AIM:	To use Aggregate Functions on a database

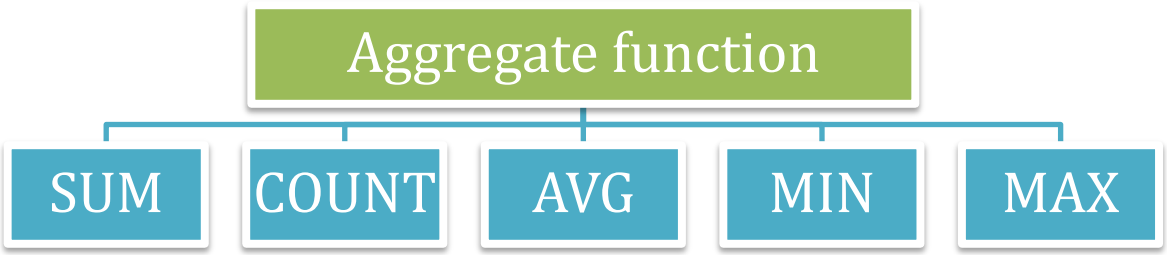
Theory:

Aggregate Functions

An aggregate function in SQL performs a calculation on multiple values and returns a single value. SQL provides many aggregate functions, including avg, count, sum, min, max, etc. An aggregate function ignores NULL values, except for the count function, when it performs the calculation.

An aggregate function in SQL returns one value after calculating multiple values of a column. We often use aggregate functions with the GROUP BY and HAVING clauses of the SELECT statement.

Types of Aggregate Functions



- COUNT FUNCTION: The count function is used to Count the number of rows in a database table. It can work on both numeric and non-numeric data types.
    - **Count Syntax**

```
SELECT COUNT(COLUMN_NAME)
FROM Table_name;
```

- SUM FUNCTION: The sumV function is used to calculate the sum of all selected columns. It works on numeric fields only.
    - **Count Syntax**

```
SELECT SUM(COLUMN_NAME)
FROM Table_name
WHERE condition;
```

- AVG FUNCTION: The AVG function is used to calculate the average value of the numeric type. AVG function returns the average of all non-null values.
    - **Count Syntax**

```
WO AVG(COLUMN_NAME)
FROM Table_name;
```

- MIN FUNCTION: MIN function is used to find the minimum value of a certain column. This function determines the smallest value of all selected values of a column.
    - **Count Syntax**

```
SELECT MIN(COLUMN_NAME)
FROM Table_name;
```

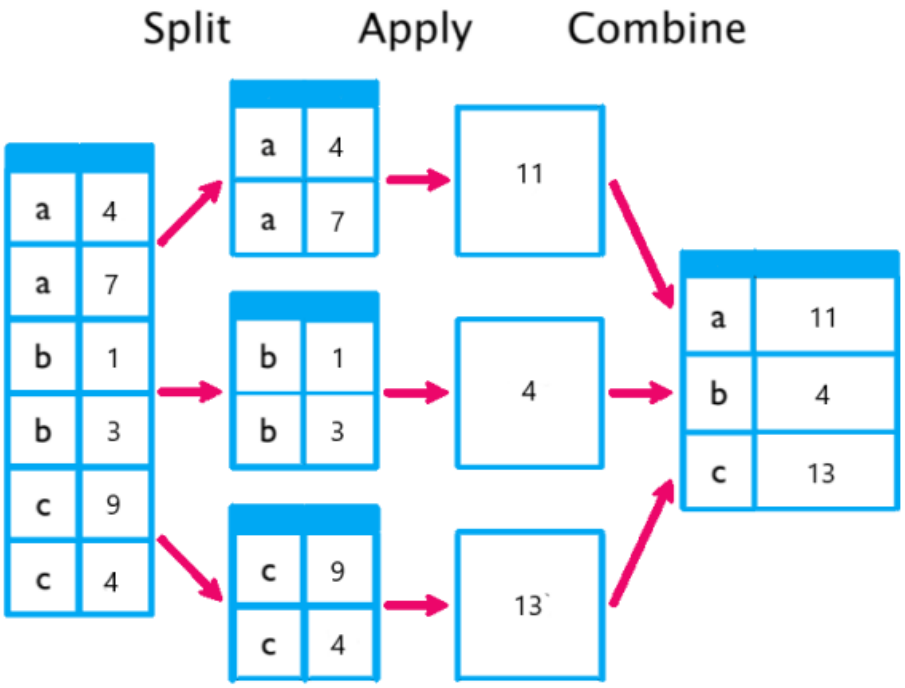
- MAX FUNCTION: MAX function is used to find the maximum value of a certain column. This function determines the largest value of all selected values of a column.
    - **Count Syntax**

```
SELECT MAX(COLUMN_NAME)
FROM Table_name;
```

Group by Function

The Group By statement is used to group together any rows of a column with the same value stored in them, based on a function specified in the statement. Generally, these functions are one of the aggregate functions such as MAX() and SUM().
The SQL Group By statement uses the split-apply-combine strategy.

- **Split**: The different groups are split with their values.
- **Apply**: The aggregate function is applied to the values of these groups.
- **Combine**: The values are combined in a single row.



- **Syntax for Using Group By**

```
SELECT column_1, function_name(column_2)
FROM table_name
WHERE [condition]
GROUP BY column_name
ORDER BY column_name;
```

Having Clause

The HAVING clause enables users to filter the results based on the groups specified. The SQL HAVING clause is used with the GROUP BY clause. The HAVING clause can be used in tandem with aggregate functions, whereas a WHERE clause cannot be.

- **Syntax for Using Group By**

```
SELECT column_1, function_name(column_2)
```

```
FROM table_name
WHERE [condition]
GROUP BY column_name
Having [condition]
ORDER BY column_name;
```

Queries

1) Orders having max amount from each restaurant

Syntax:

```
Select OrderID, CustomerID, RestName, max(Amount) as max_amount
from Orders
group by RestName;
```

Result:

OrderID	CustomerID	RestName	max_amount
691	17456	Taj hotel	2342
692	11249	Bhagat Tarachand	1204
693	12451	Kshirsagar	694
694	13934	Pizza Hut	782
695	21458	McDonalds	512
696	19403	Belgian Waffles	178

2) Orders from restaurants having avg amount greater than 500

Syntax:

```
Select OrderID, CustomerID, RestName, avg(Amount) as avg_amount
from Orders
group by RestName
Having avg(Amount)>=500;
```

Result:

OrderID	CustomerID	RestName	avg_amount
691	17456	Taj hotel	2342.0000
692	11249	Bhagat Tarachand	1013.5000
693	12451	Kshirsagar	694.0000
694	13934	Pizza Hut	782.0000

3) Max amount of orders grouped by payment method

Syntax:

```
select  max(Amount) as max_amount, PaymentMethod
from Orders
group by PaymentMethod
having max(Amount);
```

Result:

max_amount	PaymentMethod
2342	UPI
823	Cash on Delivery
1204	Net Banking

4) Number of orders from each restaurant

Syntax:

```
Select RestName, count(OrderID) as No_of_orders
from Orders
group by RestName;
```

Result:

RestName	No_of_orders
Taj hotel	1
Bhagat Tarachand	2
Kshirsagar	1
Pizza Hut	2
McDonalds	2
Belgian Waffles	1

5) To find restaurant with maximum rating

Syntax:

```
Select RestName, Rating, Location
from Restaurants
Where Rating = (Select max(Rating) from Restaurants);
```

Result:

RestName	Rating	Location
Belgian Waffles	4.6	Vashi

6) To find average rating of all restaurants in Mumbai

Syntax:

```
Select location, avg(Rating) as Average_Rating
from Restaurants
Where Location = "Mumbai";
```

Result:

location	Average_Rating
Mumbai	3.5

7) To find restaurants in each variety having avg rating more than 3

Syntax:

```
Select RestName, max(Rating) as Max_Rating
from Restaurants
group by Varieties
Having max(Rating)>3;
```

Result:

RestName	Max_Rating
Taj hotel	4
Kshirsagar	3.9
Pizza Hut	4.5
McDonalds	4.2
Belgian Waffles	4.6

8) To count the number of restaurants in each city

Syntax:

```
Select location, count(ResturID) as No_of_Restaurants
```

```
from Restaurants
group by location;
```
Result:

location	No_of_Restaurants
Mumbai	1
Vashi	2
Nerul	1
Sanpada	1
Ghatkopar	1

9) To see Customers and their maximum amount orders
Syntax:
```
Select c.CustomerName, o.Amount, o.RestName
From Customers as c
Inner Join Orders as o
On c.CustomerID = o.CustomerID
group by c.CustomerID
having max(o.Amount);
```
Result:

CustomerName	Amount	RestName
Vansh	823	Bhagat Tarachand
Yash	694	Kshirsagar
Harsh	782	Pizza Hut
Sahil	2342	Taj hotel
Pranay	178	Belgian Waffles
Rishabh	287	McDonalds

10) To see info of customers with avg amount paid more than 300
Syntax:
```
Select c.*, avg(o.Amount) as Average_Amt, o.RestName
From Customers as c
Inner Join Orders as o
On c.CustomerID = o.CustomerID
group by c.CustomerID
having avg(o.Amount)>300;
```
Result:

CustomerID	CustomerName	PhoneNo	Address	Email	Average_Amt	RestName
11249	Vansh	878398	Airoli	vansh@gmail.com	823.0000	Bhagat Tarachand
12451	Yash	873458	Andheri	yash@gmail.com	949.0000	Kshirsagar
13934	Harsh	879838	Goregaon	magician@gmail.com	782.0000	Pizza Hut
17456	Sahil	879738	Vashi	sahil.ved@gmail.com	1562.0000	Taj hotel
19403	Pranay	879838	Borivali	pranay@gmail.com	345.0000	Belgian Waffles

11) To see number of orders madeby each customer
Syntax:
```
Select c.CustomerName, count(o.OrderID) as No_of_Orders
From Customers as c
Inner Join Orders as o
On c.CustomerID = o.CustomerID
group by c.CustomerID;
```

Result:

CustomerName	No_of_Orders
Vansh	1
Yash	2
Harsh	1
Sahil	2
Pranay	2
Rishabh	1

12) To get total amount spent by each customer

Syntax:

```
Select c.CustomerName, sum(o.Amount) as Amount_spent
From Customers as c
Inner Join Orders as o
On c.CustomerID = o.CustomerID
group by c.CustomerID;
```

Result:

CustomerName	Amount_spent
Vansh	823
Yash	1898
Harsh	782
Sahil	3124
Pranay	690
Rishabh	287

13) To get details of highest rated delivery person

Syntax:

```
Select * from DeliveryPerson
where Rating = (select max(Rating) from DeliveryPerson);
```

Result:

EmployeeID	DelName	PhoneNo	Rating	ShiftTime
37	Anil	8722983	5	4pm-9pm
NULL	NULL	NULL	NULL	NULL

14) To get highest rated delivery person from every shift

Syntax:

```
Select DelName, max(Rating) as Rating
from DeliveryPerson
group by ShiftTime;
```

Result:

DelName	Rating
Rahul	5
Avishkar	0
Sunil	4
Shivam	3

15) To get number of orders delivered by each Delivery person

Syntax:

```
Select d.DelName, count(o.OrderID) as No_of_orders
```

```
From DeliveryPerson as d
LEFT JOIN Orders as o
ON d.EmployeeID = o.EmployeeID
group by o.EmployeeID;
```

Result:

DelName	No_of_orders
Rahul	1
Avishkar	2
Sunil	3
Anil	2
Shivam	0
Jay	1

16) To get name of employees whose average amount exceeded 500

Syntax:

```
Select d.DelName, avg(o.Amount) as Avg_amount
From DeliveryPerson as d
LEFT JOIN Orders as o
ON d.EmployeeID = o.EmployeeID
group by o.EmployeeID
Having avg(o.Amount)>500;
```

Result:

DelName	Avg_amount
Rahul	823.0000
Avishkar	1314.5000
Sunil	551.3333
Anil	993.0000
Jay	512.0000

Conclusion

From this experiment, I learned about the aggregate functions using group clause in MySQL. I understood how to use these functions to perform basic arithmetic functions such as sum and average on any column. I also learned different type of clauses such as the having and group by clause.