Student Report Card System

Introduction

The **Student Report Card System** is a database application developed using MySQL to manage student data, courses, and exam marks efficiently. It allows adding and updating student and course information, recording exam marks, generating report cards through stored procedures, and logging high scores using triggers.

Database & Tables Design

Database Creation:

```
CREATE DATABASE StudentDB;

USE StudentDB;
```

Tables

Students Table

Stores student details such as name, date of birth, gender, and email.

```
CREATE TABLE Students (

StudentID INT PRIMARY KEY AUTO_INCREMENT,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DOB DATE,

Gender VARCHAR(10)

);
```

```
ALTER TABLE Students ADD Email VARCHAR(100);
```

Courses Table

Stores courses with a course name and code.

```
CREATE TABLE Courses (
    CourseID INT PRIMARY KEY AUTO_INCREMENT,
    CourseName VARCHAR(100),
    CourseCode VARCHAR(10)
);
```

Marks Table

Stores the marks obtained by students in various courses, along with exam date.

```
CREATE TABLE Marks (
    MarkID INT PRIMARY KEY AUTO_INCREMENT,
    StudentID INT,
    CourseID INT,
    MarksObtained INT,
    MaxMarks INT DEFAULT 100,
    ExamDate DATE,
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
);
```

ScoreLog Table (for trigger logging)

Stores logs of high scores (≥ 90) when inserted into Marks.

```
CREATE TABLE ScoreLog (
    LogID INT PRIMARY KEY AUTO_INCREMENT,
    StudentID INT,
    CourseID INT,
    MarksObtained INT,
    LoggedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Data Insertion Queries and Sample Data

Insert Students

```
INSERT INTO Students (FirstName, LastName, DOB, Gender, Email)
VALUES
('Ravi', 'Pakala', '2005-06-15', 'male', 'ravi@example.com'),
('Arun', 'Smith', '2004-08-20', 'Male', 'arun@example.com'),
('Sruthi', 'Brown', '2005-01-10', 'FeMale', 'sruthi@example.com');
```

Insert Courses

```
INSERT INTO Courses (CourseName, CourseCode)
VALUES
('Mathematics', 'MATH101'),
('Science', 'SCI101'),
('English', 'ENG101');
```

Insert Marks

```
INSERT INTO Marks (StudentID, CourseID, MarksObtained, ExamDate)
VALUES
(1, 1, 85, '2007-07-25'),
(1, 2, 78, '2010-05-24'),
(1, 3, 92, '2011-02-24'),
(2, 1, 65, '2023-03-12'),
(2, 2, 70, '2005-06-16'),
(2, 3, 80, '2004-08-28'),
(3, 1, 95, '2005-04-13'),
(3, 2, 88, '2025-08-01'),
(3, 3, 91, '2001-05-11');
```

Update Student Email

```
UPDATE Students
SET Email = 'ravi.pakala@example.com'
WHERE StudentID = 1;
```

Delete Marks Less Than 60

```
DELETE FROM Marks WHERE MarksObtained < 60;
```

Select Students Whose First Name Starts with 'A'

```
SELECT * FROM Students
WHERE FirstName LIKE 'A%';
```

Average Marks per Student (Only Students with Avg > 80)

```
SELECT
    S.StudentID,
    CONCAT(
S.FirstName, ' ', S.LastName) AS StudentName,
    ROUND(AVG(M.MarksObtained), 2) AS AverageMarks
FROM Students S
JOIN Marks M ON S.StudentID = M.StudentID
GROUP BY S.StudentID
HAVING AVG(M.MarksObtained) > 80;
```

Students Who Scored Above Average in Mathematics

```
SELECT FirstName, LastName
FROM Students
WHERE StudentID IN (
    SELECT StudentID
    FROM Marks
    WHERE CourseID = (SELECT CourseID FROM Courses WHERE CourseName =
'Mathematics')
    AND MarksObtained > (
        SELECT AVG(MarksObtained)
        FROM Marks
        WHERE CourseID = (SELECT CourseID FROM Courses WHERE CourseName =
'Mathematics')
    )
);
```

Procedure Definition

```
DELIMITER $$
CREATE PROCEDURE GetStudentReportCard(IN stu_id INT)
BEGIN
    SELECT
        S.StudentID,
        CONCAT(S.FirstName, ' ', S.LastName) AS StudentName,
        C.CourseName,
        M.MarksObtained,
        M.MaxMarks,
        ROUND((M.MarksObtained * 100.0 / M.MaxMarks), 2) AS Percentage
    FROM Students S
    JOIN Marks M ON S.StudentID = M.StudentID
    JOIN Courses C ON M.CourseID = C.CourseID
```

```
    WHERE S.StudentID = stu_id;
END$$
DELIMITER ;
```

Call Procedure for Student ID 3

```
CALL GetStudentReportCard(3);
```

Trigger Definition

```
DELIMITER $$
CREATE TRIGGER LogHighScores
AFTER INSERT ON Marks
FOR EACH ROW
BEGIN
    IF NEW.MarksObtained >= 90 THEN
        INSERT INTO ScoreLog (StudentID, CourseID, MarksObtained)
        VALUES (NEW.StudentID, NEW.CourseID, NEW.MarksObtained);
    END IF;
END$$
DELIMITER ;
```

Insert Mark that Triggers Logging

```
INSERT INTO Marks (StudentID, CourseID, MarksObtained, ExamDate)
VALUES (2, 1, 95, '2025-08-10');
```

View ScoreLog Table

```
SELECT * FROM ScoreLog;
```

Conclusion

The Student Report Card System is a comprehensive database project that successfully stores student, course, and marks data; performs data manipulation and analysis; generates student report cards using stored procedures; and logs exceptional performances with triggers. The system demonstrates effective use of SQL features for academic data management and can be extended further with user interfaces and analytics.