CampusFlow

End-to-End Relational Database Design for University Registration

Koorosh Asil Gharehbaghi

K. N. Toosi University of Technology

Department of Computer Science, Faculty of Mathematics

June 2025

This project showcases a complete, normalized relational database system built from scratch to simulate a real-world university registration platform. It supports course management, student enrollment, instructor operations, GPA calculation, and transcript reporting — all within a clean and scalable schema design.

Overview

This project, titled **CampusFlow**, was created as a final assignment for the course *Database Systems Design*. It presents a fully normalized, relational database system designed from the ground up to simulate a real-world university registration system.

Objective

The primary goal of the project is to develop a normalized, efficient, and scalable relational database that supports:

- Course creation and assignment
- Student enrollment and transcript management
- Instructor-course management
- Grade recording
- GPA calculation and prerequisite enforcement

System Users

- **Students**: Enroll in courses, view transcripts and GPAs
- **Instructors**: Manage assigned courses, assign and update grades
- **Admins**: Add courses, assign instructors, validate prerequisites, manage records

ER Diagram

The Entity-Relationship Diagram was designed using `draw.io` and includes the following entities and relationships:

- **Entities**: Student, Instructor, Course, Department, Enrollment, Grade
- **Relationships**:
  - Many-to-Many between Student and Course via Enrollment
  - One-to-Many from Department to Course
  - One-to-Many from Instructor to Course
  - One-to-One (optional) from Enrollment to Grade

Relational Schema

All entities were translated into tables using SQL. The design adheres to the Third Normal Form (3NF) and includes:

- `CREATE TABLE` statements with primary and foreign keys
- `NOT NULL`, `UNIQUE`, and other integrity constraints
- Sample `INSERT INTO` statements for test data

Advanced Features

- **Stored Function**: To calculate GPA per student
- **View**: Automatically formats and displays student transcripts
- **Trigger**: Prevents enrollment if prerequisites are not satisfied

Files and Structure

- `schema.sql` – SQL code for schema creation and sample population
- `ER_Diagram.pdf` – Visual ERD
- `report.pdf` – Summary of design decisions
- `README.md` – Project introduction and usage instructions

Tools Used

- **ERD Design**: draw.io
- **SQL Development**: MySQL Workbench, pgAdmin
- **Documentation**: Markdown, LaTeX

Conclusion

Designing the `CampusFlow` system was an exercise in both theoretical modeling and practical implementation. It emphasizes the importance of normalization, integrity constraints, and real-world usability. The database design supports future scalability and integrates well with front-end or middleware systems.

"Designing a database is like writing a story — every entity has a role, every relationship matters, and normalization keeps the plot clean and organized."

Koorosh Asil Gharehbaghi
Spring 2025