**AIM**: To implement Data Constraints.

#### **THEORY**

Constraints are the business Rules which are enforced on the data being stored in a table are called Constraints

#### **TYPES OF CONSTRAINTS:**

- 1) Primary key
- 2) Foreign key/references
- 3) Check
- 4) Unique
- 5) Not null
- 6) Default

#### **PROCEDURE**

#### (a) PRIMARY KEY

Primary Key: A column or a set of columns that uniquely identifies each row in a table.

The PRIMARY KEY defined at column level Syntax:

CREATE TABLE tablename (Columnnamel DATATYPE

PRIMARY KEY, Columnname2 DATATYPE, columnname3

DATATYPE);

**EXAMPLE:** 

Create table Employee (empno number (4) primary key, ename varchar2 (10), job varchar2 (6), sal number (5), deptno number (7));

The PRIMARY KEY defined at table level Syntax:

**CREATE TABLE tablename** 

(Columnnamel DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, PRIMARY KEY (columnnamel, columnname2));

#### **EXAMPLE:**

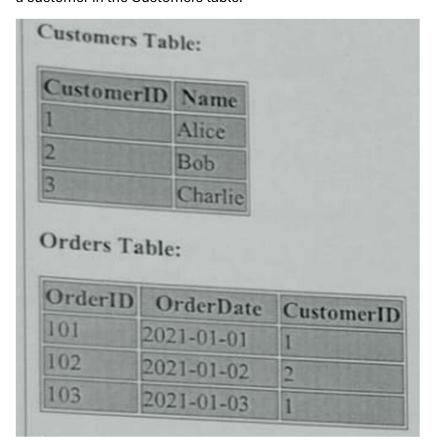
Create table Employee (empno number (4), ename varchar2 (10), job varchar2 (6), sal number (5), deptno number (7), PRIMARY KEY (empno, ename))

#### (b) FOREIGN KEY

Foreign Key: A column or a set of columns in one table that refers to the primary key columns in another table.

#### Example

Consider two tables: Orders and Customers. Each order in the Orders table is placed by a customer in the Customers table.



Here, CustomerID in the Orders table is a foreign key that references CustomerID in the Customers table.

Syntax:- Creating Tables with Foreign Keys:

**CREATE TABLE Customers (** 

CustomerID Number (5) PRIMARY KEY,

Name VARCHAR2 (100)
);

CREATE TABLE Orders (

OrderID NUMBER (10) PRIMARY KEY,

OrderDate DATE,

CustomerID NUMBER (5).

FOREIGN KEY (CustomerID) REFERENCES Customers (CustomerID)
);

#### (c) CHECK CONSTRAINT

A rule that specifies a condition all values in a column or a combination of columns must satisfy.

The CHECK Constraint defined at column level

Syntax:-

CREATE TABLE tablename (Columnnamel DATATYPE CHECK (logical expression), columnname2 DATATYPE, columnname3 DATATYPE);

**EXAMPLE** 

CREATE TABLE Employee(empno number(3), ename varchar2(20), design varchar2(15), sal number(5) CHECK(sal>500 and sal<10001), deptno number(2));

The CHECK Constraint defined at table level Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, CHECK (logical expression1), CHECK (logical expression2));

**EXAMPLE** 

CREATE TABLE Employee(empno number(3), ename varchar2(20), design varchar2(15), sal number(5), deptno number(2), CHECK(sal>500 and sal<1000));

#### (d) UNIQUE CONSTRAINT

A UNIQUE constraint in SQL ensures that all values in a column or a set of columns are distinct from one another.

The UNIQUE Constraint defined at the column level

Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE UNIQUE, columnname2 DATATYPE UNIQUE, columnname3 DATATYPE...);

**EXAMPLE:** 

CREATE TABLE Employee (empno number (3), ename varchar2(20), design varchar2(15) UNIQUE, sal number(5));

The UNIQUE Constraint defined at the table level

Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, UNIQUE (columnname1));

**EXAMPLE** 

Create table Employee (empno number(3), ename varchar2(20), design varchar2(15), sal number(5), UNIQUE(design));

While both PRIMARY KEY and UNIQUE constraints enforce uniqueness, there are some differences:

#### PRIMARY KEY:

- Cannot contain NULL values.
- There can only be one primary key constraint per table.
- Implicitly creates a unique index on the column(s).

#### **UNIQUE:**

- Can contain NULL values (although NULLs are treated as distinct values).
- Multiple unique constraints can be defined per table.
- Explicitly enforces uniqueness but does not imply primary key constraints.

#### (e) Not Null

A rule that ensures that a column cannot have a NULL value. Every row must have a value for the column(s) with this constraint.

Syntax

CREATE TABLE tablename (Columnnamel DATATYPE NOT NULL, columnname2 DATATYPE NOT NULL, columnname3 DATATYPE);

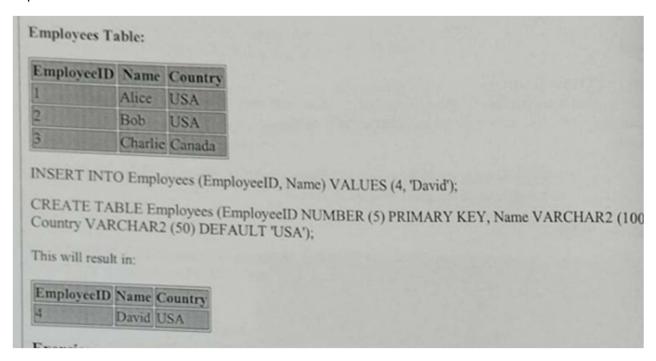
#### **EXAMPLE**

CREATE TABLE Employee (empno number (4), ename varchar2 (20) NOT NULL, design varchar2 (20), sal number(3));

#### (g) DEFAULT

A DEFAULT constraint in SQL is used to provide a default value for a column when no value is specified during the insertion of a new row. This helps ensure that the column has a valid value and can simplify the data entry process by automatically filling in common or standard values.

Consider a table Employees where the Country field should default to 'USA' if no value is provided.



1.Create a table customer Having field customer\_id ,name, address,city, pincode where customers\_id is the primary key and name should not be left blank

```
CREATE TABLE customer (
customer_id NUMBER PRIMARY KEY,
name VARCHAR2(50) NOT NULL,
address VARCHAR2(100),
city VARCHAR2(50),
pincode NUMBER(6)
);
```

2. Create a table Student having fields serial no ,roll no ,name,registration no , cgpa where roll no is primaryKey and name should not be left blank and cgpa should be greater than equal to 6.0 CREATE TABLE student (

```
serial_no NUMBER,

roll_no NUMBER PRIMARY KEY,

name VARCHAR2(50) NOT NULL,

registration_no NUMBER UNIQUE,

cgpa NUMBER(3,2) CHECK (cgpa >= 6.0)
);
```

AIM: To execute the Data Manipulation Language (DML) commands in RDBMS.

#### **OBJECTIVES**

To understand Data Manipulation Language (DML) commands

#### **THEORY**

DML commands are the most frequently used SQL commands and are used to query and manipulate the existing database objects. Some of the commands are

#### 1. INSERT

This is used to add one or more rows to a table. The values are separated by commas and the data types char and date are enclosed in apostrophes. The values must be entered in the same order as they are defined.

#### 2. SELECT

It is used to retrieve information from the table. It is generally referred to as querying the table. We can either display all columns in a table or only specify columns from the table

#### 3. UPDATE

It is wed to alter the column values in a table. A single column may be updated or more than one column could be updated

#### 4. DELETE

After inserting row in a table we can also delete them if required. The delete command consists of a from clause followed by an optional where clause

#### **PROCEDURE**

#### • INSERT COMMAND

#### a. Inserting a single row into table:

Syntax:

insert into , <expression2>)

Example:

SQL>INSERT INTO EMPLOYEE VALUES(101, 'MANU', 'LECTURER', 15000);

#### b. <u>Inserting more than one record using a single insert commands:</u>

```
Syntax:
INSERT ALL
 INTO table_name1 (column1, column2, ...) VALUES (value1, value2, ...)
 INTO table_name2 (column1, column2, ...) VALUES (value3, value4, ...)
SELECT * FROM dual;
Example 1: Inserting Multiple Rows into a Single Table
Suppose you have a students table and you want to insert multiple rows into it:
INSERT ALL
 INTO students (student_id, student_name, student_age) VALUES (1, 'Joy', 9)
 INTO students (student_id, student_name, student_age) VALUES (2, 'Smiley', 13)
 INTO students (student_id, student_name, student_age) VALUES (3, 'Happy', 11)
SELECT * FROM dual;
This statement inserts three rows into the students table in a single operation.
Example 2: Inserting Rows into Multiple Tables
You can also use INSERT ALL to insert data into multiple tables simultaneously. For
example:
INSERT ALL
 INTO students (student_id, student_name, student_age) VALUES (1, 'Joy', 9)
 INTO teachers (teacher_id, teacher_name, teacher_age) VALUES (101, 'Mr. Smith', 45)
SELECT * FROM dual;
```

### c. Skipping the fields while inserting:

Insert into <tablename> (<column namel="">, <column name3="">)&gt;values</column></column></tablename>
( <expression 1="">,<expression3>);</expression3></expression>
Other way is to give null while passing the values.
SELECT COMMAND
(a) View all rows and all columns
Syntax:
Select from tablename:
Example
Select from Employee,
(b)Selected Columns and All Rows
Syntax:
Select <column1>, <column2> from tablename,</column2></column1>
Example:
Select empno, empname from Employee;
(c)Selected Columns and selected Rows
Syntax
SELECT <column1>, <column2> FROM <tablename> WHERE <condition>;</condition></tablename></column2></column1>
Example:
Select empno, empname from Employee where
designation 'lecturer';
(d) Eliminating duplicate rows
Syntax:

SELECT DISTINCT <column1>, <column2>

FROM tablename>
Example:
Select distinct empname from Employee;
• UPDATE COMMAND
(a) Updating all rows
Syntax
Update tablename set columnnamel expession)>, <columnname2 exprsson2<="" td=""></columnname2>
Example:
Update Employee set Designation lecturer",
(b) Updating records conditionally
Syntax:
Update tablename set field values where condition;
Example:
Update Employee set sal 10000 where empno-135,
DELETE COMMAND
(a) Removal of all rows
Syntax:
Delete from table name,
Example
Delete from emp,
(b) Removal of specific rows
Syntax:
Delete from table name where <condition>,</condition>
Example:
Delete from emp where empno 135;

#### **RESULT**

The DML commands are executed successfully.

1. <u>Insert 10 records to the Students table of question No 2 of Sql constraints and write a query to show all the records of a student table</u>

Query to insert 10 record :-

#### **INSERT ALL**

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (1, 101, 'Amit Kumar', 202301, 7.8)

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (2, 102, 'Priya Sharma', 202302, 8.2)

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (3, 103, 'Rohit Verma', 202303, 7.0)

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (4, 104, 'Neha Gupta', 202304, 9.1)

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (5, 105, 'Suresh Reddy', 202305, 6.5)

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (6, 106, 'Anjali Singh', 202306, 8.7)

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (7, 107, 'Vikram Das', 202307, 7.3)

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (8, 108, 'Meera Joshi', 202308, 9.0)

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (9, 109, 'Raj Malhotra', 202309, 6.8)

INTO student (serial\_no, roll\_no, name, registration\_no, cgpa) VALUES (10, 110, 'Simran Kaur', 202310, 7.5)

SELECT 1 FROM DUAL;

Query to show all the records of a student table:

Select \* from student

#### **OUTPUT:-**

SERIAL_NO	ROLL_NO	NAME	REGISTRATION_NO	CGPA
1	101	Amit Kumar	202301	7.8
2	102	Priya Sharma	202302	8.2
3	103	Rohit Verma	202303	7.0
4	104	Neha Gupta	202304	9.1
5	105	Suresh Reddy	202305	6.5
6	106	Anjali Singh	202306	8.7
7	107	Vikram Das	202307	7.3
8	108	Meera Joshi	202308	9.0
9	109	Raj Malhotra	202309	6.8
10	110	Simran Kaur	202310	7.5