Data Cleaning and EDA using Python:

Data Cleaning:

Importing Medical inventory dataset and viewing the data

import pandas as pd

df = pd.read_csv("Medical Inventory Optimaization Dataset.csv")

df

Typeofsales	Patient_ID	Specialisation	Dept	Dateofbill	Quantity	ReturnQuantity	Final_Cost	Final_Sales	RtnMRP	Formulation	DrugName
Sale	12018098765	Specialisation6	Department1	06-01-2022	1	0	55.406	59.260	0.0	Form1	ZINC ACETATE 20MG/5ML SYP
Sale	12018103897	Specialisation7	Department1	7/23/2022	1	0	768.638	950.800	0.0	Form1	CEFTAZIDIME 2GM+AVIBACTAM 500MG
Sale	12018101123	Specialisation2	Department3	6/23/2022	1	0	774.266	4004.214	0.0	Form2	EPTIFIBATIDE 0.75MG/ML
Sale	12018079281	Specialisation40	Department1	3/17/2022	2	0	40.798	81.044	0.0	Form1	WATER FOR INJECTION 10ML SOLUTION
Sale	12018117928	Specialisation5	Department1	12/21/2022	1	0	40.434	40.504	0.0	Form1	LORAZEPAM 1MG
...	...	...	...	...	...	...	...	...	...	...	...
Sale	12018099994	Specialisation39	Department1	6/19/2022	3	0	61.436	145.200	0.0	Form1	SODIUM CHLORIDE IVF 100ML
Sale	12018047025	Specialisation4	Department1	2/24/2022	2	0	64.448	119.692	0.0	Form1	PIPERACILLIN 1GM + TAZOBACTAM 125MG
Sale	12018017139	Specialisation1	Department1	6/27/2022	4	0	74.944	642.040	0.0	Form1	PARACETAMOL 1GM IV INJ
Sale	12018044140	Specialisation20	Department1	7/30/2022	1	0	111.680	181.000	0.0	Form3	MEROPENEM 1GM INJ
Sale	12018116820	Specialisation26	Department1	10/24/2022	3	0	46.182	133.800	0.0	Form1	TRAMADOL

Displaying only 5 rows from top

df.head()

	Typeofsales	Patient_ID	Specialisation	Dept	Dateofbill	Quantity	ReturnQuantity	Final_Cost	Final_Sales	RtnMRP	Formulation	DrugNam
0	Sale	12018098765	Specialisation6	Department1	06-01-2022	1	0	55.406	59.260	0.0	Form1	ZINC ACETATI 20MG/5ML SYI
1	Sale	12018103897	Specialisation7	Department1	7/23/2022	1	0	768.638	950.800	0.0	Form1	CEFTAZIDIMI 2GM+AVIBACTAN 500MC
2	Sale	12018101123	Specialisation2	Department3	6/23/2022	1	0	774.266	4004.214	0.0	Form2	EPTIFIBATIDI 0.75MG/MI
3	Sale	12018079281	Specialisation40	Department1	3/17/2022	2	0	40.798	81.044	0.0	Form1	WATER FOI INJECTION 10MI SOLUTION
4	Sale	12018117928	Specialisation5	Department1	12/21/2022	1	0	40.434	40.504	0.0	Form1	LORAZEPAM 1MC

Displaying last 5 rows of dataset

df.tail()

Out[3]:

	Typeofsales	Patient_ID	Specialisation	Dept	Dateofbill	Quantity	ReturnQuantity	Final_Cost	Final_Sales	RtnMRP	Formulation	DrugNa
14213	Sale	12018099994	Specialisation39	Department1	6/19/2022	3	0	61.436	145.200	0.0	Form1	SODI CHLORIDE 100
14214	Sale	12018047025	Specialisation4	Department1	2/24/2022	2	0	64.448	119.692	0.0	Form1	PIPERACIL 1G TAZOBACT 125
14215	Sale	12018017139	Specialisation1	Department1	6/27/2022	4	0	74.944	642.040	0.0	Form1	PARACETAM 1GM IV
14216	Sale	12018044140	Specialisation20	Department1	7/30/2022	1	0	111.680	181.000	0.0	Form3	MEROPEN 1GM
14217	Sale	12018116820	Specialisation26	Department1	10/24/2022	3	0	46.182	133.800	0.0	Form1	TRAMAD

Checking for the null values

checking null values

df.isnull()

Out[4]:

Typeofsales	Patient_ID	Specialisation	Dept	Dateofbill	Quantity	ReturnQuantity	Final_Cost	Final_Sales	RtnMRP	Formulation	DrugName	SubCat	SubCat1
False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False

#returns boolean value if null values is present in data

df.isna().any()

Out[5]:

```
Typeofsales       False
Patient_ID        False
Specialisation    False
Dept              False
Dateofbill        False
Quantity          False
ReturnQuantity    False
Final_Cost        False
Final_Sales       False
RtnMRP            False
Formulation        True
DrugName           True
SubCat             True
SubCat1            True
dtype: bool
```

#identifying total null values

df.isna().sum()

```
Out[6]: Typeofsales          0
        Patient_ID           0
        Specialisation       0
        Dept                 0
        Dateofbill           0
        Quantity             0
        ReturnQuantity       0
        Final_Cost           0
        Final_Sales          0
        RtnMRP               0
        Formulation        653
        DrugName          1668
        SubCat            1668
        SubCat1           1692
        dtype: int64
```

number of columns contining null values

df.isna().any().sum()

```
Out[7]: 4
```

From the above code we can see we have null values in 4 different column namely Formulation, Drugname, Subcat, Subcat1. Here Formulation contains 653 values, Drugname contains 1668, Subcat contains 1688, Subcat1 contains 1692 null values.

Replacing null values with NA. So the sum of columns will remain same

df.fillna('NA')

Out[13]:

Typeofsales	Patient_ID	Specialisation	Dept	Dateofbill	Quantity	ReturnQuantity	Final_Cost	Final_Sales	RtnMRP	Formulation	DrugName	
Sale	12018098765	Specialisation6	Department1	06-01-2022	1	0	55.406	59.260	0.0	Form1	ZINC ACETATE 20MG/5ML SYP	
Sale	12018103897	Specialisation7	Department1	7/23/2022	1	0	768.638	950.800	0.0	Form1	CEFTAZIDIME 2GM+AVIBACTAM 500MG	
Sale	12018101123	Specialisation2	Department3	6/23/2022	1	0	774.266	4004.214	0.0	Form2	EPTIFIBATIDE 0.75MG/ML	
Sale	12018079281	Specialisation40	Department1	3/17/2022	2	0	40.798	81.044	0.0	Form1	WATER FOR INJECTION 10ML SOLUTION	
Sale	12018117928	Specialisation5	Department1	12/21/2022	1	0	40.434	40.504	0.0	Form1	LORAZEPAM 1MG	
...	...	...	...	...	...	...	...	...	...	...	...	
Sale	12018099994	Specialisation39	Department1	6/19/2022	3	0	61.436	145.200	0.0	Form1	SODIUM CHLORIDE IVF 100ML	EL
Sale	12018047025	Specialisation4	Department1	2/24/2022	2	0	64.448	119.692	0.0	Form1	PIPERACILLIN 1GM + TAZOBACTAM 125MG	
Sale	12018017139	Specialisation1	Department1	6/27/2022	4	0	74.944	642.040	0.0	Form1	PARACETAMOL 1GM IV INJ	
Sale	12018044140	Specialisation20	Department1	7/30/2022	1	0	111.680	181.000	0.0	Form3	MEROPENEM	

Formating Date values:

From the observation in date column, the format is in consistant. 08-06-2022 and 7/15/2022 are the two formats we can see in the date column. So we are changing the format uniformly 7/15/2022 and convert the format from m/d/y to d/m/y

```
import pandas as pd

from dateutil import parser


# Function to convert different date formats to dd/mm/yyyy

def convert_to_dd_mm_yyyy(date_str):

  try:

    # Try parsing as mm/dd/yy format

    date_parsed = parser.parse(date_str, dayfirst=False, yearfirst=False)

  except ValueError:

    # If parsing fails, try parsing as dd-mm-yyyy format

    date_parsed = parser.parse(date_str, dayfirst=True, yearfirst=False)


  # Convert the date to the desired format 'dd/mm/yyyy'

  return date_parsed.strftime('%d/%m/%Y')
```

Apply the conversion function to the 'dateofbill' column

df['Dateofbill'] = df['Dateofbill'].apply(convert_to_dd_mm_yyyy)

Display the updated DataFrame

print(df)

```
      Typeofsales   Patient_ID    Specialisation         Dept  Dateofbill  \
0            Sale  12018098765   Specialisation6  Department1  01/06/2022
1            Sale  12018103897   Specialisation7  Department1  23/07/2022
2            Sale  12018101123   Specialisation2  Department3  23/06/2022
3            Sale  12018079281  Specialisation40  Department1  17/03/2022
4            Sale  12018117928   Specialisation5  Department1  21/12/2022
...           ...          ...               ...          ...         ...
14213        Sale  12018099994  Specialisation39  Department1  19/06/2022
14214        Sale  12018047025   Specialisation4  Department1  24/02/2022
14215        Sale  12018017139   Specialisation1  Department1  27/06/2022
14216        Sale  12018044140  Specialisation20  Department1  30/07/2022
14217        Sale  12018116820  Specialisation26  Department1  24/10/2022

       Quantity  ReturnQuantity  Final_Cost  Final_Sales  RtnMRP Formulation  \
0             1               0      55.406       59.260     0.0       Form1
1             1               0     768.638      950.800     0.0       Form1
2             1               0     774.266     4004.214     0.0       Form2
3             2               0      40.798       81.044     0.0       Form1
4             1               0      40.434       40.504     0.0       Form1
...         ...             ...         ...          ...     ...         ...
14213         3               0      61.436      145.200     0.0       Form1
14214         2               0      64.448      119.692     0.0       Form1
14215         4               0      74.944      642.040     0.0       Form1
14216         1               0     111.680      181.000     0.0       Form3
14217         3               0      46.182      133.800     0.0       Form1

                                DrugName                SubCat  \
0                ZINC ACETATE 20MG/5ML SYP   SYRUP & SUSPENSION
1           CEFTAZIDIME 2GM+AVIBACTAM 500MG          INJECTIONS
2                  EPTIFIBATIDE 0.75MG/ML          INJECTIONS
3         WATER FOR INJECTION 10ML SOLUTION          INJECTIONS
4                             LORAZEPAM 1MG   TABLETS & CAPSULES
...                                     ...                 ...
```

Feature engineering:

Create column with month name from date and add that column and remaning cleaned data from MioView to other view name MioView1

Feature Engineering

df['month'] = pd.to_datetime(df['Dateofbill'], format='%d/%m/%Y').dt.strftime('%b')

df

EDA Analysis:

Summary statistics for all numerical columns

numerical_summary = df[['Quantity', 'ReturnQuantity', 'Final_Cost', 'Final_Sales', 'RtnMRP']].describe()

print(numerical_summary)

	Quantity	ReturnQuantity	Final_Cost	Final_Sales	RtnMRP
count	14218.000000	14218.000000	14218.000000	14218.000000	14218.000000
mean	2.231748	0.291954	124.823957	234.038300	29.126755
std	5.132043	1.643322	464.782794	671.261572	182.262335
min	0.000000	0.000000	40.000000	0.000000	0.000000
25%	1.000000	0.000000	44.928000	47.815000	0.000000
50%	1.000000	0.000000	53.650000	86.424000	0.000000
75%	2.000000	0.000000	77.800000	181.000000	0.000000
max	150.000000	50.000000	33178.000000	39490.000000	8014.000000

df['Typeofsales'].value_counts()

```
Out[31]: Sale       12537
         Return      1681
         Name: Typeofsales, dtype: int64
```

df['Specialisation'].value_counts()

```
Out[32]: Specialisation4      3999
         Specialisation7      2098
         Specialisation3       734
         Specialisation2       609
         Specialisation8       594
         Specialisation20      554
         Specialisation11      516
         Specialisation16      509
         Specialisation1       440
         Specialisation14      436
         Specialisation5       390
         Specialisation21      360
         Specialisation26      342
         Specialisation6       251
         Specialisation23      249
         Specialisation25      201
         Specialisation31      184
         Specialisation17      178
         Specialisation9       158
         Specialisation15      143
         Specialisation42      132
         Specialisation27      117
         Specialisation10      107
         Specialisation50      100
         Specialisation33       99
         Specialisation55       91
         Specialisation43       74
         Specialisation45       55
         Specialisation34       53
         Specialisation39       47
         Specialisation41       44
         Specialisation40       43
         Specialisation28       34
         Specialisation19       30
         Specialisation48       24
         Specialisation61       23
         Specialisation12       23
         Specialisation49       23
         Specialisation54       20
```

df['Dept'].value_counts()

```
Out[33]: Department1    12440
         Department2     1566
         Department3      212
         Name: Dept, dtype: int64
```

df['DrugName'].value_counts()

```
Out[34]: SODIUM CHLORIDE IVF 100ML                                                604
         SODIUM CHLORIDE 0.9%                                                     526
         MULTIPLE ELECTROLYTES 500ML IVF                                          467
         ONDANSETRON 2MG/ML                                                       444
         PANTOPRAZOLE 40MG INJ                                                    441
                                                                                 ...
         BASILIXIMAB 20 MG                                                          1
         MULTIVITAMIN + MULTIMINERAL + ANTIOXIDANTS + METHYLCOBALAMIN 200ML SYRUP   1
         ROPINIROLE 0.5MG TAB                                                       1
         IRON SUCROSE 100MG INJ                                                     1
         FENTANYL 12.5MCG/HR                                                        1
         Name: DrugName, Length: 751, dtype: int64
```

df['Formulation'].value_counts()

```
Out[35]: Form1     11622
         Form2      1325
         Patent      539
         Form3        79
         Name: Formulation, dtype: int64
```

df['SubCat'].value_counts()

```
Out[36]: INJECTIONS                        6500
         IV FLUIDS, ELECTROLYTES, TPN      2709
         TABLETS & CAPSULES                1505
         INHALERS & RESPULES                469
         OINTMENTS, CREAMS & GELS           364
         LIQUIDS & SOLUTIONS                265
         SYRUP & SUSPENSION                 237
         POWDER                             216
         NUTRITIONAL SUPPLEMENTS            126
         PESSARIES & SUPPOSITORIES           55
         DROPS                               53
         VACCINE                             19
         SPRAY                               12
         PATCH                               11
         LOTIONS                              5
         SOLUTION                             3
         SACHETS                              1
         Name: SubCat, dtype: int64
```

df['SubCat1'].value_counts()

```
Out[38]: INTRAVENOUS & OTHER STERILE SOLUTIONS                                  3192
         GASTROINTESTINAL & HEPATOBILIARY SYSTEM                                1738
         ANTI-INFECTIVES                                                        1647
         CARDIOVASCULAR & HEMATOPOIETIC SYSTEM                                  1480
         CENTRAL NERVOUS SYSTEM                                                 1262
         RESPIRATORY SYSTEM                                                      593
         ANAESTHETICS                                                            591
         NUTRITION                                                               331
         MUSCULO-SKELETAL SYSTEM                                                 322
         VITAMINS & MINERALS                                                     303
         HORMONES                                                                291
         IMMUNOLOGY                                                              186
         ENDOCRINE & METABOLIC SYSTEM                                            161
         DERMATOLOGY                                                             156
         EAR & MOUTH/ THROAT                                                      74
         ONCOLOGY                                                                 64
         GENITO-URINARY SYSTEM                                                    44
         CARDIIVASCULAR&HEMATOPOIETIC SYSTEM                                      38
         OPHTHALMOLOGY                                                            34
         ANTIDOTES, DETOXIFYING AGENTS & DRUGS USED IN SUBSTANCE DEPENDENCE       11
         MISCELLANEOUS                                                             8
         Name: SubCat1, dtype: int64
```

df['month'].value_counts()

```
Out[39]: Dec    1417
         Aug    1375
         Jul    1294
         Apr    1252
         May    1196
         Sep    1190
         Mar    1180
         Nov    1150
         Oct    1118
         Jun    1042
         Jan    1017
         Feb     987
         Name: month, dtype: int64
```

First moment bussiness decision: Mean,Median,Mode

```
# Calculate mean, median, and mode for numerical columns
numerical_columns = ['Quantity', 'ReturnQuantity', 'Final_Cost', 'Final_Sales', 'RtnMRP']

# Mean
mean_values = df[numerical_columns].mean()
print("Mean Values:")
print(mean_values)

# Median
median_values = df[numerical_columns].median()
print("\nMedian Values:")
print(median_values)

# Mode
mode_values = df[numerical_columns].mode().iloc[0]
```

print("\nMode Values:")

print(mode_values)

```
Mean Values:
Quantity              2.231748
ReturnQuantity        0.291954
Final_Cost          124.823957
Final_Sales         234.038300
RtnMRP               29.126755
dtype: float64

Median Values:
Quantity             1.000
ReturnQuantity       0.000
Final_Cost          53.650
Final_Sales         86.424
RtnMRP               0.000
dtype: float64

Mode Values:
Quantity             1.000
ReturnQuantity       0.000
Final_Cost          49.352
Final_Sales          0.000
RtnMRP               0.000
Name: 0, dtype: float64
```

Second Moment Business Decision: Variance, Standard Deviation, Range

import pandas as pd

Compute variance for numerical columns

variance_df = df[['Quantity', 'ReturnQuantity', 'Final_Cost', 'Final_Sales', 'RtnMRP']].var()

```
# Compute standard deviation for numerical columns
std_dev_df = df[['Quantity', 'ReturnQuantity', 'Final_Cost', 'Final_Sales', 'RtnMRP']].std()

# Compute range for numerical columns
range_df = df[['Quantity', 'ReturnQuantity', 'Final_Cost', 'Final_Sales', 'RtnMRP']].max() - df[['Quantity', 'ReturnQuantity', 'Final_Cost', 'Final_Sales', 'RtnMRP']].min()

# Display the results
print("Variance:")
print(variance_df)

print("\nStandard Deviation:")
print(std_dev_df)

print("\nRange:")
print(range_df)
```

```
Variance:
Quantity                26.337862
ReturnQuantity           2.700506
Final_Cost          216023.045394
Final_Sales         450592.097666
RtnMRP               33219.558938
dtype: float64

Standard Deviation:
Quantity              5.132043
ReturnQuantity        1.643322
Final_Cost          464.782794
Final_Sales         671.261572
RtnMRP              182.262335
dtype: float64

Range:
Quantity            150.0
ReturnQuantity       50.0
Final_Cost        33138.0
Final_Sales       39490.0
RtnMRP             8014.0
dtype: float64
```

Third Moment Business Decision: skewness, Kurtosis

```
from scipy.stats import skew, kurtosis

numerical_columns = ['Quantity', 'ReturnQuantity', 'Final_Cost', 'Final_Sales', 'RtnMRP']

for column in numerical_columns:

    col_data = df[column].dropna()  # Remove NaN values

    skew_val = skew(col_data)

    kurt_val = kurtosis(col_data)

    print(f"\n{column} Skewness: {skew_val:.4f}")

    print(f"{column} Kurtosis: {kurt_val:.4f}")
```

```
Quantity Skewness: 11.3398
Quantity Kurtosis: 180.0901

ReturnQuantity Skewness: 17.1706
ReturnQuantity Kurtosis: 409.2725

Final_Cost Skewness: 34.5046
Final_Cost Kurtosis: 2025.1537

Final_Sales Skewness: 21.0045
Final_Sales Kurtosis: 948.1888

RtnMRP Skewness: 15.7962
RtnMRP Kurtosis: 403.3826
```

Analysis:

```
import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt


# Q1) Total Distinct Patients

total_distinct_patients = df['Patient_ID'].nunique()

print(f"Q1) Total Distinct Patients: {total_distinct_patients}")


# Q2) Patient_ID count where type of sale is return

return_patients_count = df[df['Typeofsales'] == 'Return']['Patient_ID'].nunique()

print(f"Q2) Patient_ID count where type of sale is return: {return_patients_count}")


# Q3) Patient_ID count where type of sale is sales

sales_patients_count = df[df['Typeofsales'] == 'Sale']['Patient_ID'].nunique()

print(f"Q3) Patient_ID count where type of sale is sales: {sales_patients_count}")
```

```
# Q4) Overall Bounce Rate

overall_bounce_rate = (return_patients_count / total_distinct_patients) * 100

print(f"Q4) Overall Bounce Rate: {overall_bounce_rate:.2f}%")


# Q5) Bounce rate by Specialization

bounce_rate_by_specialization = df[df['Typeofsales'] ==
'Return'].groupby('Specialisation')['Patient_ID'].nunique() /
df.groupby('Specialisation')['Patient_ID'].nunique() * 100

print(f"Q5) Bounce Rate by Specialization:")

print(bounce_rate_by_specialization)


# Q6) Total cost of purchase that returns from SubCat

total_cost_return_subcat = df[df['Typeofsales'] == 'Return']['Final_Cost'].sum()

print(f"Q6) Total cost of purchase that returns from SubCat: {total_cost_return_subcat:.2f}")


# Q7) Count of drugs returned without sales

drugs_returned_without_sales_count = df[(df['Typeofsales'] == 'Return') & (df['Final_Sales'] ==
0)]['DrugName'].nunique()

print(f"Q7) Count of drugs returned without sales: {drugs_returned_without_sales_count}")


# Q8) Return items based on month

return_items_by_month = df[df['Typeofsales'] == 'Return'].groupby('month')['Patient_ID'].count()

print(f"Q8) Return items based on month:")

print(return_items_by_month)


# Q9) Total sales when sales is return or sale

total_sales_return_sale = df[df['Typeofsales'].isin(['Return', 'Sale'])]['Final_Sales'].sum()

print(f"Q9) Total sales when sales is return or sale: {total_sales_return_sale:.2f}")
```

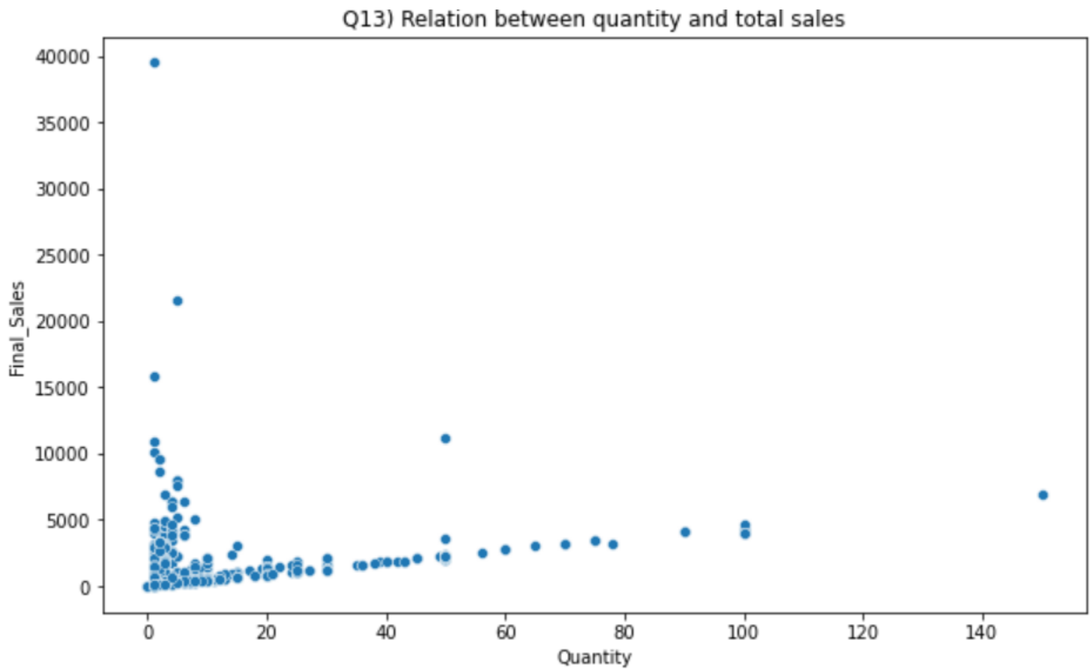
```python
# Q10) Drugs which are mostly returned

mostly_returned_drugs = df[df['Typeofsales'] == 'Return']['DrugName'].value_counts().idxmax()

print(f"Q10) Drugs which are mostly returned: {mostly_returned_drugs}")


# Q11) Total sales based on month

total_sales_by_month = df.groupby('month')['Final_Sales'].sum()

print(f"Q11) Total sales based on month:")

print(total_sales_by_month)


# Q12) Average quantity of drug purchases

average_quantity_purchases = df.groupby('DrugName')['Quantity'].mean()

print(f"Q12) Average quantity of drug purchases:")

print(average_quantity_purchases)


# Q13) Relation between quantity and total sales

plt.figure(figsize=(10, 6))

sns.scatterplot(x='Quantity', y='Final_Sales', data=df)

plt.title('Q13) Relation between quantity and total sales')

plt.show()


# Q14) Average Sales based on Specialisation

average_sales_by_specialization = df.groupby('Specialisation')['Final_Sales'].mean()

print(f"Q14) Average Sales based on Specialisation:")

print(average_sales_by_specialization)


# Q15) Frequency of return quantity

return_quantity_frequency = df[df['Typeofsales'] == 'return']['ReturnQuantity'].value_counts()

print(f"Q15) Frequency of return quantity:")

print(return_quantity_frequency)
```

```
Q1) Total Distinct Patients: 4883
Q2) Patient_ID count where type of sale is return: 1217
Q3) Patient_ID count where type of sale is sales: 4632
Q4) Overall Bounce Rate: 24.92%
Q5) Bounce Rate by Specialization:
Specialisation
Specialisation1     21.844660
Specialisation10    16.417910
Specialisation11    13.939394
Specialisation12    14.285714
Specialisation13    50.000000
Specialisation14     9.465021
Specialisation15    30.188679
Specialisation16    26.282051
Specialisation17    26.373626
Specialisation18    50.000000
Specialisation19    45.454545
Specialisation2     15.580737
Specialisation20    15.471698
```

```
Q6) Total cost of purchase that returns from SubCat: 191156.52
Q7) Count of drugs returned without sales: 249
Q8) Return items based on month:
month
Apr    129
Aug    177
Dec    151
Feb    123
Jan    110
Jul    156
Jun    116
Mar    143
May    178
Nov    133
Oct    130
Sep    135
Name: Patient_ID, dtype: int64
```

```
Q9) Total sales when sales is return or sale: 3327556.56
Q10) Drugs which are mostly returned: SODIUM CHLORIDE IVF 100ML
Q11) Total sales based on month:
month
Apr    267918.874
Aug    319996.630
Dec    412259.126
Feb    247230.908
Jan    236331.070
Jul    309785.630
Jun    216637.552
Mar    250913.852
May    300612.102
Nov    273303.022
Oct    258533.458
Sep    234034.332
Name: Final_Sales, dtype: float64
```

```
Q12) Average quantity of drug purchases:
DrugName
ACEBROPHYLLINE 100MG CAP                                            1.000000
ACEBROPHYLLINE 200MG TAB                                            0.857143
ACECLOFENAC 100MG + PARACETAMOL 325MG + SERRATIOPEPTIDASE 15MG TAB    1.000000
ACECLOFENAC 100MG + PARACETAMOL 325MG TAB                           1.000000
ACYCLOVIR 200MG TAB                                                 0.750000
                                                                      ...
ZINC ACETATE 20MG/5ML SYP                                           1.000000
ZINC OXIDE  30GM CREAM                                              1.666667
ZINC OXIDE + CALENDULA + ALOE + JOJOBA OIL 100ML CREAM              0.666667
ZOLPIDEM 10MG TAB                                                   1.000000
ZOLPIDEM 5MG TAB                                                    1.000000
Name: Quantity, Length: 751, dtype: float64
```


Q13) Relation between quantity and total sales
Final_Sales
Quantity
0
5000
10000
15000
20000
25000
30000
35000
40000
0
20
40
60
80
100
120
140

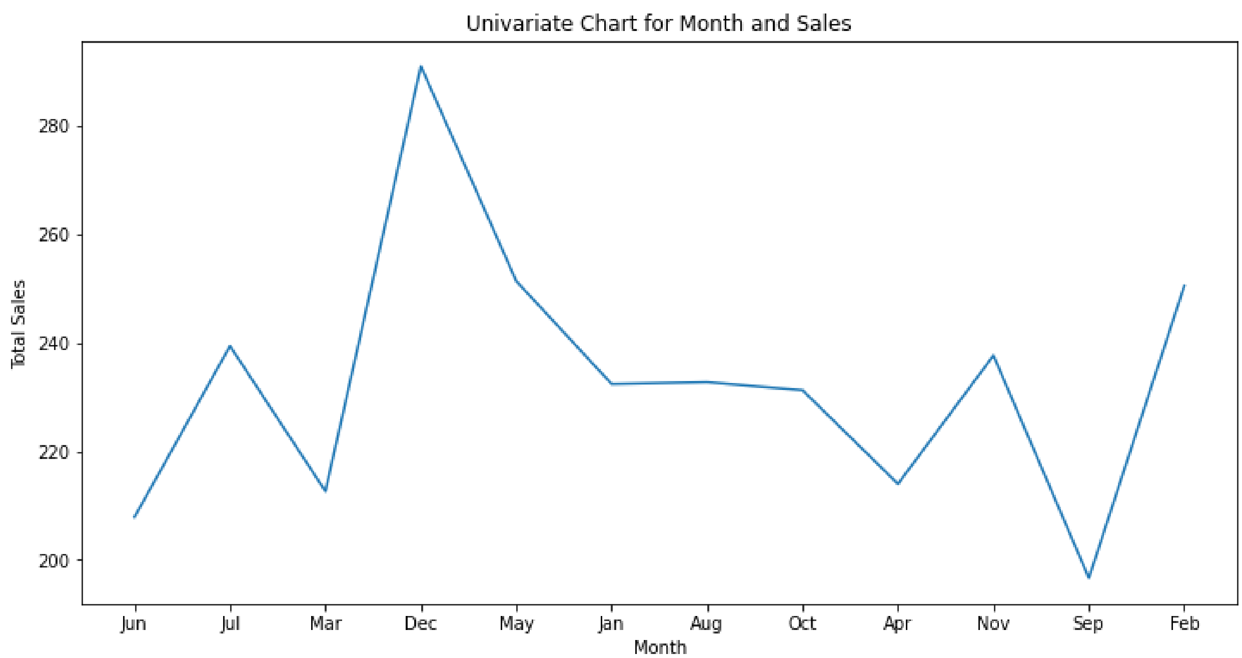
```
Q14) Average Sales based on Specialisation:
Specialisation
Specialisation1     171.599727
Specialisation10    130.044243
Specialisation11    106.872860
Specialisation12     66.252783
Specialisation13    234.074800
Specialisation14    124.797812
Specialisation15    129.600280
Specialisation16    170.053363
Specialisation17    234.976562
Specialisation18    166.272800
Specialisation19    122.689133
Specialisation2     155.996772
Specialisation20    209.146895
Specialisation21    203.732750
Specialisation22    106.925750
Specialisation23    277.191920
Specialisation24     81.176400
Specialisation25     90.791642
Specialisation26    250.757211
Specialisation27    119.352718
Specialisation28     96.545765
Specialisation3     172.143183
Specialisation31    180.635783
Specialisation33    202.323111
Specialisation34    109.033434
Specialisation35     53.040000
Specialisation37     91.583000
Specialisation38     75.821286
Specialisation39     85.267660
Specialisation4     300.532188
Specialisation40    147.677628
Specialisation41    359.269227
Specialisation42    129.751045
Specialisation43     97.807162
Specialisation44    151.843294
```

```
Q15) Frequency of return quantity:
1     901
2     404
3     161
4      70
5      50
6      18
10     14
12      7
7       7
15      6
9       6
20      4
18      3
8       3
50      3
17      3
14      3
30      2
22      2
21      2
19      1
24      1
45      1
13      1
16      1
11      1
44      1
48      1
23      1
40      1
42      1
32      1
Name: ReturnQuantity, dtype: int64
```

Univariant Analysis:

How does the 'Final_Sales' vary over different 'month' values?

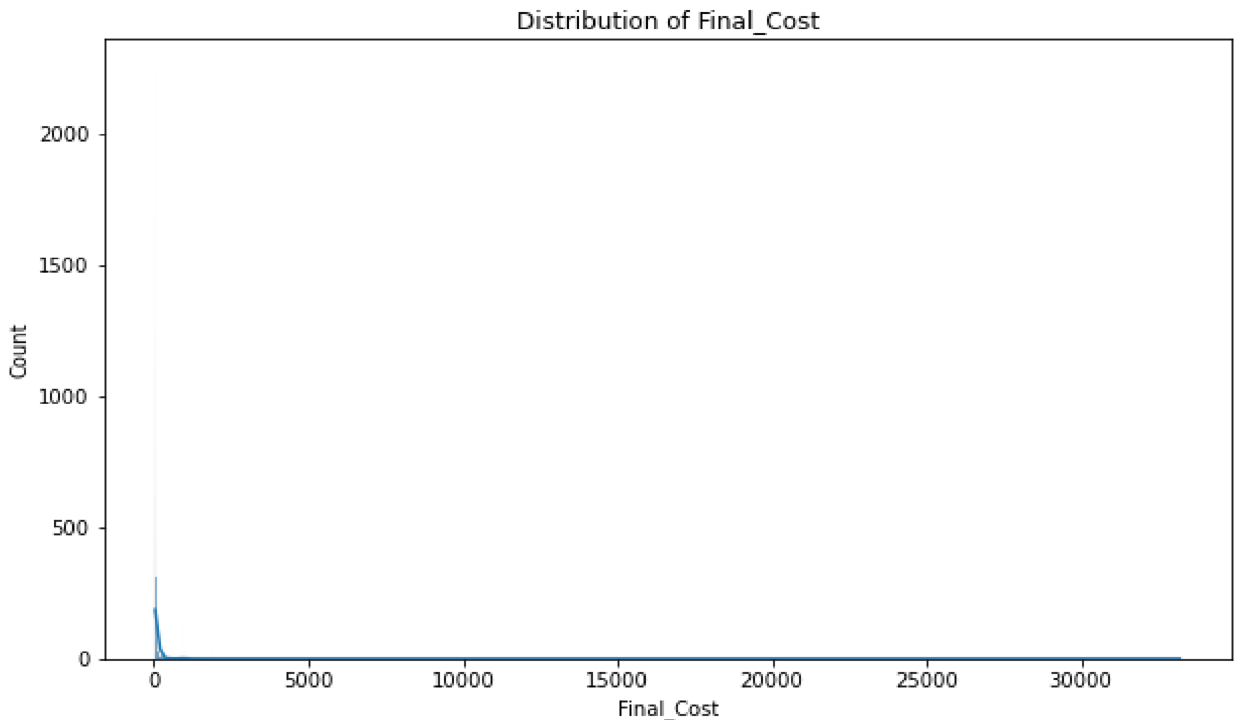
```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns


# Univariate chart for month and sales using a line plot

plt.figure(figsize=(12, 6))

sns.lineplot(x='month', y='Final_Sales', data=df, ci=None)

plt.title('Univariate Chart for Month and Sales')

plt.xlabel('Month')

plt.ylabel('Total Sales')

plt.show()
```



Is the distribution of 'Final_cost' skewed or symmetric?

```
# Is the distribution of 'Final_cost' skewed or symmetric?

import pandas as pd

import seaborn as sns
```

```
import matplotlib.pyplot as plt

# Plot a histogram and kernel density plot for 'Final_cost'
plt.figure(figsize=(10, 6))
sns.histplot(df['Final_Cost'], kde=True)
plt.title('Distribution of Final_Cost')
plt.xlabel('Final_Cost')
plt.show()

# Calculate skewness
skewness = df['Final_Cost'].skew()

# Assess skewness
if skewness > 0:
    print(f"The distribution is right-skewed (positively skewed) with skewness value: {skewness:.2f}")
elif skewness < 0:
    print(f"The distribution is left-skewed (negatively skewed) with skewness value: {skewness:.2f}")
else:
    print("The distribution is approximately symmetric.")
```

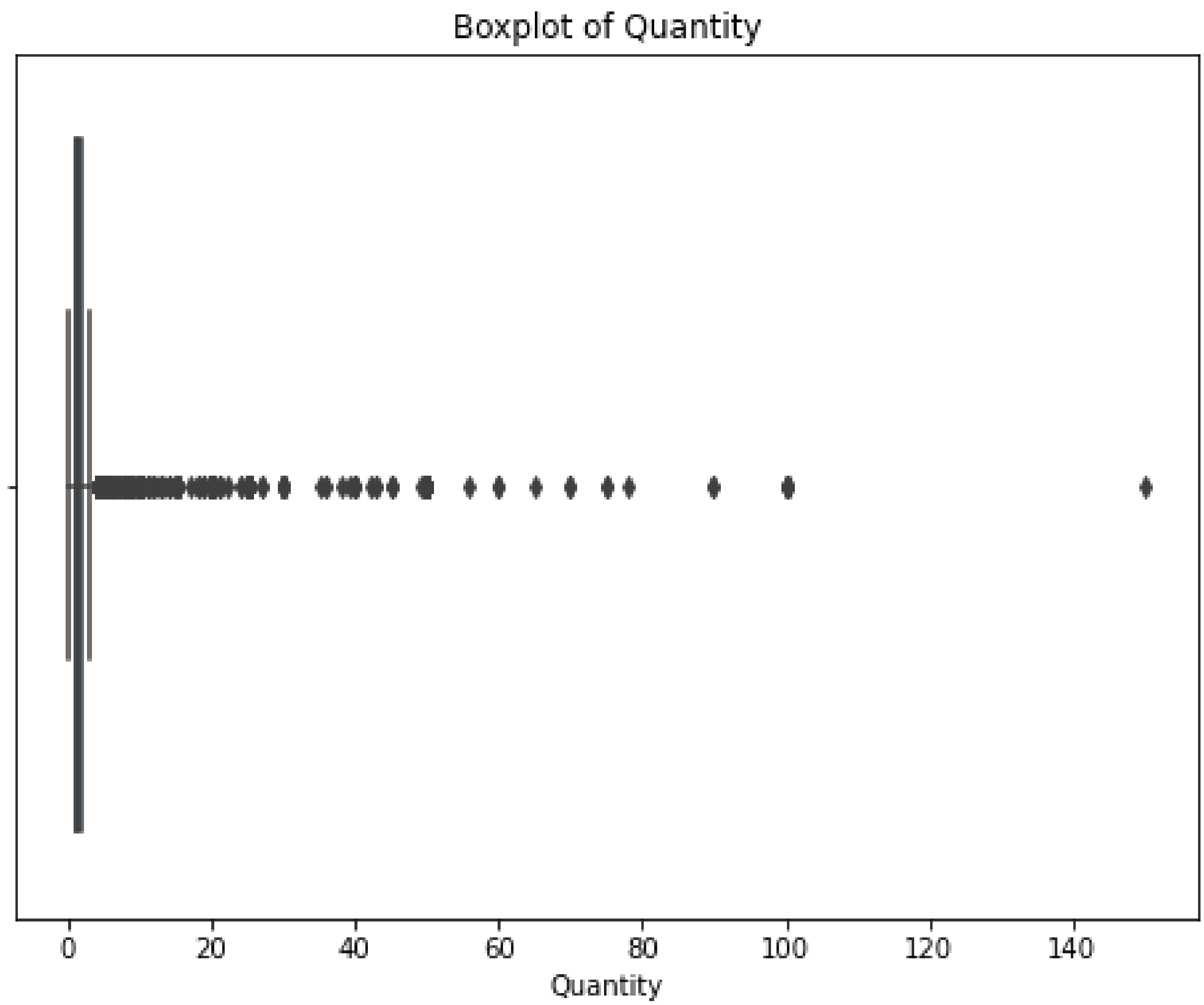


Are there extreme values in the 'Quantity' column?

```
import pandas as pd

import matplotlib.pyplot as plt

# Check for extreme values using a boxplot

plt.figure(figsize=(8, 6))

sns.boxplot(x='Quantity', data=df)

plt.title('Boxplot of Quantity')

plt.show()


# Calculate the interquartile range (IQR)

Q1 = df['Quantity'].quantile(0.25)

Q3 = df['Quantity'].quantile(0.75)

IQR = Q3 - Q1


# Identify potential outliers using the IQR method

outliers = (df['Quantity'] < (Q1 - 1.5 * IQR)) | (df['Quantity'] > (Q3 + 1.5 * IQR))
```

Display the number of potential outliers

print(f"Number of potential outliers in Quantity: {outliers.sum()}")



What are the 25th, 50th, and 75th percentiles of 'Final_Sales'?

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

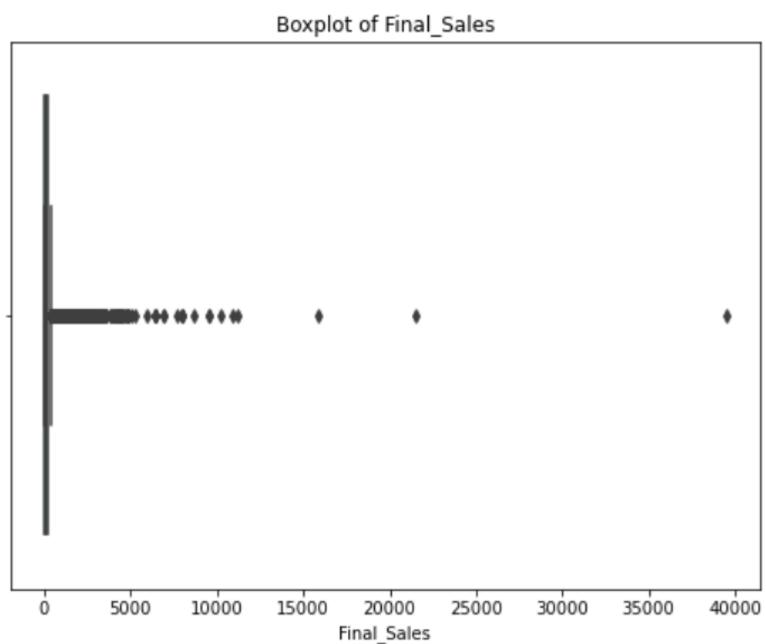
Calculate percentiles using describe method

percentiles = df['Final_Sales'].describe(percentiles=[.25, .5, .75])[['25%', '50%', '75%']]

print("Percentiles of Final_Sales:")

```
print(percentiles)


# Create a boxplot for 'Final_Sales'

plt.figure(figsize=(8, 6))

sns.boxplot(x='Final_Sales', data=df)

plt.title('Boxplot of Final_Sales')

plt.xlabel('Final_Sales')

plt.show()
```

```
Percentiles of Final_Sales:
25%     47.815
50%     86.424
75%    181.000
Name: Final_Sales, dtype: float64
```

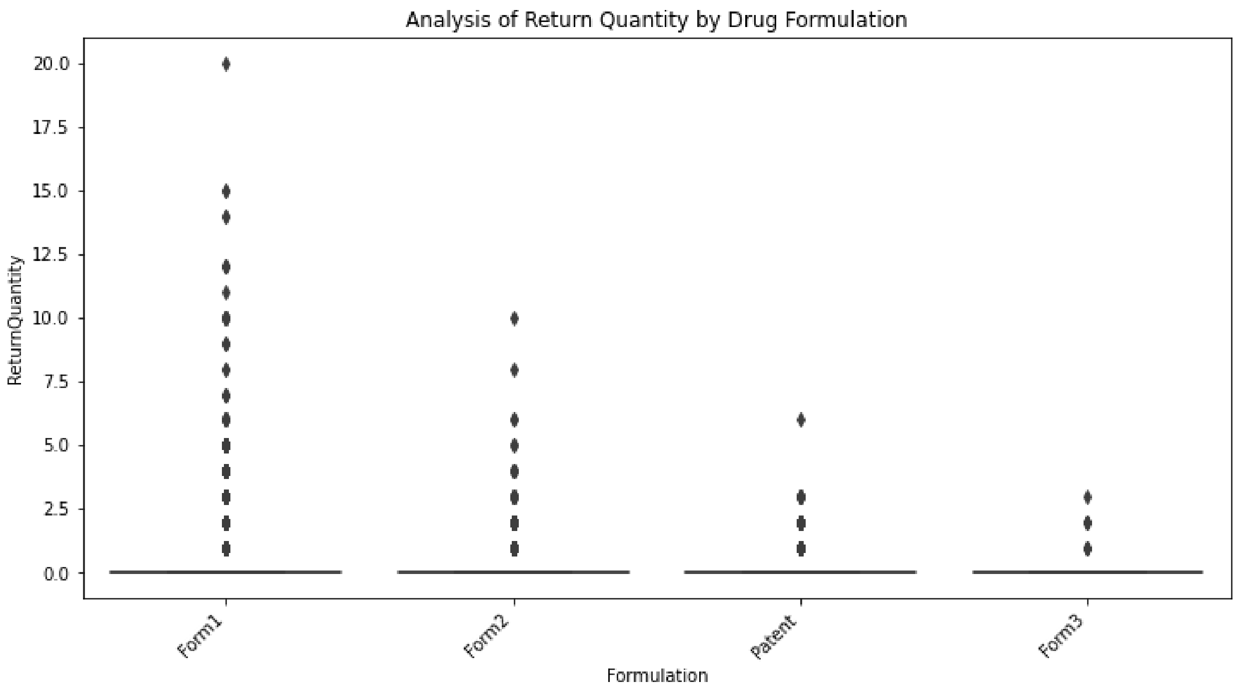


Bivariant Analysis:

Return quantity by drug formulation

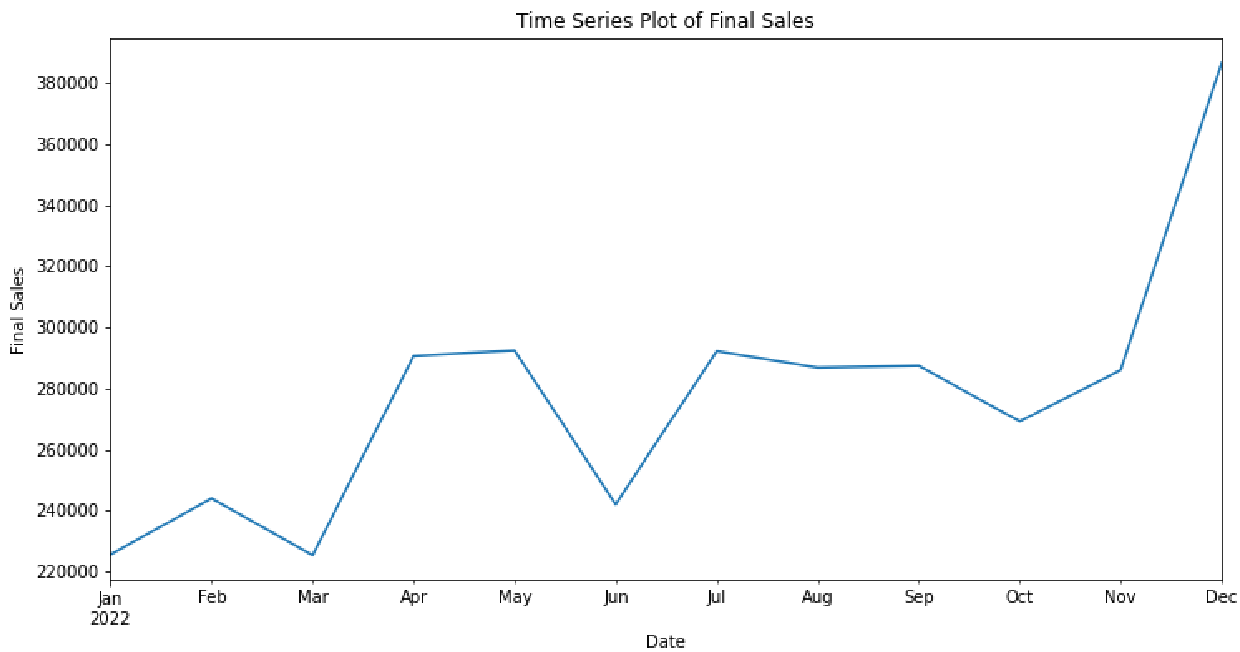
```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns


# Analysis of Return Quantity by Drug Formulation
```

```
plt.figure(figsize=(12, 6))

sns.boxplot(x='Formulation', y='ReturnQuantity', data=df)

plt.title('Analysis of Return Quantity by Drug Formulation')

plt.xticks(rotation=45, ha='right')

plt.show()
```



How Month Effect Final sales

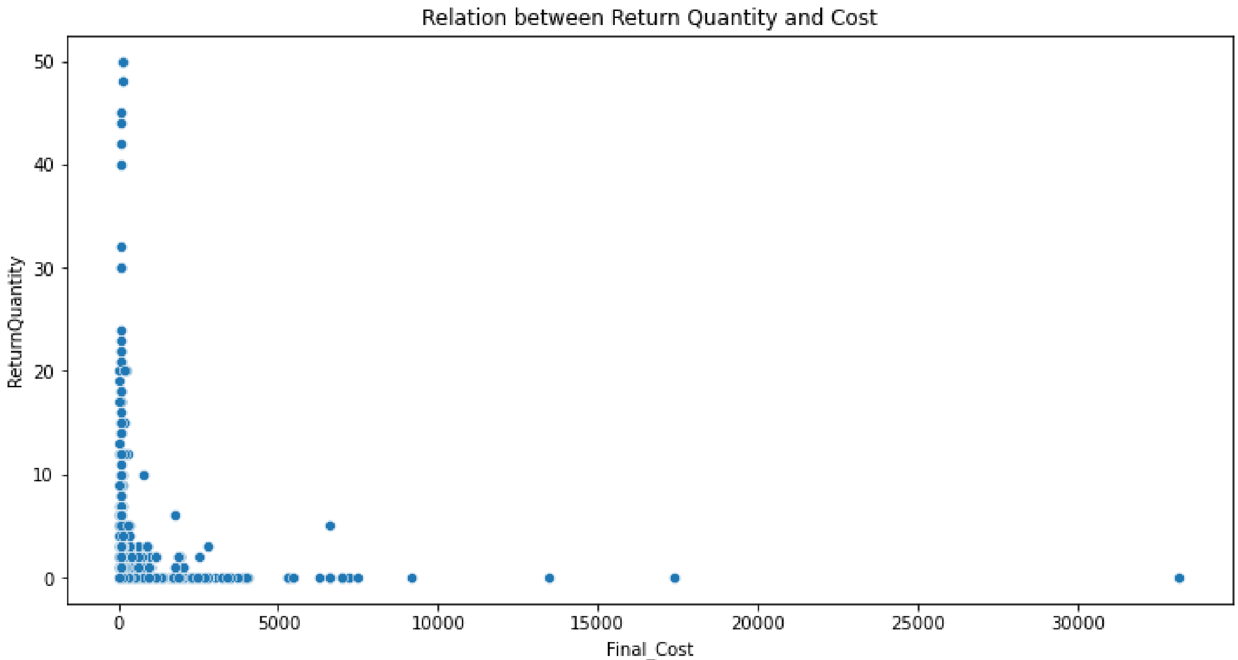
```
# Time Series Plot for Final Sales

df['Dateofbill'] = pd.to_datetime(df['Dateofbill'])  # Convert Dateofbill to datetime if it's not already

df_time_series = df.set_index('Dateofbill')

plt.figure(figsize=(12, 6))

df_time_series['Final_Sales'].resample('M').sum().plot()

plt.title('Time Series Plot of Final Sales')

plt.xlabel('Date')

plt.ylabel('Final Sales')

plt.show()
```



Here we can see we have more sales on December

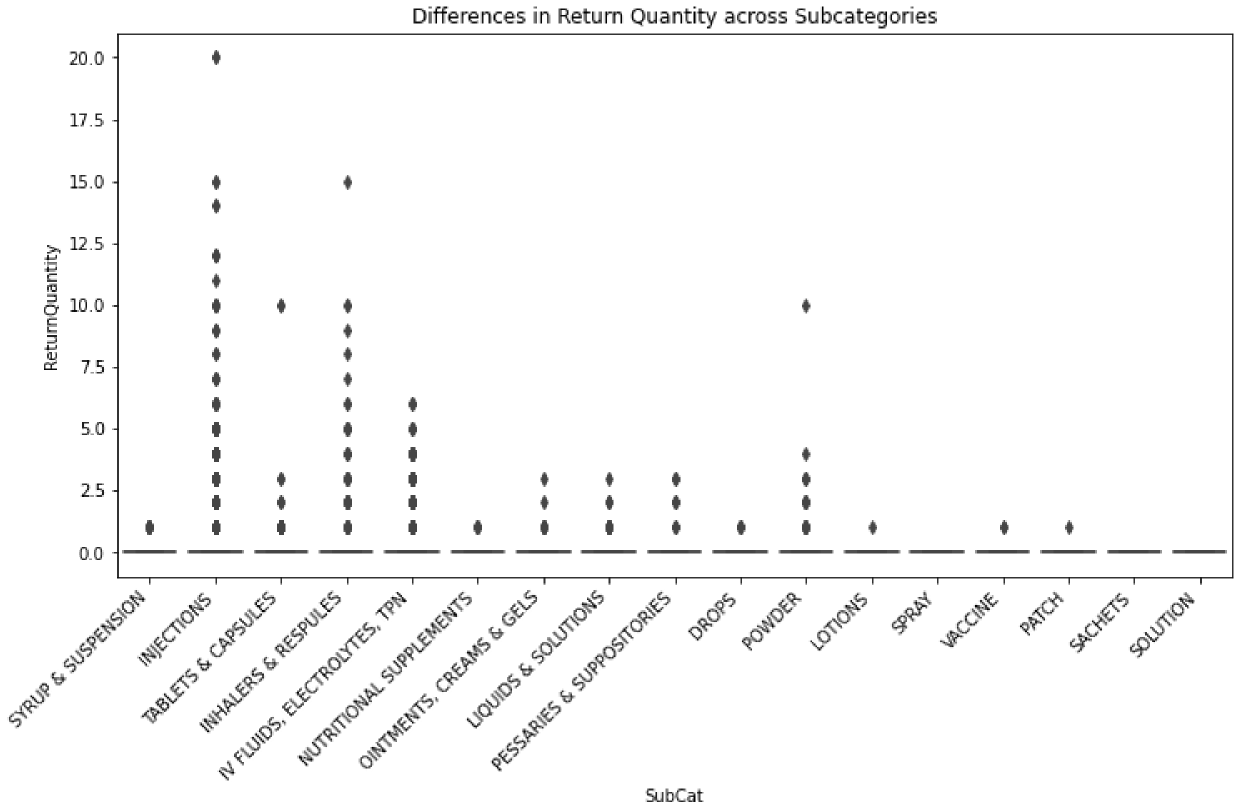
Explore how Return Quantity relates to the cost of the products

```
plt.figure(figsize=(12, 6))

sns.scatterplot(x='Final_Cost', y='ReturnQuantity', data=df)

plt.title('Relation between Return Quantity and Cost')

plt.show()
```



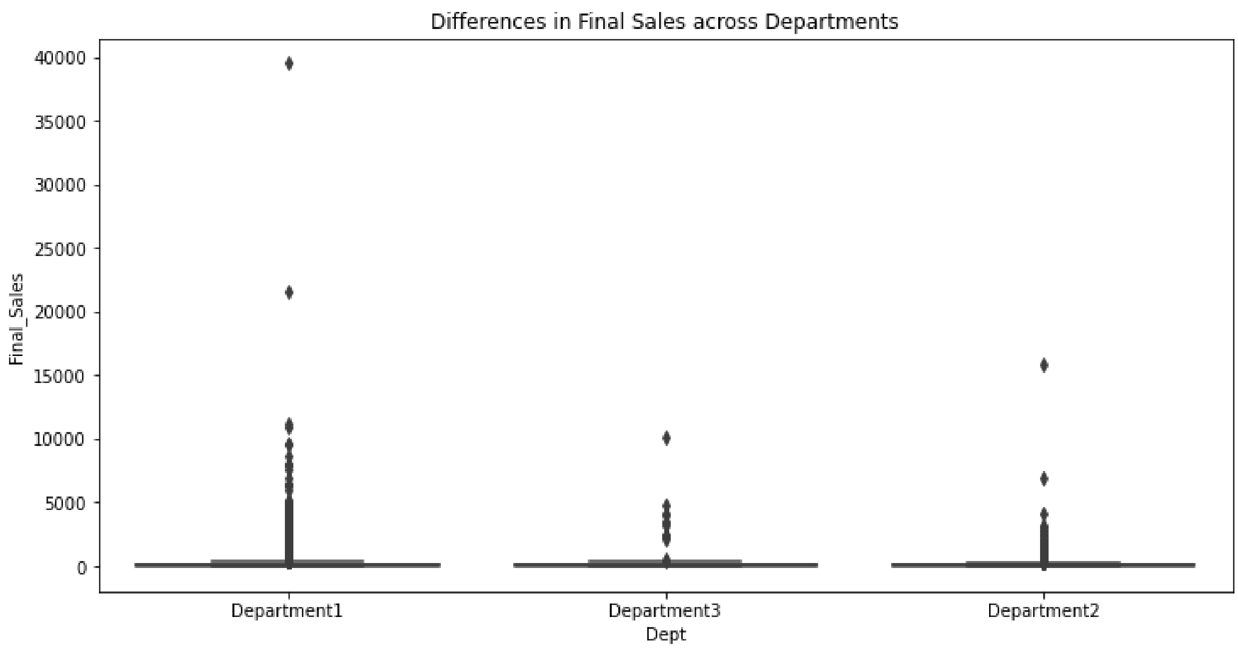
Differences in Return Quantity and Final Sales across different Subcategories

```
plt.figure(figsize=(12, 6))

sns.boxplot(x='SubCat', y='ReturnQuantity', data=df)

plt.title('Differences in Return Quantity across Subcategories')

plt.xticks(rotation=45, ha='right')

plt.show()
```



Differences in Final Sales across different Departments

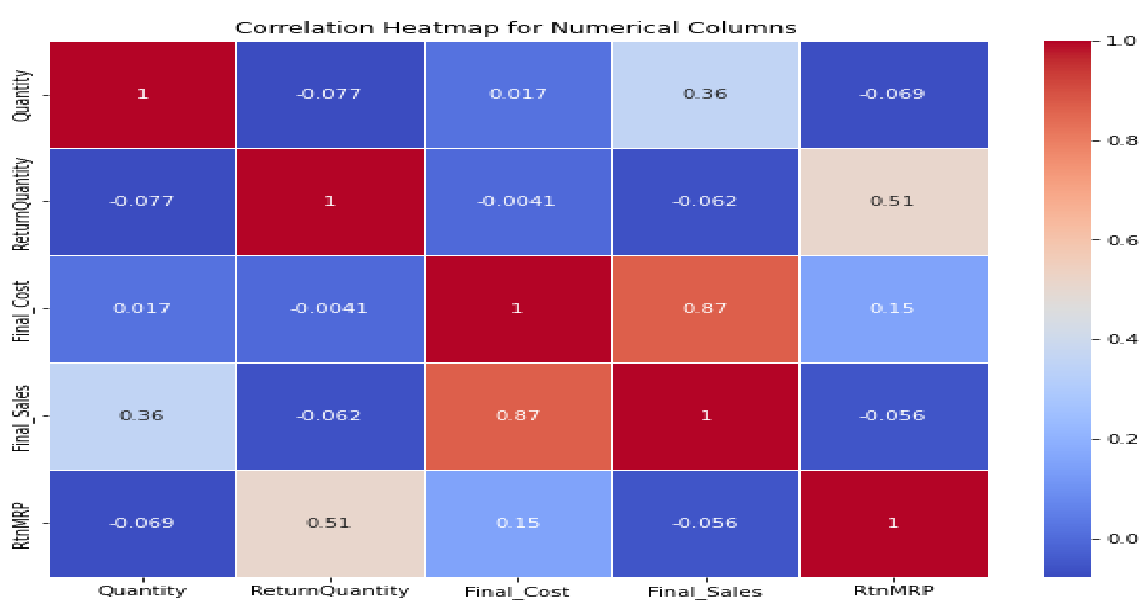
```
plt.figure(figsize=(12, 6))

sns.boxplot(x='Dept', y='Final_Sales', data=df)

plt.title('Differences in Final Sales across Departments')

plt.show()
```



Multi Variant Analysis:

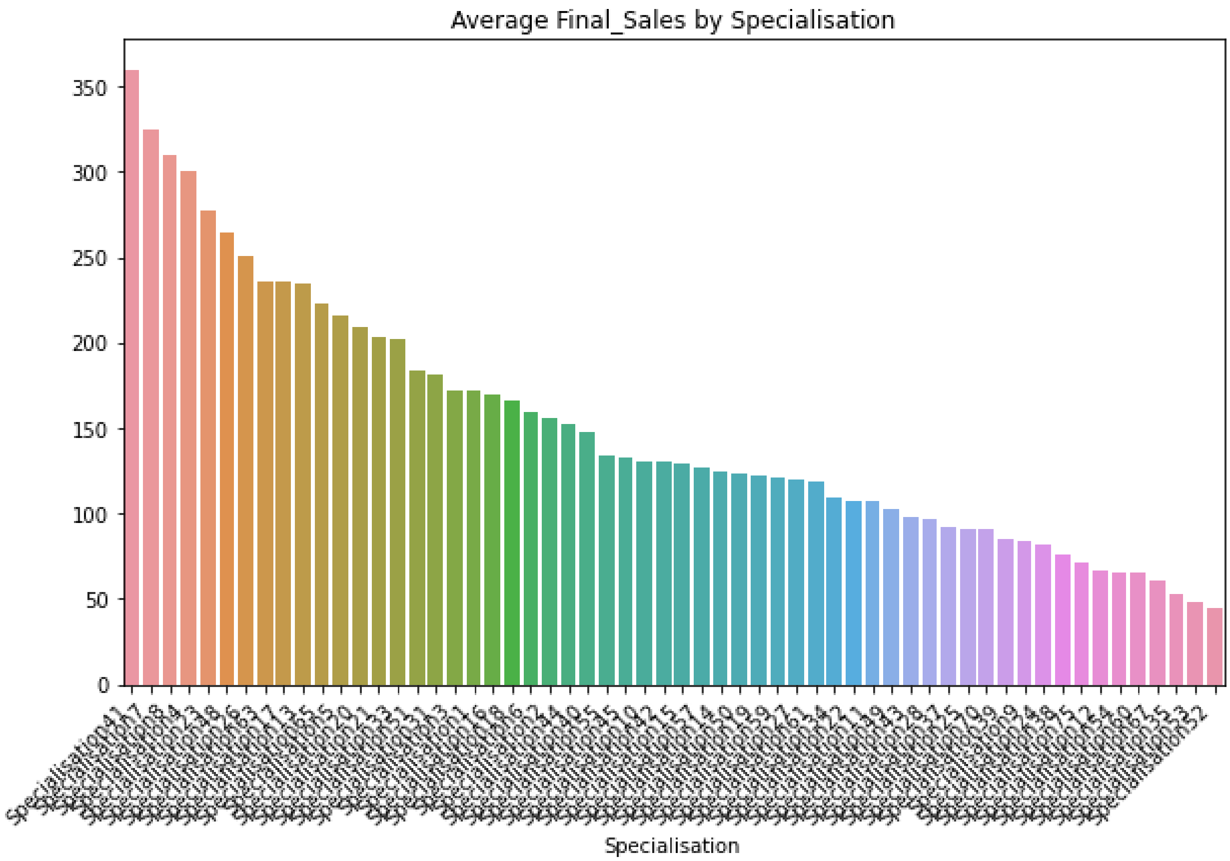
Correlation heatmap for numerical columns

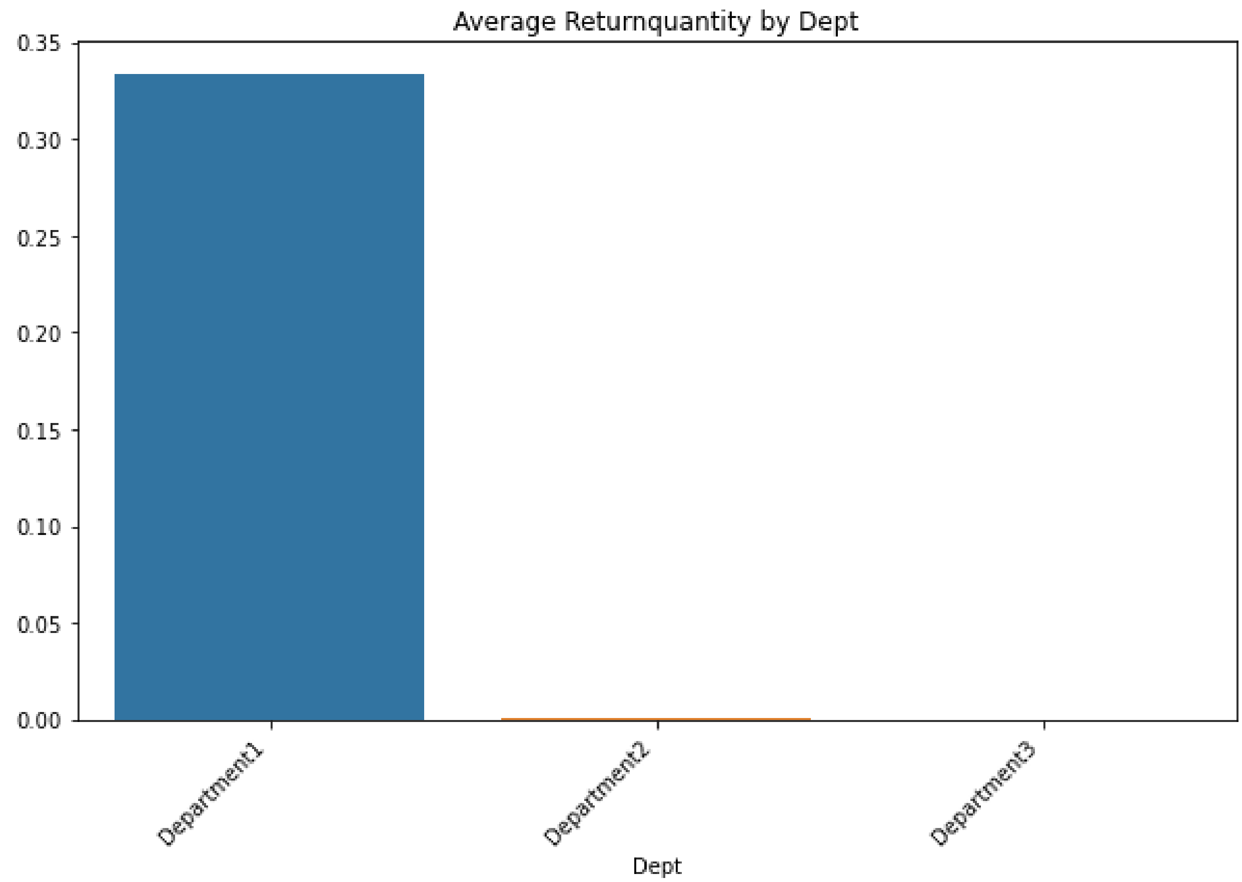
```
correlation_matrix = df[['Quantity', 'Returnquantity', 'Final_cost', 'Final_Sales', 'RtnMRP']].corr()

plt.figure(figsize=(10, 8))

sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)

plt.title('Correlation Heatmap for Numerical Columns')

plt.show()
```



How does the Specialisation or Dept impact the Final_Sales or Returnquantity?

```
# Bar plot for average Final_Sales by Specialisation

avg_sales_by_specialisation =
df.groupby('Specialisation')['Final_Sales'].mean().sort_values(ascending=False)

plt.figure(figsize=(10, 6))

sns.barplot(x=avg_sales_by_specialisation.index, y=avg_sales_by_specialisation.values)

plt.title('Average Final_Sales by Specialisation')

plt.xticks(rotation=45, ha='right')

plt.show()


# Bar plot for average Returnquantity by Dept

avg_return_quantity_by_dept =
df.groupby('Dept')['ReturnQuantity'].mean().sort_values(ascending=False)

plt.figure(figsize=(10, 6))

sns.barplot(x=avg_return_quantity_by_dept.index, y=avg_return_quantity_by_dept.values)

plt.title('Average Returnquantity by Dept')

plt.xticks(rotation=45, ha='right')

plt.show()
```


Average Final_Sales by Specialisation
350
300
250
200
150
100
50
0
Specialisation



Department 1 has highest average returns and specialization41 has highest average sales

ANOVA Analysis to check if month effect sales

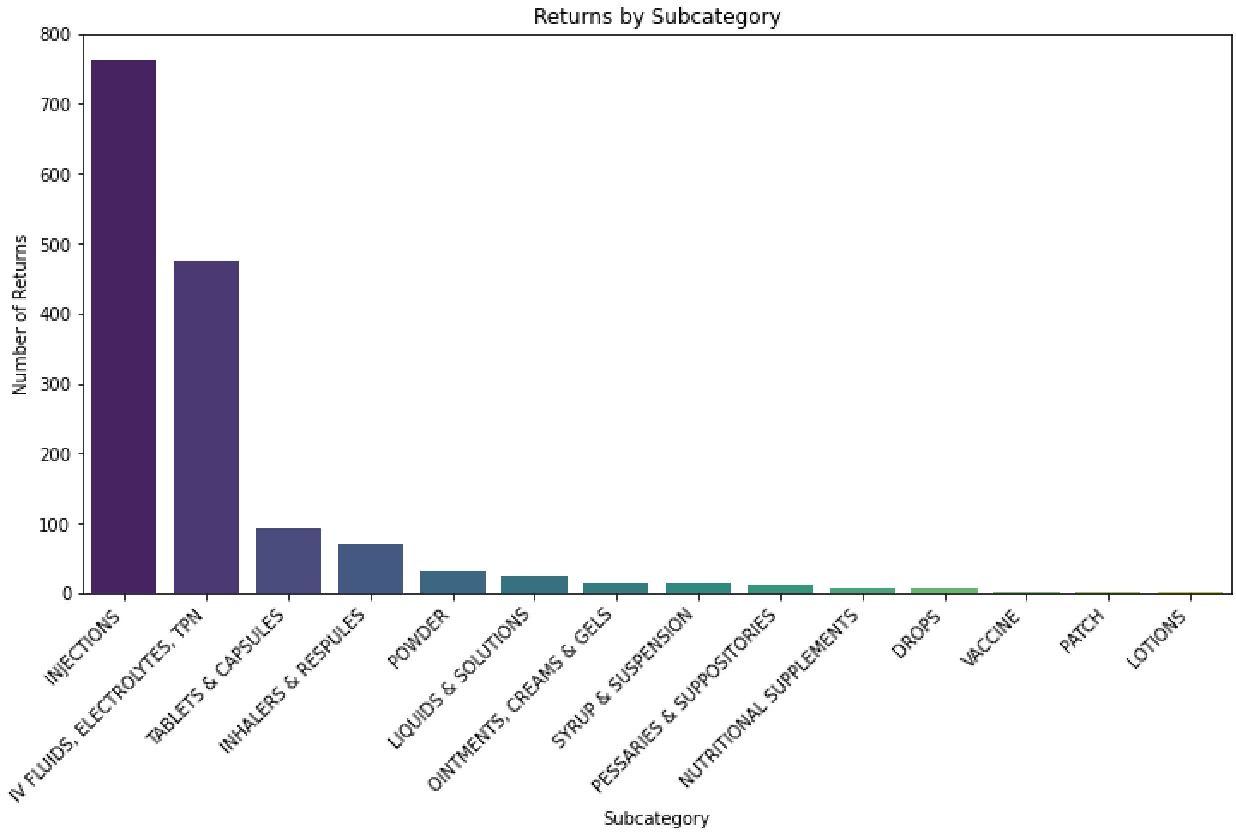
```
import pandas as pd
from scipy.stats import f_oneway


# One-way ANOVA
dept_groups = [df['Final_Sales'][df['month'] == dept] for dept in df['month'].unique()]


# Perform one-way ANOVA
anova_result = f_oneway(*dept_groups)


# Display the ANOVA result
print("ANOVA Result:")
```

```
print(anova_result)


# Check if the p-value is less than a significance level (e.g., 0.05) to determine significance
if anova_result.pvalue < 0.05:
    print("There is a significant effect of 'Month' on 'Final_Sales'.")
else:
    print("There is no significant effect of 'Month' on 'Final_Sales'.")
```

```
ANOVA Result:
F_onewayResult(statistic=1.755518422272744, pvalue=0.055855110206130425)
There is no significant effect of 'Month' on 'Final_Sales'.
```

Which subcategory has more return items

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


# Filter rows where Typeofsales is 'return'
returns_df = df[df['Typeofsales'] == 'Return']


# Count the returns for each subcategory
returns_by_subcategory = returns_df['SubCat'].value_counts()


# Plot a bar chart
plt.figure(figsize=(12, 6))
sns.barplot(x=returns_by_subcategory.index, y=returns_by_subcategory.values, palette="viridis")
plt.title('Returns by Subcategory')
plt.xlabel('Subcategory')
plt.ylabel('Number of Returns')
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for better visibility
plt.show()
```



insights:

From Sub category Injections,tablets and IV fuilds, electrolytes,TPN has highest number of returns

Department 1 has highest average returns and specialization41 has highest average sales

Department 1 contains highest average sales in department compared to other two departments

Form1 has more return quantity , form3 has lowest return quantities

In sales we can see dec month has highest sales compared to other months

Conclusion

From the above analysis we can see the sub categories Injections and tablets and IV fuilds were returned frequently , so there may be some dissatisfaction with this products we need to check for those reasons for the items returned, the data we analyse does not consists that field so there is no correct finding for the reasons

We can these two categories injections, tablets, IV fluids return frequently and it costs lots of money, finding the reason for these returns we can reduce the amount of money lose to inventory

We can find highest average sales in December month and the highest returns in may month. Finding the reason, we can increase the sales in other months.