🎓 Student Management System (SMS) Database Documentation

Prepared By: Team Lead

The **Student Management System (SMS)** is a relational database designed to efficiently manage and organize information about students, departments, courses, instructors, and enrollments within a university setting. The system supports key administrative operations such as:

- Registering students and instructors
- Assigning instructors to courses
- Enrolling students in courses
- Generating analytical and operational reports through SQL queries

The database was created using **PostgreSQL**, with structured relationships and constraints to ensure data integrity, scalability, and efficient querying.

Objective

The primary objective of this project is to design and create a relational database for a Student Management System using SQL. The database should:

- Efficiently store and manage data on students, courses, instructors, and enrollments.
- Support administrative tasks such as course assignments and enrollments.
- Allow users to generate dynamic, data-driven reports for academic insights.

Database Structure

The Student Management System is built on **six related tables**.

1. **Students' Table:** Stores student personal and academic details.

Column	Data Type	Description
Student_ID	VARCHAR (PK)	Unique identifier for each student
Name	VARCHAR	Student's full name
Gender	VARCHAR	Gender of the student
DOB	DATE	Date of birth
Department_ID	VARCHAR (FK)	References Departments.Department_ID

2. **Departments' Table:** Contains information about the university departments.

Column	Data Type	Description
Department_ID	VARCHAR (PK)	Unique identifier for each department
Department_Name	VARCHAR	Name of the department

3. **Courses' Table:** Defines courses offered by departments.

Column	Data Type	Description
Course_ID	VARCHAR (PK)	Unique course identifier
Course_Name	VARCHAR	Course title
Department_ID	VARCHAR (FK)	References Departments.Department_ID

4. **Instructors' Table:** Stores instructor data including the department they are affiliated with.

Column	Data Type	Description
Instructor_ID	VARCHAR (PK)	Unique instructor ID
Name	VARCHAR	Instructor's full name
Gender	VARCHAR	Gender of instructor
Email	VARCHAR	Instructor's email address
Hire_Date	DATE	Date of hire
Department_ID	VARCHAR (FK)	References Departments.Department_ID

5. **Enrollments' Table:** Tracks student course registrations.

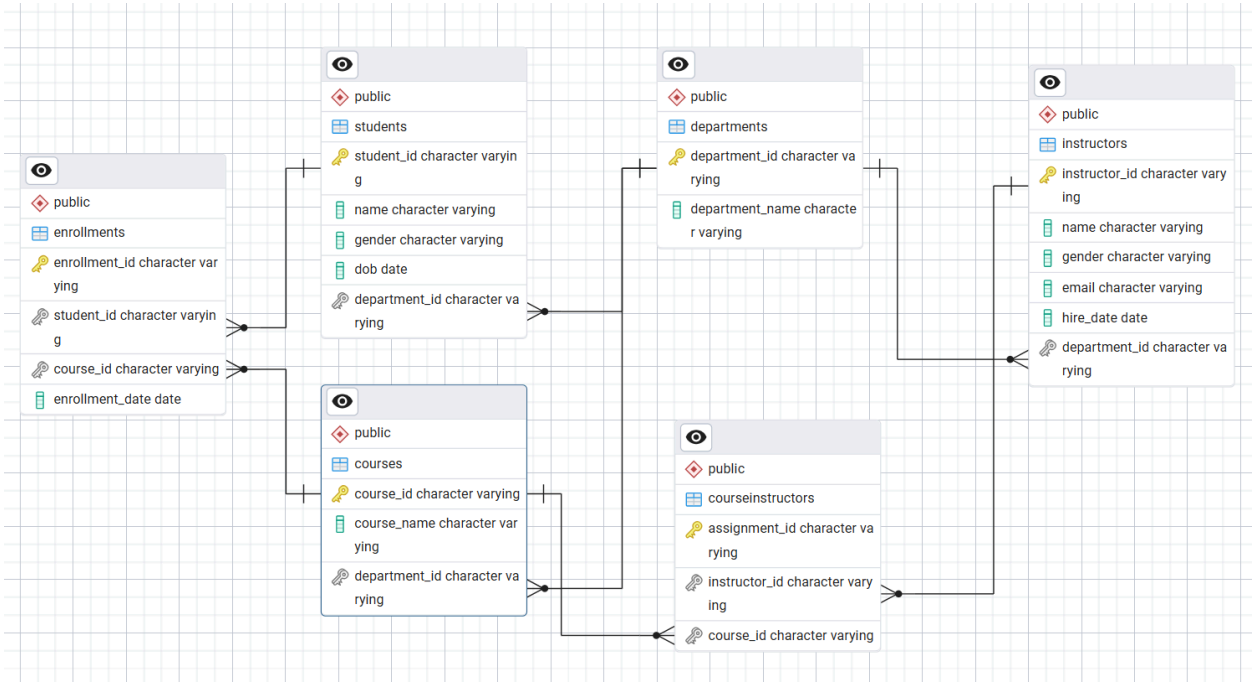
Column	Data Type	Description
Enrollment_ID	VARCHAR (PK)	Unique enrollment ID
Student_ID	VARCHAR (FK)	References Students.Student_ID
Course_ID	VARCHAR (FK)	References Courses.Course_ID
Enrollment_Date	DATE	Date of enrollment

6. **CourseInstructors' Table:** Links instructors and courses (many-to-many relationship).

Column	Data Type	Description
Assignment_ID	VARCHAR (PK)	Unique identifier for course assignment
Instructor_ID	VARCHAR (FK)	References Instructors.Instructor_ID
Course_ID	VARCHAR (FK)	References Courses.Course_ID

Entity Relationship Diagram (ERD)

The ERD visually represents the relationships between tables in the database.



Relationship Overview:

- One Department → Many Students, Courses, and Instructors
- One Student → Many Enrollments
- One Course → Many Enrollments
- One Instructor → Many Course Assignments
- Courses and Instructors have a many-to-many relationship via CourseInstructors

Data Population

Each table was populated with realistic sample data including:

- 6 Departments
- 20 Courses
- 15 Instructors
- 50 Students
- 100 Enrollment Records
- 28 Course Assignments

The dataset provides a realistic simulation of university operations.

SQL - Based Reporting Questions

Students & Enrollments Report

1. **How many students are currently enrolled in each course?**
   The result shows the number of students enrolled in each course, with Introduction to Programming CS001, Data Structures CS002, Linear Algebra CS009, having the highest (7) students and Academic Writing CS019 with the lowest (3) students.

2. **Which students are enrolled in multiple courses, and which courses are they taking?**
   The result shows that 49 students are enrolled in multiple courses, Student ID 004 - Chiamaka Obi registered three courses making her the only student with the highest number of course enrollments.

3. **What is the total number of students per department across all courses?**
   The Computer Science department has the highest number of students (12 students). Physics and Biology departments have the least number of students. (7 students each).

Course & Instructor Analysis

4. **Which courses have the highest number of enrollments?**

   CS009 (Linear Algebra), CS001 (Introduction to Programming), and CS002 (Data Structures) have the highest number of student enrollments.

5. **Which department has the least number of students?**

   The Physics and Biology Departments have the least number of students.

Data Integrity & Operational Insights

6. **Are there any students not enrolled in any course?**

   There are no students currently not enrolled in any course.

7. **How many courses does each student take on average?**

   Each student takes two courses on average.

8. **What is the gender distribution of students across courses and instructors?**

   Mr. Daniel Peters, who teaches Linear Algebra, has the highest number of female students in his class.

   Dr. Samuel Okoro and Dr. Samuel Kuti, who teaches Data Structures, has the highest number of male students in their classes.

9. **Which course has the highest number of male or female students enrolled?**

   The Data Structures course has the highest number of male students enrolled, while the Linear Algebra course has the highest number of female students enrolled.

Query Logic Summary

Query Objective	Logic Explanation
Total Students per Course	Counts enrollments grouped by course name.
Students Enrolled in Multiple Courses	Uses HAVING COUNT() > 1 to identify students in multiple courses.
Students per Department	Groups students by department via Department_ID.
Average Courses per Student	Calculates mean value of total enrolled courses per student.
Gender Distribution	Joins multiple tables to summarize male/female counts per instructor and course.

Conclusion

The **Student Management System (SMS) database** demonstrates the power of relational design in efficiently managing academic records.

By establishing clear relationships between **students, courses, instructors, and departments**, the database supports dynamic queries that produce insightful academic and administrative reports.

This design ensures:

- Data consistency through primary and foreign keys
- Scalability for future data additions
- Flexibility for advanced analytical reporting