**Group No.: B Project No.: Z**

**Project Name: IIT BHILAI COURSE MANAGEMENT SYSTEM**

**PHASE - IV**

**DATABASE MANAGEMENT & APPLICATIONS**

1. **CHANGES MADE**:
2. In the requirement specification:
   1. Application requirement 29 and 30 are duplicate of 5, 6 and 7. So, removing 29 and 30.
   2. Application requirement 27 is duplicate of 9.
3. In the ER Diagram:
   1. **PinCode** attribute was added to **Student** table of the Logical Database Design given to us but was not mentioned in the changes done to the ER Diagram there, so we are mentioning here.
   2. We have made the Term attribute of Course entity as single-valued instead of multivalued because a course can be offered in at the most one term (spring, summer or fall) at a time.
   3. In **Student** entity, **IsGraduate** attribute has been added and corresponding changes have been made in logical database design.
   4. Added multivalued **Announcement** attribute to **Course** entity.
4. In the Logical Database Design:
   1. There were no foreign key and primary key declared in many tables. So we have clearly mentioned that and used the same in SQL queries.
   2. Added **DepartmentID** in **Department** relation and made it the primary key.
   3. Added **DepartmentID** in **Student** entity as a foreign key and removed the **StudentOf** table as it is 1:N connected relation.
   4. Added **DepartmentID** in **Instructor** entity as a foreign key and removed the **FacultyIn** table as it is 1:N connected relation.
   5. Added another attribute **InstID** in **D\_Approver** table as a foreign key referencing **Instructor(InstID)** and removed **DHead** attribute from the **Department** table as it can be specified by **Approver** attribute of **D\_Approver** table.
   6. Merged **CourseDetails** into **Course** table because the separate table was not required and added another attribute **OfferingYear** in the **Course** table.
   7. Added another attribute **Term** in **CourseGradeDetails** table.
   8. Removed the table **Semester** as it was not needed.
   9. Added another table EnrolledIn corresponding to the EnrolledIn relation in ER Diagram.
   10. Added a table **Course\_Announcement** for multivalued attribute **Announcement**.
5. **SQL QUERIES TO CREATE DATABASE AND TABLES**:
   1. To create Database:

**CREATE DATABASE** course\_management;

* 1. To create Tables:
     1. Department :

**CREATE TABLE** Department(

DepartmentID **VARCHAR(10) NOT NULL**,

DName **VARCHAR(30) NOT NULL**,

**PRIMARY KEY** (DepartmentID)

);

* + 1. Student :

**CREATE TABLE** Student(

RollNo **CHAR(8) NOT NULL**,

Fname **VARCHAR(30) NOT NULL**,

Lname **VARCHAR(30) NOT NULL**,

EmailID **VARCHAR(30) NOT NULL**,

IsGraduate **BOOLEAN NOT NULL DEFAULT FALSE**,

Qualification **VARCHAR(30) NULL**,

DOB **DATE NOT NULL**,

City **VARCHAR(30) NOT NULL**,

State **VARCHAR(30) NOT NULL**,

Street **VARCHAR(30) NOT NULL**,

PinCode **VARCHAR(10) NOT NULL**,

DepartmentID **VARCHAR(10) NULL**,

**PRIMARY KEY** (RollNo),

**FOREIGN KEY** (DepartmentID) **REFERENCES** Department(DepartmentID)

**ON UPDATE** **CASCADE**

**ON DELETE SET NULL**

);

* + 1. Student\_Type :

**CREATE TABLE** Student\_Type(

RollNo **CHAR(8) NOT NULL**,

Type **VARCHAR(30) NOT NULL**,

**PRIMARY KEY** (RollNo, Type),

**FOREIGN KEY** (RollNo) **REFERENCES** Student(RollNo)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

***Note:*** In the above table, Type attribute can be BTech, MTech, Phd, Honours\_ProjectWise, Honours\_CourseWise.

* + 1. Student\_MobNo :

**CREATE TABLE** Student\_MobNo(

RollNo **CHAR(8) NOT NULL**,

MobNo **VARCHAR(15) NOT NULL**,

**PRIMARY KEY** (RollNo, MobNo),

**FOREIGN KEY** (RollNo) **REFERENCES** Student(RollNo)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

* + 1. Instructor :

**CREATE TABLE** Instructor (

InstID **CHAR(8) NOT NULL**,

FName **VARCHAR(30) NOT NULL**,

LName **VARCHAR(30) NOT NULL**,

EmailID **VARCHAR(30) NOT NULL**,

HasLeft **BOOLEAN** **NOT NULL DEFAULT FALSE**,

Qualification **VARCHAR(30) NOT NULL**,

DepartmentID **VARCHAR(10) NULL**,

**PRIMARY KEY** (InstID),

**FOREIGN KEY** (DepartmentID) **REFERENCES** Department(DepartmentID)

**ON UPDATE** **CASCADE**

**ON DELETE SET NULL**

);

* + 1. Instructor\_Type :

**CREATE TABLE** Instructor\_Type (

InstID **CHAR(8) NOT NULL**,

Type **VARCHAR(30) NOT NULL**,

**PRIMARY KEY** (InstID, Type),

**FOREIGN KEY** (InstID) **REFERENCES** Instructor(InstID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

***Note:*** In the above table, Type attribute can be Assistant, Associate, TA, Guest, etc.

* + 1. D\_Approver :

**CREATE TABLE** D\_Approver(

DepartmentID **VARCHAR(10) NOT NULL**,

Approver **VARCHAR(4) NOT NULL**,

InstID **CHAR(8) NOT NULL,**

**PRIMARY KEY** (DepartmentID, Approver),

**FOREIGN KEY** (DepartmentID) **REFERENCES** Department(DepartmentID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE,**

**FOREIGN KEY** (InstID) **REFERENCES** Instructor(InstID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

***Note:*** In the above table, Approver attribute can be HoD, DUGC OR DPGC.

* + 1. Course :

**CREATE TABLE** Course(

CourseID **CHAR(6) NOT NULL**,

CourseName **VARCHAR(30) NOT NULL**,

Term **VARCHAR(10) NOT NULL**,

OfferingYear **INT NOT NULL,**

StartingSeg **INT NOT NULL**,

EndingSeg **INT NOT NULL**,

Duration **INT NOT NULL**,

Credit **INT NOT NULL**,

Content **VARCHAR(1000) NULL**,

IsActive **BOOLEAN NOT NULL**,

**PRIMARY KEY** (CourseID)

);

***Note:*** In the above table, Term attribute can be Summer, Monsoon, Winter.

* + 1. CourseDetails\_Type :

**CREATE TABLE** CourseDetails\_Type(

CourseID **CHAR(6) NOT NULL**,

Type **VARCHAR(30) NOT NULL**,

**PRIMARY KEY** (CourseID, Type),

**FOREIGN KEY** (CourseID) **REFERENCES** Course(CourseID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

***Note:*** In the above table, Type attribute can be CALA, Proffesional\_Core, Lab, etc.

* + 1. CourseDetails\_Prerequisite :

**CREATE TABLE** CourseDetails\_Prerequisite(

CourseID **CHAR(6) NOT NULL,**

Prerequisite **CHAR(6) NOT NULL**,

**PRIMARY KEY** (CourseID, Prerequisite),

**FOREIGN KEY** (CourseID) **REFERENCES** Course(CourseID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE,**

**FOREIGN KEY** (Prerequisite) **REFERENCES** Course(CourseID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

* + 1. CourseDetails\_Archives :

**CREATE TABLE** CourseDetails\_Archives(

CourseID **CHAR(6) NOT NULL,**

Archives **VARCHAR(3000) NOT NULL**,

**PRIMARY KEY** (CourseID, Archives),

**FOREIGN KEY** (CourseID) **REFERENCES** Course(CourseID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

***Note:*** In above table, Archives attribute is a JSON string containing Course, CourseDetails\_Type, CourseDetails\_Prerequisite.

* + 1. CourseGradeDetails :

**CREATE TABLE** CourseGradeDetails(

RollNo **CHAR(8) NOT NULL**,

CourseID **CHAR(6) NOT NULL,**

Grade **CHAR(1) NOT NULL**,

Term **VARCHAR(10) NOT NULL**,

Year **INT NOT NULL,**

**PRIMARY KEY** (RollNo, CourseID, Term, Year),

**FOREIGN KEY** (RollNo) **REFERENCES** Student(RollNo)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE,**

**FOREIGN KEY** (CourseID) **REFERENCES** Course(CourseID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

* + 1. Course\_Announcement:

**CREATE TABLE** Course\_Announcement(

CourseID **CHAR(6) NOT NULL,**

Announcement **VARCHAR(3000),**

**PRIMARY KEY** (CourseID, Announcement),

**FOREIGN KEY** (CourseID) **REFERENCES** Course(CourseID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

* + 1. OfferedBy :

**CREATE TABLE** OfferedBy(

CourseID **CHAR(6) NOT NULL,**

DepartmentID **VARCHAR(10) NOT NULL**,

InstID **CHAR(8) NOT NULL,**

**PRIMARY KEY** (CourseID, DepartmentID, InstID),

**FOREIGN KEY** (CourseID) **REFERENCES** Course(CourseID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE,**

**FOREIGN KEY** (DepartmentID) **REFERENCES** Department(DepartmentID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE,**

**FOREIGN KEY** (InstID) **REFERENCES** Instructor(InstID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

* + 1. EnrolledIn :

**CREATE TABLE** EnrolledIn(

RollNo **CHAR(8) NOT NULL,**

CourseID **CHAR(6) NOT NULL,**

ApprovalStatus **BOOLEAN NOT NULL DEFAULT FALSE**,

**PRIMARY KEY** (RollNo, CourseID),

**FOREIGN KEY** (RollNo) **REFERENCES** Student(RollNo)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE,**

**FOREIGN KEY** (CourseID) **REFERENCES** Course(CourseID)

**ON UPDATE** **CASCADE**

**ON DELETE CASCADE**

);

1. **SQL QUERIES FOR ALL THE APPLICATION REQUIREMENTS**:
   1. Addition of a new department in the institute.

**INSERT INTO** Department **VALUES**(**‘**Did’, ‘Dname’);

**INSERT INTO** D\_Approver **VALUES**(‘DId’, ‘HoD’, ‘InstID1’);

**INSERT INTO** D\_Approver **VALUES**(‘DId’, ‘DUGC’, ‘InstID2’);

**INSERT INTO** D\_Approver **VALUES**(‘DId’, ‘DPGC’, ‘InstID3’);

* 1. Addition of new instructor in a department.

**INSERT INTO** Instructor **VALUES**(‘IId’, ‘Fname’, ‘Lname’, ‘EMailID’, BoolValue, ‘Qualification’, ‘DId’);

**INSERT INTO** Instructor\_Type **VALUES**(‘IId’, ‘Type’);

* 1. Addition of new course in a department by the instructor.

**INSERT INTO** Course **VALUES**(‘CId’, ‘CName’, ‘Term’, Year, SSeg, ESeg, Duration, Credit, ‘Content’, BoolValue);

**INSERT INTO** CourseDetails\_Type **VALUES**(‘CId’, ‘Type’)

**INSERT INTO** CourseDetails\_Prerequisite **VALUES**(‘CId’, ‘Prerequisite’)

**INSERT INTO** CourseDetails\_Archives **VALUES**(‘CId’, ‘Archives’)

* 1. Registration of student in the course management system.

**INSERT INTO** Student **VALUES**(‘RollNo’, ‘Fname’, ‘Lname’, ‘EmailID’, ‘Qualification’, DOB, ‘City’, ‘State’ **NULL**, ‘Street’, ‘PinCode’, ‘DepartmentID’)

**INSERT INTO** Student\_Type **VALUES**(‘RollNo’, ‘Type’)

**INSERT INTO** Student\_MobNo **VALUES**(‘RollNo’, **‘**MobNo’)

* 1. Student to enroll in a course.

**INSERT INTO** EnrollenIn **VALUES** (‘RollID’, ‘CourseID’);

* 1. The student to unenroll from a course.

**DELETE FROM** EnrolledIn

**WHERE** EnrolledIn.RollNo = ‘Roll No’ **AND** EnrolledIn.CourseID = ‘CourseID’;

* 1. DUGC to accept/reject the course registration.

For accepting the course registration:

**UPDATE** EnrolledIn

**SET** ApprovalStatus = **TRUE**

**WHERE** EnrolledIn.RollNo = ‘Roll No’ **AND** EnrolledIn.CourseID = ‘Course ID’;

For rejecting the course registration:

**UPDATE** EnrolledIn

**SET** ApprovalStatus = **FALSE**

**WHERE** EnrolledIn.RollNo = ‘Roll No’ **AND** EnrolledIn.CourseID = ‘Course ID’;

* 1. Updation of the Head of department.

**UPDATE** D\_Approver

**SET** InstID = ‘InstID1’

**WHERE** D\_Approver.DepartmentID = ‘DId’ **AND** D\_Approver.Approver = ‘HoD’

* 1. Updation of the course detail.

**UPDATE** Course, CourseDetails\_Type, CourseDetails\_Prerequisite, CourseDetails\_Archives

**SET** CourseName =’Course Name’, OfferingYear = year, Duration= duration, Credit = credit, IsActive = boolValue, Prerequisite = ‘prerequsite’ , Archives = ‘archives’, content = ’content’

**WHERE** CourseID = ‘Course Id’

* 1. Displaying the list of departments in the institute.

**SELECT** DName **FROM** Department

* 1. Displaying the list of all the courses offered by a department.

**SELECT** CourseID

**FROM** OfferedBy

**WHERE** OfferedBy.DepartmentId = ‘Department Id’

* 1. Displaying the course detail of a particular course.

**SELECT** CourseID, CourseName, OfferingYear, Duration, Credit, IsActive, Type, Prerequisite, Archives

**FROM** Course, CourseDetails\_Type, CourseDetails\_Prerequisite, CourseDetails\_Archives

**WHERE** CourseID = ‘Course Id’

* 1. Displaying the list of all the students in a particular course.

**SELECT** RollNo

**FROM** EnrolledIn

**WHERE** EnrolledIn.CourseID = ‘Course Id’

* 1. Displaying the list of head of departments.

**SELECT** \* **FROM** D\_Approver

* 1. Displaying the list of all the instructors of a particular department.

**SELECT** InstID

**FROM** Instructor

**WHERE** Instructor.DepartmentId = ‘Department Id’

* 1. Displaying the list of courses taken by an instructor.

**SELECT** CourseID

**FROM** OfferedBy

**WHERE** OfferedBy.InstId = ‘Instructor Id’

* 1. Displaying the list of all the students.

**SELECT** RollNo,Fname,Lname **FROM** Student

* 1. Displaying the list of courses taken by a student.

**SELECT** CourseID

**FROM** EnrolledIn

**WHERE** RollNo = ‘Roll No Id’

* 1. Displaying the grades of a student.

**SELECT** Grade

**FROM** CourseGradeDetails

**WHERE** RollNo = ‘Roll No’

* 1. Adding the grade of each student.

**INSERT INTO** CourseGradeDetails(RollNo,Grade) **VALUES(**‘Roll No’,‘grade’**)**

* 1. Update Department details like Phone No, office, DUGC, DPGC etc.

**UPDATE** Department, D\_Approver

**SET** DName = ‘New DName’, Approver = ‘New Approver’

**WHERE** DepartmentID = ‘DepartmentID’

* 1. Updating faculty/instructor details

Let’s suppose we want to update the student’s first name, email id and his type(eg. )

**UPDATE** Instructor , Instructor\_Type

**SET** FName = ‘First Name’, Type = ‘type’

**WHERE** InstID = ‘Instructor ID’

* 1. If an instructor leaves the institute then archive his details.

**UPDATE** Instructor

**SET** HasLeft = **TRUE**

**WHERE** InstID = ‘Instructor ID’

* 1. Updating students profile

Let’s suppose we want to update the student’s last name and his type(eg. btech (honours))

**UPDATE** Student , Student\_Type

**SET** LName = ‘Last Name’, IsGraduate = TRUE, Type = ‘type’

**WHERE** RollNo = ‘Roll No’

* 1. Converting students to alumni if they graduates

**UPDATE** Student

**SET** IsGraduate = **TRUE**

**WHERE** RollNo = ‘RollNo’

* 1. Addition of announcements.

**INSERT** **INTO** Course\_Announcement **VALUES**(‘Course ID’, ‘Announcement’)

1. **CHALLENGES FACED**:
   1. It was challenging **to maintain referential integrity constraint** in defining a foreign key.
   2. Although we tried to minimize the changes required in the previous work done by other groups, still a lot of changes were done in the Logical Database Design because of the incompatible work done by previous groups due to lack of communication.