Assignment: SQL - Case Study 1

Problem Statement:

You are a database administrator. You want to use the data to answer a few questions about your customers, especially about the sales and profit coming from different states, money spent in marketing and various other factors such as COGS (Cost of Goods Sold), budget profit etc. You plan on using these insights to help find out which items are being sold the most. You have been provided with the sample of the overall customer data due to privacy issues. But you hope that these samples are enough for you to write fully functioning SQL queries to help answer the questions.

Dataset:

The 3 key datasets for this case study:

a. **Fact Table:**

The Fact Table has 14 columns mentioned below and 4200 rows. Date, ProductID, Profit, Sales, Margin, COGS, Total Expenses, Marketing, Inventory, Budget Profit, Budget COGS, Budget Margin, Budget Sales, and Area Code

Note: COGS stands for Cost of Goods Sold

b. **Product Table:**

The Product Table has four columns named Product Type, Product, ProductID, and Type. It has 13 rows which can be broken down into further details to retrieve the information mentioned in the Fact Table.

c. **Location Table:**

Finally, the Location Table has 156 rows and follows a similar approach to Product Table. It has four columns named Area Code, State, Market, and Market Size.

```
-- Tasks Performed

use Assignments;

select * from fact$;

select * from Location$;

select * from Product$;

-- 1. Display the number of states present in the Location Table.

        select COUNT(State) from Location$;
```

```
-- 2. How many products are of regular type?

       select COUNT(ProductId) from fact$ where Type = 'Regular';

-- 3. How much spending has been done on marketing of product ID 1?

       EXEC sp_rename 'fact$.[Total Expenses]', 'TotalExpenses', 'COLUMN';

       select * from fact$

       select Marketing, TotalExpenses from fact$ where ProductId = 1;


-- 4. What is the minimum sales of a product?

       select MIN(Sales) from fact$;

-- 5. Display the max Cost of Good Sold (COGS).

       select MAX(COGS) from fact$;

-- 6. Display the details of the product where product type is coffee.

       EXEC sp_rename 'Product$.[Product Type]', 'ProductType', 'COLUMN';

       select * from Product$ where ProductType = 'Coffee'

-- 7. Display the details where total expenses are greater than 40.

       select * from fact$ where TotalExpenses > 40

-- 8. What is the average sales in area code 719?

       EXEC sp_rename 'fact$.[Area Code]', 'AreaCode', 'COLUMN';

       select AVG(AreaCode) from fact$ where AreaCode = 719;

-- 9. Find out the total profit generated by Colorado state.
       EXEC sp_rename 'Location$.[Area Code]', 'AreaCode', 'COLUMN';

       SELECT SUM(f.Profit) AS TotalProfit, L.State FROM fact$ f
       JOIN Location$ L ON f.AreaCode = L.AreaCode WHERE L.State = 'Colorado'
       GROUP BY L.State;

-- 10. Display the average inventory for each product ID.

       select AVG(Inventory) from fact$ where ProductID = 1;

-- 11. Display state in a sequential order in a Location Table.

       select 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.

       EXEC sp_rename 'fact$.[Budget Sales]', 'BudgetSales', 'COLUMN';
       EXEC sp_rename 'fact$.[Budget Margin]', 'BudgetMargin', 'COLUMN';
```

```sql
        select AVG(BudgetSales) from fact$ where BudgetMargin > 100;

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

        select SUM(Sales) 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(TotalExpenses) AS AverageExpense FROM fact$ GROUP BY
        ProductId, Date;


-- 15. Display the table with the following attributes such as date, productID,
        product_type, product, sales, profit, state, area_code.

        SELECT f.Date, f.ProductId, p.ProductType, p.Product, f.Sales, f.Profit, l.State,
        l.AreaCode FROM fact$ f JOIN Location$ l ON l.AreaCode = f.AreaCode
        JOIN Product$ p ON f.ProductId = p.productId;

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

        SELECT f.Date, f.ProductId, p.ProductType, p.Product, f.Sales, f.Profit, l.State,
        l.AreaCode (select COUNT(DISTINCT f2.Sales) FROM fact$ f2 WHERE f2.Sales >=
        f.Sales) + 1 AS SalesRank) AS SalesRank
        FROM fact$ f JOIN Location$ l ON l.AreaCode = f.AreaCode JOIN Product$ p ON
        f.ProductId = p.productId; ORDER BY f.Sales DESC;

-- 17. Find the state wise profit and sales.

        select f.Profit, f. Sales, l.State from fact$ f join Location$ l ON l.AreaCode =
        f.AreaCode;

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

        select p.Product, f.Profit, f. Sales, l.State from fact$ f
        join Location$ l ON l.AreaCode = f.AreaCode
        join Product$ p ON p.ProductId = f.ProductId


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

        SELECT ProductId, Sales * .05 AS IncreasedSales FROM fact$;

-- 20. Find the maximum profit along with the product ID and product type.

        SELECT f.productID, p.Type, MAX(f.profit) AS MaximumProfit FROM fact$ f
        JOIN Product$ p ON f.productID = p.productID GROUP BY f.productId, p.Type;

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

        IF OBJECT_ID('GetProductsByType', 'P') IS NOT NULL
        DROP PROCEDURE GetProductsByType;

        SELECT ProductId, Type, Product, ProductType FROM Product$ WHERE ProductType =
        'Coffee';
        CREATE PROCEDURE GetProductsByType
                @ProductType NVARCHAR(50)
                     AS
                     BEGIN
                     PRINT 'Procedure Start';

         SELECT ProductId, ProductType, Product, Type FROM Product$ WHERE ProductType =
        @ProductType;
               PRINT 'Procedure End';
        END;

        EXEC GetProductsByType @ProductType = '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 ProductId, SUM(TotalExpenses) AS TotalExpenses,
               CASE
                 WHEN SUM(TotalExpenses) < 60 THEN 'Profit'
                 ELSE 'Loss'
               END AS ProfitOrLoss
        FROM fact$ GROUP BY ProductId

-- 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 Date, ProductId, SUM(Sales) AS WeeklySales FROM fact$ GROUP BY ROLLUP(Date,
        ProductId);

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

        SELECT AreaCode FROM fact$ UNION SELECT AreaCode FROM Location$;

        SELECT AreaCode FROM fact$ INTERSECT SELECT AreaCode 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 FUNCTION GetProductsByType1
        (
               @ProductType NVARCHAR(50)
        )
        RETURNS TABLE
        AS
        RETURN
        (
               SELECT ProductID, ProductType, Product FROM Product$ WHERE ProductType =
               @ProductType
        );

        SELECT * FROM dbo.GetProductsByType1('Coffee');

-- 26. Change the product type from coffee to tea where product ID is 1 and undo it.

        -- Start a transaction
        BEGIN TRANSACTION;
        -- Change product type to 'tea' for ProductID 1
        UPDATE Product$
        SET ProductType = 'tea'
        WHERE ProductId = 1;

        -- Check the updated values
        SELECT * FROM Product$ WHERE ProductId = 1;

        -- Rollback the transaction to undo the change
        ROLLBACK TRANSACTION;

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

        SELECT Date, ProductId, Sales FROM fact$ WHERE TotalExpenses BETWEEN 100 AND 200;

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

        DELETE FROM Product$ WHERE ProductType = 'regular';

        SELECT * FROM Product$ WHERE ProductType = 'regular';

-- 29. Display the ASCII value of the fifth character from the column Product.

        SELECT  Product,  ASCII(SUBSTRING(Product,  5,  1))  AS  FifthCharacterASCII  FROM
        Product$;
```