TASK 01: Walmart wants to identify which branch has exhibited the highest sales growth over time. Analyse the total sales for each branch and compare the growth rate across months to find the top performer.

```
WITH MonthlySales AS (
SELECT Branch, DATE_FORMAT(Date, '%Y-%m') AS Month,
ROUND(SUM(Total),2) AS Sale_by_month
FROM walmartsales
GROUP BY Branch, Month
ORDER BY Month, Branch), prev_sale AS(
SELECT Branch, Month, Sale_by_month,
LAG(Sale_by_month) OVER (PARTITION BY Branch ORDER BY Month) AS Previous_Sales
FROM monthlysales),
growth_rate AS(
SELECT Branch, Month, Sale_by_month, Previous_sales,
ROUND(((Sale_by_month - Previous_sales)/ Previous_sales) * 100,2) AS mon_Growth_Rate
FROM prev_sale)
SELECT Branch,
ROUND(AVG(mon_growth_rate),2) as Average_growth_rate
FROM growth_rate
WHERE mon_growth_rate is NOT NULL
GROUP BY Branch
ORDER BY Average_growth_rate DESC
LIMIT 1;
```

TASK 02: Walmart needs to determine which product line contributes the highest profit to each branch. The profit margin should be calculated based on the difference between the gross income and cost of goods sold.

```
WITH ProductProfit AS (
SELECT Branch, product_line,
round(SUM(gross_income),2) AS Total_Profit
FROM walmartsales
GROUP BY Branch, product_line),
RankedProducts AS (
SELECT Branch, product_line, Total_Profit,
RANK() OVER (PARTITION BY Branch ORDER BY Total_Profit DESC) AS prod_rank
FROM ProductProfit)
SELECT Branch, product_line, Total_Profit
FROM RankedProducts
WHERE prod_rank = 1;
```

TASK 03: Walmart wants to segment customers based on their average spending behavior. Classify customers into three tiers: High, Medium, and Low spenders based on their total purchase amounts.

```
WITH spending_cte AS(
SELECT customer_id, Round(SUM(Total),4) as total_spending
FROM walmartsales
GROUP BY customer_id
ORDER BY total_spending DESC),
Percentile_cte AS (
SELECT customer_id, total_spending,
NTILE(3) OVER (ORDER BY Total_Spending DESC) AS Spending_Tier
FROM spending_cte)SELECT customer_id, total_spending,
CASE
WHEN spending_tier =1 THEN "High"
WHEN spending_tier =2 THEN "Medium"
ELSE "Low"
END AS Customer_Class
FROM percentile_cte;
```

TASK 04: Walmart suspects that some transactions have unusually high or low sales compared to the average for the product line. Identify these anomalies.

```
WITH ProductStats AS (
SELECT product_line,
AVG(Total) AS Avg_Sales, STDDEV(Total) AS Std_Dev
FROM walmartsales
GROUP BY product_line ),
Anomalies AS (
SELECT w.Invoice_ID, w.Branch, w.product_line, w.Total,
p.Avg_Sales, p.Std_Dev,
(w.Total - p.Avg_Sales) / p.Std_Dev AS Z_Score
FROM walmartsales AS w
JOIN ProductStats AS p
ON w.product_line = p.product_line )
SELECT *
FROM Anomalies
WHERE ABS(Z_Score) > 3
ORDER BY Z_Score DESC;
```

TASK 05: Walmart needs to determine the most popular payment method in each city to tailor marketing strategies.

```
WITH payment_frequency_cte AS (
SELECT City, Payment,
count(invoice_id) as frequency_method
FROM walmartsales
GROUP BY city, Payment),
frequency_rank_cte AS (
SELECT City, Payment, frequency_method,
RANK() OVER(PARTITION BY City ORDER BY frequency_method DESC) as pay_rank
FROM payment_frequency_cte)
SELECT City, Payment, pay_rank
FROM frequency_rank_cte
WHERE pay_rank =1;
```

TASK 06: Walmart wants to understand the sales distribution between male and female customers on a monthly basis.

```
With men_sales_cte AS(
SELECT DATE_FORMAT(Date, '%Y-%m') AS Month,
ROUND(sum(total),2) AS Men_sales
FROM Walmartsales
WHERE gender="Male"
GROUP BY Gender, Month),
Female_sales_cte AS(
SELECT DATE_FORMAT(Date, '%Y-%m') AS Month,
ROUND(sum(total),2) AS female_sales
FROM Walmartsales
WHERE gender="Female"
GROUP BY Gender, Month)
SELECT m.Month, m.Men_sales, f.Female_Sales
FROM men_sales_cte as m
JOIN female_Sales_cte as f
ON m.Month=f.Month
ORDER BY m.Month;
```

TASK 07: Walmart wants to know which product lines are preferred by different customer types(Member vs. Normal).

```
WITH sales_cte AS(
SELECT Customer_type, Product_line,
ROUND(Sum(total) ,2) as Sales
FROM walmartsales
GROUP BY Customer_type, Product_line),
rank_cte AS (
SELECT Customer_type, Product_line, Sales,
RANK() OVER (PARTITION BY Customer_type ORDER BY Sales DESC) as Sales_rank
FROM Sales_cte)
SELECT *
FROM rank_cte
WHERE Sales_rank =1;
```

TASK 08: Walmart needs to identify customers who made repeat purchases within a specific time frame (e.g., within 30 days).

```
SELECT Customer_ID,
COUNT(Invoice_ID) AS Purchase_Count,
MIN(date) AS First_Purchase_Date,
MAX(date) AS Last_Purchase_Date
FROM walmartsales
WHERE date BETWEEN DATE_SUB('2019-03-31', INTERVAL 30 DAY) AND '2019-03-31’
GROUP BY Customer_ID
HAVING COUNT(Invoice_ID) > 1;
```

TASK 09: Walmart wants to reward its top 5 customers who have generated the most sales Revenue.

```
WITH Sales_cte AS (
SELECT Customer_ID,
ROUND(Sum(Total),2) AS Sales
FROM walmartsales
GROUP BY Customer_ID),
rank_cte AS(
SELECT Customer_ID, Sales,
RANK() OVER (ORDER BY Sales DESC) as Sales_rank
FROM Sales_cte)
SELECT * FROM rank_cte
WHERE Sales_rank<=5;
```

TASK 10: Walmart wants to analyze the sales patterns to determine which day of the week brings the highest sales.

```
SELECT
ROUND(SUM(total),2) as Sales,
DAYNAME(Date) AS Weekday
FROM walmartsales
GROUP BY Weekday
ORDER BY Sales DESC
LIMIT 1;
```