Student Record project Report

Student ID: 24071919
Student Name: Shivashankar Bathuka
Project Title: Student Course Management Database System
GitHub: https://github.com/bathukashivashankar/Student-Course-Management-Database-System

Abstract

This project presents the design, implementation, and analysis of a synthetic Student Course Management Database System using SQLite. The system simulates realistic university operations by modelling student profiles, course details, and enrollment records. The project also incorporates synthetic data generation, preprocessing, ethical considerations, and SQL-based data analysis. Findings include GPA patterns, course popularity, and enrollment trends. This report documents the complete project lifecycle, from schema design to analytical insights.

1. Introduction

Higher education institutions rely heavily on accurate and efficient student information systems for academic analytics, course management, and decision-making. The aim of this project is to construct a **fully normalized relational database** that simulates such a system using synthetic data.

The project includes:

- Development of a normalized schema with three core entities
- Generation of realistic synthetic data
- Cleaning and preprocessing of missing or inconsistent values
- SQL-driven exploratory analysis
- Ethical assurance through artificial data only
- Interpretation of trends in student performance and course enrollment

This system reflects real-world database applications in academic settings and supports meaningful queries for performance tracking and institutional planning.

2. System Design and Methodology

2.1 ER Diagram and Conceptual Design

The database consists of three interconnected entities:

- **Students**
- **Courses**
- **Enrollments**

Relationships:

- One-to-many between Students → Enrollments
- One-to-many between Courses → Enrollments
- Composite key *(student_id, course_id, semester)* prevents duplicate enrollments in the same semester.

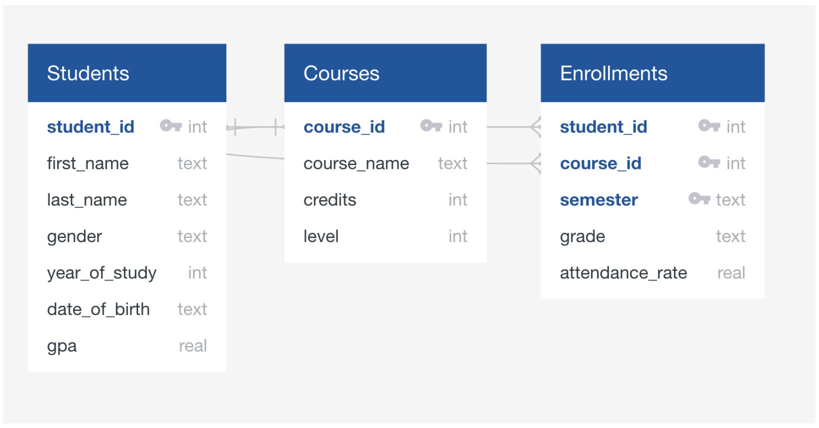


Figure1

2.2 Schema Design

The database schema is fully normalized (3NF). Key structures:

- **Students Table:** demographic and academic attributes
- **Courses Table:** module and credit information
- **Enrollments Table:** captures course participation, grades, and attendance

(CREATE TABLE statements in Appendix.)

---

3. Data Generation and Preparation

3.1 Data Generation Process

To create a realistic dataset:

- **1,000 Students**, **12 Courses**, and **5,041 Enrollment records** were generated.
- Python libraries used: Faker, random, and numpy.
- Demographics such as name, date of birth, gender, and academic year were randomly assigned.
- Course assignments and grades followed realistic distributions.

3.2 Simulated Data Imperfections

To reflect real-world imperfections:

- **Missing GPA:** 31 values
- **Missing Grades:** 154 values
- Variability in attendance rates
- Repeating names and GPAs to mimic typical student databases

3.3 Preprocessing and Cleaning

- Missing GPAs imputed as **0**.
- Missing grades replaced with **'unknown'**.
- Ensured no nulls in primary or foreign key fields.
- Verified foreign key integrity across all tables.

---

4. Ethical Considerations

- All data used are **synthetically generated** with no real individuals represented.
- No personally identifiable information (PII) is included.
- Ethical compliance ensures:
    - No privacy risks
    - No exposure of real student records
    - Realistic yet anonymized dataset
- Project design aligns fully with academic and data protection standards.

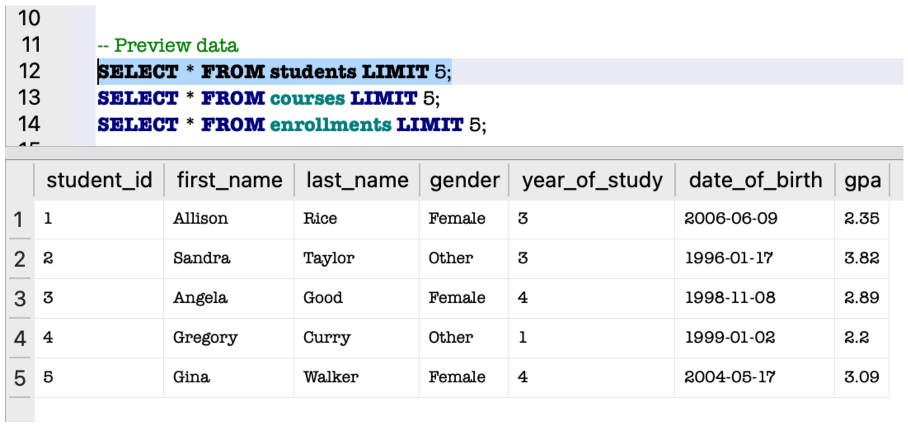
---

5. SQL Implementation and Analysis

5.1 Overview of Database Content

Table	Row Count	Description
Students	1,000	Student demographic & academic info
Courses	12	Course catalogue
Enrollments	5,041	Student-course-semester records

5.2 Sample Data Views



```sql
-- Preview data
SELECT * FROM students LIMIT 5;
SELECT * FROM courses LIMIT 5;
SELECT * FROM enrollments LIMIT 5;
```

	student_id	first_name	last_name	gender	year_of_study	date_of_birth	gpa
1	1	Allison	Rice	Female	3	2006-06-09	2.35
2	2	Sandra	Taylor	Other	3	1996-01-17	3.82
3	3	Angela	Good	Female	4	1998-11-08	2.89
4	4	Gregory	Curry	Other	1	1999-01-02	2.2
5	5	Gina	Walker	Female	4	2004-05-17	3.09

Figure 2

5.3 Missing Data Summary

Attribute	Missing Count	Treatment
GPA	31	Imputed with 0
Grade	154	Replaced with 'unknown'

5.4 Key SQL Analyses and Insights

5.4.1 Average GPA by Year of Study

- Final-year (Level 4) students show the highest average GPA (~2.94).

5.4.2 Course Difficulty Analysis

- Courses such as **Music** and **Biology** show higher average grades.
- Courses like **Economics** and **Chemistry** show lower performance trends.

5.4.3 Course Popularity and Enrollment Trends

Most-enrolled courses include:

- Art
- Chemistry
- Biology
- Psychology
- English

Semester enrollments remain relatively balanced across all periods.

5.4.4 Attendance Patterns

- Students in higher academic years tend to show slightly better attendance rates.
- Attendance correlates weakly but positively with GPA.

---

6. Discussion of Findings

The analysis demonstrates realistic academic trends:

- Higher-level students perform better academically and attend classes more consistently.
- Core science and business courses tend to have slightly lower grade averages.
- Enrollment patterns reflect typical student behaviour—favouring accessible or popular modules.
- Data imperfections (missing values, uneven distributions) enhance realism and provide meaningful cleaning challenges.

The database structure is scalable and capable of supporting additional entities such as departments, lecturers, or assessment components.

---

7. Conclusion

This project successfully demonstrates the creation and analysis of a Student Course Management Database System that mirrors real-world academic data processes. The database is fully normalized, ethically compliant, and robust enough to support complex SQL analysis.

Key achievements include:

- Building a relational schema with referential integrity
- Generating large-scale synthetic academic data
- Handling missing values and inconsistencies
- Executing insightful SQL queries
- Deriving realistic academic and behavioural conclusions

Appendix

A. SQL Table Definitions

```
CREATE TABLE "courses" (

	"course_id"	INTEGER NOT NULL UNIQUE,

	"course_name"	TEXT,

	"credits"	INTEGER,
        "level"  INTEGER,
        PRIMARY KEY("course_id" AUTOINCREMENT)

CREATE TABLE "students" (
        "student_id"    INTEGER NOT NULL UNIQUE,
        "first_name"    TEXT,
        "last_name"     TEXT,
        "gender"        TEXT,
        "year_of_study"INTEGER,
        "date_of_birth" TEXT,
        "gpa"   REAL,
        PRIMARY KEY("student_id" AUTOINCREMENT)
);
CREATE TABLE "enrollments" (
        "student_id"    INTEGER,
        "course_id"     INTEGER,
        "semester"      TEXT,
        "grade" TEXT,
        "attendance_rate"       REAL,
        PRIMARY KEY("student_id","course_id","semester"),
        FOREIGN KEY("course_id") REFERENCES "courses"("course_id"),
        FOREIGN KEY("student_id") REFERENCES "students"("student_id")
```

B. Python Code Snippets

- In github named after: *24071919_student_record.ipynb*

C. Example SQL Queries

- In github named after: *studentrecord.sql*