

Prashant Bharti

Roll No. 202251102

Indian Institute of Information Technology Vadodara

Database Management Systems Laboratory Assignment

Week 06

-by PRASHANT BHARTI

(202251102)

Quest . Design a MySQL database schema for a university to maintain student records with the following constraints. Create tables and columns as needed and free to add appropriate data to your tables based on your understanding. Your submission should include proper code for creation of necessary tables, constraints, views, etc ensuring the fulfilment of all the specified requirements.

SQL Worksheet Clear Find Actions Save

```
1  -- Create Department Table
2  CREATE TABLE Department (
3      department_id INT PRIMARY KEY,
4      department_name VARCHAR(50) NOT NULL,
5      department_head_id INT,
6      CHECK (department_id > 0)
7  );
8
9  -- Create Student Table
10 CREATE TABLE Student (
11     student_id INT PRIMARY KEY,
12     student_name VARCHAR(100) NOT NULL,
13     GPA DECIMAL(3, 2) CHECK (GPA >= 0 AND GPA <= 4),
14     CHECK (student_id > 0)
15 );
16
17 -- Create Student_Department Junction Table as a Member table
18 CREATE TABLE Student_Department (
19     student_id INT,
20     department_id INT,
21
22     PRIMARY KEY (student_id, department_id),
23     FOREIGN KEY (student_id) REFERENCES Student(student_id) DEFERRABLE INITIALLY DEFERRED ,
24     FOREIGN KEY (department_id) REFERENCES Department(department_id) DEFERRABLE INITIALLY DEFERRED
25 );
```

```

25 -- a. Every student must belong to at least one department.
26 -- Create Assertion for Every Student Must Belong to at Least One Department
27 CREATE ASSERTION student_belongs_to_department
28 CHECK (
29     NOT EXISTS (
30         SELECT 1
31         FROM Student s
32         WHERE NOT EXISTS (
33             SELECT 1
34             FROM Student_Department sd
35             WHERE sd.student_id = s.student_id
36         )
37     )
38 ) DEFERRABLE INITIALLY DEFERRED;
39

40 -- c. Each department must have a department head.
41 -- Create Assertion for Department Head
42 CREATE ASSERTION department_head_exists
43 CHECK (
44     NOT EXISTS (
45         SELECT 1
46         FROM Department d
47         WHERE d.department_head_id IS NULL
48     )
49 ) DEFERRABLE INITIALLY DEFERRED;
--

51 -- d. The maximum number of students in a department cannot exceed 100.
52 -- Create Assertion for Maximum Students in a Department
53 CREATE ASSERTION max_students_per_department
54 CHECK (
55     NOT EXISTS (
56         SELECT 1
57         FROM (
58             SELECT department_id, COUNT(*) as num_students
59             FROM Student_Department
60             GROUP BY department_id
61         ) s_count
62         WHERE s_count.num_students > 100
63     )
64 ) DEFERRABLE INITIALLY DEFERRED;
--

66 -- d.2 The maximum number of students in a department cannot exceed 100.
67 -- We can also create a trigger regarding this situation
68 CREATE OR REPLACE TRIGGER max_students_trigger
69 BEFORE INSERT ON Student_Department
70 FOR EACH ROW
71 DECLARE
72     current_students NUMBER;
73 BEGIN
74     -- Count the current number of students in the department
75     SELECT COUNT(*) INTO current_students
76     FROM Student_Department
77     WHERE department_id = :NEW.department_id;
78
79     -- Check if exceed the limit
80 IF current_students >= 100 THEN
81     RAISE_APPLICATION_ERROR(-20001, 'Maximum number of students in a department exceeded.');
```

Inserting values...

```

88  -- Insert values into Department table
89  INSERT INTO Department (department_id, department_name, department_head_id)
90  VALUES (1, 'Computer Science', 101);
91
92  INSERT INTO Department (department_id, department_name, department_head_id)
93  VALUES (2, 'Mathematics', 102);
94
95  INSERT INTO Department (department_id, department_name, department_head_id)
96  VALUES (3, 'Physics', 103);
97
98  -- Insert values into Student table
99  INSERT INTO Student (student_id, student_name, GPA)
100 VALUES (101, 'John Doe', 3.5);
101
102 INSERT INTO Student (student_id, student_name, GPA)
103 VALUES (102, 'Jane Smith', 3.8);
104
105 INSERT INTO Student (student_id, student_name, GPA)
106 VALUES (103, 'Bob Johnson', 3.2);
107

107
108 -- Insert values into Student_Department junction table
109 INSERT INTO Student_Department (student_id, department_id)
110 VALUES (101, 1);
111
112 INSERT INTO Student_Department (student_id, department_id)
113 VALUES (102, 2);
114
115 INSERT INTO Student_Department (student_id, department_id)
116 VALUES (103, 3);
117
118 INSERT INTO Student_Department (student_id, department_id)
119 VALUES (101, 2);
120
121 INSERT INTO Student_Department (student_id, department_id)
122 VALUES (102, 1);
123
124 INSERT INTO Student_Department (student_id, department_id)
125 VALUES (102, 99);
126 commit;

```

Output Tables

```

128 select * from Department;
129 select * from Student;
130 select * from Student_Department;

```

DEPARTMENT_ID	DEPARTMENT_NAME	DEPARTMENT_HEAD_ID
1	Computer Science	101
2	Mathematics	102
3	Physics	103

STUDENT_ID	STUDENT_NAME	GPA
101	John Doe	3.5
102	Jane Smith	3.8
103	Bob Johnson	3.2

Download CSV

3 rows selected.

STUDENT_ID	DEPARTMENT_ID
101	1
101	2
102	1
102	2
103	3

Download CSV

5 rows selected.

Queries.....

a. Every student must belong to at least one department.

```
-- a. Every student must belong to at least one department.
-- Create Assertion for Every Student Must Belong to at Least One Department
CREATE ASSERTION student_belongs_to_department
CHECK (
    NOT EXISTS (
        SELECT 1
        FROM Student s
        WHERE NOT EXISTS (
            SELECT 1
            FROM Student_Department sd
            WHERE sd.student_id = s.student_id
        )
    )
) DEFERRABLE INITIALLY DEFERRED;
```

b. The GPA of a student must be between 0 and 4.

```
-- Create Student Table
CREATE TABLE Student (
    student_id INT PRIMARY KEY,
    student_name VARCHAR(100) NOT NULL,
    GPA DECIMAL(3, 2) CHECK (GPA >= 0 AND GPA <= 4),
    CHECK (student_id > 0)
);
```

c. Each department must have a department head.

```
40 -- c. Each department must have a department head.
41 -- Create Assertion for Department Head
42 v CREATE ASSERTION department_head_exists
43 CHECK (
44     NOT EXISTS (
45         SELECT 1
46         FROM Department d
47         WHERE d.department_head_id IS NULL
48     )
49 ) DEFERRABLE INITIALLY DEFERRED;
```

d. The maximum number of students in a department cannot exceed 100.

```

-- d.1 The maximum number of students in a department cannot exceed 100.
-- Create Assertion for Maximum Students in a Department
CREATE ASSERTION max_students_per_department
CHECK (
    NOT EXISTS (
        SELECT 1
        FROM (
            SELECT department_id, COUNT(*) as num_students
            FROM Student_Department
            GROUP BY department_id
        ) s_count
        WHERE s_count.num_students > 100
    )
) DEFERRABLE INITIALLY DEFERRED;

-- d.2 The maximum number of students in a department cannot exceed 100.
-- We can also create a trigger regarding this situation
CREATE OR REPLACE TRIGGER max_students_trigger
BEFORE INSERT ON Student_Department
FOR EACH ROW
DECLARE
    current_students NUMBER;
BEGIN
    -- Count the current number of students in the department
    SELECT COUNT(*) INTO current_students
    FROM Student_Department
    WHERE department_id = :NEW.department_id;

    -- Check if exceed the limit
    IF current_students >= 100 THEN
        RAISE_APPLICATION_ERROR(-20001, 'Maximum number of students in a department exceeded.');
```

-----THANK YOU-----

---END---