Here’s a complete SQL Server project for beginners that involves creating a database for a basic **Student Management System**. This project covers designing tables, setting up relationships, writing SQL queries, and performing CRUD (Create, Read, Update, Delete) operations. It’s a great way to understand database schema design, table relationships, and writing SQL queries in a practical, project-based way.

**Project: Student Management System**

**Project Overview**

In this project, you will create a system that allows you to manage students, courses, and enrollments. The goal is to set up a SQL Server database to track:

1. Students and their information
2. Courses offered
3. Enrollment details linking students to courses

**Step 1: Setting Up the Database and Tables**

1. **Create the Database**: Start by creating a new database in SQL Server named StudentManagementDB.

sql

Copy code

CREATE DATABASE StudentManagementDB;

USE StudentManagementDB;

1. **Design the Database Schema**: We will have three main tables: Students, Courses, and Enrollments.
2. **Table Creation Statements**:
   * **Students Table**: This table will store student information such as ID, name, age, and email.

sql

Copy code

CREATE TABLE Students (

StudentID INT PRIMARY KEY IDENTITY,

FirstName NVARCHAR(50),

LastName NVARCHAR(50),

Age INT,

Email NVARCHAR(100) UNIQUE

);

* + **Courses Table**: This table will store course information like Course ID, course name, and description.

sql

Copy code

CREATE TABLE Courses (

CourseID INT PRIMARY KEY IDENTITY,

CourseName NVARCHAR(100),

Description NVARCHAR(255)

);

* + **Enrollments Table**: This table creates a many-to-many relationship between Students and Courses. It will include foreign keys from both Students and Courses.

sql

Copy code

CREATE TABLE Enrollments (

EnrollmentID INT PRIMARY KEY IDENTITY,

StudentID INT FOREIGN KEY REFERENCES Students(StudentID),

CourseID INT FOREIGN KEY REFERENCES Courses(CourseID),

EnrollmentDate DATE DEFAULT GETDATE()

);

**Step 2: Populating the Tables**

Insert sample data into the tables for testing.

* **Inserting Students**:

sql

Copy code

INSERT INTO Students (FirstName, LastName, Age, Email) VALUES

('John', 'Doe', 20, 'john.doe@example.com'),

('Jane', 'Smith', 22, 'jane.smith@example.com'),

('Sam', 'Taylor', 19, 'sam.taylor@example.com');

* **Inserting Courses**:

sql

Copy code

INSERT INTO Courses (CourseName, Description) VALUES

('Mathematics', 'Introductory Mathematics Course'),

('History', 'World History from 1800 to Present'),

('Computer Science', 'Basics of Computer Science');

* **Enrolling Students in Courses**:

sql

Copy code

INSERT INTO Enrollments (StudentID, CourseID) VALUES

(1, 1), (1, 2), (2, 3), (3, 1);

**Step 3: Writing SQL Queries**

Now that the tables have been populated, you can perform basic SQL operations.

1. **Retrieve All Students**:

sql

Copy code

SELECT \* FROM Students;

1. **List All Courses with Enrolled Students**:

sql

Copy code

SELECT c.CourseName, s.FirstName, s.LastName

FROM Enrollments e

JOIN Students s ON e.StudentID = s.StudentID

JOIN Courses c ON e.CourseID = c.CourseID;

1. **Find Students Enrolled in a Specific Course**:

sql

Copy code

SELECT s.FirstName, s.LastName

FROM Enrollments e

JOIN Students s ON e.StudentID = s.StudentID

WHERE e.CourseID = 1; -- Replace with desired CourseID

1. **Update a Student's Information**:

sql

Copy code

UPDATE Students

SET Email = 'new.email@example.com'

WHERE StudentID = 1;

1. **Delete an Enrollment Record**:

sql

Copy code

DELETE FROM Enrollments

WHERE EnrollmentID = 1;

**Step 4: Additional Concepts**

1. **Adding Constraints**: Add a check constraint on the Age column to ensure it’s within a valid range.

sql

Copy code

ALTER TABLE Students

ADD CONSTRAINT chk\_Age CHECK (Age >= 18 AND Age <= 60);

1. **Creating Views**: Create a view to easily see students and their courses.

sql

Copy code

CREATE VIEW StudentCourses AS

SELECT s.FirstName, s.LastName, c.CourseName

FROM Enrollments e

JOIN Students s ON e.StudentID = s.StudentID

JOIN Courses c ON e.CourseID = c.CourseID;

1. **Stored Procedures**: Write a stored procedure to add a new student and enroll them in a course.

sql

Copy code

CREATE PROCEDURE AddStudentAndEnroll

@FirstName NVARCHAR(50),

@LastName NVARCHAR(50),

@Age INT,

@Email NVARCHAR(100),

@CourseID INT

AS

BEGIN

DECLARE @StudentID INT;

INSERT INTO Students (FirstName, LastName, Age, Email)

VALUES (@FirstName, @LastName, @Age, @Email);

SET @StudentID = SCOPE\_IDENTITY();

INSERT INTO Enrollments (StudentID, CourseID)

VALUES (@StudentID, @CourseID);

END;

**Project Wrap-up**

This project should give you a good understanding of creating a database schema, managing relationships, and performing CRUD operations in SQL Server. By the end, you’ll have a complete and functional Student Management System, a foundational project for learning SQL and database management.