MEESHO SALES REPORT SQL ANALYSIS

Q1- Find total profit by region :- *"Find total profit by region is asking you to calculate the overall profit made in each region"*

```
select * from total_profit_by_region;
```

Q2- Top 3 countries by total revenue :- "Top 3 countries by total revenue is asking you to identify which three countries have generated the most income (before subtracting any costs)."

```
select * from total_revenue_by_country;
```

Q3- Average unit cost per item type:- "This question is asking you to calculate the average cost of a single unit for each type of item sold."

```
select * from unit_per_cost
```

Q4- Orders with profit margin above 40% :-" find orders where the profit margin is greater than 40% Profit margin is a percentage that tells you how much profit was made compared to the revenue. It's calculated using the formula"

```
select * from profit_margin;
```

Q5- Count of orders by payment method :- *"figure out how many orders were placed using each different payment method."*

```
select * from payment_methods;
```

Q6- Find duplicate Order IDs (if any) :- *"check if any "Order ID" values appear more than once in your dataset. Each order is supposed to have a unique ID. But sometimes, due to errors in data entry or merging datasets, the same ID might show up in multiple rows—which could lead to misleading results if not cleaned properly."*

```
select * from duplicate_order_id;
```

Q7- Monthly sales trend :- "analyse how sales change month by month across your dataset."

```
select * from monthly_sales;
```

Q8- Highest profit order :- “identify the single order that generated the most profit.”

```
select * from highest_profit_order;
```

Q9- Orders shipped more than 30 days after order date :- “identify the orders where the time between the "Order Date" and the "Ship Date" is more than 30 days.”

```
select * from more_than_30_days;
```

Q10- Total revenue by sales channel :- “calculate how much money was made (in total) through each sales channel.”

```
select * from sales_channel;
```

Q11- Orders with unit price above average :- “identify all the orders where the "Unit Price" is higher than the average unit price across the entire dataset.”

```
select * from unit_price;
```

Q12- Top item type by profit in each region :- “identify which item type generated the highest total profit in each region.”

```
select * from profit_in_each_region;
```

Q13- Find orders with negative profit (if any) :- “identify any orders where the profit was less than zero—in other words, sales where the company actually lost money.”

```
select * from negative_profit;
```

Q14- Year-over-year revenue growth :-“ calculate how much the total revenue has increased or decreased each year compared to the previous year.”

```
select * from yearly_revenew;
```

Q15- Most frequently used payment method :- “determine which payment method was used the most across all the orders in your dataset.”

```
select * from most_used_payment_method;
```

QUERIES :-

#1- Find total profit by region

```
create view total_profit_by_region as
SELECT Region, ROUND(SUM(`Total Profit`),2) AS Total_Profit
FROM sales_records
GROUP BY Region;
select * from total_profit_by_region;
```

#2- Top 3 countries by total revenue

```
create view total_revenue_by_country as
SELECT Country, ROUND(SUM(`Total Revenue`), 2) AS Revenue
FROM sales_records
GROUP BY Country
ORDER BY Revenue DESC
LIMIT 3;
select * from total_revenue_by_country;
```

#3- Average unit cost per item type

```
create view unit_per_cost as
SELECT `Item Type`, ROUND(AVG(`Unit Cost`), 2) AS Avg_Cost
FROM sales_records
GROUP BY `Item Type`;
select * from unit_per_cost
```

# 4- Orders with profit margin above 40%

```
create view profit_margin as
SELECT *
FROM sales_records
```

```
WHERE (`Total Profit` / `Total Revenue`) > 0.4;

select * from profit_margin;
```

#5- Count of orders by payment method

```
create view payment_methods as

SELECT `Payment Method`, COUNT(*) AS Order_Count

FROM sales_records

GROUP BY `Payment Method`;

select * from payment_methods;
```

#6- Find duplicate Order IDs (if any)

```
create view duplicate_order_id as

SELECT `Order ID`, COUNT(*) AS Count

FROM sales_records

GROUP BY `Order ID`

HAVING COUNT(*) > 1;

select * from duplicate_order_id;
```

#7- Monthly sales trend

```
create view monthly_sales as

SELECT FORMAT(`Order Date`, 'yyyy-MM') AS Month, SUM(`Units Sold`) AS Units

FROM sales_records

GROUP BY FORMAT(`Order Date`, 'yyyy-MM')

ORDER BY Month;

select * from monthly_sales;
```

#8- Highest profit order

```
create view highest_profit_order as

SELECT *
```

```
FROM sales_records

ORDER BY `Total Profit` DESC

LIMIT 1;

select * from highest_profit_order;
```

#9- Orders shipped more than 30 days after order date

```
create view more_than_30_days as

SELECT *

FROM sales_records

WHERE DATEDIFF(`Ship Date`, `Order Date`) > 30;

select * from more_than_30_days;
```

#10- Total revenue by sales channel

```
create view sales_channel as

SELECT `Sales Channel`, SUM(`Total Revenue`) AS Revenue

FROM sales_records

GROUP BY `Sales Channel`;

select * from sales_channel;
```

#11- Orders with unit price above average

```
create view unit_price as

SELECT *

FROM sales_records

WHERE `Unit Price` > (SELECT AVG(`Unit Price`) FROM sales_records);

select * from unit_price;
```

#12- Top item type by profit in each region

```
create view profit_in_each_region as
```

```
SELECT Region, `Item Type`, ROUND(SUM(`Total Profit`),2) AS Profit

FROM sales_records

GROUP BY Region, `Item Type`

ORDER BY Region, Profit DESC;

select * from profit_in_each_region;
```

#13- Find orders with negative profit (if any)

```
create view negative_profit as

SELECT *

FROM sales_records

WHERE `Total Profit` < 0;

select * from negative_profit;
```

#14- Year-over-year revenue growth

```
create view yearly_revenew as

SELECT YEAR(`Order Date`) AS Year, SUM(`Total Revenue`) AS Revenue

FROM sales_records

GROUP BY YEAR(`Order Date`)

ORDER BY Year;

select * from yearly_revenew;
```

#15- Most frequently used payment method

```
create view most_used_payment_method as

SELECT `Payment Method`, COUNT(*) AS Count

FROM sales_records

GROUP BY `Payment Method`

ORDER BY Count DESC

LIMIT 1;

select * from most_used_payment_method;
```