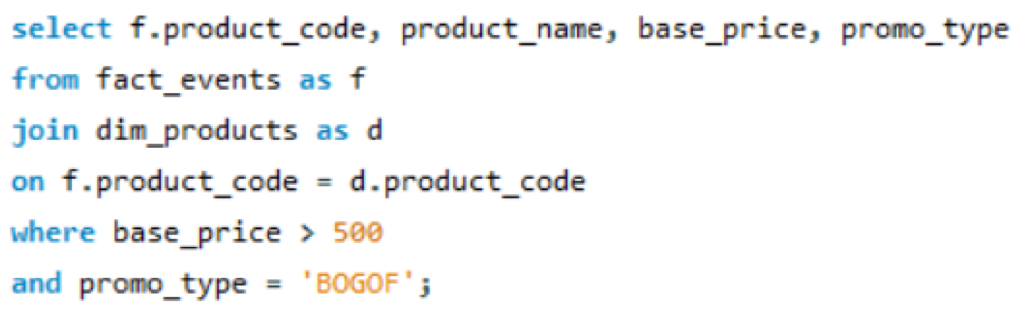
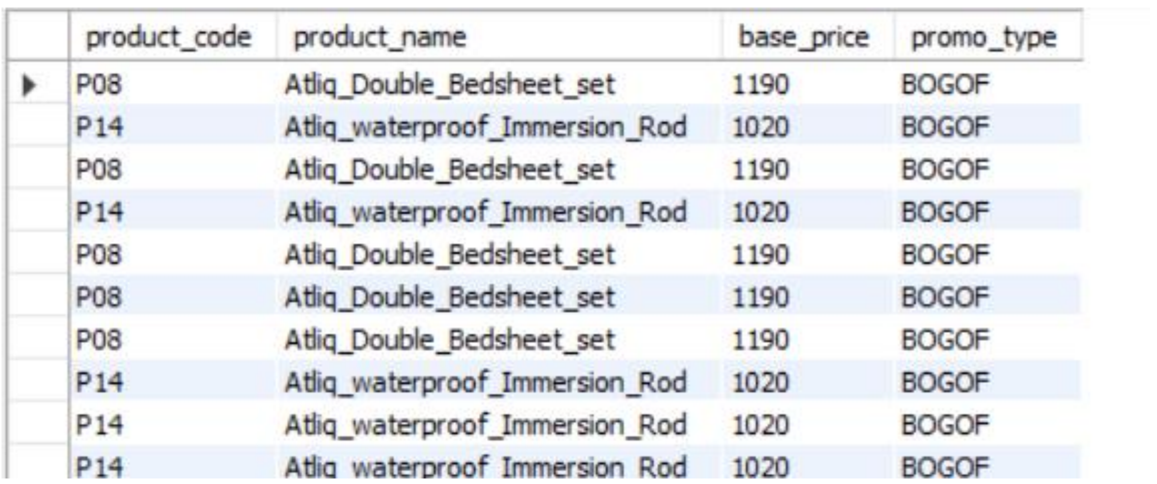
Request 1



```
select f.product_code, product_name, base_price, promo_type
from fact_events as f
join dim_products as d
on f.product_code = d.product_code
where base_price > 500
and promo_type = 'BOGOF';
```

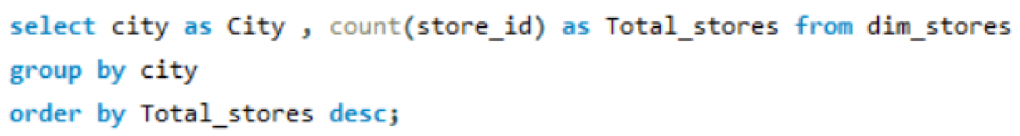
#Used JOIN to join dim products with facts event table to obtain distinct product name

#Used WHERE to implement the conditions like base price 500 and prone type as "BOGOF”



product_code	product_name	base_price	promo_type
P08	Atliq_Double_Bedsheet_set	1190	BOGOF
P14	Atliq_waterproof_Immersion_Rod	1020	BOGOF
P08	Atliq_Double_Bedsheet_set	1190	BOGOF
P14	Atliq_waterproof_Immersion_Rod	1020	BOGOF
P08	Atliq_Double_Bedsheet_set	1190	BOGOF
P08	Atliq_Double_Bedsheet_set	1190	BOGOF
P08	Atliq_Double_Bedsheet_set	1190	BOGOF
P14	Atliq_waterproof_Immersion_Rod	1020	BOGOF
P14	Atliq_waterproof_Immersion_Rod	1020	BOGOF
P14	Atliq_waterproof_Immersion_Rod	1020	BOGOF

Request 2

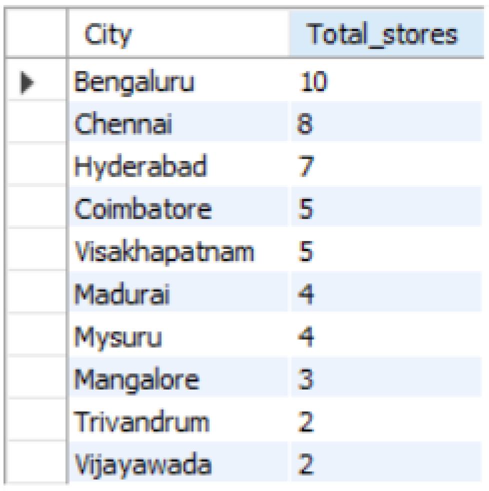


```
select city as City , count(store_id) as Total_stores from dim_stores
group by city
order by Total_stores desc;
```

#Used GROUPBY to group stores that belonged to same city

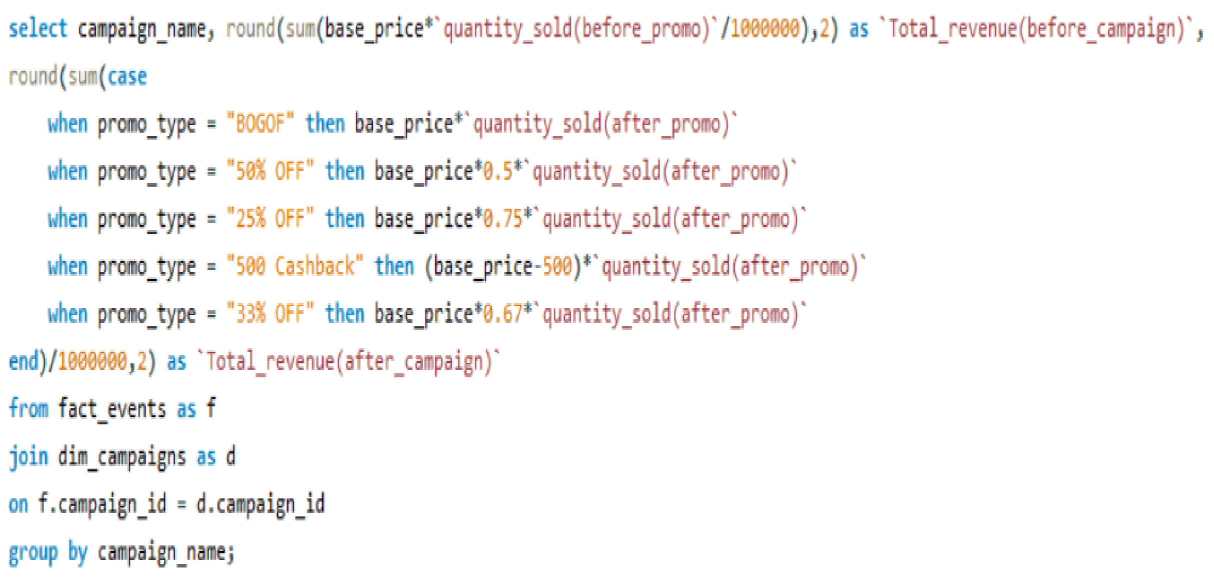
#Used COUNT to count the number of stores

#used ORDERBY to arrange the number of stores in an descending order



City	Total_stores
Bengaluru	10
Chennai	8
Hyderabad	7
Coimbatore	5
Visakhapatnam	5
Madurai	4
Mysuru	4
Mangalore	3
Trivandrum	2
Vijayawada	2

Request 3



```
select campaign_name, round(sum(base_price*`quantity_sold(before_promo)`/1000000),2) as `Total_revenue(before_campaign)`,
round(sum(case
    when promo_type = "BOGOF" then base_price*`quantity_sold(after_promo)`
    when promo_type = "50% OFF" then base_price*0.5*`quantity_sold(after_promo)`
    when promo_type = "25% OFF" then base_price*0.75*`quantity_sold(after_promo)`
    when promo_type = "500 Cashback" then (base_price-500)*`quantity_sold(after_promo)`
    when promo_type = "33% OFF" then base_price*0.67*`quantity_sold(after_promo)`
end)/1000000,2) as `Total_revenue(after_campaign)`
from fact_events as f
join dim_campaigns as d
on f.campaign_id = d.campaign_id
group by campaign_name;
```

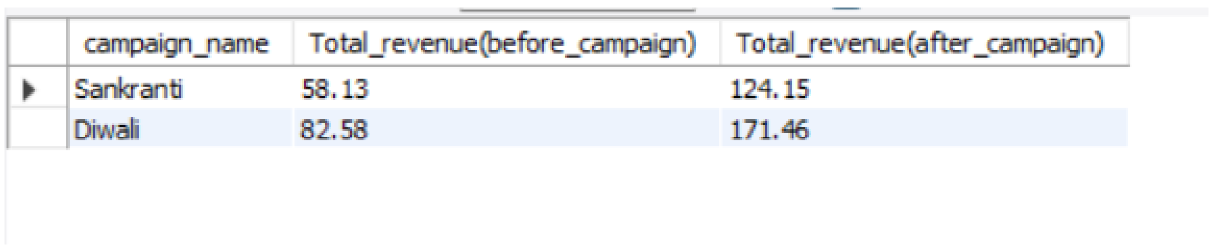
# SUM - to add all the revenues obtained before promotion

# ROUND - to round the number to the specified number of decimals

# CONCAT - to add M (denoting Millions) to the revenue value

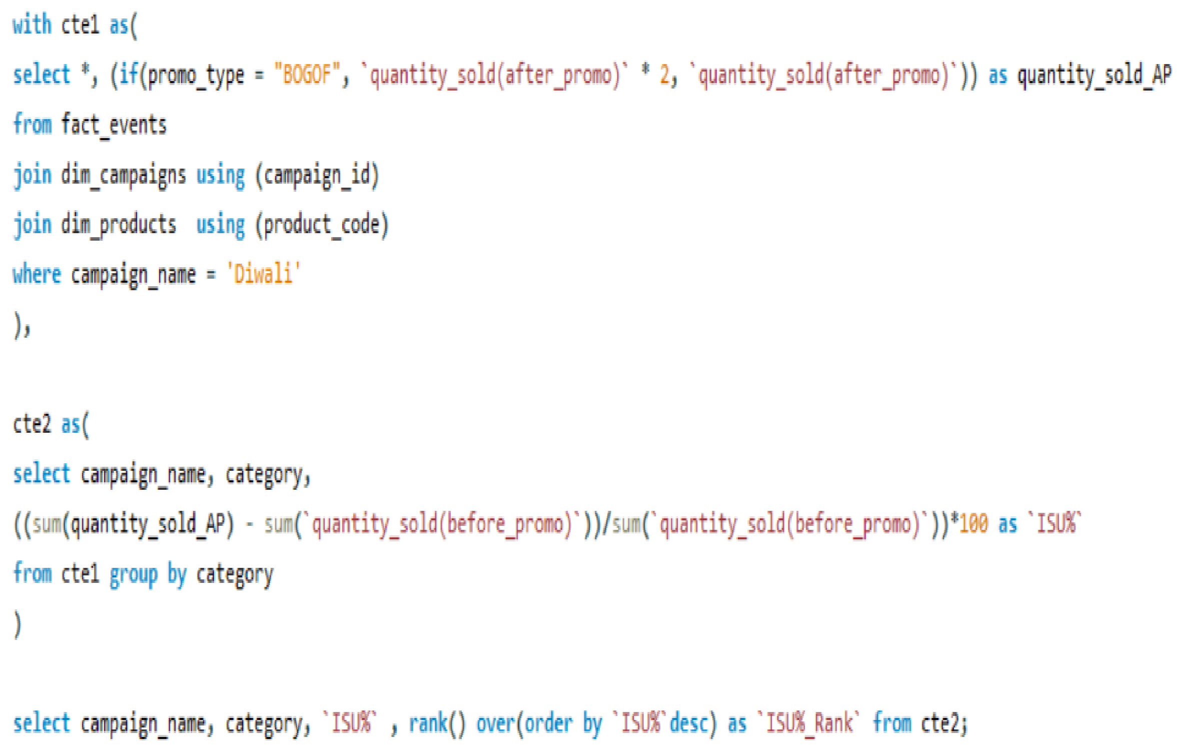
# CASE - to calculate revenue after promotion based on different promo_types

# JOIN- to join dim_campaigns table with facts table to obtain the campaign_name



campaign_name	Total_revenue(before_campaign)	Total_revenue(after_campaign)
Sankranti	58.13	124.15
Diwali	82.58	171.46

Request 4



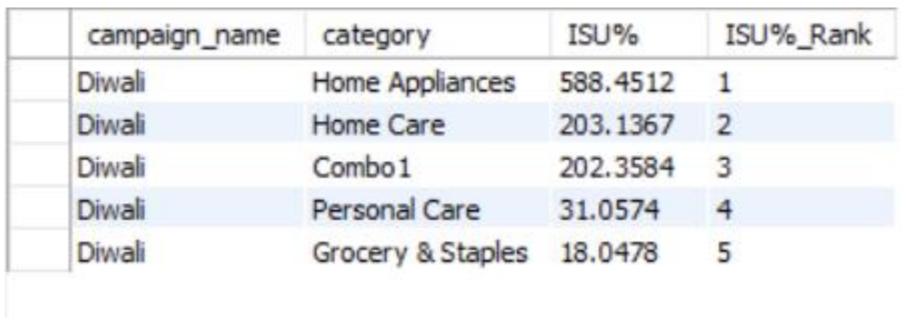
```
with cte1 as(
select *, (if(promo_type = "BOGOF", `quantity_sold(after_promo)` * 2, `quantity_sold(after_promo)`)) as quantity_sold_AP
from fact_events
join dim_campaigns using (campaign_id)
join dim_products  using (product_code)
where campaign_name = 'Diwali'
),

cte2 as(
select campaign_name, category,
((sum(quantity_sold_AP) - sum(`quantity_sold(before_promo)`))/sum(`quantity_sold(before_promo)`))*100 as `ISU%`
from cte1 group by category
)

select campaign_name, category, `ISU%` , rank() over(order by `ISU%`desc) as `ISU%_Rank` from cte2;
```

#CTE1 - used Common_Table_Expression to double the quantities, if the promotion type = "BOGOF"

#CTE2 - to calculate the Incremental Sold Units % and GROUPY to group the products based on their category from cte1

#SELECT - to determine campaign name, category from cte2

#RANK() - used window function to obtain the ranks of the categories based on their ISU%



campaign_name	category	ISU%	ISU%_Rank
Diwali	Home Appliances	588.4512	1
Diwali	Home Care	203.1367	2
Diwali	Combo1	202.3584	3
Diwali	Personal Care	31.0574	4
Diwali	Grocery & Staples	18.0478	5