Project Title:

University Management System.

OBJECTIVE:

To design and implement a relational database system for managing a university's academic and administrative operations. The system aims to streamline the management of student information, course registration, faculty details, departments and payment details.

TECHNOLOGIES USED:

- Database Management System: MySQL / Oracle.
- SQL for querying and manipulating the database.
- PL/SQL for stored procedures and functions (if using Oracle).

PROJECT DESCRIPTION:

The University Management System is a comprehensive database application designed to manage various aspects of a university's operations. The database will store information about students, faculty, courses, and their payment details allowing for efficient retrieval and management of data.

KEY FEATURES:

1. **Student Management:**
    a. Store details of students including student ID, name, address, contact information, marks and enrollment course.
    b. Implement functionalities to add new students, update student information, and manage student enrollment.
2. **Faculty Management:**
    a. Store faculty information such as faculty ID, name, department, year of service, start date, course ID and contact details.
    b. Implement functionalities to add new faculty members, update faculty details, and assign courses to faculty.
3. **Course Management:**
    a. Store course details including course ID, course name,fees, and department ID.
    b. Implement functionalities to add new courses, update course information, and manage course offerings.
4. **Department Management:**
    a. Store department details including department ID, name, faculty ID and course ID.
5. **Weak Entity Payment Management:**
    a. Store payment details including payment ID, amount and date.

DATABASE SCHEMA DESIGN:

- **Students Table:**
    - **Student ID(Primary Key)**
    - Name
    - Address
    - Contact number
    - Marks
    - Grades
    - **Course ID(Foreign Key)**
    - **Department ID(Foreign Key)**
- **Faculty Table:**
    - **Faculty ID(Primary Key)**
    - Name
    - Contact number
    - Start date
    - Year of service
    - **Course ID(Foreign Key)**
- **Course Table:**
    - **Course ID(Primary Key)**
    - Name
    - Fees
    - **Department ID(Foreign Key)**
    - **Faculty ID(Foreign Key)**
- **Payment Table:**
    - Payment ID
    - Date
    - Amount
- **Department Table:**
    - **Department ID(Primary Key)**
    - Name
    - **Faculty ID (Foreign Key)**
    - **Course ID (Foreign Key)**

SAMPLE SQL QUERIES:

1.Retrieve all students enrolled in a specific course:

```
SQL> CREATE TABLE STUDENT(STUDENT_ID NUMBER(4) PRIMARY KEY, NAME VARCHAR2(10),
ADDRESS VARCHAR2(100), CONTACT_NO NUMBER(10), MARKS NUMBER(2),
GRADE VARCHAR2(1), COURSE_ID NUMBER(4) REFERENCES COURSE(COURSE_ID),
FAC_ID NUMBER(3) REFERENCES FACULTY(FAC_ID));
```

```
SQL> SELECT * FROM STUDENT WHERE COURSE_ID='1021';
OUTPUT>
STUDENT_ID  NAME    ADDRESS  CONTACT_NO  MARKS  GRADE  COURSE_ID  DEPT_ID
==========  =====    =======  ==========  ======  =====  ========  ======
1           KABIR   MUMBAI   9156843278    67      D       1021      100
7           SIYA    PUNE     94862145317   93      A       1021      100
```

2. Add a new course:

```
SQL>CREATE TABLE COURSE(COURSE_ID NUMBER(4) PRIMARY KEY,
NAME VARCHAR2(10),FEES NUMBER(10), FAC_ID NUMBER(3)
REFERENCES FACULTY(FAC_ID),DEPT_ID NUMBER(4) REFERENCES DEPT(DEPT_ID));
```

```
SQL>INSERT INTO COURSE VALUES(1033,’HARI’,2000,103,400);
OUTPUT>
COURSE_ID   NAME   FEES    FAC_ID   DEPT_ID
=========  ======  =====   ======   =======
1033        HARI   2000     103       400
```

3. Update a student's grade:

```
SQL> UPDATE STUDENT SET GRADE = ‘C’ WHERE STUDENT_ID = 1;
OUTPUT>
STUDENT_ID  NAME    ADDRESS  CONTACT_NO  MARKS  GRADE  COURSE_ID  DEPT_ID
==========  =====    =======  ==========  ======  =====  ========  ======
1           KABIR   MUMBAI   9156843278    67        C       1021      100
```

CONCLUSION:

This University Management System project demonstrates ability to design and implement a relational database, write complex SQL queries, and manage various university operations.