MYSQL

# Grouping

* Used to group rows that have the same values.
* It summarizes data from the database.
* The GROUP BY clause returns one row for each group.
* Can be used with -
  + Sum
  + Count
  + Min
  + Max
  + Avg

Grouping with multiple columns

* Syntax -
  + *SELECT \* FROM table GROUP BY col1, col2*
* This will display the total records in each group and sub group.

# Having

* WHERE keyword can not be used with group functions.
* We have to use HAVING clause in SELECT statement to specify filter conditions for grouped results.
* If the GROUP BY clause is omitted, the HAVING clause behaves like the WHERE clause.
* used to sort the result-set in ascending or descending order.
* Sorts the records in ascending order by default.
* To sort the records in descending order, use DESC keyword.

# In and not In

* The IN operator is a shorthand for multiple OR conditions.
* The IN operator allows you to determine if a specified value matches any value from a list or from a subquery.
* Syntax -

*SELECT column1,column2,...FROM table\_name WHERE (expr|column\_1) IN ('value1','value2',...);*

* The values in the list must be separated by a comma (,).
* You can combine the IN operator with the NOT operator to determine if a value does not match any value in a list or a subquery.

# Between

* We can use BETWEEN clause to replace a combination of "greater than equal AND less than equal" conditions.
* Syntax -
  + SELECT *column\_name(s)* FROM *table\_name* WHERE *column\_name* BETWEEN

*value1* AND *value2;*

* It will return the records where *expression* is within the range of *value1* and

*value2* (inclusive).

* Values can be numbers, text or dates.

## Can be used with different commands

* + Select

## Update

* + Delete

## IN

* It will return all rows where value does not lie in the given range.

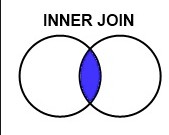
# Like

* The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.
* There are two wildcards used in conjunction with the LIKE operator:
  + % : The percent sign represents zero, one, or multiple characters
  + \_ : The underscore represents a single character
* Syntax -
  + *SELECT column1, column2, ...FROM table\_name WHERE column LIKE pattern;*

|  |  |
| --- | --- |
| Statements | Description |
| LIKE 'S%' | It finds any value which starts with 'S'. |
| LIKE '%S%' | It finds any value which have 'S' in any position. |
| LIKE '\_SS%' | It finds any value which have 'SS' in the second and third positions. |
| LIKE 'S\_%\_%' | It finds any value which starts with 'S' and have at least three characters in length. |
| LIKE '%S' | It finds any value which ends with 'S'. |
| LIKE '\_S%P' | It finds any value which have 'S' in the second position and ends with 'P'. |
| LIKE  'S P' | It finds any value in a five digit numbers which start with 'S' and ends with 'P'. |

* Let's say you wanted to search for a % or a \_ character in the MySQL LIKE condition. You can do this using an Escape character.
  + *SELECT \* FROM table\_name WHERE column\_name LIKE 'G\%';*
* We can override the default escape character in MySQL by providing the ESCAPE modifier as follows:
  + *SELECT \* FROM table\_name WHERE column\_name LIKE 'G!%' ESCAPE '!';*

# Joins

* With join, we can query data from two (or multiple) tables based on a related column which is present in both the tables.
* While performing a join, we need to specify the shared column and the condition on which we want to join tables
* You can use JOINS in the SELECT, UPDATE and DELETE statements to join multiple tables.
* Inner Join or Simple Join
* Left Outer Join or Left Join
* Right outer join or RIght Join
* Full outer join or Full Join
* ClassDetails -
  + It stores which class is assigned to which teacher like class id and teacher id
* TeacherDetails -
  + It stores the details of each teacher like ID of teacher, teacher’s name and the subject id which is taught by the teacher
* SubjectDetails -
  + This table stores the details of each subject like subject id, name of subject and total number of students which are admitted in individual subject.
* Inner Join or Simple Join
* Left Outer Join or Left Join
* Right outer join or RIght Join
* Full outer join or Full Join
* This will only return rows when there is at least one row in both tables that match the specified

join condition.

* Syntax -
  + *SELECT table1.col1, table2.col2, …. FROM table\_name1*

*INNER JOIN table\_name2 ON*

*table1.column\_name = table2.column\_name;*

* Syntax -

*( SELECT table\_1.col1, table\_2.col2, …. FROM table\_1*

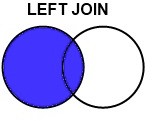
*INNER JOIN table\_2 ON*

*table\_1.column\_name = table\_2.column\_name ) INNER JOIN table\_3*

*ON*

*table\_2.column\_name = table\_3.column\_name*

# Left Outer Join

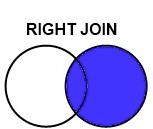
* It returns all rows from the left table specified and only those rows from the

other table where the join condition is matched.

* Syntax -
  + *SELECT table\_1.col1, table\_2.col2, …. FROM table\_1*

*LEFT JOIN table\_2 ON*

*table\_1.column\_name = table\_2.column\_name;*

* It returns all rows from the right table specified and only those rows from the left table where the join condition is matched.
* Syntax -
  + *SELECT table\_1.col1, table\_2.col2, …. FROM table\_1*

*RIGHT JOIN table\_2 ON*

*table\_1.column\_name = table\_2.column\_name;*