

Code Glossary

Module 5 L2: Introduction to the Global Problem Statement

1. **To find the starting date and ending date of the dataset:**

```
SELECT MIN(pos_date), MAX(pos_date) FROM pos_data
```

2. **To find the revenue over the years from the dataset:**

```
SELECT YEAR(pos_date) AS date_year,  
       SUM(revenue) AS Total_Revenue  
FROM pos_data GROUP BY date_year;
```

3. **To compare the individual revenue of each segment with the total revenue over the years from the dataset:**

```
SELECT YEAR(pos_date) AS date_year,  
       SUM(revenue) AS Total_Revenue,  
       SUM(CASE WHEN segment = 'Hair Care' THEN revenue ELSE 0 END) AS  
       Hair_Care_Revenue,  
       SUM(CASE WHEN segment = 'Makeup' THEN revenue ELSE 0 END) AS  
       Makeup_Revenue,  
       SUM(CASE WHEN segment = 'Skincare' THEN revenue ELSE 0 END)  
       AS Skincare_Revenue  
FROM pos_data GROUP BY date_year;
```

4. **To find the % change in the total and individual revenue of each segment over the years from the dataset:**

```
SELECT  
    date_year,  
    Total_Revenue,  
    (Total_Revenue - LAG(Total_Revenue) OVER (ORDER BY date_year)) /  
    LAG(Total_Revenue) OVER (ORDER BY date_year) * 100 AS  
    Total_Percentage_Change,  
    Hair_Care_Revenue,  
    (Hair_Care_Revenue - LAG(Hair_Care_Revenue) OVER (ORDER BY  
    date_year)) / LAG(Hair_Care_Revenue) OVER (ORDER BY date_year) * 100 AS  
    Hair_Care_Percentage_Change,  
    Makeup_Revenue,  
    (Makeup_Revenue - LAG(Makeup_Revenue) OVER (ORDER BY date_year)) /  
    LAG(Makeup_Revenue) OVER (ORDER BY date_year) * 100 AS  
    Makeup_Percentage_Change,  
    Skincare_Revenue,
```

```

        (Skincare_Revenue - LAG(Skincare_Revenue) OVER (ORDER BY date_year))
        / LAG(Skincare_Revenue) OVER (ORDER BY date_year) * 100 AS
        Skincare_Percentage_Change
FROM (
        SELECT YEAR(pos_date) AS date_year,
        SUM(revenue) AS Total_Revenue,
        SUM(CASE WHEN segment = 'Hair Care' THEN revenue ELSE 0 END) AS
        Hair_Care_Revenue,
        SUM(CASE WHEN segment = 'Makeup' THEN revenue ELSE 0 END) AS
        Makeup_Revenue,
        SUM(CASE WHEN segment = 'Skincare' THEN revenue ELSE 0 END)
        AS Skincare_Revenue
        FROM pos_data GROUP BY date_year
    ) T1;

```

5. To find the unique number of products in the dataset:

```
SELECT COUNT(DISTINCT sku_id) FROM pos_data;
```

6. To find the revenue of each individual product:

```

SELECT
    SKU_ID,
    SUM('Revenue') AS Total_Revenue
FROM pos_data
GROUP BY SKU_ID
ORDER BY total_revenue desc;

```

7. To understand if revenue and traffic are co-related by comparing products having no revenue with their traffic:

```

SELECT
    SKU_ID,
    SUM('Revenue') AS Total_Revenue,
    SUM('Page_traffic') AS Total_Traffic
FROM pos_data
GROUP BY SKU_ID
ORDER BY total_revenue asc;

```

8. To split the products into 4 types (A,B,C and D) depending on the correlation between revenue and traffic:

```

SELECT
    sku_id, total_revenue, total_traffic,
    CASE
        when total_revenue!=0 and total_traffic!=0 then 'A'
        when total_revenue=0 and total_traffic=0 then 'B'
        when total_revenue!=0 and total_traffic=0 then 'C'

```

```

        when total_revenue=0 and total_traffic!=0 then 'D'
    END AS 'prod_type'
FROM
(
SELECT
    SKU_ID, SUM('Revenue') AS Total_Revenue, SUM('Page_traffic') AS
    Total_Traffic
FROM pos_data
GROUP BY SKU_ID
ORDER BY total_revenue asc
) t1;

```

9. To find the total traffic, total revenue and total products into A,B,C and D product types from the dataset:

```

SELECT prod_type, sum(total_traffic) AS Total_Traffic, sum(total_revenue) AS
Total_Revenue, count(prod_type) AS Total_Products FROM
(
SELECT
    sku_id, total_revenue, total_traffic,
    CASE
        when total_revenue!=0 and total_traffic!=0 then 'A'
        when total_revenue=0 and total_traffic=0 then 'B'
        when total_revenue!=0 and total_traffic=0 then 'C'
        when total_revenue=0 and total_traffic!=0 then 'D'
    END AS 'prod_type'
FROM
(
SELECT
    SKU_ID, SUM('Revenue') AS Total_Revenue, SUM('Page_traffic') AS
    Total_Traffic
FROM pos_data
GROUP BY SKU_ID
ORDER BY total_revenue asc
) t1
) t2
GROUP BY prod_type
ORDER BY prod_type;
SELECT sku_id, sum(num_unique_campaigns) AS total_campaigns FROM
online_data group by sku_id;

```

10. To find the number of campaigns for each of the products from the dataset:

```

SELECT sku_id, sum(num_unique_campaigns) AS total_campaigns FROM
online_data
GROUP BY sku_id;

```

11. To understand the correlation between the different products based on campaigns, total revenue and total traffic from the dataset.

```
SELECT t2.sku_id, total_revenue, total_traffic, total_campaigns FROM
(SELECT sku_id, sum(num_unique_campaigns) AS total_campaigns FROM
online_data group by sku_id) to
RIGHT JOIN
(
SELECT
sku_id,
total_revenue,
total_traffic,
CASE
    when total_revenue!=0 and total_traffic!=0 then 'A'
    when total_revenue=0 and total_traffic=0 then 'B'
    when total_revenue!=0 and total_traffic=0 then 'C'
    when total_revenue=0 and total_traffic!=0 then 'D'
END AS 'prod_type'
FROM
(
SELECT
    SKU_ID,
    SUM('Revenue') AS Total_Revenue,
    SUM('Page_traffic') AS Total_Traffic
FROM pos_data
GROUP BY SKU_ID
ORDER BY total_revenue asc) t1) t2
ON
to.sku_id = t2.sku_id;
```

12. To understand the correlation between the different product types A,B,C and D based on campaigns, total revenue and total traffic from the dataset.

```
SELECT prod_type, sum(total_campaigns) AS Total_Campaigns,
SUM(total_traffic) AS Total_Traffic,
SUM(total_revenue) AS Total_Revenue,
COUNT(prod_type) AS Total_Products
FROM
(
SELECT t2.sku_id, prod_type, total_revenue, total_traffic, Total_Campaigns
FROM
(SELECT sku_id, sum(num_unique_campaigns) AS Total_Campaigns FROM
online_data group by sku_id) to
RIGHT JOIN
(
SELECT sku_id, total_revenue, total_traffic,
CASE
```

```

        when total_revenue!=0 and total_traffic!=0 then 'A'
        when total_revenue=0 and total_traffic=0 then 'B'
        when total_revenue!=0 and total_traffic=0 then 'C'
        when total_revenue=0 and total_traffic!=0 then 'D'
    END AS 'prod_type'
FROM
(
    SELECT SKU_ID, SUM('Revenue') AS Total_Revenue, SUM('Page_traffic') AS
    Total_Traffic
    FROM pos_data
    GROUP BY SKU_ID
) t1
) t2
on
to.sku_id = t2.sku_id
) t3
GROUP BY prod_type
ORDER BY Total_Campaigns desc;

```

13. To find the the ratio of total traffic to total campaigns for each product type A,B,C and D from the dataset:

```

SELECT prod_type, sum(total_campaigns) AS Total_Campaigns,
SUM(total_traffic) AS Total_Traffic,
SUM(total_revenue) AS Total_Revenue,
count(prod_type) AS Total_Products,
SUM(total_traffic)/SUM(total_campaigns) AS Campaign_Ratio
FROM
(
    SELECT t2.sku_id, prod_type, total_revenue, total_traffic, Total_Campaigns
    FROM
    (SELECT sku_id, sum(num_unique_campaigns) AS Total_Campaigns from
    online_data group by sku_id) to
    RIGHT JOIN
    (
        SELECT sku_id, total_revenue, total_traffic,
        CASE
            when total_revenue!=0 and total_traffic!=0 then 'A'
            when total_revenue=0 and total_traffic=0 then 'B'
            when total_revenue!=0 and total_traffic=0 then 'C'
            when total_revenue=0 and total_traffic!=0 then 'D'
        END AS 'prod_type'
    FROM
    (
        SELECT SKU_ID, SUM('Revenue') AS Total_Revenue, SUM('Page_traffic') AS
        Total_Traffic
    ) t1
    ) t2
    on
    to.sku_id = t2.sku_id
) t3
GROUP BY prod_type
ORDER BY Total_Campaigns desc;

```

```
FROM pos_data
GROUP BY SKU_ID
) t1
) t2
on
to.sku_id = t2.sku_id
) t3
GROUP BY prod_type
ORDER BY Total_Campaigns desc;
```