DATABASE MANAGEMENT SYSTEM - CSA0593

ASSIGNMENT 3

B.LAKSHMI ANJALI

192311344

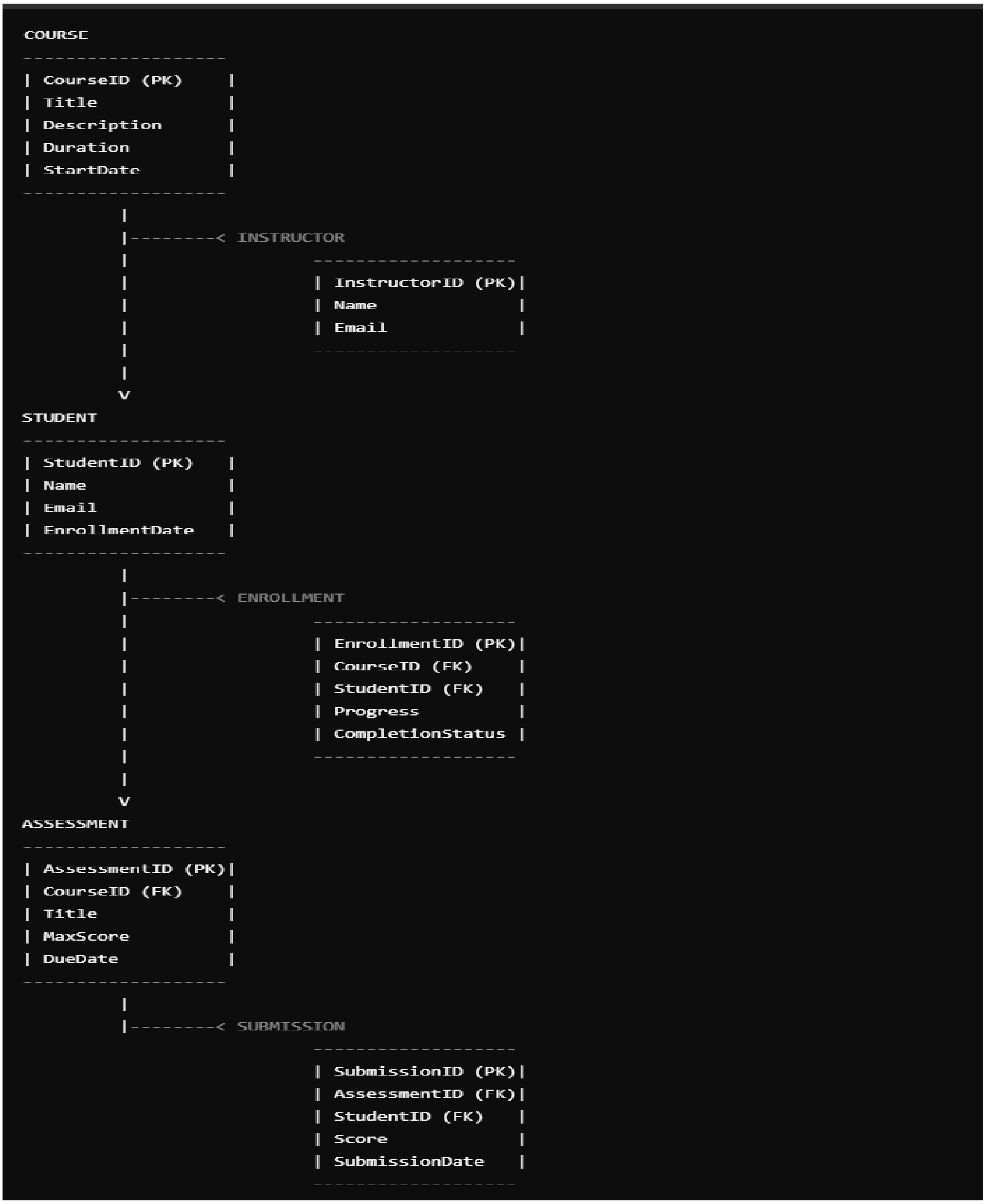
QUESTION:

Model tables for courses, instructors, students, and assessments.

- **Write stored procedures for enrolling students, tracking assessments, and updating course completions.**
- **Implement triggers to update course progress and notify instructors when new assessments are submitted.**
- **Write SQL queries to generate student progress reports, course completion statistics, and instructor feedback summaries**

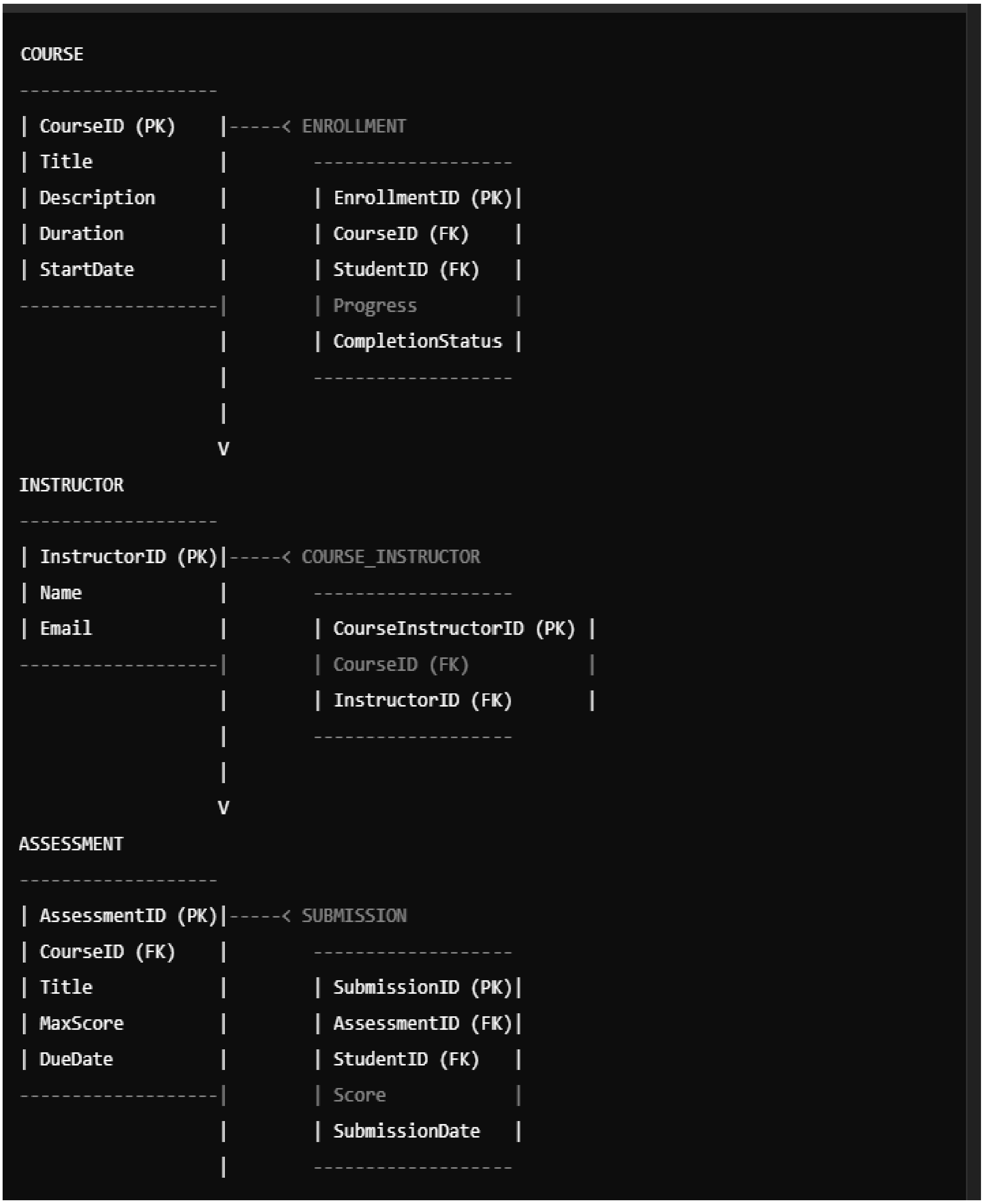
ANSWER:

CONCEPTUAL E.R.DIAGRAM:



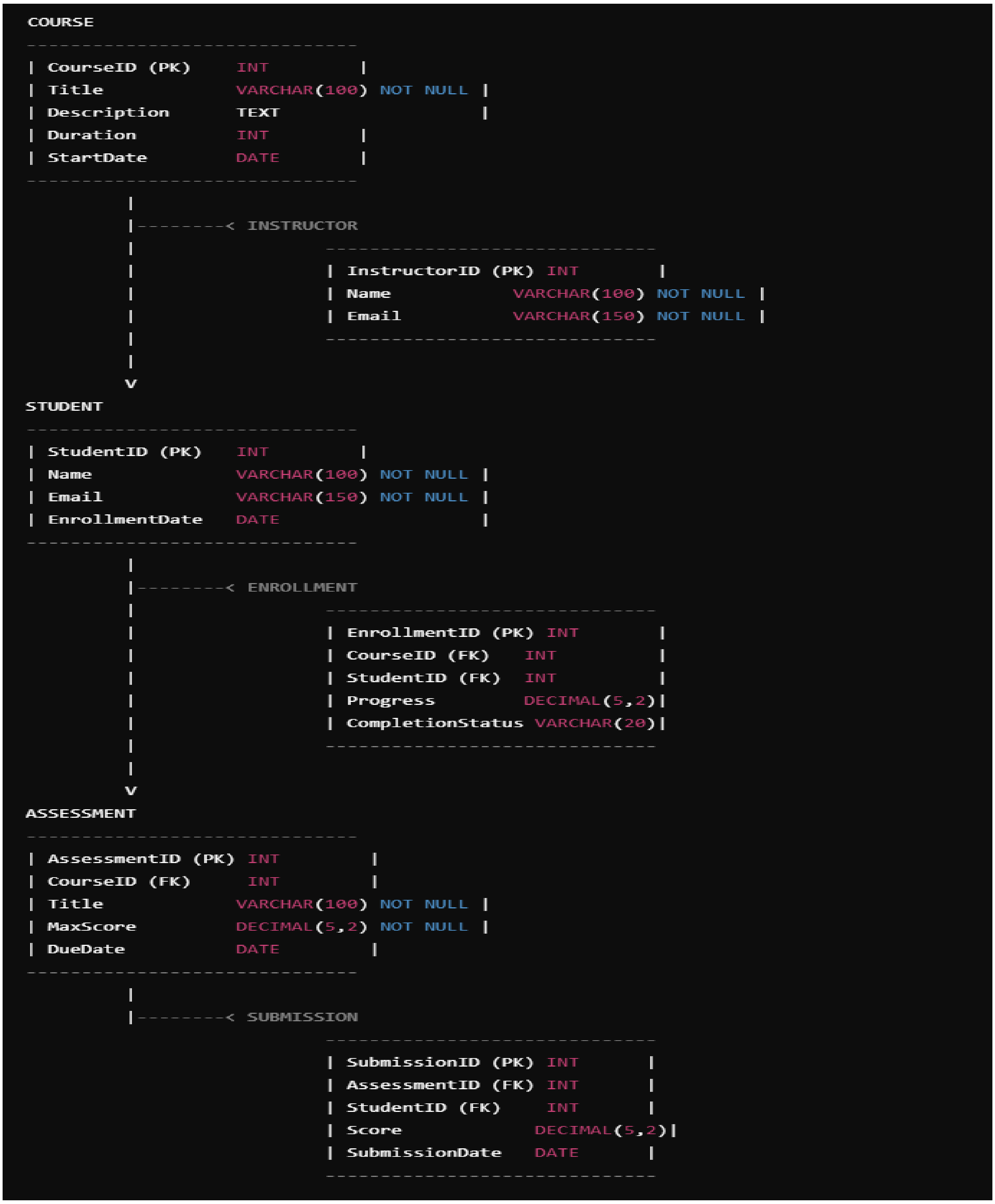
```
COURSE
-------------------
| CourseID (PK)    |
| Title            |
| Description      |
| Duration         |
| StartDate        |
-------------------
         |
         |--------< INSTRUCTOR
         |                          -------------------
         |                          | InstructorID (PK)|
         |                          | Name             |
         |                          | Email            |
         |                          -------------------
         |
         |
         v
STUDENT
-------------------
| StudentID (PK)   |
| Name             |
| Email            |
| EnrollmentDate   |
-------------------
         |
         |--------< ENROLLMENT
         |                          -------------------
         |                          | EnrollmentID (PK)|
         |                          | CourseID (FK)    |
         |                          | StudentID (FK)   |
         |                          | Progress         |
         |                          | CompletionStatus |
         |                          -------------------
         |
         v
ASSESSMENT
-------------------
| AssessmentID (PK)|
| CourseID (FK)    |
| Title            |
| MaxScore         |
| DueDate          |
-------------------
         |
         |--------< SUBMISSION
                                    -------------------
                                    | SubmissionID (PK)|
                                    | AssessmentID (FK)|
                                    | StudentID (FK)   |
                                    | Score            |
                                    | SubmissionDate   |
                                    -------------------
```

LOGICAL E.R DIAGRAM:



```
COURSE
-------------------
| CourseID (PK)    |-----< ENROLLMENT
| Title            |        -------------------
| Description      |        | EnrollmentID (PK)|
| Duration         |        | CourseID (FK)    |
| StartDate        |        | StudentID (FK)   |
-------------------|        | Progress         |
                   |        | CompletionStatus |
                   |        -------------------
                   |
                   V
INSTRUCTOR
-------------------
| InstructorID (PK)|-----< COURSE_INSTRUCTOR
| Name             |        -------------------
| Email            |        | CourseInstructorID (PK) |
-------------------|        | CourseID (FK)           |
                   |        | InstructorID (FK)       |
                   |        -------------------
                   |
                   V
ASSESSMENT
-------------------
| AssessmentID (PK)|-----< SUBMISSION
| CourseID (FK)    |        -------------------
| Title            |        | SubmissionID (PK)|
| MaxScore         |        | AssessmentID (FK)|
| DueDate          |        | StudentID (FK)   |
-------------------|        | Score            |
                   |        | SubmissionDate   |
                   |        -------------------
```

PHYSICAL E.R.DIAGRAM:



```
COURSE
------------------------------
| CourseID (PK)     INT          |
| Title             VARCHAR(100) NOT NULL |
| Description       TEXT                  |
| Duration          INT          |
| StartDate         DATE         |
------------------------------
         |
         |--------< INSTRUCTOR
         |                 ------------------------------
         |                 | InstructorID (PK) INT          |
         |                 | Name              VARCHAR(100) NOT NULL |
         |                 | Email             VARCHAR(150) NOT NULL |
         |                 ------------------------------
         |
         |
         v
STUDENT
------------------------------
| StudentID (PK)    INT          |
| Name              VARCHAR(100) NOT NULL |
| Email             VARCHAR(150) NOT NULL |
| EnrollmentDate    DATE                  |
------------------------------
         |
         |--------< ENROLLMENT
         |                 ------------------------------
         |                 | EnrollmentID (PK) INT          |
         |                 | CourseID (FK)    INT           |
         |                 | StudentID (FK)   INT           |
         |                 | Progress         DECIMAL(5,2)|
         |                 | CompletionStatus VARCHAR(20)|
         |                 ------------------------------
         |
         |
         v
ASSESSMENT
------------------------------
| AssessmentID (PK) INT          |
| CourseID (FK)     INT          |
| Title             VARCHAR(100) NOT NULL |
| MaxScore          DECIMAL(5,2) NOT NULL |
| DueDate           DATE         |
------------------------------
         |
         |--------< SUBMISSION
                           ------------------------------
                           | SubmissionID (PK) INT        |
                           | AssessmentID (FK) INT        |
                           | StudentID (FK)    INT        |
                           | Score            DECIMAL(5,2)|
                           | SubmissionDate   DATE        |
                           ------------------------------
```

MYSQL STATEMENTS:

```
mysql
CREATE DATABASE LearningManagement;

USE LearningManagement;

CREATE TABLE Courses (
  CourseID INT AUTO_INCREMENT PRIMARY KEY,
  CourseName VARCHAR(100),
  CourseDescription VARCHAR(255),
  Duration INT
);

CREATE TABLE Instructors (
  InstructorID INT AUTO_INCREMENT PRIMARY
KEY,
  FirstName VARCHAR(50),
  LastName VARCHAR(50),
  Email VARCHAR(100)
);

CREATE TABLE Students (
  StudentID INT AUTO_INCREMENT PRIMARY KEY,
  FirstName VARCHAR(50),
  LastName VARCHAR(50),
  Email VARCHAR(100)
);

CREATE TABLE Enrollments (
  EnrollmentID INT AUTO_INCREMENT PRIMARY
KEY,
  CourseID INT,
  StudentID INT,
  EnrollmentDate DATE,
  CompletionStatus VARCHAR(50),
  FOREIGN KEY (CourseID) REFERENCES
Courses(CourseID),
  FOREIGN KEY (StudentID) REFERENCES
Students(StudentID)
);

CREATE TABLE Assessments (
  AssessmentID INT AUTO_INCREMENT PRIMARY
KEY,
  CourseID INT,
  StudentID INT,
  SubmissionDate DATE,
  Score DECIMAL(10, 2),
  FOREIGN KEY (CourseID) REFERENCES
Courses(CourseID),
  FOREIGN KEY (StudentID) REFERENCES
Students(StudentID)
);

CREATE TABLE Feedback (
  FeedbackID INT AUTO_INCREMENT PRIMARY
KEY,
  AssessmentID INT,
  InstructorID INT,
  FeedbackText VARCHAR(255),
  FOREIGN KEY (AssessmentID) REFERENCES
Assessments(AssessmentID),
  FOREIGN KEY (InstructorID) REFERENCES
Instructors(InstructorID)
);
```

Stored Procedures:

```
mysql
DELIMITER //

CREATE PROCEDURE sp_EnrollStudent(
  IN courseID INT,
  IN studentID INT,
  IN enrollmentDate DATE
)
BEGIN
  INSERT INTO Enrollments (CourseID, StudentID,
EnrollmentDate, CompletionStatus)
  VALUES (courseID, studentID, enrollmentDate, 'In
Progress');
END //

CREATE PROCEDURE sp_SubmitAssessment(
  IN assessmentID INT,
  IN courseID INT,
  IN studentID INT,
  IN submissionDate DATE,
  IN score DECIMAL(10, 2)
)
BEGIN
  INSERT INTO Assessments (AssessmentID,
CourseID, StudentID, SubmissionDate, Score)
```

```
  VALUES (assessmentID, courseID, studentID,
submissionDate, score);
END //

CREATE PROCEDURE
sp_UpdateCourseCompletion(
  IN enrollmentID INT,
  IN completionStatus VARCHAR(50)
)
BEGIN
  UPDATE Enrollments
  SET CompletionStatus = completionStatus
  WHERE EnrollmentID = enrollmentID;
END //

DELIMITER;
```

Triggers:

```
mysql
DELIMITER //

CREATE TRIGGER tr_UpdateCourseProgress
AFTER UPDATE ON Enrollments
FOR EACH ROW
BEGIN
  UPDATE Students
  SET CourseProgress = (SELECT COUNT(*) FROM
Enrollments WHERE StudentID = NEW.StudentID
AND CompletionStatus = 'Completed')
  WHERE StudentID = NEW.StudentID;
END //

CREATE TRIGGER tr_NotifyInstructor
AFTER INSERT ON Assessments
FOR EACH ROW
```

```
BEGIN
  DECLARE instructorEmail VARCHAR(100);
  SELECT Email INTO instructorEmail FROM Instructors WHERE InstructorID = (SELECT InstructorID FROM Courses WHERE CourseID = NEW.CourseID);

  -- Send email notification to instructor
  INSERT INTO EmailNotifications (InstructorEmail, Subject, Body)
  VALUES (instructorEmail, 'New Assessment Submission', 'A new assessment has been submitted for your review.');
END //

DELIMITER;
```

SQL Queries:

```
mysql
-- Student Progress Report
SELECT
  Students.StudentID,
  Students.FirstName,
  Students.LastName,
  Courses.CourseName,
  Enrollments.CompletionStatus
FROM
  Students
  JOIN Enrollments ON Students.StudentID =
Enrollments.StudentID
  JOIN Courses ON Enrollments.CourseID =
Courses.CourseID;

-- Course Completion Statistics
SELECT
  Courses.CourseName,
  COUNT(*) AS TotalEnrollments,
  SUM(CASE WHEN Enrollments.CompletionStatus
= 'Completed' THEN 1 ELSE 0 END) AS
TotalCompletions
FROM
  Courses
  JOIN Enrollments ON Courses.CourseID =
Enrollments.CourseID
GROUP BY
  Courses.CourseName;

-- Instructor Feedback Summary
SELECT
  Instructors.InstructorID,
  Instructors.FirstName,
  Instructors.LastName,
  COUNT(*) AS TotalFeedback,
  AVG(FEEDBACK.Score) AS AverageScore
FROM
```

```
  Instructors
  JOIN Feedback ON Instructors.InstructorID =
Feedback.InstructorID
  JOIN Assessments ON Feedback.AssessmentID =
Assessments.AssessmentID
GROUP BY
  Instructors.InstructorID, Instructors.FirstName,
Instructors.LastName;
```

Conclusion:

This database design provides a comprehensive foundation for managing courses, instructors, students, and assessments. The stored procedures simplify student enrollment, assessment submission, and course completion updates, while the triggers ensure data consistency and accuracy. The SQL queries enable generation of student progress