S3 CSBS Database Management Systems Lab Mini Project

Team 6

Problem Statement:

Implement University Database Management System

Team Members:

1. BENITTA PAUL - 24
2. CHITHRALEKSHMI R - 22
3. DAVID VINOJ MATHEW - 23
4. DEA ELIZABETH VARGHESE - 24
5. HATHIK H - 28
6. MILIN SHOY - 42
7. MINAL SARA VINOD - 43
8. NEVIN TOM - 50
9. NOEL MATHEN ELDHO - 53
10. ROSHNI ALDRIN - 59

Theoretical Background:

Entity relationship diagram:

An Entity Relationship Diagram is a diagram that represents relationships among entities in a database. It is commonly known as an ER Diagram. An ER Diagram in DBMS plays a crucial role in designing the database. Today’s business world previews all the requirements demanded by the users in the form of an ER Diagram. Later, it's forwarded to the database administrators to design the database

Relational Entity relationship diagram:

A relational schema is a set of [relational tables](https://www.sciencedirect.com/topics/computer-science/relational-table) and associated items that are related to one another. All of the base tables, views, indexes, domains, user roles, stored modules, and other items that a user creates to fulfill the data needs of a particular enterprise or set of applications belong to one schema. [SQL](https://www.sciencedirect.com/topics/computer-science/structured-query-language) provides a statement to define a relational schema

MySQL:

MySQL is a relational database management system (RDBMS) developed by Oracle that is based on structured query language (SQL). A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or a place to hold the vast amounts of information in a corporate network.

MySQL – python connector:

**MySQL Connector/Python enables Python programs to access MySQL databases, using an API that is compliant with the**[**Python Database API Specification v2.0**](http://www.python.org/dev/peps/pep-0249/)

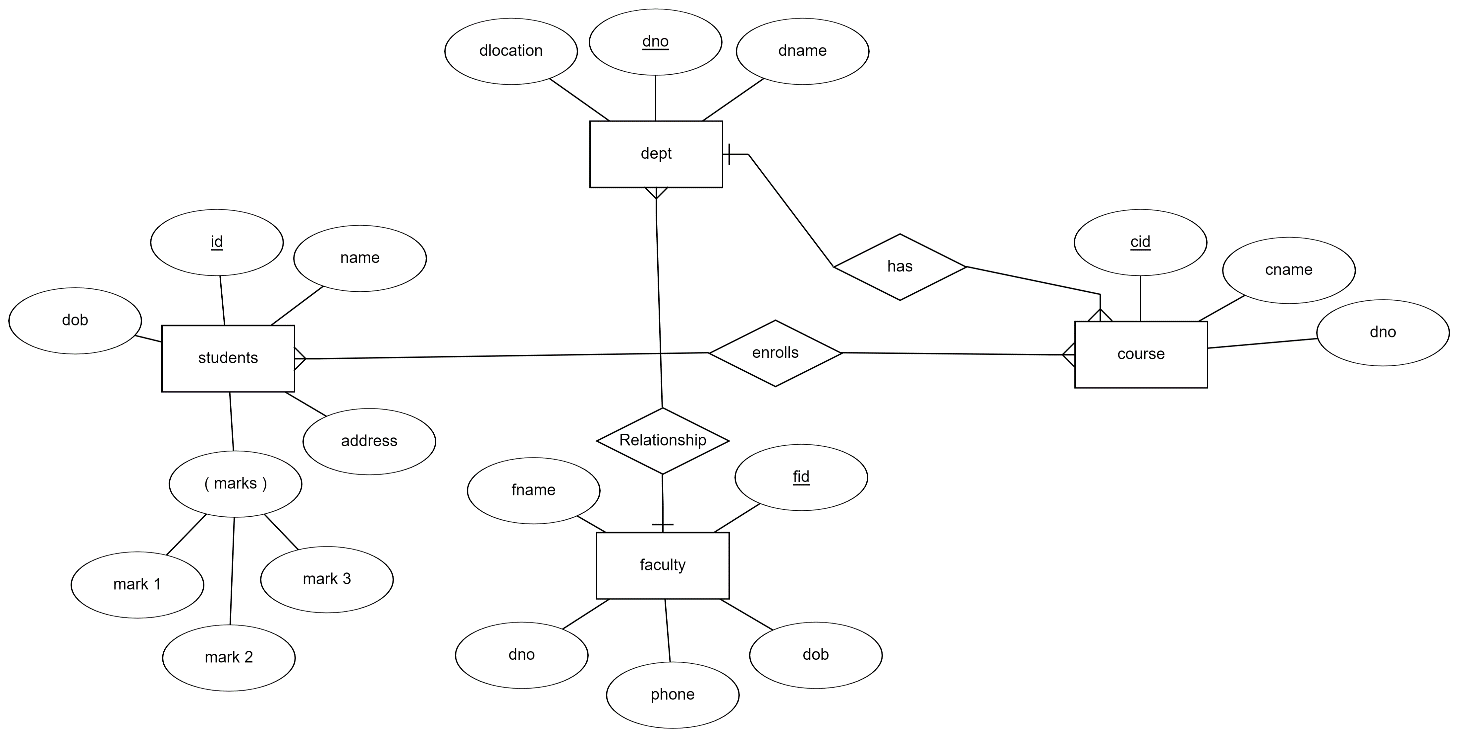
**MySQL Connector/Python includes support for:**

* **Almost all features provided by MySQL Server up to and including MySQL Server version 8.0.**

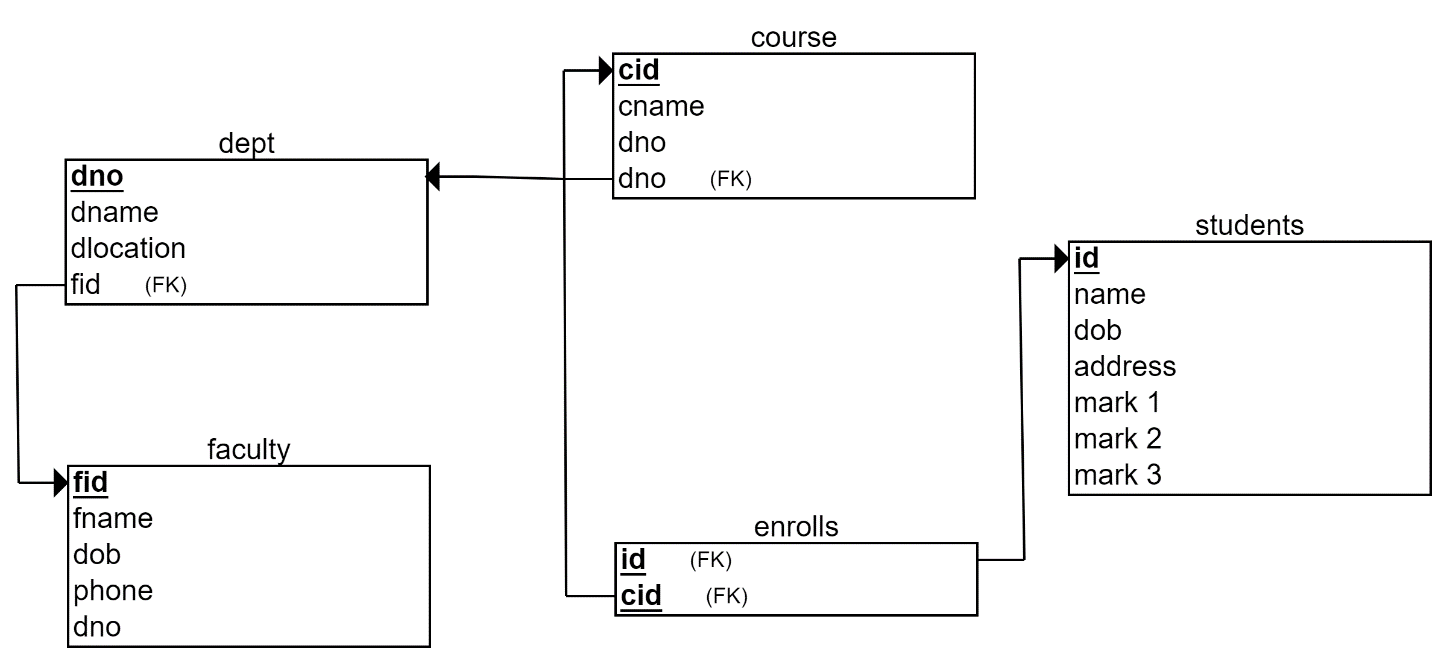
**Connector/Python 8.0 also supports X DevAPI. For documentation of the concepts and the usage of MySQL Connector/Python with X DevAPI, see**[**X DevAPI User Guide**](https://dev.mysql.com/doc/x-devapi-userguide/en/)**.**

* **Converting parameter values back and forth between Python and MySQL data types, for example Python datetime and MySQL DATETIME. You can turn automatic conversion on for convenience, or off for optimal performance.**
* **All MySQL extensions to standard SQL syntax.**
* **Protocol compression, which enables compressing the data stream between the client and server.**
* **Connections using TCP/IP sockets and on Unix using Unix sockets.**
* **Secure TCP/IP connections using SSL.**
* **Self-contained driver. Connector/Python does not require the MySQL client library or any Python modules outside the standard library.**

ER Diagram:



Relational Schema:



Program Code:

import mysql.connector as conn

mydb = conn.connect(

*host*="localhost",

*user*="root",

*password*="12345",

*database*="mathenndb"

       )

cur=mydb.cursor()

cur.execute("CREATE TABLE IF NOT EXISTS dept (dno INT AUTO\_INCREMENT PRIMARY KEY, dname VARCHAR(255), dlocation VARCHAR(20))")

cur.execute("CREATE TABLE IF NOT EXISTS course (cid INT AUTO\_INCREMENT PRIMARY KEY, cname VARCHAR(255), dno INT, FOREIGN KEY(dno) REFERENCES dept(dno))")

cur.execute("CREATE TABLE IF NOT EXISTS students (id INT AUTO\_INCREMENT PRIMARY KEY, name VARCHAR(255), dob DATE, address VARCHAR(255), mark1 INT, mark2 INT, mark3 INT, cid INT, FOREIGN KEY(cid) REFERENCES course(cid))")

cur.execute("CREATE TABLE IF NOT EXISTS faculty (fid INT AUTO\_INCREMENT PRIMARY KEY, fname VARCHAR(255), dob DATE, phone INT, dno INT, FOREIGN KEY(dno) REFERENCES dept(dno))")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*class* dept:

*def* get\_dept(*self*):

        dno = input("Enter DEPT ID : ")

        dname = input("Enter DEPT name : ")

        dlocation = input("Enter location : ")

        sql = "INSERT INTO dept(dno, dname, dlocation) VALUES (%s, %s, %s)"

        val = (dno, dname, dlocation)

        cur.execute(sql, val)

        mydb.commit()

        print("Department added successfully!")

        print()

*def* put\_dept(*self*):

        cur.execute("SELECT \* FROM dept")

        result = cur.fetchall()

        if result:

            for row in result:

                print()

                print("Department Number: " + str(row[0]))

                print("Department Name: " + row[1])

                print("Department location: " + str(row[2]))

                print()

        else:

            print("No departments available")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*class* course:

*def* get\_course(*self*):

        cid=input("Enter couse id: ")

        cname=input("Enter course name: ")

        dno=input("Enter department id: ")

        sql = "INSERT INTO course(cid,cname,dno) VALUES (%s, %s, %s)"

        val = (cid,cname,dno)

        cur.execute(sql, val)

        mydb.commit()

        print("Course added successfully!")

        print()

*def* view\_course(*self*):

        cur.execute("SELECT \* FROM course")

        result = cur.fetchall()

        if result:

            for row in result:

                print()

                print("Course ID: " + str(row[0]))

                print("Course Name: " + row[1])

                print("Department ID: " + str(row[2]))

                print()

        else:

            print("No courses available!")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*class* students:

*def* get\_student(*self*):

        id = input("Enter id: ")

        name = input("Enter name: ")

        dob  = input("Enter dob(format:YYYY-MM-DD): ")

        address = input("Enter address: ")

        cid = input("Enter course id: ")

        mark1 = mark2 = mark3 = '0'

        sql = "INSERT INTO students (id, name, dob, address, mark1, mark2, mark3, cid) VALUES (%s, %s, %s, %s, %s, %s, %s, %s)"

        val = (id, name, dob, address, mark1, mark2, mark3, cid)

        cur.execute(sql, val)

        mydb.commit()

        print("STUDENT ADDED SUCCESSFULLY!")

        print()

*def* put\_student(*self*):

        id = input("Enter id: ")

        sql = "SELECT \* FROM students WHERE id = %s"

        val = (id,)

        cur.execute(sql, val)

        result = cur.fetchone()

        if result:

            print()

            print("ID: " + str(result[0]))

            print("Name: " + result[1])

            print("DOB: " + str(result[2]))

            print("Address: " + result[3])

            print("Course: " + str(result[7]))

            print("Maths mark: " + str(result[4]))

            print("Physics mark: " + str(result[5]))

            print("Chemistry mark: " + str(result[6]))

            print()

        else:

            print("Student not found!")

*def* put\_mark(*self*):

        id = input("Enter id: ")

        sql = "SELECT name, mark1, mark2, mark3 FROM students WHERE id = %s"

        val = (id,)

        cur.execute(sql, val)

        result = cur.fetchone()

        if result:

            print()

            print("Name: " + str(result[0]))

            print("Maths mark: " + str(result[1]))

            print("Physics mark: " + str(result[2]))

            print("Chemistry mark: " + str(result[3]))

            print()

        else:

            print("Student not found!")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*class* faculty:

*def* get\_faculty(*self*):

        fid = input("Enter Faculty ID: ")

        fname = input("Enter Faculty name: ")

        dob =  input("Enter date of birth: ")

        phone = input("Enter Phone number: ")

        dno = input("Enter Department number: ")

        sql = "INSERT INTO Faculty(fid, fname, dob, phone, dno) VALUES (%s, %s, %s, %s, %s)"

        val = (fid, fname, dob, phone, dno)

        cur.execute(sql, val)

        mydb.commit()

        print("Faculty added successfully!")

        print()

*def* put\_faculty(*self*):

        cur.execute("SELECT \*, dname FROM Faculty, dept where Faculty.dno = dept.dno")

        result = cur.fetchall()

        if result:

            for row in result:

                print()

                print("FACULTY ID        : " + str(row[0]))

                print("NAME              : " + row[1])

                print("DOB               : " + str(row[2]))

                print("PHONE NUMBER      : " + str(row[3]))

                print("DEPARTMENT NUMBER : " + str(row[4]))

                print("DEPARTMENT NAME   : " + str(row[6]))

                print()

        else:

            print("No available faculties!")

*def* view\_all\_stud(*self*):

        cur.execute("SELECT \* FROM students")

        result = cur.fetchall()

        for row in result:

            print()

            print("ID            : " + str(row[0]))

            print("Name          : " + (row[1]))

            print("DOB           : " + str(row[2]))

            print("Address       : " + (row[3]))

            print("Course        : " + str(row[7]))

            print("Maths mark    : " + str(row[4]))

            print("Physics mark  : " + str(row[5]))

            print("Chemistry mark: " + str(row[6]))

            print()

*def* get\_mark(*self*):

        id=input("Enter ID:")

        m1=input("Enter math marks:")

        m2=input("Enter physics marks:")

        m3=input("Enter chemistry marks:")

        sql="update students set mark1=%s, mark2=%s, mark3=%s where id=%s"

        val=(m1, m2, m3, id,)

        cur.execute(sql, val)

        mydb.commit()

        print("Mark Added!")

        print()

*def* edit\_mark(*self*):

        id = input("Enter id: ")

        while True:

            print("\n1. Maths")

            print("2. Physics")

            print("3. Chemistry")

            print("0. Go Back")

            ch1 = input("\nEnter choice: ")

            if ch1 == "1":

                m=input("Enter Maths marks:")

                sql="UPDATE students SET mark1=%s WHERE id=%s"

                val = (m,id)

                cur.execute(sql, val)

            elif ch1 == "2":

                m=input("Enter Physics marks:")

                sql="UPDATE students SET mark2=%s WHERE id=%s"

                val = (m,id)

                cur.execute(sql, val)

            elif ch1 == "3":

                m=input("Enter Chemistry marks:")

                sql = "UPDATE students SET mark3=%s WHERE id=%s"

                val = (m,id)

                cur.execute(sql, val)

            elif ch1 == "0":

                break

            else:

                print("\nInvalid choice.Please try again.")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

dept1=dept()

stud1=students()

cr1=course()

fac1=faculty()

while True:

    print("\nUNIVERSITY DATABSE MANAGEMENT SYSTEM\n")

    print("1. STUDENT'S CORNER")

    print("2. FACULTY'S CORNER")

    print("3. ABOUT COURSES")

    print("4. ABOUT DEPARTMENTS")

    print("5. TRUNCATE(DELETE) ALL DATA")

    print("0. EXIT")

    ch = input("\nEnter your choice: ")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    if ch == "1":

        while True:

            print("\n1. Enter Student Details")

            print("2. Show Student Details")

            print("3. View marks")

            print("0. Go back")

            ch1 = input("\nEnter choice: ")

            if ch1 == "1":

                stud1.get\_student()

            elif ch1 == "2":

                stud1.put\_student()

            elif ch1 == "3":

                stud1.put\_mark()

            elif ch1=="0":

                break

            else:

                print("\nInvalid choice. Please try again.")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    elif ch == "2":

        while True:

            print("\n1. Enter Faculty Details")

            print("2. Show Faculty Details")

            print("3. View All Students")

            print("4. Enter Student Marks")

            print("5. Modify Student Marks")

            print("0. Go back")

            ch1 = input("\nEnter choice: ")

            if ch1 == "1":

                fac1.get\_faculty()

            elif ch1 == "2":

                fac1.put\_faculty()

            elif ch1 == "3":

                fac1.view\_all\_stud()

            elif ch1 == "4":

                fac1.get\_mark()

            elif ch1 == "5":

                fac1.edit\_mark()

            elif ch1=="0":

                break

            else:

                print("\nInvalid choice. Please try again.")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    elif ch == "3":

        while True:

            print("\n1. Register Course")

            print("2. View available Courses")

            print("0. Go back")

            ch1 = input("\nEnter choice: ")

            if ch1 == "1":

                cr1.get\_course()

            elif ch1 == "2":

                cr1.view\_course()

            elif ch1=="0":

                break

            else:

                print("\nInvalid choice. Please try again.")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    elif ch == "4":

        while True:

            print("\n1. Add Department")

            print("2. View Department")

            print("0. Go back")

            ch1 = input("\nEnter choice: ")

            if ch1 == "1":

                dept1.get\_dept()

            elif ch1 == "2":

                dept1.put\_dept()

            elif ch1 == "0":

                break

            else:

                print("\nInvalid choice. Please try again.")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    elif ch == '5':

*def* trun():

            print("Are You Sure to TRUNCATE All VALUES (Y/N)")

            x=input()

            if (x=="Y" or x=="y"):

                cur.execute("TRUNCATE table students")

                cur.execute("TRUNCATE table faculty")

                cur.execute("TRUNCATE table course")

                cur.execute("TRUNCATE table dept")

                mydb.commit()

                print("Tables Truncated")

            else:

                print("Revoked by User")

        trun()

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    elif ch == "0":

        print("\nHAVE A NICE DAY :)")

        break

    else:

        print("\nInvalid choice. Please try again.")

Output:

