SQL Constraints

We can use constraint to define **integrity constant**. **Integrity constraint** is a rule that restricts the values in the database. This ensures the accuracy and reliability of the data in the table. If there is any violation between the constraint and the data action, the action is aborted.

Oracle Database lets you create six types of constraints and lets you declare them in two ways.

1. **NOT NULL** Constraint
   It prohibits a database value from being null.

2. **UNIQUE** Constraint
   It prohibits multiple rows from having the same value in the same column or combination of columns but allows some value to be null.

3. **PRIMARY KEY** Constraint
   It combines a **NOT NULL** constraint and a UNIQUE constraint in a single declaration. That is, it prohibits multiple rows from having the same value in the same column or combination of columns and prohibits values from being null.

4. **FOREIGN KEY** Constraint
   It requires values in one table to match values in another table.

5. **CHECK** Constraint
   It requires a value in the database to comply with a specified condition.

6. **REF** Constraint
   It lets you further describe the relationship between the REF Column(reference an object in another object type or in a relational table) and the object it references.

Constraint clauses can appear in following statements:

1. CREATE TABLE
2. ALTER TABLE
3. CREATE VIEW
4. ALTER VIEW

Examples

1. **NOT NULL** Constraint

   Here, e_id and e_name column has a **not null** constraint.

```
CREATE TABLE EmployeeInfo(
    e_id int NOT NULL,
    e_name char(30) NOT NULL
);
```

2. **UNIQUE** Constraint

```
CREATE TABLE EmployeeInfo(
   e_id int,
   e_name varchar(20) CONSTRAINT  e_name_u UNIQUE
);
```

3. **PRIMARY KEY** Constraint

```
CREATE TABLE EmployeeInfo(
   e_id int CONSTRAINT  e_id_pk PRIMARY KEY,
   e_name char(30),
);
```

4. **FOREIGN KEY** Constraint

   First, create a department.

```
CREATE TABLE Department(
   d_id int PRIMARY KEY,
   d_name char(20) NOT NULL,
   location char(20),

);
```

Add a **FOREIGN KEY** constraint.

```
CREATE TABLE EmployeeInfo(
   E_id int PRIMARY KEY,
```

```
   e_name char(20) NUT NULL,
   salary int,
   d_id int REFERENCES Department(d_id);
);
```

5. CHECK Constraint

```
CREATE TABLE Divisions(
   div_no int CONSTRAINT CHECK_divno CHECK
    (div_no BETWEEN 10 AND 99) DISABLE,
);
```

Here, check_divno ensures that no division numbers are less than 10 and greater than 99.

Conclusion

Constraints can be specified when the table is created with the CREATE TABLE statement, or after the table is created with the ALTER TABLE statement.