LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

CC5051NI Databases

100% Individual Coursework

Autumn 2024

Credit: 15 Semester Long Module

Student Name: Rijan karki

London Met ID: 23056320

Assignment Submission Date: 25th July,2025

Word Count: 5220

I confirm that I understand my coursework needs to be submitted online via My Second Teacher Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1. Introduction

Little Angels' College (LAC) Little Angels' College (LAC) is a well-known educational institution in Nepal, that was established in 1997. Hattiban, Lalitpur is the location of LAC which caters to students of +2 and Bachelor's levels with technical and professional programs in Science, Management, Computing, and Humanities. The college has gained its reputation for academic excellence, discipline, and dedication to the development of future professionals. LAC is committed to a mission of "providing quality education that fosters intellectual growth, creativity, and leadership." Values like honesty, innovation, and respect for others as well as non-stop development are the main principles of college. It invests a lot of effort in academic and extracurricular training so that students can be successful in life.

Figure 1: Little Angles College Logo

Throughout its history, the LAC has changed to incorporate contemporary infrastructures, electronic devices, and global curriculum collaborations, which have helped them draw students from every corner of the country. LAC has a reputation for being a leader in training highly competent professionals in the fields of Computing, Networking, and Multimedia, supported by the faculty's experience and an innovative academic environment.

I am designing and building a database using SQL PLUS and Oracle XE. So, my motive is to create a system that stores information in an organized way while avoiding unnecessary repetition. To reduce the redundancy we are using normalization, which helps to make the database more efficient and easier to manage. Normalization ensures that the data is clean, consistent, and structured logically. By combining SQL PLUS and Oracle XE I think to build a database that is practical, user-friendly, and reliable for storing and accessing data. This project focuses on creating a well-organized and efficient system that meets all the necessary requirements.
Redundancy means having the same piece of data stored in multiple places unnecessarily, which can lead to confusion, inconsistency, and wasted storage space. (S.Gillis, 2021)

Normalization is a process used in database design to organize data in a way that removes redundancy and ensures it is stored efficiently and logically. This helps to maintain data accuracy and makes the database easier to manage. (Introduction of Database Normalization, 2025)

1.1.Introduction of the Business and its forte

Ms. Mary is planning to launch a digital platform called “E-Classroom Platform” for a college. This system is designed to provide a seamless online learning experience for both students and teachers. The objective is to create a user-friendly environment where everything from managing students and teachers to organizing study programs can be handled efficiently. The platform focuses on streamlining educational activities such as monitoring student progress, organizing modules, managing assessments, and facilitating communication. With its well-thought-out features, the E-Classroom Platform is set to become a vital tool for enhancing education in a structured and modern way.

1.2.Description of Current Business Activities and Operations

Little Angels’ College (LAC) is the place where you can get several undergraduate academic programs at the moment. They are BSc in Computing, BSc in Networking, BSc in Multimedia, and some others that are related to the technical field. The programs of these majors are formed by numerous compulsory modules and some of them are used for different programs. As an illustration, a module such as Programming which can be included both in the BSc in Computing and BSc in Multimedia programs, hence, facilitating the curriculum flexibility.

- Student Enrolment- Students are enrolled in only one program at a time. The college tracks which program a student belongs to but currently lacks a unified system to manage and monitor students’ academic progress digitally.
- Module Structure -Each program includes multiple modules. These modules are the foundation of the academic curriculum and are mandatory for enrolled students. Some modules can exist under multiple programs.
- Teacher Assignment -Teachers are assigned to teach specific modules. A module may have one or more teachers, and the teaching assignments are typically managed by department heads using manual processes.
- Assessments -Each module includes one or more assessments. These assessments include details such as: Assessment ID Title Deadline Weightage Assessments are given to students and linked to the module. The students' work is marked, and results are generated.
- Student Performance and Results- Students receive results for each assessment. The system is required to record: Marks obtained Total marks Component breakdown Remarks or relevant feedback This allows for accurate performance tracking per student and per module.
- Learning Resources -Modules contain essential resources (videos, documents, readings, etc.) required for delivering course content. These resources are meant to be accessed by students in a predefined sequence, ensuring a structured and progressive learning path. Students must mark one resource as completed before accessing the next one.

- Announcements and Notices -Teachers can post announcements for the modules they are assigned to. These announcements are tied to specific modules and can include reminders, updates, or academic notices. Currently, these are managed informally and not through a centralized system.

1.3.Business Rules

Business rules are a group statement that provides organizations about the constraints, certain conditions, and action to be taken for the business. It is used for businesses to manage their business structure. (Business Rules, 2025)

1. Students can enroll in only one program at a time.
2. Each program consists of multiple modules that students must complete.
3. Modules can be shared between different programs.
4. Each module has one or more assessments which are linked directly to it.
5. Teachers are assigned to specific modules, with at least one teacher per module.
6. Teachers can post announcements related to their modules, and each announcement must belong to one specific module.
7. Modules have resources that students must complete in a predefined order.
8. Students assessment results must detail total marks obtained and other relevant Information.

1.4.Assumptions

1. Each student, program, module, teacher, resource, assessment, and announcement have a unique ID for easy identification.
2. Students can view their assessment results and track their progress through the platform.
3. A module can belong to multiple programs, but the content, resources, and assessments are consistent across programs.
4. Teachers can manage multiple modules and their associated announcements, assessments, and resources.
5. The system enforces the order of resource completion for structured learning.
6. All data entries are accurate and updated regularly (e.g., student enrollments, assessment results, and resource completion status).
7. The database will handle future scalability, such as adding more programs, modules, students, or teachers.
8. The system includes basic error-handling to prevent duplication or invalid data entries.

2. Initial Entity Relationship Model

2.1. Identification of Entities and Attributes

1. Student

- Attributes: Student ID, Student Name, Student Email, Student Address

S.No.	Attribute Name	Data Type	size	Constraint
1	Student_ID	Number	10	Primary Key
2	Student_Name	Character	40	Not null
3	Student_Email	Character	30	Unique
4	Student_Address	Character	40	Not Null

Table 1: Identification of the Table Student

2. Program

- Attributes: Program ID, Program Name, Program Details, Program Duration

S.No.	Attribute Name	Data Type	Size	Constraint
1	Program_ID	Number	10	Primary Key
2	Program_Name	Character	40	Not Null
3	Program_Details	Character	40	Not Null
4	Program_Duration	Date		Not Null

Table 2: Identification of the Table Program

3. Module

- Attributes: Module ID, Module Name, Module Details, Module Duration

S.No.	Attribute Name	Data Type	Size	Constraints
1	Module_ID	Number	10	Primary Key
2	Module_Name	Character	40	Not Null
3	Module_Details	Character	40	Not Null
4	Module_Duration	Date		Not Null

Table 3: identification of the Table Module

4. Assessment

- Attributes: Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage

S.No.	Attribute Name	Data Type	Size	Constraints
1	Assessment_ID	Number	10	Primary Key
2	Assessment_Title	Character	40	Not Null
3	Asssessment_Deadline	Date		Not Null
4	Assessment_Deadline	Character	40	Not Null

Table 4:Identification of the Table Assessment

5. Teacher

- Attributes: Teacher ID, Teacher Name, Teacher Number, Teacher Email

S.No.	Attribute Name	Data Type	Size	Constraints
1	Teacher_ID	Number	10	Primary Key
2	Teacher_Name	Character	40	Not Null
3	Teacher_Number	Number	10	Unique
4	Teacher_Email	Character	30	Not Null

Table 5: Identification of the Table Teacher

6. Resource

- Attributes: Resource ID, Resource Title, Resource type, Resource Duration

S.No.	Attribute Name	Data Type	Size	Constraints
1	Resource_ID	Number	10	Primary Key
2	Resource_Title	Character	40	Not Null
3	Resource_Type	Character	40	Not Null
4	Resource_Duration	Date		Not Null

Table 6: Identification of the Table Resource

7. Announcement

- Attributes: Announcement ID, Announcement Type, Announcement Title, Announcement date

S.No.	Attribute Name	Data Type	Size	Constraints
1	Announcement_ID	Number	10	Primary Key
2	Announcement_Type	Character	40	Not Null
3	Announcement_Title	Character	40	Not Null
4	Announcement_Date	Date		Not Null

Table 7: Identification of the Table Announcement

8. Result

- Attributes: Result ID, Result Details, Marks Obtained, Module Grades

S.No.	Attribute Name	Data Type	Size	Constraints
1	Result_ID	Number	10	Primary Key
2	Result_Details	Character	40	Not Null
3	Marks_Obtained	Number	10	Not Null
4	Module_Grades	Character	40	Not Null

Table 8: Identification of the Table Result

2.2.Entity Relation Diagram

Entity Relationship Diagram (ERD) is a simple map that shows how different things in a system are connected. It's a way to visualize how parts of a system, like students, teachers, and courses, all together.

• Entities are the main things we care about, like a Student, Teacher, or Course.

• Attributes are just the details about those things, like a Student's name, ID, or email.

• Relationships shows how these things are connected. For example, a Teacher teaches a Course, or a Student takes a course. (Entity relationship diagram, n.d.)

2.2.1. Relationships

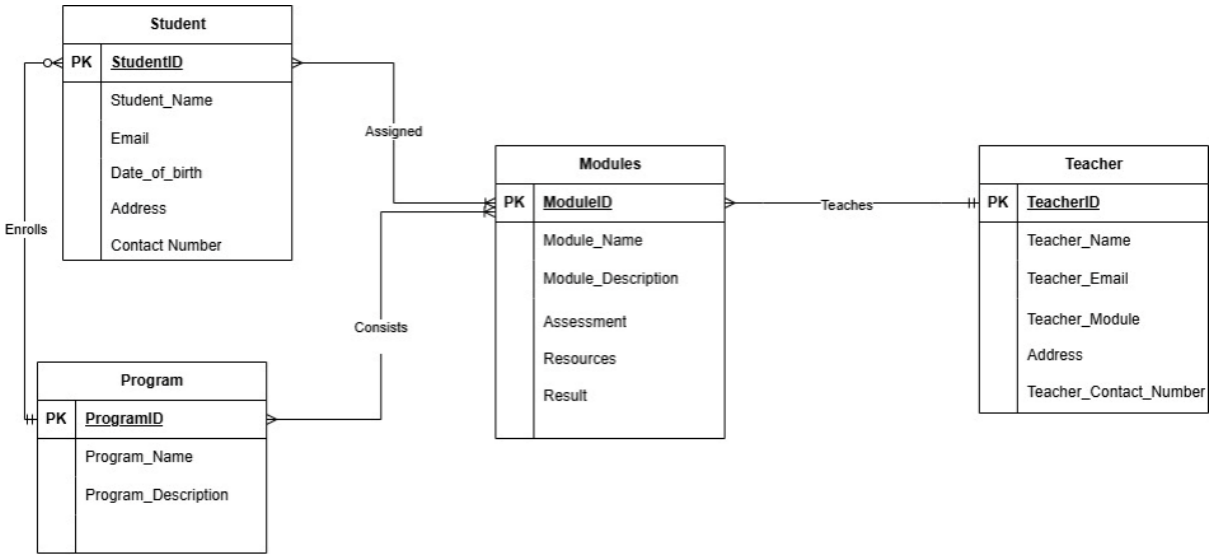

Figure 2: Initial Entity relationship Diagram

3. Normalization

Normalization is the process to simplify the data in a database by organizing the data Normalization is used to reduce excess in a connection or group of relationships. It also removes unnecessary properties such as insertion, update, and deletion anomalies. In normalization, the large tables are broken down into smaller simpler tables and they are connected through relationships.

In simple terms, it's done in steps:

1. UNF (Unnormalized Form)
   UNF is the starting point of data where information is disorganized and not structured. It often contains repeated data and multiple values in a single field, making it hard to work with or query.

2. 1NF (First Normal Form)
   1NF ensures that each column in a table holds only one value (no arrays or lists in a single column) and that each row is unique. It eliminates repeating groups and organizes data into atomic (indivisible) values.

3. 2NF (Second Normal Form)
   2NF builds on 1NF and removes partial dependencies. This means that all non-key attributes must depend on the whole primary key, not just a part of it. If there's a composite primary key, we split the data into separate tables to remove dependencies on part of the key.

4. 3NF (Third Normal Form)
   3NF takes care of transitive dependencies, ensuring that non-key attributes depend only on the primary key. If a non-key attribute depends on another non-key attribute, we separate them into different tables to ensure all data is directly related to primary key. (Introduction of Database Normalization, 2025)

3.1.UNF

Listing down all the attributes:

- Attributes: Student ID, Student Name, Student Email, Student Address, Program ID, Program Name, Program Details, Program Duration, Module ID, Module Name, Module Details, Module Duration, Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage, Teacher ID, Teacher Name, Teacher Address, Resource ID, Resource Title, Resource type, Resource Duration, Announcement ID, Announcement Content, Announcement Posted Date, Result ID, Result Details, Marks Obtained, Comments

  Here, Student ID is the primary key and repeating groups have been embraced with curly bracket.

  Final UNF,
- Student (Student ID, Student Name, Student Email, Student Address, Program ID, Program Name, Program Details, Program Duration {Module ID, Module Name, Module Details, Module Duration, Teacher ID, Teacher Name, Teacher Address {Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage, Result ID, Result Details, Marks Obtained, Comments}, {Resource ID, Resource Title, Resource type, Resource Duration}, {Announcement ID, Announcement Content, Announcement Posted Date}})

3.2.1NF

- Student (Student ID, Student Name, Student Email, Student Address, Program ID, Program Name, Program Details, Program Duration {Module ID, Module Name, Module Details, Module Duration, Teacher ID, Teacher Name, Teacher Address {Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage, Result ID, Result Details, Marks Obtained, Comments}, {Resource ID, Resource Title, Resource type, Resource Duration}, {Announcement ID, Announcement Content, Announcement Posted Date}})

- Student (Student ID, Student Name, Student Email, Student Address, Program ID, Program Name, Program Details, Program Duration)

In the Module table Student ID is carry forward as it is the key identifier

- Module (Student ID, Module ID, Module Name, Module Details, Module Duration, Teacher ID, Teacher Name, Teacher Address {Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage, Result ID, Result Details, Marks Obtained, Comments}, {Resource ID, Resource Title, Resource type, Resource Duration}, {Announcement ID, Announcement Content, Announcement Posted Date})

As the Module is not in 1NF
Again,

- Module (Student ID, Module ID, Module Name, Module Details, Module Duration, Teacher ID, Teacher Name, Teacher Address)

In the Module_list table Student ID and Module ID is carry forward as it is the key identifier

- Module_list (Student ID, Module ID, Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage, Result ID, Result Details, Marks Obtained, Comments, {Resource ID, Resource Title, Resource type, Resource Duration}, {Announcement ID, Announcement Content, Announcement Posted Date})

As the Module_list is not in 1NF
Again,

- Module_list (Student ID, Module ID, Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage, Result ID, Result Details, Marks Obtained, Comments)

In the Resource table Student ID, Module ID and Assessment ID are carry forward as it is the key identifier

- Resource (Student ID, Module ID, Assessment ID, Resource ID, Resource Title, Resource type, Resource Duration, {Announcement ID, Announcement Content, Announcement Posted Date})

As the Resource is not in 1NF
Again,

- Resource (Student ID, Module ID, Assessment ID, Resource ID, Resource Title, Resource type, Resource Duration)

  In the Announcement table Student ID, Module ID, Assessment ID and Resource ID are carry forward as it is the key identifier
- Announcement (Student ID, Module ID, Assessment ID, Resource ID, Announcement ID, Announcement Content, Announcement Posted Date)

Final 1NF,

- Student (Student ID, Student Name, Student Email, Student Address, Program ID, Program Name, Program Details, Program Duration)
- Module (Student ID, Module ID, Module Name, Module Details, Module Duration, Teacher ID, Teacher Name, Teacher Address)
- Module_list (Student ID, Module ID, Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage, Result ID, Result Details, Marks Obtained, Comments)
- Resource (Student ID, Module ID, Assessment ID, Resource ID, Resource Title, Resource type, Resource Duration)
- Announcement (Student ID, Module ID, Assessment ID, Resource ID, Announcement ID, Announcement Content, Announcement Posted Date)

3.3.2NF

1. The primary key is Student ID. All non-key attributes (Student Name, Student Email, Student Address, Program ID, Program Name, Program Details, Program Duration) are fully dependent on Student ID.
   No partial dependencies. The table remains unchanged:
   - Student Table: (Student ID, Student Name, Student Email, Student Address, Program ID, Program Name, Program Details, Program Duration)
2. The primary key is Student ID + Module ID. Attributes like Module Name, Module Details, Module Duration, and Teacher ID are fully dependent on Module ID. However, Teacher Name and Teacher Address depend only on Teacher ID, not the composite key.
   Partial dependency identified: Separate teacher-related attributes.
   - Module Table: (Student ID, Module ID, Module Name, Module Details, Module Duration, Teacher ID)
   - Teacher Table: (Teacher ID, Techer Name, Teacher Address)
3. The primary key is Student ID + Module ID + Assessment ID. Attributes like Assessment Title, Assessment Deadline, and Assessment Weightage are fully dependent on Assessment ID. However, Result ID, Result Details, Marks Obtained, and Comments depend only on Result ID, not the full composite key.
   Partial dependency identified: Separate result-related attributes.
   - Module List Table: (Student ID, Module ID, Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage)
   - Result Table: (Result ID, Result Details, Marks Obtained, Comments)
4. The primary key is Student ID + Module ID + Resource ID. Attributes like Resource Title, Resource Type, and Resource Duration are fully dependent on Resource ID.
   No partial dependencies.
   The table remains unchanged:
   - Resource Table: (Student ID, Module ID, Resource ID, Resource Title, Resource Type, Resource Duration)
5. The primary key is Student ID + Module ID + Announcement ID. Attributes like Announcement Content and Announcement Posted Date are fully dependent on Announcement ID.
   No partial dependencies.
   The table remains unchanged:
   - Announcement Table: (Student ID, Module ID, Announcement ID, Announcement Content, Announcement Posted Date)

Final 2NF,

- Student (Student ID, Student Name, Student Email, Student Address, Program ID, Program Name, Program Details, Program Duration)
- Module Table (Student ID, Module ID, Module Name, Module Details, Module Duration, Teacher ID)
- Teacher (Teacher ID, Teacher Name, Teacher Address)
- Module List (Student ID, Module ID, Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage)
- Result (Result ID, Result Details, Marks Obtained, Comments)
- Resource (Student ID, Module ID, Resource ID, Resource Title, Resource Type, Resource Duration)
- Announcement (Student ID, Module ID, Announcement ID, Announcement Content, Announcement Posted Date)

3.4.3NF

1. The primary key is Student ID. Attributes like Program ID, Program Name, Program Details, and Program Duration are dependent on Program ID, not directly on Student ID.

   Transitive dependency identified.

   - Student Table: (Student ID, Student Name, Student Email, Student Address, Program ID)
   - Program Table: (Program ID, Program Name, Program Details, Program Duration)
2. The primary key is Student ID + Module ID. Attributes like Teacher ID depend on Module ID, and there are no transitive dependencies.

   No changes needed.

   - Module Table: (Student ID, Module ID, Module Name, Module Details, Module Duration, Teacher ID)
3. The primary key is Teacher ID. Attributes Teacher Name and Teacher Address depend directly on the primary key.

   No transitive dependencies.

   The table remains unchanged:

   - Teacher Table: (Teacher ID, Teacher Name, Teacher Address)
4. The primary key is Student ID + Module ID + Assessment ID. Attributes Result ID, Result Details, Marks Obtained, and Comments depend only on Result ID, which is already in the Result Table.

   No transitive dependencies.

   The table remains unchanged:

   - Module List Table: (Student ID, Module ID, Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage)
5. The primary key is Result ID. Attributes Result Details, Marks Obtained, and Comments depend directly on Result ID.

   No transitive dependencies.

   The table remains unchanged:

   - Result Table: (Result ID, Result Details, Marks Obtained, Comments)
6. The primary key is Student ID + Module ID + Resource ID. Attributes Resource Title, Resource Type, and Resource Duration depend directly on Resource ID.

   No transitive dependencies.

   The table remains unchanged:

   - Resource Table: (Student ID, Module ID, Resource ID, Resource Title, Resource Type, Resource Duration)
7. The primary key is Student ID + Module ID + Announcement ID. Attributes Announcement Content and Announcement Posted Date depend directly on Announcement ID.

   No transitive dependencies.

   The table remains unchanged:

   - Announcement Table: (Student ID, Module ID, Announcement ID, Announcement Content, Announcement Posted Date)

Final 3NF,

- Student (Student ID, Student Name, Student Email, Student Address, Program ID)
- Program (Program ID, Program Name, Program Details, Program Duration)
- Module (Student ID, Module ID, Module Name, Module Details, Module Duration, Teacher ID)
- Teacher (Teacher ID, Teacher Name, Teacher Address)
- Module List (Student ID, Module ID, Assessment ID, Assessment Title, Assessment Deadline, Assessment Weightage)
- Result (Result ID, Result Details, Marks Obtained, Comments)
- Resource (Student ID, Module ID, Resource ID, Resource Title, Resource Type, Resource Duration)
- Announcement (Student ID, Module ID, Announcement ID, Announcement Content, Announcement Posted Date)

4. Data Dictionary

A data dictionary can be defined as a component that stores a collection of names, definitions, and attributes for data elements used in the database. The database stores metadata, that is, information about the database. These data elements are then used as part of a database, research project, or information system. (geekforgeeks, 2025)

4.1.List of Entities after Normalization and Attributes

STUDENT

S.No.	Attribute	Data Type	Size	Constraints
1	Student_ID	Number	10	Primary Key
2	Student_Name	Character	40	Not Null
3	Student_Email	Character	30	Unique
4	Student_Address	Character	40	

Table 9: Table Student

Student_Program

S.No.	Attribute Name	Data Type	Size	Constraints	Composite Constraints
1	Student_ID	Number	10	Foreign Key	Primary Key
2	Program_ID	Number	10	Foreign Key	Primary Key

Table 10: Table Student_Program

PROGRAM

S.No.	Attribute Name	Data Type	Size	Constraint	Composite Constraints
1	Program_ID	Number	10	Primary Key	
2	Program_Name	Character	50	Not Null	
3	Program_Description	Character	100	Not Null	

Table 11: Table Program

Student_Program_Module

S.No.	Attribute Name	Data Type	Size	Constraints	Composite Constraint
1	Module_ID	Number	10	Foreign Key	Primary Key
2	Student_ID	Number	10	Foreign Key	Primary Key
3	Program_ID	Number	10	Foreign Key	Primary Key

Table 12: Table Student_Program_Module

MODULE

S.No.	Attribute Name	Data Type	Size	Constraints
1	Module_ID	Number	10	Primary Key
2	Module_Name	Character	40	Not Null
3	Module_Details	Character	40	
4	Module_Duration	Number	3	
5	Program_ID	Number	10	Foreign Key

Table 13: Table Module

Module_Teacher

S.No.	Attribute Name	Data Type	Size	Constraint	Composite Constraint
1	Module_ID	Number	10	Foreign Key	Primary Key
2	Student_ID	Number	10	Foreign Key	Primary Key
3	Program_ID	Number	10	Foreign Key	Primary Key
4	Teacher_ID	Number	10	Foreign Key	Primary Key

Table 14: Table Module_Teacher

TEACHER

S.No.	Attribute Name	Data Type	Size	Constraints
1	Teacher_ID	Number	10	Primary Key
2	Teacher_Name	Character	40	Not Null
3	Teacher_Address	Character	40	

Table 15: Table Teacher

Module_Announcement

S.no.	Attribute Name	Data Type	Size	Constraint	Composite Constraint
1	Module_ID	Number	10	Foreign Key	Primary Key
2	Student_ID	Number	10	Foreign Key	Primary Key
3	Program_ID	Number	10	Foreign Key	Primary Key
4	Announcement_ID	Number	10	Foreign Key	Primary Key

Table 16: Table Module_Announcement

ASSESSMENT

S.No.	Attribute Name	Data Type	Size	Constraints
1	Assessment_ID	Number	10	Primary Key
2	Assessment_Title	Character	40	Not Null
3	Assessment_Deadline	Date		
4	Assessment_Weightage	Number	3	
5	Module_ID	Number	10	Foreign Key

Table 17: Table Assessment

Module_Assessment

S.No.	Attribute Name	Data type	Size	Constraint	Composite Constraint
1	Module_ID	Number	10	Foreign Key	Primary Key
2	Student_ID	Number	10	Foreign Key	Primary Key
3	Program_ID	Number	10	Foreign Key	Primary Key
4	Assessment_ID	Number	10	Foreign Key	Primary Key
5	Result	Number	10	Foreign Key	Primary Key

Table 18: Table Module_Assessment

RESULT

S.No.	Attribute Name	Data Type	Size	Constraints
1	Result_ID	Number	10	Primary Key
2	Result_Details	Character	40	
3	Marks_Obtained	Number		
4	Assessment_ID	Number	10	Foreign Key

Table 19: Table Result

RESOURCE

S.No.	Attribute Name	Data Type	Size	Constraints
1	Resource_ID	Number	10	Primary Key
2	Resource_Title	Character	40	Not Null
3	Resource_Type	Character	30	
4	Resource_Duration	Number	3	
5	Module_ID	Number	10	Foreign Key

Table 20: Table Resource

ANNOUNCEMENT

S.No.	Attribute Name	Data Type	Size	Constraints
1	Announcement_ID	Number	10	Primary Key
2	Announcement_Content	Character	40	
3	Announcement_PostedDate	Date		
4	Module_ID	Number	10	Foreign Key

Table 21: Table Announcement

Module_Resource

S.No.	Attribute Name	Data Type	Size	Constraint	Composite Constraint
1	Module_ID	Number	10	Foreign Key	Primary key
2	Student_ID	Number	10	Foreign Key	Primary key
3	Program_ID	Number	10	Foreign Key	Primary key
4	Resource_ID	Number	10	Foreign Key	Primary key

Table 22: Table Module_Resource

5. Final Entity Relationship Diagram (ERD)

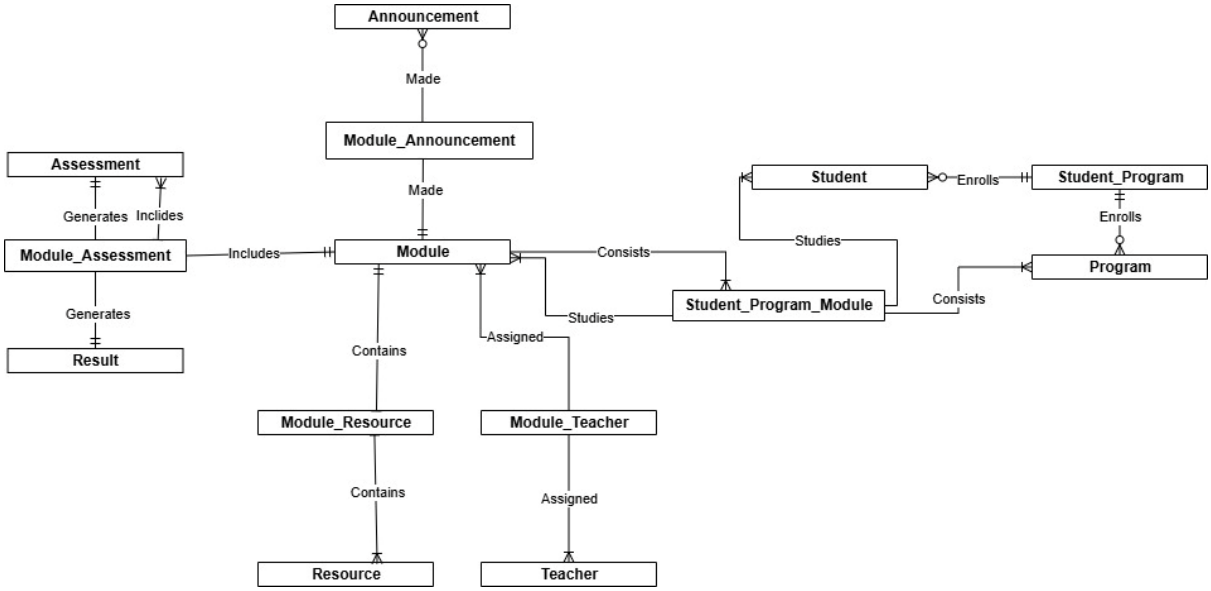

Figure 3: Final Entity Relationship Diagram(ERD)

6. Implementation

6.1. Creating User and Granting privileges:

```
C:\oraclexe\app\oracle\produ

Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production

SQL> CREATE USER rijankarki IDENTIFIED BY 23056320;
CREATE USER rijankarki IDENTIFIED BY 23056320
            *
ERROR at line 1:
ORA-01920: user name 'RIJANKARKI' conflicts with another user or role name


SQL> CREATE USER rijank IDENTIFIED BY 23056320;
CREATE USER rijank IDENTIFIED BY 23056320
            *
ERROR at line 1:
ORA-01920: user name 'RIJANK' conflicts with another user or role name


SQL> CREATE USER rijankrki IDENTIFIED BY 23056320;

User created.

SQL> GRANT CONNECT, RESOURCE TO rijankrki;

Grant succeeded.

SQL> CONNECT rijankrki/23056320
Connected.
SQL>
```

Figure 4: User creation and granting privileges

6.2.Creating Tables

```
SQL> CREATE TABLE STUDENT(
  2  Student_ID NUMBER PRIMARY KEY,
  3  Student_Name VARCHAR(40),
  4  Date_Of_Birth DATE,
  5  Email VARCHAR(30));

Table created.

SQL> DESC STUDENT;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 STUDENT_ID                                NOT NULL NUMBER
 STUDENT_NAME                                       VARCHAR2(40)
 DATE_OF_BIRTH                                      DATE
 EMAIL                                              VARCHAR2(30)

SQL> |
```

Figure 5: Creating Student Table

```
SQL> CREATE TABLE PROGRAM(
  2  Program_ID NUMBER PRIMARY KEY,
  3  Program_Name VARCHAR(50),
  4  Program_Details VARCHAR(255));

Table created.

SQL> DESC PROGRAM;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PROGRAM_ID                                NOT NULL NUMBER
 PROGRAM_NAME                                       VARCHAR2(50)
 PROGRAM_DETAILS                                    VARCHAR2(255)

SQL> |
```

Figure 6: Creating Program table

```
SQL> CREATE TABLE MODULE(
  2  Module_ID NUMBER PRIMARY KEY,
  3  Module_Name VARCHAR(50),
  4  Module_Details VARCHAR(255));

Table created.

SQL> DESC PROGRAM;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PROGRAM_ID                                NOT NULL NUMBER
 PROGRAM_NAME                                       VARCHAR2(50)
 PROGRAM_DETAILS                                    VARCHAR2(255)

SQL> |
```

Figure 7: Creating Module Table

```
SQL> CREATE TABLE TEACHER(
  2  Teacher_ID NUMBER PRIMARY KEY,
  3  Teacher_Name VARCHAR(40),
  4  Teacher_Address VARCHAR(50));

Table created.

SQL> DESC TEACHER;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TEACHER_ID                                NOT NULL NUMBER
 TEACHER_NAME                                       VARCHAR2(40)
 TEACHER_ADDRESS                                    VARCHAR2(50)

SQL> |
```

Figure 8: Creating Teacher Table

```
SQL> CREATE TABLE ASSESSMENT(
  2  Assessment_ID NUMBER PRIMARY KEY,
  3  Assessment_Title VARCHAR(50),
  4  Assessment_Deadline DATE);

Table created.

SQL>
SQL> DESC ASSESSMENT;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ASSESSMENT_ID                             NOT NULL NUMBER
 ASSESSMENT_TITLE                                   VARCHAR2(50)
 ASSESSMENT_DEADLINE                                DATE

SQL> |
```

Figure 9: Creating Assessment Table

```
SQL>
SQL> CREATE TABLE RESOURCES(
  2  Resource_ID NUMBER PRIMARY KEY,
  3  Resource_Title VARCHAR(50),
  4  Resource_Type VARCHAR(50),
  5  Resource_Duration NUMBER);

Table created.

SQL> DESC RESOURCES;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 RESOURCE_ID                               NOT NULL NUMBER
 RESOURCE_TITLE                                     VARCHAR2(50)
 RESOURCE_TYPE                                      VARCHAR2(50)
 RESOURCE_DURATION                                  NUMBER

SQL> |
```

Figure 10: Creating Resources Table

```
SQL> CREATE TABLE ANNOUNCEMENT(
  2  Announcement_ID NUMBER PRIMARY KEY,
  3  Announcement_Content VARCHAR(255),
  4  Announcement_Posted_Date DATE);

Table created.

SQL> DESC ANNOUNCEMENT;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ANNOUNCEMENT_ID                           NOT NULL NUMBER
 ANNOUNCEMENT_CONTENT                               VARCHAR2(255)
 ANNOUNCEMENT_POSTED_DATE                           DATE

SQL> |
```

Figure 11: Creating Announcement Table

```
SQL> CREATE TABLE RESULT(
  2  Result_ID NUMBER PRIMARY KEY,
  3  Result_Details VARCHAR(255),
  4  Marks_Obtained NUMBER,
  5  Comments VARCHAR(255));

Table created.

SQL> DESC RESULT;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 RESULT_ID                                 NOT NULL NUMBER
 RESULT_DETAILS                                     VARCHAR2(255)
 MARKS_OBTAINED                                     NUMBER
 COMMENTS                                           VARCHAR2(255)

SQL> |
```

Figure 12: Creating Result Table

```
SQL> CREATE TABLE Student_Program(
  2  Student_ID NUMBER,
  3  Program_ID NUMBER,
  4  PRIMARY KEY(Student_ID, Program_ID),
  5  FOREIGN KEY (Student_ID) REFERENCES STUDENT(Student_ID),
  6  FOREIGN KEY (Program_ID) REFERENCES PROGRAM(Program_ID));

Table created.

SQL> DESC Student_Program;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 STUDENT_ID                                NOT NULL NUMBER
 PROGRAM_ID                                NOT NULL NUMBER

SQL> |
```

Figure 13: Creating Student_Program Table

```
SQL> CREATE TABLE Student_Program_Module(
  2  Module_ID NUMBER,
  3  Student_ID NUMBER,
  4  Program_ID NUMBER,
  5  PRIMARY KEY(Module_ID, Student_ID, Program_ID),
  6  FOREIGN KEY (Module_ID) REFERENCES MODULE(Module_ID),
  7  FOREIGN KEY (Student_ID) REFERENCES STUDENT(Student_ID),
  8  FOREIGN KEY (Program_ID) REFERENCES PROGRAM(Program_ID));

Table created.

SQL> DESC Student_Program_Module;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MODULE_ID                                 NOT NULL NUMBER
 STUDENT_ID                                NOT NULL NUMBER
 PROGRAM_ID                                NOT NULL NUMBER

SQL> |
```

Figure 14: Creating Student_Program_Module Table

```
SQL> CREATE TABLE Module_Teacher(
  2  Module_ID NUMBER,
  3  Teacher_ID NUMBER,
  4  PRIMARY KEY (Module_ID, Teacher_ID),
  5  FOREIGN KEY(Module_ID) REFERENCES MODULE(Module_ID),
  6  FOREIGN KEY(Teacher_ID) REFERENCES TEACHER(Teacher_ID));

Table created.

SQL> DESC Module_Teacher;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MODULE_ID                                 NOT NULL NUMBER
 TEACHER_ID                                NOT NULL NUMBER

SQL> |
```

Figure 15: Creating Module_teacher Table

```
  5
SQL> CREATE TABLE Module_Announcement(
  2  Module_ID NUMBER,
  3  Announcement_ID NUMBER,
  4  PRIMARY KEY (Module_ID,Announcement_ID),
  5  FOREIGN KEY(Module_ID) REFERENCES MODULE(Module_ID),
  6  FOREIGN KEY(Announcement_ID) REFERENCES ANNOUNCEMENT(Announcement_ID));

Table created.

SQL> DESC Module_Announcement;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MODULE_ID                                 NOT NULL NUMBER
 ANNOUNCEMENT_ID                           NOT NULL NUMBER

SQL> |
```

Figure 16: Creating module_announcement Table

```
SQL> CREATE TABLE Module_Assessment(
  2  Module_ID NUMBER,
  3  Assessment_ID NUMBER,
  4  PRIMARY KEY (Module_ID, Assessment_ID),
  5  FOREIGN KEY(Module_ID) REFERENCES MODULE(Module_ID),
  6  FOREIGN KEY (Assessment_ID) REFERENCES ASSESSMENT(Assessment_ID));

Table created.

SQL> DESC Module_Assessment;]
SP2-0565: Illegal identifier.
SQL> DESC Module_Assessment;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MODULE_ID                                 NOT NULL NUMBER
 ASSESSMENT_ID                             NOT NULL NUMBER

SQL>
```

Figure 17: Creating Module_Assessment Table

```
SQL> CREATE TABLE Module_Resource(
  2  Module_ID NUMBER,
  3  Resource_ID NUMBER,
  4  PRIMARY KEY (Module_ID, Resource_ID),
  5  FOREIGN KEY (Module_ID) REFERENCES MODULE(Module_ID),
  6  FOREIGN KEY(Resource_ID) REFERENCES RESOURCES(Resource_ID));

Table created.

SQL> DESC Module_Resource;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MODULE_ID                                 NOT NULL NUMBER
 RESOURCE_ID                               NOT NULL NUMBER

SQL>
```

Figure 18: Creating Module_Resource Table

```
SQL> CREATE TABLE Assessment_Result(
  2  Assessment_ID NUMBER,
  3  Student_ID NUMBER,
  4  Result_ID NUMBER,
  5  PRIMARY KEY(Assessment_ID, Student_ID),
  6  FOREIGN KEY (Assessment_ID) REFERENCES ASSESSMENT(Assessment_ID),
  7  FOREIGN KEY (Student_ID) REFERENCES STUDENT(Student_ID),
  8  FOREIGN KEY (Result_ID) REFERENCES RESULT(Result_ID));

Table created.

SQL> DESC Assessment_Result;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ASSESSMENT_ID                             NOT NULL NUMBER
 STUDENT_ID                                NOT NULL NUMBER
 RESULT_ID                                          NUMBER

SQL> |
```

Figure 19: Creating Assessment_Result Table

6.3.Inserting And Displaying Data

```
SQL> INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (1, 'John Doe', TO_DATE('2000-01-15', 'YYYY-MM-DD'), 'johndoe@example.com');

1 row created.

SQL> INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (1, 'Ram Shah', TO_DATE('1999-02-22', 'YYYY-MM-DD'), 'ramshah@example.com');
INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (1, 'Ram Shah', TO_DATE('1999-02-22', 'YYYY-MM-DD'), 'ramshah@example.com')
*
ERROR at line 1:
ORA-00001: unique constraint (RIJANKRKI.SYS_C007720) violated


SQL> INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (2, 'Ram Shah', TO_DATE('1999-02-22', 'YYYY-MM-DD'), 'ramshah@example.com');

1 row created.

SQL> INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (3,'Rijan karki', TO_DATE('2004-11-19', 'YYYY-MM-DD), 'rijankarki@example.com');
ERROR:
ORA-01756: quoted string not properly terminated


SQL> INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (3,'Rijan karki', TO_DATE('2004-11-19', 'YYYY-MM-DD'), 'rijankarki@example.com');

1 row created.

SQL> INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (4,'Siddhant Sharma', TO_DATE('2004-08-22', 'YYYY-MM-DD'), 'siddhantsharma@example.com');

1 row created.

SQL> INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (5,'Sarwogya Rana', TO_DATE('2001-02-11', 'YYYY-MM-DD'), 'sarwogyarana@example.com');

1 row created.

SQL> INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (6,'Pratik Shahi', TO_DATE('2000-05-12', 'YYYY-MM-DD'), 'pratikshahi@example.com');

1 row created.

SQL> INSERT INTO STUDENT(Student_ID, Student_Name, Date_Of_Birth, Email) VALUES (7,'Hari kumar', TO_DATE('1998-10-19', 'YYYY-MM-DD'), 'harikumar@example.com');

1 row created.

SQL> |
```

Figure 20: Inserting Data for Student Table

```
SQL> INSERT INTO PROGRAM(Program_ID, Program_Name, Program_Details) VALUES (1, 'Business Administration', 'Bachelors in Computer Science and Technology');

1 row created.

SQL> INSERT INTO PROGRAM(Program_ID, Program_Name, Program_Details) VALUES (2, 'Computer Science', 'Bachelors in Business Administration');

1 row created.

SQL> INSERT INTO PROGRAM(Program_ID, Program_Name, Program_Details) VALUES (3, 'Civil Engineering', 'Bachelors in Civil engineering');

1 row created.

SQL> INSERT INTO PROGRAM(Program_ID, Program_Name, Program_Details) VALUES (4, 'electrical Engineering', 'Bachelors in electrical Engineering');

1 row created.

SQL> INSERT INTO PROGRAM(Program_ID, Program_Name, Program_Details) VALUES (5, 'Mechanical engineering', 'Bachelors in Mechanical engineering');

1 row created.

SQL> INSERT INTO PROGRAM(Program_ID, Program_Name, Program_Details) VALUES (6, 'Agriculture', 'Bachelors in Africulture engineering');

1 row created.

SQL> INSERT INTO PROGRAM(Program_ID, Program_Name, Program_Details) VALUES (7, 'Environment', 'Bachelors in enivironmental Science');

1 row created.

SQL> |
```

Figure 21: Inserting Data for Program Table

```
SQL> INSERT INTO MODULE(Module_ID, Module_Name, Module_Details) VALUES (1, 'Database', 'Core Understanding of Database');
1 row created.
SQL> INSERT INTO MODULE(Module_ID, Module_Name, Module_Details) VALUES (2, 'Digital Systems', 'Core Understanding of Digital systems');
1 row created.
SQL> INSERT INTO MODULE(Module_ID, Module_Name, Module_Details) VALUES (3, 'Data Structuree', 'Core Understanding of Data Structure');
1 row created.
SQL> INSERT INTO MODULE(Module_ID, Module_Name, Module_Details) VALUES (4, 'Discrete Mathematics', 'Core Understanding od discrete mathematics');
1 row created.
SQL> INSERT INTO MODULE(Module_ID, Module_Name, Module_Details) VALUES (5, 'Developmental psychology', 'Core Understanding of Developmental pyschology');
1 row created.
SQL> INSERT INTO MODULE(Module_ID, Module_Name, Module_Details) VALUES (6, 'Dynamics', 'Core Understanding of Dynamics');
1 row created.
SQL> INSERT INTO MODULE(Module_ID, Module_Name, Module_Details) VALUES (7, 'Equations', 'Core Understanding of Equations');
1 row created.
SQL> |
```

Figure 22: Inserting Data for Module Table

```
SQL> INSERT INTO TEACHER(Teacher_ID, Teacher_Name, Teacher_Address) VALUES (1, 'Dr. shyam shah', 'kathmandu');
1 row created.
SQL> INSERT INTO TEACHER(Teacher_ID, Teacher_Name, Teacher_Address) VALUES (2, 'Prof. Laxmi Yadav', 'Janakpur');
1 row created.
SQL> INSERT INTO TEACHER(Teacher_ID, Teacher_Name, Teacher_Address) VALUES (3, 'Prof. Suresh Koirala', 'Bhaktapur');
1 row created.
SQL> INSERT INTO TEACHER(Teacher_ID, Teacher_Name, Teacher_Address) VALUES (4, 'Prof. Anil Singh', 'Punjab');
1 row created.
SQL> INSERT INTO TEACHER(Teacher_ID, Teacher_Name, Teacher_Address) VALUES (5, 'Prof. Ram Panday', 'Butwal');
1 row created.
SQL> INSERT INTO TEACHER(Teacher_ID, Teacher_Name, Teacher_Address) VALUES (6, 'Prof. Nisha Rana', 'Palpa');
1 row created.
SQL> INSERT INTO TEACHER(Teacher_ID, Teacher_Name, Teacher_Address) VALUES (7, 'Prof. meera karki', 'tinthana');
1 row created.
SQL> |
```

Figure 23: Inserting Data for Teacher Table

```
SQL> INSERT INTO ASSESSMENT(Assessment_ID, Assessment_Title, Assessment_Deadline) VALUES (1, 'Quiz 1', TO_DATE('2024-01-05', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ASSESSMENT(Assessment_ID, Assessment_Title, Assessment_Deadline) VALUES (2, 'Quiz 2', TO_DATE('2024-02-05', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ASSESSMENT(Assessment_ID, Assessment_Title, Assessment_Deadline) VALUES (3, 'Project', TO_DATE('2024-03-05', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ASSESSMENT(Assessment_ID, Assessment_Title, Assessment_Deadline) VALUES (4, 'Assignment', TO_DATE('2024-05-05', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ASSESSMENT(Assessment_ID, Assessment_Title, Assessment_Deadline) VALUES (5, 'Final Exam', TO_DATE('2024-06-05', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ASSESSMENT(Assessment_ID, Assessment_Title, Assessment_Deadline) VALUES (6, 'Mid Exam', TO_DATE('2024-03-05', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ASSESSMENT(Assessment_ID, Assessment_Title, Assessment_Deadline) VALUES (7, 'Lab work', TO_DATE('2024-04-05', 'YYYY-MM-DD'));
1 row created.
SQL> |
```

Figure 24: Inserting Data for Assessment Table

```
SQL> INSERT INTO RESOURCES(Resource_ID, Resource_Title, Resource_Type, Resource_Duration) VALUES (1, 'Database Conecpts', 'Video', 1);
1 row created.
SQL> INSERT INTO RESOURCES(Resource_ID, Resource_Title, Resource_Type, Resource_Duration) VALUES (2, 'SQL Tutorial', 'Doucument', 2);
1 row created.
SQL> INSERT INTO RESOURCES(Resource_ID, Resource_Title, Resource_Type, Resource_Duration) VALUES (3, 'Normalization', 'Doucument', 2);
1 row created.
SQL> INSERT INTO RESOURCES(Resource_ID, Resource_Title, Resource_Type, Resource_Duration) VALUES (4, 'ER Diagram', 'Document', 2);
1 row created.
SQL> INSERT INTO RESOURCES(Resource_ID, Resource_Title, Resource_Type, Resource_Duration) VALUES (5, 'Indexing of Databases', 'Document', 1);
1 row created.
SQL> INSERT INTO RESOURCES(Resource_ID, Resource_Title, Resource_Type, Resource_Duration) VALUES (6, 'Transactions', 'Video', 1);
1 row created.
SQL> INSERT INTO RESOURCES(Resource_ID, Resource_Title, Resource_Type, Resource_Duration) VALUES (7, 'Database Security', 'Video', 1);
1 row created.
SQL> |
```

Figure 25: Inserting Data for Resource table

```
SQL> INSERT INTO ANNOUNCEMENT(Announcement_ID, Announcement_Content, Announcement_Posted_Date) VALUES (1, 'Notice', TO_DATE('2024-05-03', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ANNOUNCEMENT(Announcement_ID, Announcement_Content, Announcement_Posted_Date) VALUES (2, 'Event', TO_DATE('2024-05-10', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ANNOUNCEMENT(Announcement_ID, Announcement_Content, Announcement_Posted_Date) VALUES (3, 'Submission', TO_DATE('2024-05-15', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ANNOUNCEMENT(Announcement_ID, Announcement_Content, Announcement_Posted_Date) VALUES (4, 'Presentation', TO_DATE('2024-05-20', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ANNOUNCEMENT(Announcement_ID, Announcement_Content, Announcement_Posted_Date) VALUES (5, 'Lecture', TO_DATE('2024-05-25', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ANNOUNCEMENT(Announcement_ID, Announcement_Content, Announcement_Posted_Date) VALUES (6, 'workshop', TO_DATE('2024-05-30', 'YYYY-MM-DD'));
1 row created.
SQL> INSERT INTO ANNOUNCEMENT(Announcement_ID, Announcement_Content, Announcement_Posted_Date) VALUES (7, 'Class cancelled', TO_DATE('2024-06-02', 'YYYY-MM-DD'));
1 row created.
SQL> |
```

Figure 26: Inserting Data for Announcement Table

```
SQL> INSERT INTO RESULT(Result_ID, Result_Details, Marks_Obtained, Comments) VALUES (1, 'Science', 85, 'A');
1 row created.
SQL> INSERT INTO RESULT(Result_ID, Result_Details, Marks_Obtained, Comments) VALUES (2, 'Computer', 88, 'A+');
1 row created.
SQL> INSERT INTO RESULT(Result_ID, Result_Details, Marks_Obtained, Comments) VALUES (3, 'Math', 90, 'A+');
1 row created.
SQL> INSERT INTO RESULT(Result_ID, Result_Details, Marks_Obtained, Comments) VALUES (4, 'Math', 0, 'E');
1 row created.
SQL> INSERT INTO RESULT(Result_ID, Result_Details, Marks_Obtained, Comments) VALUES (5, 'Grammar', 90, 'A+');
1 row created.
SQL> INSERT INTO RESULT(Result_ID, Result_Details, Marks_Obtained, Comments) VALUES (6, 'Nepali', 90, 'A+');
1 row created.
SQL> INSERT INTO RESULT(Result_ID, Result_Details, Marks_Obtained, Comments) VALUES (7, 'Social', 90, 'A+');
1 row created.
SQL> COMMIT;
Commit complete.
SQL> |
```

Figure 27: Inserting Data for result Table

```
SQL> INSERT INTO Student_Program_Module(Module_ID, Student_ID, Program_ID) VALUES(1, 1, 1);

1 row created.

SQL> INSERT INTO Student_Program_Module(Module_ID, Student_ID, Program_ID) VALUES(2, 1, 1);

1 row created.

SQL> INSERT INTO Student_Program_Module(Module_ID, Student_ID, Program_ID) VALUES(3, 2, 1);

1 row created.

SQL> INSERT INTO Student_Program_Module(Module_ID, Student_ID, Program_ID) VALUES(4, 2, 2);

1 row created.

SQL> INSERT INTO Student_Program_Module(Module_ID, Student_ID, Program_ID) VALUES(5, 3, 2);

1 row created.

SQL> INSERT INTO Student_Program_Module(Module_ID, Student_ID, Program_ID) VALUES(6, 3, 3);

1 row created.

SQL> INSERT INTO Student_Program_Module(Module_ID, Student_ID, Program_ID) VALUES(7, 4, 3);

1 row created.

SQL> |
```

Figure 28: Inserting Data for Student_Program_module Table

```
SQL> INSERT INTO Student_Program(Student_ID, Program_ID)
  2  VALUES(1,1);

1 row created.

SQL> INSERT INTO Student_Program(Student_ID, Program_ID)
  2  VALUES(2,1);

1 row created.

SQL> INSERT INTO Student_Program(Student_ID, Program_ID)
  2  VALUES(3,2);

1 row created.

SQL> INSERT INTO Student_Program(Student_ID, Program_ID)
  2  VALUES(4,3);

1 row created.

SQL> INSERT INTO Student_Program(Student_ID, Program_ID)
  2  VALUES(5,4);

1 row created.

SQL> INSERT INTO Student_Program(Student_ID, Program_ID)
  2  VALUES(6,5);

1 row created.

SQL> INSERT INTO Student_Program(Student_ID, Program_ID)
  2  VALUES(6,5);
INSERT INTO Student_Program(Student_ID, Program_ID)
*
ERROR at line 1:
ORA-00001: unique constraint (RIJANKRKI.SYS_C007728) violated


SQL> INSERT INTO Student_Program(Student_ID, Program_ID)
  2  VALUES(7,5);

1 row created.

SQL> COMMIT;

Commit complete.

SQL> |
```

Figure 29: Inserting Data for Student_Program Table

```
SQL> INSERT INTO Module_Teacher(Module_ID, Teacher_ID) VALUES (1, 1);

1 row created.

SQL> INSERT INTO Module_Teacher(Module_ID, Teacher_ID) VALUES (2, 1);

1 row created.

SQL> INSERT INTO Module_Teacher(Module_ID, Teacher_ID) VALUES (3, 2);

1 row created.

SQL> INSERT INTO Module_Teacher(Module_ID, Teacher_ID) VALUES (4, 3);

1 row created.

SQL> INSERT INTO Module_Teacher(Module_ID, Teacher_ID) VALUES (5, 4);

1 row created.

SQL> INSERT INTO Module_Teacher(Module_ID, Teacher_ID) VALUES (6, 5);

1 row created.

SQL> INSERT INTO Module_Teacher(Module_ID, Teacher_ID) VALUES (7, 6);

1 row created.

SQL> |
```

Figure 30: Inserting Data for Module_Teacher Table

```
SQL> INSERT INTO Module_Announcement(Module_ID, Announcement_ID)VALUES (1, 1);

1 row created.

SQL> INSERT INTO Module_Announcement(Module_ID, Announcement_ID)VALUES (2, 1);

1 row created.

SQL> INSERT INTO Module_Announcement(Module_ID, Announcement_ID)VALUES (3, 2);

1 row created.

SQL> INSERT INTO Module_Announcement(Module_ID, Announcement_ID)VALUES (45, 2);
INSERT INTO Module_Announcement(Module_ID, Announcement_ID)VALUES (45, 2)
*
ERROR at line 1:
ORA-02291: integrity constraint (RIJANKRKI.SYS_C007739) violated - parent key
not found


SQL> INSERT INTO Module_Announcement(Module_ID, Announcement_ID)VALUES (4, 2);

1 row created.

SQL> INSERT INTO Module_Announcement(Module_ID, Announcement_ID)VALUES (5, 2);

1 row created.

SQL> INSERT INTO Module_Announcement(Module_ID, Announcement_ID)VALUES (6, 4);

1 row created.

SQL> INSERT INTO Module_Announcement(Module_ID, Announcement_ID)VALUES (7, 5);

1 row created.

SQL> |
```

Figure 31: Inserting Data for Module_announcement Table

```
SQL> INSERT INTO Module_Assessment(Module_ID, Assessment_ID) VALUES(1, 1);

1 row created.

SQL> INSERT INTO Module_Assessment(Module_ID, Assessment_ID) VALUES(2, 1);

1 row created.

SQL> INSERT INTO Module_Assessment(Module_ID, Assessment_ID) VALUES(3, 1);

1 row created.

SQL> INSERT INTO Module_Assessment(Module_ID, Assessment_ID) VALUES(4, 1);

1 row created.

SQL> INSERT INTO Module_Assessment(Module_ID, Assessment_ID) VALUES(5, 1);

1 row created.

SQL> INSERT INTO Module_Assessment(Module_ID, Assessment_ID) VALUES(6, 1);

1 row created.

SQL> INSERT INTO Module_Assessment(Module_ID, Assessment_ID) VALUES(7, 1);

1 row created.

SQL> |
```

Figure 32: Inserting Data for module_assessment Table

```
SQL> INSERT INTO Module_Resource(Module_ID, Resource_ID) VALUES (1, 1);

1 row created.

SQL> INSERT INTO Module_Resource(Module_ID, Resource_ID) VALUES (1, 1);
INSERT INTO Module_Resource(Module_ID, Resource_ID) VALUES (1, 1)
*
ERROR at line 1:
ORA-00001: unique constraint (RIJANKRKI.SYS_C007744) violated


SQL> INSERT INTO Module_Resource(Module_ID, Resource_ID) VALUES (2, 1);

1 row created.

SQL> INSERT INTO Module_Resource(Module_ID, Resource_ID) VALUES (3, 2);

1 row created.

SQL> INSERT INTO Module_Resource(Module_ID, Resource_ID) VALUES (4, 4);

1 row created.

SQL> INSERT INTO Module_Resource(Module_ID, Resource_ID) VALUES (5, 4);

1 row created.

SQL> INSERT INTO Module_Resource(Module_ID, Resource_ID) VALUES (6, 4);

1 row created.

SQL> INSERT INTO Module_Resource(Module_ID, Resource_ID) VALUES (7, 4);

1 row created.

SQL> |
```

Figure 33: Inserting Data for Module_Resource Table

```
SQL> INSERT INTO Assessment_Result(Assessment_ID, Student_ID, Result_ID) VALUES (1, 1, 1);
1 row created.
SQL> INSERT INTO Assessment_Result(Assessment_ID, Student_ID, Result_ID) VALUES (2, 2, 2);
1 row created.
SQL> INSERT INTO Assessment_Result(Assessment_ID, Student_ID, Result_ID) VALUES (3, 2, 2);
1 row created.
SQL> INSERT INTO Assessment_Result(Assessment_ID, Student_ID, Result_ID) VALUES (4, 3, 2);
1 row created.
SQL> INSERT INTO Assessment_Result(Assessment_ID, Student_ID, Result_ID) VALUES (2, 5, 2);
1 row created.
SQL> INSERT INTO Assessment_Result(Assessment_ID, Student_ID, Result_ID) VALUES (6, 5, 2);
1 row created.
SQL> INSERT INTO Assessment_Result(Assessment_ID, Student_ID, Result_ID) VALUES (7, 5, 2);
1 row created.
SQL> |
```

Figure 34: Inserting Data for Assessment_Result Table

6.4.Information Query

```
SQL> SELECT P.Program_Name AS "Program",
  2  (SELECT COUNT(SP.Student_ID)
  3  From Student_Program SP
  4  WHERE SP.Program_ID =P.Program_ID) AS "Total Students Enrolled"
  5  FROM
  6  PROGRAM P
  7  ORDER BY
  8  "Total Students Enrolled" DESC;

Program                                            Total Students Enrolled
-------------------------------------------------- -----------------------
Mechanical engineering                                                   2
Business Administration                                                  2
Civil Engineering                                                        1
Computer Science                                                         1
electrical Engineering                                                   1
Agriculture                                                              0
Environment                                                              0

7 rows selected.

SQL> |
```

Figure 35: Listing programs that are available in the college and the total number of students enrolled in each

```
SQL> SELECT
  2  A.Announcement_ID, A.Announcement_Content, A.Announcement_Posted_Date
  3  FROM
  4  ANNOUNCEMENT A
  5  JOIN
  6  Module_Announcement MA ON A.Announcement_ID = MA.Announcement_ID
  7  WHERE
  8  MA.Module_ID = 3
  9  AND A.Announcement_Posted_Date BETWEEN TO_DATE('2024-05-01', 'YYYY-MM-DD') AND TO_DATE('2024-05-28', 'YYYY-MM-DD')
 10  ORDER BY
 11  A.Announcement_Posted_Date;

ANNOUNCEMENT_ID
---------------
ANNOUNCEMENT_CONTENT
--------------------------------------------------------------------------------
ANNOUNCEM
---------
         2
Event
10-MAY-24


SQL> |
```

Figure 36: Listing all the annoucement made for a particular module starting from 1st May 2024 to 28th May 2024

```
SQL> SELECT
  2  M.Module_Name AS "Module",
  3  COUNT(R.Resource_ID) AS "Total Resources"
  4  FROM
  5  MODULE M
  6  LEFT JOIN
  7  Module_Resource MR ON M.Module_ID = MR.Module_ID
  8  LEFT JOIN
  9  RESOURCES R ON MR.Resource_ID = R.Resource_ID
 10  WHERE
 11  M.Module_Name LIKE 'D%'
 12  GROUP BY
 13  M.Module_Name
 14  ORDER BY
 15  M.Module_Name;

Module                                             Total Resources
-------------------------------------------------- ---------------
Data Structuree                                                  1
Database                                                         1
Developmental psychology                                         1
Digital Systems                                                  1
Discrete Mathematics                                             1
Dynamics                                                         1

6 rows selected.

SQL> |
```

Figure 37: Listing the names of all modules that begin with the letter 'D', along with the total number of resources upload for those module

.

```
SQL> SELECT
  2  S.Student_Name AS "Student",
  3  P.Program_Name AS "Enrolled Program"
  4  FROM
  5  STUDENT S
  6  JOIN
  7  Student_Program SP ON S.Student_ID = SP.Student_ID
  8  JOIN
  9  PROGRAM P ON SP.Program_ID = P.Program_ID
 10  WHERE
 11  S.Student_ID NOT IN (
 12  SELECT AR.Student_ID
 13  FROM Assessment_Result AR
 14  JOIN Module_Assessment MA ON AR.Assessment_ID = MA.Assessment_ID
 15  WHERE MA.Module_ID = 3)
 16  ORDER BY
 17  S.Student_Name;

Student
----------------------------------------
Enrolled Program
--------------------------------------------------
Hari kumar
Mechanical engineering

Pratik Shahi
Mechanical engineering

Ram Shah
Business Administration


Student
----------------------------------------
Enrolled Program
--------------------------------------------------
Rijan karki
Computer Science

Sarwogya Rana
electrical Engineering

Siddhant Sharma
Civil Engineering


6 rows selected.

SQL> |
```

Figure 38: Listing the names of all students along with their enrolled program who have not submitted any assessment of all particular module

```
SQL> SELECT
  2  T.Teacher_Name AS "Teacher",
  3  COUNT(MT.Module_ID) AS "Number of Modules"
  4  FROM
  5  TEACHER T
  6  JOIN
  7  Module_Teacher MT ON T.Teacher_ID = MT.Teacher_ID
  8  GROUP BY
  9  T.Teacher_Name
 10  HAVING
 11  COUNT(MT.Module_ID) > 1
 12  ORDER BY
 13  T.Teacher_Name;

Teacher                                Number of Modules
-------------------------------------- -----------------
Dr. shyam shah                                         2

SQL> |
```

Figure 39: Listing all the teachers who teach more than one module

6.5.Transaction Query

```
SQL> SELECT
  2  M.Module_Name AS "Module",
  3  A.Assessment_Deadline AS "Latest Deadline"
  4  FROM
  5  MODULE M, Module_Assessment MA, ASSESSMENT A
  6  WHERE
  7  M.Module_ID = MA.Module_ID
  8  AND MA.Assessment_ID = A.Assessment_ID
  9  AND A.Assessment_Deadline = (SELECT MAX(Assessment_Deadline) FROM ASSESSMENT);

no rows selected

SQL> |
```

Figure 40: Identifying the module that has the latest assessment deadline

```
SQL> SELECT *
  2  FROM(
  3  SELECT S.Student_Name, SUM(R.Marks_Obtained) AS Total_Score
  4  FROM
  5  STUDENT S,
  6  Assessment_Result AR,
  7  RESULT R
  8  WHERE
  9  S.Student_ID = AR.Student_ID
 10  AND AR.Result_ID = R.Result_ID
 11  GROUP BY
 12  S.Student_Name
 13  ORDER BY
 14  Total_Score DESC)
 15  WHERE ROWNUM <= 3;

STUDENT_NAME                           TOTAL_SCORE
-------------------------------------- -----------
Sarwogya Rana                                  264
Ram Shah                                       176
Rijan karki                                     88

SQL> |
```

Figure 41: finding the Top three students who have the highest total score across all modules

```
SQL> SELECT P.Program_Name AS "Program",
  2  COUNT(DISTINCT MA.Assessment_ID) AS "Total Assessments",
  3  AVG(R.Marks_Obtained) AS "Average Score"
  4  FROM PROGRAM P
  5  JOIN
  6  Student_Program SP ON P.Program_ID = SP.Program_ID
  7  JOIN
  8  STUDENT S ON SP.Student_ID = S.Student_ID
  9  JOIN
 10  Assessment_Result AR ON S.Student_ID = AR.Student_ID
 11  JOIN
 12  Result R ON AR.Result_ID = R.Result_ID
 13  JOIN
 14  Module_Assessment MA ON AR.Assessment_ID = MA.Assessment_ID
 15  GROUP BY
 16  P.Program_Name
 17  ORDER BY
 18  P.Program_Name;

Program                                            Total Assessments
-------------------------------------------------- -----------------
Average Score
-------------
Business Administration                                          1
          85


SQL> |
```

Figure 42: Finding the total number of assessments for each program and the average score across all

```
SQL> SELECT AVG(R.Marks_Obtained) AS Average_Score
  2  FROM
  3  MODULE M
  4  JOIN
  5  Module_Assessment MA ON M.Module_ID = MA.Module_ID
  6  JOIN
  7  Assessment_Result AR ON MA.Assessment_ID = AR.Assessment_ID
  8  JOIN
  9  RESULT R ON AR.Result_ID = R.Result_ID
 10  WHERE
 11  M.Module_Name = 'Database';

AVERAGE_SCORE
-------------
           85

SQL> |
```

Figure 43:Listing the students who have scored above the average score in the 'Database' module

```
SQL>
SQL> SELECT
  2  S.Student_Name,
  3  R.Marks_Obtained
  4  FROM STUDENT S
  5  JOIN
  6  Assessment_Result AR ON S.Student_ID = AR.Student_ID
  7  JOIN
  8  RESULT R ON AR.Result_ID = R.Result_ID
  9  JOIN
 10  Module_Assessment MA ON AR.Assessment_ID = MA.Assessment_ID
 11  JOIN
 12  MODULE M ON MA.Module_ID = M.Module_ID
 13  WHERE
 14  M.Module_Name = 'Database'
 15  AND R.Marks_Obtained > 81.5
 16  ORDER BY
 17  R.Marks_Obtained DESC;

STUDENT_NAME                           MARKS_OBTAINED
-------------------------------------- --------------
John Doe                                           85

SQL> |
```

Figure 44: Listing the students who have scored the average score in 'Database' module

```
SQL> SELECT
  2  S.Student_Name,
  3  M.Module_Name,
  4  SUM(R.Marks_Obtained) AS Total_Marks,
  5  CASE
  6  WHEN SUM(R.Marks_Obtained) >= 50 THEN 'Pass'
  7  ELSE 'Fail'
  8  END AS Remarks
  9  FROM
 10  STUDENT S
 11  JOIN
 12  Assessment_Result AR ON S.Student_ID = AR.Student_ID
 13  JOIN
 14  RESULT R ON AR.Result_ID = R.Result_ID
 15  JOIN
 16  Module_Assessment MA ON AR.Assessment_ID = MA.Assessment_ID
 17  JOIN
 18  MODULE M ON MA.Module_ID = M.Module_ID
 19  WHERE
 20  M.Module_Name = 'Database'
 21  GROUP BY
 22  S.Student_Name, M.Module_Name
 23  ORDER BY
 24  S.Student_Name;

STUDENT_NAME
--------------------------------------
MODULE_NAME                                        TOTAL_MARKS REMA
---------------------------------------------------- ----------- ----
John Doe
Database                                                    85 Pass


SQL> |
```

Figure 45: Displaying whether a student has passed or failed as remarks as per their total aggregrate

7. Critical Evaluation

The database model described in this text is intended to equip students with the insight regarding the functioning of database systems, its design, implementation, and management. Database is not only valuable as a course module and its potential for using does not reside only in this module. Databases are the basic building blocks, and the knowledge of it is transferable to diverse areas for example in Software Engineering where the designing and management of the database are necessary to be able to develop the software that is both robust and efficient. It is also relevant in many other areas like data science, Information Systems, CyberSecurity, etc. The topics covered by this coursework are from such things as the creation of the objects, application of relationship types, handling of a big dataset, its management, the use of normalization, etc. Consolidating the application of the coursework gave me an opportunity of managing a database as well as a running of the real world scenarios.

8. Dump File

```
C:\Users\Rijan>EXP system/test123 OWNER=rijankarki FILE=C:\backup\rijankarki.dmp LOG= C:\backup\export.log

Export: Release 11.2.0.2.0 - Production on Tue Jul 22 18:27:52 2025

Copyright (c) 1982, 2009, Oracle and/or its affiliates.  All rights reserved.


EXP-00056: ORACLE error 28002 encountered
ORA-28002: the password will expire within 7 days
Connected to: Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set
server uses AL32UTF8 character set (possible charset conversion)

About to export specified users ...
. exporting pre-schema procedural objects and actions
. exporting foreign function library names for user RIJANKARKI
. exporting PUBLIC type synonyms
. exporting private type synonyms
. exporting object type definitions for user RIJANKARKI
About to export RIJANKARKI's objects ...
. exporting database links
. exporting sequence numbers
. exporting cluster definitions
. about to export RIJANKARKI's tables via Conventional Path ...
. . exporting table                      ASSESSMENT          0 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                          MODULE          0 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                         PROGRAM          0 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                          RESULT          0 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                         STUDENT          0 rows exported
EXP-00091: Exporting questionable statistics.
. exporting synonyms
. exporting views
. exporting stored procedures
. exporting operators
. exporting referential integrity constraints
. exporting triggers
. exporting indextypes
. exporting bitmap, functional and extensible indexes
. exporting posttables actions
. exporting materialized views
. exporting snapshot logs
. exporting job queues
. exporting refresh groups and children
. exporting dimensions
. exporting post-schema procedural objects and actions
. exporting statistics
Export terminated successfully with warnings.

C:\Users\Rijan>
```

Figure 46: Dump File

9. Dropping Tables

```
SQL> DROP TABLE Module_Assessment;

Table dropped.

SQL> DROP TABLE Assessment_Result;

Table dropped.

SQL> DROP TABLE Module_Resource;

Table dropped.

SQL> DROP TABLE Module_Assessment;
DROP TABLE Module_Assessment
           *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> DROP TABLE Module_Teacher;

Table dropped.

SQL> DROP TABLE Student_Program_Module;

Table dropped.

SQL> DROP TABLE Student_Program;

Table dropped.

SQL> DROP TABLE RESULT;

Table dropped.

SQL> DROP TABLE RESOURCES;

Table dropped.

SQL> DROP TABLE ASSESSMENT;

Table dropped.

SQL> DROP TABLE ANNOUNCEMENT;
DROP TABLE ANNOUNCEMENT
           *
ERROR at line 1:
ORA-02449: unique/primary keys in table referenced by foreign keys


SQL> DROP TABLE Module_Announcement;

Table dropped.

SQL> DROP TABLE ANNOUNCEMENT;

Table dropped.

SQL> DROP TABLE TEACHER;

Table dropped.

SQL> DROP TABLE MODULE;
```

Figure 47: Dropping Tables

10. References

(2025, Jul 15). Retrieved from geekforgeeks: https://www.geeksforgeeks.org/software-engineering/short-note-on-data-dictionary/

Business Rules. (2025, 02 03). Retrieved from IBM: https://www.ibm.com/docs/en/baw/23.0.x?topic=rules-business

Entity relationship diagram. (n.d.). Retrieved from ATLASSIAN: https://www.atlassian.com/work-management/project-management/entity-relationship-diagram

Introduction of Database Normalization. (2025, Jan 13). Retrieved from GeekforGeeks: https://www.geeksforgeeks.org/dbms/introduction-of-database-normalization/

S.Gillis, A. (2021). *redundant.* techtarget.