import numpy as np

import pandas as pd

df=pd.read\_csv(r'/kaggle/input/walmart-sales/Walmart\_Sales.csv')

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 6435 entries, 0 to 6434

Data columns (total 8 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Store 6435 non-null int64

1 Date 6435 non-null object

2 Weekly\_Sales 6435 non-null float64

3 Holiday\_Flag 6435 non-null int64

4 Temperature 6435 non-null float64

5 Fuel\_Price 6435 non-null float64

6 CPI 6435 non-null float64

7 Unemployment 6435 non-null float64

dtypes: float64(5), int64(2), object(1)

memory usage: 402.3+ KB

In [2]:

*#Below are some worth asking questions that we are going to solve and these questions will give us*

*#oppotunities to test our analysis skills and business understanding so lets dive into it .*

\*\*🔹 High-Impact Sales Insights

How do weekly sales vary across different stores? (Identifies top-performing & struggling stores for better strategy)

Are certain months more profitable for Walmart? (Helps in inventory and staffing decisions for peak sales periods)

Which store has the highest average weekly sales? Why? (Finds best-performing stores & reasons behind their success)

Which store shows the most sales fluctuation over time? (Detects unstable stores that might need better demand planning)

What is the best time to offer discounts to maximize sales? (Optimizes promotional timing to boost revenue)

🔹 Holiday & Seasonal Impact Do holiday weeks significantly impact weekly sales? (Shows if holiday promotions work & when to invest more in ads/stock)

Which holidays bring the highest sales boost? (Finds the most profitable holiday periods to focus marketing efforts on)

🔹 Economic & External Factors

How does fuel price fluctuation impact sales? (Detects if higher fuel costs reduce customer spending at Walmart)

Does the unemployment rate affect Walmart's sales? (Finds how economic downturns impact consumer spending at Walmart)

Is there a correlation between temperature and sales? (Checks if extreme weather affects shopping behavior, useful for regional strategies) \*\*

In [3]:

df

Out[3]:

|  | Store | Date | Weekly\_Sales | Holiday\_Flag | Temperature | Fuel\_Price | CPI | Unemployment |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 05-02-2010 | 1643690.90 | 0 | 42.31 | 2.572 | 211.096358 | 8.106 |
| 1 | 1 | 12-02-2010 | 1641957.44 | 1 | 38.51 | 2.548 | 211.242170 | 8.106 |
| 2 | 1 | 19-02-2010 | 1611968.17 | 0 | 39.93 | 2.514 | 211.289143 | 8.106 |
| 3 | 1 | 26-02-2010 | 1409727.59 | 0 | 46.63 | 2.561 | 211.319643 | 8.106 |
| 4 | 1 | 05-03-2010 | 1554806.68 | 0 | 46.50 | 2.625 | 211.350143 | 8.106 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 6430 | 45 | 28-09-2012 | 713173.95 | 0 | 64.88 | 3.997 | 192.013558 | 8.684 |
| 6431 | 45 | 05-10-2012 | 733455.07 | 0 | 64.89 | 3.985 | 192.170412 | 8.667 |
| 6432 | 45 | 12-10-2012 | 734464.36 | 0 | 54.47 | 4.000 | 192.327265 | 8.667 |
| 6433 | 45 | 19-10-2012 | 718125.53 | 0 | 56.47 | 3.969 | 192.330854 | 8.667 |
| 6434 | 45 | 26-10-2012 | 760281.43 | 0 | 58.85 | 3.882 | 192.308899 | 8.667 |

6435 rows × 8 columns

In [4]:

*#High impact sales insights*

*#(1) How do weekly sales vary across different stores?*

*#(Identifies top-performing & struggling stores for better strategy)*

df['Date']=pd.to\_datetime(df['Date'],format='mixed')

print(df['Date'].info())

*#converted the type of date column to date type*

<class 'pandas.core.series.Series'>

RangeIndex: 6435 entries, 0 to 6434

Series name: Date

Non-Null Count Dtype

-------------- -----

6435 non-null datetime64[ns]

dtypes: datetime64[ns](1)

memory usage: 50.4 KB

None

In [5]:

*#Now we are going to see how weekly sales vary across different stores*

WeeklySales=df.groupby('Store')['Weekly\_Sales'].mean().reset\_index()

WeeklySales

import matplotlib.pyplot as plt

import seaborn as sns

plt.figure(figsize=(14,8))

sns.barplot(x=df['Store'],y=df['Weekly\_Sales'],data=WeeklySales,palette='pastel')

Out[5]:

<Axes: xlabel='Store', ylabel='Weekly\_Sales'>

In [6]:

first\_q = df['Weekly\_Sales'].quantile(0.25)

*# Correctly combining conditions with parentheses*

filteredSales1 = df[(df['Weekly\_Sales'] > 0) & (df['Weekly\_Sales'] < first\_q)]

print(filteredSales1)

*#These stores are poor performing and need critical attention*

plt.figure(figsize=(14,8))

sns.barplot(x=df['Store'],y=filteredSales1['Weekly\_Sales'],data=filteredSales1)

Store Date Weekly\_Sales Holiday\_Flag Temperature Fuel\_Price \

286 3 2010-05-02 461622.22 0 45.71 2.572

287 3 2010-12-02 420728.96 1 47.93 2.548

288 3 2010-02-19 421642.19 0 47.07 2.514

289 3 2010-02-26 407204.86 0 52.05 2.561

290 3 2010-05-03 415202.04 0 53.04 2.625

... ... ... ... ... ... ...

6287 44 2012-09-28 355307.94 0 64.80 3.821

6288 44 2012-05-10 337390.44 0 61.79 3.815

6289 44 2012-12-10 337796.13 0 55.10 3.797

6290 44 2012-10-19 323766.77 0 52.06 3.781

6291 44 2012-10-26 361067.07 0 46.97 3.755

CPI Unemployment

286 214.424881 7.368

287 214.574792 7.368

288 214.619887 7.368

289 214.647513 7.368

290 214.675139 7.368

... ... ...

6287 131.043000 5.407

6288 131.075667 5.217

6289 131.108333 5.217

6290 131.149968 5.217

6291 131.193097 5.217

[1609 rows x 8 columns]

Out[6]:

<Axes: xlabel='Store', ylabel='Weekly\_Sales'>

**Hence we figured out which stores are poor performing and now the company can investigate as what is that that is doing harm to these stores.**

**Now we shall also look at the top performing stores because these stores is the topmost source of revenue for the company and they must be well looked after and any change to them can significantly affect company's performance**

In [7]:

fourth\_q = df['Weekly\_Sales'].quantile(1.00)

*# Correctly combining conditions with parentheses*

filteredSales2 = df[(df['Weekly\_Sales'] > df['Weekly\_Sales'].quantile(0.75)) & (df['Weekly\_Sales'] < df['Weekly\_Sales'].quantile(1.00))]

print(filteredSales2)

*#These stores are performing amazingly*

plt.figure(figsize=(14,8))

sns.barplot(x=df['Store'],y=filteredSales2['Weekly\_Sales'],data=filteredSales2)

Store Date Weekly\_Sales Holiday\_Flag Temperature Fuel\_Price \

0 1 2010-05-02 1643690.90 0 42.31 2.572

1 1 2010-12-02 1641957.44 1 38.51 2.548

2 1 2010-02-19 1611968.17 0 39.93 2.514

4 1 2010-05-03 1554806.68 0 46.50 2.625

5 1 2010-12-03 1439541.59 0 57.79 2.667

... ... ... ... ... ... ...

5851 41 2012-10-08 1504545.94 0 71.73 3.509

5852 41 2012-08-17 1560590.05 0 65.77 3.545

5853 41 2012-08-24 1464462.85 0 69.07 3.558

6338 45 2010-12-24 1682862.03 0 30.59 3.141

6390 45 2011-12-23 1521957.99 0 42.27 3.389

CPI Unemployment

0 211.096358 8.106

1 211.242170 8.106

2 211.289143 8.106

4 211.350143 8.106

5 211.380643 8.106

... ... ...

5851 198.079565 6.432

5852 198.100106 6.432

5853 198.098420 6.432

6338 182.544590 8.724

6390 188.929975 8.523

[1608 rows x 8 columns]

Out[7]:

<Axes: xlabel='Store', ylabel='Weekly\_Sales'>

**So this concludes our report and above is the chart showing promising stores that with some extra focus can be taken to new heights.These stores come in the top 25% in terms of revenue**

In [8]:

avg\_sales=df.groupby('Store')['Weekly\_Sales'].mean().reset\_index()

avg\_sales = avg\_sales.sort\_values(by='Store',ascending=True)

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

sns.barplot(x=avg\_sales['Store'],y=avg\_sales['Weekly\_Sales'],palette='pastel')

plt.xlabel("Store")

plt.ylabel("Average Weekly Sales")

plt.title("Average Weekly Sales per Store")

plt.xticks(rotation=90) *# Rotate store labels if necessary*

plt.show()

**Now we are going to look at most sales fluctuation suffering stores . their identification will help us to fix the system**

In [9]:

df['fluctuation']=df['Weekly\_Sales'].diff().abs()

df['fluctuation']

store\_fluc=df.groupby('Store')['fluctuation'].mean().reset\_index()

store\_fluc

Out[9]:

|  | Store | fluctuation |
| --- | --- | --- |
| 0 | 1 | 121093.910986 |
| 1 | 2 | 143184.479930 |
| 2 | 3 | 38123.890699 |
| 3 | 4 | 157483.520280 |
| 4 | 5 | 38039.297413 |
| 5 | 6 | 136529.273636 |
| 6 | 7 | 59135.330210 |
| 7 | 8 | 70969.808811 |
| 8 | 9 | 42630.748881 |
| 9 | 10 | 150824.919720 |
| 10 | 11 | 106213.267133 |
| 11 | 12 | 77213.070420 |
| 12 | 13 | 152888.407133 |
| 13 | 14 | 189706.956084 |
| 14 | 15 | 68148.654545 |
| 15 | 16 | 42803.896503 |
| 16 | 17 | 81940.567762 |
| 17 | 18 | 105506.731608 |
| 18 | 19 | 119430.931818 |
| 19 | 20 | 183567.741608 |
| 20 | 21 | 72305.823077 |
| 21 | 22 | 92470.609930 |
| 22 | 23 | 138930.793706 |
| 23 | 24 | 122021.960559 |
| 24 | 25 | 55068.833916 |
| 25 | 26 | 84895.304336 |
| 26 | 27 | 155710.222797 |
| 27 | 28 | 173173.835594 |
| 28 | 29 | 61578.470559 |
| 29 | 30 | 16902.620140 |
| 30 | 31 | 90684.107622 |
| 31 | 32 | 75455.685455 |
| 32 | 33 | 27180.665455 |
| 33 | 34 | 69139.684056 |
| 34 | 35 | 100087.610699 |
| 35 | 36 | 21641.183427 |
| 36 | 37 | 21745.228951 |
| 37 | 38 | 28679.225664 |
| 38 | 39 | 118731.305455 |
| 39 | 40 | 92808.344476 |
| 40 | 41 | 99731.746643 |
| 41 | 42 | 57958.981748 |
| 42 | 43 | 37171.775734 |
| 43 | 44 | 16093.241119 |
| 44 | 45 | 71546.726713 |

In [10]:

*#plotting this will give us a better idea of the situation*

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

sns.barplot(x='Store', y='fluctuation', data=store\_fluc, palette='viridis')

plt.xticks(rotation=90)

plt.title("Stores High Revenue Fluctuation")

plt.show()

**The above chart can further be analysed to see some very vulnerable stores , those whose fluctuation is above $ 100000 and by identifying them we can fix them and increase our revenue**

In [11]:

extreme\_fluc=store\_fluc[store\_fluc['fluctuation']>100000]

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

sns.barplot(x='Store', y='fluctuation', data=extreme\_fluc, palette='viridis')

plt.xticks(rotation=90)

plt.title("Stores with extremely High Revenue Fluctuation")

plt.show()

**What is the best time to offer discounts to maximize sales? (Optimizes promotional timing to boost revenue).We are going to answer and analyse this part**

In [12]:

df['months']=df['Date'].dt.strftime('%B')

rev\_by\_month=df.groupby('months')['Weekly\_Sales'].mean().reset\_index()

month\_order = [

"January", "February", "March", "April", "May", "June",

"July", "August", "September", "October", "November", "December"

]

rev\_by\_month['months'] = pd.Categorical(rev\_by\_month['months'], categories=month\_order, ordered=True)

rev\_by\_month = rev\_by\_month.sort\_values('months')

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

sns.barplot(x='months', y='Weekly\_Sales', data=rev\_by\_month,palette='viridis')

plt.xticks(rotation=90)

plt.title("Sales by month")

plt.show()

/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:641: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.

grouped\_vals = vals.groupby(grouper)

**This chart is showing a very clear trend that the sales are the lowest in january but then it rises in the month of february and remains constant until october.The situation again changes when November starts and then the sales continues to increase and the peak sales happen in December so we can conclude that due November and December being festival heavy months people tend to buy more but once january starts the sales drop significantly and the rest of the months are doing pretty consistently.Therefore we can give good discounts in january which will encourage people to spend**

**Just like I mentioned that giving discounts in january will increase sales same is the case with December because already sales are high and giving some discount in this month will give a big boost to the already high sales**

In [13]:

sales\_by\_unemployment = df.groupby('Weekly\_Sales')['Unemployment'].mean().reset\_index()

sales\_by\_unemployment

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

sns.lineplot(x='Unemployment',y='Weekly\_Sales',data=sales\_by\_unemployment)

/usr/local/lib/python3.10/dist-packages/seaborn/\_oldcore.py:1119: FutureWarning: use\_inf\_as\_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.

with pd.option\_context('mode.use\_inf\_as\_na', True):

/usr/local/lib/python3.10/dist-packages/seaborn/\_oldcore.py:1119: FutureWarning: use\_inf\_as\_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.

with pd.option\_context('mode.use\_inf\_as\_na', True):

Out[13]:

<Axes: xlabel='Unemployment', ylabel='Weekly\_Sales'>

**The above lineplot depicts a very interesting picture in which we can see that at large when unemployment is less people tend to buy more and hence the sales are high but as unemployment rises the sales drop clearly indicating that people save money during their period and unemployment**

In [14]:

linkcode

sales\_by\_temp=df.groupby('Weekly\_Sales')['Temperature'].mean().reset\_index()

sales\_by\_temp

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

sns.lineplot(x='Temperature',y='Weekly\_Sales',data=sales\_by\_temp)

/usr/local/lib/python3.10/dist-packages/seaborn/\_oldcore.py:1119: FutureWarning: use\_inf\_as\_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.

with pd.option\_context('mode.use\_inf\_as\_na', True):

/usr/local/lib/python3.10/dist-packages/seaborn/\_oldcore.py:1119: FutureWarning: use\_inf\_as\_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.

with pd.option\_context('mode.use\_inf\_as\_na', True):

Out[14]:

<Axes: xlabel='Temperature', ylabel='Weekly\_Sales'>

**The above chart is showing that the temperature is below 20 sales drop probabily because colder temperatures discourage people to get out from their comfort zone and therefore sales drop below 20 degree.We can clearly see that sales are pretty good when the temperature is between 20 and 60 degrees and that is the perfect weather people would go out for shopping.Just like extreme cold conditions discourage people so are hot temperatures as at temperature above 80 degrees the sales start to decline significantly so the temperauture and weather heavily affect the number of sales**

**I have put in a lot of effort into this Walmart sales analysis, tried to answer the above questions and drawing out value from this analysis. This project was an exciting opportunity to apply data analysis techniques and draw valuable insights that could have a real-world impact on business strategy.**

**If you found this analysis useful or insightful, please consider upvoting it. Your support means a lot and helps me continue creating valuable content. Thank you!**