```
import pandas as pd
response=pd.read_csv("/content/Retail_Data_Response.csv")
print(response.head())
```

```
  customer_id  response
0      CS1112         0
1      CS1113         0
2      CS1114         1
3      CS1115         1
4      CS1116         1
```

```
import pandas as pd
txn=pd.read_csv("/content/Retail_Data_Transactions.csv")
print(txn.head())
```

```
  customer_id trans_date  tran_amount
0      CS5295  11-Feb-13           35
1      CS4768  15-Mar-15           39
2      CS2122  26-Feb-13           52
3      CS1217  16-Nov-11           99
4      CS1850  20-Nov-13           78
```

```
df=txn.merge(response,on="customer_id",how="left")
print(df)
```

```
       customer_id trans_date  tran_amount  response
0           CS5295  11-Feb-13           35       1.0
1           CS4768  15-Mar-15           39       1.0
2           CS2122  26-Feb-13           52       0.0
3           CS1217  16-Nov-11           99       0.0
4           CS1850  20-Nov-13           78       0.0
...            ...         ...          ...       ...
124995      CS8433  26-Jun-11           64       0.0
124996      CS7232  19-Aug-14           38       0.0
124997      CS8731  28-Nov-14           42       0.0
124998      CS8133  14-Dec-13           13       0.0
124999      CS7996  13-Dec-14           36       0.0

[125000 rows x 4 columns]
```

```
df.dtypes
df.shape
df.tail()
```

```
{"repr_error":"0","type":"dataframe"}
```

```
df.describe()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"tran_amount\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 44174.089911097886,\n
```

```
\"min\": 10.0,\n        \"max\": 125000.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
64.991912,\n          65.0,\n          125000.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"response\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
44183.14171702352,\n        \"min\": 0.0,\n        \"max\": 124969.0,\
n        \"num_unique_values\": 5,\n        \"samples\": [\n
0.11076346934039642,\n          1.0,\n          0.31384026408581317\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe"}
```

```
df.isnull().sum()
```

```
customer_id     0
trans_date      0
tran_amount     0
response       31
dtype: int64
```

```
df=df.dropna()
df
```

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

```
df['trans_date']=pd.to_datetime(df['trans_date'])
df
```

```
<ipython-input-8-6efe208ba6b4>:1: UserWarning: Could not infer format,
so each element will be parsed individually, falling back to
`dateutil`. To ensure parsing is consistent and as-expected, please
specify a format.
  df['trans_date']=pd.to_datetime(df['trans_date'])
```

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

```
set(df['response'])
```

```
{0.0, 1.0}
```

```
df.dtypes
```

```
customer_id            object
trans_date     datetime64[ns]
tran_amount             int64
response              float64
dtype: object
```

```
import pandas as pd
from scipy import stats
import numpy as np
z_score = np.abs(stats.zscore(df["response"]))
```

```
threshold = 3
outliers = z_score > threshold
print(df[outliers])
```

```
Empty DataFrame
Columns: [customer_id, trans_date, tran_amount, response]
Index: []
```

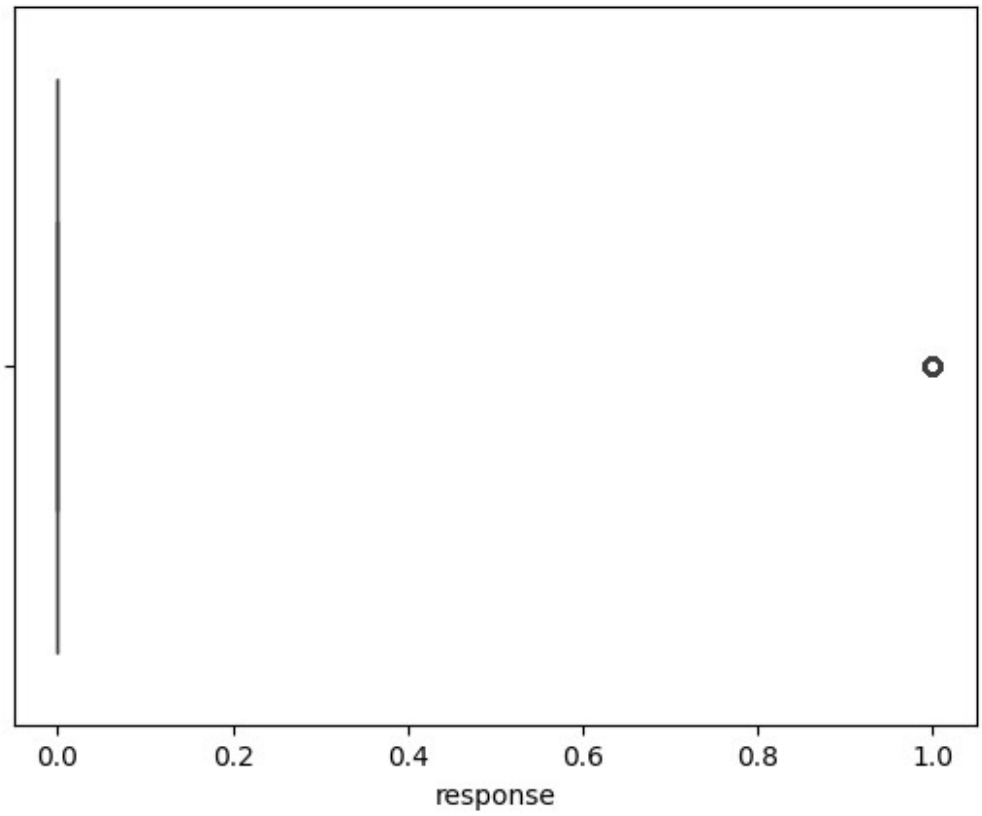
```
df
```

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

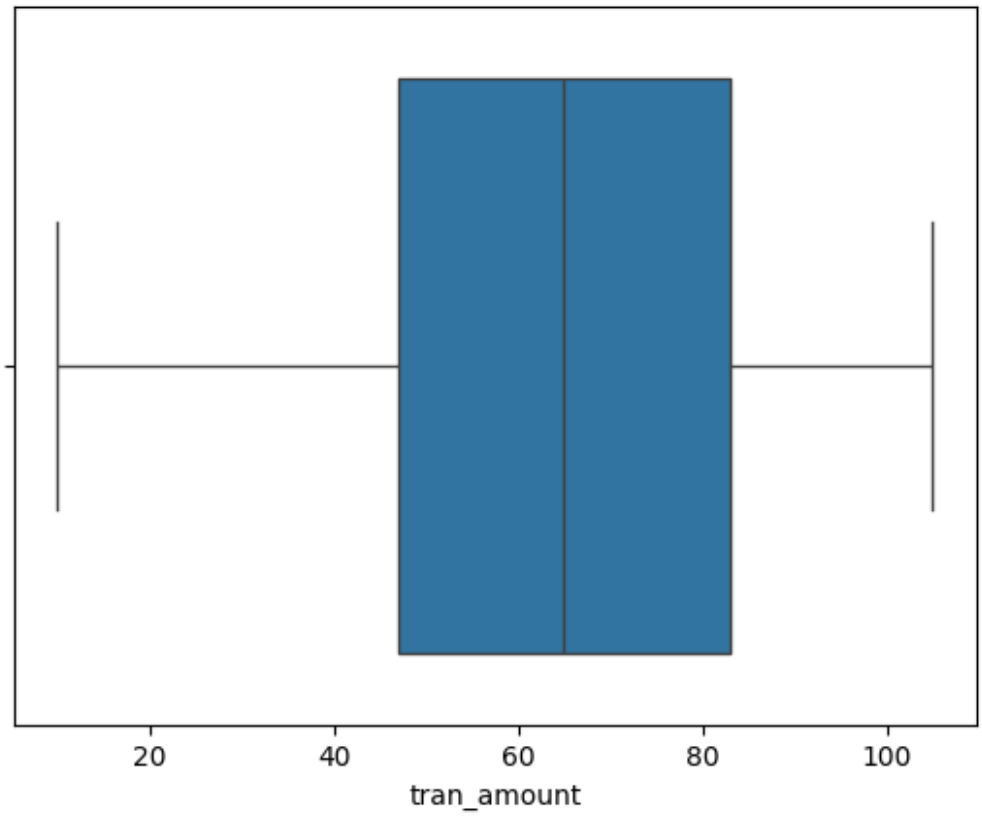
```
import pandas as pd
from scipy import stats
import numpy as np
z_score = np.abs(stats.zscore(df["tran_amount"]))
threshold = 3
outliers = z_score > threshold
print(df[outliers])
```

```
Empty DataFrame
Columns: [customer_id, trans_date, tran_amount, response]
Index: []
```

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(x=df["response"])
plt.show()
```




```
import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(x=df["tran_amount"])
plt.show()
```




```
import pandas as pd
df['trans_date'] = pd.to_datetime(df['trans_date'])
df['month'] = df['trans_date'].dt.month
print(df)
```

```
       customer_id trans_date  tran_amount  response  month
0           CS5295 2013-02-11           35       1.0      2
1           CS4768 2015-03-15           39       1.0      3
2           CS2122 2013-02-26           52       0.0      2
3           CS1217 2011-11-16           99       0.0     11
4           CS1850 2013-11-20           78       0.0     11
...            ...        ...          ...       ...    ...
124995      CS8433 2011-06-26           64       0.0      6
124996      CS7232 2014-08-19           38       0.0      8
124997      CS8731 2014-11-28           42       0.0     11
124998      CS8133 2013-12-14           13       0.0     12
124999      CS7996 2014-12-13           36       0.0     12

[124969 rows x 5 columns]
```

```
top_3_months=df.sort_values(by="tran_amount",ascending=False).head(3)
print(top_3_months)
```

```
      customer_id trans_date  tran_amount  response  month
92202      CS4180 2014-05-16          105       0.0      5
```

```
46209      CS2636 2013-10-24          105       0.0     10
92956      CS4005 2014-05-28          105       0.0      5
```

```
monthly_sales=df.groupby("month")["tran_amount"].sum()
monthly_sales=monthly_sales.sort_values(ascending=False).head(3)
print(monthly_sales)
```

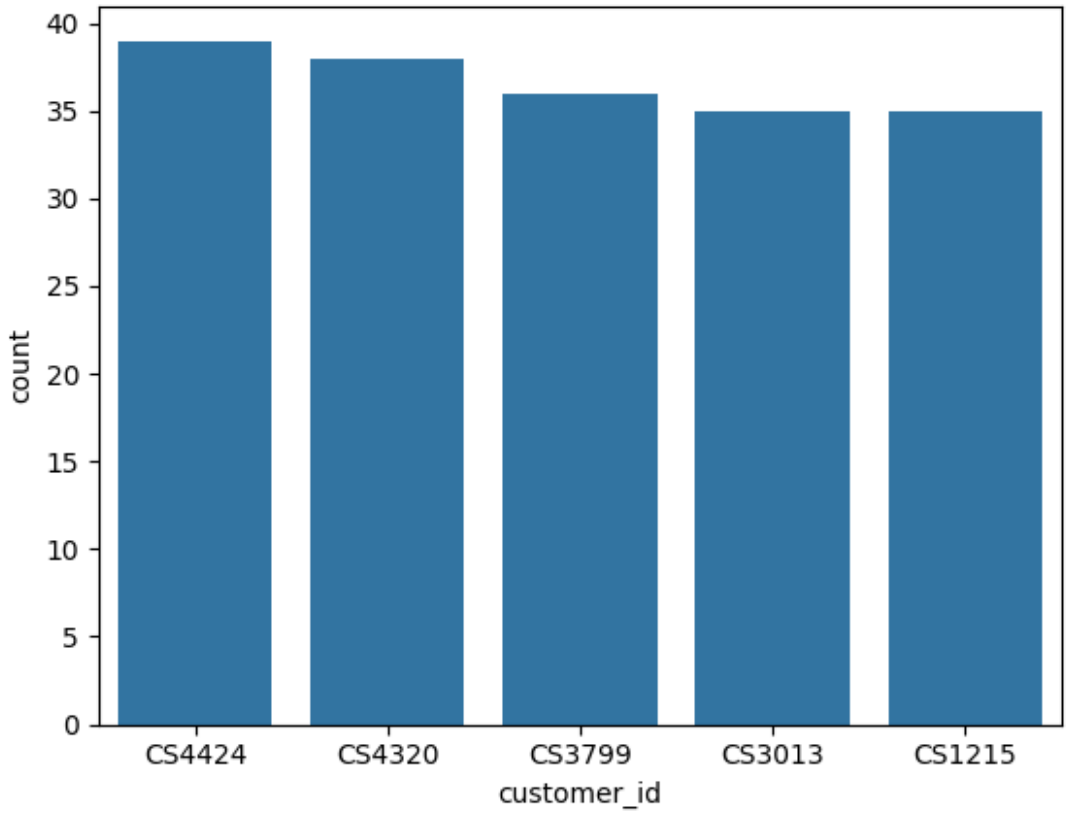
```
month
8     726775
10    725058
1     724089
Name: tran_amount, dtype: int64
```

```
Customer_counts=df["customer_id"].value_counts().reset_index()
Customer_counts.column=['customer_id','count']
top_5_customers=Customer_counts.sort_values(by='count',ascending=False
).head(5)
print(top_5_customers)
```

```
  customer_id  count
0      CS4424     39
1      CS4320     38
2      CS3799     36
3      CS3013     35
4      CS1215     35
```

```
<ipython-input-21-685ea7c8a10f>:2: UserWarning: Pandas doesn't allow
columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-
access
  Customer_counts.column=['customer_id','count']
```

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.barplot(x="customer_id", y="count", data=top_5_customers)
plt.show()
```

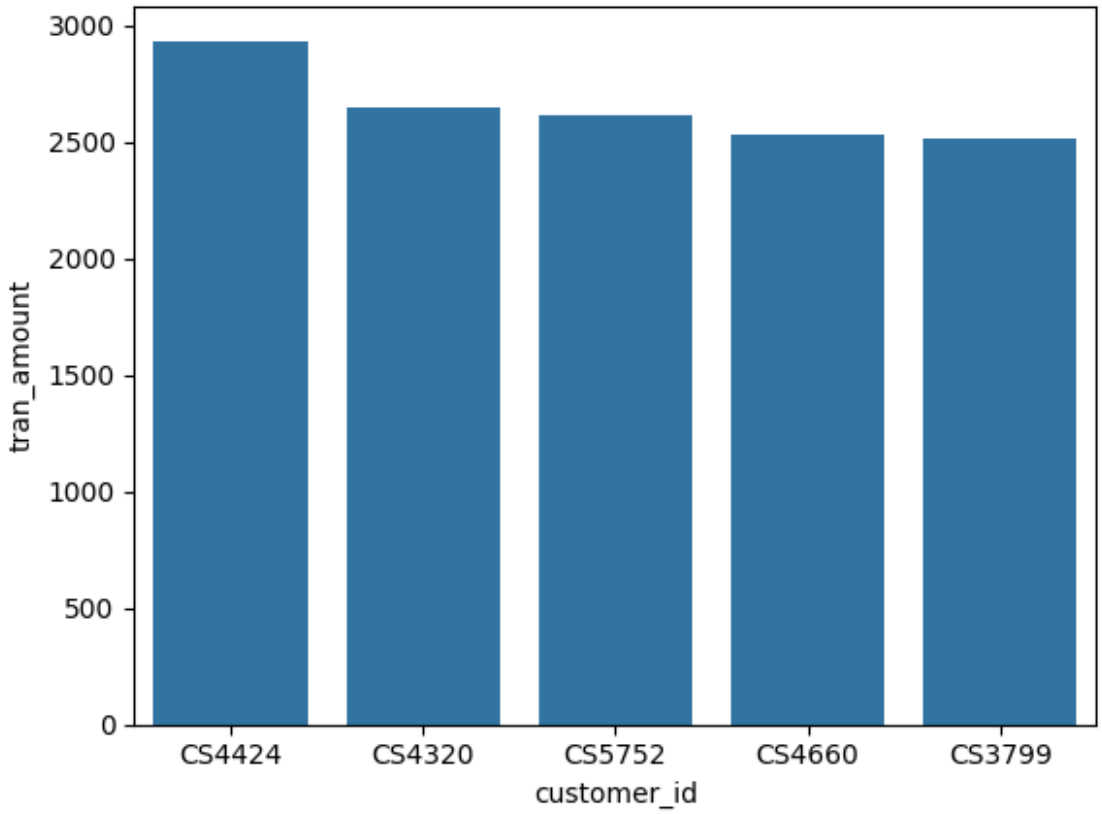



```
customer_sales =df.groupby("customer_id")
["tran_amount"] .sum().reset_index()
print(customer_sales)
top_5_sales = customer_sales.sort_values(by="tran_amount",
ascending=False).head(5)

print(top_5_customers)
```

```
     customer_id  tran_amount
0         CS1112         1012
1         CS1113         1490
2         CS1114         1432
3         CS1115         1659
4         CS1116          857
...          ...          ...
6879      CS8996          582
6880      CS8997          543
6881      CS8998          624
6882      CS8999          383
6883      CS9000          533

[6884 rows x 2 columns]
  customer_id  count
```

```
0      CS4424     39
1      CS4320     38
2      CS3799     36
3      CS3013     35
4      CS1215     35
```

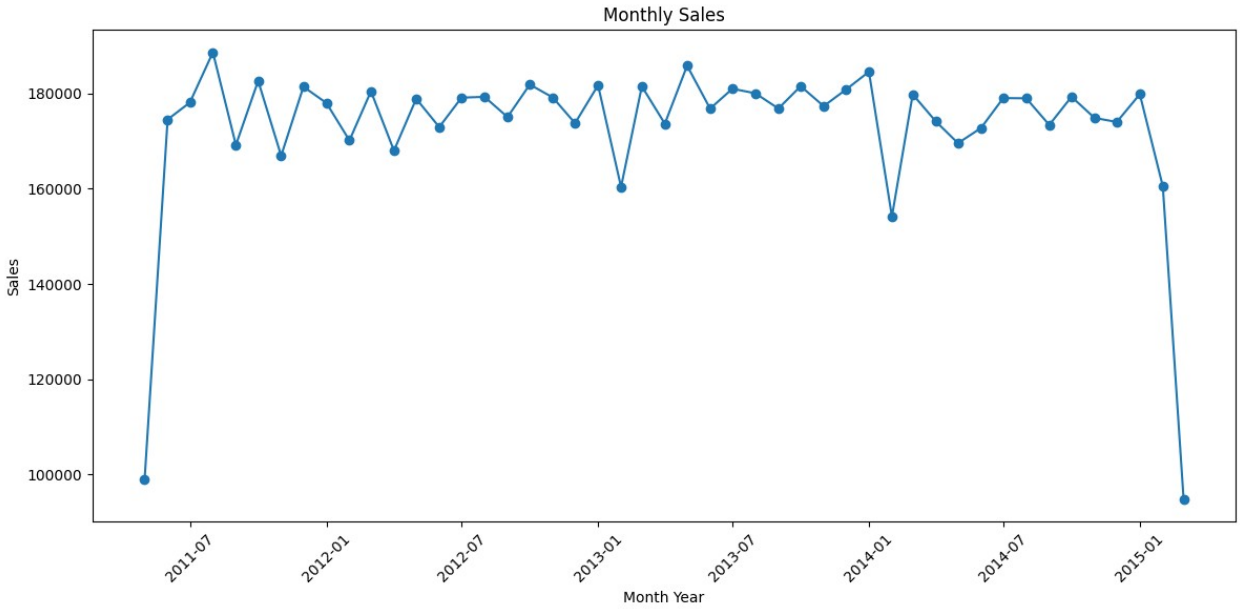
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.barplot(x="customer_id", y="tran_amount", data=top_5_sales)
plt.show()
```




```
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import pandas as pd
response = pd.read_csv("/content/Retail_Data_Response.csv")
trnx = pd.read_csv("/content/Retail_Data_Transactions.csv")
df = trnx.merge(response, on="customer_id", how="left")
df["trans_date"] = pd.to_datetime(df["trans_date"], errors='coerce')
df['month_year'] = df['trans_date'].dt.to_period('M')
monthly_sales = df.groupby("month_year")["tran_amount"].sum()
monthly_sales.index = monthly_sales.index.to_timestamp()
plt.figure(figsize=(12, 6))
plt.plot(monthly_sales.index, monthly_sales.values, marker='o')
plt.xlabel("Month Year")
```

```
plt.ylabel("Sales")
plt.title("Monthly Sales")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
<ipython-input-25-f7e387ea0451>:7: UserWarning: Could not infer
format, so each element will be parsed individually, falling back to
`dateutil`. To ensure parsing is consistent and as-expected, please
specify a format.
  df["trans_date"] = pd.to_datetime(df["trans_date"], errors='coerce')
```




```
recency=df.groupby("customer_id")["trans_date"].max()
frequency=df.groupby("customer_id")["trans_date"].count()
monetary=df.groupby("customer_id")["tran_amount"].sum()
rfm=pd.DataFrame({"recency":recency,"frequency":frequency,"monetary":m
onetary})
print(rfm)
```

```
               recency  frequency  monetary
customer_id
CS1112      2015-01-14         15      1012
CS1113      2015-02-09         20      1490
CS1114      2015-02-12         19      1432
CS1115      2015-03-05         22      1659
CS1116      2014-08-25         13       857
...                ...        ...       ...
CS8996      2014-12-09         13       582
CS8997      2014-06-28         14       543
CS8998      2014-12-22         13       624
```

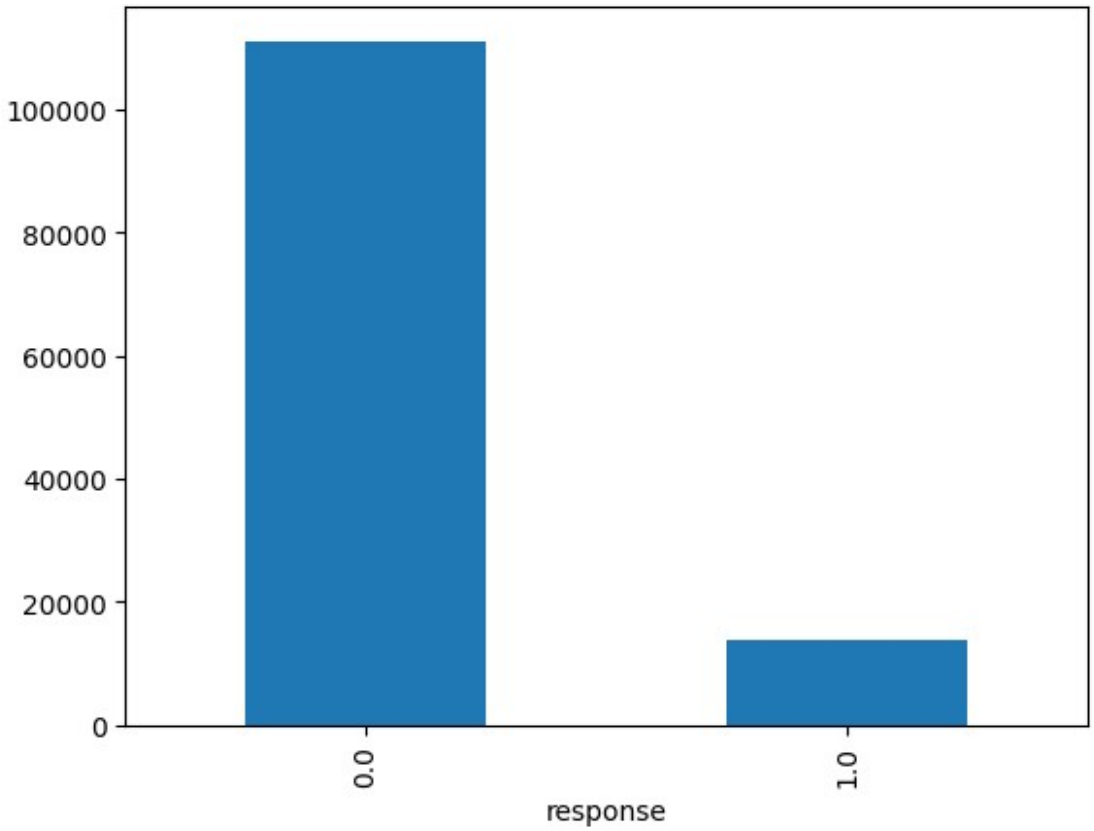
```
CS8999      2014-07-02          12       383
CS9000      2015-02-28          13       533

[6889 rows x 3 columns]
```

```
def segments_customer(row):
  if row["recency"].year>=2012 and row["frequency"]>=15 and
row["monetary"]>=1000:
    return "P0"
  elif (2011<=row["recency"].year<2012) and (10<row["frequency"]<15)
and (500<=row["monetary"]<=1000):
    return "p1"
  else:
    return "p2"
rfm["segments"]=rfm.apply(segments_customer,axis=1)
print(rfm)
```

```
                recency  frequency  monetary segments
customer_id
CS1112      2015-01-14          15      1012       P0
CS1113      2015-02-09          20      1490       P0
CS1114      2015-02-12          19      1432       P0
CS1115      2015-03-05          22      1659       P0
CS1116      2014-08-25          13       857       p2
...                ...         ...       ...      ...
CS8996      2014-12-09          13       582       p2
CS8997      2014-06-28          14       543       p2
CS8998      2014-12-22          13       624       p2
CS8999      2014-07-02          12       383       p2
CS9000      2015-02-28          13       533       p2

[6889 rows x 4 columns]
```

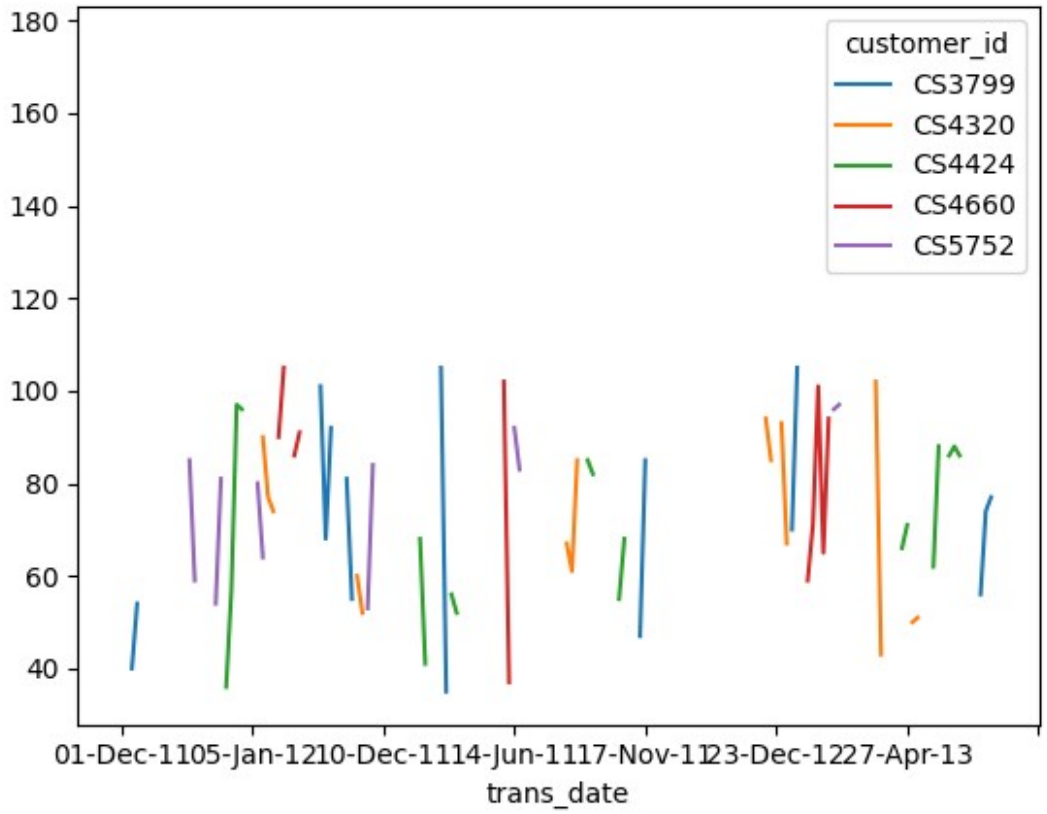
```
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import pandas as pd
response = pd.read_csv("/content/Retail_Data_Response.csv")
trnx = pd.read_csv("/content/Retail_Data_Transactions.csv")
df = trnx.merge(response, on="customer_id", how="left")
churn_count=df["response"].value_counts()
churn_count.plot(kind="bar")
```

```
<Axes: xlabel='response'>
```




```
top_5_customers = monetary.sort_values(ascending=False).head(5).index
top_customers_df = df[df["customer_id"].isin(top_5_customers)]
top_customers_sales = top_customers_df.groupby(["customer_id",
"trans_date"])["tran_amount"].sum().unstack(level=0)
top_customers_sales.plot(kind="line")
```

```
<Axes: xlabel='trans_date'>
```


customer_id
CS3799
CS4320
CS4424
CS4660
CS5752
180
160
140
120
100
80
60
40
01-Dec-11
05-Jan-12
10-Dec-11
14-Jun-11
17-Nov-11
23-Dec-12
27-Apr-13
trans_date