samples\": [\n          0.21,\n
0.72,\n          0.55\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"CustomerID\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1684.036697937429,\n        \"min\":
12346.0,\n        \"max\": 18087.0,\n        \"num_unique_values\":
22,\n        \"samples\": [\n          15299.0,\n          13694.0,\n
18087.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Country\",\n      \"properties\": {\n        \"dtype\": \"string\",\
n        \"num_unique_values\": 3,\n        \"samples\": [\n
\"United Kingdom\",\n          \"Netherlands\",\n          \"Japan\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe"}
```


We are assuming these values are correct so we are not touching it.

Feature Engineering : Creating New Columns

```
df4 = df3.copy()

df4["Total_Sales"] = df4["Quantity"] * df4["UnitPrice"]
df4
```

{"type":"dataframe","variable_name":"df4"}

```
df4.describe()
```

{"summary":"{\n \"name\": \"df4\",\n \"rows\": 8,\n \"fields\": [\n{\n   \"column\": \"Quantity\",\n   \"properties\": {\n\"dtype\": \"number\",\n   \"std\": 185496.45754809008,\n\"min\": 1.0,\n   \"max\": 530104.0,\n\"num_unique_values\": 7,\n   \"samples\": [\n530104.0,\n   10.542037034242338,\n   80995.0\n   ],\n   \"semantic_type\": \"\",\n\"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"InvoiceDate\",\n   \"properties\": {\n   \"dtype\": \"date\",\n   \"min\": \"1970-01-01 00:00:00.000530104\",\n\"max\": \"2011-12-09 12:50:00\",\n   \"num_unique_values\": 7,\n\"samples\": [\n   \"530104\",\n   \"2011-07-04 20:16:05.225087744\",\n   \"2011-10-19 12:39:00\"\n   ],\n\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"UnitPrice\",\n\"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 186793.77949489455,\n   \"min\": 0.001,\n   \"max\": 530104.0,\n   \"num_unique_values\": 8,\n   \"samples\": [\n3.90762524712132,\n   4.13,\n   530104.0\n   ],\n\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"CustomerID\",\n\"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 136042.15806984302,\n   \"min\": 1713.1415604398278,\n\"max\": 397884.0,\n   \"num_unique_values\": 8,\n\"samples\": [\n   15294.423452564064,\n   16795.0,\n397884.0\n   ],\n   \"semantic_type\": \"\",\n\"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Total_Sales\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 188352.05568719542,\n   \"min\": 0.001,\n   \"max\": 530104.0,\n   \"num_unique_values\": 8,\n   \"samples\": [\n   20.121871451639677,\n17.700000000000003,\n   530104.0\n   ],\n\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe"}

```
df4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 530104 entries, 0 to 541908
Data columns (total 9 columns):
 #   Column       Non-Null Count   Dtype 
---  ------       --------------   ----- 
 0   InvoiceNo    530104 non-null  string
 1   StockCode    530104 non-null  string
 2   Description  530104 non-null  string
 3   Quantity     530104 non-null  int64 
```

```
 4   InvoiceDate  530104 non-null  datetime64[ns]
 5   UnitPrice    530104 non-null  float64
 6   CustomerID   397884 non-null  float64
 7   Country      530104 non-null  string
 8   Total_Sales  530104 non-null  float64
dtypes: datetime64[ns](1), float64(3), int64(1), string(4)
memory usage: 40.4 MB
```

```
df4["Month"] = df4.InvoiceDate.dt.month
# df4["Day"] = df4.InvoiceDate.dt.day
# df4["Hour"] = df4.InvoiceDate.dt.hour
df4.sample(5)
```

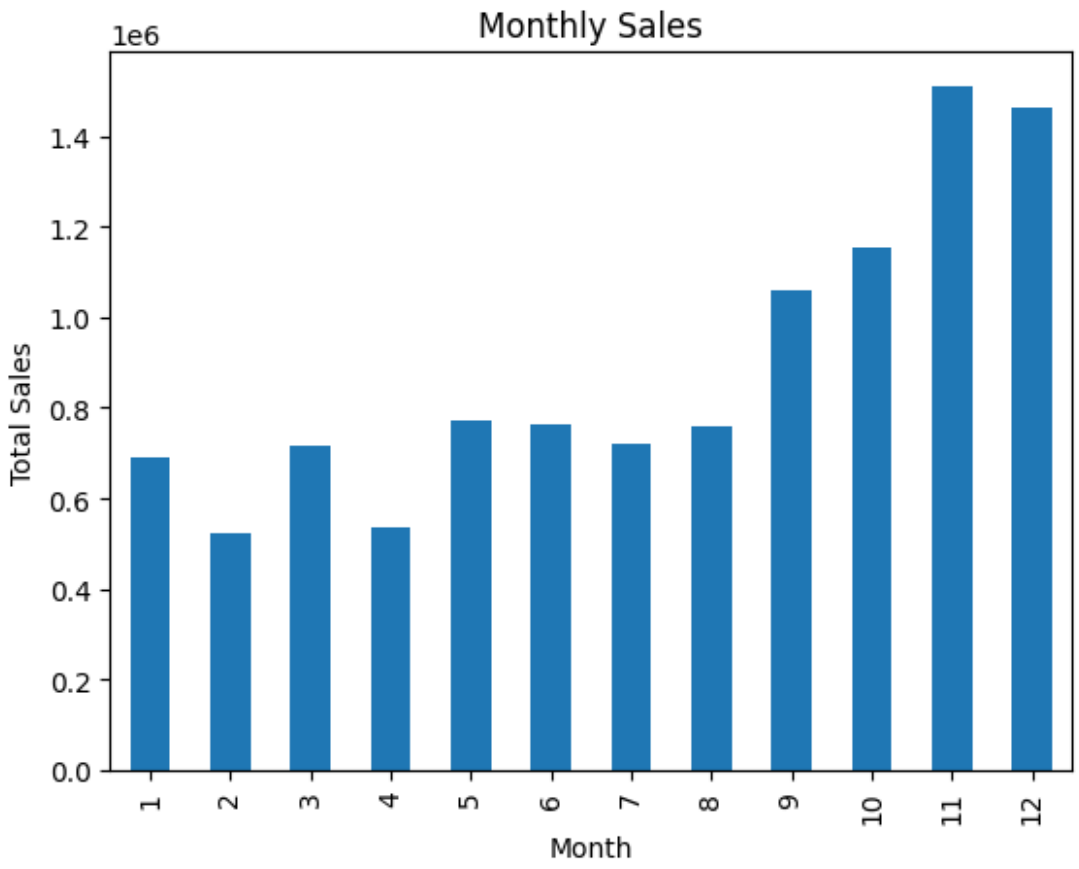

```
{"summary":"{\n  \"name\": \"df4\",\n  \"rows\": 5,\n  \"fields\": [\n{\n      \"column\": \"InvoiceNo\",\n      \"properties\": {\n\"dtype\": \"string\",\n        \"num_unique_values\": 5,\n\"samples\": [\n          \"552493\",\n          \"544087\",\n\"544677\"\n        ],\n        \"semantic_type\": \"\",\n\"description\": \"\"\n      }\n    },\n    {\n      \"column\":\"StockCode\",\n      \"properties\": {\n        \"dtype\":\"string\",\n        \"num_unique_values\": 5,\n        \"samples\":[\n          \"84946\",\n          \"22940\",\n          \"21447\"\n],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n}\n    },\n    {\n      \"column\": \"Description\",\n\"properties\": {\n        \"dtype\": \"string\",\n\"num_unique_values\": 5,\n        \"samples\": [\n          \"ANTIQUESILVER T-LIGHT GLASS\",\n          \"FELTCRAFT CHRISTMAS FAIRY\",\n\"12 IVORY ROSE PEG PLACE SETTINGS\"\n        ],\n\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Quantity\",\n      \"properties\":{\n        \"dtype\": \"number\",\n        \"std\": 63,\n\"min\": 1,\n        \"max\": 144,\n        \"num_unique_values\": 3,\n        \"samples\": [\n          144,\n          1,\n          2\n],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n}\n    },\n    {\n      \"column\": \"InvoiceDate\",\n\"properties\": {\n        \"dtype\": \"date\",\n        \"min\":\"2011-02-15 17:02:00\",\n        \"max\": \"2011-08-16 13:05:00\",\n\"num_unique_values\": 5,\n        \"samples\": [\n          \"2011-05-09 16:21:00\",\n          \"2011-02-15 17:02:00\",\n\"2011-02-22 16:13:00\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n\"column\": \"UnitPrice\",\n      \"properties\": {\n\"dtype\": \"number\",\n        \"std\": 1.1085125168440815,\n\"min\": 1.25,\n        \"max\": 4.25,\n        \"num_unique_values\":5,\n        \"samples\": [\n          2.46,\n          4.25,\n1.25\n        ],\n        \"semantic_type\": \"\",\n\"description\": \"\"\n      }\n    },\n    {\n      \"column\":\"CustomerID\",\n      \"properties\": {\n        \"dtype\":\"number\",\n        \"std\": 1886.951333059052,\n        \"min\":14646.0,\n        \"max\": 18196.0,\n        \"num_unique_values\":
```

```
3,\n        \"samples\": [\n          14646.0,\n          18196.0,\n
17530.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Country\",\n      \"properties\": {\n        \"dtype\": \"string\",\
n        \"num_unique_values\": 2,\n        \"samples\": [\n
\"United Kingdom\",\n          \"Netherlands\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Total_Sales\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
162.14294964629207,\n        \"min\": 1.25,\n        \"max\": 367.2,\n
\"num_unique_values\": 5,\n        \"samples\": [\n          2.46,\n
8.5\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Month\",\n      \"properties\": {\n        \"dtype\": \"int32\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n          5,\n
8\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```

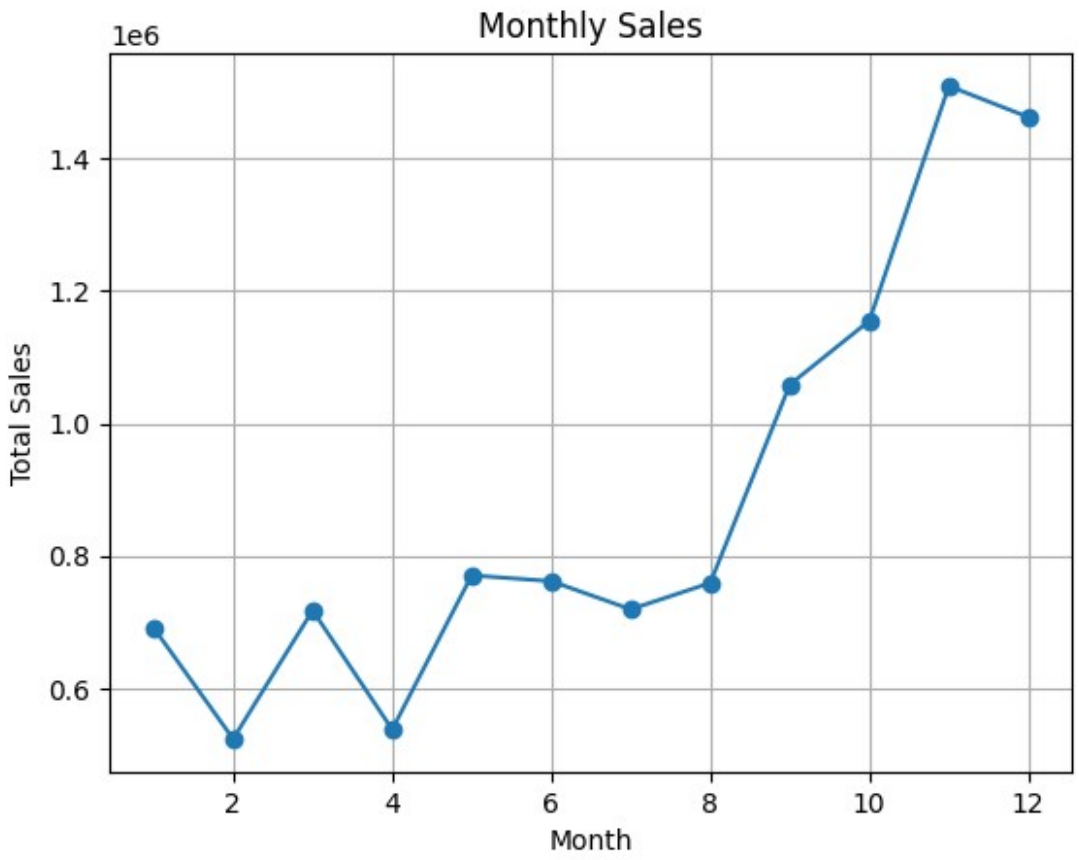

Visualization & EDA

1. Plot Monthly Sales

```
monthly_sales = df4.groupby("Month")["Total_Sales"].sum()
monthly_sales.plot(kind = "bar", title = "Monthly Sales")
plt.xlabel("Month")
plt.ylabel("Total Sales")
plt.show()
```




```
monthly_sales.plot(kind = "line", title = "Monthly Sales", marker =
"o")
plt.xlabel("Month")
plt.ylabel("Total Sales")
plt.grid()
plt.show()
```

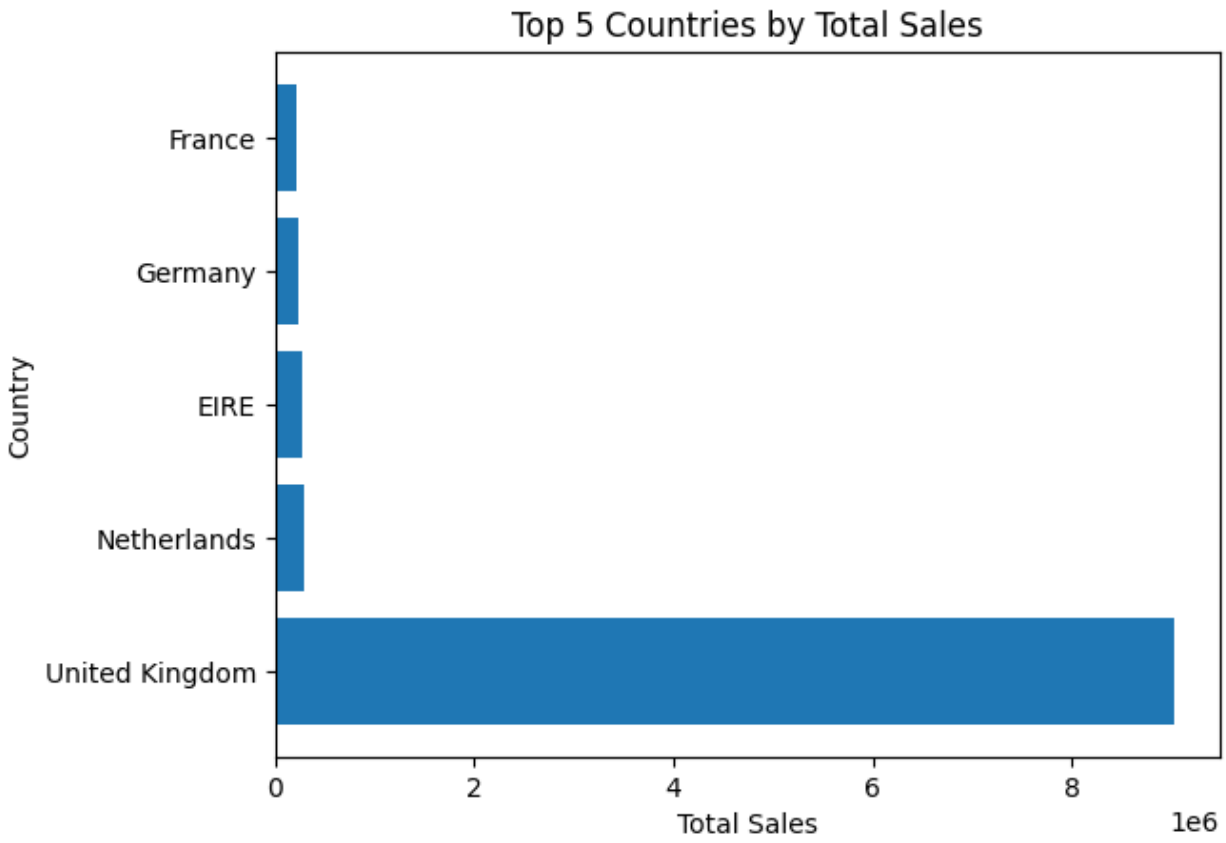


** Insights **

Total Sales started raising up in Auguest having a peek in November. This is likely due to the holiday season at the end of the year.

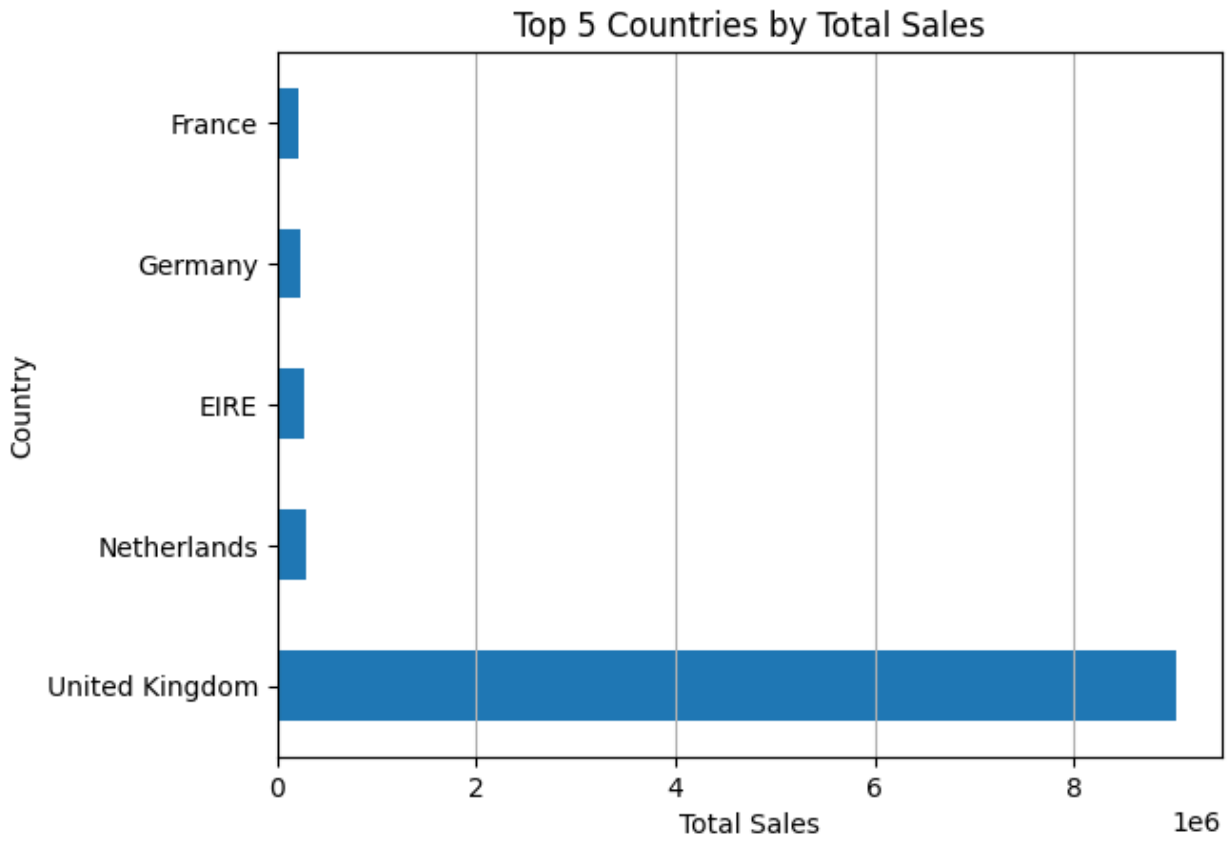
2. Top 5 Countries by Sales

```
top_countries = df4.groupby('Country')
['Total_Sales'].sum().nlargest(5)
plt.barh(top_countries.index, top_countries.values)
plt.xlabel('Total Sales')
plt.ylabel('Country')
plt.title('Top 5 Countries by Total Sales')
plt.show()
```



Do it by my self.

```
top_countries = df4.groupby('Country')
['Total_Sales'].sum().sort_values(ascending=False).head(5)
top_countries.plot(kind = "barh", title = "Top 5 Countries by Total
Sales")
plt.xlabel("Total Sales")
plt.ylabel("Country")
plt.grid(axis = "x")
plt.show()
```



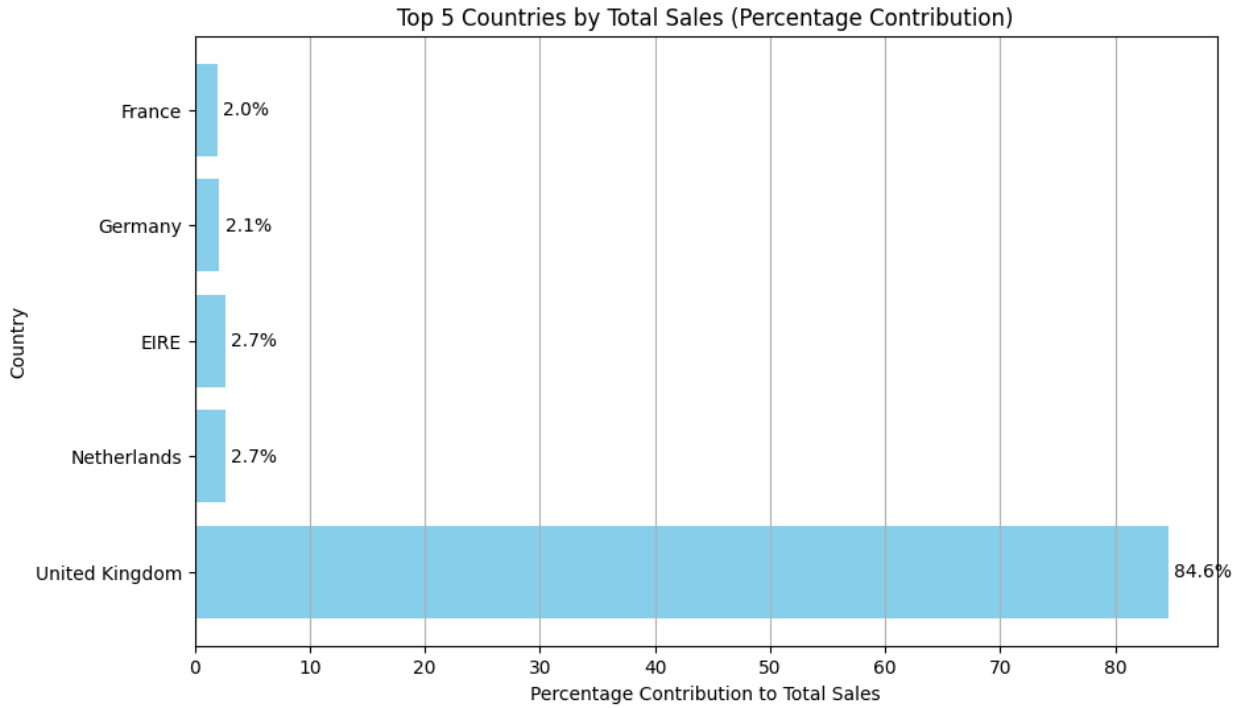
Insights

1. UK has the highest sales (around 9 million)
2. Netherlands, EIRE, Germany and France are the next 4 countries each having a sales of more than 2 million

Since these countries cover the major sales revenues, we need to pay special attention to customers in these countries and make sure our product quality and service are the best. Also to break dependancy of sales from a single country we can focus on expanding sales in other countries as well

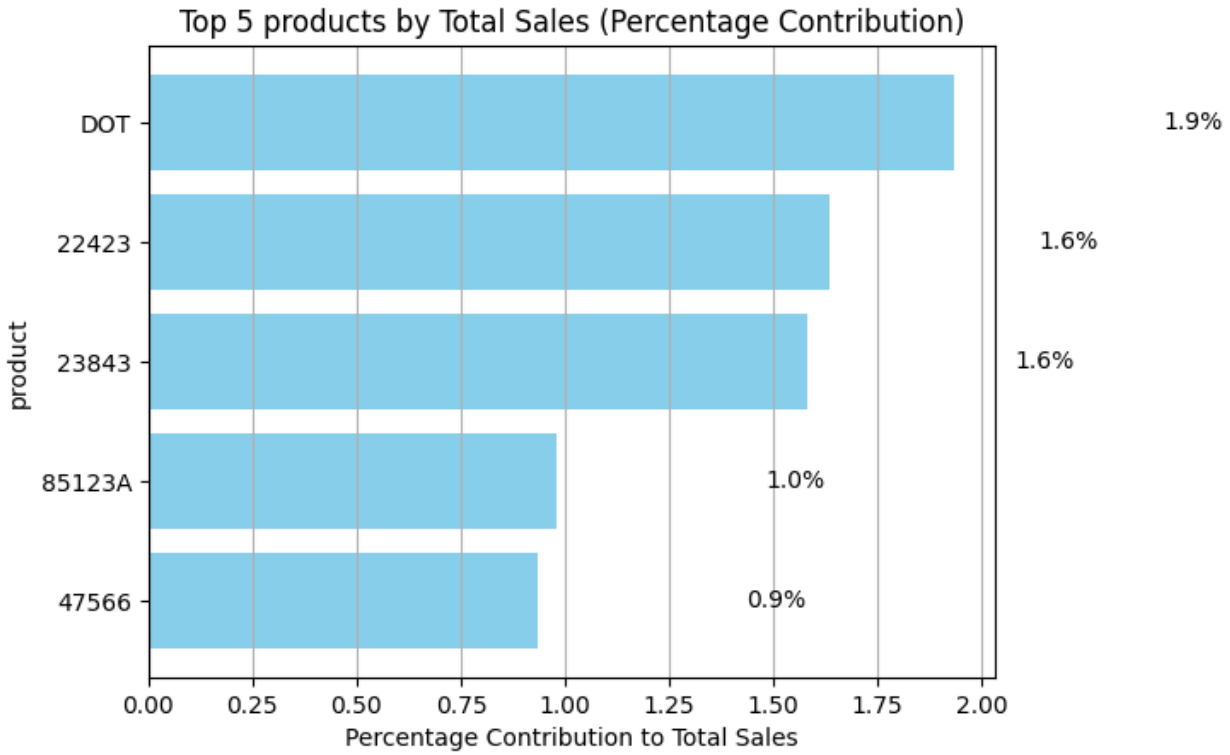
```
country_wise_sales = df4.groupby("Country")["Total_Sales"].sum()

total_sales = country_wise_sales.sum()


top_5_countries = country_wise_sales.nlargest(5)

percentages = (top_5_countries / total_sales) * 100

plt.figure(figsize=(10, 6))  # Adjust figure size for better
readability
bars = plt.barh(top_5_countries.index, percentages, color='skyblue')
plt.xlabel('Percentage Contribution to Total Sales')
```

```
plt.ylabel('Country')
plt.title('Top 5 Countries by Total Sales (Percentage Contribution)')

# Add percentage labels to the bars
for bar, percentage in zip(bars, percentages):
    plt.text(bar.get_width() + 0.5, bar.get_y() + bar.get_height() /
2, f'{percentage:.1f}%', va='center', ha='left', color='black',
fontsize=10)


plt.grid(axis='x')  # Add grid lines
plt.show()
```




```
product_wise_sales = df4.groupby("StockCode")["Total_Sales"].sum()

total_sales = product_wise_sales.sum()


top_5_products = product_wise_sales.nlargest(5)

percentages = (top_5_products / total_sales) * 100

bars = plt.barh(top_5_products.index, percentages, color='skyblue')
plt.xlabel('Percentage Contribution to Total Sales')
plt.ylabel('product')
plt.title('Top 5 products by Total Sales (Percentage Contribution)')

# Add percentage labels to the bars
```

```
for bar, percentage in zip(bars, percentages):
    plt.text(bar.get_width() + 0.5, bar.get_y() + bar.get_height() /
2, f'{percentage:.1f}%', va='center', ha='left', color='black',
fontsize=10)

plt.gca().invert_yaxis()  # Invert the y-axis to display the highest
percentage at the top
plt.grid(axis='x')  # Add grid lines
plt.show()
```




```
product_wise_sales.sort_values(ascending=False).head(5)
```

```
StockCode
DOT       206248.77
22423     174484.74
23843     168469.60
85123A    104518.80
47566      99504.33
Name: Total_Sales, dtype: float64
```

```
top_products = df4.groupby('StockCode')
['Total_Sales'].sum().sort_values(ascending=False).head(5)

for stock_code in top_products.index:
    description = df4[df4['StockCode'] == stock_code]
['Description'].iloc[0]
```

```
    total_sales = top_products[stock_code]
    print(f"StockCode: {stock_code}, Description: {description}, Total
Sales: {total_sales}")
```

```
StockCode: DOT, Description: DOTCOM POSTAGE, Total Sales: 206248.77
StockCode: 22423, Description: REGENCY CAKESTAND 3 TIER, Total Sales:
174484.74
StockCode: 23843, Description: PAPER CRAFT , LITTLE BIRDIE, Total
Sales: 168469.6
StockCode: 85123A, Description: WHITE HANGING HEART T-LIGHT HOLDER,
Total Sales: 104518.8
StockCode: 47566, Description: PARTY BUNTING, Total Sales: 99504.33
```

We don't have the dependency on a single product which is a good sign.

4. RFM Analysis

```
df4["InvoiceDate"].max()
```

```
Timestamp('2011-12-09 12:50:00')
```

We are adding one more day with the last date for our current date.

```
current_dt = df4["InvoiceDate"].max() + pd.Timedelta(days = 1)
current_dt
```

```
Timestamp('2011-12-10 12:50:00')
```

```
rfm = df4.groupby("CustomerID").agg({
    "InvoiceDate" : lambda x : (current_dt - x.max()).days,
    "InvoiceNo" : "count",
    "Total_Sales" : "sum"
})

rfm.columns = ["Recency", "Frequency", "Monetary"]

rfm
```


```
{"summary":"{\n  \"name\": \"rfm\",\n  \"rows\": 4338,\n  \"fields\":
[\n    {\n      \"column\": \"CustomerID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 1721.8084917653164,\n
\"min\": 12346.0,\n        \"max\": 18287.0,\n
\"num_unique_values\": 4338,\n        \"samples\": [\n
17785.0,\n          14320.0,\n          15977.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Recency\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 100,\n
\"min\": 1,\n        \"max\": 374,\n        \"num_unique_values\":
349,\n        \"samples\": [\n          187,\n          118,\n
69\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

```
\"Frequency\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 228,\n        \"min\": 1,\n
\"max\": 7847,\n        \"num_unique_values\": 460,\n
\"samples\": [\n          42,\n          27,\n          200\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Monetary\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 8989.230441338677,\n        \"min\":
3.75,\n        \"max\": 280206.02,\n        \"num_unique_values\":
4253,\n        \"samples\": [\n          2794.51,\n          379.35,\n
954.09\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"rfm"}
```

Verify the Recency, Frequency and Monetary in Manual

```
df4[df4.CustomerID==18283.0]["InvoiceDate"].max()
```

```
Timestamp('2011-12-06 12:02:00')
```

```
df4[df4.CustomerID == 18283]["InvoiceNo"].count()
```

```
756
```

```
df4[df4.CustomerID == 18283]["Total_Sales"].sum()
```

```
2094.88
```

Here we can see that CustomerID of "18283"'s last InvoiceDate is 6th dec, 2011

Current date is 10th dec, 2011

So, the recency is (10-6) = 4days ["Varified"]

Frequency = 756days ["Varified"]

Monetary = 2094.88$ ["Varified"]

```
# Segment Customers based on RFM
rfm["R_Segment"] = pd.qcut(rfm["Recency"], 4, labels = [4,3,2,1])
rfm["F_Segment"] = pd.qcut(rfm["Frequency"], 4, labels = [1,2,3,4])
rfm["M_Segment"] = pd.qcut(rfm["Monetary"], 4, labels = [1,2,3,4])
rfm["RFM_Score"] = rfm[["R_Segment", "F_Segment",
"M_Segment"]].sum(axis=1)
rfm
```

```
{"summary":"{\n  \"name\": \"rfm\",\n  \"rows\": 4338,\n  \"fields\":
[\n    {\n      \"column\": \"CustomerID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 1721.8084917653164,\n
\"min\": 12346.0,\n        \"max\": 18287.0,\n
\"num_unique_values\": 4338,\n        \"samples\": [\n
```

```
17785.0,\n          14320.0,\n          15977.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Recency\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 100,\n
\"min\": 1,\n        \"max\": 374,\n        \"num_unique_values\":
349,\n        \"samples\": [\n          187,\n          118,\n
69\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Frequency\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 228,\n        \"min\": 1,\n
\"max\": 7847,\n        \"num_unique_values\": 460,\n
\"samples\": [\n          42,\n          27,\n          200\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Monetary\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 8989.230441338677,\n        \"min\":
3.75,\n        \"max\": 280206.02,\n        \"num_unique_values\":
4253,\n        \"samples\": [\n          2794.51,\n          379.35,\n
954.09\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"R_Segment\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          4,\n          3,\n          1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"F_Segment\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n          4,\n
3,\n          1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"M_Segment\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          2,\n          3,\n          4\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"RFM_Score\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
2,\n        \"min\": 3,\n        \"max\": 12,\n
\"num_unique_values\": 10,\n        \"samples\": [\n          9,\n
12,\n          3\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"rfm"}

# Customers with high RFM Scores
rfm.sort_values("RFM_Score", ascending=False)

{"summary":"{\n  \"name\": \"rfm\",\n  \"rows\": 4338,\n  \"fields\":
[\n    {\n      \"column\": \"CustomerID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 1721.808491765317,\n
\"min\": 12346.0,\n        \"max\": 18287.0,\n
\"num_unique_values\": 4338,\n        \"samples\": [\n
16097.0,\n          18034.0,\n          17600.0\n        ],\n
```

```
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Recency\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 100,\n
\"min\": 1,\n        \"max\": 374,\n        \"num_unique_values\":
349,\n        \"samples\": [\n          163,\n          252,\n
250\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Frequency\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 228,\n        \"min\": 1,\n
\"max\": 7847,\n        \"num_unique_values\": 460,\n
\"samples\": [\n          114,\n          103,\n          135\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Monetary\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
8989.23044133866,\n        \"min\": 3.75,\n        \"max\":
280206.02,\n        \"num_unique_values\": 4253,\n        \"samples\":
[\n          8347.2,\n          437.23,\n          663.61\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"R_Segment\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n          3,\n
1,\n          4\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"F_Segment\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          3,\n          1,\n          4\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"M_Segment\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n          3,\n
1,\n          4\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"RFM_Score\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 2,\n        \"min\": 3,\n
\"max\": 12,\n        \"num_unique_values\": 10,\n        \"samples\":
[\n          4,\n          11,\n          7\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```


If the RFM Scores are high than they are my Valuable Customers.

```
# Create a basket matrix for association rule mining
customer_last_purchase = df4.groupby("CustomerID")
["InvoiceDate"].max()
customer_last_purchase.head()
```

```
CustomerID
12346.0   2011-01-18 10:01:00
12347.0   2011-12-07 15:52:00
```

```
12348.0   2011-09-25 13:13:00
12349.0   2011-11-21 09:51:00
12350.0   2011-02-02 16:01:00
Name: InvoiceDate, dtype: datetime64[ns]
```

```
# Recency Value
customer_last_purchase = (current_dt - customer_last_purchase).dt.days
customer_last_purchase.head()
```

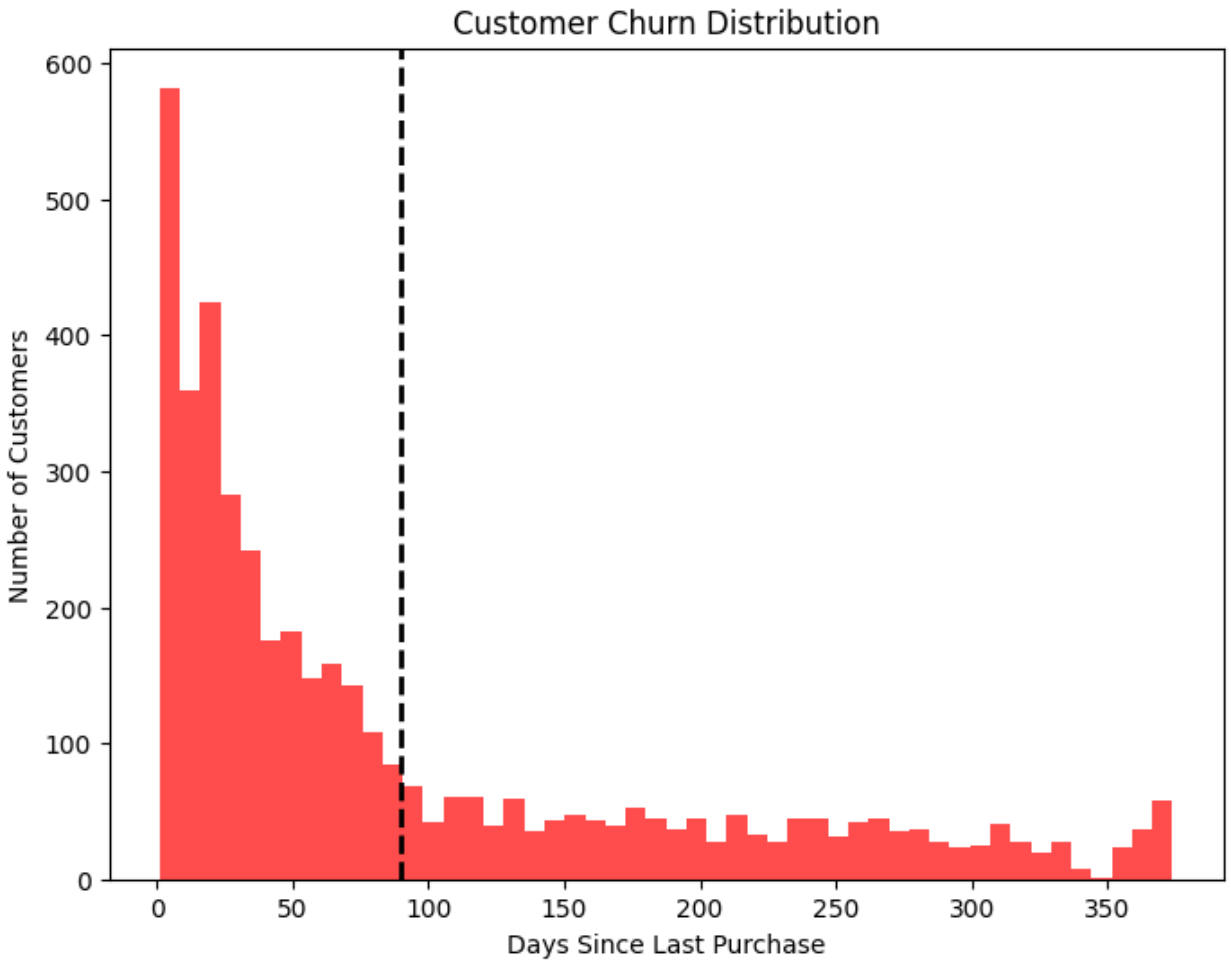
```
CustomerID
12346.0    326
12347.0      2
12348.0     75
12349.0     19
12350.0    310
Name: InvoiceDate, dtype: int64
```

```
# Define churn threshold (e.g., 90 days without purchase)
churn_threshold = 90
churned_customers = customer_last_purchase[customer_last_purchase >
churn_threshold]
churned_customers.head()
```

```
CustomerID
12346.0    326
12350.0    310
12353.0    204
12354.0    232
12355.0    214
Name: InvoiceDate, dtype: int64
```

```
print("Number of Churned Customers:", len(churned_customers))
plt.figure(figsize = (8,6))
plt.hist(customer_last_purchase, bins = 50, color = "red", alpha =
0.7)
plt.axvline(churn_threshold, color = "black", linestyle = "dashed",
linewidth = 2)
plt.title("Customer Churn Distribution")
plt.xlabel("Days Since Last Purchase")
plt.ylabel("Number of Customers")
plt.show()
```

```
Number of Churned Customers: 1449
```




```
print(f"[{pd.__name__}] total time taken: {time.time() - stime} sec")
```

```
[pandas] total time taken: 74.62005829811096 sec
```