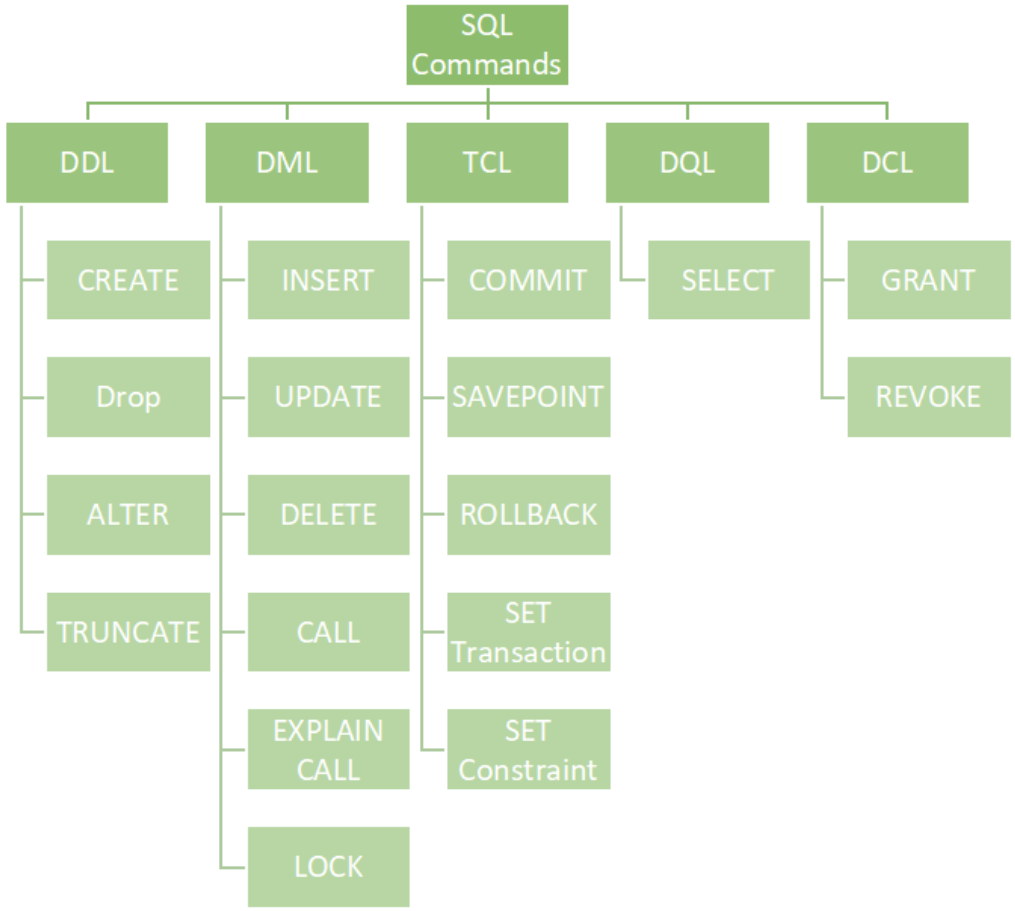
Priyanka Salvi

Batch Code:7670

SQL: Structured Query Language(SQL) as we all know is the database language by the use of which we can perform certain operations on the existing database and also we can use this language to create a database. SQL uses certain commands like Create, Drop, Insert, etc. to carry out the required tasks.
These SQL commands are mainly categorized into four categories as:

1. DDL – Data Definition Language
2. DQl – Data Query Language/ DRL Data Retrieval Language
3. DML – Data Manipulation Language
4. DCL – Data Control Language

Though many resources claim there to be another category of SQL clauses **TCL – Transaction Control Language**. So we will see in detail about TCL as well.



DDL (Data Definition Language):

DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in the database. DDL is a set of SQL commands used to create, modify, and delete database structures but not data. These commands are normally not used by a general user, who should be accessing the database via an application.
List of DDL commands:

- **CREATE**: This command is used to create the database or its objects (like table, index, function, views, store procedure, and triggers).
- **DROP**: This command is used to delete objects from the database.
- **ALTER:** This is used to alter the structure of the database.
- **TRUNCATE:** This is used to remove all records from a table, including all spaces allocated for the records are removed.
- **COMMENT**: This is used to add comments to the data dictionary.
- **RENAME:** This is used to rename an object existing in the database.

DQL (Data Query Language): Data Retrieval language.

DQL statements are used for performing queries on the data within schema objects. The purpose of the DQL Command is to get some schema relation based on the query passed to it. We can define DQL as follows it is a component of SQL statement that allows getting data from the database and imposing order upon it. It includes the SELECT statement. This command allows getting the data out of the database to perform operations with it. When a SELECT is fired against a table or tables the result is compiled into a further temporary table, which is displayed or perhaps received by the program i.e. a front-end.
List of DQL:

- **SELECT:** It is used to retrieve data from the database.

DML(Data Manipulation Language):

The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements. It is the component of the SQL statement that controls access to data and to the database. Basically, DCL statements are grouped with DML statements.

List of DML commands:

- **INSERT** : It is used to insert data into a table.
- **UPDATE:** It is used to update existing data within a table.
- **DELETE** : It is used to delete records from a database table.
- **LOCK:** Table control concurrency.
- **CALL:** Call a PL/SQL or JAVA subprogram.
- **EXPLAIN PLAN:** It describes the access path to data.

DCL (Data Control Language):

DCL includes commands such as GRANT and REVOKE which mainly deal with the rights, permissions, and other controls of the database system.

List of DCL commands:

- **GRANT:** This command gives users access privileges to the database.

- **REVOKE:** This command withdraws the user's access privileges given by using the GRANT command.

Though many resources claim there to be another category of SQL clauses TCL – Transaction Control Language. So we will see in detail about TCL as well. TCL commands deal with the transaction within the database.
List of TCL commands:

- **COMMIT:** Commits a Transaction.
- **ROLLBACK:** Rollbacks a transaction in case of any error occurs.
- **SAVEPOINT:**Sets a savepoint within a transaction.
- **SET TRANSACTION:** Specify characteristics for the transaction.

https://www.oracle.com/database/technologies/xe-prior-release-downloads.html

https://youtu.be/seFRL1GAzLY

https://www.testingdocs.com/download-install-mysql-on-windows-11/

https://dev.mysql.com/downloads/installer/

https://youtu.be/eq-e_n7lm2M

https://www.youtube.com/watch?v=WuBcTJnIuzo

https://www.youtube.com/watch?v=wEHWYuzP7VE

DDL (Data Definition Language)

- **Create:**

Syntax : create table tablename(col1 datatype,col2 datatype,….);

EX: create table Employees1(EmpId number, FirstName varchar(20), LastName varchar(20), EmailId Varchar(50), Gender char(1), MobileNo char(10));

```
SQL> create table Employees1(
  2  EmpId number,
  3  FirstName varchar(20),
  4  LastName varchar(20),
  5  EmailId Varchar(50),
  6  Gender char(1),
  7  MobileNo char(10));

Table created.
```

- **Alter:**

Syntax: ALTER TABLE table_name

ADD new_column_name column_definition

Example: alter table Employees1 add Age varchar(5);

```
SQL> alter table Employees1 add Age varchar(5);
```

```
    EMPID FIRSTNAME           LASTNAME            EMAILID
       G MOBILENO   AGE
--------- -------------------- -------------------- --------------------------------------------
- ---------- -----
       10 Priyanka            Salvi               salvipriyanka1710@gmail
       F 9930160922
       20 Pawan               Salvi               salvipawan2610@gmail
       M 9881262806
       30 Mohini              Chavan              mohinichavan123@gmail
       F 9763910985
       40 Pooja               Patil               patilpooja124@gmail
       F 9987115055
       50 Ankita              Jadhav              ankitajadhav126@gmail
       F 7021654042
```

- **Rename:**
  Syntax: rename Old _table To New_table;
  Example: rename employees1 To customer;

```
SQL> rename employees1 To customer;

Table renamed.
```

- **Truncate**
  Syntax: Truncate table table_name;
  Example: Truncate table Customer;

```
SQL> truncate table customer;

Table truncated.

SQL> select * from customer;

no rows selected
```

DQL (Data Query Language)

- **Select:**

Syntax: select * from table name;

Example: select * from Employees1;

```
SQL> select * from Employees1;

    EMPID FIRSTNAME           LASTNAME            EMAILID                                            G MOBILENO
--------- ------------------- ------------------- -------------------------------------------------- - ----------
       10 Priyanka            Salvi               salvipriyanka1710@gmail                            F 9930160922
       20 Pawan               Salvi               salvipawan2610@gmail                               M 9881262806
       30 Mohini              Chavan              mohinichavan123@gmail                              F 9763910985
       30 Pooja               Patil               patilpooja124@gmail                                F 9987115055
```

`

DML(Data Manipulation Language)

1. **Insert**

Syntax: INSERT INTO table_name (column_1, column_2, column_n)

VALUES (expression_1, expression_2, ... expression_n)

Example: insert into Employees1 (EmpId, FirstName,LastName,EmailId, Gender, MobileNo) values (010, 'Priyanka', 'Salvi', 'salvipriyanka1710@gmail', 'F', 9930160922);

```
SQL> insert into Employees1 (EmpId, FirstName,LastName,EmailId, Gender, MobileNo) values (010, 'Priyanka', 'Salvi', 'salvipriyanka1710@gmail', 'F', 9930160922);

1 row created.

SQL> insert into Employees1 (EmpId, FirstName,LastName,EmailId, Gender, MobileNo) values (020, 'Pawan', 'Salvi', 'salvipawan2610@gmail', 'M', 9881262806);

1 row created.
```

2. Update:

Syntax: PDATE table1

SET column1 = (SELECT expression1

FROM table2

WHERE conditions)

Example: update Employees1 set EmpId = 40 where MobileNo =9987115055;

```
SQL> update Employees1 set EmpId = 40
  2  where MobileNo =9987115055;

1 row updated.

SQL> select * from Employees1;

     EMPID FIRSTNAME           LASTNAME            EMAILID
        G MOBILENO
---------- -------------------- -------------------- ----------------------------------------
- ----------
        10 Priyanka             Salvi               salvipriyanka1710@gmail
        F 9930160922
        20 Pawan                Salvi               salvipawan2610@gmail
        M 9881262806
        30 Mohini               Chavan              mohinichavan123@gmail
        F 9763910985
        40 Pooja                Patil               patilpooja124@gmail
        F 9987115055
        30 Ankita               Jadhav              ankitajadhav126@gmail
        F 7021654042
```

TCL (Transaction Control Language)

3. Commit:

Syntax: commit;

```
SQL> create table Player(rank number, name varchar(10), best number);

Table created.

SQL> insert into player values(1,'Virat',183);

1 row created.

SQL> insert into player values(2,'Dhoni',183);

1 row created.

SQL> insert into player values(3,'Rohit',264);

1 row created.

SQL> select * from Player;

      RANK NAME              BEST
---------- ---------- ----------
         1 Virat              183
         2 Dhoni              183
         3 Rohit              264

SQL> commit;

Commit complete.
```

4. Savepoint:

Syntax: savepoint_name ;

```
SQL> select * from Player;

      RANK NAME              BEST
---------- ---------- ----------
         1 Virat              183
         2 Dhoni              183
         3 Rohit              264

SQL> savepoint insertion;

Savepoint created.

SQL> insert into player values(4,'Jadeja',334);

1 row created.

SQL> insert into player values(5,'KL_Rahul',252);

1 row created.

SQL> select * from Player;

      RANK NAME              BEST
---------- ---------- ----------
         1 Virat              183
         2 Dhoni              183
         3 Rohit              264
         4 Jadeja             334
         5 KL_Rahul           252

SQL> savepoint Updation;

Savepoint created.
```

5. Rollback:

Syntax: Rollback to savepoint_name;

```
SQL> rollback to insertion;

Rollback complete.

SQL> select * from Player;

      RANK NAME             BEST
---------- ---------- ----------
         1 Virat             183
         2 Dhoni             183
         3 Rohit             264
```

GROUP BY:

Syntax: SELECT column_name(s)

FROM table_name

WHERE condition

GROUP BY column_name(s)

ORDER BY column_name(s);

```
       101 Ram                  IT                50000 Delhi
    800001

       102 Amit                 Research          45000 Mumbai
    800002

       103 Tanu                 Accountant       450000 Delhi
    800003


    EMP_ID EMP_NAME             DEPT           SALARY CITY
---------- -------------------- ---------- ---------- --------------------
   PINCODE
----------
       104 sunil                IT                10000 Kolkata
    800004

       105 sunil                IT                15000 Kolkata
    800005

       106 sonam                HR               115000 Ranchi
    800006


    EMP_ID EMP_NAME             DEPT           SALARY CITY
---------- -------------------- ---------- ---------- --------------------
   PINCODE
----------
       107 sonam                HR                48000 Mumbai
    800007

       108 Priyanka             IT                70000 Banglore
    800008

8 rows selected.

SQL> select emp_name, min(salary) from Employees2 group by emp_name;

EMP_NAME             MIN(SALARY)
-------------------- -----------
Priyanka                   70000
Tanu                      450000
Ram                        50000
Amit                       45000
sunil                      10000
sonam                      48000
```

ORDER BY :

Syntax:

SELECT column1, column2, ...

FROM table_name

ORDER BY column1, column2, ... ASC|DESC;

```
8 rows selected.

SQL> select * from Employees2
  2  order by emp_name desc;

    EMP_ID EMP_NAME            DEPT           SALARY CITY
---------- ------------------- ---------- ---------- --------------------
   PINCODE
----------
       104 sunil               IT              10000 Kolkata
    800004

       105 sunil               IT              15000 Kolkata
    800005

       107 sonam               HR              48000 Mumbai
    800007


    EMP_ID EMP_NAME            DEPT           SALARY CITY
---------- ------------------- ---------- ---------- --------------------
   PINCODE
----------
       106 sonam               HR             115000 Ranchi
    800006

       103 Tanu                Accountant     450000 Delhi
    800003
```

SQL SELECT:

1. SQL SELECT DISTINCT:

Syntax: select distinct column_name from table_name;

Example: SELECT DISTINCT City from stud;

```
SQL> select * from stud;

    STUDID STUDENT_NAME         GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
         1 Ankita Jadhav        Female     1234569871 Gansoli
         2 Pooja Patil          Female     1234569881 Thane
         3 Pratiksha Maurya     Female     1234569891 Thane
         4 Mohini Chavan        Female     1534569891 Nashik
         5 Rutuja Patil         Female     2534569891 Nashik
         6 Amruta Patil         Female     2535469891 Nashik

6 rows selected.

SQL> SELECT DISTINCT City from stud;

CITY
--------------------
Thane
Gansoli
Nashik
```

2. SQL SELECT COUNT

Syntax: select count(column_name)from table_name;

Example: select count(Student_Name)from stud;

```
SQL> select count(Student_Name)from stud;

COUNT(STUDENT_NAME)
-------------------
                  6
```

6. COUNT FUNCTION WITH WHERE CLAUSE IN SQL:

Syntax: SELECT COUNT(column_name) FROM (table_name)WHERE (column_name)condition;

Example: select count(Student_Name) AS totalCityNashik from stud Where City = 'Nashik';

```
SQL> select count(Student_Name) AS totalCityNashik from stud Where City = 'Nashik';

TOTALCITYNASHIK
---------------
              3
```

7. COUNT FUNCTION WITH DISTINCT KEYWORD

Syntax: select count(distinct column_name) from table_name where (condition);

Example:

```
SQL> select count(distinct City) from stud;

COUNT(DISTINCTCITY)
-------------------
                  3
```

3. ROWNUM KEYWORD IN WHERE CLAUSE :

Syntax: SELECT column_Name1,column_Name2, ..., column_NameN FROM table_name WHERE ROWNUM <= value;

```
SQL> select * from Stud;

    STUDID STUDENT_NAME         GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
         1 Ankita Jadhav        Female     1234569871 Gansoli
         2 Pooja Patil          Female     1234569881 Thane
         3 Pratiksha Maurya     Female     1234569891 Thane
         4 Mohini Chavan        Female     1534569891 Nashik
         5 Rutuja Patil         Female     2534569891 Nashik
         6 Amruta Patil         Female     2535469891 Nashik

6 rows selected.

SQL> select * from Stud where rownum <= 4;

    STUDID STUDENT_NAME         GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
         1 Ankita Jadhav        Female     1234569871 Gansoli
         2 Pooja Patil          Female     1234569881 Thane
         3 Pratiksha Maurya     Female     1234569891 Thane
         4 Mohini Chavan        Female     1534569891 Nashik
```

4. **SQL SELECT RANDOM:**
   Syntax: SELECT column FROM
   (SELECT column FROM table
   ORDER BY dbms_random.value)
   WHERE rownum =1

5. **SQL SELECT SUM:**
   Syntax: SELECT SUM (expression)
   FROM tables
   WHERE conditions;

SQL COUNT(), AVG() AND SUM() FUNCTIONS:

1. **COUNT():**
   Syntax: SELECT COUNT(column_name)
   FROM table_name
   WHERE condition;

```
SQL> select count(emp_id)
  2  from Employees2;

COUNT(EMP_ID)
-------------
            8
```

2. **AVG():**
   Syntax: SELECT AVG(column_name)
   FROM table_name
   WHERE condition;

```
SQL> select avg(salary)
  2  from Employees2;

AVG(SALARY)
-----------
     100375
```

3. **SUM()**
   Syntax: SELECT AVG(column_name)
   FROM table_name;

```
SQL> select sum(salary)
  2  from Employees2;

SUM(SALARY)
-----------
     803000
```

SQL LOGICAL OPERATOR:

1. **SQL AND**

   Syntax: SELECT *

   FROM tables _name

   WHERE condition 1 AND condition 2;

```
SQL> select * from Stud;

    STUDID STUDENT_NAME         GENDER     MOBILE_NUM CITY                DEPARTMENT
---------- -------------------- ---------- ---------- -------------------- ----------
         1 Ankita Jadhav        Female     1234569871 Gansoli              EXTC
         2 Pooja Patil          Female     1234569881 Thane                COMP
         3 Pratiksha Maurya     Female     1234569891 Thane                COMP
         4 Mohini Chavan        Female     1534569891 Nashik               COMP
         5 Rutuja Patil         Female     2534569891 Nashik               EXTC
         6 Amruta Patil         Female     2535469891 Nashik               EXTC

6 rows selected.

SQL> select * from Stud where Department ='EXTC' AND City = 'Nashik';

    STUDID STUDENT_NAME         GENDER     MOBILE_NUM CITY                DEPARTMENT
---------- -------------------- ---------- ---------- -------------------- ----------
         5 Rutuja Patil         Female     2534569891 Nashik               EXTC
         6 Amruta Patil         Female     2535469891 Nashik               EXTC

SQL>
```

2. **SQL OR:**

Syntax: SELECT *

FROM tables _name

WHERE condition 1 OR condition 2;

```
SQL> select * from Employees2
  2  where emp_name = 'sonam' or emp_name = 'Priyanka' or emp_name = 'sunil';

    EMP_ID EMP_NAME             DEPT           SALARY CITY
---------- -------------------- ---------- ---------- --------------------
   PINCODE G EMAIL_ID
---------- - --------------------------------------------------
       104 sunil                IT              10000 Kolkata
    800004 M Sunil159@gmail.com

       105 sunil                IT              15000 Kolkata
    800005 M Sunil357@gmail.com

       106 sonam                HR             115000 Ranchi
    800006 F Sonam258@gmail.com


    EMP_ID EMP_NAME             DEPT           SALARY CITY
---------- -------------------- ---------- ---------- --------------------
   PINCODE G EMAIL_ID
---------- - --------------------------------------------------
       107 sonam                HR              48000 Mumbai
    800007 F Sonam369@gmail.com

       108 Priyanka             IT              70000 Banglore
    800008 F Priyanka1710@gmail.com
```

3. SQL NOT:

Syntax: : SELECT *
FROM tables _name
WHERE condition 1 OR condition 2;

```
SQL> select * from stud
  2  where not City = 'Nashik';

    STUDID STUDENT_NAME         GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
DEPARTMENT
----------
         1 Ankita Jadhav        Female     1234569871 Gansoli


         2 Pooja Patil          Female     1234569881 Thane


         3 Pratiksha Maurya     Female     1234569891 Thane
```

4. Like:

Syntax:

SELECT * FROM Customers

WHERE CustomerName LIKE 'a%';

```
SQL> select * from stud;

    STUDID STUDENT_NAME         GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
DEPARTMENT
----------
         1 Ankita Jadhav        Female     1234569871 Gansoli

         2 Pooja Patil          Female     1234569881 Thane

         3 Pratiksha Maurya     Female     1234569891 Thane


    STUDID STUDENT_NAME         GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
DEPARTMENT
----------
         4 Mohini Chavan        Female     1534569891 Nashik

         5 Rutuja Patil         Female     2534569891 Nashik

         6 Amruta Patil         Female     2535469891 Nashik


6 rows selected.

SQL> select * from stud
  2  where Student_Name like 'P%';

    STUDID STUDENT_NAME         GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
DEPARTMENT
----------
         2 Pooja Patil          Female     1234569881 Thane

         3 Pratiksha Maurya     Female     1234569891 Thane


SQL>
```

5. Not Like:

Syntax:

SELECT * FROM Customers

WHERE CustomerName NOT LIKE 'a%';

```
SQL> select * from stud
  2  where Student_Name not like 'A%';

    STUDID STUDENT_NAME          GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
DEPARTMENT
----------
         2 Pooja Patil          Female     1234569881 Thane

         3 Pratiksha Maurya     Female     1234569891 Thane

         4 Mohini Chavan        Female     1534569891 Nashik


    STUDID STUDENT_NAME          GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
DEPARTMENT
----------
         5 Rutuja Patil         Female     2534569891 Nashik


SQL>
```

6. Between:

SELECT column_name(s)

FROM table_name

WHERE column_name BETWEEN value1 AND value2;

```
         2 Pooja Patil          Female     1234569881 Thane

         3 Pratiksha Maurya     Female     1234569891 Thane


    STUDID STUDENT_NAME          GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
DEPARTMENT
----------
         4 Mohini Chavan        Female     1534569891 Nashik

         5 Rutuja Patil         Female     2534569891 Nashik

         6 Amruta Patil         Female     2535469891 Nashik


6 rows selected.

SQL> select * from stud
  2  where StudId between 1 and 4;

    STUDID STUDENT_NAME          GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
DEPARTMENT
----------
         1 Ankita Jadhav        Female     1234569871 Gansoli

         2 Pooja Patil          Female     1234569881 Thane

         3 Pratiksha Maurya     Female     1234569891 Thane


    STUDID STUDENT_NAME          GENDER     MOBILE_NUM CITY
---------- -------------------- ---------- ---------- --------------------
DEPARTMENT
----------
         4 Mohini Chavan        Female     1534569891 Nashik
```

7. **All :**

Syntax: SELECT column_name(s)
FROM table_name
WHERE column_name comparison_operator ALL
(SELECT column_name
FROM table_name
WHERE condition(s));

```
SQL> select product_name
  2  from products
  3  where product_id = all (select product_id
  4  from orderdetail
  5  where quantity = 6 or quantity = 4);

PRODUCT_NAME
--------------------
Dal
Rice
Wheat
Sugar
Tea
```

8. **Any:**

Syntax: SELECT column_name(s)
FROM table_name
WHERE column_name comparison_operator ANY
(SELECT column_name
FROM table_name
WHERE condition(s));

```
SQL> select distinct product_name
  2  from products
  3  where product_id = any (select product_id from orderdetail);

PRODUCT_NAME
--------------------
Wheat
Sugar
Dal
Rice
Tea
```

9. Exists:

Syntax: SELECT column_name(s)
FROM table_name
WHERE EXISTS
(SELECT column_name(s)
FROM table_name
WHERE condition);

```
SQL> select Student_name, City
  2  from stud
  3  where exists (select * from orderdetail
  4  where stud.StudId = orderdetail.orderdetailid);

STUDENT_NAME         CITY
-------------------- --------------------
Ankita Jadhav        Gansoli
Pooja Patil          Thane
Pratiksha Maurya     Thane
Mohini Chavan        Nashik
Rutuja Patil         Nashik
Amruta Patil         Nashik
```

10. Some:

Syntax: SELECT column_name(s)
FROM table_name
WHERE expression comparison_operator SOME (subquery)

```
SQL> select emp_name
  2  from Employees2
  3  where salary> some (select salary from Employees2
  4  where dept='IT');

EMP_NAME
--------------------
Tanu
sonam
Priyanka
Ram
sonam
Amit
sunil
```

SQL COMPARISION OPERATORS:

1. **Equal to:**
   Syntax: select * from table_name
   Where column_name [comparision operator] <expression>;

```
SQL> select * from product
  2  where price = 69;

PRODUCT_NAME             PRICE
-------------------- ----------
wheat                        69
```

2. **Greater than (>):**
   Syntax: select * from table_name
   Where column_name [comparision operator] <expression>;

```
SQL> select * from product
  2  where price > 80;

PRODUCT_NAME               PRICE
-------------------- ----------
oil                          250
Jam                          150
rice                          82
```

3. **Less than (<):**
   Syntax: select * from table_name
   Where column_name [comparison operator] <expression>;

```
SQL> select * from product
  2  where price < 90;

PRODUCT_NAME               PRICE
-------------------- ----------
rice                          82
wheat                         69
dal                           33
```

4. **Greater than or equal to (>=):**
   Syntax: select * from table_name
   Where column_name [comparison operator] <expression>;

```
SQL> select * from products
  2  where price >= 40;

PRODUCT_NAME             PRICE
-------------------- ----------
Dal                          52
Rice                         72
Wheat                        82
Sugar                        82
Tea                          45
```

5. **Less than or equal to (<=):**
   Syntax: select * from table_name
   Where column_name [comparison operator] <expression>;

```
SQL> select * from products
  2  where price <= 40;

PRODUCT_NAME               PRICE
-------------------- ----------
Salt                         20
```

6. **Not equal to (<>):**
   Syntax: select * from table_name
   Where column_name [comparison operator] <expression>;

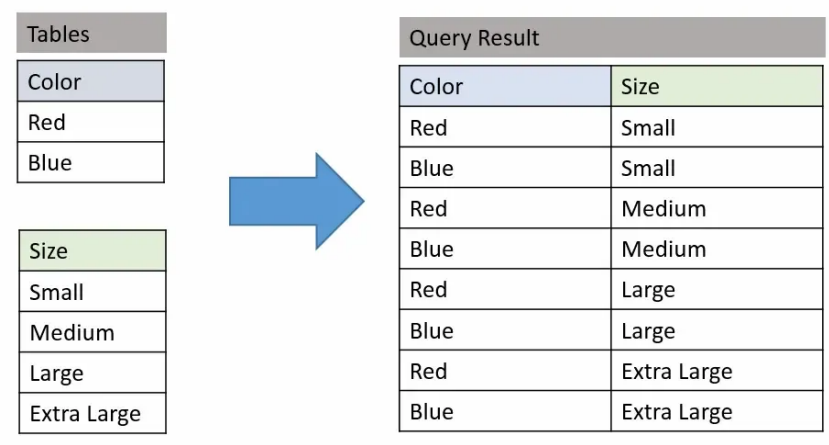
```
SQL> select * from products
  2  where price <> 82;

PRODUCT_NAME               PRICE
-------------------- ----------
Dal                          52
Rice                         72
Tea                          45
Salt                         20
```

SQL JOINS

1. **Cross join:**

   Syntax: SELECT TableName1.columnName1, TableName2.columnName2 FROM TableName1 CROSS JOIN TableName2 ON TableName1.ColumnName = TableName2.ColumnName;



Tables

Color
Red
Blue

Size
Small
Medium
Large
Extra Large

Query Result

Color	Size
Red	Small
Blue	Small
Red	Medium
Blue	Medium
Red	Large
Blue	Large
Red	Extra Large
Blue	Extra Large

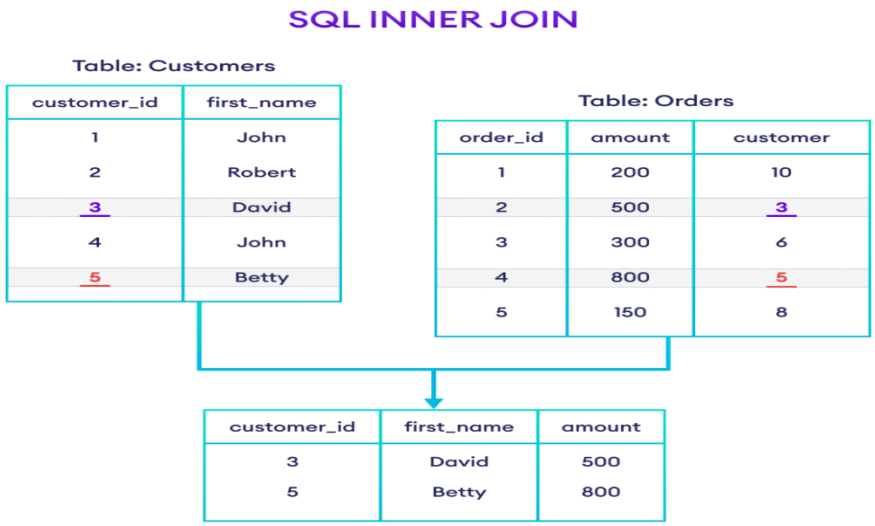
2. **Inner join:**

   Syntax: SELECT column_name(s)

   FROM table1

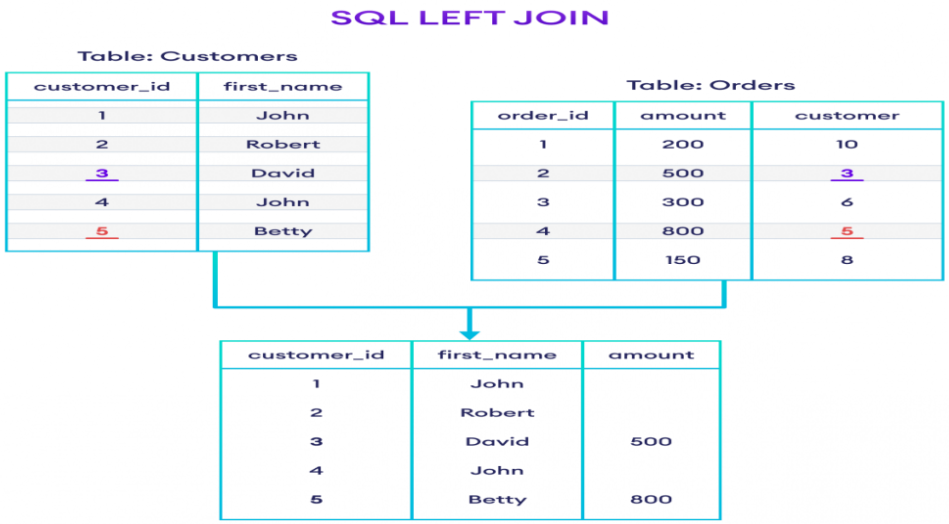
   INNER JOIN table2

   ON table1.column_name = table2.column_name;



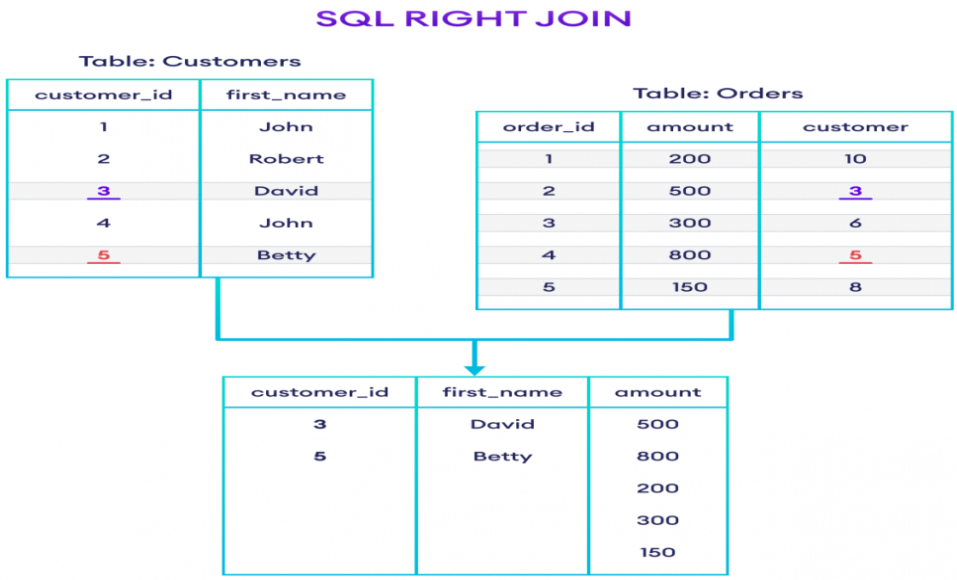
3. **Outer left join:**

   Syntax: SELECT table1.column1, table2.column2....
   FROM table1
   LEFTJOIN table2
   ON table1.column_field = table2.column_field;



4. **Outer right join:**

   Syntax: SELECT table1.column1, table2.column2.....
   FROM table1
   RIGHT JOIN table2
   ON table1.column_field = table2.column_field;



5. **Outer full join:**

   Syntax: SELECT *
   FROM table1
   FULL OUTER JOIN table2
   ON table1.column_name = table2.column_name;

SQL CONSTRAINTS:

1. Not Null :
   Syntax:
   ALTER TABLE Persons
   MODIFY COLUMN Age int NOT NULL;

```
SQL> alter table Employees2 modify emp_name varchar2(20) constraint Employees2_empname_NN not null;

Table altered.

SQL> desc Employess2
ERROR:
ORA-04043: object Employess2 does not exist


SQL> desc Employees2
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMP_ID                                             NUMBER(38)
 EMP_NAME                                  NOT NULL VARCHAR2(20)
 DEPT                                               VARCHAR2(10)
 SALARY                                             NUMBER(38)
 CITY                                               VARCHAR2(20)
 PINCODE                                            NUMBER(38)
```

2. **Unique constraints:**
   Syntax:
   ALTER TABLE table_name
   ADD CONSTRAINT unique_constraint_name UNIQUE(column_name1, column_nam2);

```
SQL> desc Employees2
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMP_ID                                             NUMBER(38)
 EMP_NAME                                  NOT NULL VARCHAR2(20)
 DEPT                                               VARCHAR2(10)
 SALARY                                             NUMBER(38)
 CITY                                               VARCHAR2(20)
 PINCODE                                            NUMBER(38)
 GENDER                                             CHAR(1)

SQL> alter table Employees2 add constraint Employees2_EMP_ID_un unique(emp_id);

Table altered.
```

3. **Check constraints:**
   Syntax: alter table table_name

Add constraint column_name_check check (column_name in (condition));

```
    EMP_ID EMP_NAME             DEPT           SALARY CITY
---------- -------------------- ---------- ---------- --------------------
   PINCODE G
---------- -
       104 sunil                IT              10000 Kolkata
    800004 M

       105 sunil                IT              15000 Kolkata
    800005 M

       106 sonam                HR             115000 Ranchi
    800006 F


    EMP_ID EMP_NAME             DEPT           SALARY CITY
---------- -------------------- ---------- ---------- --------------------
   PINCODE G
---------- -
       107 sonam                HR              48000 Mumbai
    800007 F

       108 Priyanka             IT              70000 Banglore
    800008 F


8 rows selected.

SQL> alter table Employees2
  2  add constraint gender_check check(Gender in('M', 'F'));

Table altered.
```

4. **Primary Key:**

Syntax:

ALTER TABLE table_name

ADD CONSTRAINT constraint_name PRIMARY KEY (column1, column2, ... column_n);

```
SQL> alter table Employees2 add constraint Employees2_pk primary key(Email_Id);

Table altered.

SQL> desc Employees2
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMP_ID                                             NUMBER(38)
 EMP_NAME                                  NOT NULL VARCHAR2(20)
 DEPT                                               VARCHAR2(10)
 SALARY                                             NUMBER(38)
 CITY                                               VARCHAR2(20)
 PINCODE                                            NUMBER(38)
 GENDER                                             CHAR(1)
 EMAIL_ID                                  NOT NULL VARCHAR2(50)
```

5. **Foreign key :**

Syntax: ALTER TABLE table_name

ADD CONSTRAINT constraint_name

FOREIGN KEY (column1, column2, ... column_n)

REFERENCES parent_table (column1, column2, ... column_n);

```
SQL> alter table department
  2  add foreign key (emp_id) references Employees2(emp_id);

Table altered.

SQL> desc Employees2
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMP_ID                                             NUMBER(38)
 EMP_NAME                                  NOT NULL VARCHAR2(20)
 DEPT                                               VARCHAR2(10)
 SALARY                                             NUMBER(38)
 CITY                                               VARCHAR2(20)
 PINCODE                                            NUMBER(38)
 GENDER                                             CHAR(1)
 EMAIL_ID                                  NOT NULL VARCHAR2(50)

SQL> desc department
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMP_ID                                             NUMBER
 MOBILE_NO                                          NUMBER
```