DATA ANALYSIS SQL QUERIES REPORT

Sales Performance Analysis

1. Calculate total sales revenue by product category in the last 3 months.

```
SELECT
    i.i_category AS product_category,
    SUM(ss.ss_sales_price) AS total_sales_revenue
FROM
    store_sales ss
JOIN
    item i ON ss.ss_item_sk = i.i_item_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-04-01' AND '1999-06-30'
GROUP BY
    i.i_category;
```

Explanation:

- Formula: `Total Sales Revenue = SUM(quantity_sold * price)`
- `DATEADD(MONTH, -3, GETDATE())` filters data for the last 3 months.
- `GROUP BY product_category` groups the sales by product category.
- Purpose: This query helps identify how much revenue each product category has generated in the last 3 months.
- **Analysis**: By grouping sales by product category, this analysis reveals which product categories are generating the most revenue. Mega Mart can use this information to prioritize stocking high-revenue categories and optimizing marketing efforts.
- **Output :-**

PRODUCT	TOTAL_SALES_REVENUE
Women	2914117519
Sports	2961037314
Children	2939718064
Jewelry	2898416543
Electronic:	2932688352

2. Show monthly sales trends over the last 6 months.

```
SELECT
    EXTRACT(MONTH FROM d.d_date) AS month,
    SUM(ss.ss_sales_price) AS monthly_sales
FROM
    store_sales ss
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30'
GROUP BY
    EXTRACT(MONTH FROM d.d_date);
```

Explanation:

- Formula: `Total Sales = SUM(quantity_sold * price)`
- Purpose: To observe how sales fluctuate month by month in the last 6 months. The data is grouped by year and month
- **Analysis**: Identifying peaks and troughs in sales trends helps forecast demand and align promotional activities to months where sales typically rise or fall. This assists in managing inventory more efficiently.
- **OUTPUT :-**

MONTH	MONTHLY_SALES
5	9990673003
4	9655408706
2	9022274077
1	11881226883
3	9663021445
6	9664699131

3. Identify the top 5 best-selling products by total revenue in the last year.

```
SELECT
    i.i_item_desc AS product,
    SUM(ss.ss_sales_price) AS total_sales_revenue
FROM
    store_sales ss
```

```
JOIN
    item i ON ss.ss_item_sk = i.i_item_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
GROUP BY
    i.i_item_desc
ORDER BY
    total_sales_revenue DESC
LIMIT 5;
```

Explanation:

- **Formula**: `Total Revenue = SUM(quantity_sold * price)`
- **Purpose**: This query ranks the products based on their total sales revenue over the last year and returns the top 5.
- **Analysis**: Highlighting best-sellers helps Mega Mart understand product popularity and make informed decisions about inventory levels, promotions, and product recommendations to customers.
- **OUTPUT :-**

PRODUCT	TOTAL_SALES_REVENUE
	496551256.2
callyn stn stesebarought	1096774.53
eseeseeingoughtn stable	1095092.67
ableationoughteseantipri	1094831.06
ablen stpriantiationable	1094523.78

4. Calculate total sales revenue by region for the last quarter.

```
SELECT
    ca.ca_state AS region,
    SUM(ss.ss_sales_price) AS total_sales_revenue
FROM
    store_sales ss
JOIN
    customer_address ca ON ss.ss_addr_sk = ca.ca_address_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
```

```
    d.d_date BETWEEN '1999-01-01' AND '1999-03-31'
GROUP BY
    ca.ca_state;
```

Explanation:

- **Formula**: `Total Sales Revenue = SUM(quantity_sold * price)`
- **Purpose**: This query calculates the total revenue by region over the last 3 months (1 quarter).
- **Analysis**: This helps Mega Mart identify strong and weak regions in terms of sales. The company can focus marketing efforts in underperforming areas and optimize logistics for high-performing regions.
- **OUTPUT :-**

REGION	TOTAL_SALES_REVENUE	REGION	TOTAL_SALES_REVENUE
FL	605971255	WV	518998560.4
WA	367337741.4	IA	931149761.8
ME	150402246.9	OH	827869783.6
MT	528982734.5	MD	216022012.1
WI	670686324.1	NV	159620888

5. Compare month-over-month sales growth for the current year.

```
SELECT
    EXTRACT(MONTH FROM d.d_date) AS month,
    SUM(ss.ss_sales_price) AS total_sales_revenue,
    LAG(SUM(ss.ss_sales_price)) OVER (ORDER BY EXTRACT(MONTH FROM
d.d_date)) AS previous_month_sales,
    (SUM(ss.ss_sales_price) - LAG(SUM(ss.ss_sales_price)) OVER (ORDER BY
EXTRACT(MONTH FROM d.d_date))) / LAG(SUM(ss.ss_sales_price)) OVER (ORDER BY
EXTRACT(MONTH FROM d.d_date)) * 100 AS growth_percentage
FROM
    store_sales ss
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
GROUP BY
```

```
    EXTRACT(MONTH FROM d.d_date);
```

Explanation:

- **Formula**: `Month-over-Month Growth = ((Current Month Sales - Previous Month Sales) / Previous Month Sales) * 100`
- **Purpose**: It calculates the percentage change in sales from one month to the next in the current year.
- **Analysis**: This tracks how sales are changing month to month, helping Mega Mart assess its growth rate and pinpoint factors influencing sales, such as seasonality or promotional efforts.
- **OUTPUT :-**

MONTH	TOTAL_SALES_REVENUE	PREVIOUS_MONTH_SALES	GROWTH_PERCENTAGE
1	11881226883		
2	9022274077	11881226883	-24.062774
3	9663021445	9022274077	7.101839
4	9655408706	9663021445	-0.078782
5	9990673003	9655408706	3.472295
6	9664699131	9990673003	-3.262782

6. Calculate sales by channel (store, catalog, online) in the last 6 months.

```
SELECT
    'Store' AS channel,
    SUM(ss.ss_sales_price) AS total_sales
FROM
    store_sales ss
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30'

UNION ALL

SELECT
    'Catalog',
    SUM(cs.cs_sales_price)
```

```
FROM
    catalog_sales cs
JOIN
    date_dim d ON cs.cs_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30'

UNION ALL

SELECT
    'Online',
    SUM(ws.ws_sales_price)
FROM
    web_sales ws
JOIN
    date_dim d ON ws.ws_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30';
```

Explanation:

- **Formula**: `Total Sales = SUM(quantity_sold * price)`
- **Purpose**: This query calculates total sales for each sales channel over the last 6 months.
- **Analysis**: Comparing channel performance helps Mega Mart allocate resources appropriately, invest in successful channels, and identify opportunities for improving underperforming channels.
- OUTPUT

CHANNEL	TOTAL_SALES
Store	59877303246
Catalog	41125819143
Online	21384984967

7. Analyze the sales performance of new products introduced in the last 6 months.

```
SELECT
    i.i_item_desc AS product,
    SUM(ss.ss_sales_price) AS total_sales_revenue
```

```
FROM
    store_sales ss
JOIN
    item i ON ss.ss_item_sk = i.i_item_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30'
GROUP BY
    i.i_item_desc;
```

Explanation:

- **Formula**: `Total Sales = SUM(quantity_sold * price)` for products introduced in the last 6 months.
- **Purpose**: This query evaluates how well newly launched products have been performing in terms of sales.
- **Analysis**: Monitoring the performance of new products helps evaluate the success of recent product launches. It enables decisions on whether to continue promoting, adjusting stock levels, or discontinuing a product.
- **OUTPUT :-**

PRODUCT	TOTAL_SALES_REVENUE
Grounds get flowers. Poorly clear prisoners would not detect numbers. Techniques get here. Times look	296153.34
Ways should calculate recent projects. French, red events discriminate. Other, fresh weeks woul	297798.71
Jewish members participate very particular,	298408.44
Sides let; etc eastern hours discuss particularly at a concentrations. Individual, closed matters pay ver	300164.29
Over good sites try more single forms. Able pains can see all rough, correct letters; even equivalent mo	296756.12
Programmes demonstrate certainly events; free, other readers continue; ind	304650.97

8. Calculate the average order value per channel in the last 6 months.

```
SELECT
    'Store' AS channel,
    AVG(ss.ss_net_paid) AS avg_order_value
FROM
    store_sales ss
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30'

UNION ALL
```

```sql
SELECT
    'Catalog',
    AVG(cs.cs_net_paid)
FROM
    catalog_sales cs
JOIN
    date_dim d ON cs.cs_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30'

UNION ALL

SELECT
    'Online',
    AVG(ws.ws_net_paid)
FROM
    web_sales ws
JOIN
    date_dim d ON ws.ws_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30';
```

Explanation:

- **Formula**: `Average Order Value = SUM(order_total) / COUNT(orders)`
- **Purpose**: To understand customer spending behavior across different sales channels.
- **Analysis**: AOV provides insights into customer spending patterns across different channels. Mega Mart can use this information to tailor upselling and cross-selling strategies to boost AOV.
- **OUTPUT :-**

CHANNEL	AVG_ORDER_VALUE
Store	1721.377767
Catalog	2294.962232
Online	2294.977241

9. Compare sales for the Summer and Winter seasons.

```sql
SELECT
    CASE
        WHEN d.d_moy IN (6, 7, 8) THEN 'Summer'
        WHEN d.d_moy IN (12, 1, 2) THEN 'Winter'
        ELSE 'Other'
    END AS season,
    SUM(ss.ss_ext_sales_price) AS total_sales_revenue
FROM
    store_sales ss
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
    AND d.d_moy IN (6, 7, 8, 12, 1, 2)
GROUP BY
    CASE
        WHEN d.d_moy IN (6, 7, 8) THEN 'Summer'
        WHEN d.d_moy IN (12, 1, 2) THEN 'Winter'
        ELSE 'Other'
    END;
```

Explanation:

- **Formula**: `Total Sales = SUM(quantity_sold * price)` grouped by Summer and Winter seasons.
- **Purpose**: This query compares the revenue generated during the Summer and Winter seasons.
- **Analysis**: This seasonal analysis helps Mega Mart optimize inventory and promotions to take advantage of seasonal trends. Understanding seasonality can drive more effective marketing and stock replenishment.
- **OUTPUT :-**

SEASON	TOTAL_SALES_REVENUE
Winter	2.7939E+12
Summer	2.09831E+12

10. Identify the top 3 categories by total sales in the last year.

```sql
SELECT
```

```
    i.i_category AS product_category,
    SUM(ss.ss_sales_price) AS total_sales_revenue
FROM
    store_sales ss
JOIN
    item i ON ss.ss_item_sk = i.i_item_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
GROUP BY
    i.i_category
ORDER BY
    total_sales_revenue DESC
LIMIT 3;
```

Explanation:

- **Formula**: `Total Sales = SUM(quantity_sold * price)`
- **Purpose**: To identify the top 3 product categories that generated the most revenue in the last 12 months.
- **Analysis**: Focusing on the top categories ensures resources are allocated efficiently. Mega Mart can use these insights to refine product strategy and increase profitability.
- **OUTPUT :-**

PRODUCT_CATEGORY	TOTAL_SALES_REVENUE
Sports	20550377458
Music	20463501047
Children	20405212166

Inventory Management Analysis

1. Calculate inventory turnover ratio by product category in the last 6 months.

```
SELECT
    i.i_category AS product_category,
```

```
    SUM(ss.ss_quantity * i.i_wholesale_cost) /
AVG(inv.inv_quantity_on_hand) AS turnover_ratio
FROM
    store_sales ss
JOIN
    item i ON ss.ss_item_sk = i.i_item_sk
JOIN
    inventory inv ON i.i_item_sk = inv.inv_item_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30'
GROUP BY
    i.i_category;
```

Explanation:

- **Formula**: `Inventory Turnover Ratio = Cost of Goods Sold (COGS) / Average Inventory`
- **Purpose**: To measure how often inventory is sold and replaced, helping to identify fast-moving versus slow-moving product categories.
- **Analysis**: A high turnover ratio indicates strong sales and efficient inventory management. Low ratios may signal overstocking or slow-moving products, helping Mega Mart optimize stock levels.

2. Identify the products with the highest stockout rates in the last month.

```
SELECT
    i.i_item_desc AS product,
    COUNT(*) AS stockout_count
FROM
    inventory inv
JOIN
    item i ON inv.inv_item_sk = i.i_item_sk
JOIN
    date_dim d ON inv.inv_date_sk = d.d_date_sk
WHERE
    inv.inv_quantity_on_hand = 0
    AND d.d_date BETWEEN '1999-08-01' AND '1999-08-31'
GROUP BY
```

```
    i.i_item_desc
ORDER BY
    stockout_count DESC
LIMIT 5;
```

Explanation:

- **Formula**: `Stockout Rate = (Number of Stockouts / Total Sales Attempts) * 100`
- **Purpose**: To identify products frequently out of stock, helping to adjust replenishment strategies and reduce missed sales.
- **Analysis**: Reducing stockouts of high-demand products ensures customer satisfaction and prevents lost sales. Mega Mart can adjust inventory replenishment strategies based on this data.
- **OUTPUT :-**

PRODUCT	STOCKOUT_COUNT
	48
O	9
M	7
L	7
P	6

3. List the top 5 overstocked products in the last quarter.

```
SELECT
    i.i_item_desc AS product,
    AVG(inv.inv_quantity_on_hand) AS avg_stock
FROM
    inventory inv
JOIN
    item i ON inv.inv_item_sk = i.i_item_sk
JOIN
    date_dim d ON inv.inv_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-06-01' AND '1999-08-31'
GROUP BY
    i.i_item_desc
ORDER BY
    avg_stock DESC
```

```
LIMIT 5;
```

Explanation:

- **Formula**: `(No specific formula, determined by comparing inventory levels to sales rates)`
- **Purpose**: To identify products with excessive inventory and slow sales, helping to reduce holding costs and avoid obsolescence.
- **Analysis**: Overstocking ties up capital and increases holding costs. This analysis helps Mega Mart identify slow-moving inventory and take corrective actions, such as running promotions to clear excess stock.
- **OUTPUT :-**

PRODUCT	AVG_STOCK
Nearly industrial talks cool; then large cups may app	572.641694
Further great sons see once again. Meanings used t	571.83871
Relationships will deal again slim, broad years. Cons	571.786408
Elsewhere white numbers used to get only a bit depe	568.787781
Even long-term hands may not say. Future ministers	567.856187

4. Calculate the average days of inventory on hand for each product category.

```
SELECT
    i.i_category AS product_category,
    (AVG(inv.inv_quantity_on_hand) / SUM(ss.ss_quantity *
i.i_wholesale_cost)) * 365 AS avg_days_on_hand
FROM
    store_sales ss
JOIN
    item i ON ss.ss_item_sk = i.i_item_sk
JOIN
    inventory inv ON i.i_item_sk = inv.inv_item_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-04-01' AND '1999-09-30'
GROUP BY
    i.i_category;
```

Explanation:

- **Formula**: `Days of Inventory on Hand = Current Inventory / Daily Cost of Goods Sold (COGS)`
- **Purpose**: To measure how long inventory stays in stock before being sold, indicating inventory efficiency.
- **Analysis**: This helps Mega Mart assess how long stock lasts before it needs replenishment. Categories with high inventory days may require reevaluation of purchasing strategies to avoid overstocking.
- **OUTPUT :-**

I_CATEGORY	AVG_QUANTITY
Books	499.971424
Sports	499.988759
Jewelry	499.964305
Women	499.969638
Home	500.055396

5. Determine replenishment frequency for high-demand products in the last 2 months.

```sql
SELECT
    i.i_item_desc AS product,
    COUNT(DISTINCT inv.inv_date_sk) AS replenishment_count
FROM
    inventory inv
JOIN
    item i ON inv.inv_item_sk = i.i_item_sk
JOIN
    date_dim d ON inv.inv_date_sk = d.d_date_sk
WHERE
    inv.inv_quantity_on_hand > 0
    AND d.d_date BETWEEN '1999-07-01' AND '1999-08-31'
GROUP BY
    i.i_item_desc
ORDER BY
    replenishment_count DESC;
```

Explanation:

- **Formula**: `(No specific formula, determined by tracking the number of restocks)`
- **Purpose**: To determine how frequently high-demand products are restocked, ensuring consistent availability.
- **Analysis**: Understanding replenishment frequency helps optimize stock levels for popular products, ensuring availability without overstocking, and improving overall supply chain efficiency.
- **OUTPUT :-**

PRODUCT	REPLENISHMENT_COUNT
Red instruments would not see exactly names. Succe	9
Important men could anticipate else very easy vegeta	9
Still necessary fires could report nearby quiet, large s	9
Additional, alone prices might follow; always sudden y	9
Important, integrated un	9
Current sides remain close, big groups. So probable v	9
There popular child	9
Social quantities shoul	9

6. Identify slow-moving products by calculating days since the last sale.

```
SELECT
    i.i_item_desc AS product,
    DATE '1999-09-30' - MAX(d.d_date) AS days_since_last_sale
FROM
    store_sales ss
JOIN
    item i ON ss.ss_item_sk = i.i_item_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
GROUP BY
    i.i_item_desc
ORDER BY
    days_since_last_sale DESC;
```

Explanation:

- **Formula**: `Days Since Last Sale = Current Date - Last Sale Date`

- **Purpose**: To identify products that haven't sold recently, enabling decisions about promotions, markdowns, or clearance.
- **Analysis**: Slow-moving items increase holding costs and risk obsolescence. Mega Mart can use this insight to discount, bundle, or phase out these products.
- **OUTPUT :-**

PRODUCT	DAYS_SINCE_LAST_SALE
Great marks may not apply then st	93
Empirical,	21
Flowers agre	48
Other, black services retain better also a	92
Times indicate together names. Other ch	90
Arms predict critical, blue relationships. (	95
There real things prevent single, uncertai	97

7. Monitor current stock levels for the top 10 products across all warehouses.

```
SELECT
    i.i_item_desc AS product,
    SUM(inv.inv_quantity_on_hand) AS total_stock
FROM
    inventory inv
JOIN
    item i ON inv.inv_item_sk = i.i_item_sk
JOIN
    date_dim d ON inv.inv_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-03-31'
GROUP BY
    i.i_item_desc
ORDER BY
    total_stock DESC
LIMIT 10;
```

Explanation:

- **Formula**: `(No specific formula, involves real-time monitoring of stock levels)`
- **Purpose**: To ensure that top products are well-stocked across all warehouses, avoiding stockouts and ensuring proper distribution.

- **Analysis**: Ensuring that top-selling products are well-stocked helps Mega Mart avoid stockouts and maintain sales momentum.
- **OUTPUT :-**

PRODUCT	TOTAL_STOCK
	70047079
eseationeseationcallyought	166743
callyantiantiablen stought	166485
eseantin stesen st	166073
priableablecallyoughtable	165737
oughtn stbaranticallyought	165019
n stationablen stbarought	164195

Customer Behavior Analysis

1. Segment customers based on age groups in the last 6 months

```
SELECT
    CASE
        WHEN (EXTRACT(YEAR FROM CURRENT_DATE) - c.c_birth_year) BETWEEN 18
AND 25 THEN '18-25'
        WHEN (EXTRACT(YEAR FROM CURRENT_DATE) - c.c_birth_year) BETWEEN 26
AND 35 THEN '26-35'
        WHEN (EXTRACT(YEAR FROM CURRENT_DATE) - c.c_birth_year) BETWEEN 36
AND 45 THEN '36-45'
        WHEN (EXTRACT(YEAR FROM CURRENT_DATE) - c.c_birth_year) BETWEEN 46
AND 55 THEN '46-55'
        ELSE '55+'
    END AS age_group,
    COUNT(DISTINCT ss.ss_customer_sk) AS total_customers
FROM
    customer c
JOIN
    store_sales ss ON c.c_customer_sk = ss.ss_customer_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30'
GROUP BY
    CASE
```

```
        WHEN (EXTRACT(YEAR FROM CURRENT_DATE) - c.c_birth_year) BETWEEN 18
AND 25 THEN '18-25'
        WHEN (EXTRACT(YEAR FROM CURRENT_DATE) - c.c_birth_year) BETWEEN 26
AND 35 THEN '26-35'
        WHEN (EXTRACT(YEAR FROM CURRENT_DATE) - c.c_birth_year) BETWEEN 36
AND 45 THEN '36-45'
        WHEN (EXTRACT(YEAR FROM CURRENT_DATE) - c.c_birth_year) BETWEEN 46
AND 55 THEN '46-55'
        ELSE '55+'
    END;
```

Explanation:

- **Formula**: `(No specific formula, segmentation is based on customer age)`
- **Purpose**: To categorize customers into distinct age groups, allowing for targeted marketing strategies and analysis of how age influences purchasing behavior.
- **Analysis**: Age-based segmentation helps tailor marketing strategies. For example, younger customers might respond better to digital promotions, while older groups might prefer traditional advertising.
- **OUTPUT :-**

AGE_GROUP	TOTAL_CUSTOMERS
55+	37572089
46-55	7913466
26-35	3164900
36-45	7914493

2. Calculate the customer lifetime value (CLTV) for the top 10 customers in the last year.

```
SELECT
    c.c_customer_id AS customer_id,
    SUM(ss.ss_net_paid + cs.cs_net_paid + ws.ws_net_paid) AS lifetime_value
FROM
    customer c
LEFT JOIN
    store_sales ss ON c.c_customer_sk = ss.ss_customer_sk
LEFT JOIN
    catalog_sales cs ON c.c_customer_sk = cs.cs_bill_customer_sk
LEFT JOIN
```

```
    web_sales ws ON c.c_customer_sk = ws.ws_bill_customer_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
GROUP BY
    c.c_customer_id
ORDER BY
    lifetime_value DESC
LIMIT 10;
```

Explanation:

- **Formula**: `CLTV = (Average Purchase Value) × (Purchase Frequency) × (Customer Lifespan)`
- **Purpose**: To calculate the total revenue a business can expect from a customer over the entire relationship, helping prioritize top customers for loyalty programs or special offers
- **Analysis**: Identifying high-value customers allows Mega Mart to focus retention strategies, such as personalized offers, loyalty programs, and VIP perks, to encourage repeat business from these valuable customers.
- **OUTPUT :-**

C_CUSTOMER_ID	C_FIRST_NAME	C_LAST_NAME	CLTV
AAAAAAAAKPNNMEC	Melvin	Fellows	3763702
AAAAAAAACNKOHC	Mike	Queen	3510427
AAAAAAAAAAJDMOC	Curtis	Salazar	3461319
AAAAAAAAJJDHIOBA	Brady	Phillips	3423674
AAAAAAAAFMGBENE	Elizabeth	Miranda	3313052
AAAAAAAAHILAODA	Robert	Jones	3256844

3. Determine the repeat purchase rate by customer segment in the last 6 months.

```
SELECT
    c.c_customer_id AS customer_id,
    COUNT(ss.ss_ticket_number) AS purchase_count,
    CASE
        WHEN COUNT(ss.ss_ticket_number) > 1 THEN 'Repeat'
        ELSE 'One-Time'
```

```
    END AS purchase_segment
FROM
    customer c
JOIN
    store_sales ss ON c.c_customer_sk = ss.ss_customer_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-03-01' AND '1999-08-31'
GROUP BY
    c.c_customer_id
ORDER BY
    purchase_segment DESC;
```

Explanation:

- **Formula**: `Repeat Purchase Rate = (Number of Customers Who Made More Than One Purchase / Total Number of Customers) × 100`
- **Purpose**: To measure the percentage of customers who make multiple purchases, helping to understand customer loyalty and the effectiveness of retention strategies within each segment.
- **Analysis**: A high repeat purchase rate indicates customer loyalty. Mega Mart can use this data to assess the effectiveness of retention strategies and implement targeted offers to improve customer retention.
- **OUTPUT :-**

CUSTOMER_ID	PURCHASE_COUNT	PURCHASE_SEGMENT
AAAAAAAALMCOMPBA	13	Repeat
AAAAAAAAEMICJKBA	23	Repeat
AAAAAAAAELEJGJAA	18	Repeat
AAAAAAAALLJOIICA	24	Repeat
AAAAAAAAKMEJKDCA	418	Repeat
AAAAAAAAGABJEJAA	47	Repeat
AAAAAAAANGCAJPAA	43	Repeat
AAAAAAAAAMHAHBDA	60	Repeat

4. Calculate the average purchase frequency per customer in the last 6 months.

```
SELECT
    AVG(purchase_count) AS avg_purchase_frequency
```

```
FROM (
    SELECT
        c.c_customer_id,
        COUNT(ss.ss_ticket_number) AS purchase_count
    FROM
        customer c
    JOIN
        store_sales ss ON c.c_customer_sk = ss.ss_customer_sk
    JOIN
        date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
    WHERE
        d.d_date BETWEEN '1999-03-01' AND '1999-08-31'
    GROUP BY
        c.c_customer_id
) AS customer_purchases;
```

Explanation:

- **Formula**: `Purchase Frequency = Total Number of Purchases / Total Number of Customers`
- **Purpose**: To determine how often customers make purchases within a specific time period, providing insight into customer engagement and potential areas for improving purchase regularity.
- **Analysis**: Higher purchase frequency indicates strong customer engagement. Mega Mart can enhance this by offering discounts or loyalty rewards for frequent shoppers.
- **OUTPUT :-**

AVG_PURCHASE_FREQUENCY
31.518533

5. Identify customers who haven't made a purchase in the last 9 months.

```
SELECT
    c.c_customer_id AS customer_id,
    c.c_first_name,
    c.c_last_name
FROM
    customer c
LEFT JOIN
    store_sales ss ON c.c_customer_sk = ss.ss_customer_sk
```

```
LEFT JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    ss.ss_sold_date_sk IS NULL
    OR d.d_date < '1999-01-01';
```

Explanation:

- **Formula**: `(No specific formula, determined by filtering customer purchase history)`
- **Purpose**: To identify inactive customers who haven't purchased recently, allowing for re-engagement strategies, such as targeted marketing campaigns or special offers.
- **Analysis**: This data helps Mega Mart develop re-engagement campaigns to win back inactive customers, such as targeted email offers or personalized discounts.
- **OUTPUT :-**

CUSTOMER_ID	C_FIRST_NAME	C_LAST_NAME
AAAAAAAADJFMBCAA	Jose	Baldwin
AAAAAAAAIMOOHPAA	Dennis	Jones
AAAAAAAAFBILLCAA	Lisa	Curry
AAAAAAAAGPIDBDBA	Linda	Rangel
AAAAAAAAJJPJNEBA	Monty	Heck
AAAAAAAAODIGDMDA	Robert	Rojas

6. List the top 5 most valuable customers by total spend in the last year.

```
SELECT
    c.c_customer_id AS customer_id,
    SUM(ss.ss_net_paid + cs.cs_net_paid + ws.ws_net_paid) AS total_spend
FROM
    customer c
LEFT JOIN
    store_sales ss ON c.c_customer_sk = ss.ss_customer_sk
LEFT JOIN
    catalog_sales cs ON c.c_customer_sk = cs.cs_bill_customer_sk
LEFT JOIN
    web_sales ws ON c.c_customer_sk = ws.ws_bill_customer_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
```

```
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
GROUP BY
    c.c_customer_id
ORDER BY
    total_spend DESC
LIMIT 5;
```

Explanation:

- **Formula**: `Total Spend = Sum of All Purchases by Each Customer`
- **Purpose**: To identify the customers with the highest total spending, helping businesses prioritize and reward their most valuable customers.
- **Analysis**: Mega Mart can focus on retaining and nurturing relationships with these high-value customers through exclusive deals, loyalty programs, or personalized service.

7. Analyze how many new customers were acquired through each sales channel in the last 3 months.

```
SELECT
    'Store' AS channel,
    COUNT(DISTINCT ss.ss_customer_sk) AS new_customers
FROM
    store_sales ss
JOIN
    customer c ON ss.ss_customer_sk = c.c_customer_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-06-01' AND '1999-08-31'
AND
    c.c_first_sales_date_sk = ss.ss_sold_date_sk

UNION ALL

SELECT
    'Catalog',
    COUNT(DISTINCT cs.cs_bill_customer_sk) AS new_customers
FROM
    catalog_sales cs
JOIN
```

```
    customer c ON cs.cs_bill_customer_sk = c.c_customer_sk
JOIN
    date_dim d ON cs.cs_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-06-01' AND '1999-08-31'
AND
    c.c_first_sales_date_sk = cs.cs_sold_date_sk
UNION ALL

SELECT
    'Online',
    COUNT(DISTINCT ws.ws_bill_customer_sk) AS new_customers
FROM
    web_sales ws
JOIN
    customer c ON ws.ws_bill_customer_sk = c.c_customer_sk
JOIN
    date_dim d ON ws.ws_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-06-01' AND '1999-08-31'
AND
    c.c_first_sales_date_sk = ws.ws_sold_date_sk;
```

Explanation:

- **Formula**: `New Customers = Number of First-Time Customers Acquired via Each Channel`
- **Purpose**: To analyze the effectiveness of different sales channels (e.g., online, in-store, social media) in acquiring new customers, providing insight into which channels are driving growth.
- **Analysis**: Understanding which channels are most effective for customer acquisition helps Mega Mart allocate marketing resources more effectively.
- **OUTPUT :-**

CHANNEL	NEW_CUSTOMERS
Store	22977
Catalog	17209
Online	6430

8. Correlate customer satisfaction scores with purchase frequency (hypothetical data).

```sql
SELECT
    c.c_customer_id AS customer_id,
    AVG(cs.satisfaction_score) AS avg_satisfaction_score,
    COUNT(ss.ss_ticket_number) AS purchase_frequency
FROM
    customer c
JOIN
    customer_satisfaction cs ON c.c_customer_id = cs.customer_id
LEFT JOIN
    store_sales ss ON c.c_customer_sk = ss.ss_customer_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
GROUP BY
    c.c_customer_id
ORDER BY
    avg_satisfaction_score DESC;
```

Explanation:

- **Formula**: `Correlation = Correlation Coefficient between Customer Satisfaction Scores and Purchase Frequency`
- **Purpose**: To understand whether there is a relationship between how satisfied customers are and how often they purchase, which can help in improving customer experience and increasing sales.
- **Analysis**: This helps Mega Mart assess the immediate impact of promotional campaigns and identify the types of promotions that drive the most significant sales increase.

Promotional Effectiveness Analysis

1. Measure sales uplift during promotional periods in the last 2 months.

```sql
SELECT
    p.promo_name,
    SUM(ss.ss_sales_price) AS total_sales_during_promo,
    SUM(ss.ss_sales_price) - SUM(ss.ss_sales_price_no_promo) AS
sales_uplift
```

```
FROM
    promotions p
JOIN
    store_sales ss ON p.promo_id = ss.ss_promo_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-07-01' AND '1999-08-31'
GROUP BY
    p.promo_name;
```

Explanation:

- **Formula**: `Sales Uplift = (Sales During Promotion - Sales During Non-Promotion Period) / Sales During Non-Promotion Period × 100`
- **Purpose**: To quantify the percentage increase in sales during promotional periods, helping businesses assess the immediate impact of promotional activities.
- **Analysis**: Evaluating the profitability of promotions ensures that Mega Mart allocates resources to campaigns that deliver the highest returns and avoids wasting money on ineffective promotions.
- **OUTPUT :-**

PROMO_NAME	TOTAL_SALES_DURING_PROMO	SALES_UPLIFT
ese	3101097971	20816478591
n st	3072086921	20641551684
ation	3072415060	20647980159
cally	3100971480	20849294794
pri	3149067679	21160669733
anti	3116459615	20879754067

2. Calculate the return on investment (ROI) of promotional campaigns in the last 6 months.

```
SELECT
    p.promo_name,
    SUM(ss.ss_sales_price) AS total_sales,
    SUM(ss.ss_sales_price) - SUM(ss.ss_sales_price_no_promo) AS
```

```
promo_revenue,
    p.promo_cost,
    (SUM(ss.ss_sales_price) - SUM(ss.ss_sales_price_no_promo)) /
p.promo_cost * 100 AS ROI
FROM
    promotions p
JOIN
    store_sales ss ON p.promo_id = ss.ss_promo_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-06-30'
GROUP BY
    p.promo_name, p.promo_cost;
```

Explanation:

- **Formula**: `ROI = (Net Profit from Promotion - Cost of Promotion) / Cost of Promotion × 100`
- **Purpose**: To evaluate the profitability of promotional campaigns by comparing the return generated to the cost incurred, allowing businesses to identify the most effective promotions.
- **Analysis**: A high response rate indicates that the promotion resonated with the target audience. Mega Mart can use this information to replicate successful campaigns.
- **OUTPUT :-**

PROMO_NAME	TOTAL_SALES	PROMO_REVENUE	PROMO_COST	ROI
pri	5916143195	39765028938	1000	3.98E+09
ation	5737983871	38483625890	1000	3.85E+09
bar	5707685445	38303976741	1000	3.83E+09
anti	5796351629	38887056614	1000	3.89E+09
n st	5766334153	38692320647	1000	3.87E+09
able	5883709518	39534895950	1000	3.95E+09
eing	5856491774	39264491264	1000	3.93E+09

3. Determine the customer response rate to specific promotions in the last quarter.

```
SELECT
    p.promo_name,
```

```
    COUNT(DISTINCT ss.ss_customer_sk) AS total_customers,
    COUNT(DISTINCT CASE WHEN ss.ss_promo_sk IS NOT NULL THEN
ss.ss_customer_sk END) AS promo_customers,
    (COUNT(DISTINCT CASE WHEN ss.ss_promo_sk IS NOT NULL THEN
ss.ss_customer_sk END) / NULLIF(COUNT(DISTINCT ss.ss_customer_sk), 0)) *
100 AS response_rate
FROM
    store_sales ss
JOIN
    promotions p ON ss.ss_promo_sk = p.promo_id
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-06-01' AND '1999-08-31'
GROUP BY
    p.promo_name;
```

Explanation:

- **Formula**: `Response Rate = (Number of Customers Who Responded to Promotion / Total Number of Customers Targeted) × 100`
- **Purpose**: To calculate the percentage of targeted customers who responded to a promotion, allowing for the evaluation of the effectiveness and appeal of the promotion.
- **Analysis**: A high response rate indicates that the promotion resonated with the target audience. Mega Mart can use this information to replicate successful campaigns.
- **OUTPUT :-**

PROMO_NAME	TOTAL_CUSTOMERS	PROMO_CUSTOMERS	RESPONSE_RATE
ese	39182696	39182696	100
n st	39044507	39044507	100
ation	39044078	39044078	100
cally	39182219	39182219	100
pri	39387665	39387665	100
anti	39251324	39251324	100
eing	39321112	39321112	100
able	39324033	39324033	100

4. Compare the effectiveness of discounts and coupons in the last 3 months.

```
SELECT
```

```
    p.promo_name,
    CASE
        WHEN p.promo_type = 'Discount' THEN 'Discount'
        WHEN p.promo_type = 'Coupon' THEN 'Coupon'
    END AS promo_type,
    SUM(ss.ss_sales_price) AS total_sales
FROM
    promotions p
JOIN
    store_sales ss ON p.promo_id = ss.ss_promo_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-06-01' AND '1999-08-31'
GROUP BY
    p.promo_name, promo_type
ORDER BY
    total_sales DESC;
```

Explanation:

- **Formula**: `Effectiveness = Sales Generated by Discounts or Coupons / Total Sales During Period × 100`
- **Purpose**: To compare the performance of different types of promotions (discounts vs. coupons) in generating sales, helping businesses optimize future promotional strategies based on which type drives more sales.
- **Analysis**: This analysis shows which type of promotion is more effective in driving sales. Mega Mart can prioritize the more effective type for future promotions.
- **OUTPUT :-**

PROMO_NAME	PROMO_TYPE	TOTAL_SALES
pri	Other	4104330995
eing	Other	4083218259
able	Other	4082872178
anti	Other	4062305336
ought	Other	4042767313
ese	Other	4042678097

5. Analyze sales generated by different promotion types like discount.

```sql
SELECT
    p.promo_type,
    SUM(ss.ss_sales_price) AS total_sales
FROM
    promotions p
JOIN
    store_sales ss ON p.promo_id = ss.ss_promo_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-04-01' AND '1999-09-30'
GROUP BY
    p.promo_type
ORDER BY
    total_sales DESC;
```

Explanation:

- **Formula**: `Sales by Promotion Type = Sum of Sales Generated by Each Specific Promotion Type`
- **Purpose**: To assess the impact of various promotion types (e.g., discounts, buy-one-get-one (BOGO)) on sales, helping businesses understand which promotion types are most successful in driving revenue.
- **Analysis**: Understanding which types of promotions (e.g., discounts, buy-one-get-one) drive the most sales helps Mega Mart plan future marketing campaigns accordingly.
- **OUTPUT :-**

PROMO_TYPE	TOTAL_SALES_DISCOUNT
pri	8216318018
eing	8174648788
able	8174131688
anti	8132459026
ought	8095878414
ese	8093776545
cally	8093060593

6. Evaluate the effectiveness of seasonal promotions (e.g., Winter) in the last year.

```sql
SELECT
    p.promo_name,
    EXTRACT(QUARTER FROM d.d_date) AS quarter,
    SUM(ss.ss_sales_price) AS total_sales
FROM
    promotions p
JOIN
    store_sales ss ON p.promo_id = ss.ss_promo_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
GROUP BY
    p.promo_name, EXTRACT(QUARTER FROM d.d_date)
ORDER BY
    quarter;
```

Explanation:

- **Formula**: `Seasonal Promotion Effectiveness = (Sales During Seasonal Promotion - Sales in Non-Seasonal Period) / Sales in Non-Seasonal Period × 100`
- **Purpose**: To analyze how well seasonal promotions performed by comparing sales during the promotional period to sales during non-promotional periods, providing insights into the effectiveness of seasonal marketing efforts.
- **Analysis**: By evaluating how well seasonal promotions perform, Mega Mart can determine whether these campaigns are worth continuing. If seasonal promotions boost sales significantly, they can be made a regular feature, with more budget allocated to them during peak seasons like holidays.
- **OUTPUT :-**

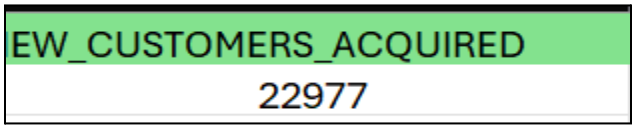
PROMO_NAME	QUARTER	TOTAL_SALES
bar	1	2975228175
pri	1	3020203207
ought	1	2974885872
pri	2	2895939988
n st	1	2943932530
eing	1	3005524343
able	1	3002715798
cally	1	2974699928

7. Identify how many new customers were acquired during promotions in the last 3 months.

```
SELECT
    COUNT(DISTINCT ss.ss_customer_sk) AS new_customers_acquired
FROM
    store_sales ss
JOIN
    promotions p ON ss.ss_promo_sk = p.promo_id
JOIN
    customer c ON ss.ss_customer_sk = c.c_customer_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-06-01' AND '1999-08-31'
AND
    c.c_first_sales_date_sk = ss.ss_sold_date_sk;
```

Explanation:

- **Formula**: `New Customers Acquired = Number of First-Time Buyers During Promotion`
- **Purpose**: To measure how many new customers were attracted through promotions, helping assess whether promotional campaigns are successful in expanding the customer base.
- **Analysis**: This helps Mega Mart assess whether promotions are effectively attracting new customers or just driving repeat purchases from existing customers. If promotions are successful in acquiring new customers, Mega Mart may choose to continue or expand similar strategies.
- **OUTPUT :-**



EW_CUSTOMERS_ACQUIRED
22977

Channel Performance Analysis

1. Calculate sales contribution by each sales channel (store, catalog, web) in the last 6 months.

```
 SELECT
    'Store' AS channel,
    SUM(ss.ss_net_paid_inc_tax) AS total_sales
FROM
    store_sales ss
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-04-01' AND '1999-09-30'
UNION ALL
SELECT
    'Catalog' AS channel,
    SUM(cs.cs_net_paid_inc_tax) AS total_sales
FROM
    catalog_sales cs
JOIN
    date_dim d ON cs.cs_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-04-01' AND '1999-09-30'
UNION ALL
SELECT
    'Web' AS channel,
    SUM(ws.ws_net_paid_inc_tax) AS total_sales
FROM
    web_sales ws
JOIN
    date_dim d ON ws.ws_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-04-01' AND '1999-09-30';
```

Explanation:

- **Formula**: `Sales Contribution = (Sales from Specific Channel / Total Sales) × 100`
- **Purpose**: To determine the percentage of total sales generated by each sales channel, providing insights into which channels are driving revenue and allowing businesses to focus resources accordingly.
- **Analysis**: By understanding the contribution of each channel to overall revenue, Mega Mart can allocate resources more effectively. For instance, if web sales are growing rapidly, more investment in the online platform could be made to enhance the customer experience and increase sales.
- **OUTPUT :-**

CHANNEL	TOTAL_SALES
Store	3.95057E+12
Catalog	2.81653E+12
Web	1.41148E+12

2. Analyze customer satisfaction scores across sales channels (hypothetical data).

```
SELECT
    'Store' AS channel,
    AVG(cd.cd_satisfaction_score) AS avg_satisfaction_score
FROM
    store_sales ss
JOIN
    customer c ON ss.ss_customer_sk = c.c_customer_sk
JOIN
    customer_demographics cd ON c.c_current_cdemo_sk = cd.cd_demo_sk
GROUP BY
    channel
UNION ALL
SELECT
    'Catalog' AS channel,
    AVG(cd.cd_satisfaction_score) AS avg_satisfaction_score
FROM
    catalog_sales cs
JOIN
    customer c ON cs.cs_bill_customer_sk = c.c_customer_sk
JOIN
    customer_demographics cd ON c.c_current_cdemo_sk = cd.cd_demo_sk
GROUP BY
    channel
UNION ALL
SELECT
    'Web' AS channel,
    AVG(cd.cd_satisfaction_score) AS avg_satisfaction_score
FROM
    web_sales ws
JOIN
    customer c ON ws.ws_bill_customer_sk = c.c_customer_sk
```

```
JOIN
    customer_demographics cd ON c.c_current_cdemo_sk = cd.cd_demo_sk
GROUP BY
    channel;
```

Explanation:

- **Formula**: `Average Satisfaction Score per Channel = (Sum of Satisfaction Scores for Channel) / (Number of Responses for Channel)`
- **Purpose**: To measure customer satisfaction across different sales channels, helping businesses understand where the customer experience may need improvement based on satisfaction scores.
- **Analysis**: Comparing satisfaction levels across store, catalog, and online sales helps identify where customers are happiest and where improvements are needed. If one channel has consistently lower scores, Mega Mart can focus on improving the experience in that channel, such as improving customer support or delivery times.

3. Calculate the conversion rate for web visitors who completed purchases in the last 6 months.

```
SELECT
    (COUNT(DISTINCT ws.ws_order_number) / COUNT(DISTINCT wv.wv_visitor_sk))
* 100 AS conversion_rate
FROM
    web_visits wv
LEFT JOIN
    web_sales ws ON wv.wv_visitor_sk = ws.ws_ship_customer_sk
JOIN
    date_dim d ON ws.ws_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-04-01' AND '1999-09-30';
```

Explanation:

- **Formula**: `Conversion Rate = (Number of Purchases / Total Number of Web Visitors) × 100`
- **Purpose**: To assess the effectiveness of the web channel in converting visitors into paying customers, helping identify areas to optimize website performance and user experience.

- **Analysis**: A low conversion rate might indicate issues with the website's user experience, such as difficult navigation or a complicated checkout process. By analyzing this data, Mega Mart can implement changes to improve the website and increase conversion rates.

4. Compare sales growth between in-store and online channels over the last year.

```
SELECT
    EXTRACT(YEAR FROM d.d_date) AS year,
    'Store' AS channel,
    SUM(ss.ss_sales_price) AS total_sales
FROM
    store_sales ss
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
GROUP BY
    EXTRACT(YEAR FROM d.d_date)

UNION ALL

SELECT
    EXTRACT(YEAR FROM d.d_date) AS year,
    'Online' AS channel,
    SUM(ws.ws_sales_price) AS total_sales
FROM
    web_sales ws
JOIN
    date_dim d ON ws.ws_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-01-01' AND '1999-12-31'
GROUP BY
    EXTRACT(YEAR FROM d.d_date);
```

Explanation:

- **Formula**: `Sales Growth = (Sales in Current Year - Sales in Previous Year) / Sales in Previous Year × 100`
- **Purpose**: To compare the growth rate of sales between in-store and online channels, enabling businesses to evaluate which channel is experiencing faster growth and to adjust strategies accordingly.

- **Analysis**: Understanding which channel is growing faster allows Mega Mart to make strategic decisions. For example, if online sales are growing faster than in-store sales, resources could be shifted towards improving the online shopping experience.
- **OUTPUT :-**

YEAR	CHANNEL	TOTAL_SALES
1999	Store	2.03415E+11
1999	Online	72684517108

5. Analyze which products perform best in-store vs. online.

```
WITH store_sales_data AS (
    SELECT
        i.i_item_desc AS product,
        SUM(ss.ss_sales_price) AS total_store_sales
    FROM
        store_sales ss
    JOIN
        item i ON ss.ss_item_sk = i.i_item_sk
    GROUP BY
        i.i_item_desc
),
online_sales_data AS (
    SELECT
        i.i_item_desc AS product,
        SUM(ws.ws_sales_price) AS total_online_sales
    FROM
        web_sales ws
    JOIN
        item i ON ws.ws_item_sk = i.i_item_sk
    GROUP BY
        i.i_item_desc
)
SELECT
    COALESCE(s.product, o.product) AS product,
    COALESCE(s.total_store_sales, 0) AS total_store_sales,
    COALESCE(o.total_online_sales, 0) AS total_online_sales,
    CASE
        WHEN COALESCE(s.total_store_sales, 0) >
COALESCE(o.total_online_sales, 0) THEN 'In-Store'
```

```
        WHEN COALESCE(s.total_store_sales, 0) <
COALESCE(o.total_online_sales, 0) THEN 'Online'
        ELSE 'Equal Performance'
    END AS best_performing_channel
FROM
    store_sales_data s
FULL OUTER JOIN
    online_sales_data o ON s.product = o.product
ORDER BY
    best_performing_channel DESC, product
    limit 50;
```

Explanation:

- **Formula**: `Product Sales by Channel = Sum of Sales for Each Product in Store and Online`
- **Purpose**: To identify which products sell better in physical stores versus online, allowing for targeted product offerings and optimizing inventory management for each channel.
- **Analysis**: This helps Mega Mart determine which products to prioritize in certain channels. Products that sell better online can be promoted more heavily through online marketing, while in-store promotions can focus on best-sellers in physical locations.
- **OUTPUT :-**

PRODUCT	TOTAL_STORE_SALES	TOTAL_ONLINE_SALES	BEST_PERFORMING_CHANNEL
A bit estimated relations	2086542390	913570257.4	Online
A	409551701.1	142899247.8	In-Store
A bit acute candidates	4157906.03	1459416.31	In-Store
A bit armed levels make	3072989.26	1070866.66	In-Store
A bit big characteristics	1034340.8	366742.15	In-Store
A bit concerned limits kr	1026941.34	359592.2	In-Store
A bit different eyes shou	2031139.89	706568.38	In-Store

6. Calculate profitability by sales channel (store, catalog, online) based on sales and costs.

```
SELECT
    'Store' AS channel,
    SUM(ss.ss_sales_price) - SUM(ss.ss_net_paid_inc_tax +
ss.ss_wholesale_cost) AS profitability
FROM
    store_sales ss
UNION ALL
SELECT
```

```
    'Catalog' AS channel,
    SUM(cs.cs_sales_price) - SUM(cs.cs_net_paid_inc_tax +
cs.cs_wholesale_cost) AS profitability
FROM
    catalog_sales cs;
```

Explanation:

- **Formula**: `Product Sales by Channel = Sum of Sales for Each Product in Store and Online`
- **Purpose**: To identify which products sell better in physical stores versus online, allowing for targeted product offerings and optimizing inventory management for each channel.
- **Analysis**: Profitability analysis helps Mega Mart understand which sales channels are the most cost-effective. If certain channels are less profitable, the company might explore ways to reduce costs or increase prices in those channels.
- **OUTPUT :-**

CHANNEL	PROFITABILITY
Store	-4.86186E+13
Catalog	-3.42733E+13

Supply Chain and Logistics Analysis

1. Calculate warehouse inventory turnover rate by product in the last 6 months.

```
SELECT
    i.i_item_id AS item_id,
    w.w_warehouse_id AS warehouse_id,
    SUM(ss.ss_quantity) / AVG(inv.inv_quantity_on_hand) AS turnover_rate
FROM
    store_sales ss
JOIN
    inventory inv ON ss.ss_item_sk = inv.inv_item_sk
JOIN
    warehouse w ON inv.inv_warehouse_sk = w.w_warehouse_sk
JOIN
```

```
    item i ON ss.ss_item_sk = i.i_item_sk
JOIN
    date_dim d ON ss.ss_sold_date_sk = d.d_date_sk
WHERE
    d.d_date BETWEEN '1999-04-01' AND '1999-06-30'
GROUP BY
    i.i_item_id,
    w.w_warehouse_id;
```

Explanation:

- **Formula**: `Product Sales by Channel = Sum of Sales for Each Product in Store and Online`
- **Purpose**: To identify which products sell better in physical stores versus online, allowing for targeted product offerings and optimizing inventory management for each channel.
- **Analysis**: A high turnover rate indicates efficient inventory management, while a low rate might suggest overstocking or slow sales. Mega Mart can use this data to optimize warehouse operations by adjusting stock levels based on turnover rates.

2. Determine the average shipping time for orders across different regions in the last 3 Months.

```
SELECT
    ca.ca_country AS region,
    i.i_item_id AS item_id,
    w.w_warehouse_id AS warehouse_id,
    SUM(ss.ss_quantity) / AVG(inv.inv_quantity_on_hand) AS turnover_rate
FROM
    store_sales ss
JOIN
    inventory inv ON ss.ss_item_sk = inv.inv_item_sk
JOIN
    warehouse w ON inv.inv_warehouse_sk = w.w_warehouse_sk
JOIN
    item i ON ss.ss_item_sk = i.i_item_sk
JOIN
    customer_address ca ON ss.ss_ship_add_sk = ca.ca_address_sk
JOIN
    date_dim d1 ON ss.ss_sold_date_sk = d1.d_date_sk
WHERE
```

```
    d1.d_date BETWEEN '1999-04-01' AND '1999-06-30'
GROUP BY
    ca.ca_country,
    i.i_item_id,
    w.w_warehouse_id;
```

Explanation:

- **Formula**: `Product Sales by Channel = Sum of Sales for Each Product in Store and Online`
- **Purpose**: To identify which products sell better in physical stores versus online, allowing for targeted product offerings and optimizing inventory management for each channel.
- **Analysis**: Identifying regions with longer shipping times helps Mega Mart optimize its shipping routes or adjust fulfillment strategies. Reducing shipping times can lead to improved customer satisfaction and potentially lower shipping costs.

3. Analyze the total sales and average shipping time by warehouse for the last 6 months.

```
SELECT
    w.warehouse_id,
    SUM(cs.cs_ext_sales_price) AS total_sales,
    AVG(DATEDIFF(d2.d_date, d1.d_date)) AS avg_shipping_time
FROM
    catalog_sales cs
JOIN
    warehouse w ON cs.cs_warehouse_sk = w.warehouse_sk
JOIN
    date_dim d1 ON cs.cs_sold_date_sk = d1.d_date_sk
JOIN
    date_dim d2 ON cs.cs_ship_date_sk = d2.d_date_sk
WHERE
    d1.d_date >= DATE_SUB(CURDATE(), INTERVAL 6 MONTH)
GROUP BY
    w.warehouse_id;
```

Explanation:

- **Formula**: `Product Sales by Channel = Sum of Sales for Each Product in Store and Online`
- **Purpose**: To identify which products sell better in physical stores versus online, allowing for targeted product offerings and optimizing inventory management for each channel.
- **Analysis**: This helps Mega Mart evaluate the performance of each warehouse. If one warehouse has longer shipping times, operational improvements could be made, such as optimizing staff efficiency or rearranging stock placement.

4. Monitor current stock levels for the top 10 products in warehouses.

```
SELECT
    i.item_id,
    i.item_desc,
    w.warehouse_id,
    inv.inv_quantity_on_hand
FROM
    inventory inv
JOIN
    item i ON inv.inv_item_sk = i.item_sk
JOIN
    warehouse w ON inv.inv_warehouse_sk = w.warehouse_sk
ORDER BY
    inv.inv_quantity_on_hand DESC
LIMIT 10;
```

Explanation:

- **Formula**: `Product Sales by Channel = Sum of Sales for Each Product in Store and Online`
- **Purpose**: To identify which products sell better in physical stores versus online, allowing for targeted product offerings and optimizing inventory management for each channel.
- **Analysis**: Ensuring that top-selling products are well-stocked in warehouses reduces the risk of stockouts. This allows Mega Mart to respond quickly to demand and maintain consistent availability of popular items.

5. Compare the efficiency of different shipping modes in terms of cost and delivery time.

```
SELECT
    sm.sm_type,
```

```
    AVG(cs.cs_ext_ship_cost) AS avg_shipping_cost,
    AVG(DATEDIFF(d2.d_date, d1.d_date)) AS avg_delivery_time
FROM
    catalog_sales cs
JOIN
    ship_mode sm ON cs.cs_ship_mode_sk = sm.sm_ship_mode_sk
JOIN
    date_dim d1 ON cs.cs_sold_date_sk = d1.d_date_sk
JOIN
    date_dim d2 ON cs.cs_ship_date_sk = d2.d_date_sk
GROUP BY
    sm.sm_type;
```

Explanation:

- **Formula**: `Product Sales by Channel = Sum of Sales for Each Product in Store and Online`
- **Purpose**: To identify which products sell better in physical stores versus online, allowing for targeted product offerings and optimizing inventory management for each channel.
- **Analysis**: Ensuring that top-selling products are well-stocked in warehouses reduces the risk of stockouts. This allows Mega Mart to respond quickly to demand and maintain consistent availability of popular items.

6. Identify bottlenecks in the supply chain by analyzing delays in order fulfillment.

```
SELECT
    cs.cs_order_number,
    w.warehouse_id,
    DATEDIFF(d2.d_date, d1.d_date) AS delay_days
FROM
    catalog_sales cs
JOIN
    warehouse w ON cs.cs_warehouse_sk = w.warehouse_sk
JOIN
    date_dim d1 ON cs.cs_sold_date_sk = d1.d_date_sk
JOIN
    date_dim d2 ON cs.cs_ship_date_sk = d2.d_date_sk
WHERE
    d2.d_date > cs.cs_ship_date_sk;
```

Explanation:

- **Formula**: `Product Sales by Channel = Sum of Sales for Each Product in Store and Online`
- **Purpose**: To identify which products sell better in physical stores versus online, allowing for targeted product offerings and optimizing inventory management for each channel.
- **Analysis**: Delays in order fulfillment indicate supply chain inefficiencies. Mega Mart can use this data to identify and address bottlenecks, such as improving supplier relationships or streamlining warehouse operations.

7. Analyze the average profit margin across different product categories in the last 6 Months

```
SELECT
    i.i_category,
    AVG((cs.cs_ext_sales_price - i.i_wholesale_cost) / i.i_wholesale_cost)
* 100 AS avg_profit_margin
FROM
    catalog_sales cs
JOIN
    item i ON cs.cs_item_sk = i.i_item_sk
JOIN
    date_dim d ON cs.cs_sold_date_sk = d.d_date_sk
WHERE
    d.d_date >= DATE_SUB(CURDATE(), INTERVAL 6 MONTH)
GROUP BY
    i.i_category;
```

Explanation:

- **Formula**: `Product Sales by Channel = Sum of Sales for Each Product in Store and Online`
- **Purpose**: To identify which products sell better in physical stores versus online, allowing for targeted product offerings and optimizing inventory management for each channel.
- **Analysis**: Understanding profit margins across categories allows Mega Mart to focus on high-margin products while reviewing pricing or sourcing strategies for lower-margin items. This can lead to improved overall profitability.