SQL Case Study – 1

Tasks

1.Display the number of states present in the LocationTable.
```
SELECT COUNT(distinct state) Total_state
from Location
```

2.How many products are of regular type?
```
SELECT COUNT(*) Total_Product FROM Product
WHERE type = 'Regular'
```

3.How much spending has been done on marketing of product ID 1?
```
SELECT SUM(Marketing) Total_spend
from fact
WHERE ProductId = 1
```

4.What is the minimum sales of a product?
```
SELECT MIN(Sales) Min_Sale
from fact
```

5.Display the max Cost of Good Sold (COGS).
```
SELECT MAX(cogs) Max_Cost
from fact
```

6.Display the details of the product where product type is coffee.
```
SELECT *FROM Product
WHERE Product_Type = 'coffee'
```

7.Display the details where total expenses are greater than 40.
```
SELECT *
from fact
WHERE Total_Expenses>40
ORDER by Total_Expenses ASC
```

8.What is the average sales in area code 719?
```
SELECT AVG(Sales) avg_sales
from fact
WHERE Area_Code = 719
```

9.Find out the total profit generated by Colorado state.
```
SELECT SUM(Profit) total_Profit
from fact f INNER JOIN location l
on (f.Area_Code = l.Area_Code)
WHERE l.state = 'Colorado'
```

10.Display the average inventory for each product ID.
```
SELECT productid,AVG(Inventory) AVG_Inventory
from fact
GROUP by ProductId
ORDER by ProductId ASC
```
11.Display state in a sequential order in a Location Table.

```
SELECT distinct state
from location
ORDER by state ASC
```

12.Display the average budget of the Product where the average budget margin should be greater than 100.

```
SELECT ProductId , AVG(Budget_Margin) Avg_Budget
from fact
GROUP by productid
HAVING AVG(Budget_Margin) >100
```

13.What is the total sales done on date 2010-01-01?

```
SELECT SUM(Sales) Total_Sal
from fact
WHERE Date = '2010-01-01'
```

14.Display the average total expense of each product ID on an individual date.

```
SELECT Productid,date, AVG(Total_Expenses) Avg_Total_Expense
from fact
GROUP by ProductId,date
ORDER by ProductId ASC
```

15.Display the table with the following attributes such as date, product ID, product _ type, product, sales, profit, state, area code.

```
SELECT f.date , f.ProductId,f.Profit,f.Sales, p.Product_Type,p.Product,l.state ,l.Area_Code
from fact f INNER JOIN Product p
on (f.ProductId= p.ProductId) INNER JOIN
location l ON(f.Area_Code=l.Area_Code)
```

16.Display the rank without any gap to show the sales wise rank.

```
SELECT sales ,l.state,DENSE_RANK() OVER ( order by sales desc)
from fact
```

17.Find the state wise profit and sales.

```
SELECT l.state , SUM(f.Profit) State_Profit ,SUM(f.sales) State_Total_Sales
from fact f INNER JOIN location l
on ( f.Area_code = l.Area_code)
GROUP by l.state
```

18.Find the state wise profit and sales along with the product name.

```
SELECT l.state , SUM(f.Profit) State_Total_Profit,SUM(f.sales) State_Total_Sales , p.Product
from fact f INNER JOIN Product p
on (f.ProductId= p.ProductId) INNER JOIN
location l ON(f.Area_Code=l.Area_Code)
GROUP by l.state ,p.Product
```

19.If there is an increase in sales of 5%, calculate the increasedsales.

```
SELECT sales , sales*1.05
from fact
```

20.Find the maximum profit along with the product ID and producttype.

```
SELECT p.ProductId ,p.Product_Type,MAX(f.Profit) max_Profit
from fact f INNER JOIN Product p
on (f.ProductId= p.ProductId)
GROUP by p.ProductId ,p.Product_Type
```

21.Create a stored procedure to fetch the result according to the product type from Product Table.

```
CREATE PROCEDURE productByType
@product NVARCHAR(50)
as
BEGIN
SELECT *FROM Product
WHERE Product_Type = @product
END
EXEC productByType 'coffee'
EXEC productByType @product = 'coffee'
```

22.Write a query by creating a condition in which if the total expenses is less than 60 then it is a profit or else loss.

```
SELECT Total_Expenses ,
case
when Total_Expenses<60 then 'Profit'
Else 'Loss'
END as result
from fact
```

23.Give the total weekly sales value with the date and product ID details. Use roll-up to pull the data in hierarchical order

```
SELECT DATEPART(week ,date ) week_no ,productid,sum(sales) total_sales from fact
GROUP by ROLLUP (DATEPART(week,date),productid)
```

24.Apply union and intersection operator on the tables which consist of attribute area code.

```
select area_code from fact
UNION
SELECT area_code from location


select area_code from fact
INTERSECT
SELECT area_code from location
```

25.Create a user-defined function for the product table to fetch a particular product type based upon the user's preference.

```
create or ALTER FUNCTION getpt (@pt NVARCHAR (50))
```

```
returns TABLE
as
RETURN
(
SELECT *
from Product
where Product_Type = @pt
);

SELECT * from dbo.getpt('Coffee')
```

26.Change the product type from coffee to tea where product IDis 1 and undo it.

```
BEGIN TRANSACTION
UPDATE Product
set Product_Type= 'tea'
WHERE ProductId = 1

ROLLBACK
```

27.Display the date, product ID and sales where total expenses are between 100 to 200.

```
SELECT date , ProductId,Sales
from fact
WHERE Total_Expenses BETWEEN 100 and 200
```

28.Delete the records in the Product Table for regular type.

```
delete from Product
WHERE type = 'Regular'
```

29.Display the ASCII value of the fifth character from the columnProduct.

```
SELECT Product , ASCII(SUBSTRING(Product,5,1))from Product
```