SUPERMARKETBILLLINGSYSTEM

OBJECTIVES OF THE PROJECT :

It is Design to stremline and Enhance the checkout the process, improve accuracy and provide valuable insights for management.

- Here we can see the customerslist and billing process of their products
- We can see here how the sales is going on .
- We can observe the different categories of the products
- The Dataset contain information about the Customers, Categories, Prducts, Sales, Saleitems, And Discounts
- By analyzing this data we aim to uncover and insights that can be improve business operations and decision making.

Create Table:

Write SQL statements to create all the tables with the specified columns and foreign key references.

create customer table :

1. **Customers**

```
Create table Customers(
CustomerID INT Primary key,
FirstName VARCHAR(20),
LastName VARCHAR(20),
Email VARCHAR(20),
Phone VARCHAR(15),
RegistrationDate DATE);
```

2. Products

```sql
create table Products(
ProductID INT PRIMARY KEY,
ProductName VARCHAR(250) NOT NULL,
Category VARCHAR(300),
Price DECIMAL(10,2) not null,
StockQuantity INt not null
);
```

3. Category

```sql
CategoryID INT PRIMARY KEY,
CategoryName VARCHAR(250)
);
```

4. Sales

```sql
create table Sales(
SaleID INT PRIMARY KEY,
customerID INT,
SaleDate DATE,
TotalAmount DECIMAL(10,2),S
Foreign key (customerID) References Customers(customerID)
);
```

5. Saleitems

```sql
create table Saleitems(
SaleitemID INT PRIMARY KEY,
SaleID INT,
ProductID INT,
Quantity INT,
Price DECIMAL(10.2),
FOREIGN KEY (SaleID) REFERENCES Sales(SaleID),
FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
```

6.Discounts

```
create table Discounts(
DiscountID INT PRIMARY KEY,
ProductID INT,
DiscountPercentage DECIMAL(5,2),
StartDate DATE,
EndDate DATE,
Foreign key (ProductID) REFERENCES Products(ProductID)
);
```

Inserting the values into tables

Customers:

Insert values into customers

```
insert into Customers( CustomerID, FirsrtName, LastName, Email, Phone, RegistrationDate)
values(1, 'Varma', 'Raj', 'Varma.Raj@mycomapany.com', 9877887766, 2023-01-15),
(2, 'Raj', 'Reddy', 'Raj.Reddy@mycompany.com', 988776654, 202-02-21),
(3, 'Mike', 'Joe', 'Mike.Joe@mycompany.com', 78887787878, 2023-03-25),
(4, 'Scott', 'James', 'Scott.James@mycompany.com', 76656565656, 2023-04-0),
(5, 'Marry', 'Comm', 'Marry.Comm@mycompany.com', 9889876656, 2023-05-23),
(6, 'Shiny', 'Rani', 'Shiny.Rani@mycompany.com', 8778787878, 2023-06-27),
(7, 'Mikky', 'Roy', 'Mikky.Roy@mycompany.com', 9887878788, 2023-07-12),
(8, 'Rose', 'Mary', 'Rose.Mary@mycompany.com', 78878656765, 2023-08-28),
(9, 'Mike', 'Jussy', 'Mike.Jussy@mycompany.com', 9889878787, 2023-09-0),
(10, 'Yashik', 'rao', 'Yashik.Rao@mycompany.com', 98878787788, 2023-10-25);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap

customerID	FirstName	LastName	Email	Phone	RegistrationDate
1	Varma	Raj	Varma.Raj@mycomapny.com	9877887766	2023-01-15
2	Raj	Reddy	Raj.Reddy@mycompany.com	988776654	2023-02-21
3	Mike	Joe	Mike.Joe@mycompany.com	78887787878	2023-03-25
4	Scott	James	Scott.James@mycompany.com	76656565656	2023-04-30
5	Marry	Comm	Marry.Comm@mycompany.com	9889876656	2023-05-23
6	Shiny	Rani	Shainy.Rani@mycompany.com	8778787878	2023-06-27
7	Mikky	Roy	Mikky.Roy@mycompany.com	9887878788	2023-07-12
8	Rose	Mary	Rose.Mary@mycompany.com	78878656756	2023-08-28
9	Mike	Jussy	Mike.Jussy@mycompany.com	9889878787	2023-09-30
10	Yashik	Rao	Yashik.Rao@mycompany.com	9887878788	2023-10-25

Products:

```sql
--Insert values Products---
Insert into Products(ProductID, ProductName, Category, Price, StockQuantity)
values(111, 'Banana', 'Fruit', 12.2, 120),
(112, 'Milk', 'Dairy', 10.2, 50),
(113, 'Muffins', 'Bakery', 50.2, 20),
(114,  'Chicken', 'Meat', 35.4, 25),
(115, 'Egg' , 'Dairy', 12.4, 100),
(116, 'Cabbage', 'Vegetables', 15.5, 80),
(118, 'Bread', 'Bakery', 50.5, 150),
(117, 'Toothpaste', 'Persnalcare', 120.0, 120),
(119, 'Rice', 'Grains', 200.3, 180),
(120, 'Apple', 'Fruit', 40.5, 160);
```

ProductID	ProductName	Category	Price	StockQuantity
111	Banana	Fruit	12.20	120
112	Milk	Dairy	10.20	50
113	Muffins	Bakery	50.20	20
114	Chicken	Meat	35.40	25
115	Egg	Dairy	12.40	100
116	Cabbage	Vegetables	15.50	80
117	Toothpaste	Persnalcare	120.00	120
118	Bread	Bakery	50.50	150
119	Rice	Grains	200.30	180
120	Apple	Fruit	40.50	160

categories

```sql
---Insert values Categories;---
Insert  into Categories(CategoryID, CategoryName)
values (501, 'Dairy'),
(502, 'Bakery'),
(503, 'Fruits'),
(504, 'Vegetables'),
(505, 'Meat'),
(506, 'Presonnalcare'),
(507, 'Grains'),
(508, 'Frozen'),
(509, 'Pulses'),
(510, 'Snacks');
```

CategoryID	CategoryName
501	Dairy
502	Bakery
503	Fruits
504	Vegetables
505	Meat
506	Presonnalcare
507	Grains
508	Frozen
509	Pulses
510	Snacks

Sales:

```
---insert values sales---------
Insert into Sales(SaleID, customerID, SaleDate, TotalAmount)
Values(221, 1, '2024-08-1', 45),
(222, 2, '2024-08-2', 55),
(223, 3, '2024-08-3', 100),
(224, 4, '2024-08-4', 150),
(225, 5, '2024-08-5', 240),
(226, 6, '2024-08-6', 300),
(227, 7, '2024-08-7', 350),
(228, 8, '2024-08-8', 250),
(229, 9, '2024-08-9', 360),
(230, 10, '2024-08-10', 120);
```

Result Grid | Filter Rows:

SaleID	customerID	SaleDate	TotalAmount
221	1	2024-08-01	45.00
222	2	2024-08-02	55.00
223	3	2024-08-03	100.00
224	4	2024-08-04	150.00
225	5	2024-08-05	240.00
226	6	2024-08-06	300.00
227	7	2024-08-07	350.00
228	8	2024-08-08	250.00
229	9	2024-08-09	360.00
230	10	2024-08-10	120.00

saleitems

```
---Insert values Saleitems--
  (400, 230, 120, 15, 15.15)
Insert into Saleitems( SaleitemID, SaleID, ProductID, Quantity, Price)
  values(401, 221, 111, 50, 10.80),
  (402, 222, 112, 60, 15.20),
  (403, 223, 113, 124, 45.08),
  (404, 224, 114, 60, 476.66),
  (405, 225, 115, 45, 12.33),
  (406, 226, 116, 54, 23.43),
  (407, 227, 117, 20, 45.52),
  (408, 228, 118, 35, 20.12),
  (409, 229, 119, 25, 35.22),
  (410, 230, 120, 35, 15.23);
```

esult Grid | Filter Rows: | Edit:

SaleitemID	SaleID	ProductID	Quantity	Price
401	221	111	50	11.00
402	222	112	60	15.00
403	223	113	124	45.00
404	224	114	60	477.00
405	225	115	45	12.00
406	226	116	54	23.00
407	227	117	20	46.00
408	228	118	35	20.00
409	229	119	25	35.00
410	230	120	35	15.00

Discounts

```
 insert vales in Dsicounts
 Insert into Discounts(DiscountID, ProductID, DiscountPercentage, StartDate, EndDate)
values(2001, 111, 55.45, '2024-01-1', '2024-1-31'),
(2002, 112, 23.32, '2024-02-1', '2024-02-28'),
(2003, 113, 45.43, '2024-03-1', '2024-03-30'),
(2004, 114, 44.34, '2024-04-1', '2024-04-30'),
(2005, 115, 55.64, '2024-05-1', '2024-05-31'),
(2006, 116, 34.55, '2024-06-1', '2024-06-30'),
(2007, 117, 34.54, '2024-07-1', '2024-07-31'),
(2008, 118, 55.43, '2024-08-1', '2024-08-31'),
(2009, 119, 23.43, '2024-09-1', '2024-09-30'),
(2010, 120, 44.43, '2024-10-1', '2024-10-31');
```

DiscountID	ProductID	DiscountPercentage	StartDate	EndDate
2001	111	55.45	2024-01-01	2024-01-31
2002	112	23.32	2024-02-01	2024-02-28
2003	113	45.43	2024-03-01	2024-03-30
2004	114	44.34	2024-04-01	2024-04-30
2005	115	55.64	2024-05-01	2024-05-31
2006	116	34.55	2024-06-01	2024-06-30
2007	117	34.54	2024-07-01	2024-07-31
2008	118	55.43	2024-08-01	2024-08-31
2009	119	23.43	2024-09-01	2024-09-30
2010	120	44.43	2024-10-01	2024-10-31

3.Write a query to select all products from the Products table where the Price is greater than $50.

```
Select * From ProductsS
where Price > 50;
```

Filter Rows: Edit:

ProductID	ProductName	Category	Price	StockQuan
S	Muffins	Bakery	50.20	20
117	Toothpaste	Persnalcare	120.00	120
118	Bread	Bakery	50.50	150
119	Rice	Grains	200.30	180
NULL	NULL	NULL	NULL	NULL

4. Where Clause (AND/OR):

- Write a query to select all SaleItems where the Quantity is greater than 5 and the Price is less than $20.

```
72      Select * from SaleItems
73      Where Quantity > 5 AND Price < 20
74      Limit 1000;
75
```

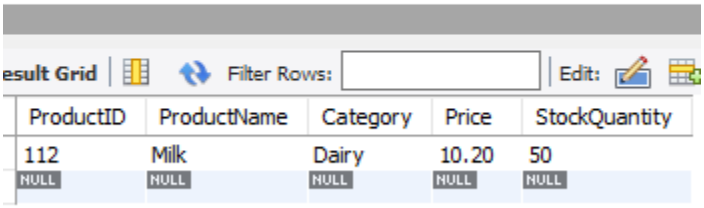
esult Grid Filter Rows: Edit:

SaleitemID	SaleID	ProductID	Quantity	Price
401	221	111	50	11.00
402	222	112	60	15.00
405	225	115	45	12.00
410	230	120	35	15.00
NULL	NULL	NULL	NULL	NULL

5. LIKE Operator:

- Write a query to select all Products where the ProductName contains 'Milk'.

```
79  Select * from Products
80  where ProductName like 'Milk';
81
```



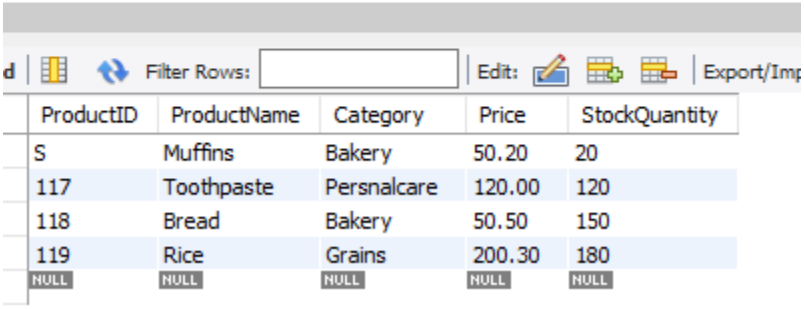
esult Grid | Filter Rows: | Edit:

ProductID	ProductName	Category	Price	StockQuantity
112	Milk	Dairy	10.20	50
NULL	NULL	NULL	NULL	NULL

6. CASE Statement:

- Write a query to select ProductName, Price, and a new column DiscountedPrice from the Products table. If a product has a discount applicable today, calculate the DiscountedPrice using the discount percentage from the Discounts table.

```
select p.ProductName, p.Price,
Case When d.Discountpercentage IS NOT NULL
AND CURDATE() BETWEEN d.startDate AND d.EndDate
THEN p.Price *(1-d.DiscountPercentage/100)
ELSE p.Price END AS DiscountPrice
FROM Products p
LEFT JOIN
Discount d ON p.ProductID = d. ProductID;
```



d | Filter Rows: | Edit: | Export/Imp

ProductID	ProductName	Category	Price	StockQuantity
S	Muffins	Bakery	50.20	20
117	Toothpaste	Persnalcare	120.00	120
118	Bread	Bakery	50.50	150
119	Rice	Grains	200.30	180
NULL	NULL	NULL	NULL	NULL

7. Subquery:

- Write a query to find all Customers who have made purchases totaling more than $500. Use a subquery to find these CustomerIDs.

```
Select * from Customers
where customerID IN(
Select  customerID from Sales
Group By customerID Having SUM(TotalAmount) > 500);
```

sult Grid | Filter Rows: | Edit: | Export/Imp

customerID	FirstName	LastName	Email	Phone	RegistrationDate
NULL	NULL	NULL	NULL	NULL	NULL

8. Group By:

- Write a query to get the total sales amount for each product category. Join Products with SaleItems and Sales, and group by Category.

```
Select p.Category,
SUM(si.Quantity * p.Price) AS TotalSalesAmount
FROM
Products p
JOIN
Saleitems si ON p.ProductID = si.ProductID
JOIN
Sales s ON si.SaleID = s.SaleID
GROUP BY
p.Category;
```

Result Grid | Filter Rows: | Export: | Wrap Cell

Category	TotalSalesAmount
Fruit	2027.50
Dairy	1170.00
Bakery	7992.30
Meat	2124.00
Vegetables	837.00
Persnalcare	2400.00
Grains	5007.50

9. Having Clause:

- Write a query to get the total quantity sold for each product, but only include products that have been sold more than 50 times. Use the HAVING clause.

```
219  SELECT
220     p.productName,
221   SUM(si.Quantity) AS 'Total Quantity Sold'
222  FROM
223       Products p
224  JOIN
225  Saleitems si ON p.ProductID = si.ProductID
226  GROUP BY  p.ProductNAme
227  HAVING SUM(si.Quantity) > 50
228  Limit 0, 1000;
229
```

Result Grid | Filter Rows: | Export: | Wrap Cell C

productName	Total Quantity Sold
Milk	60
Muffins	124
Chicken	60
Cabbage	54

-

10. Limit:

```
233  select productName, Price From Products
234  order By Price Desc
235  limit 5;
236
```

esult Grid | Filter Rows: | Export: | Wrap C

productName	Price
Rice	200.30
Toothpaste	120.00
Bread	50.50
Muffins	50.20
Apple	40.50

11. Inner Join:

- Write a query to join Sales with Customers to get a list of all sales with CustomerName and TotalAmount.

```
241  SELECT
242      c.FirstName,
243      s.TotalAmount
244  FROM
245      Sales s
246  INNER JOIN
247      Customers c ON s.customerID = c.customerID;
```

FirstName	TotalAmount
Varma	45.00
Raj	55.00
Mike	100.00
Scott	150.00
Marry	240.00
Shiny	300.00
Mikky	350.00
Rose	250.00
Mike	360.00
Yashik	120.00

12. Outer Join:

- Write a query to get a list of all Products and any associated Discounts. Include products that might not have any discounts.

```
252  Select
253  p.ProductID,
254  p.ProductName,
255  p.Price,
256  d.DiscountID,
257  d.DiscountPercentage,
258  d.StartDate,
259  d.EndDate
260  From  Products p
261  LEFT JOIN
262   Discounts d ON p.ProductID = d.ProductID;
263
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ProductID	ProductName	Price	DiscountID	DiscountPercentage	StartDate	EndDate
111	Banana	12.20	2001	55.45	2024-01-01	2024-01-31
112	Milk	10.20	2002	23.32	2024-02-01	2024-02-28
113	Muffins	50.20	2003	45.43	2024-03-01	2024-03-30
114	Chicken	35.40	2004	44.34	2024-04-01	2024-04-30
115	Egg	12.40	2005	55.64	2024-05-01	2024-05-31
116	Cabbage	15.50	2006	34.55	2024-06-01	2024-06-30
117	Toothpaste	120.00	2007	34.54	2024-07-01	2024-07-31
118	Bread	50.50	2008	55.43	2024-08-01	2024-08-31
119	Rice	200.30	2009	23.43	2024-09-01	2024-09-30
120	Apple	40.50	2010	44.43	2024-10-01	2024-10-31

13. Join with Aggregation:

Write a query to get the average sale amount per customer. Use an INNER JOIN between Sales and Customers, and group by CustomerI

```sql
SELECT
    c.CustomerID,
    AVG(s.TotalAmount) AS AverageSaleAmount
FROM
    Sales s
INNER JOIN
    Customers c ON s.CustomerID = c.CustomerID
GROUP BY
    c.CustomerID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

CustomerID	FirstName	AverageSalesAmount
1	Varma	45.000000
2	Raj	55.000000
3	Mike	100.000000
4	Scott	150.000000
5	Marry	240.000000
6	Shiny	300.000000
7	Mikky	350.000000
8	Rose	250.000000
9	Mike	360.000000
10	Yashik	120.000000

14. Subquery with Join:

- Write a query to find all Products that have been sold at a discounted price. Use a subquery to filter products with active discounts.

```sql
Select p.ProductID, p.productName, p.Price From Products p
Where p.ProductID IN(
Select d.ProductID From Discounts d
where CURDATE() BETWEEN d.StartDate AND d.EndDate
);
```

esult Grid | Filter Rows: | Edit: | Export/Import:

ProductID	productName	Price
118	Bread	50.50
NULL	NULL	NULL

15. Advanced Join:

- Write a query to list FirstName, LastName, ProductName, Quantity, and Price for all sales. Use INNER JOIN to link Sales, SaleItems, Products, and Customers.

```
Select c.FirstName, c.LastName, p.ProductName, si.Quantity, p.Price From Sales s
INNER JOIN
       Saleitems si ON s.SaleID = si.SaleID
INNER JOIN
      Products p ON si.ProductID = p. productID
INNER JOIN
   Customers c on s.customerID  = c.customerID;
```

sult Grid | Filter Rows: | Export: | Wrap Cell Content:

FirstName	LastName	ProductName	Quantity	Price
Varma	Raj	Banana	50	12.20
Raj	Reddy	Milk	60	10.20
Mike	Joe	Muffins	124	50.20
Scott	James	Chicken	60	35.40
Marry	Comm	Egg	45	12.40
Shiny	Rani	Cabbage	54	15.50
Mikky	Roy	Toothpaste	20	120.00
Rose	Mary	Bread	35	50.50
Mike	Jussy	Rice	25	200.30
Yashik	Rao	Apple	35	40.50