DATABASE DESIGN REPORT

By

Uttara Naidu

Submitted to

The University of Liverpool

14/07/2025

TABLE OF CONTENTS

Chapter 1. DESIGN PROCESS

1.1   Relational Database Design

The database design was done to comply with 1NF, 2NF and 3NF normalization stages. An ERD diagram designed accordingly is shown in figure 1. The records were populated in the database using a batch script called 'data_entry.sql'.
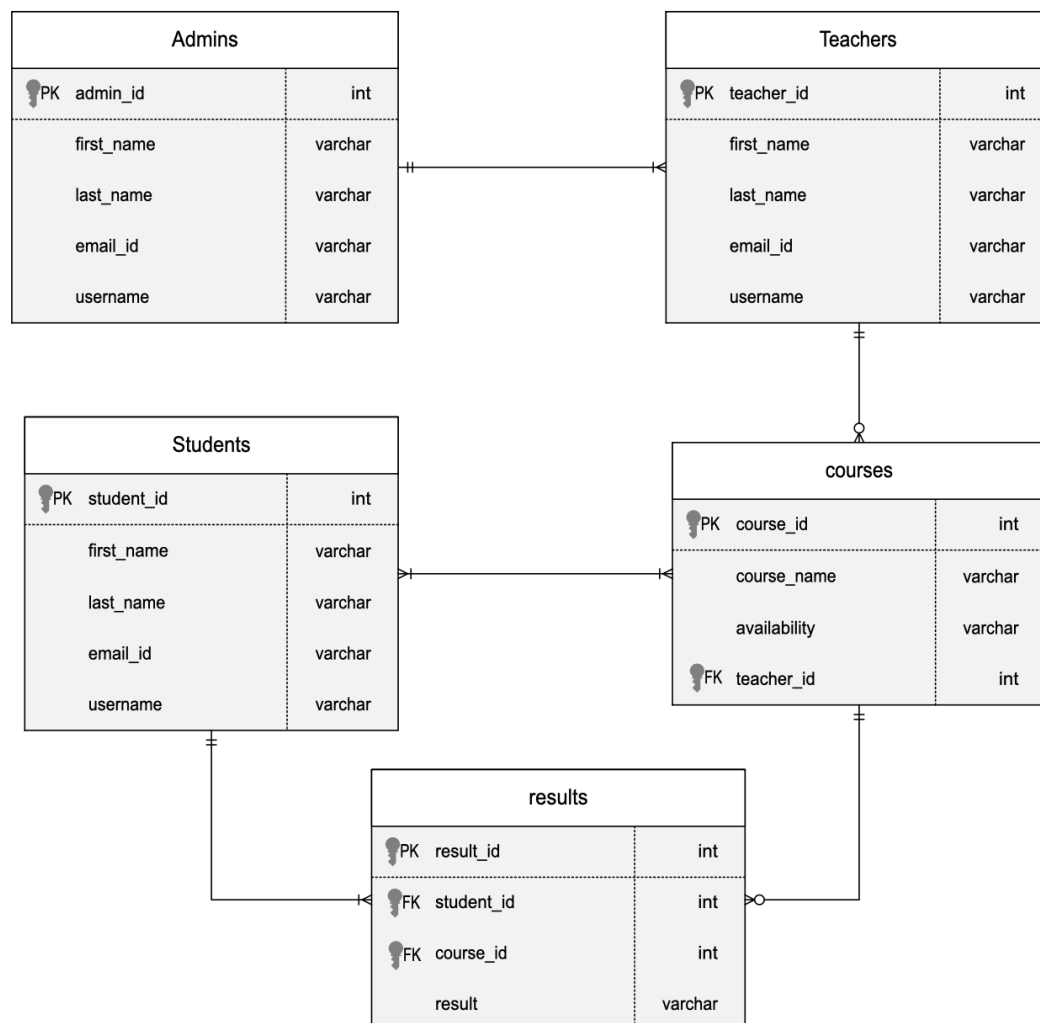

Figure 1. Entity Relationship Diagram

1.2   1NF

"In first normal form or 1NF, all attributes have only 1 piece of data and there should be no repeated data groups in an entity" (UoL,2025). Following this, each column of each table has a

single entity, e.g. splitting the name into two columns as `first_name` and `last_name`; thereby fulfilling the 1NF normalization.

1.3 2NF

The tables are 2NF compliant when all attributes are dependent on a single primary key (UoL,2025) and it is already in 1NF (Halpin,2024). This is achieved by introducing a primary key column for each of the tables, e.g. in table 'students', all columns like `first_name`, `last_name`, `email_id` and username depend on the primary key 'student_id'; thereby satisfying 2NF stage.

1.4 3NF

The tables are in 3NF when it is in 2NF, the attributes are mutually independent, and no indirect or transitive dependency exists between attributes (Halpin,2024). This was taken care of especially in 'results' table where only the primary keys 'course_id' and 'student_id' were used, eliminating the need for adding 'teacher_id' which would have created a transitive dependency with 'course_id'. A *View* named 'viewResult' was created using natural join to display students' results corresponding to each course and the teacher who graded it.

1.5 Requirements fulfillment

The functional requirement analysis led to choosing of a Relational data model with a decision to have separate tables for 'admins', 'teachers' and 'students' aligning with REQ003.

For table 'courses', complying with REQ004, a foreign key was introduced as 'teacher_id' to relate the respective courses with the faculty member, and an additional column 'availability' was added.

A table 'results' was created as a junction table (Halpin,2024) by pulling the primary keys from tables 'courses' and 'students' thereby establishing a relationship between the

two. Executing requirement REQ005, the teachers are permitted to update the results with the possible values `pass` or `fail` for each student.

Since the only privilege the students will have is viewing the available courses, a separate table `View` was created called `availableCourses` fulfilling REQ001. Configuring a frontend application with a call to this `View` in the backend will achieve the desired outcome. The students can use the View `viewResult` for retrieving their grades.

1.5.1 Setting privileges

A separate script is created `create_users.sql` aligned with the attribute `username` in tables `admins`, `students`, and `teachers`. Executing this script will create 3 users with their respective privileges; thereby complying with REQ006.

Chapter 2. CONCLUSION

A database design covering the core concepts of normalization, data entry and data retrieval for a college was designed and implemented successfully. Future improvements include using procedures for query calls, viewing the available courses along with the teacher's name and sorting the students sequentially by their IDs while viewing results.

(Word count: 496)

REFERENCES

Halpin, T.A. and Morgan, A.J. (2024) 'Information Modeling and Relational Databases'. Third edition. Cambridge, MA: Morgan Kaufmann.

Microsoft (2025), 'Map Many-to-Many Relationships'. Available: https://learn.microsoft.com/en-us/ssms/visual-db-tools/map-many-to-many-relationships-visual-database-tools. (Accessed on:13 July 2025)

University of Liverpool (2025) 'Lecturecast: Relational Database Model', *Week 2: Relational database model*. Available at: https://liverpool-online-study.com/course/view.php?id=3123&section=5. (Accessed on: 21 June 2025)