**University Database**

A database management system (DBMS) is a structured collection of related data and software designed to access and manage this data efficiently. This system, often referred to simply as the database, stores information crucial to an organization. The main objective of a DBMS is to provide an efficient and user-friendly way to manage and retrieve data.

Schema Overview

**Classroom**: (Attributes: building, room\_number, capacity)

**Department:** (Attributes: dept\_name, building, budget)

**Course:** (Attributes: course\_id, title, dept\_name, credits)

**Instructor:** (Attributes: ID, name, dept\_name, salary)

**Section:** (Attributes: course\_id, sec\_id, semester, year, building, room\_number, time\_slot\_id)

**Teaches:** (Attributes: ID, course\_id, sec\_id, semester, year)

**Student:** (Attributes: ID, name, dept\_name, tot\_cred)

**Takes:** (Attributes: ID, course\_id, sec\_id, semester, year, grade)

**Advisor:** (Attributes: s\_ID, i\_ID)

**Time Slot:** (Attributes: time\_slot\_id, day, start\_time, end\_time)

**Prerequisite:** (Attributes: course\_id, prereq\_id)

**Description:**

The university database is designed to seamlessly integrate various types of academic and administrative information. Key components include:

**Classroom:** Manages physical details of spaces, including their locations and capacities.

**Department:** Organizes academic divisions by name, location, and budget.

**Course:** Provides data on course offerings, including their identification and credit information.

The Instructor table keeps records of faculty members, linking them to their respective departments. The Section table outlines course offerings for each semester, specifying classroom allocations. The Teaches table ensures clear connections between instructors and the sections they teach.

Students are represented in the Student table, which includes essential details like ID, name, and credits earned. The Takes table keeps track of student enrollment and grades. Furthermore, the Advisor table connects students with their academic advisors, facilitating guidance.

The Time Slot entity organizes course schedules, while the Prereq table defines course prerequisites. Altogether, this university database ensures efficient handling of academic tasks, maintaining data accuracy, and supporting the institution's educational goals. It enhances the learning and teaching experience for both students and faculty.

**Key Considerations:**

**Data Integrity:** Ensure the proper implementation of foreign key relationships to maintain consistency across tables, such as linking the dept\_name in the course table to the department table.

**Normalization:** Regularly evaluate the schema to reduce redundancy and prevent duplicating information across multiple tables.

**Access Control:** Apply stringent access control measures to safeguard sensitive information, particularly student data and financial records.

**Backup and Recovery:** Establish a consistent backup schedule and a clear recovery process to protect against data loss.

**Performance Optimization**: Monitor and enhance the database's performance to manage high query volumes effectively, especially during peak times like student registration.

**Scalability:** Design the schema to accommodate future growth, allowing for the seamless addition of new courses, students, or departments.

**Documentation:** Maintain comprehensive documentation of the schema and its relationships to simplify maintenance and support future enhancements.

**Problem no – 01**

**Problem Name:** Write SQL queries using integrity constraints to create tables for a database.

-- Create classroom table

CREATE TABLE classroom (

building VARCHAR(15),

room\_number VARCHAR(7),

capacity NUMERIC(4, 0),

PRIMARY KEY (building, room\_number) );

-- Create department table

CREATE TABLE department (

dept\_name VARCHAR(20),

building VARCHAR(15),

budget NUMERIC(12, 2) CHECK (budget > 0),

PRIMARY KEY (dept\_name),

FOREIGN KEY (building) REFERENCES classroom(building) );

-- Create course table

CREATE TABLE course (

course\_id VARCHAR(7),

title VARCHAR(50),

dept\_name VARCHAR(20),

credits NUMERIC(2, 0) CHECK (credits > 0),

PRIMARY KEY (course\_id),

FOREIGN KEY (dept\_name) REFERENCES department(dept\_name) ON DELETE SET NULL );

-- Create instructor table

CREATE TABLE instructor (

ID VARCHAR(5),

name VARCHAR(20) NOT NULL,

dept\_name VARCHAR(20),

salary NUMERIC(8, 2) CHECK (salary > 29000),

PRIMARY KEY (ID),

FOREIGN KEY (dept\_name) REFERENCES department(dept\_name) ON DELETE SET NULL );

-- Create section table

CREATE TABLE section (

course\_id VARCHAR(8),

sec\_id VARCHAR(8),

semester VARCHAR(6) CHECK (semester IN ('Fall', 'Winter', 'Spring', 'Summer')),

year NUMERIC(4, 0) CHECK (year > 1701 AND year < 2100),

building VARCHAR(15),

room\_number VARCHAR(7),

time\_slot\_id VARCHAR(4),

PRIMARY KEY (course\_id, sec\_id, semester, year),

FOREIGN KEY (course\_id) REFERENCES course(course\_id) ON DELETE CASCADE,

FOREIGN KEY (building, room\_number) REFERENCES classroom(building, room\_number) );