Prashant Bharti **Roll No. 202251102**

Indian Institute of Information Technology Vadodara

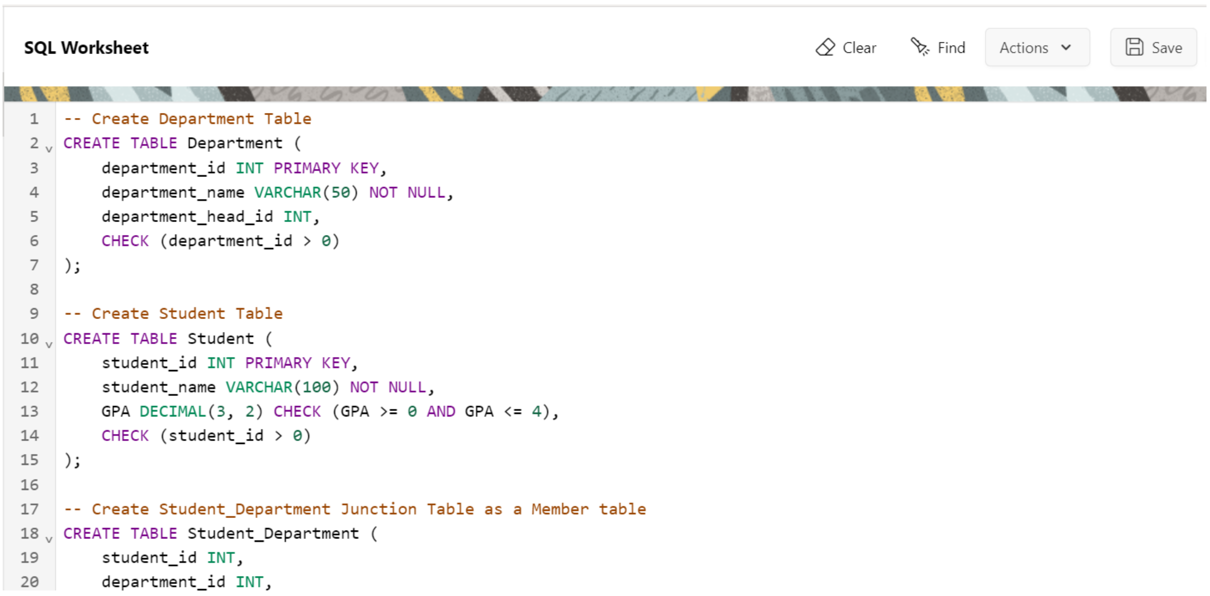
Database Management Systems Laboratory Assignment

Week 06

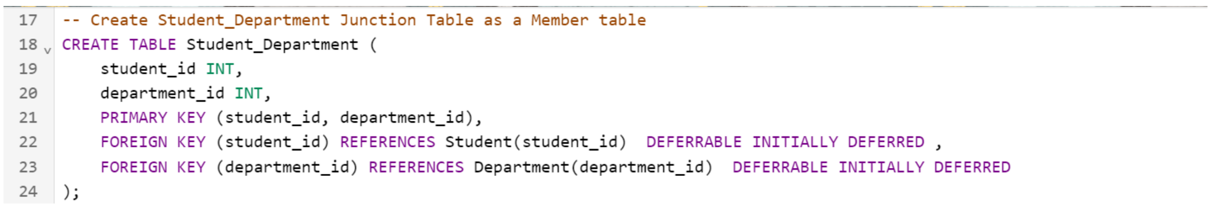
-by **PRASHANT BHARTI**

(202251102)

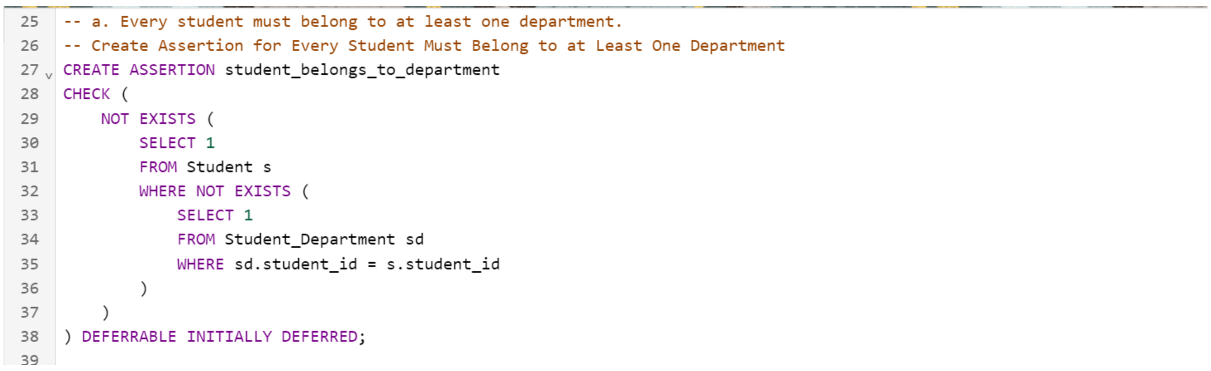
Quest . Design a MySQL database schema for a university to maintain student records with the following constraints. Create tables and columns as needed and free to add appropriate data to your tables based on your understanding. Your submission should include proper code for creation of necessary tables,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



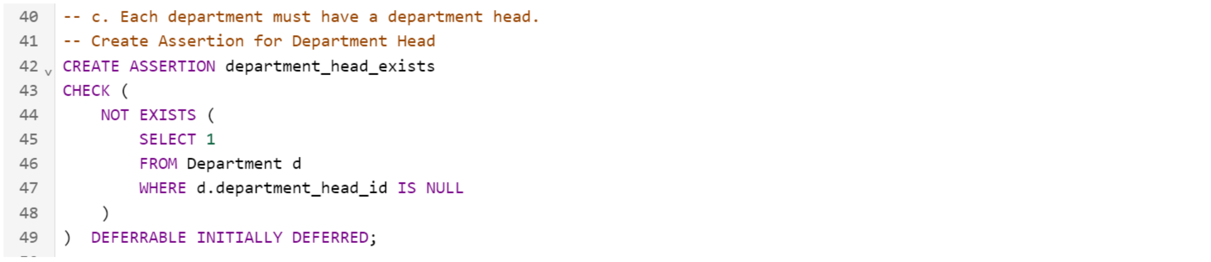
```sql
-- Create Department Table
CREATE TABLE Department (
    department_id INT PRIMARY KEY,
    department_name VARCHAR(50) NOT NULL,
    department_head_id INT,
    CHECK (department_id > 0)
);

-- 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)
);

-- Create Student_Department Junction Table as a Member table
CREATE TABLE Student_Department (
    student_id INT,
    department_id INT,
```



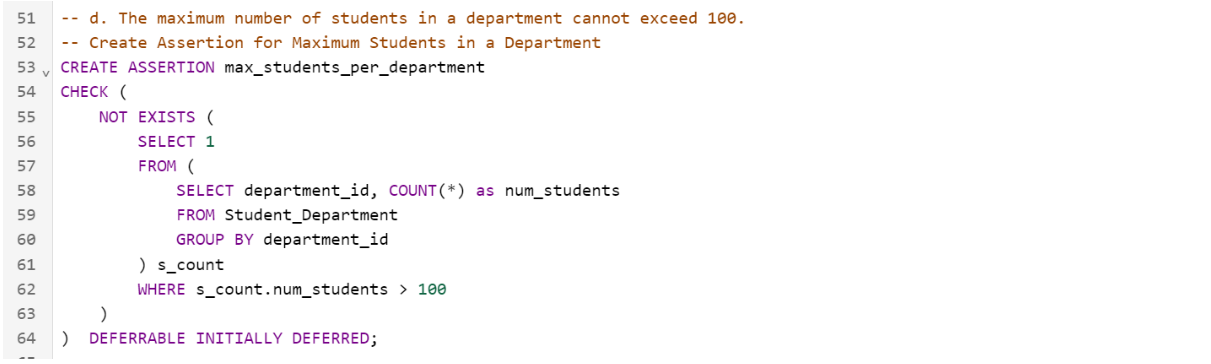
```sql
-- Create Student_Department Junction Table as a Member table
CREATE TABLE Student_Department (
    student_id INT,
    department_id INT,
    PRIMARY KEY (student_id, department_id),
    FOREIGN KEY (student_id) REFERENCES Student(student_id)  DEFERRABLE INITIALLY DEFERRED ,
    FOREIGN KEY (department_id) REFERENCES Department(department_id)  DEFERRABLE INITIALLY DEFERRED
);
```



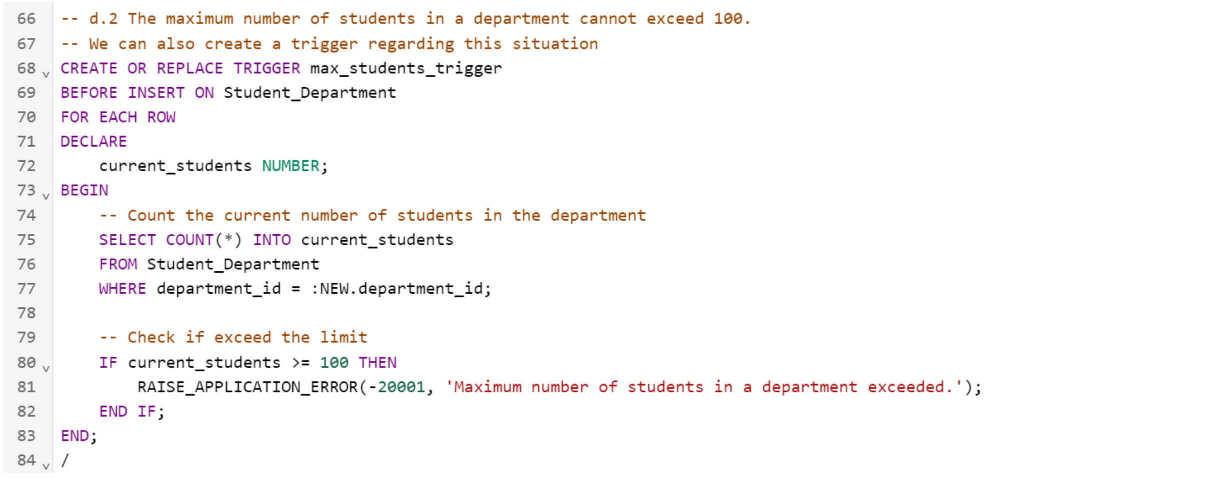
```sql
-- a. Every student must belong to at least one department.
-- Create Assertion for Every Student Must Belong to at Least One Department
CREATE ASSERTION student_belongs_to_department
CHECK (
    NOT EXISTS (
        SELECT 1
        FROM Student s
        WHERE NOT EXISTS (
            SELECT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



```sql
-- c. Each department must have a department head.
-- Create Assertion for Department Head
CREATE ASSERTION department_head_exists
CHECK (
    NOT EXISTS (
        SELECT 1
        FROM Department d
        WHERE d.department_head_id IS NULL
    )
)  DEFERRABLE INITIALLY DEFERRED;
```



```sql
-- d. 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;
```



```sql
-- 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.');
    END IF;
END;
/
```

Inserting values…

```sql
-- Insert values into Department table
INSERT INTO Department (department_id, department_name, department_head_id)
VALUES (1, 'Computer Science', 101);

INSERT INTO Department (department_id, department_name, department_head_id)
VALUES (2, 'Mathematics', 102);

INSERT INTO Department (department_id, department_name, department_head_id)
VALUES (3, 'Physics', 103);

-- Insert values into Student table
INSERT INTO Student (student_id, student_name, GPA)
VALUES (101, 'John Doe', 3.5);

INSERT INTO Student (student_id, student_name, GPA)
VALUES (102, 'Jane Smith', 3.8);

INSERT INTO Student (student_id, student_name, GPA)
VALUES (103, 'Bob Johnson', 3.2);

```

```sql

-- Insert values into Student_Department junction table
INSERT INTO Student_Department (student_id, department_id)
VALUES (101, 1);

INSERT INTO Student_Department (student_id, department_id)
VALUES (102, 2);

INSERT INTO Student_Department (student_id, department_id)
VALUES (103, 3);

INSERT INTO Student_Department (student_id, department_id)
VALUES (101, 2);

INSERT INTO Student_Department (student_id, department_id)
VALUES (102, 1);

INSERT INTO Student_Department (student_id, department_id)
VALUES (102, 99);
commit;
```

Output Tables

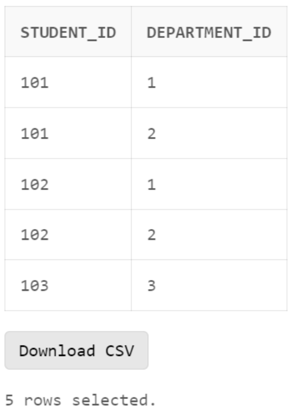
```sql
select * from Department;
select * from Student;
select * from Student_Department;
```

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…..

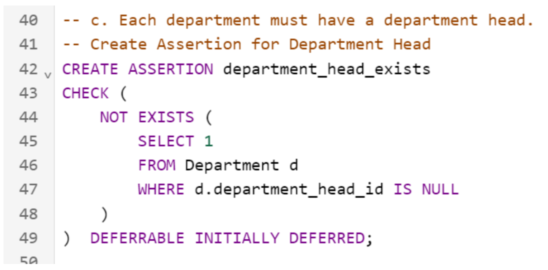
a. Every student must belong to at least one department.

```
-- 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.



```
-- c. Each department must have a department head.
-- Create Assertion for Department Head
CREATE ASSERTION department_head_exists
CHECK (
    NOT EXISTS (
        SELECT 1
        FROM Department d
        WHERE d.department_head_id IS NULL
    )
)  DEFERRABLE INITIALLY DEFERRED;
```

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

```sql
-- 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;
```

```sql
-- 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.');
    END IF;
END;
/
```

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

---END---