AGGREGATE FUNCTIONS IN SQL

1. Aggregate / Group Functions

- Group Functions, as the name suggests, are functions that operate on groups (sets) of values and returns one result per group
- Group function returns a single result row for every group of queried rows
- Based on the query statement it may return single or multiple rows

Commonly used aggregate functions are:

- **SUM**
- **AVG**
- **MAX**
- **MIN**
- **COUNT**

Consider the following table: Employee

EMP_ID	EMP_NAME	SALARY	DEP_NAME	BRANCH_NAME
1	amit	10000	HR	Kolkata
2	ajay	20000	Marketing	Mumbai
3	sima	16000	HR	Ahmedabad
4	dipa	40000	Admin	Kolkata
5	anuj		Marketing	Ahmedabad

- **AVG**: Returns an average value, ignoring null values.

Syntax: `AVG([DISTINCT] column_name)`

Example: `SELECT AVG(salary) as "Average Salary" FROM Employee;`

Output: `21500`

The above query displays the average salary of all the employees in the table Employee

- **MAX**: Returns the maximum value, ignoring null values.

Syntax: `MAX([DISTINCT] column_name)`

Example: `SELECT MAX(salary) as "Maximum Salary" FROM Employee where Dep_Name='HR';`

Output:16000

The above query displays the maximum salary of all the employees in HR Department in the table Employee.

- **MIN**: Returns the minimum value, ignoring null values.

Syntax: `MIN([DISTINCT] column_name)`

Example: `SELECT MIN(salary) as "Minimum Salary" FROM Employee where Dep_Name='HR';`

Output: 10000

The above query displays the minimum salary of all the employees in HR Department in the table Employee.

- **COUNT**: Returns the count of not null values ignoring null values.

Syntax: `COUNT([DISTINCT] column_name)`

Example: `SELECT COUNT(DISTINCT Dep_name) Departments FROM Employee;`

Output: 3

The above query displays the count of different departments in the table Employee.

- **COUNT**(*) : Count function with asterisk returns the count of total number of rows including null values

Syntax: `COUNT(*)`

Example: `SELECT COUNT(*) FROM Employee;`

Output: 5

The above query displays the total number of rows in table Employee.

2. GROUP BY clause

Creates a data set, containing several sets of records grouped together based on a condition.

Syntax: SELECT <columnName1>[,<columnName2>], AGGREGATE FUNCTION(<expression>) FROM Table_Name GROUP BY <columnName1>[,<columnName2>] ;

Example: `SELECT dep_name,COUNT(emp_id) "No of Employee" FROM Employee GROUP BY dep_name;`

Output:

```
HR                    2
Marketing              2
Admin                 1
```

The above query displays the number of employee in each department.

3. WHERE clause

Used to apply a filter condition before grouping the rows.

Syntax: SELECT <columnName1>[,<columnName2>], AGGREGATE FUNCTION(<expression>) FROM Table_Name WHERE <condition_before_grouping_rows> GROUP BY <columnName1>[,<columnName2>] ;

Example: `SELECT Dep_Name,COUNT(Salary) FROM Employee WHERE Salary>15000 GROUP BY Dep_Name;`

Output:

```
HR                          1
Marketing                   1
Admin                       1
```

The above query displays department wise count of salary with more than 15000.

4. HAVING clause

Used to apply a filter condition on aggregated values.

Syntax: SELECT <columnName1>[,<columnName2>], AGGREGATE FUNCTION(<expression>) FROM Table_Name WHERE <condition_before_grouping_rows> GROUP BY <columnName1>[,<columnName2>] HAVING <condition_on_grouped_result>;

Example: `SELECT Dep_Name, SUM(Salary) FROM Employee WHERE Salary>12000GROUP BY Dep_Name HAVING SUM(Salary)<30000;`

Output:

```
HR                    16000
Marketing             20000
```

The above query displays the departments for which the total salary is less than 30000 (Admin department is filtered from the result set total salary for Admin department is 40000)