In [365…
```
from IPython.display import Image

#Display image
Image(filename= "C:\\Users\eyang\OneDrive\Desktop\OQD-INTERNSHIP\Data Analyst Jo
```

Out[365…

QUANTIUM PROJECT REPORT

Introduction

This report provides an complete analysis of customer purchases behavior of a functional chips company as well as transaction behaviors under the **Quantium Data Analysis Job simulation**. his project is part of an internship program hosted by **ONLY QUALITY DATA**

Two dataset have been used to complete this analysis

- **Q_Transaction_Data** that contain all the trasactiins data,
- **Q_Purchase_Behaviour** that contain all the infortion about the differents customers

Objective:

The goal of this project is to use python to examine and clean trasaction and customers data, create charts and graphs that can be used for commercial recommendations and also to identify the customer segment based on the purchasing behaviour.

Dataset Overview:

Q_Transaction_Data

1. DATE: The date when the transaction occurred. This helps analyze trends over time, such as daily, monthly, or yearly sales patterns.

2. STORE_NBR: A unique identifier for each store where the transaction took place. Useful for store-level analysis, such as comparing sales across different locations.

3. LYLTY_CARD_NBR: A unique loyalty card number assigned to customers. This enables customer-level analysis and links transactions to customer profiles in the

Q_Purchase_Behaviour dataset.

4. TXN_ID: A unique identifier for each transaction. Essential for distinguishing individual transactions, especially when multiple transactions occur on the same day.
5. PROD_NBR: A unique identifier for each product. Useful for analyzing product-level sales and linking to product details.
6. PROD_NAME:The name of the product purchased in the transaction. Helps in identifying specific items for category-level analysis or understanding customer preferences.
7. PROD_QTY:The quantity of the product purchased in the transaction. Useful for calculating total product sales and understanding bulk purchasing behavior.
8. TOT_SALES:The total sales amount for the transaction (likely the product of PROD_QTY and the price per unit). Crucial for revenue analysis and profitability metrics.

Q_Purchase_Behaviour

1. LYLTY_CARD_NBR:A unique identifier assigned to each customer's loyalty card. This column is essential for linking the customer data to their corresponding transactions in the Q_Transaction_Data dataset. It enables customer-level insights and segmentation.
2. LIFESTAGE:Represents the demographic life stage of the customer, such as Young Singles/Couples, Young Families, or Older Singles/Couples. This column provides critical information for analyzing customer behavior and targeting specific demographics.
3. PREMIUM_CUSTOMER:Indicates the premium status of the customer, categorized as Premium, Mainstream, or Budget. This helps in understanding purchasing power and customer value, allowing segmentation based on spending habits.

In [266…
```
#Import Python Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
```

In [268…
```
# Suppress FutureWarnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

Data Inspection & Cleaning

In [271…
```
#Load Purchase Behaviour Dataset
customer_df= pd.read_excel("C:\\Users\eyang\OneDrive\Desktop\OQD-INTERNSHIP\Data
```

```
#Load Customer Transaction Dataset
transaction_df= pd.read_excel("C:\\Users\eyang\OneDrive\Desktop\OQD-INTERNSHIP\D
```

In [272…
```
#Display the first 5 rows of customer dataset
customer_df.head()
```

Out[272…

	LYLTY_CARD_NBR	LIFESTAGE	PREMIUM_CUSTOMER
0	1000	YOUNG SINGLES/COUPLES	Premium
1	1002	YOUNG SINGLES/COUPLES	Mainstream
2	1003	YOUNG FAMILIES	Budget
3	1004	OLDER SINGLES/COUPLES	Mainstream
4	1005	MIDAGE SINGLES/COUPLES	Mainstream

In [273…
```
#Display the first 5 rows of transactions dataset
transaction_df.head()
```

Out[273…

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_Q
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	
2	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	
3	2018-08-17	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	
4	2018-08-18	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	

In [274…
```
#Identify the columns in the Customer Dataset
customer_df.columns
```

Out[274…
```
Index(['LYLTY_CARD_NBR', 'LIFESTAGE', 'PREMIUM_CUSTOMER'], dtype='object')
```

In [275…
```
#Describing the transactions Dataset
transaction_df.describe()
```

Out[275…

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR
count	264836	264836.00000	2.648360e+05	2.648360e+05	264836.000000
mean	2018-12-30 00:52:12.879215616	135.08011	1.355495e+05	1.351583e+05	56.583157
min	2018-07-01 00:00:00	1.00000	1.000000e+03	1.000000e+00	1.000000
25%	2018-09-30 00:00:00	70.00000	7.002100e+04	6.760150e+04	28.000000
50%	2018-12-30 00:00:00	130.00000	1.303575e+05	1.351375e+05	56.000000
75%	2019-03-31 00:00:00	203.00000	2.030942e+05	2.027012e+05	85.000000
max	2019-06-30 00:00:00	272.00000	2.373711e+06	2.415841e+06	114.000000
std	NaN	76.78418	8.057998e+04	7.813303e+04	32.826638

In [276…
```
#Check for nulls in customer dataset
customer_df.isnull().sum()
```

Out[276…
```
LYLTY_CARD_NBR      0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
dtype: int64
```

In [277…
```
#Check for nulls in transaction  Dataset
transaction_df.isnull().sum()
```

Out[277…
```
DATE              0
STORE_NBR         0
LYLTY_CARD_NBR    0
TXN_ID            0
PROD_NBR          0
PROD_NAME         0
PROD_QTY          0
TOT_SALES         0
dtype: int64
```

In [278…
```
#Show Information of Transaction Dataset
transaction_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264836 entries, 0 to 264835
Data columns (total 8 columns):
 #   Column          Non-Null Count   Dtype         
---  ------          --------------   -----         
 0   DATE            264836 non-null  datetime64[ns]
 1   STORE_NBR       264836 non-null  int64         
 2   LYLTY_CARD_NBR  264836 non-null  int64         
 3   TXN_ID          264836 non-null  int64         
 4   PROD_NBR        264836 non-null  int64         
 5   PROD_NAME       264836 non-null  object        
 6   PROD_QTY        264836 non-null  int64         
 7   TOT_SALES       264836 non-null  float64       
dtypes: datetime64[ns](1), float64(1), int64(5), object(1)
memory usage: 16.2+ MB
```

In [279…
```
#Show Information of Customer Dataset
customer_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72637 entries, 0 to 72636
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   LYLTY_CARD_NBR    72637 non-null  int64 
 1   LIFESTAGE         72637 non-null  object
 2   PREMIUM_CUSTOMER  72637 non-null  object
dtypes: int64(1), object(2)
memory usage: 1.7+ MB
```

In [280…
```
#Check the number of row in a customer dataset
len(customer_df)
```

Out[280… 72637

In [281…
```
#Check the number of row in a transaction dataset
len(transaction_df)
```

Out[281… 264836

In [282…
```
#use print to check the number of rw in the customer dataset
row_count = len(customer_df)
print("The Number of row in the customer table:", len(customer_df) )
```

```
The Number of row in the customer table: 72637
```

In [283…
```
#Sorting the transaction according to the Total Sales
transaction_df.sort_values(by='TOT_SALES', ascending = False)
```

Out[283...

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PRO
69762	2018-08-19	226	226000	226201	4	Dorito Corn Chp Supreme 380g	
69763	2019-05-20	226	226000	226210	4	Dorito Corn Chp Supreme 380g	
69496	2018-08-15	49	49303	45789	14	Smiths Crnkle Chip Orgnl Big Bag 380g	
55558	2019-05-14	190	190113	190914	14	Smiths Crnkle Chip Orgnl Big Bag 380g	
171815	2018-08-17	24	24095	20797	14	Smiths Crnkle Chip Orgnl Big Bag 380g	
...	...	...	...	...	...	...	
259695	2018-11-13	41	41089	38002	76	Woolworths Medium Salsa 300g	
259707	2018-10-18	41	41267	38201	76	Woolworths Medium Salsa 300g	
197005	2018-08-11	167	167121	168928	76	Woolworths Medium Salsa 300g	
216449	2019-03-01	264	264032	262778	76	Woolworths Medium Salsa 300g	
150019	2018-11-01	268	268303	264733	35	Woolworths Mild Salsa 300g	

264836 rows × 8 columns

In [284...

```
#removing  the rows with outliers on transaction dataset
transaction_df = transaction_df[transaction_df['PROD_QTY'] !=200]
transaction_df
```

Out[284...

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PR
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	
2	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	
3	2018-08-17	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	
4	2018-08-18	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	
...	...	...	...	...	...	...	
264831	2019-03-09	272	272319	270088	89	Kettle Sweet Chilli And Sour Cream 175g	
264832	2018-08-13	272	272358	270154	74	Tostitos Splash Of Lime 175g	
264833	2018-11-06	272	272379	270187	51	Doritos Mexicana 170g	
264834	2018-12-27	272	272379	270188	42	Doritos Corn Chip Mexican Jalapeno 150g	
264835	2018-09-22	272	272380	270189	74	Tostitos Splash Of Lime 175g	

264834 rows × 8 columns

In [285...

```
#Verifying changes
transaction_df.describe()
```

Out[285...

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NB
count	264834	264834.000000	2.648340e+05	2.648340e+05	264834.00000
mean	2018-12-30 00:52:10.292938240	135.079423	1.355488e+05	1.351576e+05	56.58355
min	2018-07-01 00:00:00	1.000000	1.000000e+03	1.000000e+00	1.00000
25%	2018-09-30 00:00:00	70.000000	7.002100e+04	6.760050e+04	28.00000
50%	2018-12-30 00:00:00	130.000000	1.303570e+05	1.351365e+05	56.00000
75%	2019-03-31 00:00:00	203.000000	2.030940e+05	2.026998e+05	85.00000
max	2019-06-30 00:00:00	272.000000	2.373711e+06	2.415841e+06	114.00000
std	NaN	76.784063	8.057990e+04	7.813292e+04	32.82644

In [286...

```
#Merge both tables using the loyalty card number as reference
merged_df = pd.merge(transaction_df,customer_df, on='LYLTY_CARD_NBR')
merged_df
```

Out[286…

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PRO
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	
2	2018-11-10	1	1307	346	96	WW Original Stacked Chips 160g	
3	2019-03-09	1	1307	347	54	CCs Original 175g	
4	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	
…	…	…	…	…	…	…	
264829	2019-03-09	272	272319	270088	89	Kettle Sweet Chilli And Sour Cream 175g	
264830	2018-08-13	272	272358	270154	74	Tostitos Splash Of Lime 175g	
264831	2018-11-06	272	272379	270187	51	Doritos Mexicana 170g	
264832	2018-12-27	272	272379	270188	42	Doritos Corn Chip Mexican Jalapeno 150g	
264833	2018-09-22	272	272380	270189	74	Tostitos Splash Of Lime 175g	

264834 rows × 10 columns

In [287…
```
merged_df.columns
```

Out[287…
```
Index(['DATE', 'STORE_NBR', 'LYLTY_CARD_NBR', 'TXN_ID', 'PROD_NBR',
       'PROD_NAME', 'PROD_QTY', 'TOT_SALES', 'LIFESTAGE', 'PREMIUM_CUSTOMER'],
      dtype='object')
```

In [288…
```
'''we are changing the data type of  store  number to make sur that it is not re
this will help for a better visualisation'''

merged_df['STORE_NBR'] = merged_df['STORE_NBR'] .astype(str)
merged_df['STORE_NBR'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 264834 entries, 0 to 264833
Series name: STORE_NBR
Non-Null Count   Dtype
--------------   -----
264834 non-null  object
dtypes: object(1)
memory usage: 2.0+ MB
```

Calculating Key Measures

In [290…

```
Total_sales= merged_df['TOT_SALES'].sum()
print('Total Sales is ', Total_sales, 'dollars')

Total_Qty= transaction_df['PROD_QTY'].sum()
print('Total Quantity is ', Total_Qty, 'Units')

Total_members = customer_df['LYLTY_CARD_NBR'].nunique()
print('Total Member is ', Total_members)
```

```
Total Sales is  1933115.0 dollars
Total Quantity is  504724 Units
Total Member is  72637
```

Creating visual

In [292…

```
#Displaying the totale sales by category and top 5 products
total_sales_by_category = merged_df.groupby('PROD_NAME')['TOT_SALES'].sum().rese
top_5_products = total_sales_by_category.sort_values(by='TOT_SALES', ascending =
top_5_products
```

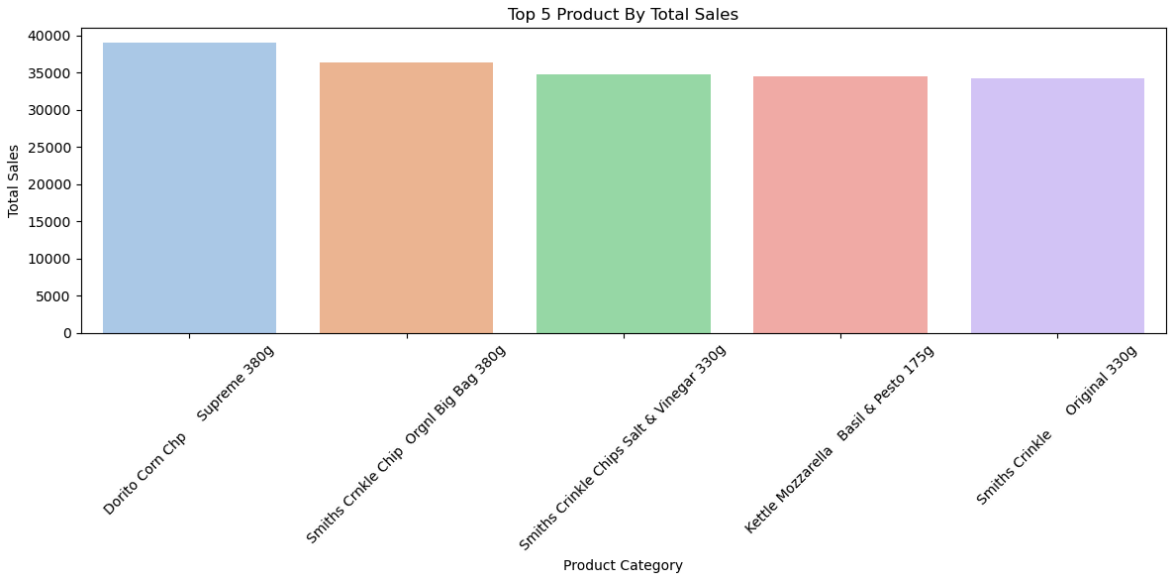
Out[292…

	PROD_NAME	TOT_SALES
11	Dorito Corn Chp Supreme 380g	39052.0
86	Smiths Crnkle Chip Orgnl Big Bag 380g	36367.6
77	Smiths Crinkle Chips Salt & Vinegar 330g	34804.2
33	Kettle Mozzarella Basil & Pesto 175g	34457.4
76	Smiths Crinkle Original 330g	34302.6

In [293…

```
#Show the bar chart of the Top 5 Product By Total Sales
plt.figure(figsize=(12,6))
sns.barplot(x='PROD_NAME', y='TOT_SALES',data=top_5_products, palette='pastel')
plt.xlabel('Product Category')
plt.ylabel('Total Sales')
plt.title('Top 5 Product By Total Sales')
plt.xticks(rotation=45)
plt.tight_layout()

#Show the plot
plt.show()
```



In [294…
```
#Calculate the value counts for the PREMIUM CUSTOMER column
premium_customer_counts = customer_df['PREMIUM_CUSTOMER'].value_counts()
premium_customer_counts
```

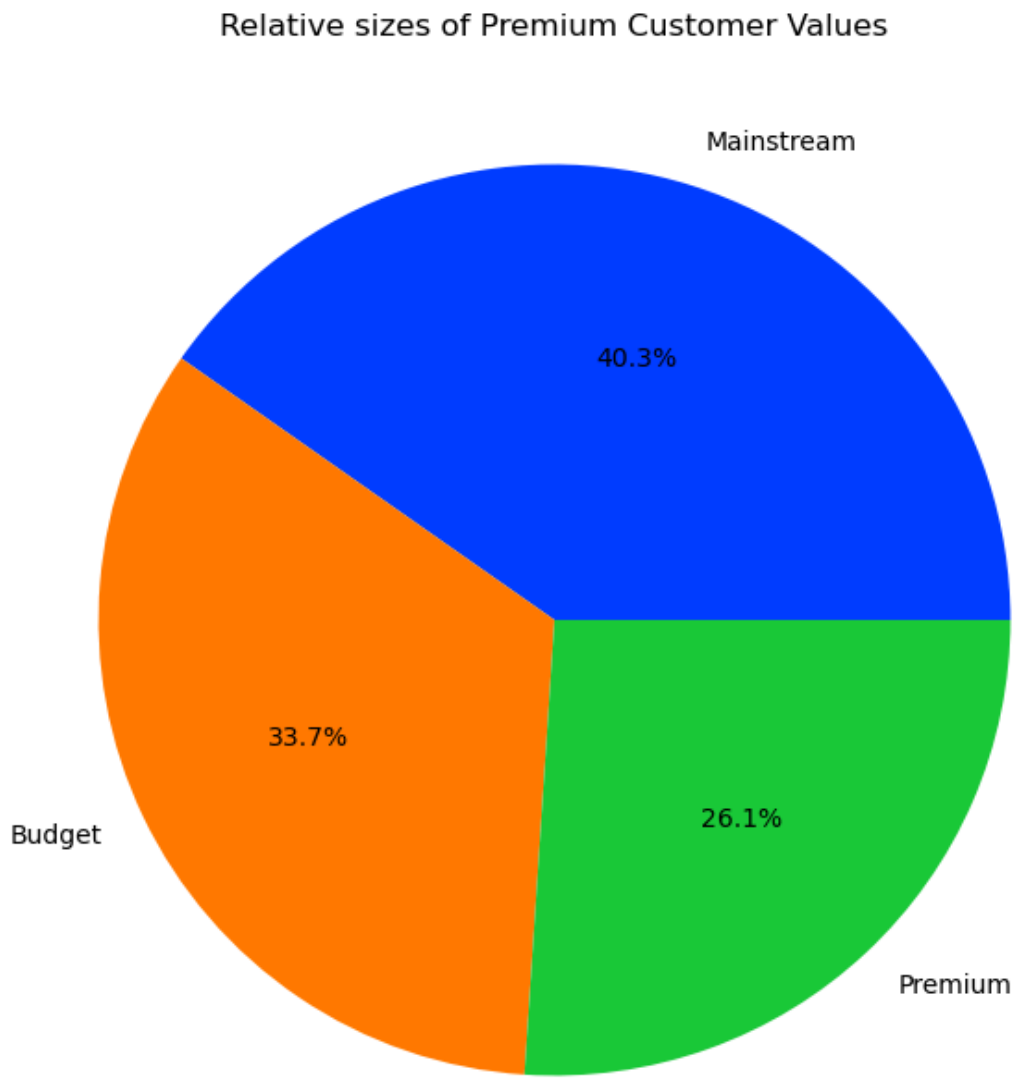
Out[294…
```
PREMIUM_CUSTOMER
Mainstream    29245
Budget        24470
Premium       18922
Name: count, dtype: int64
```

In [295…
```
plt.figure(figsize=(8,8))

premium_customer_counts.plot(kind='pie', autopct='%1.1f%%',  colors=sns.color_pa

plt.title('Relative sizes of Premium Customer Values')
plt.ylabel('') #removes the default ylabel

#show the plot
plt.show()
```



In [296…
```
#Calculate total sales sold by store
total_sales_by_store = merged_df.groupby('STORE_NBR')['TOT_SALES'].sum().reset_i

#Sort by total sales and get the top 5 stores
top_5_stores_by_sales = total_sales_by_store.sort_values(by='TOT_SALES', ascendi
top_5_stores_by_sales
```

Out[296…

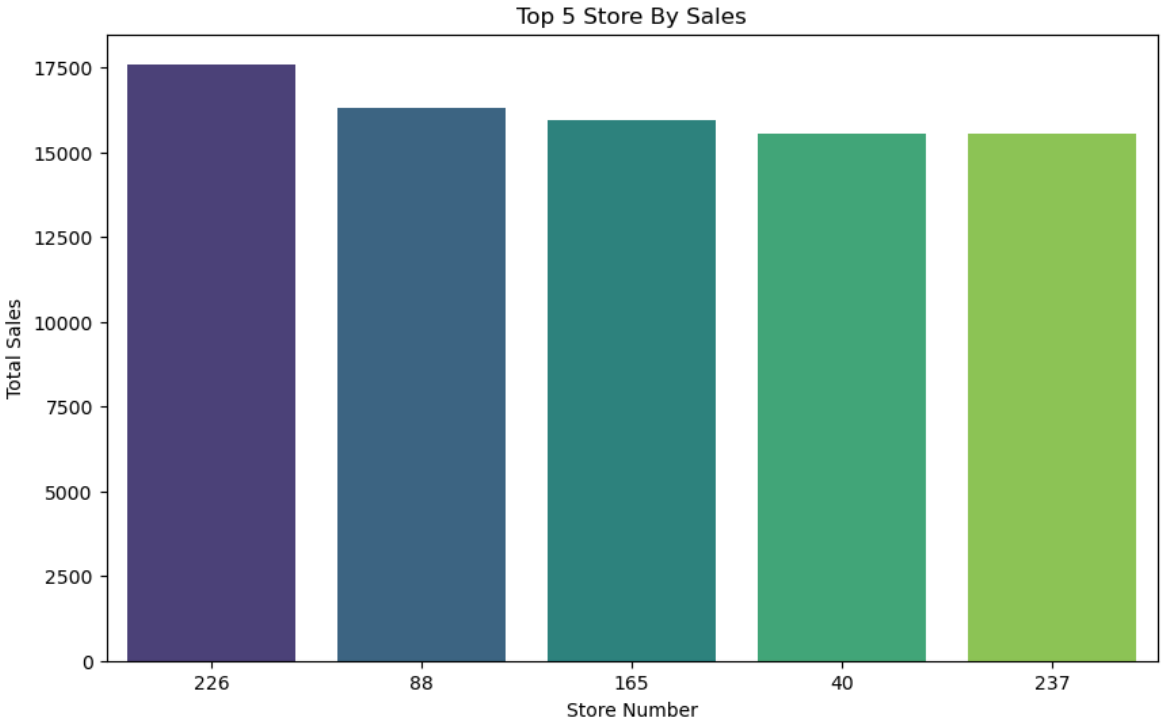
	STORE_NBR	TOT_SALES
141	226	17605.45
259	88	16333.25
73	165	15973.75
207	40	15559.50
153	237	15539.50

In [297…
```
#Plotting the bar graph
plt.figure(figsize=(10,6))

sns.barplot(x='STORE_NBR', y='TOT_SALES', data= top_5_stores_by_sales, palette='
```

```
plt.xlabel('Store Number')
plt.ylabel('Total Sales')
plt.title('Top 5 Store By Sales')

plt.show()
```



In [298…

```
#Calculate total quantity sold by store
total_qty_by_store = merged_df.groupby('STORE_NBR')['PROD_QTY'].sum().reset_inde

#Sort by total quantit and get the top 5 stores
top_5_stores_by_qty = total_qty_by_store.sort_values(by='PROD_QTY', ascending =
top_5_stores_by_qty
```

Out[298…

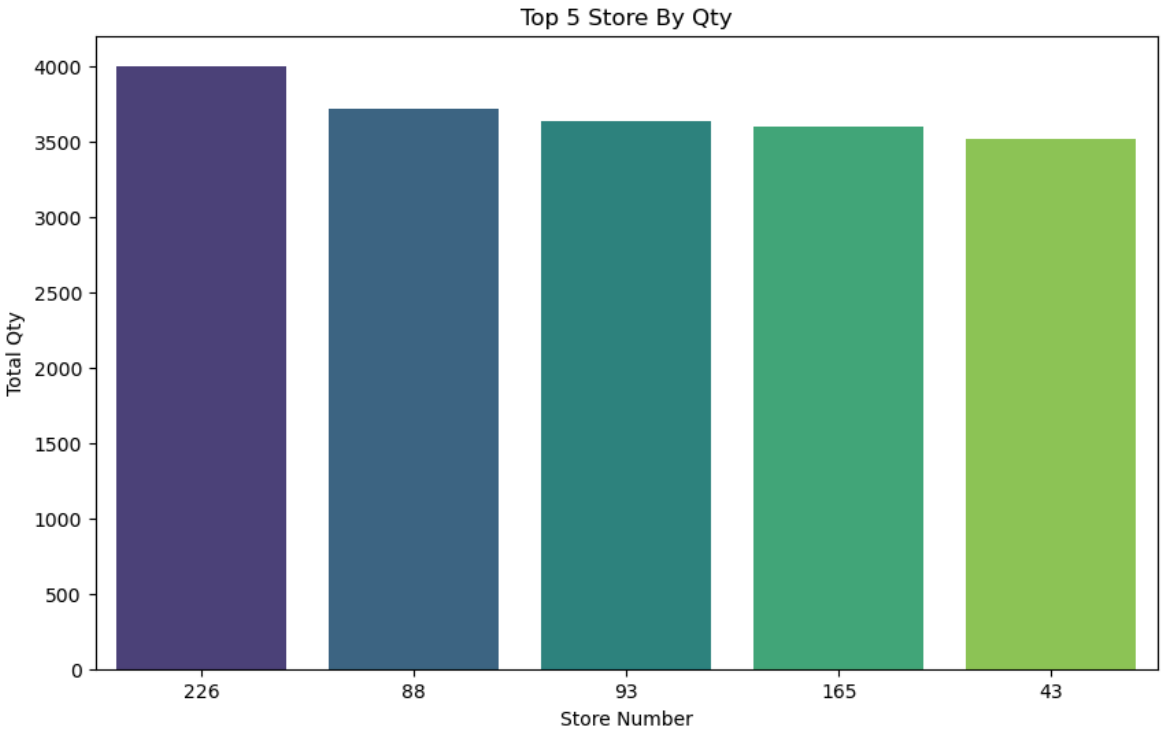
	STORE_NBR	PROD_QTY
141	226	4001
259	88	3718
265	93	3639
73	165	3602
210	43	3519

In [299…

```
#Plotting the bar graph
plt.figure(figsize=(10,6))

sns.barplot(x='STORE_NBR', y='PROD_QTY', data= top_5_stores_by_qty, palette='vir
plt.xlabel('Store Number')
plt.ylabel('Total Qty')
plt.title('Top 5 Store By Qty')

plt.show()
```



In [300…
```
#Calculate total sales per lifestage
sales_lifestage=merged_df.groupby('LIFESTAGE')['TOT_SALES'].sum().reset_index()
sales_lifestage.sort_values(by='TOT_SALES', ascending=False)
```

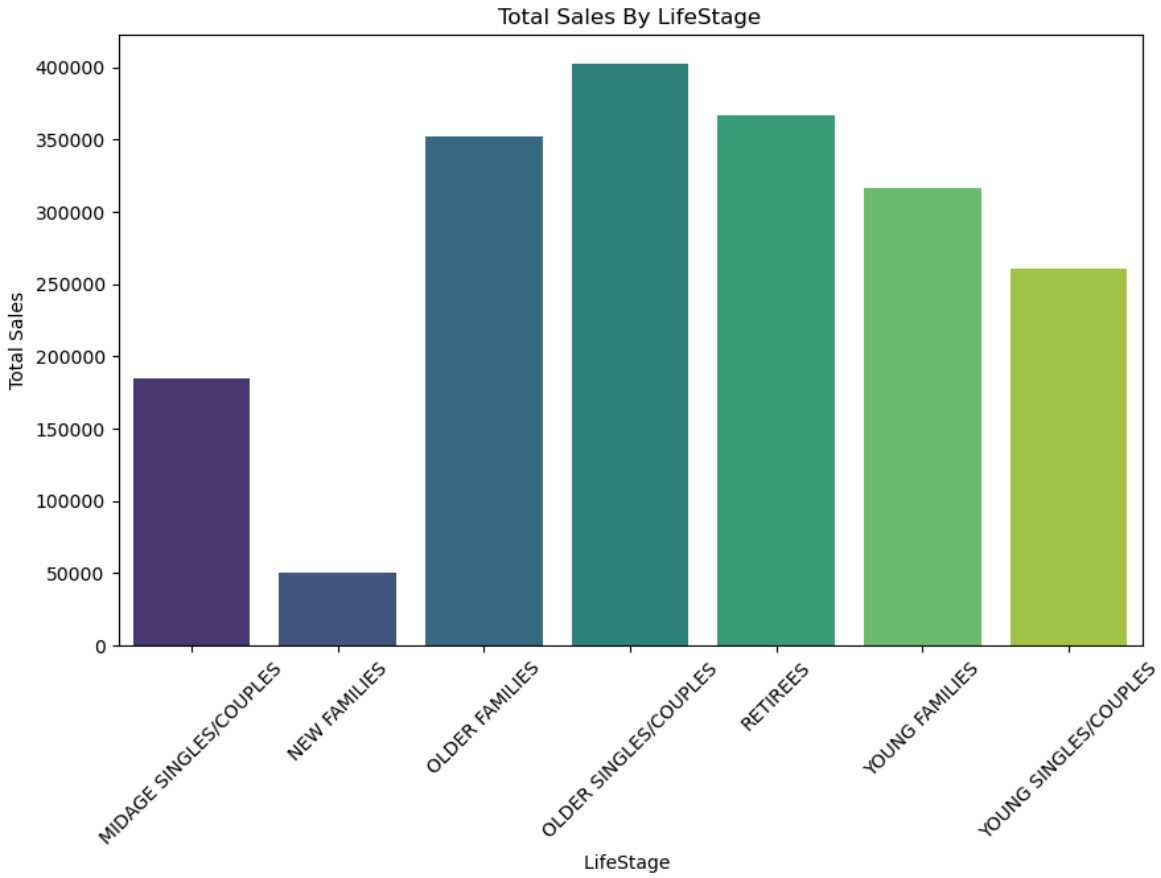
Out[300…

	LIFESTAGE	TOT_SALES
3	OLDER SINGLES/COUPLES	402426.75
4	RETIREES	366470.90
2	OLDER FAMILIES	352467.20
5	YOUNG FAMILIES	316160.10
6	YOUNG SINGLES/COUPLES	260405.30
0	MIDAGE SINGLES/COUPLES	184751.30
1	NEW FAMILIES	50433.45

In [301…
```
#Plotting the bar graph
plt.figure(figsize=(10,6))

sns.barplot(x='LIFESTAGE', y='TOT_SALES', data= sales_lifestage, palette='viridi
plt.xlabel('LifeStage ')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.title('Total Sales By LifeStage')

plt.show()
```



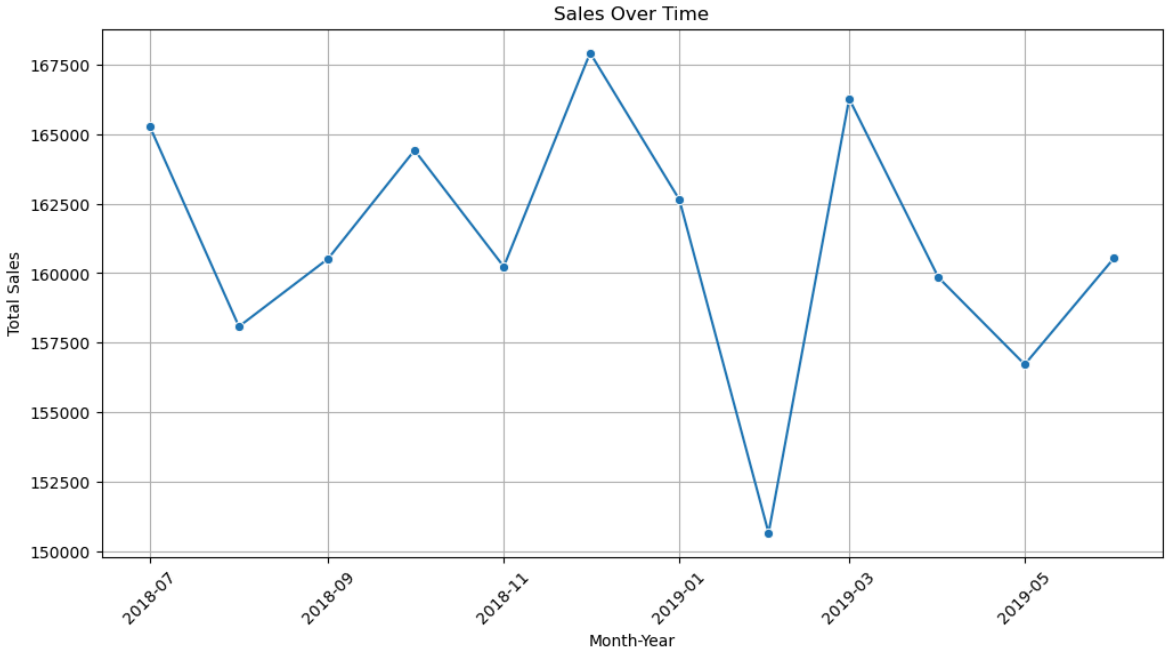
In [302…

```
#Create 'MONTH_YEAR' column
merged_df['MONTH_YEAR'] = merged_df['DATE'].dt.to_period('M').dt.to_timestamp()

#Calculate total Sales over Time
sales_over_time= merged_df.groupby('MONTH_YEAR')['TOT_SALES'].sum().reset_index(

# Plotting the line chart
plt.figure(figsize = (12,6))
sns.lineplot(x='MONTH_YEAR', y='TOT_SALES', data= sales_over_time, marker='o')
plt.xlabel('Month-Year')
plt.ylabel('Total Sales')
plt.title('Sales Over Time')
plt.grid(True)
plt.xticks(rotation=45)

#Show the plot
plt.show()
```



1.Key Measures

- **Total Sales**: $1,933,115
- **Total Quantity Sold**: 504,724 units
- **Total Unique Customers**: 72,637

--

2. Top Products by Sales

The five products with the highest sales revenue are:

Product Name	Total Sales ($)
Dorito Corn Chip Supreme 380g	39,052
Smiths Crinkle Chip Original Big Bag 380g	36,367.6
Smiths Crinkle Chip Salt & Vinegar 330g	34,804.2
Kettle Mozzarella Basil & Pesto 175g	34,457.4
Smiths Crinkle Original 330g	34,302.6

---

3. Customer Count By Spending Category

Customer count per spending category

Category	Customer Count
Nainstream	29,245
Budget	24,470

Category	Customer Count
Premium	18,922

4. Top Stores by Sales and Quantity

Store Number	Total Sales ($
Store 226	17,605.45
Store 88	16,333.25
Store 165	15,973.75
Store 48	15,559.50
Store 237	15,539.50

Quantity Leaders

Store Number	Total Sales ($
Store 226	4,001
Store 88	3.718
Store 165	3,639
Store 48	3,602
Store 237	3,519

5. Monthly Sales Trends

We have noticed a pick of sales on December 2018 at 167,913.4, likely because of the holidays season. and also we have a significant drop the occured on February 2019 due the End of Holiday season.

6. Sales Breakdown by Life Stage

presentation according to the lifestage to following stage bearkdown:

Life Stage	Total Sales ($
Order singles/Couples	402426.75
Retirees	366470.90
Older Families	352467.20
Young Families	316160.10
Young Singles/Couples	260405.30
Mid Age Singles/Couples	184751.30

Life Stage	Total Sales ($
New Families	50433.45

Key Insight: Order singles/couples and retirees are the principals contributord to sales.In conclusion, they are the key customer segments.

7. Recommendations

Focus on Key Customer Segments

Order Singles/Couples & Retirees: - These segments contribute the most to sales. Implement targeted marketing campaigns offering promotions or discounts tailored to their preferences.

For example: - Bundle deals for single households. - Discounts on frequently purchased products like snacks for retirees.

Young Families: This segment has significant sales potential but ranks fourth. Offer family-oriented promotions such as "Buy More, Save More" deals on family-sized products.

Expand Availability of Top-Selling Products

```
      - Ensure top products like Dorito Corn Chip Supreme 380g
and Smiths Crinkle Chip Original Big Bag 380g are consistently
in stock across all stores.
      - Promote these items with prominent in-store placement
and online advertisements.
```

Optimize Store Performance

Top-Performing Stores (e.g., Store 226): Study their operational strategies (e.g., layout, customer service, local demand patterns) and replicate successful practices in other stores.

Low-Performing Stores: Identify underperforming stores and address challenges such as poor product variety, low inventory, or ineffective marketing.

Seasonal Sales Strategy

Peak in December: Capitalize on the holiday season by running aggressive marketing campaigns and stocking high-demand products.

Drop in February: Introduce post-holiday promotions to maintain customer interest and sales momentum. For example: - Loyalty programs to encourage repeat purchases. - Discounts on products with slower turnover during this period.

Leverage Spending Categories

Mainstream Customers: As the largest group, target them with consistent value offerings and mid-range promotions.

Budget Customers: Offer affordable bundles and loyalty rewards to encourage repeat visits and purchases.

Premium Customers: Focus on exclusive products and premium experiences (e.g., limited editions, gourmet snacks).

Improve Product Assortment for Life Stages

Young Singles/Couples & Mid-Age Singles/Couples: Develop snack packs, meal kits, or convenience-oriented products appealing to these groups.

New Families: Introduce promotions for essential baby and child-related items or groceries tailored for young families.

Enhance Monthly Sales Trend Monitoring

Use insights from monthly trends to: - Anticipate and prepare for demand spikes. - Implement targeted strategies during expected slow months to maintain consistent sales.

By implementing these strategies, the business can better align with customer needs, enhance store efficiency, and drive higher revenue.