Database Project

UNIVERSITY SYSTEM

DECEMBER 2023

PREPARED BY

Our team

PRESENTED TO

Eng. Yassmina

Project Team

Marwan Muhammed Hussin Embaby
22010248
GUI and business rules

Abdulrahman Saied Abdulrahman
22010129
ERD, Logical & Relational schemas

Momen Adel Ali Elsayed
22010196
SQL code Implementation.

Marwan Hassan Ibrahim Ahmed
20221447788
Data population.

Hagar Ali Mosaad Elhawary
22010288
Project Report & Normalization

University Management System Database Project:

Comprehensive Overview

The University Management System Database Project is an ambitious initiative aimed at enhancing the efficiency and organization of various administrative processes within a university. This comprehensive database system is meticulously designed to handle a myriad of tasks, ranging from student enrollment and course management to faculty details and departmental administration

1. Project Objectives

The primary objectives of this project revolve around creating a centralized and robust database system that streamlines the complex interactions within a university ecosystem. The system aims to provide seamless management of student information, course offerings, faculty records, and departmental activities. By doing so, it seeks to enhance the overall operational effectiveness of the university.

2. Scope of the Project

The project encompasses the entire university landscape, covering diverse aspects of academic and administrative functions. Student-centric features include enrollment, academic record tracking, and personalized information management. Course-related functionalities involve the addition, modification, and deletion of courses, ensuring an up-to-date representation of the academic curriculum.

Features of our system :

1. Student Management:

Registration and tracking of student personal details.Academic record management, including grades and enrollment history. User-friendly interfaces for adding, updating, and deleting student information.

2.Course Management:

Dynamic course catalog with details such as course code, name, and credits.Administrative tools for adding, modifying, and removing courses.Real-time updates to ensure accurate representation of available courses.

3. Department Management:

Administration of academic departments, including the addition of new departments.Roster management to assign members to specific departments.Streamlined departmental activities for optimal organizational structure

4. Enrollment and Grade Management:

User-friendly enrollment procedures for students to register in courses.Grade management functionalities, including grade calculations and updates.Transparent and efficient processes for recording and managing student grades

BUSINESS RULES

Business rules in a database refer to specific guidelines or constraints that dictate how data should be stored, processed, and managed within the context of a particular database or system. These rules ensure that the data remains accurate, consistent, and aligned with the project objectives.

The below section shows the entities and business rules related to each:

01 STUDENT

- Each student must have a unique Student_ID, First name, Last name, Date of Birth, Phone number, Email address, Academic year, and GPA.
- Each student can have multiple grades.
- Each student can enroll in multiple classes.

02 COURSE

- Each course must have a unique Course_ID , Course name, Instructor_ID, prerequisite, and Credit hours.
- Each Course can be enrolled in by multiple students.

03 INSTRUCTOR

- Each Instructor must have a unique Instructor_ID, First name, Last name, Date of Birth, Phone number, Email address, Department_ID.
- Each Instructor can teach multiple courses.
- Each instructor must only be in one department.

04 DEPARTMENT

- Each department must have Department_ID, Department name, and Department head ID.
- Each department can have multiple instructors

05 GRADE

- Each grade must have a Grade_ID, Student_ID, Course_ID, grade.
- Each grade must only be for one student.

Business rules can cover a variety of aspects, including:

1. Data Integrity: Rules that maintain the accuracy and reliability of data. For example, enforcing unique constraints on primary keys to avoid duplicate records.

2. Data Validation: Rules that ensure the correctness of data values. This could involve checks on data types, ranges, or formats.

3. Entity Relationships: Rules defining how different entities (tables) in the database are related to each other. This includes specifying foreign key relationships.

4. Security Constraints: Rules governing who has access to specific data and under what conditions. This helps protect sensitive information.

5. Default Values: Rules specifying default values for certain attributes in the absence of user input.

6. Workflow Rules: Rules dictating how the database responds to certain events or triggers, often reflecting the business process. For instance, updating inventory levels when a sale is made.

7. Normalization Rules: As discussed earlier, rules for organizing data through normalization to minimize redundancy and improve data structure.

Database design

The organization of data according to a database model.

Database design can be generally defined as a collection of tasks or processes that enhance the designing, development, implementation, and maintenance of an enterprise data management system. Designing a proper database reduces the maintenance cost thereby improving data consistency and the cost-effective measures are greatly influenced in terms of disk storage space. The designer should follow the constraints and decide how the elements correlate and what kind of data must be stored.

The main objectives behind database designing are to produce physical and logical design models of the proposed database system. The logical model is primarily concentrated on the requirements of data and the considerations must be made in terms of monolithic considerations hence the stored physical data must be stored independent of the physical conditions. On the other hand, the physical database design model includes a translation of the logical design model of the database by keeping control of physical media using hardware resources and software systems such as (DBMS).

Why is Database Design important?

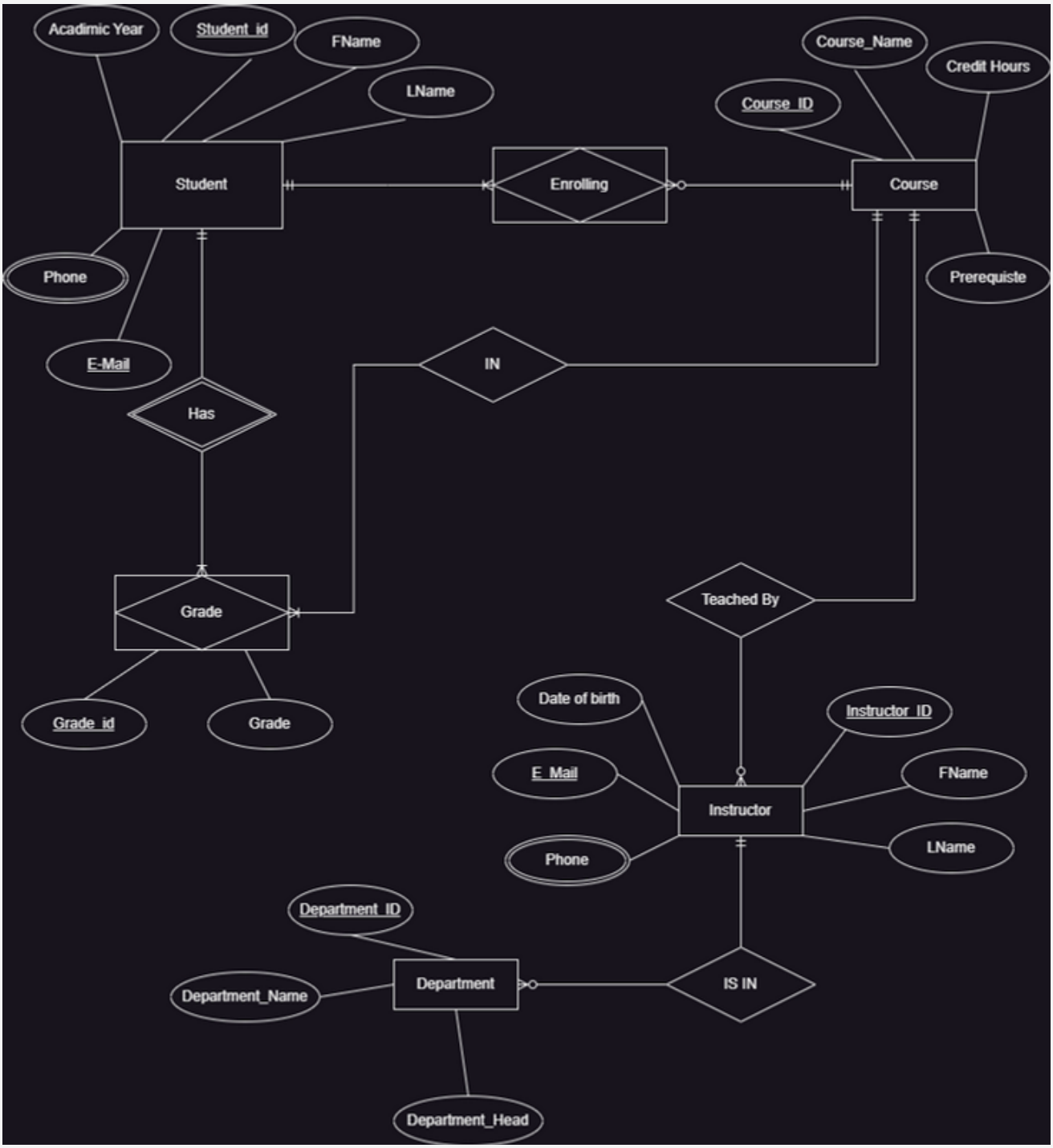
The important consideration that can be taken into account while emphasizing the importance of database design can be explained in terms of the following points given below.

- Database designs provide the blueprints of how the data is going to be stored in a system. A proper design of a database highly affects the overall performance of any application.
- The designing principles defined for a database give a clear idea of the behavior of any application and how the requests are processed.
- Another instance to emphasize the database design is that a proper database design meets all the requirements of users.
- Lastly, the processing time of an application is greatly reduced if the constraints of designing a highly efficient database are properly implemented.

ERD

The ERD illustrates the relationships between entities such as Student, Course and Department. Relationships include one-to-many associations, capturing the core structure of the university system.

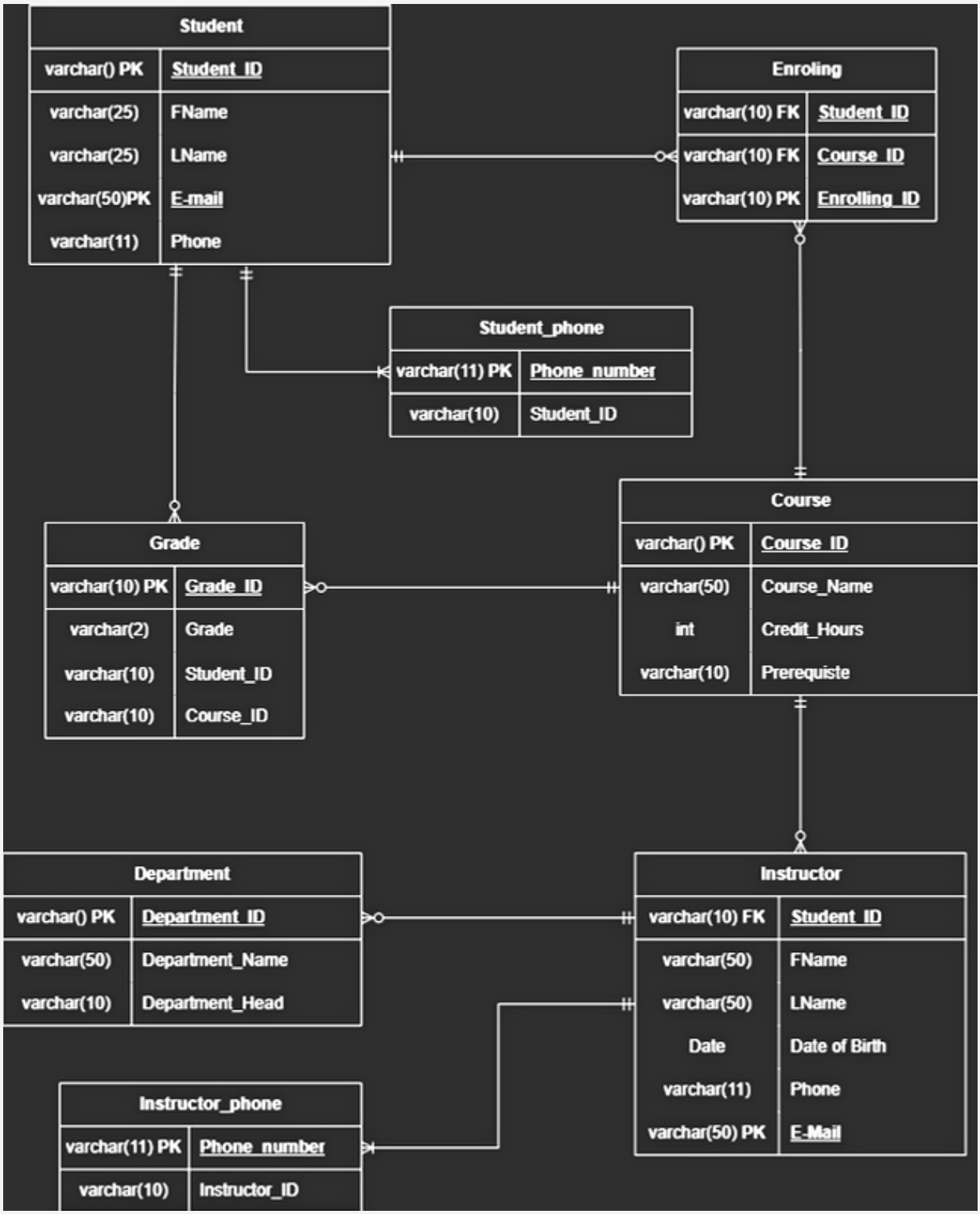
The ERD that describes the system we've created :



Database Schema

The database schema is organized into tables: Student, Course, Faculty, Department, and Enrollment. Each table has defined attributes, such as student ID, course code, etc. Normalization techniques have been applied to ensure data integrity.

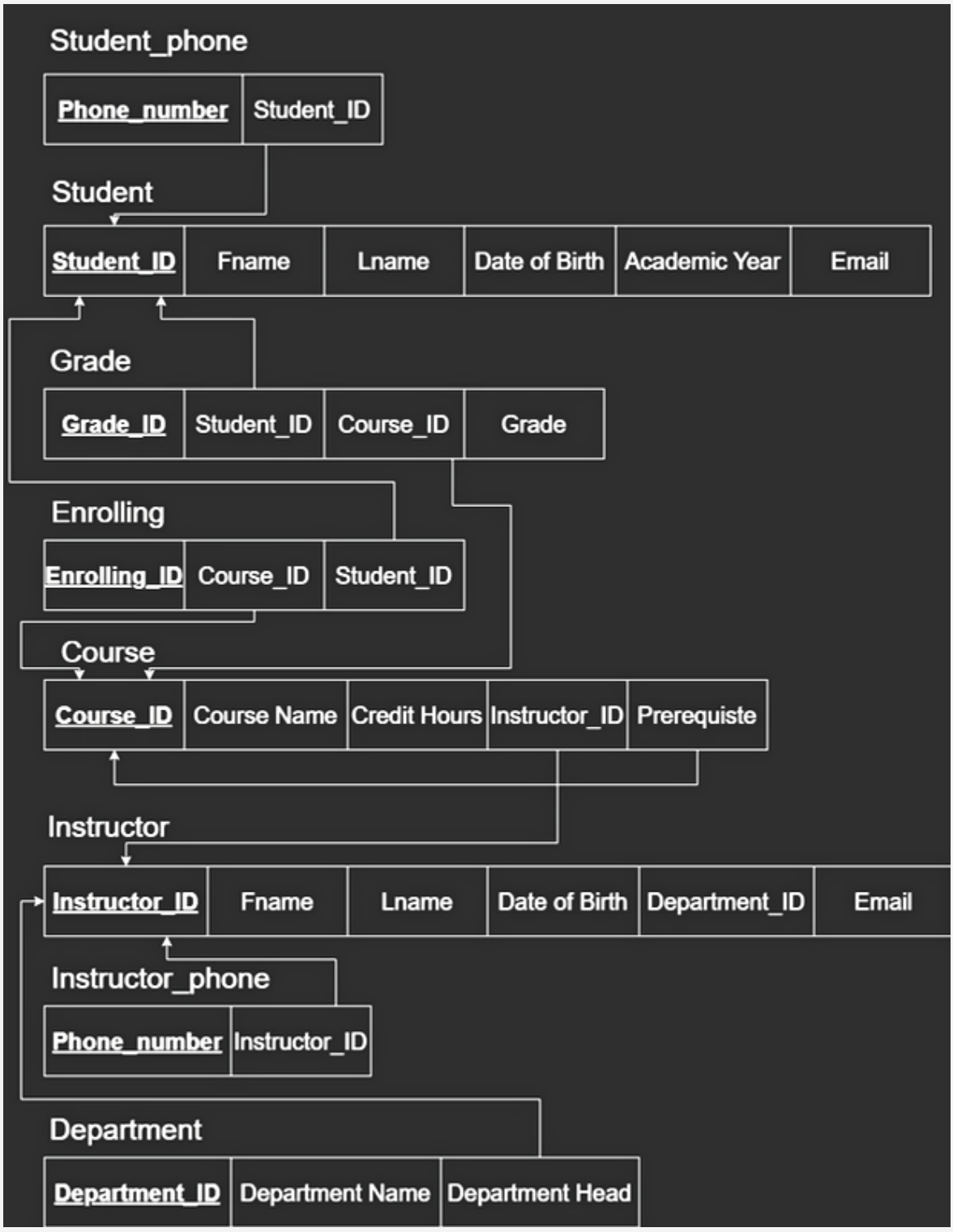
Logical schema describes the system we've created :



Database Schema

The database schema is organized into tables: Student, Course, Faculty, Department, and Enrollment. Each table has defined attributes, such as student ID, courseID, etc. Normalization techniques have been applied to ensure data integrity.

Relational schema that describes the system we've created :



What the ERD & the schemas tell us:

- Students, instructors, departments, courses, and grades are the main entities of our database.
  - **The primary keys are unique identifiers for each record in a table.** Here are the primary keys for the main entities in the project:

    1. Student Table:
    - StudentID

    2. Course Table:
    - CourseID

    3. Grade Table:
    - GradeID

    4. Department Table:
    - DepartmentID

    5. Enrollment Table:
    - Composite Primary Key: StudentID, CourseID.

    6. Instructor Table:
    - InstructorID

- **Foreign keys establish relationships between tables.** Here are the foreign keys that link different tables within the project**:**

  1. Enrollment Table:
  -StudentID
  -CourseID

  These foreign keys in the Enrollment Table establish relationships with the Student and Course tables, linking enrolled students and their respective courses.

  2. Instructor Table:

  - DepartmentID:
  Represents the department to which the instructor is assigned.
  CourseID:
  If there is a direct link between instructors and specific courses they teach.

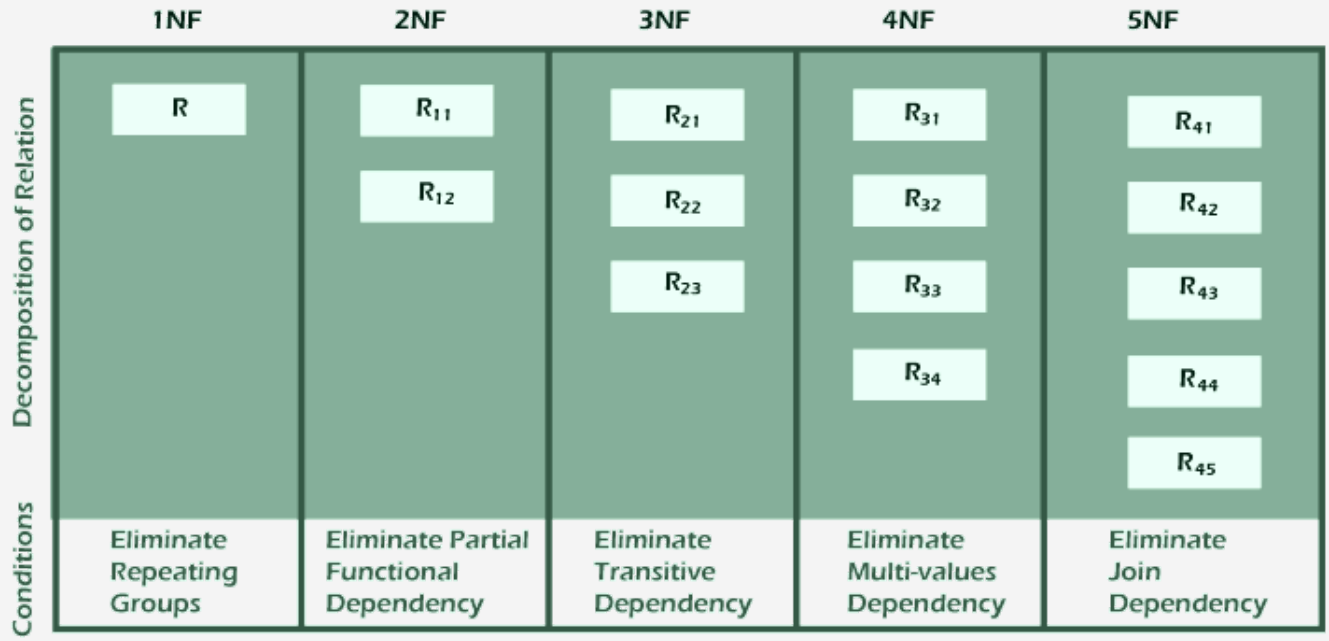
NORMALIZATION

What is Normalization?

- Normalization is the process of organizing the data in the database.
- Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies.
- Normalization divides the larger table into smaller ones and links them using relationships.
- The normal form is used to reduce redundancy from the database table.

Why do we need Normalization?

The main reason for normalizing the relations is to remove these anomalies. Failure to eliminate anomalies leads to data redundancy and can cause data integrity and other problems as the database grows. Normalization consists of a series of guidelines that help to guide you in creating a good database structure.



First Normalization Form:

In the context of the University System Database Project, let's consider the concept of the first normalization form (1NF). The main goal of 1NF is to ensure that each column in a table contains only atomic (indivisible) values, and that there are no repeating groups of columns. Here's how 1NF might be applied to some of the tables in our project:

Example:

Table before 1NF

Student_ID	Name	Courses
2001	Mohamed Nasef	Machine Learning, Data Science, Data Tools
1005	Marwa Belal	Data Science, linear Algebra

Table after 1NF

Student_ID	Name	Course_Name
2001	Mohamed Nasef	Machine Learning
2001	Mohamed Nasef	Data Science
2001	Mohamed Nasef	Data Tools
1005	Marwa Belal	Data Science
1005	Marwa Belal	linear Algebra

By doing this, we ensure that each cell in the table contains atomic values, and there are no repeating groups. This restructuring allows for more efficient querying and avoids data redundancy.

Second Normalization Form:

The second Normal Form (2NF) is based on the concept of fully functional dependency. The second Normal Form applies to relations with composite keys, that is, relations with a primary key composed of two or more attributes. A relation with a single-attribute primary key is automatically in at least 2NF. A relation that is not in 2NF may suffer from the update anomalies. To be in the second normal form, a relation must be in the first normal form and the relation must not contain any partial dependency. A relation is in 2NF if it has No Partial Dependency Here's how 2NF might be applied to some of the tables in our project:

Example:

Table before 2NF

Student_ID	Course_ID	Course_Name	credit_Hours
2001	1202	Machine Learning	3
1005	3200	Data Science	3

In this case, the CourseName and Credits are functionally dependent on the CourseCode but not on the entire primary key (StudentID, CourseCode). This is a partial dependency.

To achieve 2NF, we break the table into **two separate tables**:

Student_ID	Course_ID
2001	1202
1005	3200

Course_ID	Course_Name	credit_Hours
1202	Machine Learning	3
3200	Data Science	3

Now, CourseName and Credits are directly related to the CourseCode in the CourseDetails table, avoiding partial dependencies and achieving 2NF.

Third Normalization Form:

To achieve the Third Normal Form (3NF) in database design, you generally follow these steps:

1. Ensure 1NF:

- Make sure your table is in the First Normal Form (1NF).
- Each table cell should contain a single, atomic value.
- All entries in a column should be of the same data type.

2. Ensure 2NF:

- Make sure your table is in the Second Normal Form (2NF).
- Identify and remove partial dependencies. All non-prime attributes (attributes not part of the primary key) should be fully functionally dependent on the entire primary key.

3. Remove Transitive Dependencies:

- Identify and eliminate transitive dependencies. In other words, ensure that there are no non-prime attributes that depend on other non-prime attributes.

4. Create Separate Tables:

- If transitive dependencies exist, create separate tables for the related information.
- Move the transitive-dependent attributes to a new table, and establish relationships using foreign keys.

5. Define Primary and Foreign Keys:

- Ensure that each table has a primary key.
- Establish foreign key relationships between tables to maintain referential integrity.

By following these steps, you can systematically transform a database design into the Third Normal Form, reducing redundancy and ensuring that data is stored in a way that promotes integrity and flexibility.

System Architecture

How our system was implemented:

The database is implemented using a relational database management system **MySQL**.

The backend is developed in **Php**, with a front-end interface built using **React with typescript**. The following link takes us to the interface we've made to our database:
https://universitydb.rf.gd

Conclusion.

This project not only provides a foundation for effective data management within a university system but also serves as an educational resource for understanding the importance of normalization in database design. As we conclude this project, we acknowledge the ongoing importance of maintaining a dynamic and responsive database structure to accommodate the evolving needs of educational institutions.

In future iterations, considerations may be given to additional features, optimizations, and potential expansion of the database to encompass a broader range of functionalities. Nevertheless, the current design stands as a testament to the principles of normalization, ensuring data accuracy, reducing redundancy, and providing a solid foundation for efficient data retrieval and manipulation within a university system.