In [76]:
```
import pandas as pd
from datetime import datetime, timedelta
```

In [77]:
```
# Load the Data
data = pd.read_csv('sales_data.csv')
data.head()
```

Out[77]:

	Date	Time	StoreID	CustomerID	OrderID	Product Name	Price
0	2024-08-17	10:56:35	1	2730	8000	Product H	15.99
1	2024-08-17	10:56:35	1	2730	8000	Product J	9.99
2	2024-08-17	10:56:35	1	2730	8000	Product G	6.49
3	2024-08-17	10:56:35	1	2730	8000	Product E	3.99
4	2024-08-17	10:56:35	1	2730	8000	Product E	3.99

In [78]:
```
# Create a DataFrame
df = pd.DataFrame(data, columns=['Date', 'Time', 'StoreID', 'CustomerID', 'OrderID'
```

In [79]:
```
df.info
```

Out[79]:
```
<bound method DataFrame.info of                Date      Time  StoreID  CustomerID
OrderID Product Name  Price
0       2024-08-17  10:56:35        1        2730     8000    Product H  15.99
1       2024-08-17  10:56:35        1        2730     8000    Product J   9.99
2       2024-08-17  10:56:35        1        2730     8000    Product G   6.49
3       2024-08-17  10:56:35        1        2730     8000    Product E   3.99
4       2024-08-17  10:56:35        1        2730     8000    Product E   3.99
...            ...       ...      ...         ...      ...          ...    ...
108796  2024-05-16  10:56:35      100        2376     7953    Product C   7.99
108797  2024-05-16  10:56:35      100        2376     7953    Product C   7.99
108798  2024-05-16  10:56:35      100        2376     7953    Product A  10.99
108799  2024-05-16  10:56:35      100        2376     7953    Product C   7.99
108800  2024-05-16  10:56:35      100        2376     7953    Product J   9.99

[108801 rows x 7 columns]>
```

In [80]:
```
df.columns
```

Out[80]:
```
Index(['Date', 'Time', 'StoreID', 'CustomerID', 'OrderID', 'Product Name',
       'Price'],
      dtype='object')
```

In [81]:
```
store_count = df ['StoreID'].nunique()
print(f"Number of Unique stores: {store_count}")
```

```
Number of Unique stores: 100
```

In [82]:
```
# Most prevelant products in baskets
prevalent_products = df['Product Name'].value_counts().head(5)
print(f"Top 5 of our Most Prevalent Products: {prevalent_products}")
```

```
Top 5 of our Most Prevalent Products: Product Name
Product A    11236
Product C    10977
Product J    10957
Product H    10885
Product E    10872
Name: count, dtype: int64
```

In [ ]:
```
# Did not have qauntity so I Grouped by StoreID, CustomerID, Date, and Time to find
grouped_data = data.groupby(['StoreID', 'CustomerID', 'Date', 'Time']).size().reset

grouped_data.head()
```

Out[ ]:

	StoreID	CustomerID	Date	Time	ItemCount
0	1	264	2024-03-15	10:56:35	4
1	1	264	2024-07-14	10:56:35	9
2	1	340	2024-03-22	10:56:35	6
3	1	340	2024-10-22	10:56:35	6
4	1	340	2024-11-23	10:56:35	7

In [84]:
```
# Define How much a Large Basket is (7)
large_basket_threshold = 7

# Filter transactions for large baskets
large_basket_transactions = grouped_data[grouped_data['ItemCount'] >= large_basket_

# All of large-basket transactions
print(f"Transactions with large baskets (7 or more items):\n {large_basket_transact
```

```
Transactions with large baskets (7 or more items):
        StoreID  CustomerID        Date      Time  ItemCount
1             1         264  2024-07-14  10:56:35          9
4             1         340  2024-11-23  10:56:35          7
10            1         669  2024-02-15  10:56:35          9
14            1         836  2024-02-04  10:56:35          8
15            1         853  2024-03-11  10:56:35         10
...         ...         ...         ...       ...        ...
19862       100        9605  2024-09-23  10:56:35          8
19863       100        9605  2024-12-14  10:56:35         10
19865       100        9681  2024-08-26  10:56:35         10
19867       100        9748  2024-07-11  10:56:35         10
19868       100        9941  2024-08-24  10:56:35         10

[7876 rows x 5 columns]
```

In [85]:
```
# Calculate the frequency of large-basket transactions
large_basket_frequency = large_basket_transactions.shape[0]
print(f"Frequency of large-basket transactions (7 or more items): {large_basket_fre
```

```
Frequency of large-basket transactions (7 or more items): 7876
```

In [86]:
```
# Sort stores showing large-basket buyers in descending order
large_basket_stores_sorted = large_basket_stores.sort_values(by='LargeBasketCount',

# Display the top 10 stores with large-basket buyers
top_10_large_basket_stores = large_basket_stores_sorted.head(10)
print(f"Top 10 stores with large-basket buyers and their counts:\n{top_10_large_bas
```

```
Top 10 stores with large-basket buyers and their counts:
    StoreID  LargeBasketCount
34       35               138
32       33               133
65       66               126
6         7               124
45       46               123
75       76               119
48       49               118
42       43               118
8         9               118
33       34               115
```

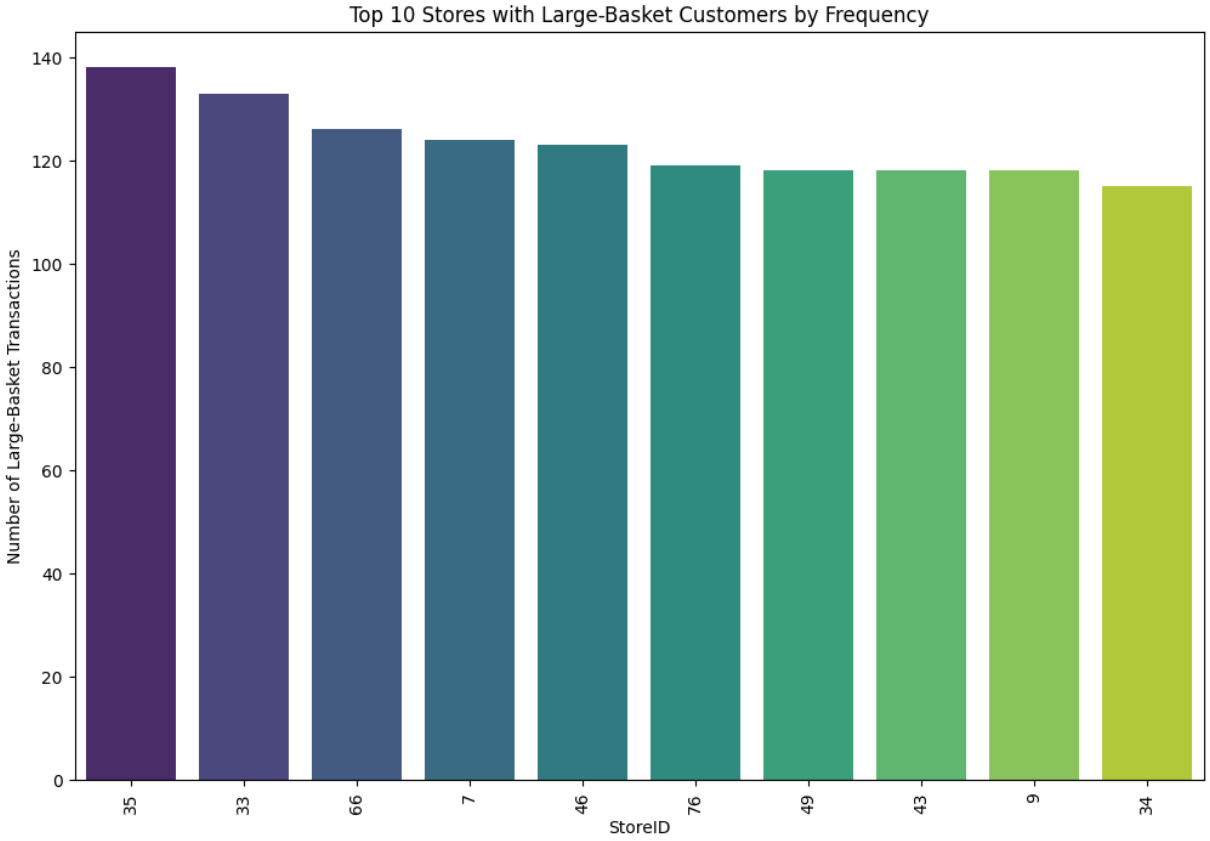
In [87]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [96]:
```
# Visualization of Top 10 Stores By Large-Basket Frequency
plt.figure(figsize=(12, 8))
sns.barplot(x='StoreID', y='LargeBasketCount', data=top_10_large_basket_stores, pal
plt.xlabel('StoreID')
plt.ylabel('Number of Large-Basket Transactions')
plt.title('Top 10 Stores with Large-Basket Customers by Frequency')
plt.xticks(rotation=90)
plt.show()
```

```
C:\Users\eriks\AppData\Local\Temp\ipykernel_25224\1057542087.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14
.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x='StoreID', y='LargeBasketCount', data=top_10_large_basket_stores, pa
lette='viridis', order=top_10_large_basket_stores['StoreID'])
```



In [ ]:
```
# Filter original data for large-basket transactions
large_basket_data = data.merge(large_basket_transactions, on=['StoreID', 'CustomerI

# Group by Product Name and count the number of occurrences
large_basket_product_frequency = large_basket_data['Product Name'].value_counts().r
large_basket_product_frequency.columns = ['Product Name', 'Frequency']

# Number of Products shown
top_n = 10

# Get the top-n products
top_n_large_basket_products = large_basket_product_frequency.head(top_n)

# Display the top-n products for large-basket transactions
print(f"Top {top_n} products for large-basket customers:\n{top_n_large_basket_produ
```

```
Top 10 products for large-basket customers:
  Product Name  Frequency
0    Product A       6984
1    Product E       6795
2    Product J       6748
3    Product F       6721
4    Product C       6691
5    Product D       6690
6    Product G       6658
7    Product H       6614
8    Product B       6567
9    Product I       6515
```

In [124…
```
# Created categories for each of my products
```

```
product_to_category = {
    'Product A': 'Category1',
    'Product B': 'Category2',
    'Product C': 'Category3',
    'Product D': 'Category4',
    'Product E': 'Category5',
    'Product F': 'Category6',
    'Product G': 'Category7',
    'Product H': 'Category8',
    'Product I': 'Category9',
    'Product J': 'Category10',

}

# Add a 'Category' column to your data
data['Category'] = data['Product Name'].map(product_to_category)
print(data.head())
```

```
         Date      Time  StoreID  CustomerID  OrderID Product Name  Price  \
0  2024-08-17  10:56:35        1        2730     8000    Product H  15.99
1  2024-08-17  10:56:35        1        2730     8000    Product J   9.99
2  2024-08-17  10:56:35        1        2730     8000    Product G   6.49
3  2024-08-17  10:56:35        1        2730     8000    Product E   3.99
4  2024-08-17  10:56:35        1        2730     8000    Product E   3.99

     Category
0   Category8
1  Category10
2   Category7
3   Category5
4   Category5
```

In [ ]:
```
# Group by Category and calculate the average number of items per category
category_average = large_basket_data.groupby('Category').size().reset_index(name='A

# Display the categorical makeup of their baskets
print(f"Categorical makeup of their baskets on average:\n{category_average}")

# Create a bar chart for the categorical makeup of their baskets
plt.figure(figsize=(12, 8))
sns.barplot(x='Category', y='AverageItemCount', data=category_average, palette='vir
plt.xlabel('Category')
plt.ylabel('Average Number of Items')
plt.title('Categorical Makeup of Baskets on Average')
plt.xticks(rotation=90)
plt.show()
```

```
Categorical makeup of their baskets on average:
     Category  AverageItemCount
0   Category1             11236
1  Category10             10957
2   Category2             10803
3   Category3             10977
4   Category4             10856
5   Category5             10872
6   Category6             10729
7   Category7             10736
8   Category8             10885
9   Category9             10750
C:\Users\eriks\AppData\Local\Temp\ipykernel_25224\486443953.py:9: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14
.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x='Category', y='AverageItemCount', data=category_average, palette='vi
ridis')
```