**Primary and Foreign key Constraints**

**A primary key is a field in a table which uniquely identifies each row/record in a database table. Primary keys must contain unique values. A primary key column cannot have NULL values.**

**A table can have only one primary key, which may consist of single or multiple fields. When multiple fields are used as a primary key, they are called a composite key.**

**If a table has a primary key defined on any field(s), then you cannot have two records having the same value of that field(s).**

**CREATE TABLE CUSTOMERS( ID INT NOT NULL,**

**NAME VARCHAR (20) NOT NULL,**

**AGE INT NOT NULL,**

**ADDRESS CHAR (25) ,**

**SALARY DECIMAL (18, 2),**

**PRIMARY KEY (ID)**

**);**

**For defining a PRIMARY KEY constraint on multiple columns, use the SQL syntax given below.**

**CREATE TABLE CUSTOMERS(**

**ID INT NOT NULL,**

**NAME VARCHAR (20) NOT NULL,**

**AGE INT NOT NULL,**

**ADDRESS CHAR (25) ,**

**SALARY DECIMAL (18, 2),**

**PRIMARY KEY (ID, NAME)**

**);**

**Foreign key**

**A foreign key is a key used to link two tables together. This is sometimes also called as a referencing key.**

**A Foreign Key is a column or a combination of columns whose values match a Primary Key in a different table.**

**The relationship between 2 tables matches the Primary Key in one of the tables with a Foreign Key in the second table.**

**Example**

**Consider the structure of the following two tables.**

**CUSTOMERS table**

**CREATE TABLE CUSTOMERS(**

**ID INT NOT NULL,**

**NAME VARCHAR (20) NOT NULL,**

**AGE INT NOT NULL,**

**ADDRESS CHAR (25) ,**

**SALARY DECIMAL (18, 2),**

**PRIMARY KEY (ID)**

**);**

**ORDERS table**

**CREATE TABLE ORDERS (**

**ID INT NOT NULL,**

**DATE DATE,**

**CUSTOMER\_ID INT references CUSTOMERS(ID),**

**AMOUNT double,**

**PRIMARY KEY (ID)**

**);**

**Using Group By and Having Clause**

**An important component for analyst to summarize the data such as sales, profit, cost and salary.**

**In SQL, GROUP BY Clause is one of the tools to summarize or aggregate the data series. For example, sum up the daily sales and combine in a single quarter and show it to the senior management.**

**Similarly, if we want to count how many employees in each department of the company.**

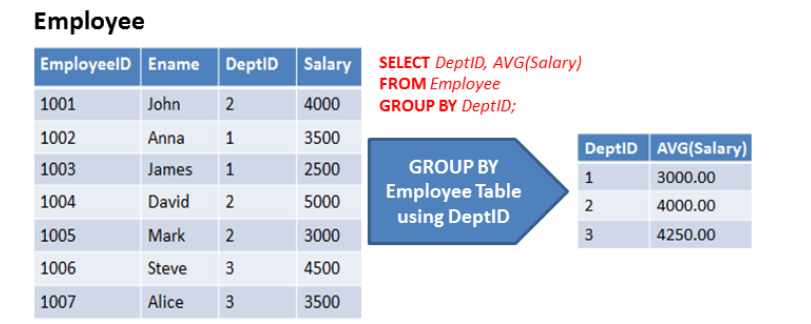
**The GROUP BY clause is a SQL command that is used to group rows that have the same values.**

**The GROUP BY clause is used in the SELECT statement .Optionally it is used in conjunction with aggregate functions to produce summary reports from the database.**

**After Grouping the data, we can filter the grouped record using HAVING Clause. HAVING Clause returns the grouped records which match the given condition.**

**GROUP BY Syntax**

**SELECT statements... GROUP BY column\_name1[,column\_name2,...] [HAVING condition];**

****



**Having Clause**

**HAVING Clause utilized in SQL as a conditional Clause with GROUP BY Clause. This conditional clause returns rows where aggregate function results matched with given conditions only.**

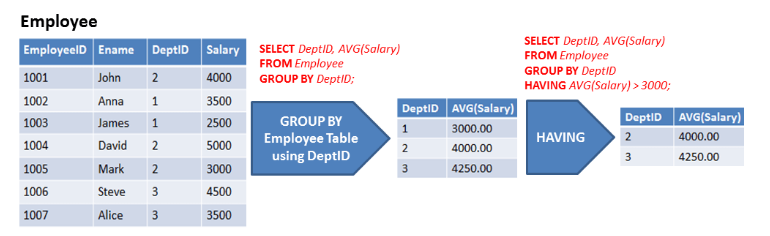
**It added in the SQL because WHERE Clause cannot be combined with aggregate results.**

**The primary purpose of the WHERE Clause is to deal with individual records.**

**HAVING Clause always utilized in combination with GROUP BY Clause.**

**HAVING Clause restricts the data on the group records rather than individual records.**

**WHERE and HAVING can be used in a single query.**

****



**In above example, Table is grouped based on DeptID column and these grouped rows filtered using HAVING Clause with condition AVG(Salary) > 3000.**

**SQL Aggregate Functions**

* **SQL aggregation function is used to perform the calculations on multiple rows of a single column of a table. It returns a single value.**
* **It is also used to summarize the data.**

**1. COUNT FUNCTION**

**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 function uses the COUNT(\*) that returns the count of all the rows in a specified table. COUNT(\*) considers duplicate and Null.**

**Syntax**

**COUNT(\*) or COUNT( [ALL|DISTINCT] expression )**

**Sample table:**

**PRODUCT\_MAST**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PRODUCT** | **COMPANY** | **QTY** | **RATE** | **COST** |
| **Item1** | **Com1** | **2** | **10** | **20** |
| **Item2** | **Com2** | **3** | **25** | **75** |
| **Item3** | **Com1** | **2** | **30** | **60** |
| **Item4** | **Com3** | **5** | **10** | **50** |
| **Item5** | **Com2** | **2** | **20** | **40** |
| **Item6** | **Cpm1** | **3** | **25** | **75** |
| **Item7** | **Com1** | **5** | **30** | **150** |
| **Item8** | **Com1** | **3** | **10** | **30** |
| **Item9** | **Com2** | **2** | **25** | **50** |
| **Item10** | **Com3** | **4** | **30** | **120** |

**Example: COUNT()**

**SELECT COUNT(\*)**

**FROM PRODUCT\_MAST;**

**Output:** 10

**Example: COUNT with WHERE**

**SELECT COUNT(\*)**

**FROM PRODUCT\_MAST;**

**WHERE RATE>=20;**

**Output:7**

**Example: COUNT() with DISTINCT**

**SELECT COUNT(DISTINCT COMPANY)**

**FROM PRODUCT\_MAST;**

**Output:3**

**Example: COUNT() with GROUP BY**

**SELECT COMPANY, COUNT(\*)**

**FROM PRODUCT\_MAST**

**GROUP BY COMPANY;**

**Output:**

**Com1 5**

**Com2 3**

**Com3 2**

**Example: COUNT() with HAVING**

**SELECT COMPANY, COUNT(\*)**

**FROM PRODUCT\_MAST**

**GROUP BY COMPANY**

**HAVING COUNT(\*)>2;**

**Output:**

**Com1 5**

**Com2 3**

**2. SUM Function**

**Sum function is used to calculate the sum of all selected columns. It works on numeric fields only.**

**Syntax**

**SUM() or SUM( [ALL|DISTINCT] expression )**

**Example: SUM()**

**SELECT SUM(COST)**

**FROM PRODUCT\_MAST;**

**Output:**

**670**

**Example: SUM() with WHERE**

**SELECT SUM(COST)**

**FROM PRODUCT\_MAST**

**WHERE QTY>3;**

**Output:**

**320**

**Example: SUM() with GROUP BY**

**SELECT SUM(COST)**

**FROM PRODUCT\_MAST**

**WHERE QTY>3**

**GROUP BY COMPANY;**

**Output:**

**Com1 150**

**Com2 170**

**Example: SUM() with HAVING**

**SELECT COMPANY, SUM(COST)**

**FROM PRODUCT\_MAST**

**GROUP BY COMPANY**

**HAVING SUM(COST)>=170;**

**Output:**

**Com1 335**

**Com3 170**

**3. 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.**

**Syntax**

**AVG() or AVG( [ALL|DISTINCT] expression )**

**Example:**

**SELECT AVG(COST)**

**FROM PRODUCT\_MAST;**

**Output:**

**67.00**

**4. 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.**

**Syntax**

**MAX() or MAX( [ALL|DISTINCT] expression )**

**Example:**

**SELECT MAX(RATE)**

**FROM PRODUCT\_MAST;**

**Output: 30**

**5. 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.**

**Syntax**

**MIN() or MIN( [ALL|DISTINCT] expression )**

**Example:**

**SELECT MIN(RATE) FROM PRODUCT\_MAST;**

**Output:**

**10**