**SQL**

SQL- Structure Query Language

DBMS- Data base management system

RDMS- Relational data base management system

**SQL**  is a language to interact with computer to store data,it is of 5 types:

**TYPES OF SQL**

**DDL** = Data definition language.

**DML** = Data manipulation language.

**DCL** = Data control language.

**DQL** = Data query language.

**TCL** = Transaction control language.

DDL-create,alter,drop,truncate.(outer structure)

DML-insert,update,delete.(inner structure,data access)

DCL-grant,revoke.(giving permissions to data)

TCL-rollback(undo),commit(permanently save the data),savepoint(rollback particular).

Local host- data can be used within the system server only.

**Basic comments in SQL**

1. \L - to show the list of databases.
2. \dt - to show the table.
3. \c databasename- to connect with data base.

**DDL COMMENTS:**

1. Creation of database

CREATE DATABASE databasename.

1. Creation of table

CREATE TABLE tablename(column1 name datatype ,column2 name varchar(size));

1. Table name modification using alter.

ALTER table tablename rename to newname;

1. To add new column to table.

ALTER table tablename add columnname columntype;

1. To delete a existig column.

ALTER table tablename DROP columnname;

1. To delete values from table.

TRUNCATE tablename;

**DML COMMENTS:**

1. To insert values into table.

INSERT into tablename values(val1,val2,val3);

1. To update values into table.

UPDATE from tablename columnname= ‘value’ where uniquecolumnname= ‘value’;

1. To delete values from table.

DELETE from tablename columname= ‘value’ where uniquecolumnname= ‘value’;

**ORDERBY COMMENT:**

Select \* from tablename order by columname;

To order by ascending:

Select \* from tablename order by columname asc;

To order by descending:

Select \* from tablename order by columname desc;

To order by certain limit:

Select \* from tablename limit value;

**TO CHECK NULL VALUES:**

Select \* from tablename where columnname isnull;

LIKE COMMENT:

‘%letter’- name ends with s.

‘letter%’-name starts with s.

‘%letter%’-name with s in middle.

Select \* from tablename columnname like % ‘value’;

IN-To check whether the value is present in column or table.

BETWEEN-To check between certain range.

AS-rename as with alias

GROUPBY- The GROUP BY statement is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns

**Joins**

The join is used to join two tables which has atleast one common column.The joins have 5 types.

1)Inner join - Returns only matching rows from both tables.

SELECT column1, column2, ...FROM table1 INNER JOIN table2 ON table1.common\_column = table2.common\_column;

2)Left join or Left outter join - Returns all rows from the left table and matching rows from the right table, with NULL for non-matching rows from the right table.

SELECT column1, column2, … FROM table1 LEFT JOIN table2 ON table1.common\_column = table2.common\_column;

3)Right join or Right outter join - Returns all rows from the right table and matching rows from the left table, with NULL for non-matching rows from the left table.

SELECT column1, column2, … FROM table1 RIGHT JOIN table2 ON table1.common\_column = table2.common\_column;

4)Full join or Full outer join - Returns all rows when there’s a match in one of the tables, with NULL for non-matching rows from both tables.

SELECT column1, column2, … FROM table1 FULL JOIN table2 ON table1.common\_column = table2.common\_column;

5)Cross join - Returns the Cartesian product of both tables, combining every row from the first table with every row from the second.

SELECT column1, column2, … FROM table1 CROSS JOIN table2;

**RANKING**

RANK() - Assigns a rank to each row within a partition, but if there are ties (duplicate values), it will leave gaps in the ranking.

RANK() OVER (PARTITION BY column\_name ORDER BY column\_name [ASC|DESC])

DENSE RANK() - DENSE\_RANK() is similar to RANK(), but it doesn't leave gaps in the ranking sequence. If there are ties, the next rank is immediately after the tied rank.

DENSE\_RANK() OVER (PARTITION BY column\_name ORDER BY column\_name [ASC|DESC])

ROW\_NUMBER() - ROW\_NUMBER() assigns a unique sequential integer to each row, regardless of any ties. This means that even if two rows have the same values in the ORDER BY clause, each row gets a distinct number.

ROW\_NUMBER() OVER (PARTITION BY column\_name ORDER BY column\_name [ASC|DESC])

COMBINATION OF ALL THE 3

SELECT Name, Dept, Salary, RANK() OVER (PARTITION BY Dept ORDER BY Salary DESC)AS rank, DENSE\_RANK() OVER (PARTITION BY Dept ORDER BY Salary DESC) AS dense\_rank, ROW\_NUMBER() OVER (PARTITION BY Dept ORDER BY Salary DESC) AS row\_number FROM Employees;