**Univ Case Study Documentation**

**By Mostafa Mohamed**

**iti DM44 Alex**

**Table of Content**

1. **Introduction**
2. **Database Design**
3. **SQL Implementation**
4. **PLSQL Implementation**
5. **Automation Scripts**
6. **Java Application Development**
7. **Integration and Reporting**
8. **Conclusion**

**Introduction**

In the contemporary landscape of educational institutions, effective management of data plays a pivotal role in ensuring streamlined operations and facilitating informed decision-making processes. The implementation of a robust data management system is imperative for universities to navigate through the complexities of student information, course structures, departmental functionalities, and academic performance metrics. The following sections provide an overview of the objectives, technologies employed, and the holistic approach undertaken in the development of a comprehensive data management system for a university setting.

**1. Project Overview**

The objective of this project is to design and implement a comprehensive data management system tailored specifically for the intricate needs of a university environment. Leveraging a plethora of technologies including SQL, PLSQL, Advanced PLSQL, Red Hat, Bash scripting, Java SE, and object-oriented programming (OOP) principles, the project aims to address various facets encompassing database design, SQL and PLSQL implementation, automation scripts, Java application development, and seamless integration between disparate components.

**2. Objectives**

The primary objective of the project is to develop a cohesive data management system that empowers university departments, courses and students alike with efficient access to pertinent information while ensuring data integrity, security, and scalability. By adhering to industry best practices and leveraging cutting-edge technologies, the system endeavors to streamline administrative processes, enhance academic operations, and foster a conducive learning environment.

**3. Technologies Used**

The project harnesses the capabilities of several key technologies to realize its objectives:

* Database design, ERD preparation, mapping and normalization.
* SQL, PLSQL, and Advanced PLSQL for database management and manipulation.
* Bash scripting for automation of routine tasks and system monitoring using git bash on windows.
* Java SE and OOP principles for the development of a robust and user-friendly application interface.

**Database Design**

In the foundational phase of the project, meticulous attention was devoted to designing a robust and scalable database schema tailored to the specific needs of a university environment. The database schema serves as the backbone of the data management system, orchestrating the organization, storage, and retrieval of critical information pertaining to students, courses, departments, and academic performance metrics.

**1. Relational Database Schema**

The relational database schema was meticulously crafted to encapsulate the essential entities and their interrelationships within the university ecosystem. The schema adheres to the principles of normalization to ensure data integrity, minimize redundancy, and optimize query performance. Entities such as students, courses, departments, and grades were identified as the cornerstone elements necessitating careful delineation within the schema.

**2. Normalization Process**

Normalization, a fundamental tenet of database design, was rigorously applied to eliminate data anomalies and redundancy while preserving the integrity of the database structure. By decomposing the database schema into progressively normalized forms, redundancies were mitigated, and dependencies were systematically organized to facilitate efficient data manipulation and maintenance.

And here’s the final result :

STUDENT(student\_id(pk), first\_name, last\_name, street, city, state,gender, email, birthdate, com\_grade(drived),dept\_id(fk))

STUDENT-PHONE(student\_id(fk,cpk),phone(cpk))

DEPARTMENT(dept\_id(pk), dept\_name)

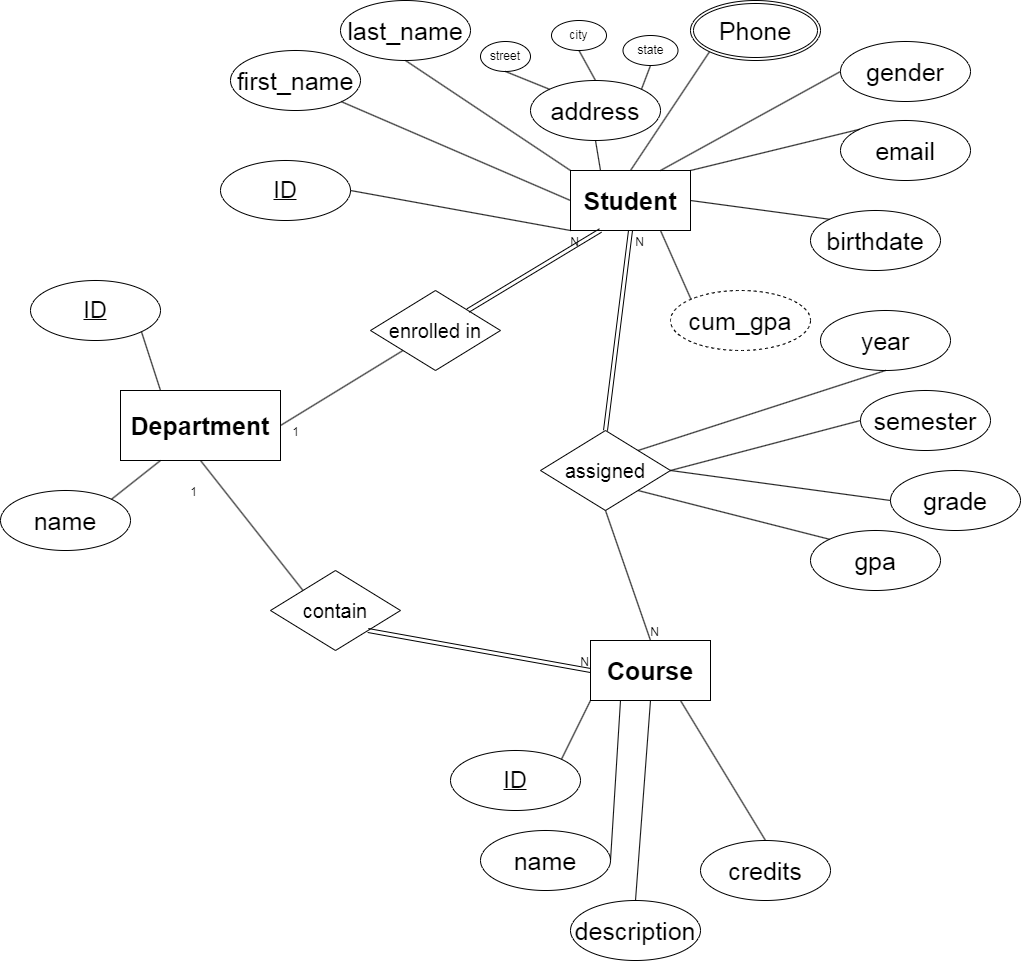
COURSE(course\_id(pk), course\_name, credits, description, dept\_id(fk))

ASSIGNED(student\_id(fk,cpk), course\_id(fk,cpk), year(cpk), semester, gpa)

**3. Database Design Documentation**

Attached herewith is the Entity-Relationship Diagram (ERD), providing a visual representation of the relational schema delineated during the design phase. Additionally, the finalized database schema in SQL format is included, elucidating the structure and attributes of each entity along with their respective relationships and constraints. This comprehensive documentation serves as a blueprint for the subsequent stages of SQL implementation, ensuring a coherent and cohesive realization of the envisioned data management system.

And here’s the ERD of the database:



**SQL Implementation**

The SQL implementation phase constitutes a pivotal stage in the development of the university data management system, wherein the meticulously crafted database schema is translated into executable SQL scripts for database creation, population, and validation. This section delineates the procedural intricacies involved in each facet of SQL implementation, ensuring the seamless instantiation and validation of the relational database schema.

**1. SQL Scripts for Database Creation**

A series of meticulously crafted SQL scripts were formulated to instantiate the relational database schema delineated during the design phase. These scripts encapsulate the creation of tables, definition of constraints, and establishment of relationships, adhering to the normalized structure prescribed by the database design. The SQL scripts serve as the foundational blueprint for the subsequent stages of data population and validation, laying the groundwork for the realization of a robust and scalable database infrastructure.

**2. Data Population**

Following the creation of the database schema, the next imperative step entailed the population of the database with representative sample data sets. The data population process involved the systematic insertion of records into the respective tables, ensuring comprehensive coverage of student profiles, course offerings, departmental affiliations, and academic performance metrics. The sample data sets were meticulously curated to encompass diverse scenarios and edge cases, facilitating comprehensive testing and validation of the database functionality.

Involved 3 main departments, 10 student in each, 20 course in each and the rest of related data.

**3. Testing and Validation**

The culmination of SQL implementation necessitated rigorous testing and validation procedures to ascertain the correctness, integrity, and coherence of the instantiated database schema. A battery of test cases spanning diverse usage scenarios and edge conditions were meticulously devised and executed to evaluate the efficacy of the database infrastructure. The testing regimen encompassed verification of data integrity constraints, assessment of referential integrity, and validation of query performance metrics to ensure optimal database responsiveness and reliability.

**PLSQL Implementation**

The PLSQL implementation phase encompasses the development and deployment of triggers and procedures aimed at facilitating data manipulation and enforcing data integrity constraints within the university data management system. This section elucidates the functionality and rationale behind each PLSQL construct, elucidating their pivotal role in ensuring the coherence and reliability of the database infrastructure.

**1. Triggers for Cumulative GPA Calculation**

**T\_CUM\_GPA\_DELETE Trigger:**

* Executes after a record is deleted from the **assigned** table.
* Calculates the cumulative GPA (**cum\_gpa**) for the affected student after the deletion.
* Updates the **cum\_gpa** attribute in the **student** table for the respective student.

**T\_CUM\_GPA\_UPDATE Trigger:**

* Executes after a record in the **assigned** table is updated.
* Recalculates the cumulative GPA (**cum\_gpa**) for the affected student if the **gpa** attribute is modified.
* Updates the **cum\_gpa** attribute in the **student** table for the respective student.

**T\_CUM\_GPA\_INSERT Trigger:**

* Executes after a record is inserted into the **assigned** table.
* Calculates the cumulative GPA (**cum\_gpa**) for the affected student after the insertion.
* Updates the **cum\_gpa** attribute in the **student** table for the respective student.
* Takes in consideration when **cum\_gpa** calculation only the last trial to pass a course.

**2. Triggers for Cascading Deletion**

**delete\_cascade\_department Trigger:**

* Executes before a record is deleted from the **DEPARTMENT** table.
* Cascades deletion to associated records in the **ASSIGNED**, **STUDENT**, and **COURSE** tables based on the department being deleted.

**delete\_cascade\_course Trigger:**

* Executes before a record is deleted from the **COURSE** table.
* Cascades deletion to associated records in the **ASSIGNED** table based on the course being deleted.

**delete\_cascade\_student Trigger:**

* Executes before a record is deleted from the **STUDENT** table.
* Cascades deletion to associated records in the **ASSIGNED** and **STUDENT\_PHONE** tables based on the student being deleted.

**Automation Scripts**

Automation scripts play a pivotal role in enhancing operational efficiency by automating routine tasks and facilitating proactive monitoring of critical system parameters. This section provides an overview and explanation of two automation scripts, namely the Database Backup Script and the Disk Space Monitoring Script, which are designed to run periodically via cron jobs to ensure the robustness and reliability of the university data management system.

**1. Bash Scripts Overview**

Bash scripting serves as the cornerstone of automation within the university data management system, enabling the execution of complex tasks with minimal human intervention. These scripts leverage the inherent capabilities of Bash to interact with system utilities, execute commands, and manipulate data, thereby streamlining administrative workflows and enhancing system resilience.

**2. Database Backup Script**

The Database Backup Script is designed to automate the process of backing up the university database, ensuring data integrity and facilitating disaster recovery. Key features of the script include:

* **Database Connection Details:** Specifies the connection parameters (host, port, username, password) required to access the university database.
* **Backup Directory:** Defines the directory where database backups are stored.
* **Backup Process:** Utilizes the **mysqldump** utility to create a backup of the university database, generating a timestamped backup file in SQL format within the designated backup directory.
* **Backup Verification:** Checks the success status of the backup process and provides appropriate feedback on completion or failure.
* **Periodic Backup:** Configured to run periodically via cron jobs, ensuring continuous backup every specified period of time.

**3. Disk Space Monitoring Script**

The Disk Space Monitoring Script is instrumental in proactively monitoring disk space usage on the server hosting the university data management system, thereby preempting potential issues arising from disk space exhaustion. Key features of the script include:

* **Threshold Definition:** Sets a predefined threshold for disk space usage, beyond which alerts are triggered.
* **Disk Space Query:** Queries the system to retrieve current disk space usage and calculates the percentage occupied.
* **Alert Generation:** Generates alerts in the form of log entries, indicating whether disk space is within acceptable limits or if it has exceeded the defined threshold.
* **Periodic Execution:** Configured to run periodically via cron jobs, ensuring continuous monitoring of disk space usage and timely detection of anomalies.

By incorporating these automation scripts into the university data management system, administrators can streamline critical operations, mitigate potential risks, and uphold the integrity and availability of essential data assets.

**Java Application Development**

The Java application developed for the university data management system adheres to the Model-View-Controller (MVC) architectural pattern, ensuring modularity, maintainability, and separation of concerns. The application comprises various components, including models (DTOs), controllers, and views, each serving distinct roles in facilitating data interaction and user interface design.

**Model (Data Transfer Objects - DTOs):**

* The application defines several Data Transfer Objects (DTOs) representing entities within the university system, including Student, Department, Phone, Course, Assigned, and CoursesReport.
* These DTOs encapsulate data attributes and provide a structured representation for interacting with the underlying database.

**Controller:**

* Individual controllers are implemented for each scene in the application, such as StudentController, DepartmentController, PhoneController, CourseController, AssignedController, and CoursesReportController.
* Controllers orchestrate user actions, such as button clicks and text input, and coordinate with the Model and View components to handle data manipulation and validation.
* Data validation logic is implemented in the controllers to ensure that user inputs adhere to specified rules and constraints, such as non-empty fields, valid IDs, valid data types and existence of foreign keys.
* Alerts are used to notify users of validation errors or the success/failure of operations.

**View:**

* The user interface is composed of distinct scenes corresponding to different entities, including Student, Department, Phone, Course, Assigned, and CoursesReport.
* Each scene is equipped with intuitive controls, such as buttons, text fields, and table views, facilitating data interaction and manipulation.
* Color-coded labels provide visual cues to users, distinguishing primary key fields (dark yellow) which is mandatory for data retrieval, secondary fields (light yellow) which aren’t necessarily CPK but can be used for data retrieval, and regular data (black), aiding in data comprehension and input validation, clarifying data (white) not used only to display clarifying data form another tables as getting department name along with department id.
* Functionality for data retrieval, insertion, updating, and deletion is seamlessly integrated into each scene, promoting efficient data management and user engagement.
* The Phone scene supports the display and management of multiple phone numbers per student, enhancing flexibility and usability.

**DAO Implementation:**

* For each entity, such as Student, Department, Course, etc., a corresponding DAO has been meticulously crafted to handle database communication.
* These DAOs encapsulate database-specific logic and abstract away the complexities of SQL queries and database interactions from the rest of the application.
* Each DAO provides methods for inserting, updating, retrieving, and deleting entity instances, adhering to the specific requirements and constraints of the university data management system.
* To ensure data integrity, DAOs implement validation checks to verify the existence of entities before performing insertion or update operations. For instance, when adding a new student or department, the DAO validates whether the entity already exists in the database.

**Continuous Database Connection:**

* The Java application establishes and maintains a continuous connection to the database throughout its lifecycle, ensuring uninterrupted data access and interaction.
* By employing robust connection management mechanisms, the application optimizes resource utilization and minimizes latency, thereby enhancing overall performance and responsiveness.

**Functionality:**

* Users interact with the application by populating text fields and engaging with graphical elements to perform CRUD operations (Create, Read, Update, Delete) on university data entities.
* Data retrieval necessitates inputting primary key fields, while data manipulation operations require complete field entries and adherence to specified data constraints.
* Clearing functionality enables users to reset text fields, enhancing usability and facilitating iterative data entry.

**Courses report:**

* The Courses Report page provides a comprehensive overview of courses, detailing department affiliation, assigned year, enrolled students, and average GPA, offering valuable insights into academic performance and course enrollment trends.

The Java application embodies principles of modularity, usability, and reliability, providing a robust platform for managing university data and empowering users with efficient data manipulation capabilities.

**Conclusion**

In conclusion, the comprehensive data management system developed for the university represents a culmination of meticulous planning, meticulous implementation, and diligent execution. By leveraging a combination of SQL, PLSQL, Bash scripting, Java SE, and Object-Oriented Programming (OOP) principles, the system embodies robustness, scalability, and flexibility, catering to the diverse needs of modern educational institutions.

From the meticulous database design to the seamless integration of automation scripts and Java application development, every aspect of the project has been meticulously crafted to ensure efficiency, reliability, and usability. The incorporation of Model-View-Controller (MVC) architecture and Data Access Objects (DAOs) facilitates modularity, maintainability, and extensibility, laying a solid foundation for future enhancements and expansions.

The project's success is not only evident in the functional prowess of the data management system but also in the attention to detail paid to documentation, presentation, and user experience. Through concise documentation, engaging presentations, and user-friendly interfaces, the project aims to empower users with the tools and insights necessary for effective data management and decision-making.

In essence, the data management system for the university exemplifies the synergy of technical expertise, strategic planning, and creative problem-solving, embodying the ethos of excellence and innovation in educational technology. As the project reaches its culmination, it stands as a testament to the collaborative efforts, dedication, and ingenuity of all those involved, poised to make a significant impact in the realm of university administration and data management.