| Ex no: 9 | Create and Manage Views and Integrity Constraints |
|---|---|
| Date: | |

**Aim:**

To create and manage views and integrity constraints in a relational database.

**Procedure:**

**1. Use the CREATE VIEW command to create views.**
**2. Use the SELECT statement to query data from views.**
**3. Use the DROP VIEW command to delete views if needed.**
**4. Define integrity constraints such as PRIMARY KEY, FOREIGN KEY, UNIQUE, and CHECK.**
**5. Add constraints to existing tables using the ALTER TABLE command.**
**6. Verify the creation of views and constraints by querying the database metadata.**
**7. Document the SQL queries used for creating and managing views and constraints.**
**8. Execute the queries in the SQL environment.**
**9. Validate the results by checking the structure and data of the created views.**
**10. Ensure data integrity by testing different scenarios.**

**Queries:**

**1. Use the CREATE VIEW command to create views.**

**Query**;

```
-- Create Employees table
CREATE TABLE employees (
   employee_id INT PRIMARY KEY,
   name VARCHAR(50),
   department VARCHAR(50),
   role VARCHAR(50),
   salary DECIMAL(10, 2)
);
-- Insert data into Employees table
INSERT INTO employees (employee_id, name, department, role, salary) VALUES
(1, 'Alice', 'HR', 'Manager', 60000),
(2, 'Bob', 'HR', 'Assistant', 40000),
(3, 'Charlie', 'Sales', 'Manager', 75000),
(4, 'David', 'Sales', 'Representative', 45000),
(5, 'Eve', 'IT', 'Developer', 70000);


-- Create VIEW Employees-VIEW
CREATE VIEW high_salary_employees AS
SELECT employee_id, name, department, salary
FROM  employees
WHERE salary > 50000;
```

**Output:**

**VIEW CREATED**

**2. Use the SELECT statement to query data from views.**

**Query:**

```
SELECT * FROM high_salary_employees;
```

**Output:**

| employee_id | name | department | salary |
|---|---|---|---|
| 1 | Alice | HR | 60000.00 |
| 3 | Charlie | Sales | 75000.00 |
| 5 | Eve | IT | 70000.00 |

**3. Use the DROP VIEW command to delete views if needed.**

**Query:**

```
DROP VIEW IF EXISTS high_salary_employees;
```

**4. Define integrity constraints such as PRIMARY KEY, FOREIGN KEY, UNIQUE, and CHECK.**

**Query:**

```
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   FirstName VARCHAR(50) NOT NULL,
   LastName VARCHAR(50) NOT NULL,
   DepartmentID INT,
   UNIQUE (FirstName, LastName),
   CHECK (DepartmentID > 0)
);
```

**To add foreign key**

```
CREATE TABLE Departments (
   DepartmentID INT PRIMARY KEY,
   DepartmentName VARCHAR(50) NOT NULL
);

ALTER TABLE Employees
ADD CONSTRAINT FK_Department
FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID);
```

**5. Add constraints to existing tables using the ALTER TABLE command.**

**Query:**

```
ALTER TABLE Employees
ADD CONSTRAINT UQ_Employee UNIQUE (FirstName, LastName);
```

**6. Test the queries with different group by columns.**

**Query:**

```
-- Check views
SELECT * FROM information_schema.views WHERE table_name = 'Employees';

-- Check constraints
SELECT * FROM information_schema.table_constraints WHERE table_name = 'Employees';
```

| | | |
|---|---|---|
| **Algorithm** | **15** | |
| **Program** | **30** | |
| **Execution** | **30** | |
| **Output & Result** | **15** | |
| **Viva / Record** | **10** | |
| **Total** | **100** | |

**Result:**

Thus, we created SQL queries for querying to implement create and manage views and integrity constraints