3. Removing specific data from a table

The ***DELETE*** statement is used to delete rows from a table.

Syntax for DELETE Statement:

DELETE FROM <table_name> WHERE <condition>;

where <table_name> is the table whose records are to be deleted.

The *WHERE* clause in the SQL DELETE command is optional and it identifies the rows in the column that get deleted. If you do not include the WHERE clause, all the rows in the table are deleted.

For example,

```
mysql>DELETE FROM Student WHERE Rollno = 10;
```

The above statement shall delete the record only for roll number 10.

Deleting all rows from a table:

To delete all the rows from the Student table, the DELETE statement will be:

```
mysql>DELETE FROM Student;
```

Note: Delete command will not affect the table structure, attributes and constraints.

10.9 DATA QUERY LANGUAGE (SQL QUERY PROCESSING)

After creating the database, the table is created and the data is stored into it. Now, it is time to perform query processing on the already created tables to retrieve and view data on the screen. Retrieving information from the tables is done mainly using the SELECT command. The SQL SELECT statement is used to fetch data from one or more database tables. It is used to select rows and columns from a database/relation.

10.9.1 SQL SELECT Statement

This command can perform projection as well as selection. It is the most extensively used SQL command. The SELECT statement can be used to retrieve a subset of rows or columns from one or more tables present in a database.

1. Projection: Selecting Specific Columns

This is the capability of SQL to return only specific attributes and all rows from the relation.

For example,

```
SELECT Name, Gender FROM Student;
```

The above command will display only Name and Gender attributes from the relation Student.

Resultant table: Student

Name	Gender
Raj Kumar	M
Deep Singh	M
Ankit Sharma	M
Radhika Gupta	F
Payal Goel	F
Diksha Sharma	F
Gurpreet Kaur	F
Akshay Dureja	M
Shreya Anand	F
Prateek Mittal	M

10 rows in a set (0.01 sec)

Selection: Selecting Specific Rows—WHERE Clause

This capability of SQL returns the tuples from a relation with all the attributes that satisfies a particular condition. Use of WHERE clause is required when specific tuples are to be fetched or manipulated. To select all the columns from a table, the asterisk (*) can be used.

Syntax: SELECT *FROM <table-name>;

OR SELECT * FROM <which_table>
WHERE <conditions_to_satisfy>;

For example,

```
SELECT * FROM Student WHERE Gender = 'M';
```

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no
1.	Raj Kumar	M	93	17-11-2000	NULL
2.	Deep Singh	M	98	22-08-1996	NULL
3.	Ankit Sharma	M	76	02-02-2000	NULL
8.	Akshay Dureja	M	90	05-05-1997	NULL
10.	Prateek Mittal	M	75	25-12-2000	NULL

5 rows in a set (0.01 sec)

The above command displays all attributes but specific rows which satisfy the condition where Gender is Male from the Student table.

To select all the columns and rows from the table, the command used is:

```
mysql>SELECT * FROM Student;
```

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no
1.	Raj Kumar	M	93	17-11-2000	NULL
2.	Deep Singh	M	98	22-08-1996	NULL
3.	Ankit Sharma	M	76	02-02-2000	NULL
4.	Radhika Gupta	F	78	03-12-1999	NULL
5.	Payal Goel	F	82	21-04-1998	NULL
6.	Diksha Sharma	F	80	17-12-1999	NULL
7.	Gurpreet Kaur	F	65	04-01-2000	NULL
8.	Akshay Dureja	M	90	05-05-1997	NULL
9.	Shreya Anand	F	70	08-10-1999	NULL
10.	Prateek Mittal	M	75	25-12-2000	NULL

10 rows in a set (0.02 sec)

The above command displays all the rows of all the columns, according to the column-list defined in the table structure. The salient features of SQL SELECT statement are as follows:

- SELECT command displays the columns of the table in the same order in which they are selected from the table.

- In order to retrieve all the columns in the column-list from a table using SELECT command, asterisk (*) is used and the columns are displayed in the same order in which they are stored in the table.
- All the statements (inclusive of SELECT statement) in SQL are terminated with a semicolon (;). Use of semicolon is dependent on the version in use.

CTM: The asterisk (*) means "All". SELECT * means displaying all the columns from a relation.

Using WHERE clause

```
mysql>SELECT * FROM Student WHERE Rollno<=8;
```

The above command shall display only those records whose Rollno is less than or equal to 8.

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no
1.	Raj Kumar	M	93	17-11-2000	NULL
2.	Deep Singh	M	98	22-08-1996	NULL
3.	Ankit Sharma	M	76	02-02-2000	NULL
4.	Radhika Gupta	F	78	03-12-1999	NULL
5.	Payal Goel	F	82	21-04-1998	NULL
6.	Diksha Sharma	F	80	17-12-1999	NULL
7.	Gurpreet Kaur	F	65	04-01-2000	NULL
8.	Akshay Dureja	M	90	05-05-1997	NULL

8 rows in a set (0.02 sec)

When a WHERE clause is used with a SELECT statement, the SQL query processor goes through the entire table one row/record at a time and checks each row to determine whether the condition specified is true with respect to that row or not. If it evaluates to True, the corresponding row is selected, retrieved and displayed, else it returns an empty set (*i.e.*, no data found).

CTM: SQL is case-insensitive, which means keywords like SELECT and select have the same meaning in SQL statements.

3. Re-ordering Columns while Displaying Query Results

While displaying the result for a query, the order of the columns to be displayed can be changed according to the user's requirement. But this is done only for display purpose and no actual (physical) rearrangement of the columns is done.

For example,

```
mysql>SELECT Name, Rollno, DOB, Marks from Student;
```

After executing the above statement, the column shall be displayed in the changed order as Name shall be displayed as the first column, Rollno as the second column, DOB as the third column and Marks as the fourth column.

Resultant table: Student

Name	Rollno	DOB	Marks
Raj Kumar	1.	17-11-2000	93
Deep Singh	2.	22-08-1996	98
Ankit Sharma	3.	02-02-2000	76
Radhika Gupta	4.	03-12-1999	78
Payal Goel	5.	21-04-1998	82
Diksha Sharma	6.	17-12-1999	80
Gurpreet Kaur	7.	04-01-2000	65
Akshay Dureja	8.	05-05-1997	90
Shreya Anand	9.	08-10-1999	70
Prateek Mittal	10.	25-12-2000	75

10 rows in a set (0.02 sec)

CTM: The order in which the columns are displayed using the SELECT command is in accordance with the order in which they are actually stored in the table.

4. Eliminating Duplicate/Redundant Data—DISTINCT clause

DISTINCT clause is used to remove duplicate rows from the results of a SELECT statement. It is used to retrieve only unique values for a column in the table. The DISTINCT keyword can be used only once with a given SELECT statement.

Syntax:

SELECT DISTINCT <column-name> from <*table-name*>;

For example,

Suppose we have added values in Mobile_no column and added a new column Stream to the table Student:

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no	Stream
1.	Raj Kumar	M	93	17-11-2000	9586774748	Science
2.	Deep Singh	M	98	22-08-1996	8988886577	Commerce
3.	Ankit Sharma	M	76	02-02-2000	8567490078	Science
4.	Radhika Gupta	F	78	03-12-1999	9818675444	Humanities
5.	Payal Goel	F	82	21-04-1998	9845639990	Vocational
6.	Diksha Sharma	F	80	17-12-1999	9897666650	Humanities
7.	Gurpreet Kaur	F	65	04-01-2000	7560567890	Science
8.	Akshay Dureja	M	90	05-05-1997	9560567890	Commerce
9.	Shreya Anand	F	70	08-10-1999	8876543988	Vocational
10.	Prateek Mittal	M	75	25-12-2000	9999967543	Science

10 rows in a set (0.02 sec)

With reference to the given table, if we write the SELECT statement as:

```
mysql>SELECT Stream from Student;
```

this statement shall return all the tuples for field Stream from table Student. It will return duplicate values also. Thus, in order to remove these duplicate values, DISTINCT clause is used.

Now, we write a query for displaying the distinct contents on the basis of field stream from Student table:

Resultant table: Student

Stream
Science
Commerce
Science
Humanities
Vocational
Humanities
Science
Commerce
Vocational
Science

Science displayed 4 times

10 rows in a set (0.02 sec)

For example,

```
mysql>SELECT DISTINCT Stream from
Student;
```

Resultant table: Student

Stream
Science
Commerce
Humanities
Vocational

Science displayed once only

4 rows in a set (0.02 sec)

10.9.2 SQL Operators

While working with SELECT statement using WHERE clause, condition-based query processing is carried out using four types of SQL operators:

(a) Arithmetic Operators
(b) Relational Operators
(c) Logical Operators
(d) Special Operators

Table 10.3: SQL Operators and their Functions

OPERATOR/FUNCTION	DESCRIPTION
ARITHMETIC OPERATORS	
+, –, *, /, %	Used in Mathematical calculations.
RELATIONAL (COMPARISON) OPERATORS	
=, >, <, >=, <=, <>	Used in Conditional expressions.
LOGICAL OPERATORS	
AND/ OR/ NOT	Used in Conditional expressions.
SPECIAL OPERATORS	Used in Conditional expressions.
BETWEEN/ NOT BETWEEN	Checks whether an attribute value is within a range or not.
IS NULL/ IS NOT NULL	Checks whether an attribute value is NULL or not NULL.
LIKE/ NOT LIKE	Checks whether an attribute matches a given string pattern or not.
IN/ NOT IN	Checks whether an attribute value matches any value with a given list or not.
DISTINCT	Permits only unique values. Eliminates duplicate ones.

(a) Arithmetic Operators

Arithmetic operators are used to perform simple arithmetic operations like addition (+), subtraction (-), multiplication (*), division (/) and modulus (%). These operators are used with conditional expressions and for performing simple mathematical calculations. The arithmetic operators with SELECT command are used to retrieve rows computed with or without reference to any table.

For example,

```
mysql>SELECT 5 + 10;
```

OR

```
mysql>SELECT 5 + 10 FROM DUAL;
```

The above statement shall evaluate the expression 5 + 10 and returns the value 15 as the result.

```
+---------------+
| 5 + 10        |
+---------------+
| 15            |
+---------------+
```

```
mysql>SELECT SIN(PI()/4), (4+1)*5;
+---------------+-------------+
| SIN(PI()/4) | (4+1)*5   |
+---------------+-------------+
| 0.707107    |     25    |
+---------------+-------------+
```

```
mysql>SELECT 5 * 4 FROM DUAL;
+---------------+
| 5 * 4         |
+---------------+
| 20            |
+---------------+
```

```
mysql>SELECT 47 % 5 FROM DUAL;
+---------------+
| 47 % 5        |
+---------------+
|    2          |
+---------------+
```

The modulus (%) operator returns the remainder as the answer after performing the division operation. Hence, the above statement, 47 % 5 shall return the value 2 as the output.

POINT TO REMEMBER

In the above statement, DUAL is the default table in MySQL. It is a one-row, one-column dummy table.

Evaluating Scalar expression with SELECT statement

MySQL permits calculations on the contents of the columns and then displays the calculated result using SELECT statement. We can write scalar expression and constant values for the selected columns. If we are taking NULL value in the expression, it shall result in NULL only. Along with NULL, arithmetic operators can be used while evaluating scalar expressions.

For example,

mysql>`SELECT Rollno, Name, Marks + 10 FROM Student;`

The above command, on execution, shall increment the value for all the rows of the field Marks by 10 and shall display the Rollno, Name and Marks for all the students increased by 10.

Resultant table: Student

Rollno	Name	Marks + 10
1.	Raj Kumar	103
2.	Deep Singh	108
3.	Ankit Sharma	86
4.	Radhika Gupta	88
5.	Payal Goel	92
6.	Diksha Sharma	90
7.	Gurpreet Kaur	75
8.	Akshay Dureja	100
9.	Shreya Anand	80
10.	Prateek Mittal	85

10 rows in a set (0.02 sec)

(b) **Relational Operators**

A relational (comparison) operator is a mathematical symbol which is used to compare two values of the same or compatible data types. Comparison operators are used for conditions where two expressions are required to be compared with each other, which results in either true or false. They are used with WHERE clause.

The following table describes different types of comparison operators in SQL:

OPERATOR	DESCRIPTION
=	Equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
<>, !=	Not equal to

For comparing character data type values, < means earlier in the alphabetical sequence and > means later in the alphabetical sequence.

For example,

```
mysql>SELECT Rollno, Name, Marks
      FROM Student
      WHERE Marks>=90;
```

The above command shall display the Rollno, Name and Marks of all the students with marks either equal to or greater than 90.

Resultant table: Student

Rollno	Name	Marks
1.	Raj Kumar	93
2.	Deep Singh	98
3.	Akshay Dureja	90

3 rows in a set (0.02 sec)

For example,

```
mysql>SELECT * FROM Student
      WHERE Stream <> 'Commerce';
```

The above command shall display the records of all the students who are not from Commerce stream.

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no	Stream
1.	Raj Kumar	M	93	17-11-2000	9586774748	Science
3.	Ankit Sharma	M	76	02-02-2000	8567490078	Science
4.	Radhika Gupta	F	78	03-12-1999	9818675444	Humanities
5.	Payal Goel	F	82	21-04-1998	9845639990	Vocational
6.	Diksha Sharma	F	80	17-12-1999	9897666650	Humanities
7.	Gurpreet Kaur	F	65	04-01-2000	7560875609	Science
9.	Shreya Anand	F	70	08-10-1999	8876543988	Vocational
10.	Prateek Mittal	M	75	25-12-2000	9999967543	Science

8 rows in a set (0.02 sec)

Thus, while using relational operators in a WHERE clause with a SELECT statement, the database program goes through the entire table checking each record one by one and compares with the condition specified. If it is true, the corresponding row is selected for display, otherwise ignored.

CTM: While comparing character, date and time data using relational operators, it should be enclosed in single quotation marks.

(c) Logical Operators

The SQL logical operators are the operators used to combine multiple conditions to narrow data selected and displayed on the basis of the condition specified in an SQL statement. Logical operators are also known as Boolean operators. The three logical operators in SQL are—AND, OR and NOT operator. Out of these, AND and OR operators are termed as Conjunctive operators since these two operators combine two or more conditions. The AND and OR operators are used to filter records based on more than one condition.

These operators provide a means to make multiple comparisons with different operators in the same SQL statement.

CTM: The order of precedence for logical operators (AND, OR, NOT operator) is NOT(!), AND(&&) and OR(||).

1. *AND operator*

   The AND operator displays a record and returns a true value if all the conditions (usually two conditions) specified in the WHERE clause are true.

Condition 1	Condition 2	Result (AND operation)
True	True	True
True	False	False
False	True	False
False	False	False

As shown in the table, when both condition 1 and condition 2 are true, then only the result is true. If either of them is false, the result becomes false.

For example, to list the details of all the students who have secured more than 80 marks and are male.

```
mysql>SELECT * FROM Student
      WHERE Marks > 80 and Gender= 'M';
```

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no	Stream
1.	Raj Kumar	M	93	17-11-2000	9586774748	Science
2.	Deep Singh	M	98	22-08-2000	8988886577	Commerce
8.	Akshay Dureja	M	90	05-05-1997	9560567890	Commerce

3 rows in a set (0.02 sec)

2. *OR operator*

The OR operator displays a record and returns a true value if either of the conditions (usually two conditions) specified in the WHERE clause is true.

Condition 1	Condition 2	Result (OR operation)
True	True	True
True	False	True
False	True	True
False	False	False

As shown in the table, when either condition 1 or condition 2 is true, the result is true. If both of them are false, then only the result becomes false.

For example, to display the roll number, name and stream of all the students who are in either Science or Commerce stream.

```
mysql>SELECT Rollno, Name, Stream FROM Student
      WHERE Stream = 'Science' OR Stream = 'Commerce';
```

Resultant table: Student

Rollno	Name	Stream
1.	Raj Kumar	Science
2.	Deep Singh	Commerce
3.	Ankit Sharma	Science
7.	Gurpreet Kaur	Science
8.	Akshay Dureja	Commerce
10.	Prateek Mittal	Science

6 rows in a set (0.02 sec)

3. *NOT operator*

NOT operator is also termed as a negation operator. Unlike the other two operators, this operator takes only one condition and gives the reverse of it as the result. It returns a false value if the condition holds true and vice versa.

The NOT operator displays a record and returns a true value if either of the conditions (usually two conditions) specified in the WHERE clause is true.

Condition	Result (NOT operation)
True	False
False	True

As shown in the table, when the condition is true, the result is false. If the condition is false, then the result becomes true.

For example, to display the name and marks of all the students who are not in the vocational stream.

```
mysql>SELECT Name, Marks FROM Student
      WHERE NOT (Stream = 'Vocational');
```

Resultant table: Student

Name	Marks
Raj Kumar	93
Deep Singh	98
Ankit Sharma	76
Radhika Gupta	78
Diksha Sharma	80
Gurpreet Kaur	65
Akshay Dureja	90
Prateek Mittal	75

8 rows in a set (0.02 sec)

(d) **SQL Special Operators**

There are some special operators in SQL that perform some specific functions.

1. *Conditions Based on a Range—BETWEEN...AND*

SQL provides a BETWEEN operator that defines a range of values that the column value must fall within for the condition to become true. The range includes both the lower and upper value. The values can be numbers, text or dates.

Syntax for BETWEEN:

```
mysql>SELECT <column_name(s)>
      FROM <table_name>
      WHERE <column_name> BETWEEN <value1> AND <value2>;
```

For example,

```
mysql>SELECT Rollno, Name, Marks FROM Student
      WHERE Marks BETWEEN 80 and 100;
```

The above command displays Rollno, Name along with Marks of those students whose Marks lie in the range of 80 to 100 (both 80 and 100 are included in the range).

Resultant table: Student

Rollno	Name	Marks
1.	Raj Kumar	93
2.	Deep Singh	98
5.	Payal Goel	82
6.	Diksha Sharma	80
8.	Akshay Dureja	90

5 rows in a set (0.02 sec)

☞ **NOT BETWEEN**

The NOT BETWEEN operator works opposite to the BETWEEN operator. It retrieves the rows which do not satisfy the BETWEEN condition.

For example,

```
mysql>SELECT Rollno, Name, Marks FROM Student
      WHERE Marks NOT BETWEEN 80 AND 100;
```

Resultant table: Student

Rollno	Name	Marks
3.	Ankit Sharma	76
4.	Radhika Gupta	78
9.	Shreya Anand	70
10.	Prateek Mittal	75

4 rows in a set (0.02 sec)

2. *Conditions Based on a List—IN*

   To specify a list of values, IN operator is used. This operator selects values that match any value in the given list. The SQL IN condition is used to help reduce the need for multiple **OR conditions** in a SELECT statement.

   Syntax for IN: SELECT <column_name(s)>
   FROM <table_name>
   WHERE <column_name> IN (value1,value2,...);

   For example,

```
mysql>SELECT * FROM Student WHERE Stream IN ('Science', 'Commerce',
      'Humanities');
```

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no	Stream
1.	Raj Kumar	M	93	17-11-2000	9586774748	Science
2.	Deep Singh	M	98	22-08-1996	8988886577	Commerce
3.	Ankit Sharma	M	76	02-02-2000	8567490078	Science
4.	Radhika Gupta	F	78	03-12-1999	9818675444	Humanities
6.	Diksha Sharma	F	80	17-12-1999	9897666650	Humanities
7.	Gurpreet Kaur	F	65	04-01-2000	7560875609	Science
8.	Akshay Dureja	M	90	05-05-1997	9560567890	Commerce
10.	Prateek Mittal	M	75	25-12-2000	9999967543	Science

8 rows in a set (0.02 sec)

The above command displays all those records whose Stream is either Science or Commerce or Humanities.

NOT IN

The NOT IN operator works opposite to IN operator. It matches, finds and returns the rows that do not match the list.

For example,

```
mysql>SELECT * FROM Student WHERE Stream NOT IN ('Science', 'Commerce',
    'Humanities');
```

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no	Stream
5.	Payal Goel	F	82	21-04-1998	9845639990	Vocational
9.	Shreya Anand	F	70	08-10-1999	8876543988	Vocational

2 rows in a set (0.02 sec)

3. *Conditions Based on Pattern—LIKE*

The LIKE operator is used to search for a specified pattern in a column. This operator is used with the columns of type CHAR and VARCHAR. The LIKE operator searches the column to find if a part of this column matches the string specified in the parentheses after the LIKE operator in the command.

Conditions Based on Pattern—WILD CARD CHARACTERS

The SQL LIKE condition allows you to use wild cards to perform pattern matching. SQL provides two wild card characters that are used while comparing the strings with LIKE operator:

(a) Percent (%): Matches any string

(b) Underscore (_): Matches any one character

Syntax for LIKE: SELECT <column_name(s)>
FROM <table_name>
WHERE <column_name> LIKE <pattern>;

For example,

```
mysql>SELECT * FROM Student WHERE Name LIKE "D%";
```

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no	Stream
2.	Deep Singh	M	98	22-08-1996	8988886577	Commerce
6.	Diksha Sharma	F	80	17-12-1999	9897666650	Humanities

2 rows in a set (0.02 sec)

The above command shall display those records where the name begins with the character 'D'.

```
mysql>SELECT * FROM Student WHERE Name LIKE "%a";
```

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no	Stream
3.	Ankit Sharma	M	76	02-02-2000	8567490078	Science
4.	Radhika Gupta	F	78	03-12-1999	9818675444	Humanities
6.	Diksha Sharma	F	80	17-12-1999	9897666650	Humanities
8.	Akshay Dureja	M	90	05-05-1997	9560567890	Commerce

4 rows in a set (0.02 sec)

The above command shall display the records for those students whose name ends with the letter 'a'.

```
mysql>SELECT * FROM Student WHERE Name LIKE "%e%";
```

Resultant table: Student

Rollno	Name	Gender	Marks	DOB	Mobile_no	Stream
2.	Deep Singh	M	98	22-08-1996	8988886577	Commerce
5.	Payal Goel	F	82	21-04-1998	9845639990	Vocational
7.	Gurpreet Kaur	F	65	04-01-2000	7560875609	Science
8.	Akshay Dureja	M	90	05-05-1997	9560567890	Commerce
9.	Shreya Anand	F	70	08-10-1999	8876543988	Vocational
10.	Prateek Mittal	M	75	25-12-2000	9999967543	Science

6 rows in a set (0.02 sec)

As the resultant table shows, the above command displays the records of all the students whose name contains the character 'e' anywhere in it.

```
mysql>SELECT * FROM Student WHERE Name LIKE "_e%";
```

Resultant table: Student

Rollno	Name	DOB
2.	Deep Singh	22-08-1996

1 row in a set (0.02 sec)

This command shall display the Rollno, Name and DOB of all the students whose name contains the letter 'e' at the second place.

☞ **NOT LIKE**

The NOT LIKE operator works opposite to LIKE operator. It matches, finds and returns the rows that do not match the specified pattern.

```
mysql>SELECT Rollno, Name, Marks, DOB FROM Student
      WHERE Name NOT LIKE "%r_ _";
```

Resultant table: Student

Rollno	Name	Marks	DOB
1.	Raj Kumar	93	17-11-2000
2.	Deep Singh	98	22-08-1996
4.	Radhika Gupta	78	03-12-1999
5.	Payal Goel	82	21-04-1998
7.	Gurpreet Kaur	65	04-01-2000
8.	Akshay Dureja	90	05-05-1997
9.	Shreya Anand	70	08-10-1999
10.	Prateek Mittal	75	25-12-2000

8 rows in a set (0.02 sec)

This command shall display the Rollno, Name, Marks and DOB of all the students whose name does not contain the letter 'r' from the third last position.

4. *SORTING IN SQL—ORDER BY*

The SQL ORDER BY clause is used to sort the data in ascending or descending order based on one or more columns. The ORDER BY keyword is used to sort the result-set by one or more fields in a table. This clause sorts the records in the ascending order (ASC) by default. Therefore, in order to sort the records in descending order, DESC keyword is to be used. Sorting using ORDER BY clause can be done on multiple columns, separated by comma.

Syntax for ORDER BY Clause:

SELECT<column-list> FROM <table_name> [WHERE <condition>] ORDER BY <column_name> [ASC|DESC];

Here, WHERE clause is optional.

For example,

(i) To display the roll number, name and marks of students on the basis of their marks in the ascending order.

```
mysql>SELECT Rollno, Name, Marks FROM Student
      ORDER BY Name;
```

Rollno	Name	Marks
8	Akshay Dureja	90
3	Ankit Sharma	76
2	Deep Singh	98
6	Diksha Sharma	80
7	Gurpreet Kaur	65
5	Payal Goel	82
10	Prateek Mittal	75
4	Radhika Gupta	78
1	Raj Kumar	93
9	Shreya Anand	70

(ii) To display the roll number, name and marks of all the students in the descending order of their marks and ascending order of their names.

```
mysql>SELECT Rollno, Name, Marks FROM Student
      ORDER BY Marks DESC, Name;
```

MEMORY BYTES

- SQL is Structured Query Language that is used to create, modify and access a database.
- The various processing capabilities of SQL are: Data Definition Language (DDL), Data Manipulation Language (DML), Data Control Language (DCL) and Data Query Language (DQL).
- DDL is used to create and delete tables, views or indexes.
- DML is used to modify and update the rows of the table.
- DESCRIBE or DESC is used to show the structure of a table.
- The SELECT statement is used to fetch data from one or more database tables.
- SELECT * means display all columns.