University Management System
DBMS Project Documentation

Project Overview

The University Management System (UMS) is a database-driven application designed to manage academic, administrative, and financial operations of a university. This project focuses on the practical implementation of core Database Management System (DBMS) concepts such as CRUD operations, joins, subqueries, transactions, indexing, stored procedures, and triggers.

1. User & Role Management

Entities: Students, Teachers, Courses, Departments, Admins

Features:

1 Add, update, and delete student and teacher records

2 Assign students to departments

3 Assign teachers to courses

4 Activate or deactivate users

DBMS Concepts Used: CRUD operations, Primary Keys, Foreign Keys, NOT NULL constraints

Example: An admin inserts a new student record or updates a student’s semester using SQL UPDATE queries.

2. Course Registration System

This is a critical module that manages student enrollment in courses while enforcing academic rules.

1 Students enroll in available courses

2 Enforce maximum credit-hour limits per semester

3 Prevent duplicate course enrollments

4 Allow students to drop or withdraw from courses

DBMS Concepts Used: Joins, Subqueries, Transactions, Rollbacks

Transaction Example: If any enrollment fails during course registration, all previous inserts are rolled back to maintain data consistency.

3. Faculty Management

Entities: Faculty, Departments

1 Manage faculty records

2 Assign faculty members to departments

3 Track courses taught by each faculty member

DBMS Concepts Used: CRUD operations, Joins, Indexing on faculty_id

4. Attendance Management

1 Mark daily attendance for students

2 View attendance percentage of students

3 Generate teacher-wise and course-wise attendance reports

DBMS Concepts Used: Aggregate Functions (COUNT, SUM), GROUP BY

Example: Identify students with attendance below 75%.

5. Examination & Results System

1 Store marks for quizzes, midterm, and final exams

2. Calculate total marks and grades
3. Calculate GPA and CGPA

DBMS Concepts Used: Aggregate Functions, Grouping, Joins

Example: Calculate CGPA per student and course-wise average marks.

6. Fee & Payments Module

1. Generate semester fee records
2. Maintain fee payment history
3. Apply fine for late payments
4. Track paid and unpaid fee status

DBMS Concepts Used: Subqueries, Triggers

Trigger Example: After a payment is inserted, the system automatically updates the fee status to 'Paid'.

7. Stored Procedures

1. register_student_course()
2. calculate_cgpa()
3. generate_semester_fee()

DBMS Concepts Used: Stored Procedures to encapsulate business logic inside the database.

8. Triggers (Automatic Enforcement)

1. Prevent course enrollment if credit-hour limit is exceeded
2. Automatically update CGPA when marks are inserted or updated
3. Prevent deletion of a course if students are enrolled

DBMS Concepts Used: Triggers and Constraints