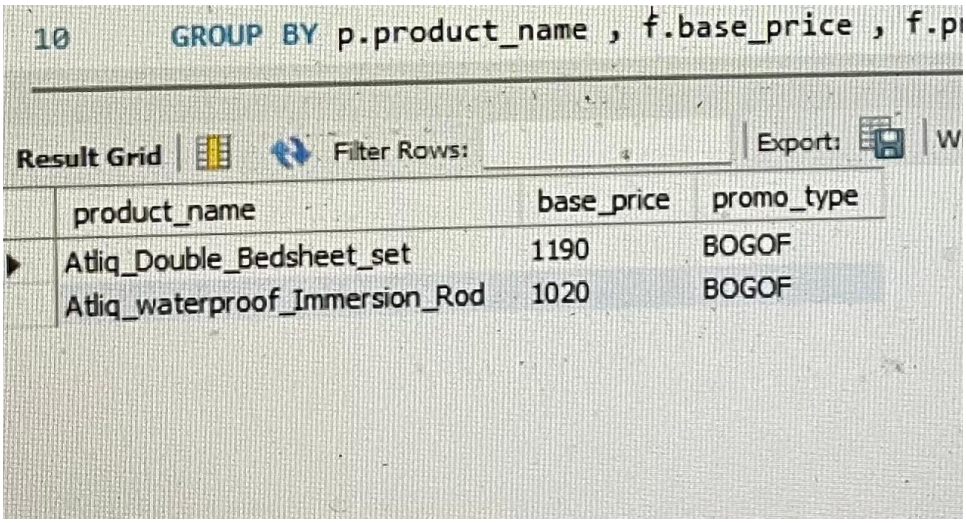
Analyse Promotions and Provide Tangible Insights to Sales Director

Ad-hoc Business Request

1)Provide a list of products with a base price greater than 500 and that are featured in promo type of 'BOGOF' (Buy One Get One Free). This information will help us identify high-value products that are currently being heavily discounted, which can be useful for evaluating our pricing and promotion strategies.

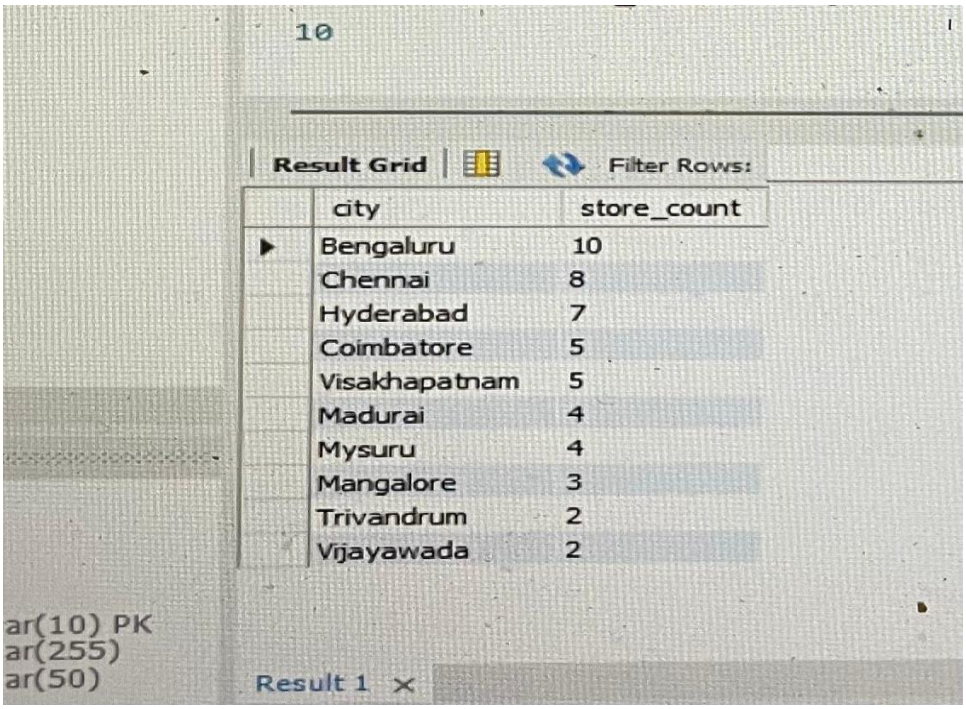
```
SELECT
   p.product_name, f.base_price, f.promo_type
FROM
   fact_events AS f
      JOIN
   dim_products AS p ON p.product_code = f.product_code
WHERE
   f.base_price > 500
      AND f.promo_type = 'BOGOF'
GROUP BY p.product_name , f.base_price , f.promo_type
ORDER BY p.product_name;
```



2. Generate a report that provides an overview of the number of stores in each city.

The results will be sorted in descending order of store counts, allowing us to identify the cities with the highest store presence. The report includes two essential fields: city and store count, which will assist in optimizing our retail operations.

```
Select
   city,
   COUNT(store_id) AS store_count
FROM
   dim_stores
GROUP BY
   city
ORDER BY
   store_count DESC;
```



Note: The quantity after promotion has been counted as 1 for the promo type BOGOF. So, we need to multiply the quantity by 2 to get the adjusted quantity after the promo. Additionally, we need to calculate the discounted price. For that, I have created a view

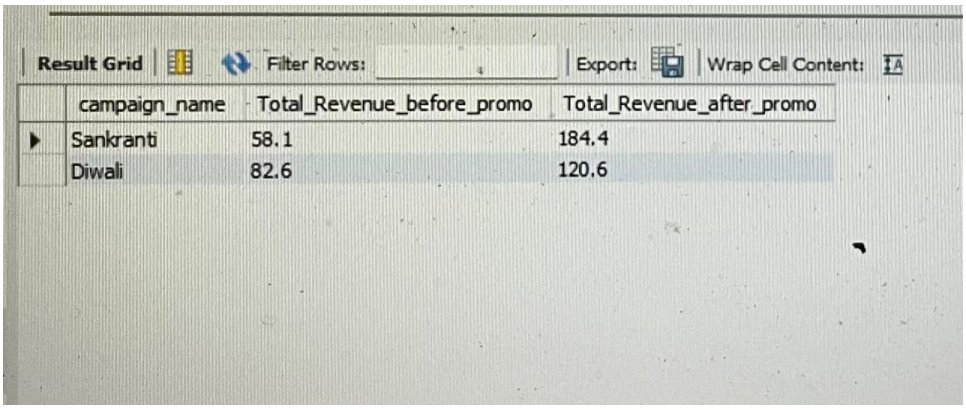
```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `discount_price` AS
    SELECT
        `fact_events`.`event_id` AS `event_id`,
        `fact_events`.`store_id` AS `store_id`,
        `fact_events`.`campaign_id` AS `campaign_id`,
        `fact_events`.`product_code` AS `product_code`,
        `fact_events`.`base_price` AS `base_price`,
```

```
    `fact_events`.`promo_type` AS `promo_type`,
    `fact_events`.`quantity_sold_before_promo` AS `quantity_sold_before_promo`,
    `fact_events`.`quantity_sold_after_promo` AS `quantity_sold_after_promo`,
    (CASE
      WHEN (`fact_events`.`promo_type` = '50% off') THEN (`fact_events`.`base_price` * 0.5)
      WHEN (`fact_events`.`promo_type` = '25% off') THEN (`fact_events`.`base_price` * 0.25)
      WHEN (`fact_events`.`promo_type` = '33% off') THEN (`fact_events`.`base_price` * 0.33)
      WHEN (`fact_events`.`promo_type` = 'BOGOF') THEN (`fact_events`.`base_price` * 0.5)
      WHEN (`fact_events`.`promo_type` = '500 Cashback') THEN 500
      ELSE `fact_events`.`base_price`
    END) AS `Discount_price`,
    (CASE
      WHEN (`fact_events`.`promo_type` = 'BOGOF') THEN (`fact_events`.`quantity_sold_after_promo` * 2)
      ELSE `fact_events`.`quantity_sold_before_promo`
    END) AS `Adjusted_quantity`
  FROM
    `fact_events`
```

3)Generate a report that displays each campaign along with the total revenue generated before and after the campaign? The report includes three key fields: campaign_name, total_ revenue (before_promotion), total_revenUe(after_promotion). This report should

help in evaluating the financial impact of our promotional campaigns. (Display the values in millions)

```
SELECT
   dc.campaign_name,
   ROUND(SUM(dp.base_price * dp.quantity_sold_before_promo) / 1000000,
        1) AS Total_Revenue_before_promo,
   ROUND(SUM((dp.base_price * dp.adjusted_quantity) - dp.Discount_price) / 1000000,
        1) AS Total_Revenue_after_promo
FROM
   discount_price AS dp
     JOIN
   dim_campaigns AS dc ON dc.campaign_id = dp.campaign_id
GROUP BY dc.campaign_name;
```

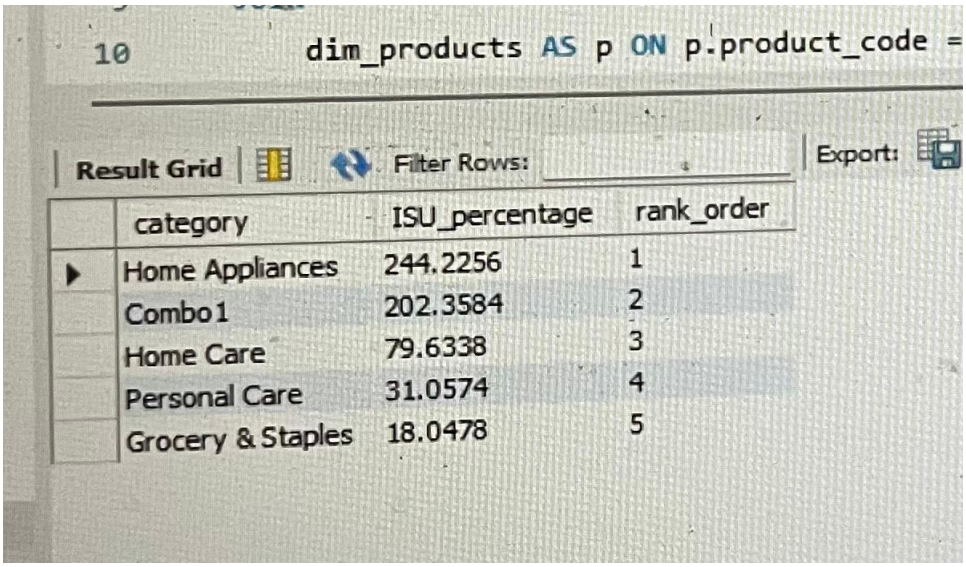


campaign_name	Total_Revenue_before_promo	Total_Revenue_after_promo
Sankranti	58.1	184.4
Diwali	82.6	120.6

4)Produce a report that calculates the Incremental Sold Quantity (ISU%) for each category during the Diwali campaign. Additionally, provide rankings for the categories based on their ISU%. The report will include three key fields: category, isu %, and rank order. This information will assist in assessing the category-wise success and impact of the Diwali campaign on incremental sales.

Note: ISU% (Incremental Sold Quantity Percentage) is calculated as the percentage increase/decrease in quantity sold (after promo) compared to quantity sold (before promo)

```
SELECT
   p.category,
   ((SUM(dp.quantity_sold_after_promo) - SUM(dp.quantity_sold_before_promo)) /
SUM(dp.quantity_sold_before_promo)) * 100 AS ISU_percentage,
   RANK() OVER (ORDER BY ((SUM(dp.quantity_sold_after_promo) -
SUM(dp.quantity_sold_before_promo)) / SUM(dp.quantity_sold_before_promo)) DESC) AS
rank_order
FROM
   discount_price AS dp
JOIN
   dim_campaigns AS dc ON dc.campaign_id = dp.campaign_id
JOIN
   dim_products AS p ON p.product_code = dp.product_code
WHERE
   dc.campaign_name = 'Diwali'
GROUP BY
   p.category
ORDER BY
   ISU_percentage DESC;
```



category	ISU_percentage	rank_order
Home Appliances	244.2256	1
Combo1	202.3584	2
Home Care	79.6338	3
Personal Care	31.0574	4
Grocery & Staples	18.0478	5

5. Create a report featuring the Top 5 products, ranked by Incremental Revenue Percentage (IR%), across all campaigns. The report will provide essential information including product name, category, and ir%. This analysis helps identify the most successful products in terms of incremental revenue across our campaigns, assisting in product optimization.

```
SELECT
   p.product_name,
   p.category,
   ((SUM(dp.base_price * dp.adjusted_quantity) - SUM(dp.base_price *
dp.quantity_sold_before_promo)) / SUM(dp.base_price * dp.quantity_sold_before_promo))
* 100 AS IR_percentage
FROM
   discount_price AS dp
      JOIN
   dim_products AS p ON p.product_code = dp.product_code
GROUP BY p.product_name , p.category
```

ORDER BY IR_percentage DESC

LIMIT 5;