EMPOVATION DATA ANALYSIS

INTRODUCTION

The provided report offers an insightful analysis of an electronic dataset. This report is divided into four batches, each containing specific questions and queries relevant to the analysis.

DATA COLLECTION

The dataset is structured and secondary, containing over 150,000 rows and multiple columns. The data was sourced from Maven Playground. The analysis was conducted to answer specific questions detailed in each batch.

TOOLS

- **SQL**: Used for data analysis.
- **Excel Editor**: Utilized for data cleaning.

DATA CLEANING AND TRANSFORMATION

The following cleaning procedures were performed:

- Changed data types where necessary.
- Filtered rows to remove irrelevant data.
- Removed errors to ensure data accuracy.
- Removed null values to maintain data integrity.

DATA DESCRIPTION

Below is a detailed description of the tables and fields in the dataset:

Tables and Fields

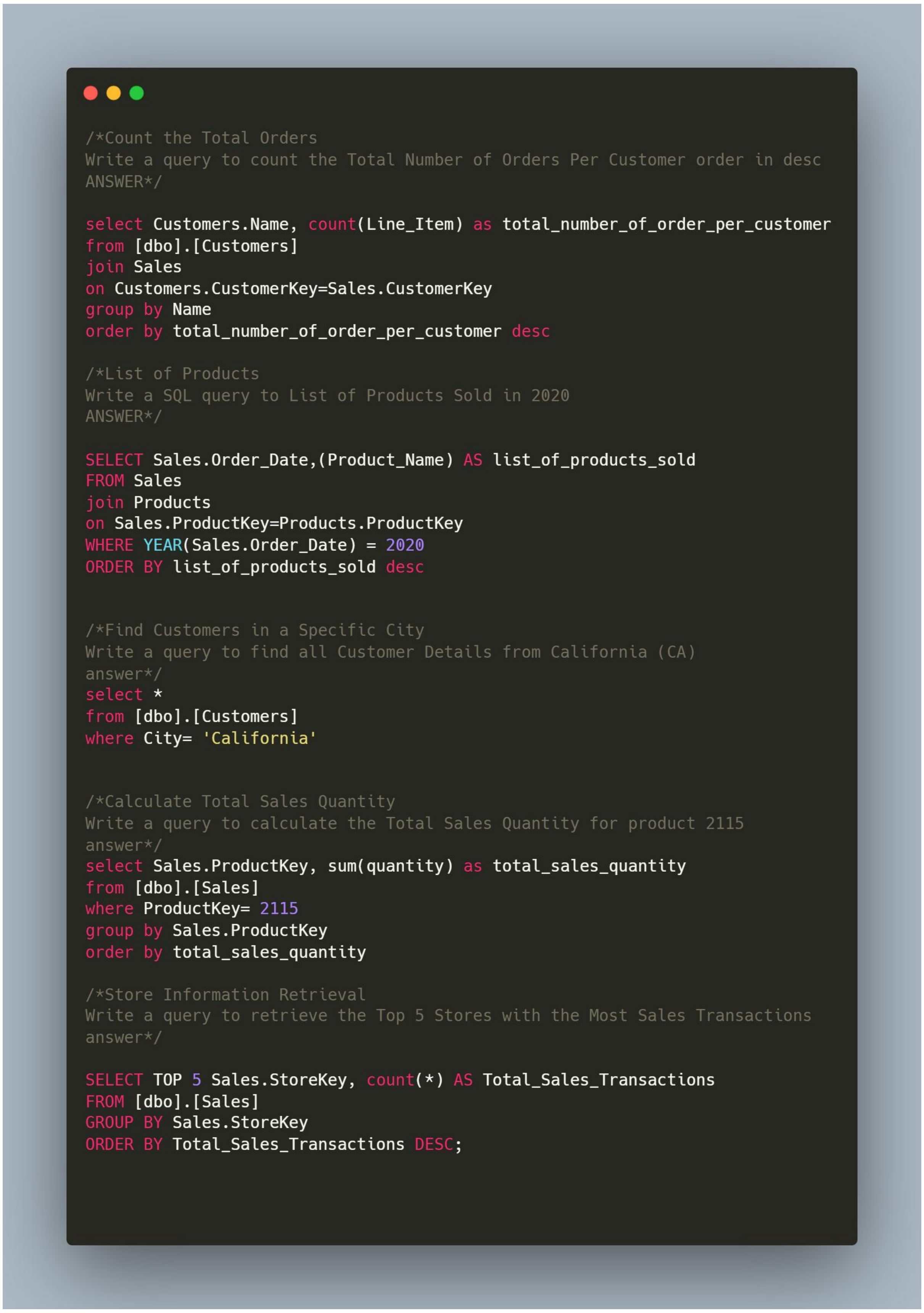
Table	Field	Description
Sales	Order Number	Unique ID for each order
Sales	Line Item	Identifies individual products in an order
Sales	Order Date	Date the order was placed
Sales	Delivery Date	Date the order was delivered
Sales	CustomerKey	Unique key identifying the customer who placed the order
Sales	StoreKey	Unique key identifying the store that processed the order
Sales	ProductKey	Unique key identifying the product purchased
Sales	Quantity	Number of items purchased
Sales	Currency Code	Currency used to process the order
Categories	CategoryKey	Key to identify product categories
Table	**Field**	**Description**

Categories	Category	Product category name
Categories	SubcategoryKey	Key to identify product subcategories
Categories	Subcategory	Product subcategory name
Customers	CustomerKey	Primary key to identify customers
Customers	Gender	Customer gender
Customers	Name	Customer full name
Customers	City	Customer city
Customers	State Code	Customer state (abbreviated)
Customers	State	Customer state (full)
Customers	Zip Code	Customer zip code
Customers	Country	Customer country
Customers	Continent	Customer continent
Customers	Birthday	Customer date of birth
Products	ProductKey	Primary key to identify products
Products	Product Name	Product name
Products	Brand	Product brand
Products	Color	Product color
Products	Unit Cost USD	Cost to produce the product in USD
Products	Unit Price USD	Product list price in USD
Products	SubcategoryKey	Key to identify product subcategories
Products	Subcategory	Product subcategory name
Stores	StoreKey	Primary key to identify stores
Stores	Country	Store country
Stores	State	Store state
Stores	Square Meters	Store footprint in square meters
Stores	Open Date	Store open date
Exchange Rates	Date	Date of the exchange rate
Exchange Rates	Currency	Currency code
Exchange Rates	Exchange	Exchange rate compared to USD

BATCH ANALYSIS

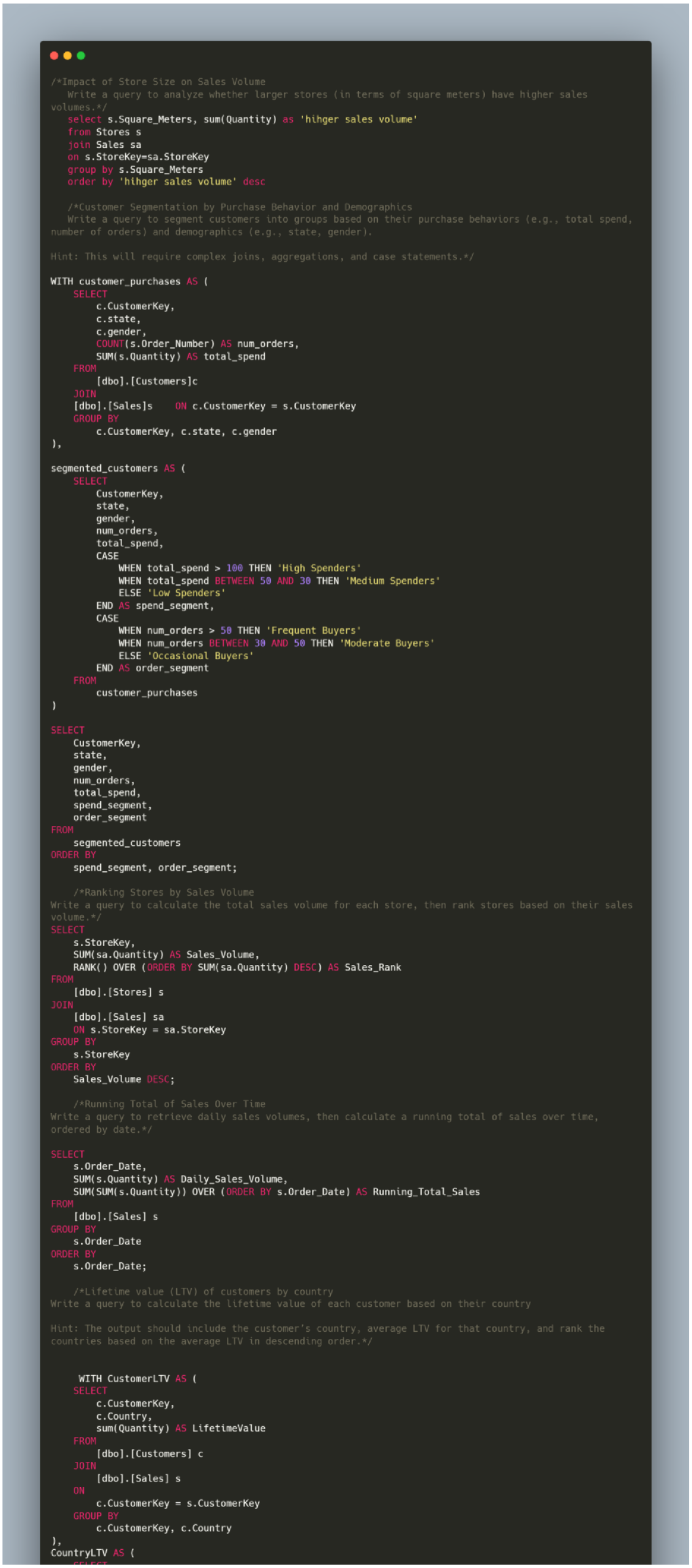
BATCH A: Example Questions and Queries

- **Question 1**: Identify the top 10 products by total sales revenue.
- **Query**:

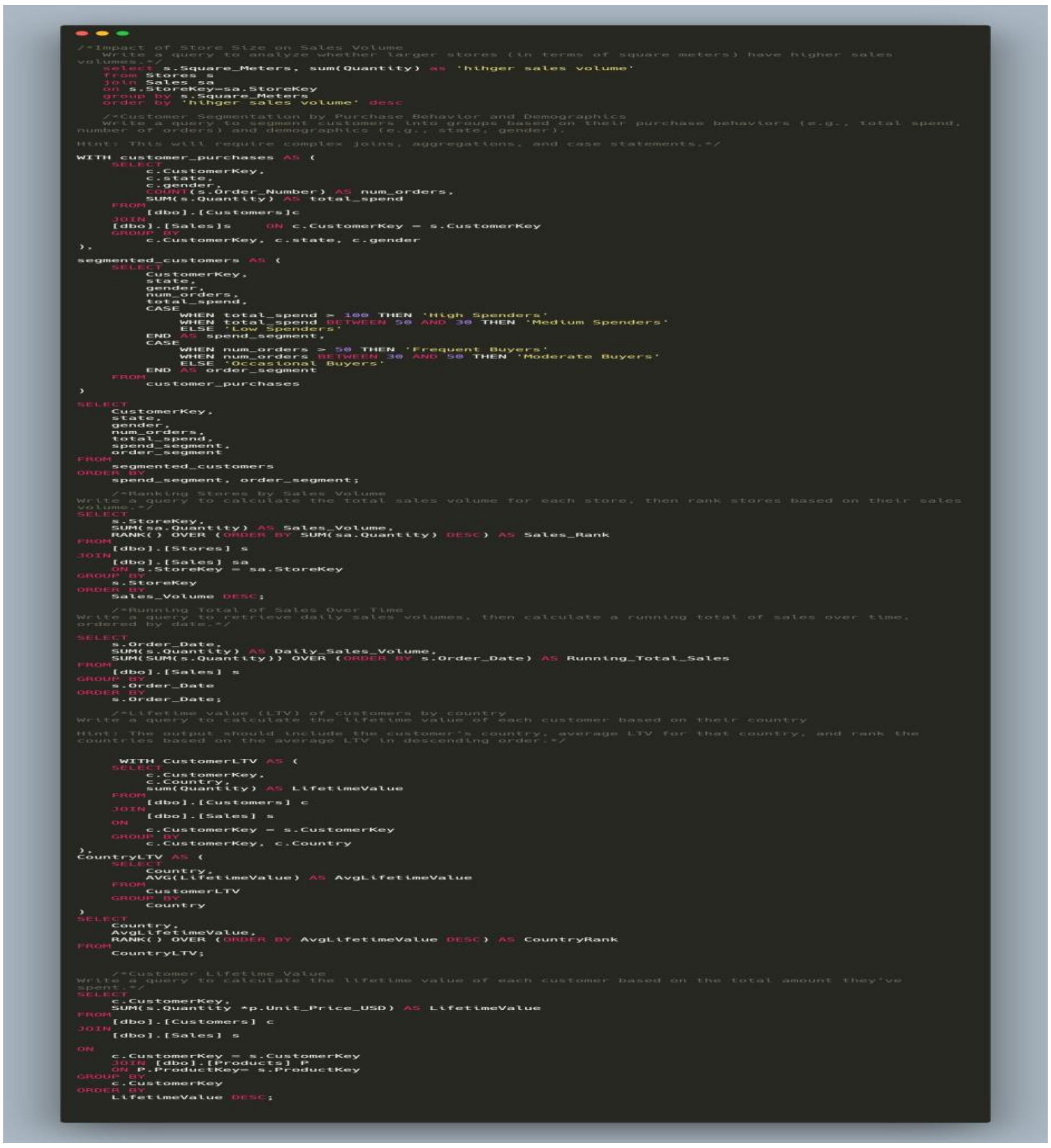


```sql
/*Count the Total Orders
Write a query to count the Total Number of Orders Per Customer order in desc
ANSWER*/

select Customers.Name, count(Line_Item) as total_number_of_order_per_customer
from [dbo].[Customers]
join Sales
on Customers.CustomerKey=Sales.CustomerKey
group by Name
order by total_number_of_order_per_customer desc

/*List of Products
Write a SQL query to List of Products Sold in 2020
ANSWER*/

SELECT Sales.Order_Date,(Product_Name) AS list_of_products_sold
FROM Sales
join Products
on Sales.ProductKey=Products.ProductKey
WHERE YEAR(Sales.Order_Date) = 2020
ORDER BY list_of_products_sold desc


/*Find Customers in a Specific City
Write a query to find all Customer Details from California (CA)
answer*/
select *
from [dbo].[Customers]
where City= 'California'


/*Calculate Total Sales Quantity
Write a query to calculate the Total Sales Quantity for product 2115
answer*/
select Sales.ProductKey, sum(quantity) as total_sales_quantity
from [dbo].[Sales]
where ProductKey= 2115
group by Sales.ProductKey
order by total_sales_quantity

/*Store Information Retrieval
Write a query to retrieve the Top 5 Stores with the Most Sales Transactions
answer*/

SELECT TOP 5 Sales.StoreKey, count(*) AS Total_Sales_Transactions
FROM [dbo].[Sales]
GROUP BY Sales.StoreKey
ORDER BY Total_Sales_Transactions DESC;
```

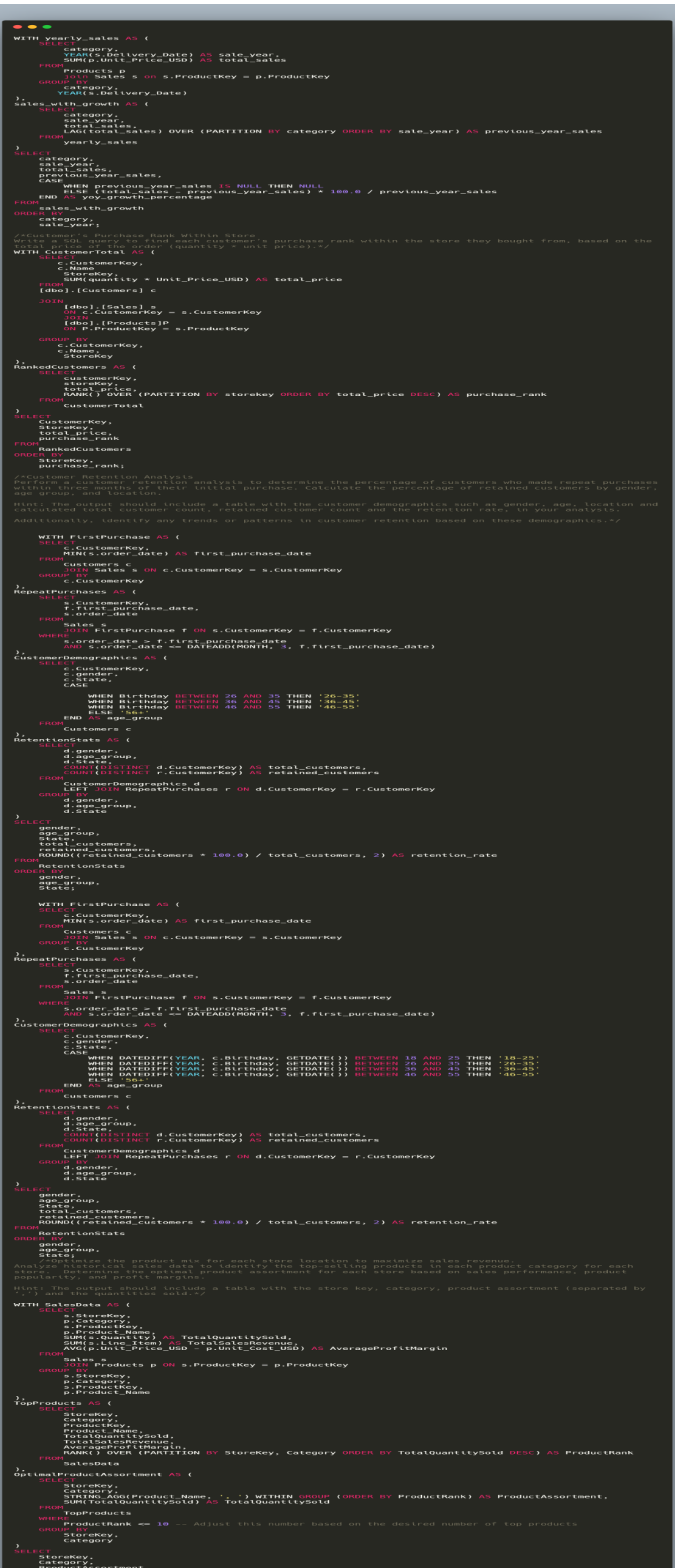
BATCH B



```sql
/*Impact of Store Size on Sales Volume
   Write a query to analyze whether larger stores (in terms of square meters) have higher sales
volumes.*/
   select s.Square_Meters, sum(Quantity) as 'hihger sales volume'
   from Stores s
   join Sales sa
   on s.StoreKey=sa.StoreKey
   group by s.Square_Meters
   order by 'hihger sales volume' desc

   /*Customer Segmentation by Purchase Behavior and Demographics
   Write a query to segment customers into groups based on their purchase behaviors (e.g., total spend,
number of orders) and demographics (e.g., state, gender).

Hint: This will require complex joins, aggregations, and case statements.*/

WITH customer_purchases AS (
    SELECT
        c.CustomerKey,
        c.state,
        c.gender,
        COUNT(s.Order_Number) AS num_orders,
        SUM(s.Quantity) AS total_spend
    FROM
        [dbo].[Customers]c
    JOIN
    [dbo].[Sales]s    ON c.CustomerKey = s.CustomerKey
    GROUP BY
        c.CustomerKey, c.state, c.gender
),

segmented_customers AS (
    SELECT
        CustomerKey,
        state,
        gender,
        num_orders,
        total_spend,
        CASE
            WHEN total_spend > 100 THEN 'High Spenders'
            WHEN total_spend BETWEEN 50 AND 30 THEN 'Medium Spenders'
            ELSE 'Low Spenders'
        END AS spend_segment,
        CASE
            WHEN num_orders > 50 THEN 'Frequent Buyers'
            WHEN num_orders BETWEEN 30 AND 50 THEN 'Moderate Buyers'
            ELSE 'Occasional Buyers'
        END AS order_segment
    FROM
        customer_purchases
)

SELECT
    CustomerKey,
    state,
    gender,
    num_orders,
    total_spend,
    spend_segment,
    order_segment
FROM
    segmented_customers
ORDER BY
    spend_segment, order_segment;

    /*Ranking Stores by Sales Volume
Write a query to calculate the total sales volume for each store, then rank stores based on their sales
volume.*/
SELECT
    s.StoreKey,
    SUM(sa.Quantity) AS Sales_Volume,
    RANK() OVER (ORDER BY SUM(sa.Quantity) DESC) AS Sales_Rank
FROM
    [dbo].[Stores] s
JOIN
    [dbo].[Sales] sa
    ON s.StoreKey = sa.StoreKey
GROUP BY
    s.StoreKey
ORDER BY
    Sales_Volume DESC;

    /*Running Total of Sales Over Time
Write a query to retrieve daily sales volumes, then calculate a running total of sales over time,
ordered by date.*/

SELECT
    s.Order_Date,
    SUM(s.Quantity) AS Daily_Sales_Volume,
    SUM(SUM(s.Quantity)) OVER (ORDER BY s.Order_Date) AS Running_Total_Sales
FROM
    [dbo].[Sales] s
GROUP BY
    s.Order_Date
ORDER BY
    s.Order_Date;

    /*Lifetime value (LTV) of customers by country
Write a query to calculate the lifetime value of each customer based on their country

Hint: The output should include the customer’s country, average LTV for that country, and rank the
countries based on the average LTV in descending order.*/


     WITH CustomerLTV AS (
    SELECT
        c.CustomerKey,
        c.Country,
        sum(Quantity) AS LifetimeValue
    FROM
        [dbo].[Customers] c
    JOIN
        [dbo].[Sales] s
    ON
        c.CustomerKey = s.CustomerKey
    GROUP BY
        c.CustomerKey, c.Country
),
CountryLTV AS (
```



```sql
/*Impact of Store Size on Sales Volume
    Write a query to analyze whether larger stores (in terms of square meters) have higher sales
volumes.*/
    select s.Square_Meters, sum(Quantity) as 'hihger sales volume'
    from Stores s
    join Sales sa
    on s.StoreKey=sa.StoreKey
    group by s.Square_Meters
    order by 'hihger sales volume' desc

    /*Customer Segmentation by Purchase Behavior and Demographics
    Write a query to segment customers into groups based on their purchase behaviors (e.g., total spend,
number of orders) and demographics (e.g., state, gender).

Hint: This will require complex joins, aggregations, and case statements.*/

WITH customer_purchases AS (
    SELECT
        c.CustomerKey,
        c.state,
        c.gender,
        COUNT(s.Order_Number) AS num_orders,
        SUM(s.Quantity) AS total_spend
    FROM
        [dbo].[Customers]c
    JOIN
    [dbo].[Sales]s     ON c.CustomerKey = s.CustomerKey
    GROUP BY
        c.CustomerKey, c.state, c.gender
),

segmented_customers AS (
    SELECT
        CustomerKey,
        state,
        gender,
        num_orders,
        total_spend,
        CASE
            WHEN total_spend > 100 THEN 'High Spenders'
            WHEN total_spend BETWEEN 50 AND 30 THEN 'Medium Spenders'
            ELSE 'Low Spenders'
        END AS spend_segment,
        CASE
            WHEN num_orders > 50 THEN 'Frequent Buyers'
            WHEN num_orders BETWEEN 30 AND 50 THEN 'Moderate Buyers'
            ELSE 'Occasional Buyers'
        END AS order_segment
    FROM
        customer_purchases
)

SELECT
    CustomerKey,
    state,
    gender,
    num_orders,
    total_spend,
    spend_segment,
    order_segment
FROM
    segmented_customers
ORDER BY
    spend_segment, order_segment;

    /*Ranking Stores by Sales Volume
Write a query to calculate the total sales volume for each store, then rank stores based on their sales
volume.*/
SELECT
    s.StoreKey,
    SUM(sa.Quantity) AS Sales_Volume,
    RANK() OVER (ORDER BY SUM(sa.Quantity) DESC) AS Sales_Rank
FROM
    [dbo].[Stores] s
JOIN
    [dbo].[Sales] sa
    ON s.StoreKey = sa.StoreKey
GROUP BY
    s.StoreKey
ORDER BY
    Sales_Volume DESC;

    /*Running Total of Sales Over Time
Write a query to retrieve daily sales volumes, then calculate a running total of sales over time,
ordered by date.*/

SELECT
    s.Order_Date,
    SUM(s.Quantity) AS Daily_Sales_Volume,
    SUM(SUM(s.Quantity)) OVER (ORDER BY s.Order_Date) AS Running_Total_Sales
FROM
    [dbo].[Sales] s
GROUP BY
    s.Order_Date
ORDER BY
    s.Order_Date;

    /*Lifetime value (LTV) of customers by country
Write a query to calculate the lifetime value of each customer based on their country

Hint: The output should include the customer's country, average LTV for that country, and rank the
countries based on the average LTV in descending order.*/


     WITH CustomerLTV AS (
    SELECT
        c.CustomerKey,
        c.Country,
        sum(Quantity) AS LifetimeValue
    FROM
        [dbo].[Customers] c
    JOIN
        [dbo].[Sales] s
    ON
        c.CustomerKey = s.CustomerKey
    GROUP BY
        c.CustomerKey, c.Country
),
CountryLTV AS (
    SELECT
        Country,
        AVG(LifetimeValue) AS AvgLifetimeValue
    FROM
        CustomerLTV
    GROUP BY
        Country
)
SELECT
    Country,
    AvgLifetimeValue,
    RANK() OVER (ORDER BY AvgLifetimeValue DESC) AS CountryRank
FROM
    CountryLTV;


    /*Customer Lifetime Value
Write a query to calculate the lifetime value of each customer based on the total amount they've
spent.*/
SELECT
    c.CustomerKey,
    SUM(s.Quantity *p.Unit_Price_USD) AS LifetimeValue
FROM
    [dbo].[Customers] c
JOIN
    [dbo].[Sales] s

ON
    c.CustomerKey = s.CustomerKey
    JOIN [dbo].[Products] P
    ON P.ProductKey= s.ProductKey
GROUP BY
    c.CustomerKey
ORDER BY
    LifetimeValue DESC;
```



```sql
WITH yearly_sales AS (
    SELECT
        category,
        YEAR(s.Delivery_Date) AS sale_year,
        SUM(p.Unit_Price_USD) AS total_sales
    FROM
        Products p
        join Sales s on s.ProductKey = p.ProductKey
    GROUP BY
        category,
       YEAR(s.Delivery_Date)
),
sales_with_growth AS (
    SELECT
        category,
        sale_year,
        total_sales,
        LAG(total_sales) OVER (PARTITION BY category ORDER BY sale_year) AS previous_year_sales
    FROM
        yearly_sales
)
SELECT
    category,
    sale_year,
    total_sales,
    previous_year_sales,
    CASE
        WHEN previous_year_sales IS NULL THEN NULL
        ELSE (total_sales - previous_year_sales) * 100.0 / previous_year_sales
    END AS yoy_growth_percentage
FROM
    sales_with_growth
ORDER BY
    category,
    sale_year;

/*Customer's Purchase Rank Within Store
Write a SQL query to find each customer's purchase rank within the store they bought from, based on the
total price of the order (quantity * unit price).*/
WITH CustomerTotal AS (
    SELECT
       c.CustomerKey,
       c.Name
        StoreKey,
        SUM(quantity * Unit_Price_USD) AS total_price
    FROM
    [dbo].[Customers] c

    JOIN
        [dbo].[Sales] s
        ON c.CustomerKey = s.CustomerKey
        JOIN
        [dbo].[Products]P
        ON P.ProductKey = s.ProductKey

    GROUP BY
       c.CustomerKey,
       c.Name,
        StoreKey
),
RankedCustomers AS (
    SELECT
        customerKey,
        storeKey,
        total_price,
        RANK() OVER (PARTITION BY storekey ORDER BY total_price DESC) AS purchase_rank
    FROM
        CustomerTotal
)
SELECT
    CustomerKey,
    StoreKey,
    total_price,
    purchase_rank
FROM
    RankedCustomers
ORDER BY
    StoreKey,
    purchase_rank;

/*Customer Retention Analysis
Perform a customer retention analysis to determine the percentage of customers who made repeat purchases
within three months of their initial purchase. Calculate the percentage of retained customers by gender,
age group, and location.

Hint: The output should include a table with the customer demographics such as gender, age, location and
calculated total customer count, retained customer count and the retention rate, in your analysis.

Additionally, identify any trends or patterns in customer retention based on these demographics.*/


    WITH FirstPurchase AS (
    SELECT
        c.CustomerKey,
        MIN(s.order_date) AS first_purchase_date
    FROM
        Customers c
        JOIN Sales s ON c.CustomerKey = s.CustomerKey
    GROUP BY
        c.CustomerKey
),
RepeatPurchases AS (
    SELECT
        s.CustomerKey,
        f.first_purchase_date,
        s.order_date
    FROM
        Sales s
        JOIN FirstPurchase f ON s.CustomerKey = f.CustomerKey
    WHERE
        s.order_date > f.first_purchase_date
        AND s.order_date <= DATEADD(MONTH, 3, f.first_purchase_date)
),
CustomerDemographics AS (
    SELECT
        c.CustomerKey,
        c.gender,
        c.State,
        CASE

            WHEN Birthday BETWEEN 26 AND 35 THEN '26-35'
            WHEN Birthday BETWEEN 36 AND 45 THEN '36-45'
            WHEN Birthday BETWEEN 46 AND 55 THEN '46-55'
            ELSE '56+'
        END AS age_group
    FROM
        Customers c
),
RetentionStats AS (
    SELECT
        d.gender,
        d.age_group,
        d.State,
        COUNT(DISTINCT d.CustomerKey) AS total_customers,
        COUNT(DISTINCT r.CustomerKey) AS retained_customers
    FROM
        CustomerDemographics d
        LEFT JOIN RepeatPurchases r ON d.CustomerKey = r.CustomerKey
    GROUP BY
        d.gender,
        d.age_group,
        d.State
)
SELECT
    gender,
    age_group,
    State,
    total_customers,
    retained_customers,
    ROUND((retained_customers * 100.0) / total_customers, 2) AS retention_rate
FROM
    RetentionStats
ORDER BY
    gender,
    age_group,
    State;


    WITH FirstPurchase AS (
    SELECT
        c.CustomerKey,
        MIN(s.order_date) AS first_purchase_date
    FROM
        Customers c
        JOIN Sales s ON c.CustomerKey = s.CustomerKey
    GROUP BY
        c.CustomerKey
),
RepeatPurchases AS (
    SELECT
        s.CustomerKey,
        f.first_purchase_date,
        s.order_date
    FROM
        Sales s
        JOIN FirstPurchase f ON s.CustomerKey = f.CustomerKey
    WHERE
        s.order_date > f.first_purchase_date
        AND s.order_date <= DATEADD(MONTH, 3, f.first_purchase_date)
),
CustomerDemographics AS (
    SELECT
        c.CustomerKey,
        c.gender,
        c.State,
        CASE
            WHEN DATEDIFF(YEAR, c.Birthday, GETDATE()) BETWEEN 18 AND 25 THEN '18-25'
            WHEN DATEDIFF(YEAR, c.Birthday, GETDATE()) BETWEEN 26 AND 35 THEN '26-35'
            WHEN DATEDIFF(YEAR, c.Birthday, GETDATE()) BETWEEN 36 AND 45 THEN '36-45'
            WHEN DATEDIFF(YEAR, c.Birthday, GETDATE()) BETWEEN 46 AND 55 THEN '46-55'
            ELSE '56+'
        END AS age_group
    FROM
        Customers c
),
RetentionStats AS (
    SELECT
        d.gender,
        d.age_group,
        d.State,
        COUNT(DISTINCT d.CustomerKey) AS total_customers,
        COUNT(DISTINCT r.CustomerKey) AS retained_customers
    FROM
        CustomerDemographics d
        LEFT JOIN RepeatPurchases r ON d.CustomerKey = r.CustomerKey
    GROUP BY
        d.gender,
        d.age_group,
        d.State
)
SELECT
    gender,
    age_group,
    State,
    total_customers,
    retained_customers,
    ROUND((retained_customers * 100.0) / total_customers, 2) AS retention_rate
FROM
    RetentionStats
ORDER BY
    gender,
    age_group,
    State;
    /*Optimize the product mix for each store location to maximize sales revenue.
Analyze historical sales data to identify the top-selling products in each product category for each
store.  Determine the optimal product assortment for each store based on sales performance, product
popularity, and profit margins.

Hint: The output should include a table with the store key, category, product assortment (separated by
',') and the quantities sold.*/

WITH SalesData AS (
    SELECT
        s.StoreKey,
        p.Category,
        s.ProductKey,
        p.Product_Name,
        SUM(s.Quantity) AS TotalQuantitySold,
        SUM(s.Line_Item) AS TotalSalesRevenue,
        AVG(p.Unit_Price_USD - p.Unit_Cost_USD) AS AverageProfitMargin
    FROM
        Sales s
        JOIN Products p ON s.ProductKey = p.ProductKey
    GROUP BY
        s.StoreKey,
        p.Category,
        s.ProductKey,
        p.Product_Name
),
TopProducts AS (
    SELECT
        StoreKey,
        Category,
        ProductKey,
        Product_Name,
        TotalQuantitySold,
        TotalSalesRevenue,
        AverageProfitMargin,
        RANK() OVER (PARTITION BY StoreKey, Category ORDER BY TotalQuantitySold DESC) AS ProductRank
    FROM
        SalesData
),
OptimalProductAssortment AS (
    SELECT
        StoreKey,
        Category,
        STRING_AGG(Product_Name, ', ') WITHIN GROUP (ORDER BY ProductRank) AS ProductAssortment,
        SUM(TotalQuantitySold) AS TotalQuantitySold
    FROM
        TopProducts
    WHERE
        ProductRank <= 10 -- Adjust this number based on the desired number of top products
    GROUP BY
        StoreKey,
        Category
)
SELECT
    StoreKey,
    Category,
    ProductAssortment
```

RECOMMENDATIONS

1. Enhance Product Assortment Based on Sales Data

- **Top-Selling Products**: Focus on stocking the top-selling products identified in each category for each store to ensure popular products are readily available.
- **Optimize Inventory**: Regularly review and update the product assortment based on ongoing sales data to maintain optimal inventory levels and reduce stockouts or overstock situations.

2. Improve Customer Retention Strategies

- **Targeted Marketing**: Use retention rate data by gender and age group to create targeted marketing campaigns. Tailor marketing efforts to further engage segments with higher retention rates.
- **Loyalty Programs**: Implement loyalty programs or incentives to encourage repeat purchases, especially within the first three months of the initial purchase.

3. Leverage Demographic Insights

- **Personalized Offers**: Use demographic data to offer personalized discounts or product recommendations based on customer preferences and purchasing behavior.
- **Location-Based Strategies**: Develop location-specific strategies based on the performance of products in different stores. Consider expanding the availability or promoting products that perform well in specific stores.

4. Maximize Profit Margins

- **Product Pricing**: Review the average profit margins of products and adjust pricing strategies to maximize profitability. Highlight and market high-margin products prominently.
- **Cost Management**: Monitor the cost of goods sold and identify opportunities to negotiate better prices with suppliers or reduce production costs without compromising quality.

5. Data-Driven Decision Making

- **Continuous Analysis**: Regularly perform similar analyses to stay updated on sales trends, customer behavior, and product performance. Use these insights to make informed business decisions.
- **Invest in Analytics**: Invest in advanced analytics tools and technologies to enhance data collection, analysis, and reporting capabilities for more precise and actionable insights.

6. Enhance Store Performance

- **Store Layout Optimization**: Optimize the store layout to highlight top-performing products and categories, making them more accessible and attractive to customers.

- **Staff Training**: Train store staff to understand the product mix and effectively promote top-selling products, improving the overall customer experience and increasing sales.

Conclusion

By implementing these recommendations, the business can enhance its product offerings, improve customer retention, maximize profit margins, and make data-driven decisions to optimize overall performance. Continuous monitoring and analysis will ensure that strategies remain effective and adapt to changing market conditions.