Customer Life time value Data Cleaning,EDA

In [ ]:
```
import pandas as pd
import plotly.express as px
```

In [ ]:
```
#df=pd.read_excel("./Dataset/online_retail_II.xlsx",sheet_name="Year 2010-2011",dtype_backend="pyarrow")
#df.head()
```

In [ ]:
```
#df.info()
```

In [ ]:
```
#df.to_csv("./Dataset/Online_retail_dec2010-dec2011.csv",index=False)
```

In [ ]:
```
df=pd.read_csv("./Dataset/Online_retail_dec2010-dec2011.csv",dtype_backend="pyarrow")
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541910 entries, 0 to 541909
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   Invoice      541910 non-null  string[pyarrow]
 1   StockCode    541910 non-null  string[pyarrow]
 2   Description  540456 non-null  string[pyarrow]
 3   Quantity     541910 non-null  int64[pyarrow]
 4   InvoiceDate  541910 non-null  string[pyarrow]
 5   Price        541910 non-null  double[pyarrow]
 6   Customer ID  406830 non-null  int64[pyarrow]
 7   Country      541910 non-null  string[pyarrow]
dtypes: double[pyarrow](1), int64[pyarrow](2), string[pyarrow](5)
memory usage: 59.1 MB
```

In [ ]:
```
df.head()
```

Out[ ]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850	United Kingdom

Data Cleaning

In [ ]:
```
df[df.isnull().any(axis=1)].head()
```

Out[ ]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
622	536414	22139	<NA>	56	2010-12-01 11:52:00	0.00	<NA>	United Kingdom
1443	536544	21773	DECORATIVE ROSE BATHROOM BOTTLE	1	2010-12-01 14:32:00	2.51	<NA>	United Kingdom
1444	536544	21774	DECORATIVE CATS BATHROOM BOTTLE	2	2010-12-01 14:32:00	2.51	<NA>	United Kingdom
1445	536544	21786	POLKADOT RAIN HAT	4	2010-12-01 14:32:00	0.85	<NA>	United Kingdom
1446	536544	21787	RAIN PONCHO RETROSPOT	2	2010-12-01 14:32:00	1.66	<NA>	United Kingdom

In [ ]:
```
df.dropna(inplace=True)
df.isna().sum()
```

Out[ ]:
```
Invoice        0
StockCode      0
Description    0
Quantity       0
InvoiceDate    0
Price          0
Customer ID    0
Country        0
dtype: int64
```

In [ ]:
```
df[df.isnull().any(axis=1)]
```

Out[ ]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country

In [ ]:
```
c=df["Invoice"].str.startswith('C')
c.unique()
```

Out[ ]:
```
<ArrowExtensionArray>
[False, True]
Length: 2, dtype: bool[pyarrow]
```

In [ ]:
```
df[c==True].head()
```

Out[ ]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
141	C536379	D	Discount	-1	2010-12-01 09:41:00	27.50	14527	United Kingdom
154	C536383	35004C	SET OF 3 COLOURED FLYING DUCKS	-1	2010-12-01 09:49:00	4.65	15311	United Kingdom
235	C536391	22556	PLASTERS IN TIN CIRCUS PARADE	-12	2010-12-01 10:24:00	1.65	17548	United Kingdom
236	C536391	21984	PACK OF 12 PINK PAISLEY TISSUES	-24	2010-12-01 10:24:00	0.29	17548	United Kingdom
237	C536391	21983	PACK OF 12 BLUE PAISLEY TISSUES	-24	2010-12-01 10:24:00	0.29	17548	United Kingdom

if Invoice startswith "C" it means that transction is a cancelled order

In [ ]:
```
cleandf=df[c!=True]
cleandf.head()
```

Out[ ]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850	United Kingdom

In [ ]:
```
del df
```

In [ ]:
```
cleandf[cleandf["Price"]==0].head()
```

Out[ ]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
9302	537197	22841	ROUND CAKE TIN VINTAGE GREEN	1	2010-12-05 14:02:00	0.0	12647	Germany
33576	539263	22580	ADVENT CALENDAR GINGHAM SACK	4	2010-12-16 14:36:00	0.0	16560	United Kingdom
40089	539722	22423	REGENCY CAKESTAND 3 TIER	10	2010-12-21 13:45:00	0.0	14911	EIRE
47068	540372	22090	PAPER BUNTING RETROSPOT	24	2011-01-06 16:41:00	0.0	13081	United Kingdom
47070	540372	22553	PLASTERS IN TIN SKULLS	24	2011-01-06 16:41:00	0.0	13081	United Kingdom

price = 0 order is invalid (no use)

In [ ]:
```
cleandf=cleandf[cleandf["Price"]>0]
cleandf.head()
```

Out[ ]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850	United Kingdom

In [ ]:
```
cleandf["Revenue"]=cleandf["Quantity"]*cleandf["Price"]
cleandf.head()
```

Out[ ]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country	Revenue
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850	United Kingdom	15.30
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850	United Kingdom	20.34
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850	United Kingdom	22.00
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850	United Kingdom	20.34
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850	United Kingdom	20.34

In [ ]:
```
cleandf.convert_dtypes(dtype_backend="pyarrow")
cleandf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 397885 entries, 0 to 541909
Data columns (total 9 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   Invoice      397885 non-null  string[pyarrow]
 1   StockCode    397885 non-null  string[pyarrow]
 2   Description  397885 non-null  string[pyarrow]
 3   Quantity     397885 non-null  int64[pyarrow]
 4   InvoiceDate  397885 non-null  string[pyarrow]
 5   Price        397885 non-null  double[pyarrow]
 6   Customer ID  397885 non-null  int64[pyarrow]
 7   Country      397885 non-null  string[pyarrow]
 8   Revenue      397885 non-null  double[pyarrow]
dtypes: double[pyarrow](2), int64[pyarrow](2), string[pyarrow](5)
memory usage: 49.7 MB
```

In [ ]:
```
#cleandf.to_csv("./Dataset/Cleaned_Online_retail_dec2010-dec2011.csv",index=False)
```

In [ ]:
```
df2=cleandf[["Invoice","InvoiceDate","Customer ID","Country","Revenue"]]
df2.head()
```

Out[ ]:

	Invoice	InvoiceDate	Customer ID	Country	Revenue
0	536365	2010-12-01 08:26:00	17850	United Kingdom	15.30
1	536365	2010-12-01 08:26:00	17850	United Kingdom	20.34
2	536365	2010-12-01 08:26:00	17850	United Kingdom	22.00
3	536365	2010-12-01 08:26:00	17850	United Kingdom	20.34
4	536365	2010-12-01 08:26:00	17850	United Kingdom	20.34

EDA

In [ ]:
```
len(df2["Customer ID"].unique())
```

Out[ ]:
```
4338
```

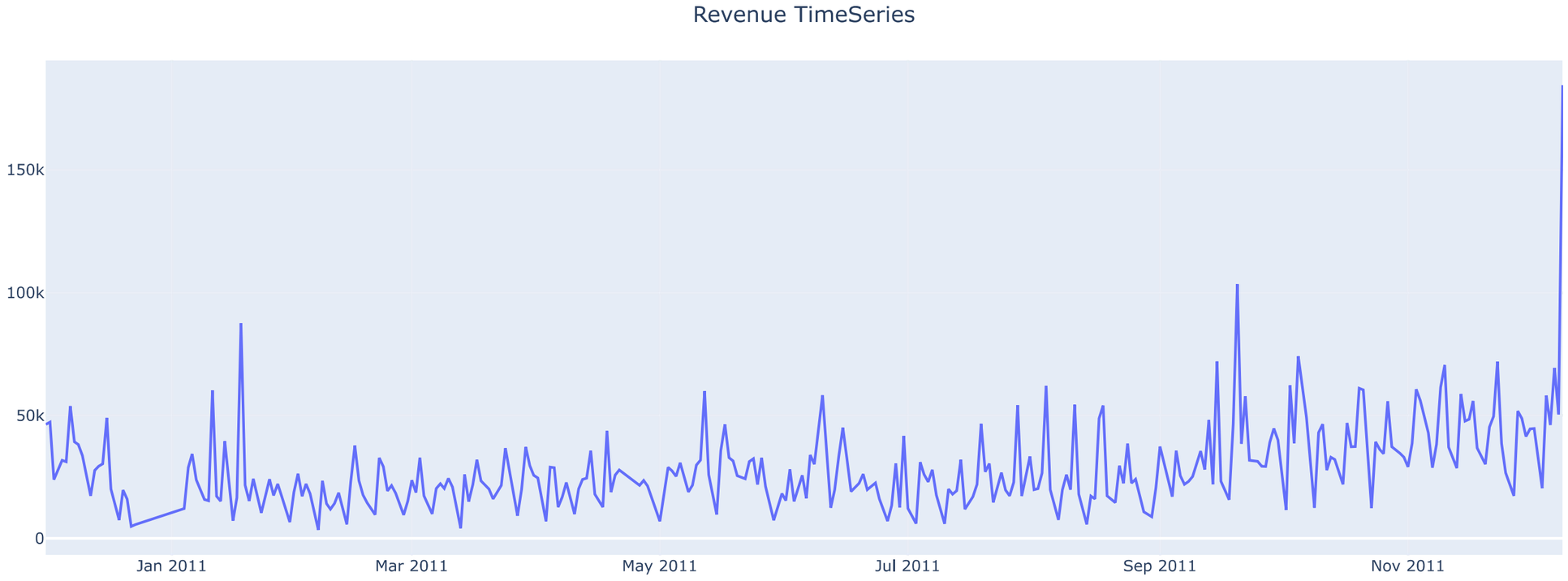
In [ ]:
```
df2['InvoiceDate'] = pd.to_datetime(df2['InvoiceDate']).dt.date
df2.head()
```

```
C:\Users\dvplo\AppData\Local\Temp\ipykernel_14688\1535909738.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[ ]:

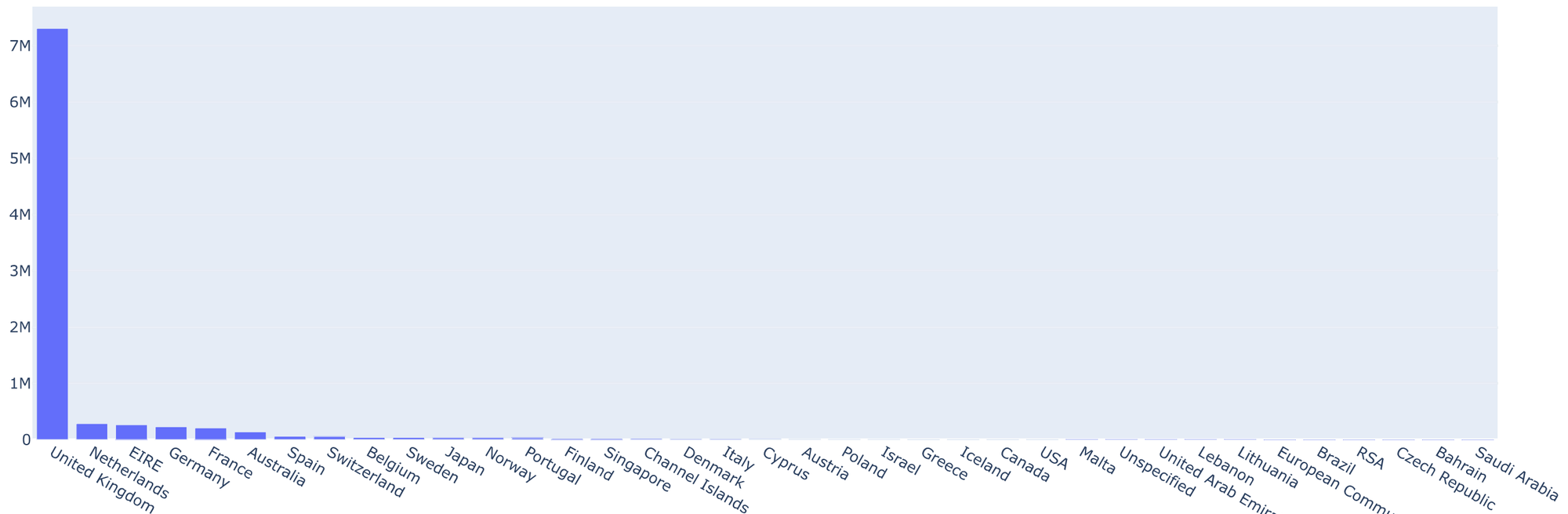
	Invoice	InvoiceDate	Customer ID	Country	Revenue
0	536365	2010-12-01	17850	United Kingdom	15.30
1	536365	2010-12-01	17850	United Kingdom	20.34
2	536365	2010-12-01	17850	United Kingdom	22.00
3	536365	2010-12-01	17850	United Kingdom	20.34
4	536365	2010-12-01	17850	United Kingdom	20.34

In [ ]:
```
timeseries_grp=df2.groupby('InvoiceDate')['Revenue'].sum()
fig=px.line(timeseries_grp,x=timeseries_grp.index ,y="Revenue")
fig.update_layout(xaxis_title="",yaxis_title="",title={
            'text' : 'Revenue TimeSeries',
            'x':0.5,
            'xanchor': 'center'
        })
fig.show("notebook")
```



In [ ]:

```
dfc=df2.groupby('Country')['Revenue'].sum().sort_values(ascending=False)
fig=px.bar(dfc,x=dfc.index,y="Revenue")
fig.update_layout(xaxis_title="",yaxis_title="",title={
            'text' : 'Revenue by Country',
            'x':0.5,
            'xanchor': 'center'
        })
fig.show("notebook")
```



In [ ]:
```
df2.groupby(['Customer ID','Invoice']).agg({'Revenue':['sum'],'InvoiceDate':['max']}).head()
```

Out[ ]:

		Revenue	InvoiceDate
		sum	max
Customer ID	Invoice		
12346	**541431**	77183.60	2011-01-18
12347	**537626**	711.79	2010-12-07
	542237	475.39	2011-01-26
	549222	636.25	2011-04-07
	556201	382.52	2011-06-09

In [ ]:
```
max_date=df2['InvoiceDate'].max()
max_date
```

Out[ ]:
```
datetime.date(2011, 12, 9)
```

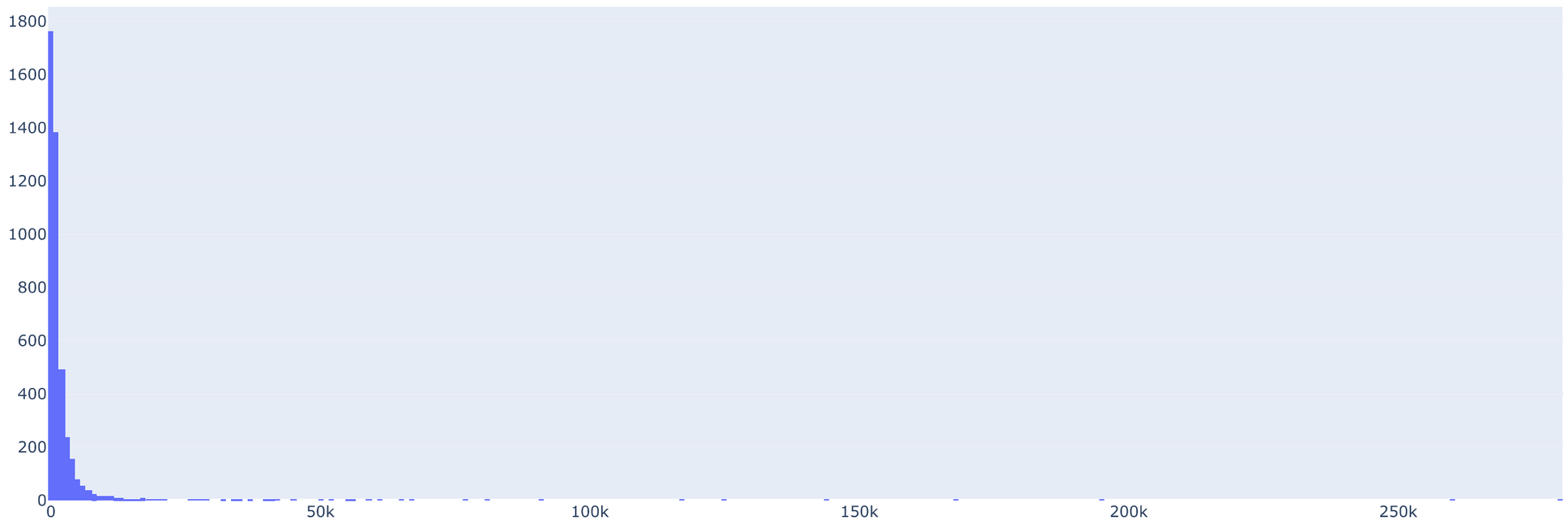
In [ ]:
```
customer_RFM=df2.groupby('Customer ID').agg(Lifespan_in_days=('InvoiceDate',lambda x:(x.max()-x.min()).days),
                                            Recency_in_days=('InvoiceDate',lambda x:(max_date-x.max()).days),
                                            Frequency=('Invoice','nunique'),
                                            Total_sales=('Revenue','sum'),
                                            Avg_sales=('Revenue','mean')
                                            )
customer_RFM.head()
```

Out[ ]:

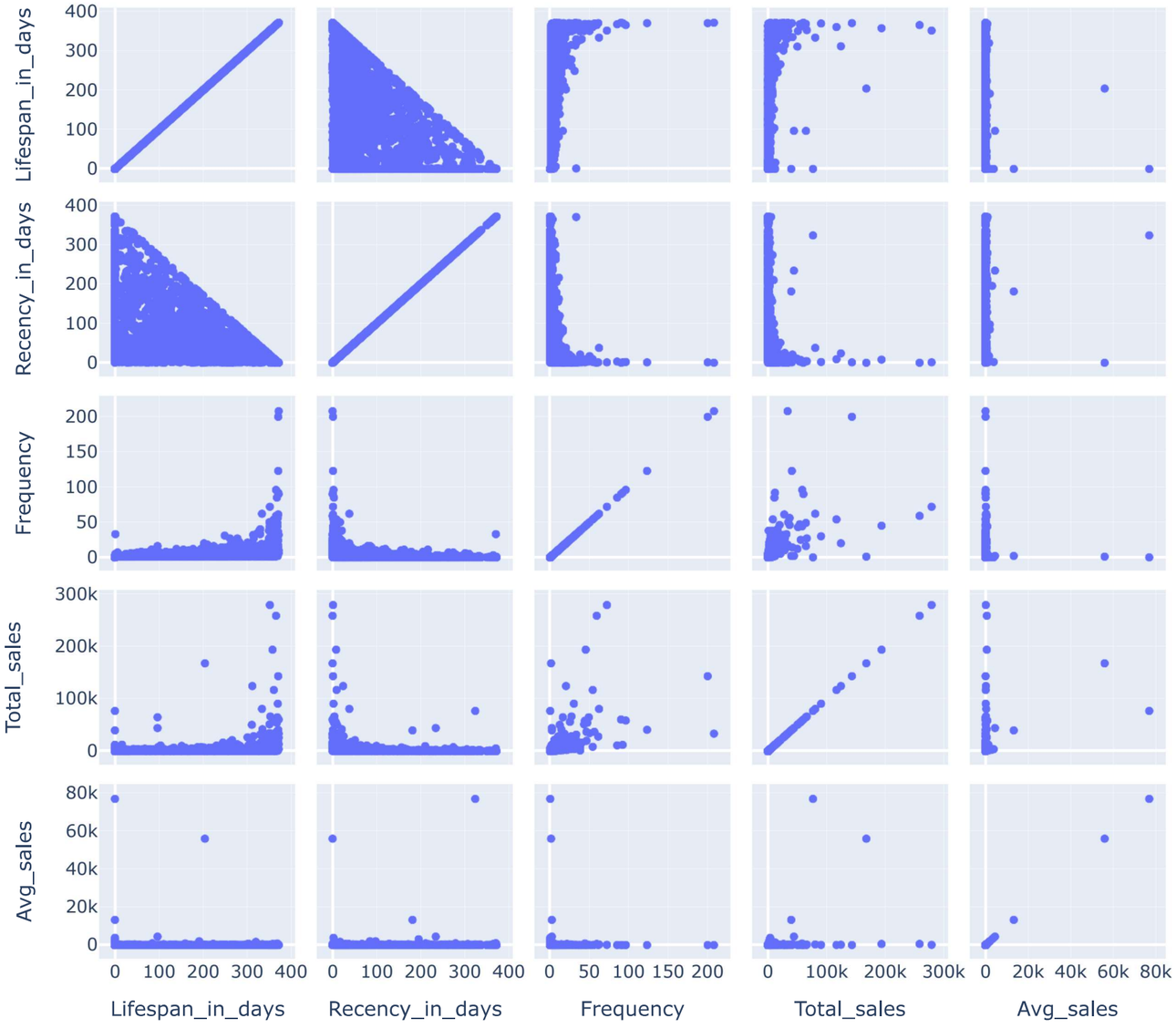
Customer ID	Lifespan_in_days	Recency_in_days	Frequency	Total_sales	Avg_sales
12346	0	325	1	77183.60	77183.600000
12347	365	2	7	4310.00	23.681319
12348	283	75	4	1797.24	57.975484
12349	0	18	1	1757.55	24.076027
12350	0	310	1	334.40	19.670588

In [ ]:
```
fig=px.histogram(customer_RFM,x="Frequency")
fig.update_layout(xaxis_title="",yaxis_title="",title={
            'text' : 'Histogram of Customers Frequency',
            'x':0.5,
            'xanchor': 'center'
        })
fig.show("notebook")
```

In [ ]:
```
fig=px.histogram(customer_RFM,x="Total_sales")
fig.update_layout(xaxis_title="",yaxis_title="",title={
            'text' : 'Histogram of Total Sales',
            'x':0.5,
            'xanchor': 'center'
        })
fig.show("notebook")
```



In [ ]:
```
fig=px.scatter_matrix(customer_RFM)
fig.update_layout(height=800, width=900)
fig.show("notebook")
```



In [ ]:
```
customer_RFM.corr()
```

Out[ ]:

	Lifespan_in_days	Recency_in_days	Frequency	Total_sales	Avg_sales
Lifespan_in_days	1.000000	-0.513980	0.476524	0.225659	-0.010097
Recency_in_days	-0.513980	1.000000	-0.260783	-0.122241	0.024510
Frequency	0.476524	-0.260783	1.000000	0.553650	-0.005205
Total_sales	0.225659	-0.122241	0.553650	1.000000	0.287264
Avg_sales	-0.010097	0.024510	-0.005205	0.287264	1.000000