**UNIVERSITY INSTITUTE OF COMPUTING**

**CASE STUDY REPORT**

**ON**

**PARTICULAR CASE STUDY**

Program Name: BCA

Subject Name/Code: Database Management System (23CAT-251)

**Submitted by: Submitted to:**

**Name:** Kanak Panwar **Name: Mr. Arvinder Singh**

**UID:23BCA**10724 **Designation: Professor**

**Section:4’a’**

ABSTRACT

* Introduction:

📘 **Introduction: University Student Information System– Database Project**

A University Student Information System is a comprehensive application built using a Database Management System (DBMS) to store, manage, and retrieve student-related data efficiently. In a university environment, managing information such as student admissions, course registrations, grades, attendance, faculty schedules, and exam results becomes complex due to the volume and variety of data. A DBMS plays a crucial role in handling this complexity by providing a structured and secure way to manage large amounts of interconnected information.

The system enables real-time access, data consistency, reduced redundancy, and enhanced data integrity, which are essential for the smooth functioning of academic institutions. With a DBMS, administrators can generate reports, track student performance, and ensure proper coordination between departments. It also helps students access their academic records and schedules through a user-friendly interface, promoting transparency and ease of use.

In essence, a University Student Information System powered by a DBMS provides a robust foundation for academic data management, improving overall operational efficiency and decision-making within the institution.

* Technique:

**Here’s a well-structured "Technique" section for your University Student Information System using DBMS:**

The **University Student Information System** is developed using the principles and tools of **Database Management Systems (DBMS)**. The system is designed with a focus on **data normalization**, **efficient querying**, and **relational database design** to ensure accuracy, performance, and scalability.

#### **Key Techniques Used:**

1. **Relational Database Model:**
   * Data is stored in tables (relations) with predefined relationships (using foreign keys).
   * Examples: Students, Courses, Faculty, Enrolments, Results.
2. **Entity-Relationship (ER) Modelling:**
   * Before designing the database, an ER diagram is created to identify entities, attributes, and relationships.
   * This model helps in designing a normalized and logically sound schema.
3. **Normalization:**
   * The database is normalized (1NF, 2NF, 3NF) to eliminate redundancy and improve data integrity.
   * Ensures that updates, deletions, and insertions are performed without anomalies.
4. **SQL (Structured Query Language):**
   * Used for creating tables (CREATE), inserting data (INSERT), retrieving data (SELECT), updating (UPDATE), and deleting (DELETE) records.
   * Advanced SQL techniques like joins, subqueries, and stored procedures are used for complex operations.
5. **Data Integrity and Constraints:**
   * **Primary keys** ensure each record is unique.
   * **Foreign keys** maintain relationships between tables.
   * **Constraints** like NOT NULL, CHECK, and UNIQUE enforce valid data entry.
6. **User Access Control:**
   * Different roles (e.g., admin, student, faculty) are assigned different access rights using DBMS features like GRANT and REVOKE.
7. **Backup and Recovery:**
   * Regular data backups and recovery plans ensure that student records are safe from data loss.

* System Configuration:

Here’s a solid **"System Configuration"** section for your **University Student Information System using DBMS**:

### **System Configuration:**

The development and deployment of the **University Student Information System** require both hardware and software components to ensure smooth performance, reliability, and scalability. Below is the recommended configuration:

#### **1. Hardware Requirements:**

| Component | Specification |
| --- | --- |
| **Processor** | Intel Core i5 or higher |
| **RAM** | Minimum 8 GB |
| **Hard Disk** | 500 GB or higher |
| **Monitor** | 15” or larger (HD resolution) |
| **Network** | High-speed internet for multi-user access |
| **Backup Storage** | External HDD / Cloud Storage |

#### **2. Software Requirements:**

| Software Component | Specification |
| --- | --- |
| **Operating System** | Windows 10 / 11, Linux (Ubuntu), or macOS |
| **Database Management System** | MySQL / PostgreSQL / Oracle / MS SQL Server |
| **Front-End (Optional)** | HTML, CSS, JavaScript (for web interface) |
| **Back-End (Optional)** | PHP / Python / Java / Node.js |
| **Development Tools** | VS Code, MySQL Workbench / gamin |
| **Web Server (if web-based)** | Apache / Nginx / XAMPP |
| **Security Tools** | Antivirus, Firewall, Role-based Access Control |

#### **3. Network Configuration (for multi-user access):**

* **Server Setup:** Centralized DBMS server connected to client systems via LAN/Wi-Fi.
* **Client Systems:** Student, faculty, and admin systems connect through a web app or desktop interface.
* **Authentication:** User login with roles (admin, student, faculty).
* **Data Access:** Controlled by DBMS permissions to ensure secure access.
* To successfully develop and run the University Student Information System using a Database Management System (DBMS), a suitable combination of hardware and software resources is essential.
* On the hardware side, a system with at least an Intel Core i5 processor, 8 GB of RAM, and a minimum of 500 GB hard disk storage is recommended to ensure smooth operation. A standard HD display monitor (15 inches or larger) is sufficient for development and administrative use. For network-based access, a reliable high-speed internet connection is necessary. Additionally, backup storage—such as an external hard drive or a secure cloud solution—should be in place to safeguard critical data.
* On the software side, the system can run on any major operating system, such as Windows 10/11, Linux (like Ubuntu), or macOS. For the DBMS, popular options include MySQL, PostgreSQL, Oracle, or Microsoft SQL Server, depending on the institution’s needs. If a front-end interface is used, technologies such as HTML, CSS, and JavaScript can be employed, while back-end development can be handled using PHP, Python, Java, or Node.js. Development tools like Visual Studio Code and database clients like MySQL Workbench or gamin support the coding and database management process.
* If the system is deployed over a network for access by students, faculty, and administrators, a basic client-server network setup is required. The server hosts the database, and authorized client systems connect via a secure network (LAN or Wi-Fi). Access control is enforced through user authentication and role-based permissions to ensure data privacy and integrity.
* Security tools such as firewalls, antivirus programs, and database-level access restrictions are also implemented to protect against unauthorized access and potential data breaches.
* INPUT:

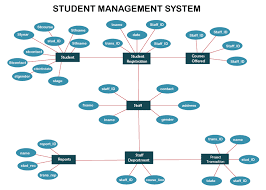
The **University Student Information System** takes various inputs from students, faculty, and administrative staff to manage academic and administrative processes. These inputs are collected through user-friendly forms, web portals, or desktop applications and stored in a structured format within the database.

#### **Key Inputs Include:**

1. **Student Information:**
   * Full name, student ID, date of birth, contact number, email address
   * Address, nationality, guardian details
   * Course selection during admission
2. **Faculty Information:**
   * Faculty ID, name, department, designation
   * Subjects taught, contact details, qualifications
3. **Course and Subject Data:**
   * Course ID, course name, duration, credit hours
   * Subject codes, subject names, faculty assigned
4. **Enrolment Details:**
   * Student ID and selected courses/subjects
   * Enrolment date and semester
5. **Exam and Grades:**
   * Exam schedules, subject-wise marks
   * Grade entry by faculty
6. **Attendance Records:**
   * Daily attendance input for each student by subject
   * Absentees marked and stored
7. **Library and Hostel Information (if applicable):**
   * Book issue/return entries
   * Hostel room allocation details
8. **User Login Credentials:**
   * Usernames and passwords for different roles (admin, student, faculty)

All of these inputs are validated before being stored in the database to ensure **accuracy, completeness, and integrity** of the data.

* ER DIAGRAM:



* TABLE REALTION:

In the **University Student Information System**, the database is designed using a **relational model**, where data is stored in interrelated tables. These tables are connected using **primary keys** and **foreign keys** to maintain referential integrity and ensure efficient querying.

#### **Key Tables and Their Relationships:**

1. **Students Table**
   * student (Primary Key)
   * Related to: Enrolments, Results, Attendance, Login, Library
2. **Faculty Table**
   * faculties (Primary Key)
   * Related to: Courses, Subjects, Results, Attendance
3. **Courses Table**
   * coursed (Primary Key)
   * Related to: Students, Subjects, Enrolments
4. **Subjects Table**
   * subject (Primary Key)
   * Related to: Courses (via coursed)
   * Related to: Faculty, Results, Attendance
5. **Enrolments Table**
   * enrolment (Primary Key)
   * student (Foreign Key from Students)
   * coursed (Foreign Key from Courses)
6. **Results Table**
   * result (Primary Key)
   * student (Foreign Key from Students)
   * subject (Foreign Key from Subjects)
   * faculties (Foreign Key from Faculty)
7. **Attendance Table**
   * attendance (Primary Key)
   * student (Foreign Key from Students)
   * subject (Foreign Key from Subjects)
   * faculties (Foreign Key from Faculty)
8. **Login Table**
   * lucinid (Primary Key)
   * username, password, role
   * student or faculties as Foreign Key (based on user type)
9. **Library Table** (if included)
   * transactinide (Primary Key)
   * student (Foreign Key from Students)
   * bookie, issue date, return date

* TABULAR FORMAT:

### **1. Students Table**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| student | INT (Primary Key) | Unique identifier for each student |
| first name | VARCHAR | Student's first name |
| last-named | VARCHAR | Student's last name |
| dob | DATE | Date of birth |
| email | VARCHAR | Student's email address |
| phone number | VARCHAR | Student's contact number |
| address | TEXT | Residential address |
| nationality | VARCHAR | Student's nationality |

### **2. Faculty Table**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| faculties | INT (Primary Key) | Unique identifier for each faculty member |
| first name | VARCHAR | Faculty's first name |
| last-named | VARCHAR | Faculty's last name |
| department | VARCHAR | Faculty's department (e.g., Computer Science) |
| email | VARCHAR | Faculty's email address |
| phone number | VARCHAR | Faculty's contact number |

### **3. Courses Table**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| coursed | INT (Primary Key) | Unique identifier for each course |
| course name | VARCHAR | Name of the course |
| duration | INT | Duration of the course (in semesters) |
| credit hours | INT | Credit hours for the course |

### **4. Subjects Table**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| subject | INT (Primary Key) | Unique identifier for each subject |
| subject name | VARCHAR | Name of the subject |
| coursed | INT (Foreign Key) | Links to Courses.course\_id |
| faculty\_id | INT (Foreign Key) | Links to Faculty.faculty\_id (Faculty teaching) |

### **5. Enrollments Table**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| enrollment\_id | INT (Primary Key) | Unique identifier for each enrollment |
| student\_id | INT (Foreign Key) | Links to Students.student\_id |
| course\_id | INT (Foreign Key) | Links to Courses.course\_id |

### **6. Results Table**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| result\_id | INT (Primary Key) | Unique identifier for each result |
| student\_id | INT (Foreign Key) | Links to Students.student\_id |
| subject\_id | INT (Foreign Key) | Links to Subjects.subject\_id |
| faculty\_id | INT (Foreign Key) | Links to Faculty.faculty\_id (Faculty who evaluates) |
| marks | INT | Marks obtained by the student in the subject |
| grade | VARCHAR | Grade awarded to the student |

### **7. Attendance Table**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| attendance\_id | INT (Primary Key) | Unique identifier for each attendance record |
| student\_id | INT (Foreign Key) | Links to Students.student\_id |
| subject\_id | INT (Foreign Key) | Links to Subjects.subject\_id |
| faculty\_id | INT (Foreign Key) | Links to Faculty.faculty\_id |
| attendance\_date | DATE | Date of the class session |
| status | VARCHAR | Attendance status (Present/Absent) |

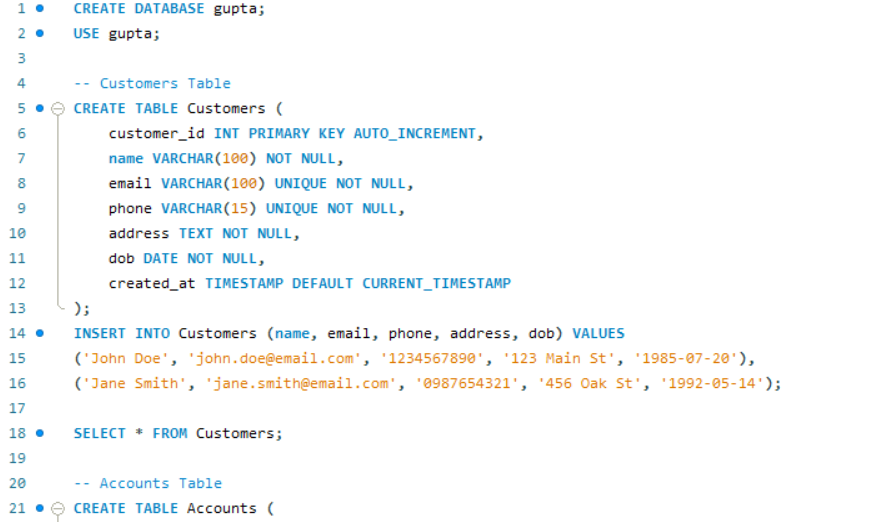
### **8. Login Table**

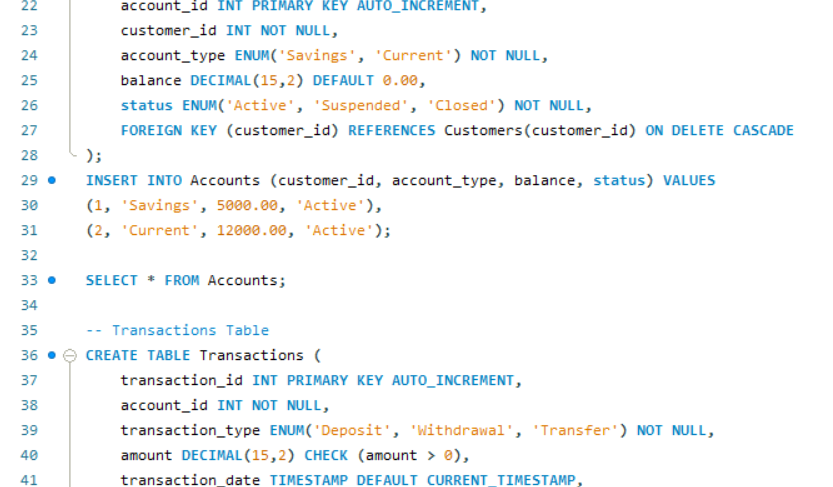
| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| login\_id | INT (Primary Key) | Unique identifier for each login record |
| username | VARCHAR | Username for the user (student/faculty) |
| password | VARCHAR | Password for the login account |
| role | VARCHAR | Role of the user (Admin/Student/Faculty) |
| student\_id | INT (Foreign Key) | Links to Students.student\_id (for students) |
| faculty\_id | INT (Foreign Key) | Links to Faculty.faculty\_id (for faculty members) |

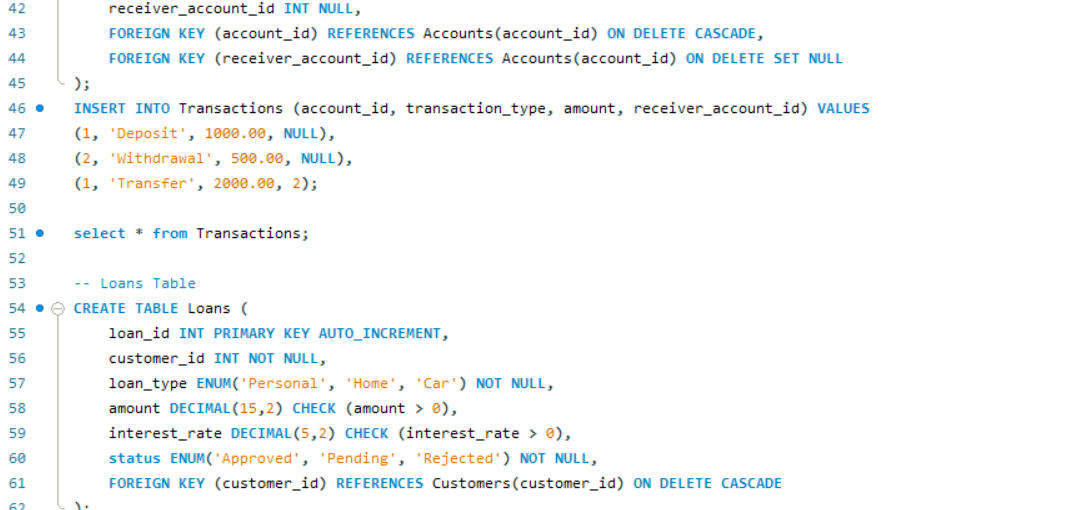
### **9. Library Table**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| transaction\_id | INT (Primary Key) | Unique identifier for each transaction |
| student\_id | INT (Foreign Key) | Links to Students.student\_id |
| book\_id | INT | Identifier for the book being issued |
| issue\_date | DATE | Date the book was issued |
| return\_date | DATE | Date the book is returned |

* TABLE CREATION:

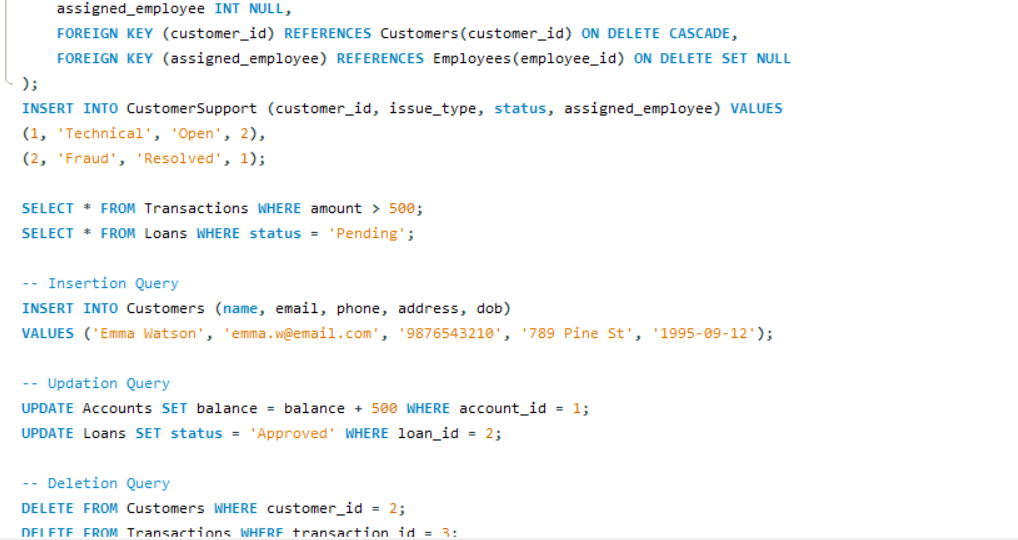


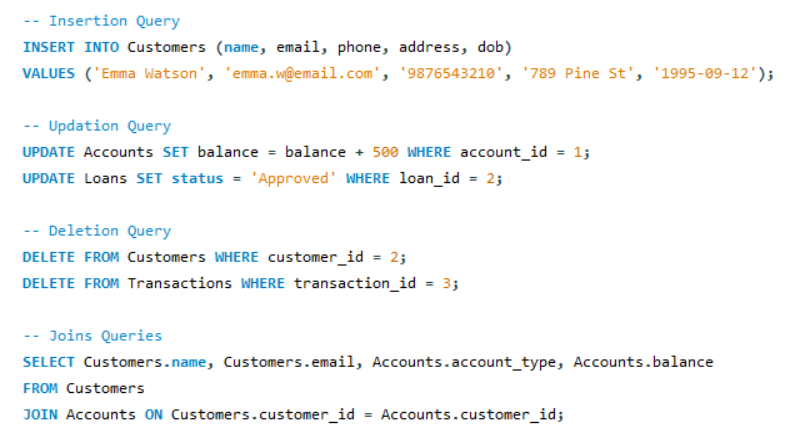


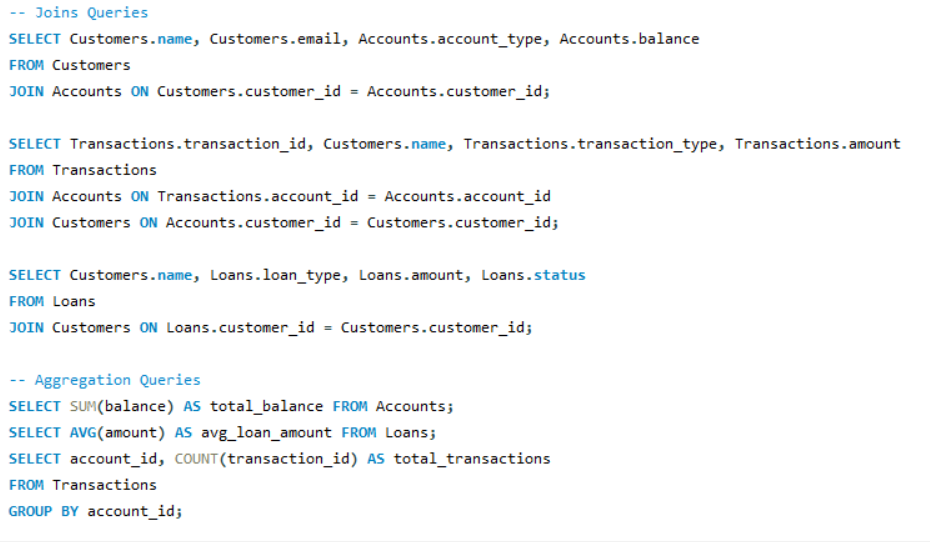


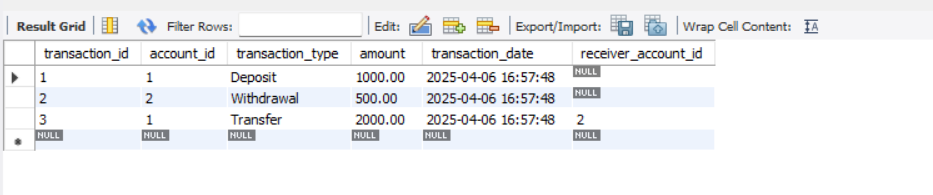


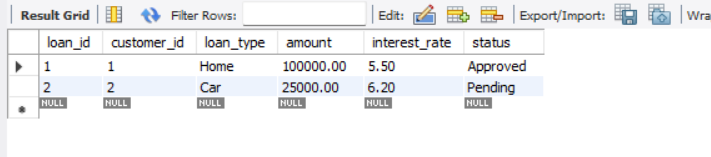
* SQL QUERIES WITH OUTPUT:











* SUMMARY:

The **University Student Information System (USIS)** is a comprehensive solution designed to manage the various aspects of student and faculty data within a university setting. Using a **Database Management System (DBMS)**, the system organizes, stores, and retrieves essential information related to students, faculty, courses, subjects, exams, attendance, and more.

#### **Key Features:**

* **Student Management:** The system stores and manages detailed records of each student, including personal details, academic history, enrolled courses, and grades. It also tracks student attendance and library book transactions.
* **Faculty Management:** Faculty information, such as personal details, departments, subjects taught, and results, are maintained. Faculty members can also input student grades and monitor attendance.
* **Course and Subject Management:** The system allows the university to manage courses offered, subject details, and faculty assignments to each subject.
* **Enrollment Management:** Students can be enrolled in courses through the system, and the relationships between students, courses, and subjects are clearly maintained.
* **Results and Exam Management:** Faculty can enter student results for each subject, and the system stores grade information for each student across various subjects.
* **Attendance Management:** The system records daily student attendance for each subject, tracking whether students are present or absent for each class.
* **Library Management:** The system keeps track of library transactions, such as the issuance and return of books to students.
* **User Authentication and Security:** The login system ensures secure access to the platform for students, faculty, and administrators. Each user has a unique role, with different levels of access to the system's features.

#### **Database Design:**

The system is built on a **relational database** with interconnected tables for **students, faculty, courses, subjects, enrollments, results, attendance, login credentials, and library transactions**. **Primary keys** and **foreign keys** are used to ensure data integrity and establish relationships between tables.

#### **Technology Stack:**

* **DBMS:** Popular DBMS like MySQL, PostgreSQL, or Oracle.
* **Frontend:** Web-based interface built with HTML, CSS, JavaScript.
* **Backend:** PHP, Python, Java, or Node.js for server-side programming.
* **Security:** User authentication, role-based access control, and data encryption.

This system ensures **efficiency, accuracy, and accessibility** of student-related data, making it easier for administrators, students, and faculty to interact with the system. It is a scalable solution that can grow alongside the university, adapting to future requirements.

* CONCLUSION:

The **University Student Information System (USIS)** is a vital tool for managing and organizing student and faculty data within a university. By leveraging a **relational database management system (DBMS)**, the solution provides an efficient, scalable, and secure way to store, retrieve, and update essential academic and administrative information.

This system streamlines various processes such as **student enrollment, course management, faculty assignments, attendance tracking**, and **exam results management**, offering a centralized platform for all stakeholders, including **students, faculty, and administrators**. The system’s **user authentication** and **role-based access control** ensure that sensitive information is protected and only accessible to authorized individuals.

Through its structured design, **relational tables**, and **interconnected relationships**, the USIS guarantees data integrity and consistency, making it easier for the university to manage academic records and administrative tasks. Additionally, it enhances the user experience by providing easy access to real-time information for students and faculty alike.

In conclusion, the **University Student Information System** is an essential component in the digital transformation of educational institutions, helping universities improve **efficiency, transparency**, and **accuracy** in managing academic and administrative processes. It can serve as a foundation for future expansion and feature enhancement, such as the integration of **online exams**, **AI-driven recommendations for students**, or **advanced analytics for academic performance**.

The system's adaptability and robust database design ensure that it remains a valuable tool for years to come, supporting the growing needs of educational institutions.

🎯 Final Thought:  
The **University Student Information System (USIS)** represents a modern solution to the challenges faced by educational institutions in managing student and faculty data. As universities continue to grow and adopt digital technologies, systems like the USIS play a crucial role in simplifying administrative tasks, enhancing the student experience, and ensuring transparency in academic operations.

By centralizing essential data, automating processes like enrollment, attendance, and result tracking, and providing seamless access to information, the USIS not only improves operational efficiency but also fosters a more collaborative and informed academic environment. Furthermore, its flexible architecture ensures that it can evolve alongside the institution’s changing needs.

In an era where **data-driven decisions**, **real-time access**, and **security** are paramount, the USIS sets the stage for a more connected, organized, and productive academic ecosystem. The integration of innovative technologies in such systems will continue to shape the future of education, enabling institutions to better serve their students, faculty, and stakeholders.

As educational institutions continue to evolve, having a robust, scalable, and well-designed system like the USIS will be essential in staying ahead of the curve and fostering an environment where both academic and administrative success can thrive.