SQL

Data Definition Language:

Create:

```
Create Table employees (
   id INT PRIMARY KEY AUTO_INCREMENT,
   name VARCHAR(30),
   department VARCHAR(20),
   salary int
);
```

It is used to create a table.

Alter:

```
Alter table employees
ADD hire_date DATE;
```

It is used add new column in the existing table.

Drop:

```
Drop table employees;
```

It delete the table from database.

Truncate:

```
Truncate table employees;
```

It delete the all values in the table.

Data Manipulation Language :

Insert:

```
Insert into employees (Name, Department, Salary)
 Values ('Sam', 'HR', 50000);
```

Used to insert values in the table according to the columns.

Update:

Update employees set Salary = 40000

where Name = 'Tony';

Used to update the existing value in the table.

Delete:

Delete from employees

where Name = 'George';

Deletes the existing value from the table

Select:

Select * from employees;

Displays all the column value in the given table.

Data Query Language:

Select:

Select Name, Department from employees

where Salary > 30000;

Displays a specific column value that is given.

Clauses:

Where:

Select * from employees

where Department = 'HR';

Displays the values from the condition.

Between:

Select * from employees where Salary between 3000 and 7000;

Displays the values between the given range

Group by:

Select Department, count(*) as Employee_Count from employees

Group by Department;

Used group the rows that have the same values in specified columns. Mostly used with aggregate functions.

Having:

Select Department, avg(Salary) as Avg_Salary from employees

Group by Department

Having Avg_Salary > 50000;

Having is used instead of where while working with group.

Order by:

Select * from employees

Order by Salary desc;

Displays the column values either in ascending or descending order.

Limit:

Select * from employees

Order by Salary desc limit 3;

Display the values in a limited manner according to the limit values given.

Distinct:

Select distinct Department from employees;

Displays the duplicate values as a single while a same column have duplicate values.

In:

Select * from employees

where Department in ('HR', 'IT');

Displays values which matches the given condition

Like:

Select * from employees

where Name like 'A%';

Display rows based on a pattern using wildcard characters (% for multiple characters, _ for a single character).

Is null / Is not null:

Select * from employees where Department is null;

Displays the rows which is not null and null.

Constains:

Primary key:

Create table departments (

Dept_Id int primary key auto_increment,

Dept_Name varchar(30)

);

Does not allows duplicate and null values. Used to uniquely identify a values in the table.

Foreign key:

Create table employees (

Id int primary key auto_increment,

Name varchar(50),

Dept_Id int,

foreign key (Dept_Id) references departments(Dept_Id)

);

Creates a relationship between tables. It refers the table using primary key.

Unique key:

Alter table employees

add unique (Name);

Does not allows duplicate values but allows null values.(While inserting)

Check:

Alter table employees

add check (Salary > 0);

Checks the condition while inserting the values if doesn't meet condition it throws error.

Not null:

Alter table employees

modify Name varchar(50) not null;

Does not allows null values while inserting but allows duplicate values.

SQL Aggregation Functions:

Sum:

Select sum(Salary) as Total_Salary from employees;

Display the total sum of the given column.

Average:

Select avg(Salary) as Average_Salary from employees;

Displays the average value of the given column.

Count:

Select count(*) as Employee_Count from employees;

Displays the total count of the given column.

Minimum:

Select min(Salary) as Lowest_Salary from employees;

Displays least or minimum value from the given column.

Maximum:

Select max(Salary) as Highest_Salary from employees;

Displays the highest or maximum value from the column.

SQL Operators

Arithmetic operators:

Select Salary + 1000 as New_Salary from employees;

Does arithmetic operation

Comparison operators:

Select * from employees

where Salary > 3000;

Does Comparison operation.

Logical operators:

Select * from employees

where Department = 'HR' and Salary > 3000;

Does logical operation.

Joins:

Inner Join:

Select employees.Name, departments.Dept_Name

from employees inner join departments on employees.Dept_Id = departments.Dept_Id;

Displays all the matching values from the joined tables.

Left Join:

Select employees.Name, departments.Dept_Name
from employees left join departments on employees.Dept_Id = departments.Dept_Id;
Displays all the values from the left side table and only matching values from the right side table

Right Join:
Select employees.Name, departments.Dept_Name
from employees right join departments on employees.Dept_Id = departments.Dept_Id;
Displays all the values from the right side table and only matching values from the left side table.

Transaction Control Language:

Commit:
Start transaction;
Insert into employees (Name, Department, Salary) values ('Ram', 'Finance', 30000);
commit;
Save changes made in a transaction.

Rollback:
Start transaction;
Insert into employees (Name, Department, Salary) values ('Silva', 'IT', 40000);
rollback;
Undo changes made in a transaction.

Savepoint:
start transaction;
savepoint sp1;
Insert into employees (Name, Department, Salary) values ('Jebin', 'HR', 55000);
rollback to sp1;
A savepoint within a transaction.

Data Control Language:

Grant:

Grant Select, Insert On Mydatabase.Employees To 'Silva'@'localhost';

Grant specific privileges to a user.

Revoke:

Revoke Insert On Mydatabase.Employees From 'Silva'@'localhost';

Revoke specific privileges from a user.