University Management System

Introduction to the Problem

Introduction:

The University Management System (UMS) is a centralized software solution that efficiently manages student, faculty, and course-related information within educational institutions. It streamlines administrative processes, enhances communication, and ensures accurate records.

Objective:

The University Management System (UMS) aims to efficiently manage student, faculty, and course-related information. It streamlines administrative processes, enhances communication, and ensures accurate records.

Purpose:

The University Management System (UMS) facilitates smooth enrolment, grading, attendance tracking, and resource allocation. It supports academic collaboration, data consistency, and informed decision-making within the university ecosystem.

Logical Database Design

Entities:

- Student
- Faculty
- Course
- Department
- Classroom
- Enrolment
- Grade
- Assignment
- Attendance
- Schedule
- Library

- Book

*Attributes of Entities:*

- _*Student*_

StudentID, FirstName, LastName, DateOfBirth, Address, PhoneNumber, Email, DepartmentID

- _*Faculty*_

FacultyID, FirstName, LastName, DateOfBirth, Address, PhoneNumber, Email, DepartmentID

- _*Course*_

CourseID, CourseName, Credits, DepartmentID, FacultyID

- _*Department*_

DepartmentID, DepartmentName, OfficeNumber, PhoneNumber

- _*Classroom*_

ClassroomID, RoomNumber, Building, Capacity

- _*Enrollment*_

EnrollmentID, StudentID, CourseID, EnrollmentDate

- _*Grade*_

GradeID, StudentID, CourseID, Grade

- _*Assignment*_

AssignmentID, CourseID, AssignmentTitle, DueDate

- _*Attendance*_

AttendanceID, StudentID, CourseID, Date, Status (Present/Absent)

- _*Schedule*_

ScheduleID, CourseID, ClassroomID, DayOfWeek, StartTime, EndTime

- _*Library*_

LibraryID, LibraryName, Location, PhoneNumber

- _*Book*_

BookID, Title, Author, ISBN, PublishedYear, LibraryID

*Relationships:*

- Student to Enrolment: One-to-Many (A student can enrol in multiple courses)

- Course to Enrolment: One-to-Many (A course can have multiple students)
- Course to Faculty: Many-to-One (A course is taught by one faculty member)
- Faculty to Department: Many-to-One (A faculty member belongs to one department)
- Course to Department: Many-to-One (A course is offered by one department)
- Course to Classroom: Many-to-Many (A course can be scheduled in multiple classrooms, and a classroom can host multiple courses)
- Course to Assignment: One-to-Many (A course can have multiple assignments)
- Student to Grade: One-to-Many (A student can have multiple grades for different courses)
- Course to Grade: One-to-Many (A course can have multiple grades for different students)
- Student to Attendance: One-to-Many (A student can have multiple attendance records)
- Course to Attendance: One-to-Many (A course can have multiple attendance records)
- Student to Library: Many-to-Many (A student can borrow books from multiple libraries, and a library can lend books to multiple students)
- Library to Book: One-to-Many (A library can have multiple books)

Functional Dependencies and Normalization:

To maintain data integrity and minimize redundancy, the database tables will be normalized to at least the third normal form (3NF):

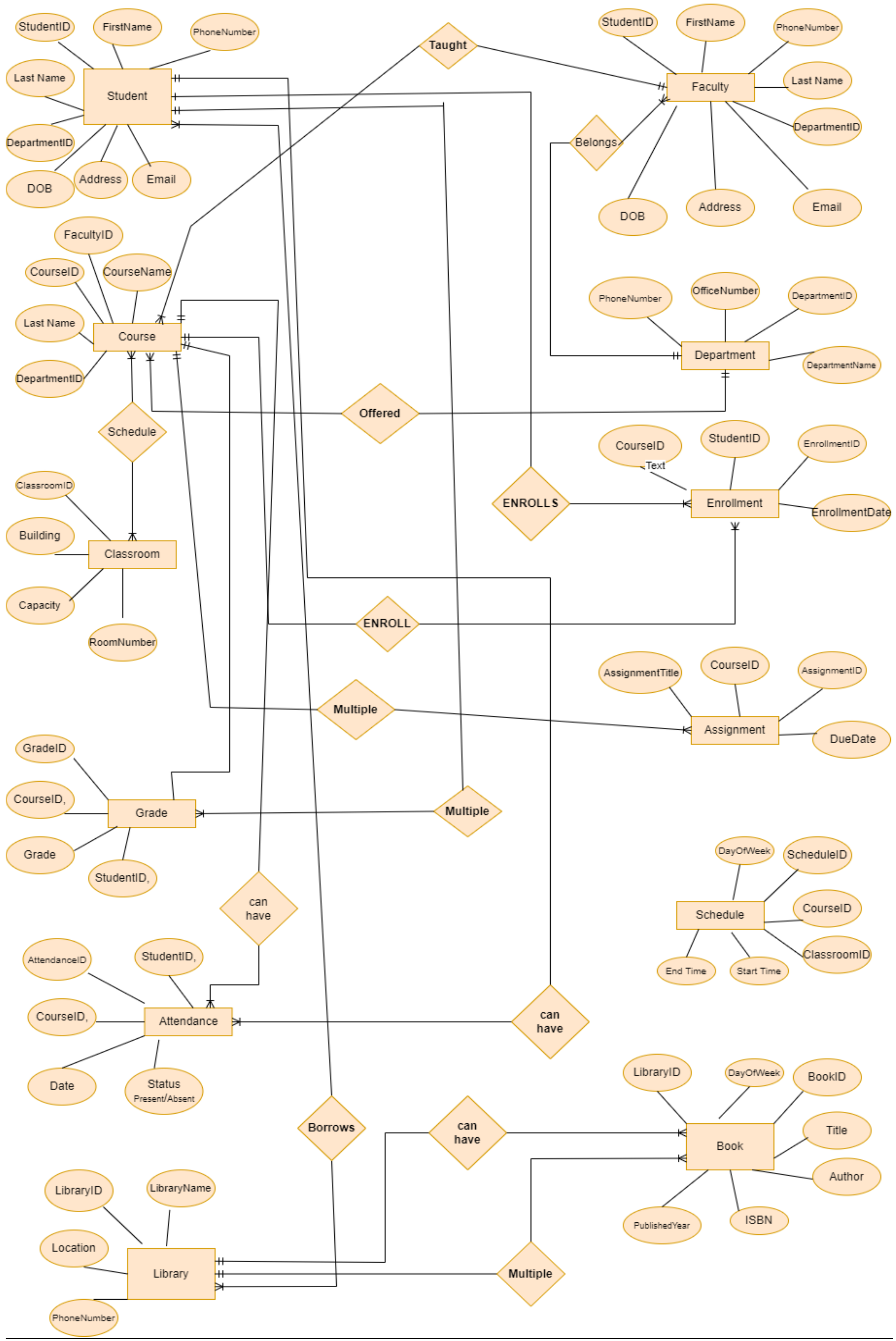
- First Normal Form (1NF): Ensure that each table has a primary key and that each column contains atomic values.
- Second Normal Form (2NF): Ensure that all non-key attributes are fully functionally dependent on the primary key.
- Third Normal Form (3NF): Ensure that there are no transitive dependencies, meaning all non-key attributes are directly dependent on the primary key.

Complete Enhanced Entity Relationship Diagram:

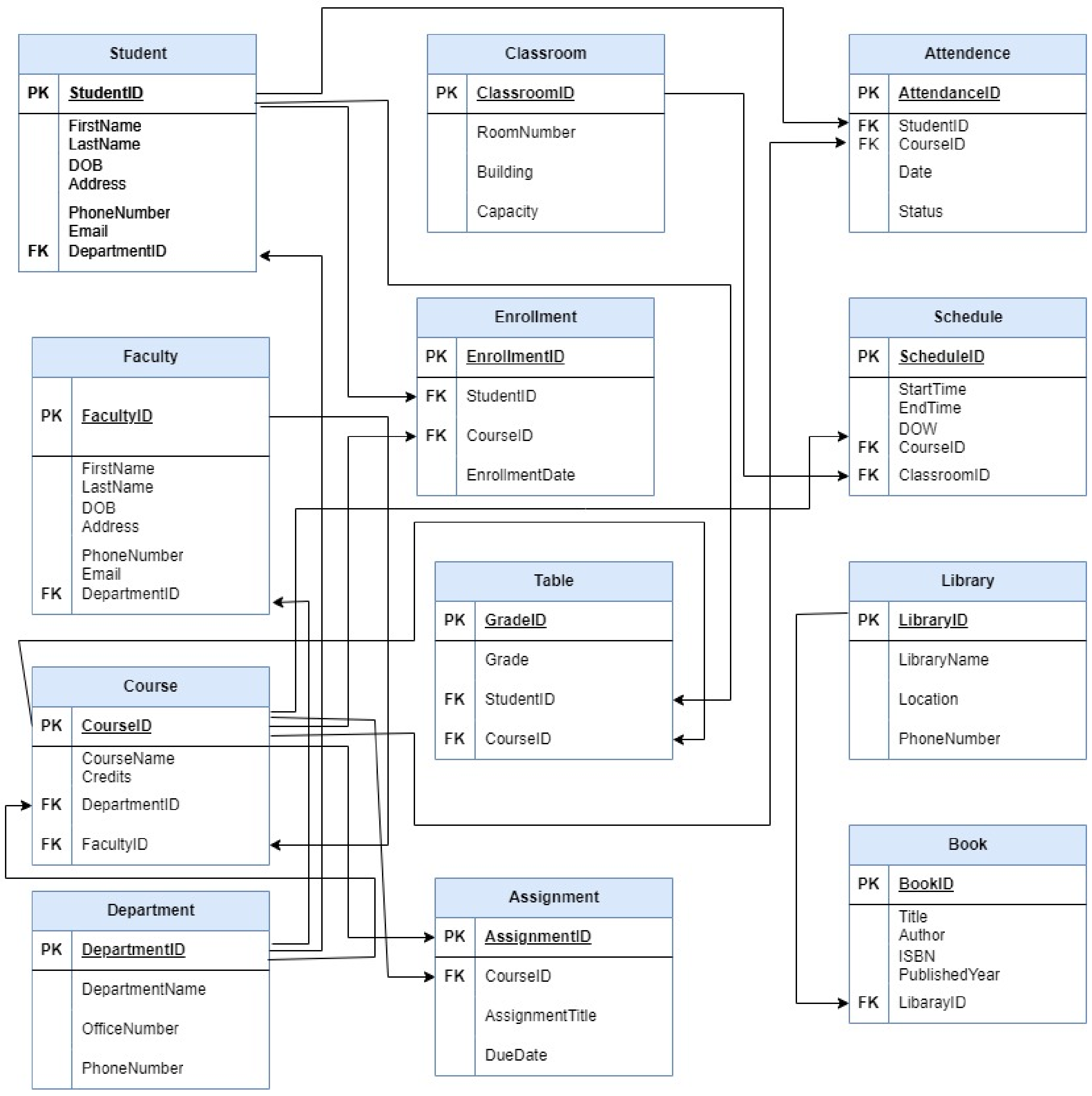
An Enhanced Entity Relationship (EER) Diagram will visually represent the entities, their attributes, and the relationships between them. This diagram will include:

- Entities with their respective attributes.
- Primary and foreign keys.
- Relationships and their cardinalities.

Entity Relationship Diagram (ERD):



Entity Relationship Model (ERM):



Implementation of Sample Data in SQL:

Create Tables:

-- Table for Department

```
CREATE TABLE Department (
    DepartmentID INT PRIMARY KEY,
    DepartmentName VARCHAR(50),
    OfficeNumber VARCHAR(10),
    PhoneNumber VARCHAR(15)
);
```

-- Table for Student

```
CREATE TABLE Student (
    StudentID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    DateOfBirth DATE,
    Address VARCHAR(100),
    PhoneNumber VARCHAR(15),
    Email VARCHAR(50),
    DepartmentID INT,
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
);
```

-- Table for Faculty

```
CREATE TABLE Faculty (
    FacultyID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    DateOfBirth DATE,
    Address VARCHAR(100),
    PhoneNumber VARCHAR(15),
    Email VARCHAR(50),
    DepartmentID INT,
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
);
```

-- Table for Course

```
CREATE TABLE Course (
    CourseID INT PRIMARY KEY,
    CourseName VARCHAR(100),
    Credits INT,
    DepartmentID INT,
    FacultyID INT,
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID),
    FOREIGN KEY (FacultyID) REFERENCES Faculty(FacultyID)
);
```

-- Table for Classroom

```
CREATE TABLE Classroom (
    ClassroomID INT PRIMARY KEY,
    RoomNumber VARCHAR(10),
    Building VARCHAR(50),
    Capacity INT
);
```

-- Table for Enrollment

```
CREATE TABLE Enrollment (
    EnrollmentID INT PRIMARY KEY,
    StudentID INT,
    CourseID INT,
    EnrollmentDate DATE,
    FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
    FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
);
```

-- Table for Grade

```
CREATE TABLE Grade (
    GradeID INT PRIMARY KEY,
    StudentID INT,
    CourseID INT,
    Grade CHAR(2),
```

```sql
    FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
    FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
);

-- Table for Assignment
CREATE TABLE Assignment (
    AssignmentID INT PRIMARY KEY,
    CourseID INT,
    AssignmentTitle VARCHAR(100),
    DueDate DATE,
    FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
);

-- Table for Attendance
CREATE TABLE Attendance (
    AttendanceID INT PRIMARY KEY,
    StudentID INT,
    CourseID INT,
    Date DATE,
    Status CHAR(1),
    FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
    FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
);

-- Table for Schedule
CREATE TABLE Schedule (
    ScheduleID INT PRIMARY KEY,
    CourseID INT,
    ClassroomID INT,
    DayOfWeek VARCHAR(10),
    StartTime TIME,
    EndTime TIME,
    FOREIGN KEY (CourseID) REFERENCES Course(CourseID),
    FOREIGN KEY (ClassroomID) REFERENCES Classroom(ClassroomID)
);
```

-- Table for Library

```
CREATE TABLE Library (
    LibraryID INT PRIMARY KEY,
    LibraryName VARCHAR(50),
    Location VARCHAR(100),
    PhoneNumber VARCHAR(15)
);
```

-- Table for Book

```
CREATE TABLE Book (
    BookID INT PRIMARY KEY,
    Title VARCHAR(100),
    Author VARCHAR(50),
    ISBN VARCHAR(20),
    PublishedYear INT,
    LibraryID INT,
    FOREIGN KEY (LibraryID) REFERENCES Library(LibraryID)
);
```

Insert Queries:

-- Insert records into Department table

```
INSERT INTO Department (DepartmentID, DepartmentName, OfficeNumber, PhoneNumber) VALUES
(1, 'Computer Science', 'CS101', '123-456-7890'),
(2, 'Mathematics', 'MATH101', '123-456-7891'),
(3, 'Physics', 'PHYS101', '123-456-7892'),
(4, 'Chemistry', 'CHEM101', '123-456-7893'),
(5, 'Biology', 'BIO101', '123-456-7894');
```

-- Insert records into Student table

```
INSERT INTO Student (StudentID, FirstName, LastName, DateOfBirth, Address, PhoneNumber, Email,
DepartmentID) VALUES
(1, 'John', 'Doe', '2000-01-15', '123 Main St', '123-456-7890', 'john.doe@example.com', 1),
(2, 'Jane', 'Smith', '2001-02-20', '456 Oak St', '123-456-7891', 'jane.smith@example.com', 2),
(3, 'Emily', 'Johnson', '1999-03-25', '789 Pine St', '123-456-7892', 'emily.johnson@example.com', 3),
(4, 'Michael', 'Brown', '2000-04-30', '101 Maple St', '123-456-7893', 'michael.brown@example.com', 4),
```

```sql
(5, 'Sarah', 'Davis', '2001-05-05', '202 Elm St', '123-456-7894', 'sarah.davis@example.com', 5);
```

-- Insert records into Faculty table

```sql
INSERT INTO Faculty (FacultyID, FirstName, LastName, DateOfBirth, Address, PhoneNumber, Email,
DepartmentID) VALUES
(1, 'Alice', 'Green', '1980-06-15', '321 Cedar St', '123-456-7890', 'alice.green@example.com', 1),
(2, 'Bob', 'White', '1975-07-20', '654 Spruce St', '123-456-7891', 'bob.white@example.com', 2),
(3, 'Charlie', 'Black', '1982-08-25', '987 Birch St', '123-456-7892', 'charlie.black@example.com', 3),
(4, 'David', 'Brown', '1978-09-30', '110 Willow St', '123-456-7893', 'david.brown@example.com', 4),
(5, 'Eve', 'Gray', '1985-10-05', '220 Walnut St', '123-456-7894', 'eve.gray@example.com', 5);
```

-- Insert records into Course table

```sql
INSERT INTO Course (CourseID, CourseName, Credits, DepartmentID, FacultyID) VALUES
(1, 'Introduction to Computer Science', 4, 1, 1),
(2, 'Calculus I', 3, 2, 2),
(3, 'Physics I', 4, 3, 3),
(4, 'Organic Chemistry', 4, 4, 4),
(5, 'Biology I', 4, 5, 5);
```

-- Insert records into Classroom table

```sql
INSERT INTO Classroom (ClassroomID, RoomNumber, Building, Capacity) VALUES
(1, '101', 'Engineering', 30),
(2, '202', 'Science', 40),
(3, '303', 'Arts', 50),
(4, '404', 'Commerce', 60),
(5, '505', 'Law', 70);
```

-- Insert records into Enrollment table

```sql
INSERT INTO Enrollment (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES
(1, 1, 1, '2024-01-10'),
(2, 2, 2, '2024-01-11'),
(3, 3, 3, '2024-01-12'),
(4, 4, 4, '2024-01-13'),
(5, 5, 5, '2024-01-14');
```

-- Insert records into Grade table

```
INSERT INTO Grade (GradeID, StudentID, CourseID, Grade) VALUES
(1, 1, 1, 'A'),
(2, 2, 2, 'B'),
(3, 3, 3, 'A'),
(4, 4, 4, 'C'),
(5, 5, 5, 'B');
```

-- Insert records into Assignment table

```
INSERT INTO Assignment (AssignmentID, CourseID, AssignmentTitle, DueDate) VALUES
(1, 1, 'Homework 1', '2024-02-01'),
(2, 2, 'Homework 2', '2024-02-02'),
(3, 3, 'Homework 3', '2024-02-03'),
(4, 4, 'Homework 4', '2024-02-04'),
(5, 5, 'Homework 5', '2024-02-05');
```

-- Insert records into Attendance table

```
INSERT INTO Attendance (AttendanceID, StudentID, CourseID, Date, Status) VALUES
(1, 1, 1, '2024-01-15', 'P'),
(2, 2, 2, '2024-01-16', 'A'),
(3, 3, 3, '2024-01-17', 'P'),
(4, 4, 4, '2024-01-18', 'A'),
(5, 5, 5, '2024-01-19', 'P');
```

-- Insert records into Schedule table

```
INSERT INTO Schedule (ScheduleID, CourseID, ClassroomID, DayOfWeek, StartTime, EndTime) VALUES
(1, 1, 1, 'Monday', '09:00:00', '10:30:00'),
(2, 2, 2, 'Tuesday', '10:00:00', '11:30:00'),
(3, 3, 3, 'Wednesday', '11:00:00', '12:30:00'),
(4, 4, 4, 'Thursday', '12:00:00', '13:30:00'),
(5, 5, 5, 'Friday', '13:00:00', '14:30:00');
```

-- Insert records into Library table

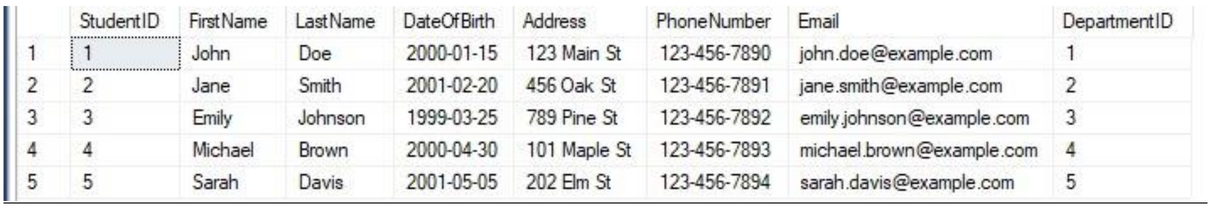
```
INSERT INTO Library (LibraryID, LibraryName, Location, PhoneNumber) VALUES
(1, 'Central Library', '123 Library St', '123-456-7890'),
(2, 'Science Library', '456 Science Rd', '123-456-7891'),
(3, 'Engineering Library', '789 Engineering Ln', '123-456-7892'),
(4, 'Arts Library', '101 Arts Blvd', '123-456-7893'),
(5, 'Law Library', '202 Law Ave', '123-456-7894');
```

-- Insert records into Book table

```
INSERT INTO Book (BookID, Title, Author, ISBN, PublishedYear, LibraryID) VALUES
(1, 'Introduction to Algorithms', 'Thomas H. Cormen', '978-0262033848', 2009, 1),
(2, 'Advanced Engineering Mathematics', 'Erwin Kreyszig', '978-0470458365', 2011, 2),
(3, 'Modern Physics', 'Kenneth S. Krane', '978-1118061145', 2012, 3),
(4, 'Organic Chemistry', 'Paula Yurkanis Bruice', '978-0321803221', 2013, 4),
(5, 'Biology', 'Neil A. Campbell', '978-0321558237', 2008, 5);
```

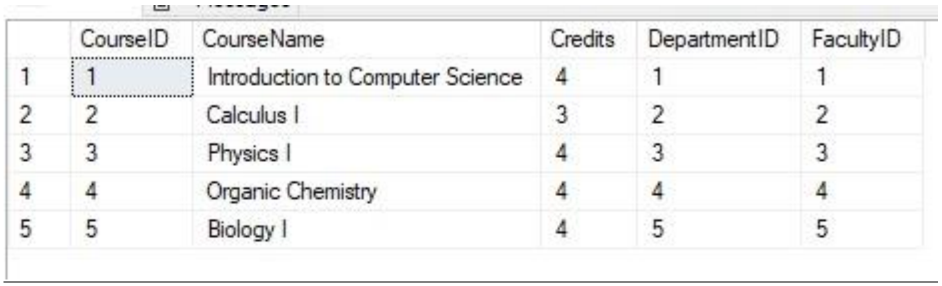
Data Tables Used:

Student



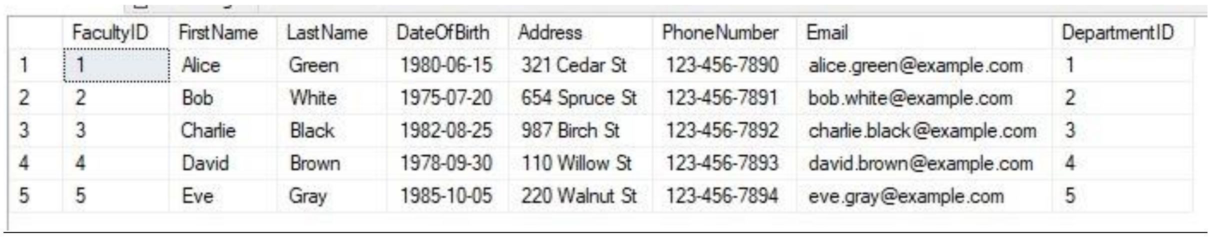
	StudentID	FirstName	LastName	DateOfBirth	Address	PhoneNumber	Email	DepartmentID
1	1	John	Doe	2000-01-15	123 Main St	123-456-7890	john.doe@example.com	1
2	2	Jane	Smith	2001-02-20	456 Oak St	123-456-7891	jane.smith@example.com	2
3	3	Emily	Johnson	1999-03-25	789 Pine St	123-456-7892	emily.johnson@example.com	3
4	4	Michael	Brown	2000-04-30	101 Maple St	123-456-7893	michael.brown@example.com	4
5	5	Sarah	Davis	2001-05-05	202 Elm St	123-456-7894	sarah.davis@example.com	5

Course



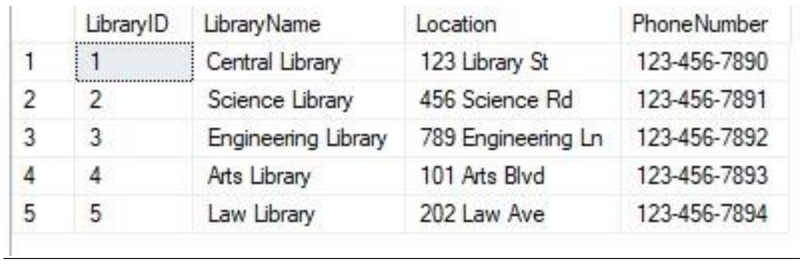
	CourseID	CourseName	Credits	DepartmentID	FacultyID
1	1	Introduction to Computer Science	4	1	1
2	2	Calculus I	3	2	2
3	3	Physics I	4	3	3
4	4	Organic Chemistry	4	4	4
5	5	Biology I	4	5	5

Faculty



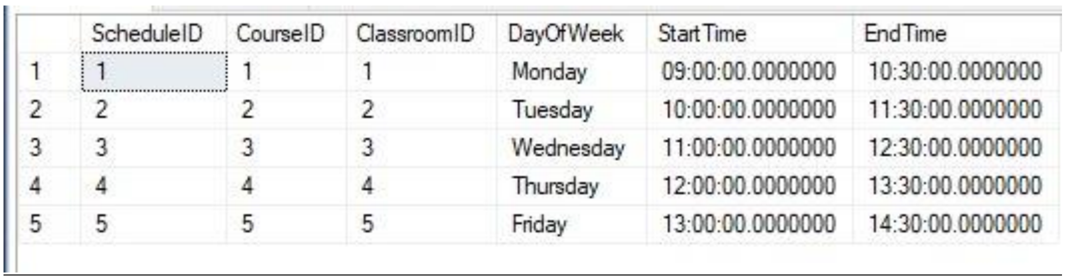
	FacultyID	FirstName	LastName	DateOfBirth	Address	PhoneNumber	Email	DepartmentID
1	1	Alice	Green	1980-06-15	321 Cedar St	123-456-7890	alice.green@example.com	1
2	2	Bob	White	1975-07-20	654 Spruce St	123-456-7891	bob.white@example.com	2
3	3	Charlie	Black	1982-08-25	987 Birch St	123-456-7892	charlie.black@example.com	3
4	4	David	Brown	1978-09-30	110 Willow St	123-456-7893	david.brown@example.com	4
5	5	Eve	Gray	1985-10-05	220 Walnut St	123-456-7894	eve.gray@example.com	5

Library



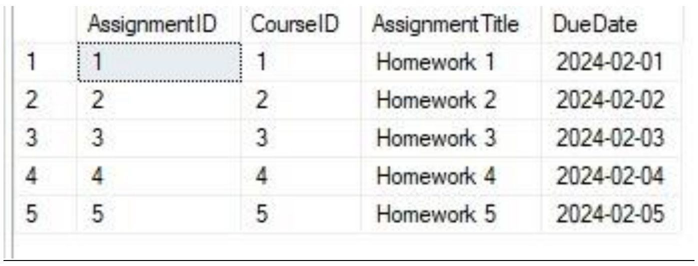
	LibraryID	LibraryName	Location	PhoneNumber
1	1	Central Library	123 Library St	123-456-7890
2	2	Science Library	456 Science Rd	123-456-7891
3	3	Engineering Library	789 Engineering Ln	123-456-7892
4	4	Arts Library	101 Arts Blvd	123-456-7893
5	5	Law Library	202 Law Ave	123-456-7894

Schedule



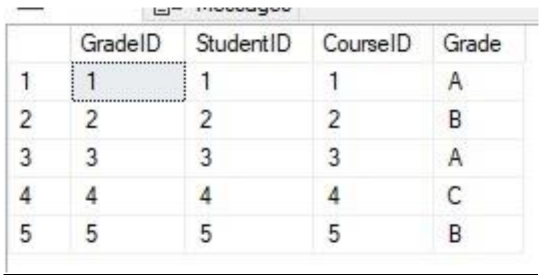
	ScheduleID	CourseID	ClassroomID	DayOfWeek	StartTime	EndTime
1	1	1	1	Monday	09:00:00.0000000	10:30:00.0000000
2	2	2	2	Tuesday	10:00:00.0000000	11:30:00.0000000
3	3	3	3	Wednesday	11:00:00.0000000	12:30:00.0000000
4	4	4	4	Thursday	12:00:00.0000000	13:30:00.0000000
5	5	5	5	Friday	13:00:00.0000000	14:30:00.0000000

Assignment



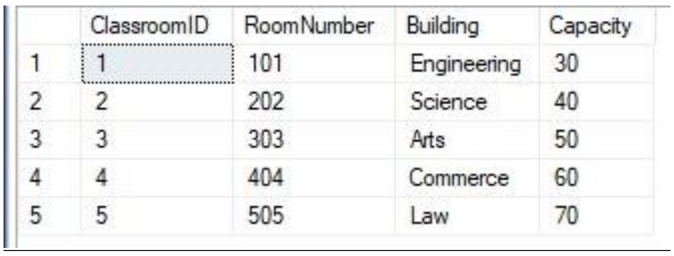
	AssignmentID	CourseID	AssignmentTitle	DueDate
1	1	1	Homework 1	2024-02-01
2	2	2	Homework 2	2024-02-02
3	3	3	Homework 3	2024-02-03
4	4	4	Homework 4	2024-02-04
5	5	5	Homework 5	2024-02-05

Grade



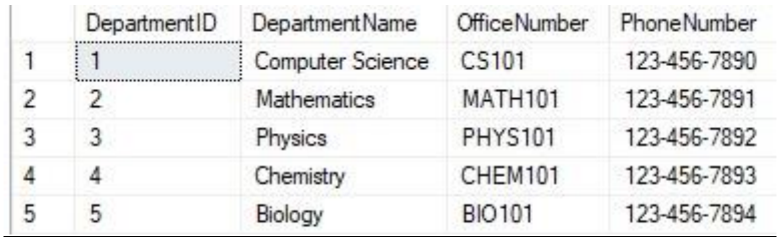
	GradeID	StudentID	CourseID	Grade
1	1	1	1	A
2	2	2	2	B
3	3	3	3	A
4	4	4	4	C
5	5	5	5	B

Classroom



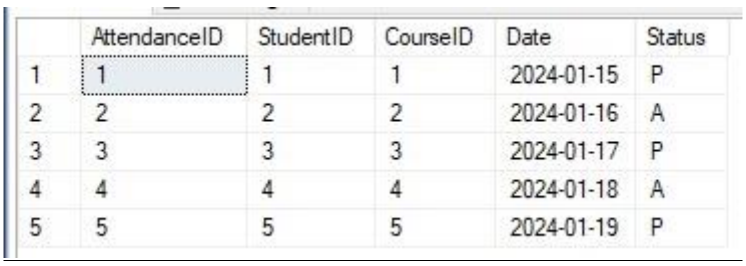
	ClassroomID	RoomNumber	Building	Capacity
1	1	101	Engineering	30
2	2	202	Science	40
3	3	303	Arts	50
4	4	404	Commerce	60
5	5	505	Law	70

Department



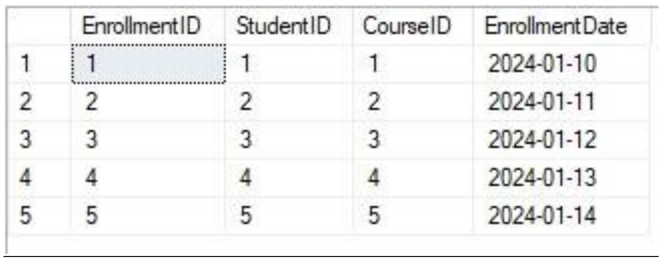
	DepartmentID	DepartmentName	OfficeNumber	PhoneNumber
1	1	Computer Science	CS101	123-456-7890
2	2	Mathematics	MATH101	123-456-7891
3	3	Physics	PHYS101	123-456-7892
4	4	Chemistry	CHEM101	123-456-7893
5	5	Biology	BIO101	123-456-7894

Attendance



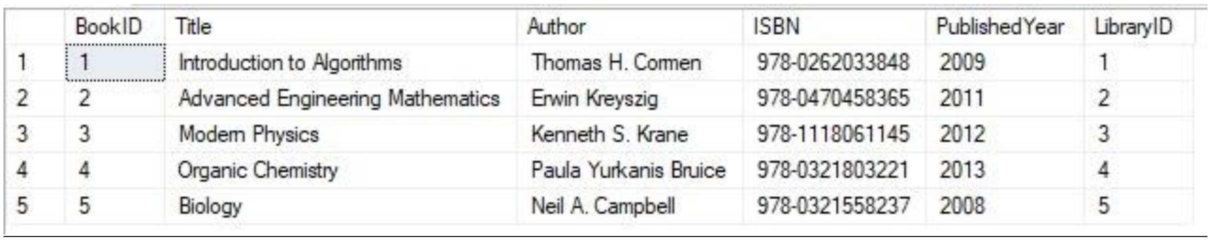
	AttendanceID	StudentID	CourseID	Date	Status
1	1	1	1	2024-01-15	P
2	2	2	2	2024-01-16	A
3	3	3	3	2024-01-17	P
4	4	4	4	2024-01-18	A
5	5	5	5	2024-01-19	P

Enrollment



	EnrollmentID	StudentID	CourseID	EnrollmentDate
1	1	1	1	2024-01-10
2	2	2	2	2024-01-11
3	3	3	3	2024-01-12
4	4	4	4	2024-01-13
5	5	5	5	2024-01-14

Book



	BookID	Title	Author	ISBN	PublishedYear	LibraryID
1	1	Introduction to Algorithms	Thomas H. Cormen	978-0262033848	2009	1
2	2	Advanced Engineering Mathematics	Erwin Kreyszig	978-0470458365	2011	2
3	3	Modern Physics	Kenneth S. Krane	978-1118061145	2012	3
4	4	Organic Chemistry	Paula Yurkanis Bruice	978-0321803221	2013	4
5	5	Biology	Neil A. Campbell	978-0321558237	2008	5

Queries we Applied:

```sql
select distinct LibraryID from Book;

select *from Book where Author='Thomas H. Cormen';

select top 3 *from Department;

select *from Classroom where RoomNumber='101' and Building='Engineering';

select *from Classroom where Capacity between 50 and 70;

select *from Classroom where Capacity Not between 50 and 70;

select *from Department where DepartmentName='%s';

select max(capacity) as max_capacity from Classroom;

select min(DateOfBirth) as DOB from faculty;

select avg(capacity) as avg_capacity from Classroom;

select sum(capacity) as total_capacity from Classroom;

select *from Library order by LibraryName ASC;

select *from Course order by CourseName DESC;

select *from faculty where (FirstName='Alice' AND LastName='Green') OR DateOfBirth='1980-06-15';
```

-----Joins------

```sql
Select d.DepartmentName,s.Address from Department as d inner join Student as s on
d.DepartmentID=s.DepartmentID;
```

```
select *from Library as l full outer join book as b on l.LibraryID=b.LibraryID;

select e.EnrollmentDate,c.CourseName from Enrollment as e right join Course as c on e.CourseID=c.CourseID;

select *from Enrollment as e left join Course as c on e.CourseID=c.CourseID;
```

-----view------

```
create view v1 as select ScheduleID,CourseID,ClassroomID from Schedule;

select *from v1;   --executing a view

drop v1;   ---dropping view

create view v2 as select CourseID, CourseName, Credits, DepartmentID, FacultyID from Course;

select *from v2;
```

-----Procedure-----

-----creating a procedure

```
create proc lstname
as
BEGIN
select *from Book where Author='Thomas H. Cormen';
select *from Faculty where FirstName='Alice';
end
```

------alter a procedure

```
alter proc lstname
```

```
as
BEGIN
select *from Book where Author='Thomas H. Cormen';
select *from Faculty where FirstName='Alice';
end
```

------drop a procedure

```
drop proc lstname;
```

------default parameters in procedure

```
alter proc lstname
@name1 varchar(50)='Thomas H. Cormen',
@name2 varchar(50)='Alice'
as
BEGIN
select *from Book where Author=@name1;
select *from Faculty where FirstName=@name2;

end
```

-----passing parameters to proc

```
alter proc lstname
@name1 varchar(50),
@name2 varchar(50)
as
begin
select *from Book where Author=@name1;
select *from Faculty where FirstName=@name2;
end
```

------procedure with parameters

```
lstname 'Thomas H. Cormen','Alice';
```

-----procedure without parameters

lstname;

----------creating triggers

```
create trigger lib_forinsert
on Library
after insert
as
begin
	print 'table changed';
end
```

---------trigger for diplaying inserted record

```
ALTER trigger lib_forinsert
on Library
after insert
as
begin
	select *from inserted;
end
```

--------trigger for displaying deleted record

```
create trigger lib_fordelete
on Library
after delete
as
begin
```

```
	select *from deleted;
end
```

-------- DDL trigger

```
use project
go
create trigger cr_table
on database
for create_table,alter_table,drop_table
as
begin
	print 'YOU CANNOT CREATE,alter,drop TABLE';
rollback transaction
end

create table hello
(id int)
```

-----display data of all tables

```
select *from Student;
select *from Faculty;
select *from Course;
select *from Department;
select *from Classroom;
select *from Grade;
select *from Assignment;
select *from Schedule;
select *from Library;
select *from Book;
select *from Enrollment;
select *from Attendance;
```