January 2025

01

SIPSMART DRINK RETAIL STORE

BUSINESS INTELLIGENCE REPORT

Prepared by:
Maryam Khedhiri
Samar Kilani
Yessin Jribi

BA/IT

Table of Contents

COMPANY INTRODUCTION

SipSmart Beverages is a mid-sized distributor of premium drinks, specializing in non-alcoholic beverages, energy drinks, and craft sodas across the country.

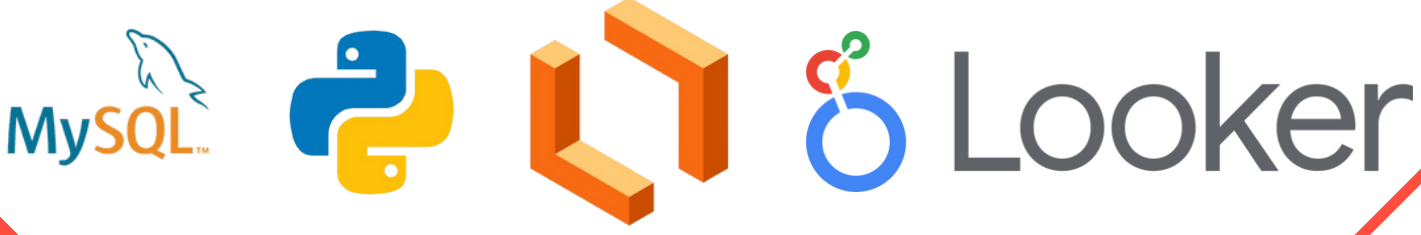
The company operates in the highly competitive drinks industry, managing a diverse portfolio of products across multiple brands, stores, and locations. With a focus on delivering quality products and enhancing customer satisfaction, the company aims to optimize its sales performance, inventory management, and supplier relationships.

In this context, the need for a robust business intelligence solution was identified to address key challenges, such as understanding sales trends, evaluating supplier contributions, identifying top-performing products and stores, and optimizing stock turnover rates. Leveraging sales and inventory data, the objective is to gain actionable insights that can drive strategic decision-making and operational efficiency.

Goals and Deliverables:

1. Understand Sales Performance: Identify the top-performing stores, brands, and cities in terms of sales to allocate resources effectively and replicate success strategies.
2. Evaluate Vendor Contributions: Assess vendor performance to strengthen relationships with high-performing suppliers and diversify dependencies.
3. Optimize Inventory Management: Analyze stock turnover rates to improve efficiency and ensure consistent product availability across stores.
4. Monitor Growth Trends: Visualize month-to-month sales growth to uncover seasonal patterns and key growth drivers.
5. Actionable Recommendations: Provide data-driven insights and decisions to enhance marketing strategies, operational planning, and overall business performance.

TOOLS AND SOFTWARES USED:

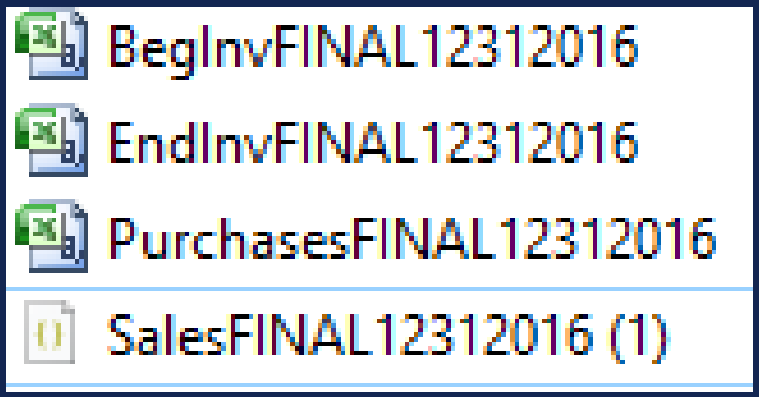


DATA GATHERING

To analyze the activities of SipSmart Beverages, we will examine data related to inventory and sales operations covering the period from December 2015 to February 2016.This data was taken from kaggle.com.

The collected datasets are the following :

- **BegInvFINAL12312016.csv**
- **EndInvFINAL12312016.csv**
- **PurchasesFINAL12312016.csv**
- **SalesFINAL12312016.json**



Some Data Observations :

Beginning Inventory Database

```
1  InventoryId,"Store","City","Brand","Description","Size","onHand","Price","startDate"
2  1_HARDERSFIELD_58,1,"HARDERSFIELD",58,"Gekkeikan Black & Gold Sake","750mL",8,12.99,2016-01-01
3  1_HARDERSFIELD_60,1,"HARDERSFIELD",60,"Canadian Club 1858 VAP","750mL",7,10.99,2016-01-01
4  1_HARDERSFIELD_62,1,"HARDERSFIELD",62,"Herradura Silver Tequila","750mL",6,36.99,2016-01-01
```

Ending Inventory Database

```
1  InventoryId,"Store","City","Brand","Description","Size","onHand","Price","endDate"
2  1_HARDERSFIELD_58,1,"HARDERSFIELD",58,"Gekkeikan Black & Gold Sake","750mL",11,12.99,2016-12-31
3  1_HARDERSFIELD_62,1,"HARDERSFIELD",62,"Herradura Silver Tequila","750mL",7,36.99,2016-12-31
4  1_HARDERSFIELD_63,1,"HARDERSFIELD",63,"Herradura Reposado Tequila","750mL",7,38.99,2016-12-31
```

Purchases Database

```
1  InventoryId,"Store","Brand","Description","Size","VendorNumber","VendorName","PONumber","PODate","ReceivingDate","InvoiceDate","PayDate","PurchasePrice","Quantity","Dollars","Classification"
2  69_MOUNTMEND_8412,69,8412,"Tequila Ocho Plata Fresno","750mL",105,"ALTAMAR BRANDS LLC       ",8124,"2015-12-21","2016-01-02","2016-01-04","2016-02-16",35.71,6,214.26,1
3  30_CULCHETH_5255,30,5255,"TGI Fridays Ultimte Mudslide","1.75L",4466,"AMERICAN VINTAGE BEVERAGE ",8137,"2015-12-22","2016-01-01","2016-01-07","2016-02-21",9.35,4,37.4,1
4  34_PITMERDEN_5215,34,5215,"TGI Fridays Long Island Iced","1.75L",4466,"AMERICAN VINTAGE BEVERAGE ",8137,"2015-12-22","2016-01-02","2016-01-07","2016-02-21",9.41,5,47.05,1
5  1_HARDERSFIELD_5255,1,5255,"TGI Fridays Ultimte Mudslide","1.75L",4466,"AMERICAN VINTAGE BEVERAGE ",8137,"2015-12-22","2016-01-01","2016-01-07","2016-02-21",9.35,6,56.1,1
6  76_DONCASTER_2034,76,2034,"Glendalough Double Barrel","750mL",388,"ATLANTIC IMPORTING COMPANY ",8169,"2015-12-24","2016-01-02","2016-01-09","2016-02-16",21.32,5,106.6,1
7  S_SUTTON_3348,S,3348,"Bombay Sapphire Gin","1.75L",480,"BACARDI USA INC         ",8106,"2015-12-20","2016-01-02","2016-01-12","2016-02-05",22.38,6,134.28,1
```

Sales Database

```
1  {"InventoryId":"1_HARDERSFIELD_1004","Store":1,"Brand":1004,"Description":"Jim Beam w\/2 Rocks Glasses","Size":"75
2  {"InventoryId":"1_HARDERSFIELD_1004","Store":1,"Brand":1004,"Description":"Jim Beam w\/2 Rocks Glasses","Size":"75
3  {"InventoryId":"1_HARDERSFIELD_1004","Store":1,"Brand":1004,"Description":"Jim Beam w\/2 Rocks Glasses","Size":"75
4  {"InventoryId":"1_HARDERSFIELD_1004","Store":1,"Brand":1004,"Description":"Jim Beam w\/2 Rocks Glasses","Size":"75
5  {"InventoryId":"1_HARDERSFIELD_1005","Store":1,"Brand":1005,"Description":"Maker's Mark Combo Pack","Size":"375mL
6  {"InventoryId":"S1_HARDERSFIELD_10058","Store":1,"Brand":10058,"Description":"F Coppola Dmd Ivry Cab Svgn","Size":
7  {"InventoryId":"1_HARDERSFIELD_10058","Store":1,"Brand":10058,"Description":"F Coppola Dmd Ivry Cab Svgn","Size":"
8  {"InventoryId":"1_HARDERSFIELD_10058","Store":1,"Brand":10058,"Description":"F Coppola Dmd Ivry Cab Svgn","Size":"
9  {"InventoryId":"1_HARDERSFIELD_1006","Store":1,"Brand":1006,"Description":"Jim Beam Candy Cane 4\/50mLs","Size":"5
10 {"InventoryId":"1_HARDERSFIELD_10062","Store":1,"Brand":10062,"Description":"Terroirs du Rhone","Size":"750mL","Sa
```

```
1  sses","Size":"750mL","SalesQuantity":1,"SalesDollars":16.49,"SalesPrice":16.49,"SalesDate":"1\/1\/2016","Volume":7
2  sses","Size":"750mL","SalesQuantity":2,"SalesDollars":32.98,"SalesPrice":16.49,"SalesDate":"1\/2\/2016","Volume":7
3  sses","Size":"750mL","SalesQuantity":1,"SalesDollars":16.49,"SalesPrice":16.49,"SalesDate":"1\/3\/2016","Volume":7
4  sses","Size":"750mL","SalesQuantity":1,"SalesDollars":14.49,"SalesPrice":14.49,"SalesDate":"1\/8\/2016","Volume":7
5  ","Size":"375mL 2 Pk","SalesQuantity":2,"SalesDollars":69.98,"SalesPrice":34.99,"SalesDate":"1\/9\/2016","Volume":
6  ab Svgn","Size":"750mL","SalesQuantity":2,"SalesDollars":29.98,"SalesPrice":14.99,"SalesDate":"1\/23\/2016","Volum
7  b Svgn","Size":"750mL","SalesQuantity":1,"SalesDollars":14.99,"SalesPrice":14.99,"SalesDate":"1\/25\/2016","Volume
8  b Svgn","Size":"750mL","SalesQuantity":1,"SalesDollars":14.99,"SalesPrice":14.99,"SalesDate":"1\/29\/2016","Volume
9  50mLs","Size":"50mL 4 Pk","SalesQuantity":1,"SalesDollars":3.99,"SalesPrice":3.99,"SalesDate":"1\/29\/2016","Volum
10 ize":"750mL","SalesQuantity":1,"SalesDollars":8.99,"SalesPrice":8.99,"SalesDate":"1\/1\/2016","Volume":750,"Classi
```

```
6","Volume":750,"Classification":1,"ExciseTax":0.79,"VendorNo":12546,"VendorName":"JIM BEAM BRANDS COMPANY     "}
6","Volume":750,"Classification":1,"ExciseTax":1.57,"VendorNo":12546,"VendorName":"JIM BEAM BRANDS COMPANY     "}
6","Volume":750,"Classification":1,"ExciseTax":0.79,"VendorNo":12546,"VendorName":"JIM BEAM BRANDS COMPANY     "}
6","Volume":750,"Classification":1,"ExciseTax":0.79,"VendorNo":12546,"VendorName":"JIM BEAM BRANDS COMPANY     "}
16","Volume":375,"Classification":1,"ExciseTax":0.79,"VendorNo":12546,"VendorName":"JIM BEAM BRANDS COMPANY     "}
/2016","Volume":750,"Classification":2,"ExciseTax":0.22,"VendorNo":2000,"VendorName":"SOUTHERN WINE & SPIRITS NE "
2016","Volume":750,"Classification":2,"ExciseTax":0.11,"VendorNo":2000,"VendorName":"SOUTHERN WINE & SPIRITS NE "}
2016","Volume":750,"Classification":2,"ExciseTax":0.11,"VendorNo":2000,"VendorName":"SOUTHERN WINE & SPIRITS NE "}
/2016","Volume":50,"Classification":1,"ExciseTax":0.05,"VendorNo":12546,"VendorName":"JIM BEAM BRANDS COMPANY     "
":750,"Classification":2,"ExciseTax":0.11,"VendorNo":10754,"VendorName":"PERFECTA WINES              "}
```

DATA CLEANING

In data cleaning, we focused on converting dates from the *'MM-DD-YYYY' format to 'DD-MM-YYYY'* and *standardizing the size*. Since we are focusing on drinks, we standardized the size for all products to 'L' instead of 'mL' or 'Liter.We also did *remove duplicates* from the data.

Sales Data Cleaning :

```
[ ]  import pandas as pd
     import numpy as np
     import re
     import json
     from google.colab import files

     uploaded = files.upload()
     salesfinal = list(uploaded.keys())[0]
```

```
data = []
with open(salesfinal, 'r') as f:
    for line in f:
        try:
            data.append(json.loads(line))  # Load each line as a separate JSON object
        except json.JSONDecodeError as e:
            print(f"Warning: Skipping invalid JSON line: {line.strip()} - Error: {e}")

#  Convert JSON data to a pandas DataFrame
df = pd.DataFrame(data)
# Replace Missing Values with Column Averages (Numeric Columns)
numeric_cols = ["SalesQuantity", "SalesDollars", "SalesPrice", "Volume"]
for col in numeric_cols:
    if col in df.columns:
        df[col] = pd.to_numeric(df[col], errors='coerce')  # Ensure numeric type
        df[col].fillna(df[col].mean(), inplace=True)  # Fill missing values with the column average

# Remove Duplicate Rows
df.drop_duplicates(inplace=True)

#  Standardize Date Format for 'SalesDate'
if 'SalesDate' in df.columns:
    df['SalesDate'] = pd.to_datetime(df['SalesDate'], errors='coerce')  # Convert to datetime format
    df['SalesDate'] = df['SalesDate'].dt.strftime('%d-%m-%Y')  # Convert format to day-month-year
```

DATA CLEANING

```python
#  Standardize Size Format (Convert mL to L, Liter to 1L)
def standardize_size(size):
    """ Convert mL to L and Liter to 1L """
    if pd.isna(size):
        return np.nan  # Keep NaN as NaN
    size = str(size).lower().strip()  # Convert to lowercase and remove spaces
    if "ml" in size:
        num = re.findall(r'\d+', size)  # Extract numbers
        if num:
            return f"{int(num[0]) / 1000}L"    # Convert mL to L
    elif "liter" in size or "litre" in size:
        return "1L"  # Standardize all "Liter" mentions to "1L"
    return size  # Return unchanged if no match

if 'Size' in df.columns:
    df["Size"] = df["Size"].apply(standardize_size)


# Display Sample of Cleaned Data
print(df.head())
```

Cleaned Sales Data

	InventoryId	Store	Brand	Description	Size	SalesQuantity	SalesDollars	SalesPrice	SalesDate	Volume	Classification	ExciseTax	VendorNo	VendorName
0	1_HARDERSFIELD_1004	1	1004	Jim Beam w/2 Rocks Glasses	0.75L	1	16.49	16.49	01-01-2016	750	1	0.79	12546	JIM BEAM BRANDS COMPANY
1	1_HARDERSFIELD_1004	1	1004	Jim Beam w/2 Rocks Glasses	0.75L	2	32.98	16.49	02-01-2016	750	1	1.57	12546	JIM BEAM BRANDS COMPANY
2	1_HARDERSFIELD_1004	1	1004	Jim Beam w/2 Rocks Glasses	0.75L	1	16.49	16.49	03-01-2016	750	1	0.79	12546	JIM BEAM BRANDS COMPANY
3	1_HARDERSFIELD_1004	1	1004	Jim Beam w/2 Rocks Glasses	0.75L	1	14.49	14.49	08-01-2016	750	1	0.79	12546	JIM BEAM BRANDS COMPANY
4	1_HARDERSFIELD_1005	1	1005	Maker's Mark Combo Pack	0.375L	2	69.98	34.99	09-01-2016	375	1	0.79	12546	JIM BEAM BRANDS COMPANY

DATA CLEANING

Purchase Data Cleaning :

```python
import pandas as pd
import numpy as np
import re
from google.colab import files

# Upload purchases data
uploaded = files.upload()

# Load purchases data
purchases = pd.read_csv(list(uploaded.keys())[0])

# Replace Missing Values with Column Averages (Numeric Columns)
numeric_cols = ["Brand", "Size", "VendorNumber", "PONumber"]  # Numeric columns specified
for col in numeric_cols:
    if col in purchases.columns:
        purchases[col] = pd.to_numeric(purchases[col], errors='coerce')  # Ensure numeric type
        purchases[col].fillna(purchases[col].mean(), inplace=True)  # Fill missing values with the column average

# Remove Duplicate Rows
purchases.drop_duplicates(inplace=True)

# Standardize Date Formats for PODate and ReceivingDate
for date_col in ["PODate", "ReceivingDate"]:
    if date_col in purchases.columns:
        purchases[date_col] = pd.to_datetime(purchases[date_col], errors='coerce')  # Convert to datetime format
        purchases[date_col] = purchases[date_col].dt.strftime('%d-%m-%Y')  # Convert format to day-month-year
```

```python
# Standardize Size Format
def standardize_size(size):
    """ Convert mL to L and Liter to 1L """
    if pd.isna(size):
        return np.nan  # Keep NaN as NaN
    size = str(size).lower().strip()  # Convert to lowercase and remove spaces
    if "ml" in size:
        num = re.findall(r'\d+', size)  # Extract numbers
        if num:
            return f"{int(num[0]) / 1000}L"  # Convert mL to L
    elif "liter" in size or "litre" in size:
        return "1L"  # Standardize all "Liter" mentions to "1L"
    return size  # Return unchanged if no match

if "Size" in purchases.columns:
    purchases["Size"] = purchases["Size"].apply(standardize_size)

# Display Sample of Cleaned Data
print(purchases.head())
```

DATA CLEANING

Cleaned Purchase Data

	InventoryId	Store	Brand	Description	Size	\
0	69_MOUNTMEND_8412	69	8412	Tequila Ocho Plata Fresno	25.0	
1	30_CULCHETH_5255	30	5255	TGI Fridays Ultimte Mudslide	25.0	
2	34_PITMERDEN_5215	34	5215	TGI Fridays Long Island Iced	25.0	
3	1_HARDERSFIELD_5255	1	5255	TGI Fridays Ultimte Mudslide	25.0	
4	76_DONCASTER_2034	76	2034	Glendalough Double Barrel	25.0	

	VendorNumber	VendorName	PONumber	PODate	\
0	105	ALTAMAR BRANDS LLC	8124	21-12-2015	
1	4466	AMERICAN VINTAGE BEVERAGE	8137	22-12-2015	
2	4466	AMERICAN VINTAGE BEVERAGE	8137	22-12-2015	
3	4466	AMERICAN VINTAGE BEVERAGE	8137	22-12-2015	
4	388	ATLANTIC IMPORTING COMPANY	8169	24-12-2015	

	ReceivingDate	InvoiceDate	PayDate	PurchasePrice	Quantity	Dollars	\
0	02-01-2016	2016-01-04	2016-02-16	35.71	6	214.26	
1	01-01-2016	2016-01-07	2016-02-21	9.35	4	37.40	
2	02-01-2016	2016-01-07	2016-02-21	9.41	5	47.05	
3	01-01-2016	2016-01-07	2016-02-21	9.35	6	56.10	
4	02-01-2016	2016-01-09	2016-02-16	21.32	5	106.60	

	Classification
0	1
1	1
2	1
3	1
4	1

DATA CLEANING

Beg_inventory Data Cleaning :

```python
import pandas as pd
from google.colab import files
uploaded = files.upload()


# Load inventory data
df= pd.read_csv('BegInvFINAL12312016.csv')

import pandas as pd
import numpy as np
import re



# Replace Missing Values with Column Averages (Numeric Columns)
numeric_cols = ["onHand", "Price"]  # Columns where we can calculate the average
for col in numeric_cols:
    df[col].fillna(df[col].mean(), inplace=True)  # Fill missing values with the column average

# Remove Duplicate Rows
df.drop_duplicates(inplace=True)

# Standardize Date Format
# Assuming there's a 'Date' column (replace with actual date column name)
if 'startDate' in df.columns:
    df['startDate'] = pd.to_datetime(df['startDate'], errors='coerce')  # Convert to datetime format
    df['startDate'] = df['startDate'].dt.strftime('%d-%m-%Y')  # Convert format to day-month-year
```

```python
# Standardize Size Format
def standardize_size(size):
    """ Convert mL to L and Liter to 1L """
    if pd.isna(size):
        return np.nan  # Keep NaN as NaN
    size = str(size).lower().strip()  # Convert to lowercase and remove spaces
    if "ml" in size:
        num = re.findall(r'\d+', size)  # Extract numbers
        if num:
            return f"{int(num[0]) / 1000}L"  # Convert mL to L
    elif "liter" in size or "litre" in size:
        return "1L"  # Standardize all "Liter" mentions to "1L"
    return size  # Return unchanged if no match

df["Size"] = df["Size"].apply(standardize_size)


#Display Sample of Cleaned Data
print(df.head())
```

DATA CLEANING

Cleaned Beg_inventory Data

	InventoryId	Store	City	Brand	Description	Size	onHand	Price	startDate
0	1_HARDERSFIELD_58	1	HARDERSFIELD	58	Gekkeikan Black & Gold Sake	0.75L	8	12.99	01-01-2016
1	1_HARDERSFIELD_60	1	HARDERSFIELD	60	Canadian Club 1858 VAP	0.75L	7	10.99	01-01-2016
2	1_HARDERSFIELD_62	1	HARDERSFIELD	62	Herradura Silver Tequila	0.75L	6	36.99	01-01-2016
3	1_HARDERSFIELD_63	1	HARDERSFIELD	63	Herradura Reposado Tequila	0.75L	3	38.99	01-01-2016
4	1_HARDERSFIELD_72	1	HARDERSFIELD	72	No. 3 London Dry Gin	0.75L	6	34.99	01-01-2016

DATA CLEANING

End_inventory Data Cleaning :

```
[ ] import pandas as pd
    from google.colab import files
    uploaded = files.upload()


    # Load inventory data
    df= pd.read_csv('EndInvFINAL12312016.csv')

    import pandas as pd
    import numpy as np
    import re



    # Replace Missing Values with Column Averages (Numeric Columns)
    numeric_cols = ["onHand", "Price"]  # Columns where we can calculate the average
    for col in numeric_cols:
        df[col].fillna(df[col].mean(), inplace=True)  # Fill missing values with the column average

    # Remove Duplicate Rows
    df.drop_duplicates(inplace=True)

    # Standardize Date Format
    # Assuming there's a 'Date' column (replace with actual date column name)
    if 'endDate' in df.columns:
        df['endDate'] = pd.to_datetime(df['endDate'], errors='coerce')  # Convert to datetime format
        df['endDate'] = df['endDate'].dt.strftime('%d-%m-%Y')  # Convert format to day-month-year
```

```
# Standardize Size Format
def standardize_size(size):
    """ Convert mL to L and Liter to 1L """
    if pd.isna(size):
        return np.nan  # Keep NaN as NaN
    size = str(size).lower().strip()  # Convert to lowercase and remove spaces
    if "ml" in size:
        num = re.findall(r'\d+', size)  # Extract numbers
        if num:
            return f"{int(num[0]) / 1000}L"  # Convert mL to L
    elif "liter" in size or "litre" in size:
        return "1L"  # Standardize all "Liter" mentions to "1L"
    return size  # Return unchanged if no match

df["Size"] = df["Size"].apply(standardize_size)


#Display Sample of Cleaned Data
print(df.head())
```

Data Cleaning

Cleaned End_inventory Data

```
  df[col].fillna(df[col].mean(), inplace=True)  # Fill missing values with the column average
         InventoryId  Store           City  Brand                  Description  \
0  1_HARDERSFIELD_58      1  HARDERSFIELD     58  Gekkeikan Black & Gold Sake
1  1_HARDERSFIELD_62      1  HARDERSFIELD     62     Herradura Silver Tequila
2  1_HARDERSFIELD_63      1  HARDERSFIELD     63   Herradura Reposado Tequila
3  1_HARDERSFIELD_72      1  HARDERSFIELD     72         No. 3 London Dry Gin
4  1_HARDERSFIELD_75      1  HARDERSFIELD     75    Three Olives Tomato Vodka

    Size  onHand  Price     endDate
0  0.75L      11  12.99  31-12-2016
1  0.75L       7  36.99  31-12-2016
2  0.75L       7  38.99  31-12-2016
3  0.75L       4  34.99  31-12-2016
4  0.75L       7  14.99  31-12-2016
```

FACTS AND DIMENSIONS TABLES

FactSales:
SalesQuantity
SalesDollars
SalesPrice
StoreId
ProductId
DateId

FactInventory:
InventoryId
onHandStart
onHandEnd
StockRecieved
StockSold
StockTurnover
StoreId
ProductId
DateId
VendorId

StoreDim:
StoreId
City

DateDim:
DateId
Day
Month
Quarter
Year

ProductDim:
ProductId
Brand

VendorDim:
VendorId
VendorName

TABLES' CREATION

Creation of dimension Tables

Creation of database :

```
1   -- 1 Create the database
2   CREATE DATABASE Drinks;
3   USE Drinks;  -- Select the database to use
4
```

Creation of product dimension table :

```
5   -- 2 Create Dimension Tables
6   CREATE TABLE ProductDim (
7       ProductId INT PRIMARY KEY AUTO_INCREMENT,
8
9       Brand VARCHAR(255)
10  );
```

Creation of store dimension table :

```
12  CREATE TABLE StoreDim (
13      StoreId INT PRIMARY KEY AUTO_INCREMENT,
14      Store VARCHAR(255),
15      City VARCHAR(255)
16  );
```

Creation of date dimension table :

```
CREATE TABLE DateDim (
    DateId INT PRIMARY KEY AUTO_INCREMENT,
    Day INT,
    Month INT,
    Quarter INT,
    Year INT
);
```

Creation of vendor dimension table :

```
CREATE TABLE VendorDim (
    VendorId INT PRIMARY KEY AUTO_INCREMENT,
    VendorName VARCHAR(50) UNIQUE
);
```

TABLES' CREATION

Creation of Facts Tables

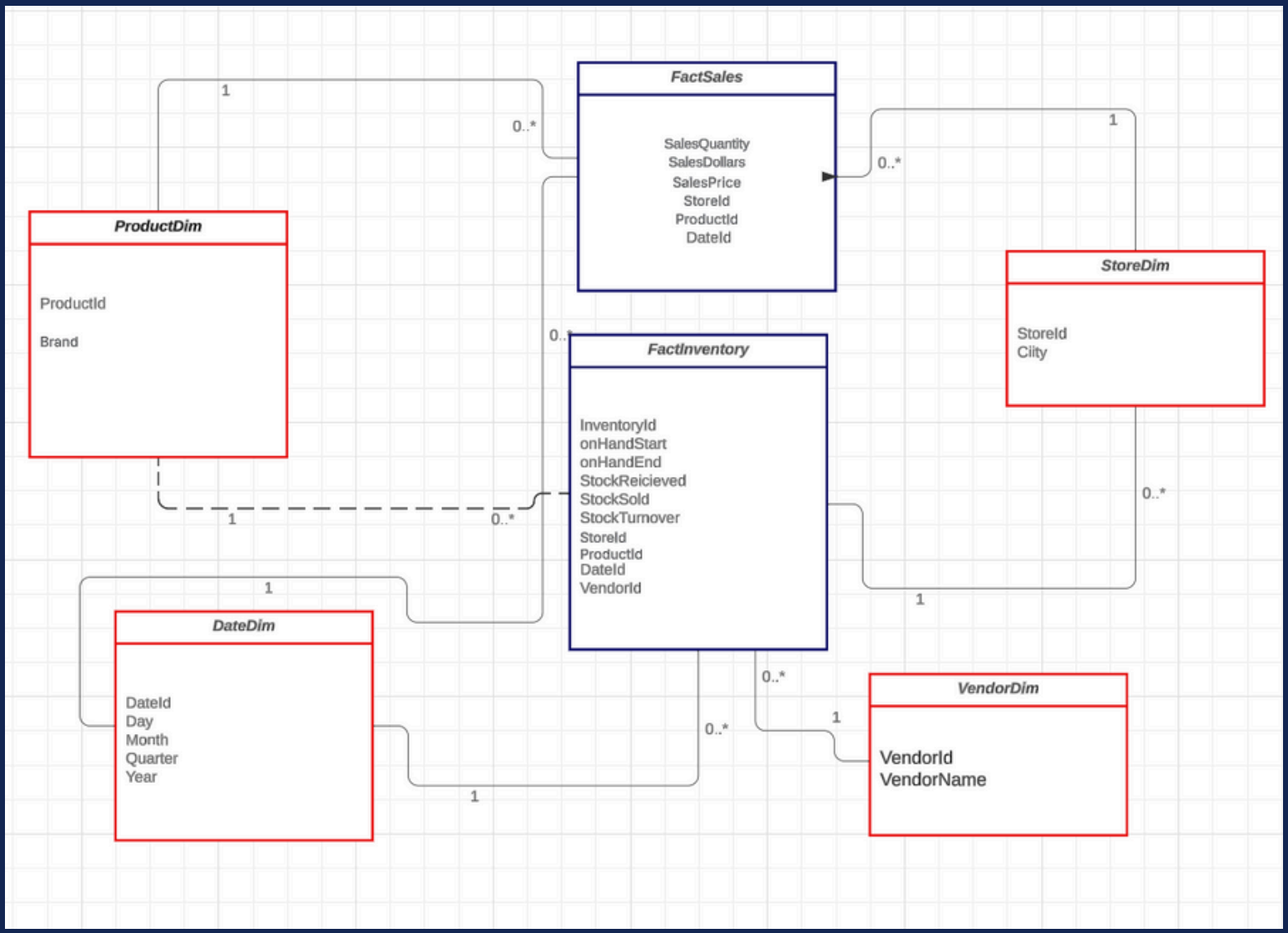
Creation of Inventory Fact table :

```
CREATE TABLE FactInventory (
    InventoryId INT PRIMARY KEY AUTO_INCREMENT,
    onHandStart INT,
    onHandEnd INT,
    StockReceived INT,
    StockSold INT,
    StockTurnover DECIMAL(10,2),
    StoreId INT,
    ProductId INT,
    DateId INT,
    VendorId INT,
    FOREIGN KEY (StoreId) REFERENCES StoreDim(StoreId),
    FOREIGN KEY (ProductId) REFERENCES ProductDim(ProductId),
    FOREIGN KEY (DateId) REFERENCES DateDim(DateId),
    FOREIGN KEY (VendorId) REFERENCES VendorDim(VendorId)
);
```

Creation of Sales Fact table :

```
CREATE TABLE FactSales (
    SalesId INT PRIMARY KEY AUTO_INCREMENT,
    SalesQuantity INT,
    SalesDollars DECIMAL(10,2),
    SalesPrice DECIMAL(10,2),
    StoreId INT,
    ProductId INT,
    DateId INT,
    VendorId INT,
    FOREIGN KEY (StoreId) REFERENCES StoreDim(StoreId),
    FOREIGN KEY (ProductId) REFERENCES ProductDim(ProductId),
    FOREIGN KEY (DateId) REFERENCES DateDim(DateId),
);
```

FACT CONSTELLATION SCHEMA



DATA TRANSFORMATION

Mapping the Dimensions' IDs and Creating the Dimensions:

```
# Step 1: Prepare the dimension mappings for IDs
# Create ProductDim
product_dim = beginning_inventory[['Brand']].drop_duplicates().reset_index(drop=True)
product_dim['ProductId'] = range(1, len(product_dim) + 1)

# Create StoreDim
store_dim = beginning_inventory[['Store', 'City']].drop_duplicates().reset_index(drop=True)
store_dim['StoreId'] = range(1, len(store_dim) + 1)

# Create VendorDim using VendorNumber and VendorName
vendor_dim = purchases[['VendorNumber', 'VendorName']].drop_duplicates().reset_index(drop=True)
vendor_dim.rename(columns={'VendorNumber': 'VendorId'}, inplace=True)
```

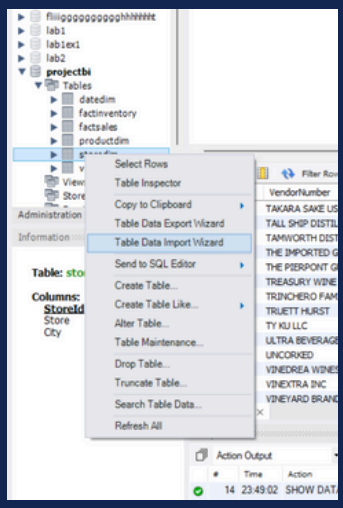
```
# Create DateDim
def extract_date_parts(date_column):
    # Ensure date_column is a Series
    date_column = pd.Series(date_column)

    return pd.DataFrame({
        'DateId': range(1, len(date_column) + 1),
        'Day': date_column.dt.day,
        'Month': date_column.dt.month,
        'Quarter': date_column.dt.quarter,
        'Year': date_column.dt.year,
        # Include the original date column for merging
        'Date': date_column
    })

dates = pd.concat([sales['SalesDate'], purchases['ReceivingDate']])
dates = pd.to_datetime(dates.dropna().unique(), format='%d-%m-%Y', errors='coerce')
date_dim = extract_date_parts(dates)  # Pass dates as a Series
```

Creating Fact Tables and calculating necessary measures:

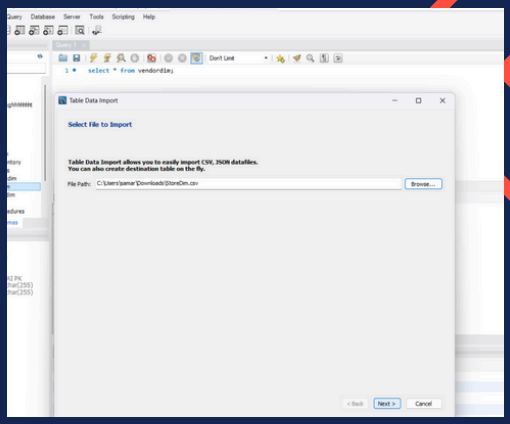
```python
# Step 3: Create FactSales
# Map ProductId, StoreId, DateId, and VendorId
sales_fact = sales.copy()
sales_fact = sales_fact.merge(product_dim, on='InventoryId', how='left')
sales_fact = sales_fact.merge(store_dim, on='Store', how='left')
sales_fact['SalesDate'] = pd.to_datetime(sales_fact['SalesDate'], format='%d-%m-%Y', errors='coerce')

# Merge on 'SalesDate' and 'Date' (datetime columns)
sales_fact = sales_fact.merge(date_dim, left_on='SalesDate', right_on='Date', how='left')

sales_fact = sales_fact.merge(vendor_dim, on='VendorName', how='left')


# Fill null values with defaults
sales_fact['SalesQuantity'] = sales_fact['SalesQuantity'].fillna(0)
sales_fact['SalesDollars'] = sales_fact['SalesDollars'].fillna(0.0)
sales_fact['SalesPrice'] = sales_fact['SalesPrice'].fillna(0.0)
sales_fact['StoreId'] = sales_fact['StoreId'].fillna(-1).astype(int)
sales_fact['ProductId'] = sales_fact['ProductId'].fillna(-1).astype(int)
sales_fact['DateId'] = sales_fact['DateId'].fillna(-1).astype(int)

# Select relevant columns for FactSales
sales_fact_table = sales_fact[['SalesQuantity', 'SalesDollars', 'SalesPrice', 'StoreId', 'ProductId', 'DateId']]
```

```python
# Step 4: Create FactInventory
# Merge beginning and ending inventory
inventory_fact = beginning_inventory.merge(end_inventory, on=['InventoryId', 'Store'], suffixes=('_start', '_end'), how='outer')

# Map ProductId and StoreId
inventory_fact = inventory_fact.merge(product_dim, on='InventoryId', how='left')  # Add ProductId
inventory_fact = inventory_fact.merge(store_dim, on='Store', how='left')  # Add StoreId

# Fill null values for start and end inventory
inventory_fact['onHandStart'] = inventory_fact['onHand_start'].fillna(0).astype(int)
inventory_fact['onHandEnd'] = inventory_fact['onHand_end'].fillna(0).astype(int)

# Calculate StockReceived and StockSold
stock_received = purchases.groupby('InventoryId').size().reindex(inventory_fact['InventoryId'], fill_value=0)
inventory_fact['StockReceived'] = stock_received.values

stock_sold = sales.groupby('InventoryId')['SalesQuantity'].sum().reindex(inventory_fact['InventoryId'], fill_value=0)
inventory_fact['StockSold'] = stock_sold.values

# Calculate StockTurnover (Avoid division by zero)
inventory_fact['StockTurnover'] = (
    inventory_fact['StockSold'] / inventory_fact['onHandStart'].replace(0, np.nan)
).fillna(0).round(2)
```

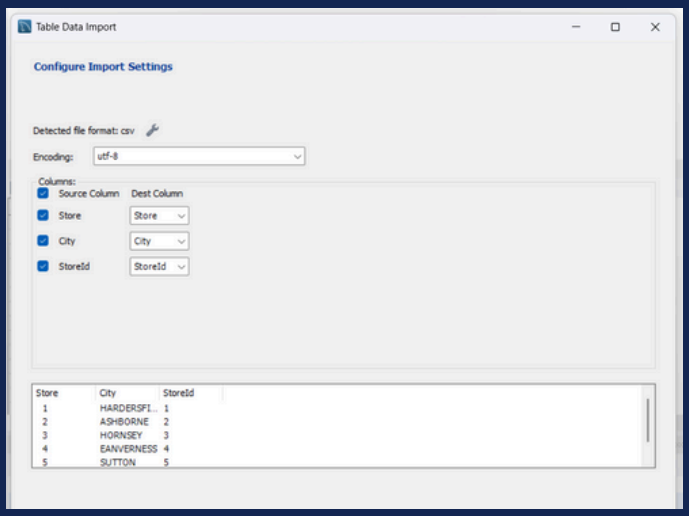
Table data import wizard



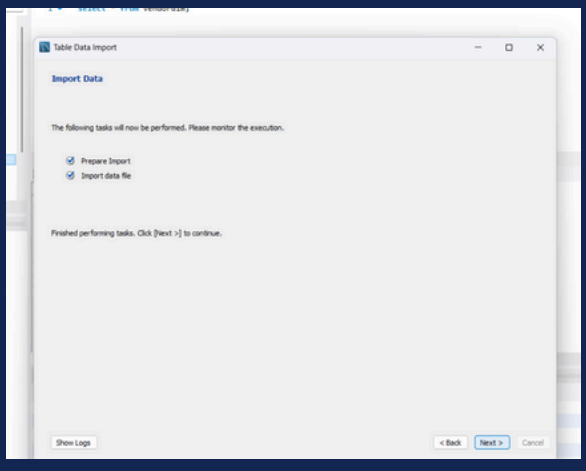
1/ Table data import wizard



2/ Select csv file to import



3/Configure import settings



4/Import Data

Created Tables:

Facts Tables

Factinventory table :

InventoryId	onHandStart	onHandEnd	StockReceived	StockSold	StockTurnover	StoreId	ProductId	DateId
1_HARDERSFIELD_126	5	35	24	15	3.00	1	9	363
1_HARDERSFIELD_13397	30	7	13	29	0.97	1	2003	363
1_HARDERSFIELD_13614	5	23	8	17	3.40	1	2016	363
1_HARDERSFIELD_1964	20	11	4	1	0.05	1	280	363
1_HARDERSFIELD_20204	5	9	52	31	6.20	1	2349	363
1_HARDERSFIELD_2040	39	15	1	9	0.23	1	297	363
1_HARDERSFIELD_20533	6	13	8	11	1.83	1	2365	363
1_HARDERSFIELD_20800	28	66	9	23	0.82	1	2380	363
1_HARDERSFIELD_2232	3	4	2	4	1.33	1	374	363
1_HARDERSFIELD_2464	4	22	11	6	1.50	1	423	363
1_HARDERSFIELD_2889	2	4	2	1	0.50	1	585	363
1_HARDERSFIELD_3354	36	50	7	17	0.47	1	722	363
1_HARDERSFIELD_35103	5	22	3	5	1.00	1	2909	363

FactSales table :

SalesId	SalesQuantity	SalesDollars	SalesPrice	StoreId	ProductId	DateId
388	2	55.98	27.99	1	9	9
389	1	27.99	27.99	1	9	10
390	1	27.99	27.99	1	9	16
391	1	27.99	27.99	1	9	5
392	1	27.99	27.99	1	9	23
393	1	27.99	27.99	1	9	25
394	1	27.99	27.99	1	9	28
395	2	55.98	27.99	1	9	13
396	1	27.99	27.99	1	9	20
397	1	27.99	27.99	1	9	8
936	1	9.99	9.99	1	2003	2
937	2	25.98	12.99	1	2003	15
1064	3	41.97	13.99	1	2016	17

Created Tables:

Dimension Tables

Product dimension table

ProductId	Brand
9	126
280	1964
297	2040
374	2232
423	2464
585	2889
722	3354
804	3566
1026	4187
1254	5195
1766	8450
2003	13397
2016	13614
2214	17814

Vendor dimension table

VendorId	vendorName
54	AAPER ALCOHOL & CHEMICAL CO
60	ADAMBA IMPORTS INTL INC
1703	ALISA CARR BEVERAGES
105	ALTAMAR BRANDS LLC
200	AMERICAN SPIRITS EXCHANGE
4466	AMERICAN VINTAGE BEVERAGE
287	APPOLO VINEYARDS LLC
388	ATLANTIC IMPORTING COMPANY
480	BACARDI USA INC
516	BANFI PRODUCTS CORP
90059	BLACK COVE BEVERAGES
2396	BLACK PRINCE DISTILLERY INC
1265	BLACK ROCK SPIRITS LLC

Date dimension table

DateId	Day	Month	Quarter	Year
1	21	12	4	2015
2	22	12	4	2015
3	24	12	4	2015
4	20	12	4	2015
5	25	12	4	2015
6	23	12	4	2015
7	29	12	4	2015
8	27	12	4	2015
9	1	1	1	2016
10	30	12	4	2015
11	28	12	4	2015
12	31	12	4	2015
13	3	1	1	2016
14	4	1	1	2016

Store dimension table

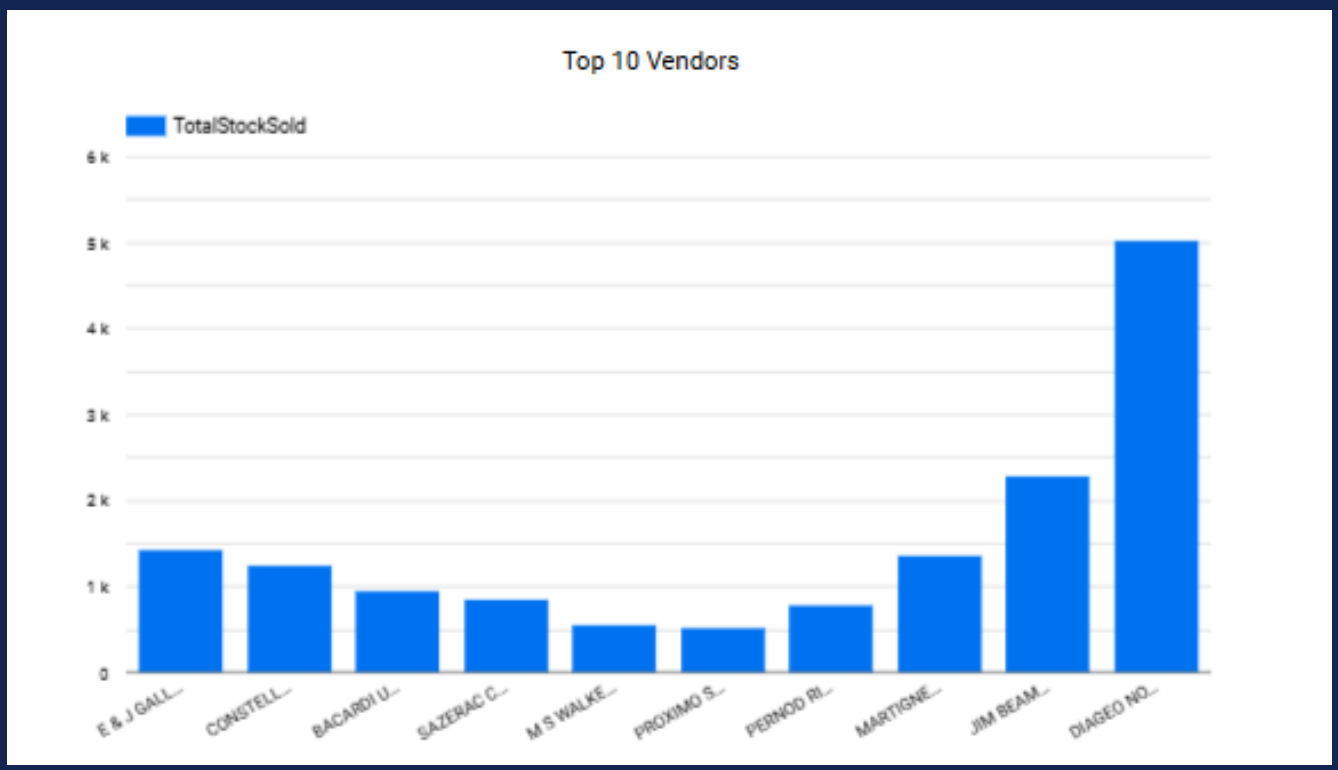
StoreId	Store	City
1	1	HARDERSFIELD
2	2	ASHBORNE
3	3	HORNSEY
4	4	EANVERNESS
5	5	SUTTON
6	6	GOULCREST
7	7	STANMORE
8	8	ALNERWICK
9	9	BLACKPOOL
10	10	HORNSEY
11	11	CARDEND
12	12	LEESIDE
13	13	TARMSWORTH
14	14	BROMWICH

Data Visualization: Top 10 Vendors

We implemented **ROLAP in MySQL**, **emulated the analytical cubes** within the database, and **visualized the insights** using **Looker** for comprehensive analysis.

```sql
SELECT
    v.VendorName AS Vendor,
    SUM(fi.StockSold) AS TotalStockSold
FROM
    FactInventory fi
JOIN
    VendorDim v ON fi.VendorId = v.VendorId
GROUP BY
    v.VendorName
ORDER BY
    TotalStockSold DESC
LIMIT 10;
```

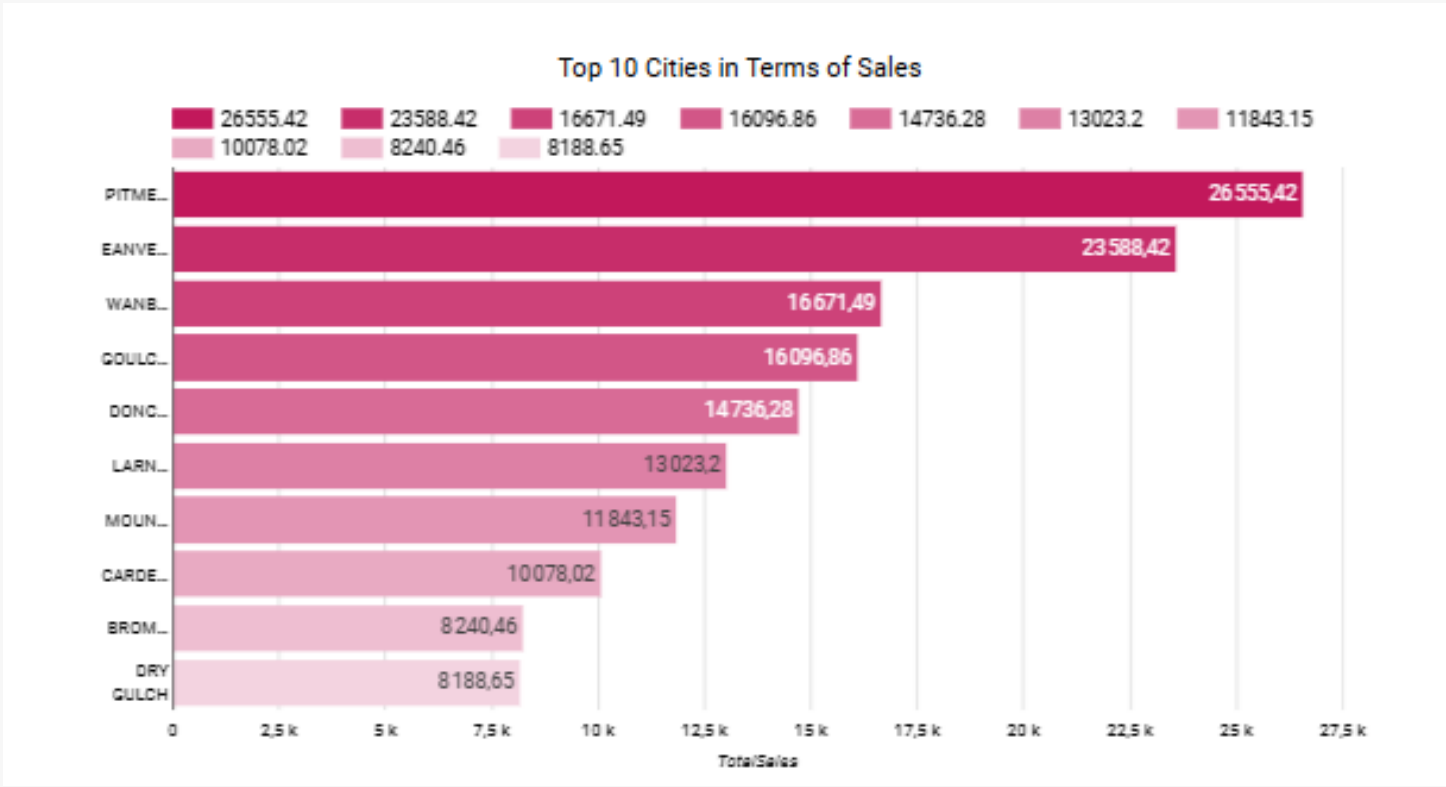
Looker



The analysis identified the top ten vendors based on stock sold. DIAGEO NORTH AMERICA INC is the leading vendor, contributing significantly with 5,000 units sold, followed by Jim Beam Brands at 2,800 units.

Data Visualization: Top 10 Cities

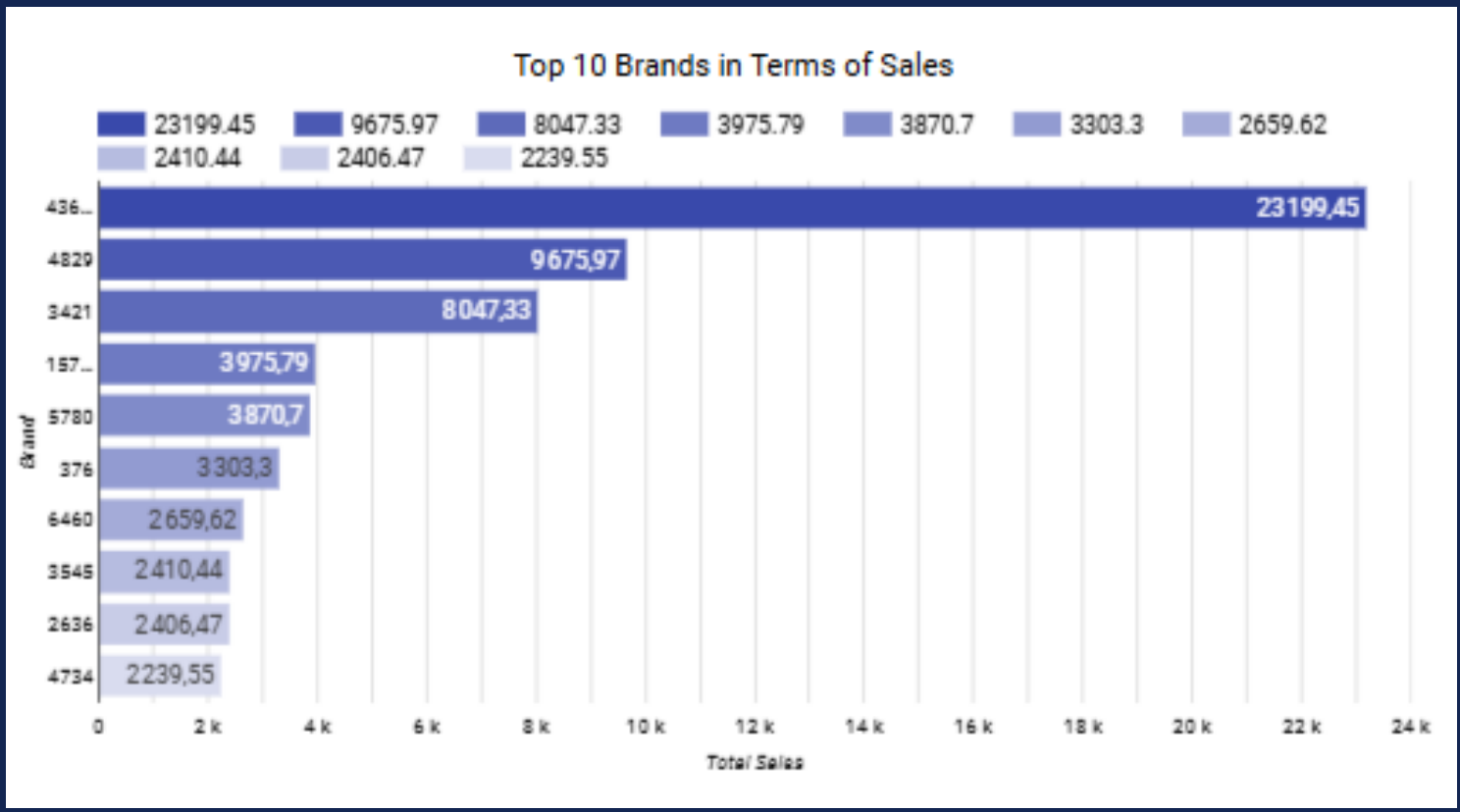

```sql
SELECT
    s.City,
    SUM(fs.SalesDollars) AS TotalSales
FROM
    FactSales fs
JOIN
    StoreDim s ON fs.StoreId = s.StoreId
GROUP BY
    s.City
ORDER BY
    TotalSales DESC
LIMIT 10;
```



The analysis reveals that PITMERDEN is the top-performing city in terms of sales, contributing a significant total of $26,555.42. This highlights the city's importance as a major revenue driver for the company.

Data Visualization: Top 10 Brands

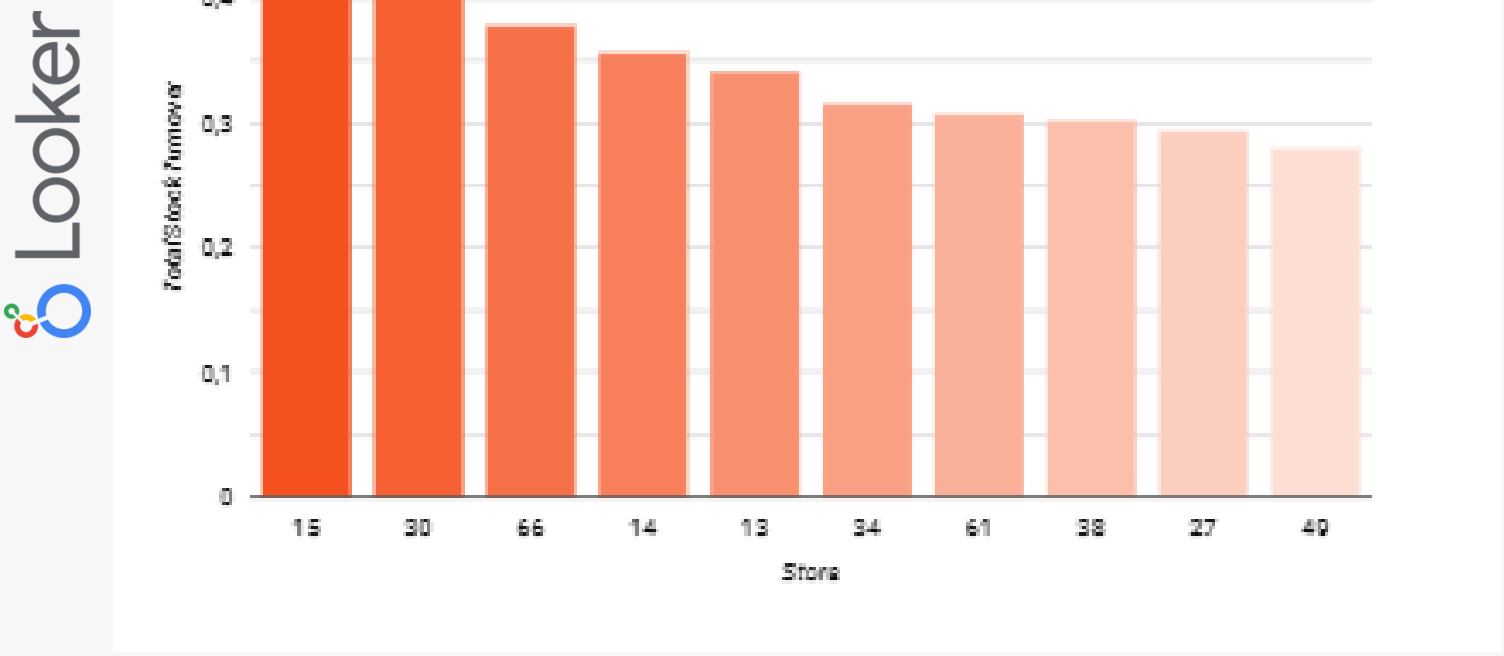

```sql
SELECT
    p.Brand AS Brand,
    SUM(fs.SalesDollars) AS TotalSales
FROM
    FactSales fs
JOIN
    ProductDim p ON fs.ProductId = p.ProductId
GROUP BY
    p.Brand
ORDER BY
    TotalSales DESC
LIMIT 10;
```



The analysis reveals that is the brand 4360 is the top-performing brand in terms of sales, contributing a significant total of $23,199.45. This highlights the city's importance as a major revenue driver for the company.

Data Visualization: Stores with highest Stock Turnover

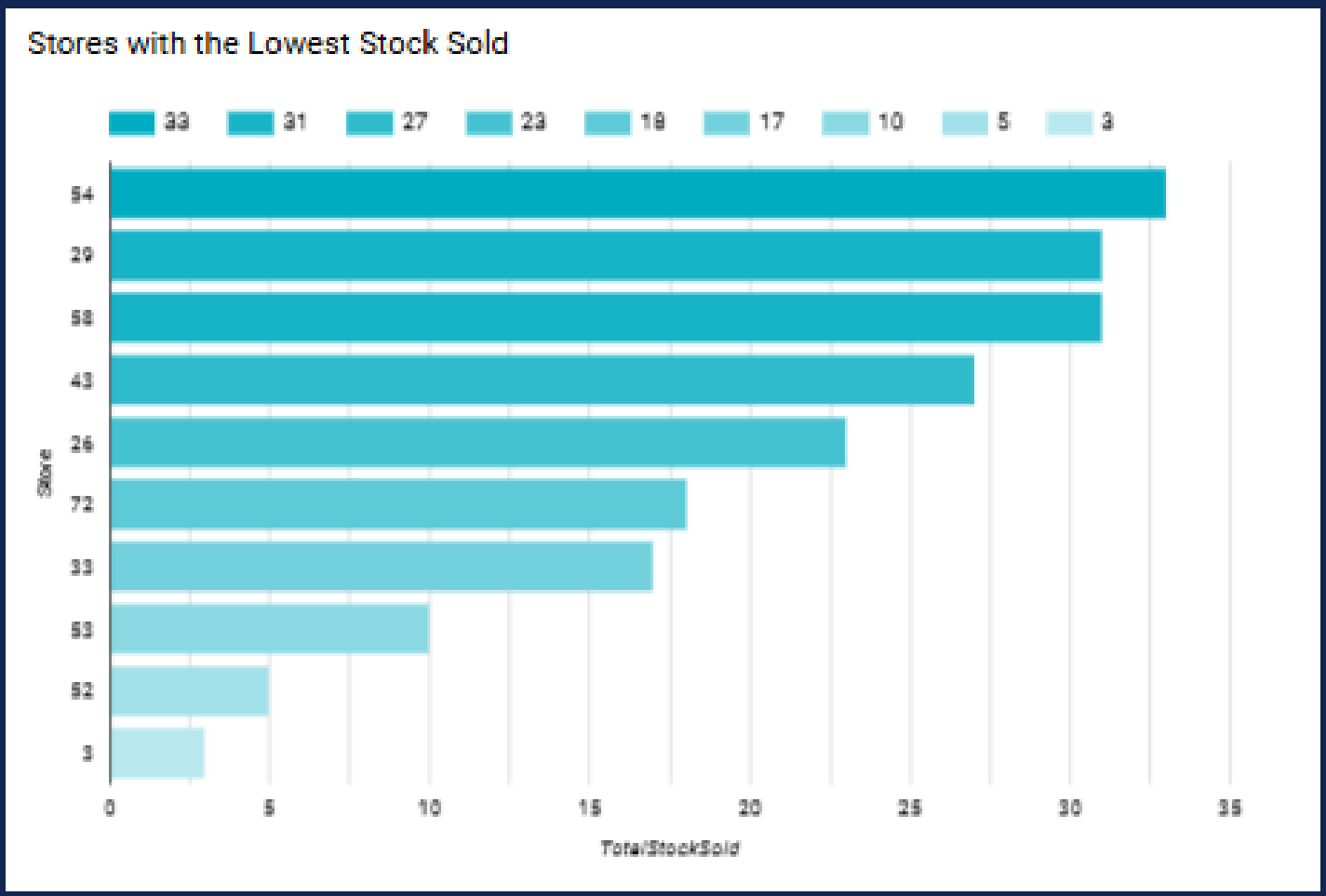

```sql
SELECT
    s.Store AS Store,
    SUM(fi.StockTurnover) AS TotalStockTurnover
FROM
    FactInventory fi
JOIN
    StoreDim s ON fi.StoreId = s.StoreId
GROUP BY
    s.Store
ORDER BY
    TotalStockTurnover DESC
LIMIT 10;
```



Store 15 has the highest stock turnover rate of 49.71%, indicating its efficiency in converting inventory into sales and its pivotal role in optimizing the company's inventory management.

Data Visualization: Stores with the Lowest Stock Sold

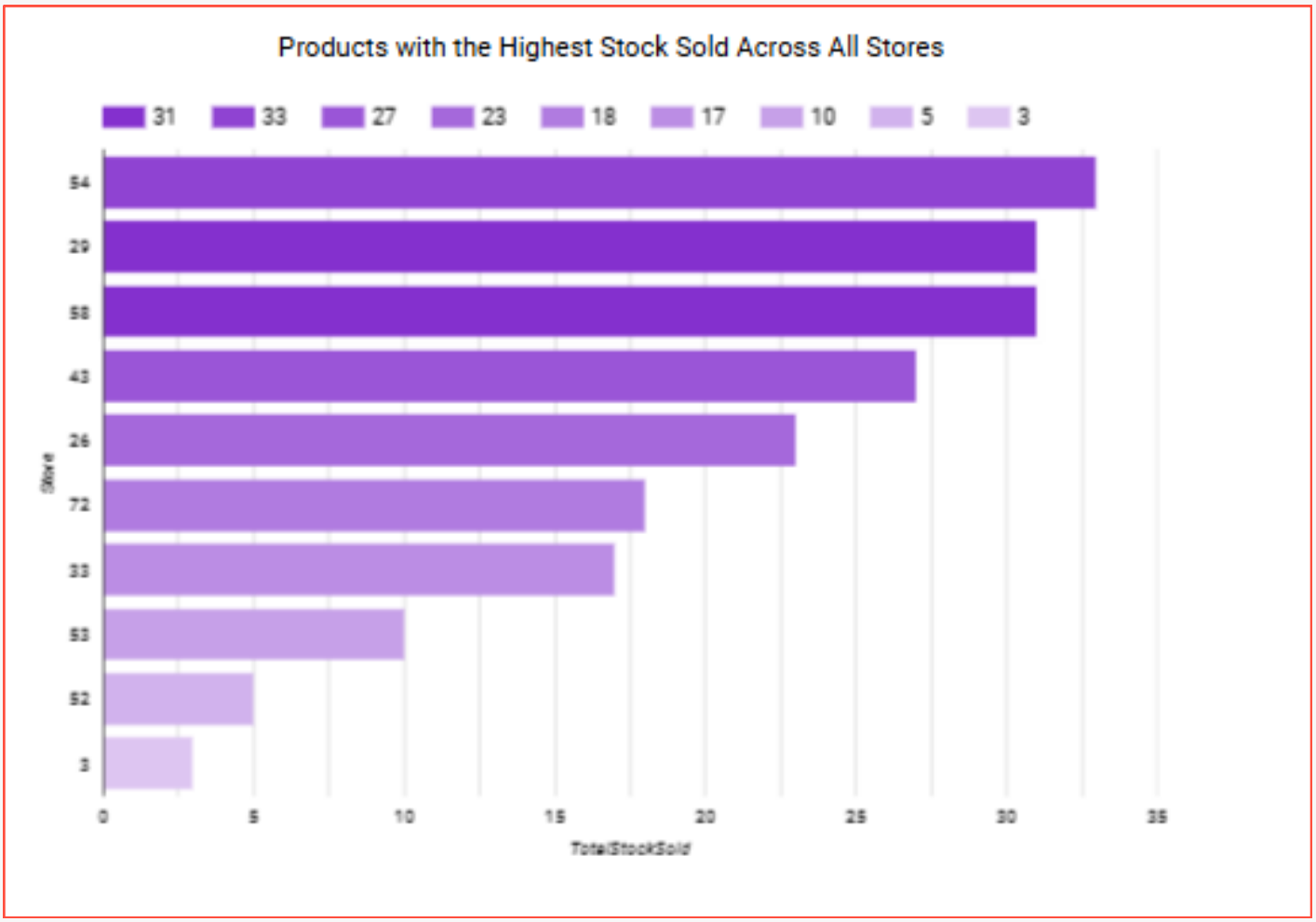
```sql
SELECT
    s.Store AS Store,
    SUM(fi.StockSold) AS TotalStockSold
FROM
    FactInventory fi
JOIN
    StoreDim s ON fi.StoreId = s.StoreId
GROUP BY
    s.Store
ORDER BY
    TotalStockSold ASC
LIMIT 10;
```



Store 3 has the lowest stock sold, with only 3 units, highlighting its underperformance in sales and potential challenges in demand generation or inventory management.

Data Visualization: Products with the Highest Stock Sold Across All Stores

```sql
SELECT
    p.Brand AS Product,
    SUM(fi.StockSold) AS TotalStockSold
FROM
    FactInventory fi
JOIN
    ProductDim p ON fi.ProductId = p.ProductId
GROUP BY
    p.Brand
ORDER BY
    TotalStockSold DESC
LIMIT 10;
```

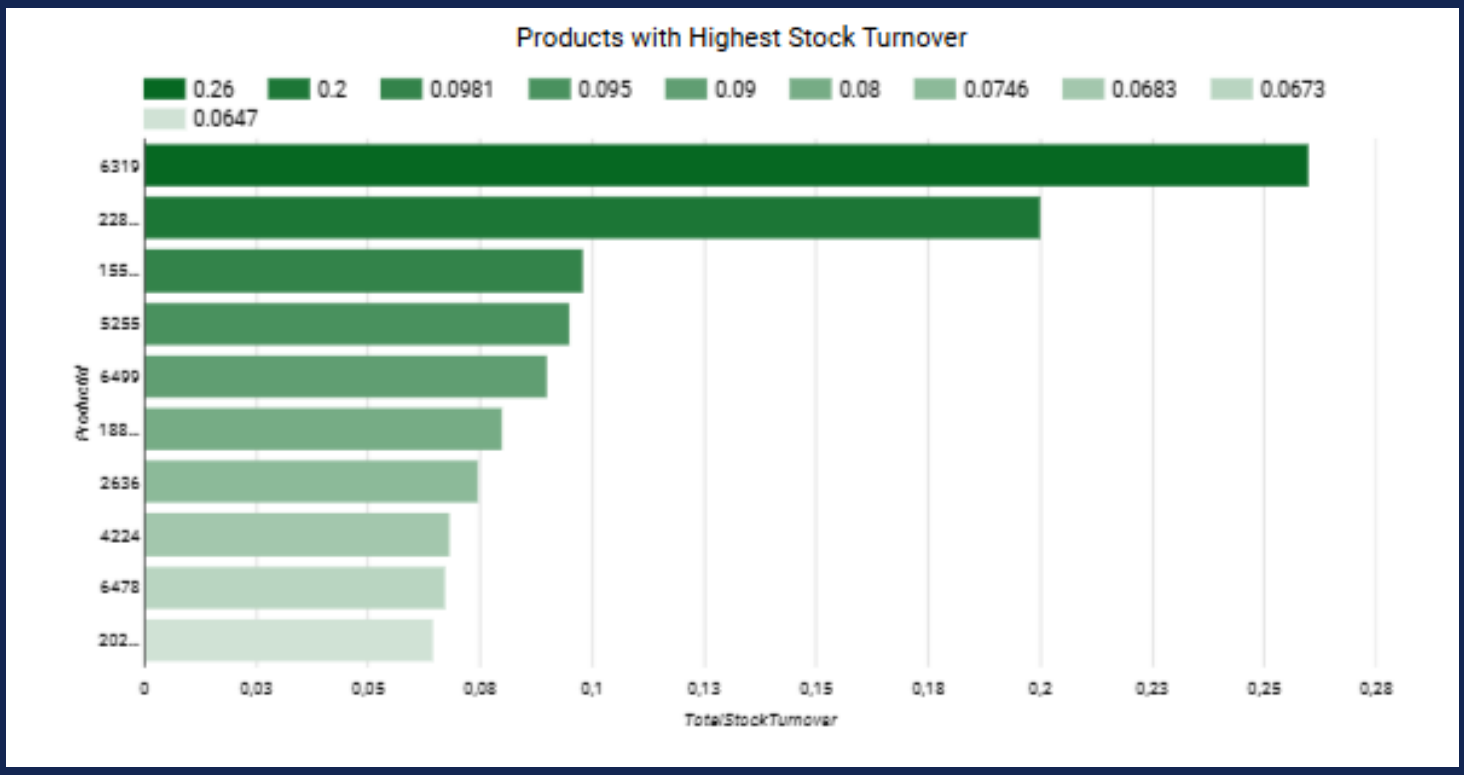


Looker

The product HARDERSFIELD-54 has the highest stock sold across all stores, with a total of 31 units, highlighting its popularity and strong customer demand.

Data Visualization: Products with Highest Stock Turnover

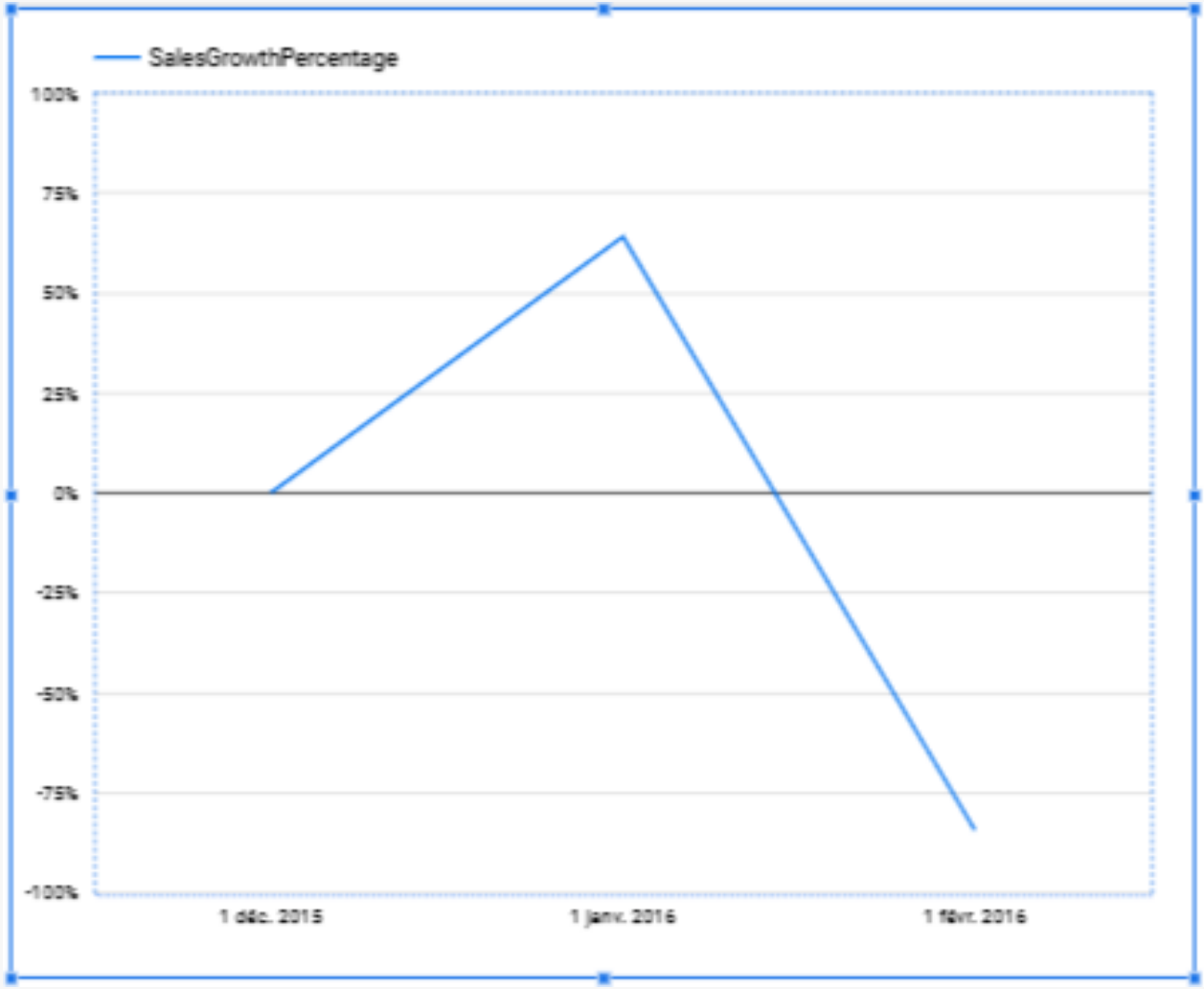

```sql
SELECT
    p.Brand AS Product,
    SUM(fi.StockTurnover) AS TotalStockTurnover
FROM
    FactInventory fi
JOIN
    ProductDim p ON fi.ProductId = p.ProductId
GROUP BY
    p.Brand
ORDER BY
    TotalStockTurnover DESC
LIMIT 10;
```



The product **HARDERSFIELD-3619** has the **highest stock turnover across all stores**, with a **rate of 26%**, indicating its **efficiency in inventory movement** and strong demand relative to its stock level

Data Visualization: Month-to-Month Sales Growth

```sql
SELECT
    CONCAT(d.Year, '-', LPAD(d.Month, 2, '0')) AS Month,
    SUM(fs.SalesDollars) AS TotalSales,
    (SUM(fs.SalesDollars) - LAG(SUM(fs.SalesDollars)) OVER (ORDER BY d.Year, d.Month)) /
    NULLIF(LAG(SUM(fs.SalesDollars)) OVER (ORDER BY d.Year, d.Month), 0) * 100 AS SalesGrowthPercentage
FROM
    FactSales fs
JOIN
    DateDim d ON fs.DateId = d.DateId
GROUP BY
    d.Year, d.Month
ORDER BY
    d.Year, d.Month;
```



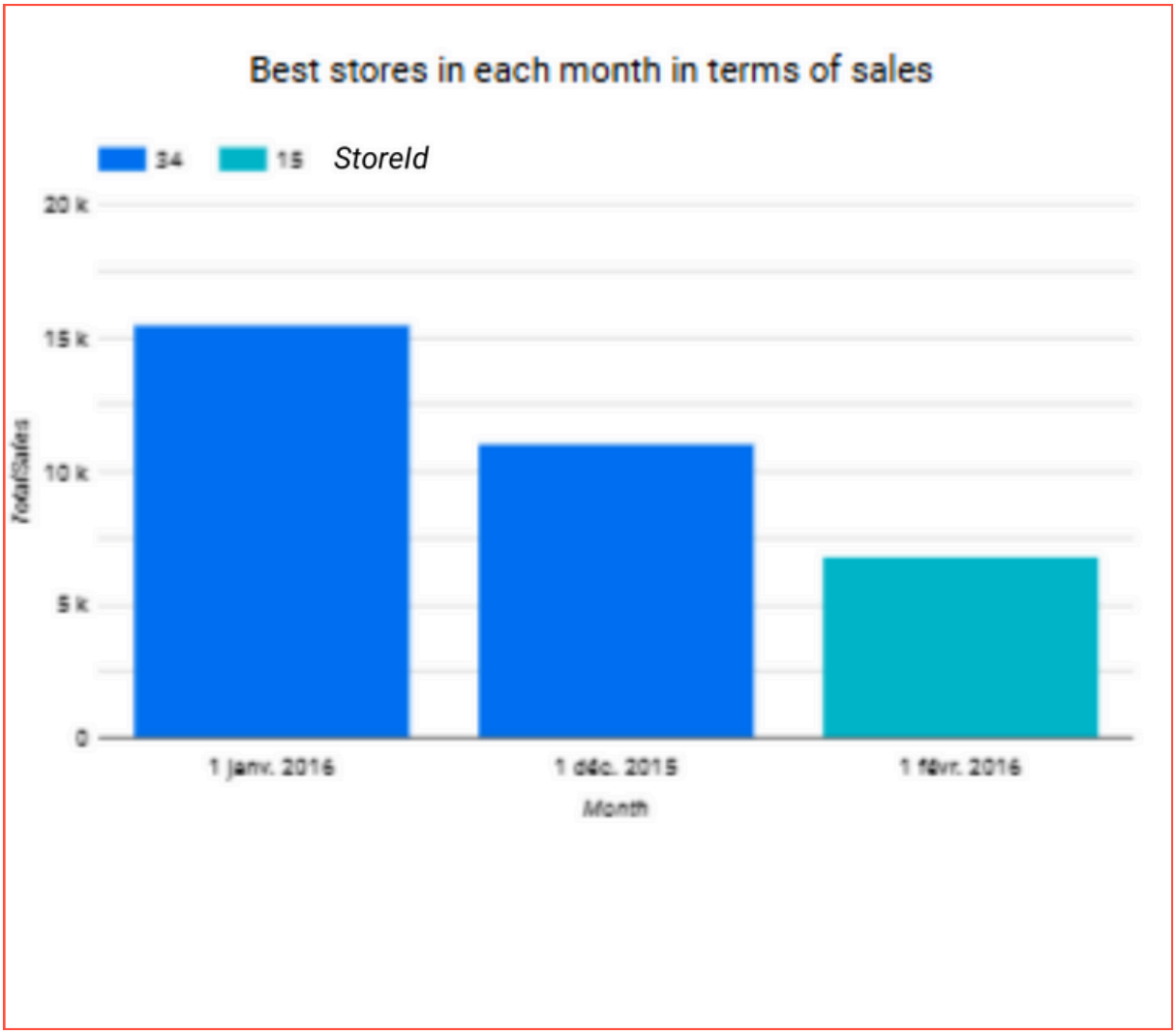
The month-to-month sales growth visualization revealed that January 1, 2016, experienced the highest growth rate, reaching 68%. This indicates a significant spike in sales during this period.

Data Visualization: Best Stores each month in terms of Sales

Cube For Monthly Stores Sales

```sql
CREATE OR REPLACE VIEW MonthlyStoreSales AS
SELECT
    d.Year,
    d.Month,
    s.Store,
    SUM(f.SalesDollars) AS TotalSales
FROM
    FactSales f
    JOIN DateDim d ON f.DateId = d.DateId
    JOIN StoreDim s ON f.StoreId = s.StoreId
GROUP BY
    d.Year, d.Month, s.Store;
```

```sql
SELECT
    Year,
    Month,
    Store,
    TotalSales
FROM
    MonthlyStoreSales
WHERE
    (Year, Month, TotalSales) IN (
        SELECT
            Year, Month, MAX(TotalSales)
        FROM
            MonthlyStoreSales
        GROUP BY
            Year, Month
    );
```



Store 34 demonstrated the highest sales performance in both December 2015 and January 2016, with a peak in January 2016, achieving over 15k in sales. Store 15 emerged as the top-performing store in February 2016, albeit with a comparatively lower total sales figure.

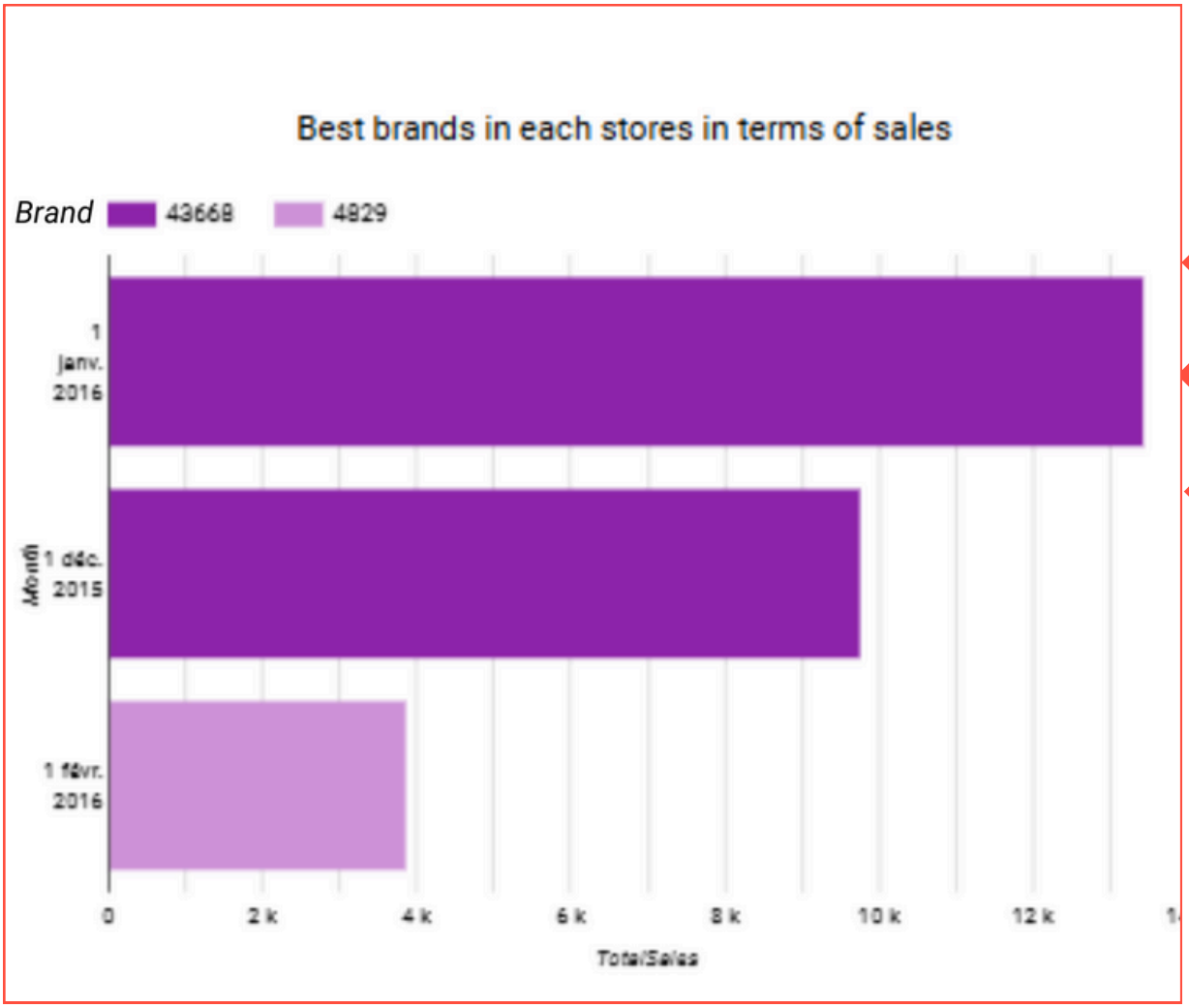
Data Visualization: Best Brands each Month in Terms of Sales

Cube For Monthly Brands Sales

```sql
CREATE OR REPLACE VIEW MonthlyBrandSales AS
SELECT
    d.Year,
    d.Month,
    p.Brand,
    SUM(f.SalesDollars) AS TotalSales
FROM
    FactSales f
    JOIN DateDim d ON f.DateId = d.DateId
    JOIN ProductDim p ON f.ProductId = p.ProductId
GROUP BY
    d.Year, d.Month, p.Brand;
```

```sql
SELECT
    Year,
    Month,
    Brand,
    TotalSales
FROM
    MonthlyBrandSales
WHERE
    (Year, Month, TotalSales) IN (
        SELECT
            Year, Month, MAX(TotalSales)
        FROM
            MonthlyBrandSales
        GROUP BY
            Year, Month
    );
```

Looker



Brand 43668 consistently outperformed other brands in sales, with a significant lead in January 2016 and December 2015. In February 2016, Brand 4829 emerged as the top-selling brand but with a lower overall sales figure compared to Brand 43668 in the prior months.

Decisions to Take

Vendor Relationships:

- Strengthen partnerships with DIAGEO NORTH AMERICA INC and Jim Beam Brands by ensuring a consistent supply and exploring exclusive collaboration opportunities.

Key Market Focus:

- Invest in PITMERDEN as a key market by increasing marketing efforts, promotional activities, and tailored product offerings to drive further growth.

Brand Strategies:

- Focus on Brand 4360 to maintain and boost its strong sales performance.
- Promote Brand 43668 as a primary revenue driver while

Decisions to Take

Store Performance:

- **Analyze and replicate the successful operational strategies of Store 15, which has the highest stock turnover rate, across other stores.**
- **Investigate Store 34's consistently high performance, especially in January 2016, and apply its strategies to other locations.**
- **Address underperformance in Store 3 by identifying its challenges and implementing targeted improvement measures.**

Product Optimization:

- **Ensure consistent stock availability for HARDERSFIELD-54, the highest-selling product, to meet demand and avoid stockouts.**
- **Regularly replenish HARDERSFIELD-3619 to maintain its high turnover rate and optimize inventory management.**

CONCLUSION

The decisions outlined provide a comprehensive roadmap for leveraging the company's key strengths and addressing critical challenges. By focusing on high-performing vendors, brands, and stores, while addressing areas of improvement and optimizing inventory management, the company can enhance its overall sales performance, operational efficiency, and market competitiveness. These data-driven strategies position the company for sustainable growth and improved customer satisfaction.