By default, the ADD clause adds a column at the end of the table. Use the AFTER keyword to add a column at a particular position.

```
ALTER TABLE Customer ADD Date_of_Birth date AFTER `last_name`;
```

- Dropping a column from the table:-

```
ALTER TABLE table_name DROP COLUMN column_name
```

- Renaming a table:-

```
RENAME TABLE current_table_name TO new_table_name
```

- ADD, MODIFY, RENAME, CHANGE, DROP, ADD COLUMN are a subset of Alter.

SQL Constraints :-

SQL Constraints are used to specify rules for data in a table

```
CREATE TABLE table_name(
    column1 datatype constraints,
    column2 datatype constraint,
    ....);
```

- Not Null :- Ensures that a column cannot have a NULL value.
- UNIQUE :- Ensures that all values in a column are different.
- PRIMARY KEY :- A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table.
- Foreign Key :- Uniquely identifies a row/record in another table.
- CHECK :- Ensures that all values in a column satisfies a specific condition.
- DEFAULT :- Sets a default value for a column when no value is specified
- INDEX :- Used to create and retrieve data from the database quickly.

SQL Not NULL Constraints :-

- By default, a column can hold null values.
- The NOT NULL Constraint enforces a column to not accept NULL values.
- You cannot insert a new record, or update a record without adding a value to this field.

Unique Constraints:-

- ensures all values in the columns are different.
- A primary key constraint automatically has a unique constraint.
- we can have many UNIQUE constraints per table, but only one PRIMARY KEY constraint.

PRIMARY Key Constraints:-

- The PRIMARY KEY constraint uniquely identifies each record in a table.
- Primary keys should contain UNIQUE values, and cannot contain NULL values.
- A table can have only one primary key.

FOREIGN KEY constraint:-

- A Foreign key is a key used to link two tables together.
- It is a field (or collection of fields) in one table that refers to PRIMARY keys in another table.
- The table containing the foreign key is called the child table.
- The table containing the candidate key is called the referenced or parent table.
- The FOREIGN KEY constraint is used to prevent:
  - actions that would destroy links between tables.
  - Invalid data being inserted into the foreign key column, because it has to be one of the values contained in the table it points to.

- To create a primary key constraint on a column when the table is already created, we use the ADD clause with the ALTER command.

```
ALTER TABLE table_name ADD PRIMARY KEY (column_name);
```

- To drop a PRIMARY KEY constraint from an already created table,

```
ALTER TABLE table_name DROP PRIMARY KEY;
```

COMPOUND SEARCH CONDITIONS EXAMPLES

- SELECT * FROM employee.distributor
where (state = 'maharastra' AND
distributor_id <> 7000)
OR (distributor_id = 1000);

- UPDATE employee.distributor SET state =
'rajasthan'
where distributor_id = 6000 OR
(distributor_id > 5000 AND City <>
'Unknown');

• WHERE Clause Predicates

The where clause supports the following types of predicates:

• Comparison = Equal <> Not Equal < Less than <= Less than or equal	Pattern Matching • LIKE (used for wildcard filtering)	• BETWEEN • IN • IS NULL

Examples:-

• SELECT UserID, Name, Salary FROM tblUser where Salary > 50000

• SELECT * FROM tblUser WHERE city = 'KOLKATA';

• SELECT UserId, Name FROM tblUser WHERE UserId BETWEEN 33 AND 36;

• SELECT UserId, Name FROM tblUser WHERE UserId IN (4, 22, 36);

• IS NULL is used to select rows if a specified column value contains a null value.

SELECT * FROM tblUser WHERE City IS NULL

SET OPERATIONS

- Used to join the result of two (or more) SELECT statements.
- Types are
  - Union
  - UnionAll
  - Intersect
  - Minus

Union

- The Union clause/operator is used to combine the result of two or more SELECT statements with identical columns without returning any duplicate rows.

```
SELECT col1, col2, col3 FROM tables [where conditions]
UNION
SELECT col1, col2, col3 FROM tables [where conditions];
```

- There should be same number of columns in both SELECT statements.
- The column names from the first SELECT statement in the UNION operator are used as the column names for the result set.

Union All :- Union All doesn't remove any duplicates.

- INTERSECT OPERATION :- (not in mysql)

Used to combine two SELECT statements with identical columns and returns rows only common rows returned by the two select statements.

- MINUS OPERATION :- (not in mysql)

Combines results of two SELECT statements and returns only those in the final result, which belongs to the first set of result.

- DUPLICATE ROWS

```
SELECT DISTINCT column_list FROM table-name
```

Example:
```
SELECT DISTINCT * FROM store
```

Operator Precedence :-

- To display employees having salary greater than 1800 and whose name starts with either 's' or 'p', we write the below query:

```
SELECT * FROM EMPLOYEE
WHERE Salary > 1800 and (Name like 's%' or Name like 'p%');
```

• Numerical Function - Syntax :-

```
SELECT numerical_expression as OPERATION_NAME
[FROM table_name WHERE CONDITION];
```

• ORDER BY :- The order by clause causes the tuples in the result of a query to appear in sorted order.

```
SELECT UserID, Name, Salary FROM tblUser ORDER BY Salary
```

* The salary is in the ascending order here.

* By default, the order by clause lists items in ascending order.

If we want the salary in descending order :-

```
SELECT UserID, Name, Salary FROM tblUser ORDER BY Salary desc
```

If we want ascending order of one column and descending order of another :-

```
SELECT UserID, Name, Salary FROM tblUser ORDER BY City desc, Salary asc
```

Other Syntaxes :-

SQL Built-in Functions :-

- String
- Numeric
- Date
- Bin
- Cast

Numeric Function :-

```
SELECT numerical_expression as OPERATION_NAME
[FROM table_name WHERE CONDITION];
```

Example :-

```
SELECT (25+7) AS ADDITION;
SELECT (cos1) AS COS;
SELECT 20DIV6 As DIVISION;
```

Wildcard Filtering :-

They are often used with the operators like LIKE and NOT LIKE in conjunction with the WHERE clause.

• % wildcard is used in searching or filtering a record by matching any string of zero or more characters.

```
SELECT statement .... WHERE column_name LIKE 'XXX%'
```

Here :-

• 'SELECT statement' is the standard SQL SELECT command.

• "WHERE" is the keyword used to apply the filter.

• "LIKE" is the comparison operator used in conjunction with wildcards.

- Types of Wildcards:

Type	Description
%	The percent character indicates any character with any number of counts
-	The hyphen character indicates a range of the character within the braces [a-c]
_	The underscore character indicates any one character
[]	The square bracket indicates any one value with the brackets
^	The caret character indicates all the character except available in the brackets: [^xyz]
#	The hash sign indicates a single numeric character

- 'xxx' is any specified starting pattern such as single character or more and "%" matches any number of characters starting from zero(0).
- % can be used in the first place, last place or both the sides.
- ("%xxx") , ("xxx%"), ("%xxx%").

For example:-

1) If we want to search a movie whos name is starting with eternal.

```
SELECT * FROM movies WHERE title LIKE 'eternal%';
```

2) If a movie title has 'harry potter' in the middle

```
SELECT * FROM movies WHERE title LIKE '%harry potter%';
```

3) Below is syntax where we get all film-noir movies from year 1990.

```
SELECT * FROM movies WHERE genres LIKE '%film-noir%' AND
                           title LIKE "%1990%"
```

- The underscore (_) wildcard is used to match exactly one character.

```
SELECT statement .... WHERE column-name LIKE 'xxx_'
```

- Below is the SQL statement that can help you to take a look at all movie names from the year 2010 that belongs only to the children category.

```
SELECT * FROM movies WHERE genres LIKE "children_" and title like '%2010%';
```

- Script that retrieves movie name not belonging to 1900s.

```
SELECT * FROM movies WHERE title NOT LIKE '%19__%';
```

- Escape Operator:-

We use the below query to retrieve movie name that has the % character in its name:

```
SELECT * FROM movies WHERE title LIKE '%#%%' ESCAPE '#';
```

. Here note that the double "%%" in the LIKE predicate, the first one in red"%" is treated as part of the string to be searched for. The other one is used to match any number of characters that follow.