**1. Primary Key Constraint**

* **Purpose**: Uniquely identifies each record in a table. A primary key column must have unique values and cannot contain NULL values.
* **Syntax**:

sql

Copy code

CREATE TABLE table\_name (

column\_name data\_type PRIMARY KEY

);

* **Example**:

sql

Copy code

CREATE TABLE employees (

employee\_id INT NOT NULL,

name VARCHAR(50),

PRIMARY KEY (employee\_id)

);

**2. Foreign Key Constraint**

* **Purpose**: Ensures that a value in one table matches a value in another table. It enforces referential integrity by linking columns across tables.
* **Syntax**:

sql

Copy code

CREATE TABLE table\_name (

column\_name data\_type,

FOREIGN KEY (column\_name) REFERENCES other\_table (other\_column)

);

* **Example**:

sql

Copy code

CREATE TABLE orders (

order\_id INT NOT NULL,

customer\_id INT,

FOREIGN KEY (customer\_id) REFERENCES customers (customer\_id)

);

**3. Unique Constraint**

* **Purpose**: Ensures that all values in a column (or group of columns) are unique, meaning no two rows can have the same values in the specified column(s).
* **Syntax**:

sql

Copy code

CREATE TABLE table\_name (

column\_name data\_type,

UNIQUE (column\_name)

);

* **Example**:

sql

Copy code

CREATE TABLE users (

user\_id INT NOT NULL,

email VARCHAR(100),

UNIQUE (email)

);

**4. Not Null Constraint**

* **Purpose**: Ensures that a column cannot have a NULL value. Every row in the table must contain a value for this column.
* **Syntax**:

sql

Copy code

CREATE TABLE table\_name (

column\_name data\_type NOT NULL

);

* **Example**:

sql

Copy code

CREATE TABLE products (

product\_id INT NOT NULL,

name VARCHAR(50) NOT NULL

);

**5. Check Constraint**

* **Purpose**: Ensures that all values in a column satisfy a specific condition. It's used to enforce business rules at the database level.
* **Syntax**:

sql

Copy code

CREATE TABLE table\_name (

column\_name data\_type,

CHECK (condition)

);

* **Example**:

sql

Copy code

CREATE TABLE employees (

employee\_id INT,

age INT,

CHECK (age >= 18)

);

**Note**: In MySQL, CHECK constraints were not enforced prior to MySQL 8.0.16. They were included for compatibility with other databases, but support for enforcement was added in MySQL 8.0.16.

**6. Default Constraint**

* **Purpose**: Provides a default value for a column when no value is specified during the insertion of a row.
* **Syntax**:

sql

Copy code

CREATE TABLE table\_name (

column\_name data\_type DEFAULT default\_value

);

* **Example**:

sql

Copy code

CREATE TABLE employees (

employee\_id INT NOT NULL,

department VARCHAR(50) DEFAULT 'General'

);

**7. Index Constraint**

* **Purpose**: While not strictly a constraint, an index can be created to improve the performance of queries. An index on a column ensures that search operations on the column are faster.
* **Syntax**:

sql

Copy code

CREATE INDEX index\_name ON table\_name (column\_name);

* **Example**:

sql

Copy code

CREATE INDEX idx\_employee\_name ON employees (name);

**8. Auto Increment Constraint**

* **Purpose**: Automatically generates a unique value for a column, typically used for primary key fields.
* **Syntax**:

sql

Copy code

CREATE TABLE table\_name (

column\_name INT AUTO\_INCREMENT

);

* **Example**:

sql

Copy code

CREATE TABLE students (

student\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(50)

);

**9. Composite Key (Multiple Columns Primary Key)**

* **Purpose**: A combination of multiple columns that together uniquely identify a row in a table.
* **Syntax**:

sql

Copy code

CREATE TABLE table\_name (

column1 data\_type,

column2 data\_type,

PRIMARY KEY (column1, column2)

);

* **Example**:

sql

Copy code

CREATE TABLE course\_enrollments (

student\_id INT,

course\_id INT,

PRIMARY KEY (student\_id, course\_id)

);

**10. Drop a Constraint**

* **Purpose**: To remove an existing constraint.
* **Syntax for dropping constraints**:

sql

Copy code

ALTER TABLE table\_name DROP CONSTRAINT constraint\_name;

* **Example** (for dropping a foreign key constraint):

sql

Copy code

ALTER TABLE orders DROP FOREIGN KEY fk\_customer\_id;

**11. Enable and Disable Constraints**

* **Purpose**: Sometimes constraints might be disabled for bulk operations and then re-enabled.
* **Syntax for disabling and enabling constraints**:

sql

Copy code

SET foreign\_key\_checks = 0; -- Disable foreign key checks

SET foreign\_key\_checks = 1; -- Enable foreign key checks

**Constraints in Action**

* **Primary Key and Foreign Key**: Use these constraints together to model relationships between entities.
* **Unique and Check**: Use CHECK to enforce more complex rules while UNIQUE helps ensure there are no duplicate values in a column.
* **Default and Not Null**: Use DEFAULT to ensure that a column has a value even when no explicit value is provided, and NOT NULL ensures that a value is provided.

**Example of a Table with Multiple Constraints:**

sql

Copy code

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(50) NOT NULL,

email VARCHAR(100) UNIQUE,

department VARCHAR(50) DEFAULT 'General',

age INT CHECK (age >= 18),

manager\_id INT,

FOREIGN KEY (manager\_id) REFERENCES employees(employee\_id)

);

This example creates a table with the following constraints:

* employee\_id is the primary key.
* name cannot be NULL.
* email must be unique.
* department defaults to 'General' if no value is provided.
* age must be greater than or equal to 18.
* manager\_id references another employee\_id in the same table (self-referencing foreign key).

select a.rollno,a.empname,a.salary,b.age,b.phoneno,b.place

from office a

left join empdetails as b on a.rollno=b.rollno

union all

select a.rollno,a.empname,a.salary,b.age,b.phoneno,b.place

from office a

right join empdetails as b on a.rollno=b.rollno

inner join salary as c on a.sid=c.salaryid

select a.rollno,a.empname,a.salary,b.age,b.phoneno,b.place

from office a,empdetails as b where a.rollno=b.rollno

create table salary(sid int primary key ,srange int)

create table office(regid int primary key ,ename

varchar(20),salaryid int, FOREIGN KEY (salaryid)

REFERENCES salary(sid))

create table officedetails (sno int primary key,regid

int ,ename varchar(20), age int,dob date,

place varchar(30) not null ,FOREIGN KEY (regid) REFERENCES office(regid))