

ATLIQ SUPERMARKET ANALYSIS

SQL Report



1. **Provide a list of products with a base price greater than 500 and featured in the promo type of 'BOGOF' (Buy One Get One Free).**

**Select**

**distinct** product\_name,

base\_price,

promo\_type

**from**

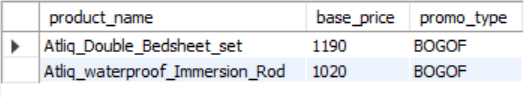
retail\_events\_db.dim\_products p

**join** retail\_events\_db.fact\_events fe **on** fe.product\_code = p.product\_code

**where**

base\_price > **500**

**and** promo\_type = "BOGOF";



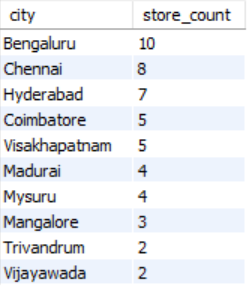
**2. Generate a report that provides an overview of the number of stores in each city. The results will be sorted in descending order of store counts, allowing us to identify the cities with the highest store presence. The report includes two essential fields: city and store count, which will assist in optimizing our retail operations.**

**Select** city,**count**(store\_id) **as** store\_count

**from** retail\_events\_db.dim\_stores

**group** **by** city

**order** **by** store\_count **desc**;

****

1. **Generate a report that displays each campaign and the total revenue generated before and after the campaign. The report includes three key fields: campaign\_name, total\_revenue (before\_promotion), and total\_revenue (after\_promotion). This report should help in evaluating the financial impact of our promotional campaigns. (Display the values in millions).**

**WITH** MY\_CTE **AS** (

**SELECT**

\*,

**CASE**

**WHEN** promo\_type = '50% OFF' **THEN** base\_price \* **0**.**5**

**WHEN** promo\_type = '25% OFF' **THEN** base\_price \* **0**.**75**

**WHEN** promo\_type = '33% OFF' **THEN** base\_price \* **0**.**67**

**WHEN** promo\_type = 'BOGOF' **THEN** base\_price \* **0**.**5**

**ELSE** base\_price - **500**

END **AS** Promotional\_price,

**CASE**

**WHEN** promo\_type = 'BOGOF' **THEN** `quantity\_sold(after\_promo)` \* **2**

**ELSE** `quantity\_sold(after\_promo)`

END **AS** AdjustedQuantity

**FROM**

retail\_events\_db.fact\_events

)

**SELECT**

campaign\_name,

**round**(**sum**(base\_price\*`quantity\_sold(before\_promo)`)/**1000000**,**2**) **as** 'Total revenue before promotion(M)',

**round**(**sum**(Promotional\_price\*AdjustedQuantity)/**1000000**,**2**) **as** 'Total revenue after promotion(M)'

**FROM**

MY\_CTE MC

**JOIN**

retail\_events\_db.dim\_campaigns c **ON** c.campaign\_id = MC.campaign\_id

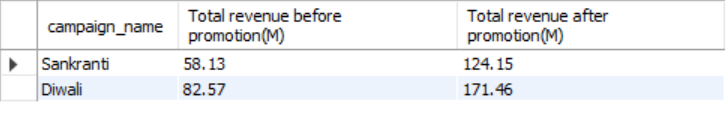
**GROUP** **BY**

campaign\_name

**ORDER** **BY**

'Total revenue before promotion(M)',

'Total revenue after promotion(M)' **DESC**;

****

**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.**

**WITH** MYT\_CTE **AS** (

**SELECT**

\*,

**CASE**

**WHEN** promo\_type = 'BOGOF' **THEN** `quantity\_sold(after\_promo)` \* **2**

**ELSE** `quantity\_sold(after\_promo)`

END **AS** AdjustedQuantity

**FROM**

retail\_events\_db.fact\_events

),

Ranked\_CTE **AS** (

**SELECT**

category,

**ROUND**((**SUM**(AdjustedQuantity) - **SUM**(`quantity\_sold(before\_promo)`)) \* **100** / **SUM**(`quantity\_sold(before\_promo)`), **1**) **AS** incremental\_sold\_quantity

**FROM**

MYT\_CTE

**JOIN** retail\_events\_db.dim\_products **AS** p **ON** p.product\_code = MYT\_CTE.product\_code

**WHERE**

campaign\_id = 'CAMP\_DIW\_01'

**GROUP** **BY**

category

)

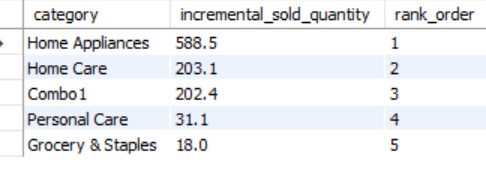
**SELECT**

\*,

**RANK**() **OVER**(**ORDER** **BY** incremental\_sold\_quantity **DESC**) **AS** rank\_order

**FROM**

Ranked\_CTE;



**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.**

**WITH** MY\_CTE **AS** (

**SELECT**

\*,

**CASE**

**WHEN** promo\_type = "50% OFF" **THEN** base\_price \* **0**.**5**

**WHEN** promo\_type = "25% OFF" **THEN** base\_price \* **0**.**75**

**WHEN** promo\_type = "33% OFF" **THEN** base\_price \* **0**.**67**

**WHEN** promo\_type = "BOGOF" **THEN** base\_price \* **0**.**5**

**ELSE** base\_price - **500**

END **AS** Promotional\_price,

**CASE**

**WHEN** promo\_type = 'BOGOF' **THEN** `quantity\_sold(after\_promo)` \* **2**

**ELSE** `quantity\_sold(after\_promo)`

END **AS** AdjustedQuantity

**FROM**

retail\_events\_db.fact\_events

)

**SELECT**

product\_name, category,

**ROUND**(

(**SUM**(Promotional\_price \* AdjustedQuantity) - **SUM**(base\_price \* `quantity\_sold(before\_promo)`))\***100**

/ **SUM**(base\_price \* `quantity\_sold(before\_promo)`),

**2**

) **AS** revenue\_change

**FROM**

MY\_CTE MC

**JOIN** retail\_events\_db.dim\_products p **ON** p.product\_code = MC.product\_code

**GROUP** **BY**

product\_name,category

**ORDER** **BY**

revenue\_change **DESC**

**limit** **5**;

